EPSBENCH: BENCHMARKING LLM AGENTS FOR EARNINGS FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Quarterly earnings forecasting requires an agent to connect financial evidence to the income-statement line items that produce a company's reported earnings. We introduce EPSBench, a benchmark for studying this ability across 149 firm-quarters from 86 US companies. Agents read a fixed collection of filings, earnings-call transcripts, and news, then estimate company-specific income-statement drivers with cited evidence. A shared accounting calculator converts these estimates into GAAP diluted earnings per share (EPS), providing a common basis for comparing forecast assumptions. We evaluate six models under a common harness, assessing point forecast error relative to date-matched analyst consensus and checking whether their prediction intervals achieve the stated coverage. Three models show no detectable paired difference in point forecast error from consensus, and three have higher error. Across all six models, nominal 80% intervals contain reported EPS in only 37–70% of runs that return an interval, and most misses fall above the upper bound because the forecasts run low. The driver records trace most of the error gap between models to operating-income estimates.

1 INTRODUCTION

Forecasts inform decisions that must be made before their outcomes are known, including investment and business planning (Ball & Ghysels, 2018; Zou et al., 2022). Some forecasting tasks ask only for a direction or a probability (Chen et al., 2022; Karger et al., 2025). Forecasting a reported quantity asks for a magnitude, which an agent must assemble from incomplete and sometimes conflicting evidence. An agent may find a company's sales guidance, recent cost increases, and historical financial statements, but must judge how much those facts move the next quarter's earnings, since higher sales need not imply higher profit if costs rise faster. A useful forecast puts a size on these effects and states what remains uncertain.

Recent benchmarks have made forecasting a substantial part of agent evaluation. ForecastBench elicits probabilities for unresolved events and compares models with human forecasters (Karger et al., 2025). FutureX evaluates search-enabled agents on diverse future events, including numerical financial predictions, while FinDeepForecast generates live corporate and macroeconomic forecasting tasks, including EPS (Zeng et al., 2026; Li et al., 2026). For EPS, both ask the agent for the predicted figure itself, leaving the route to it to whatever the model writes in its trajectory. An analyst reaches EPS from below, estimating revenue, then the costs charged against it, and finally the items beneath them, with each estimate resting on a particular piece of evidence. A benchmark can ask an agent to forecast in the same order, so that those estimates and their evidence are recorded alongside the final number, which lets an error be traced to the stage of the statement where it enters.

We introduce EPSBench, a benchmark that organizes quarterly earnings forecasting around each company's income statement. Earnings provide a recurring numerical target with a reported outcome, and the statement specifies how a company's reported line items combine into GAAP diluted earnings per share (EPS).[1] EPSBench contains 149 firm-quarters from 86 US companies. For each

[1] US companies prepare their filed financial statements under Generally Accepted Accounting Principles (GAAP). Companies and analysts also report adjusted or non-GAAP earnings that exclude selected items. EPSBench evaluates GAAP diluted EPS throughout, so agents must distinguish adjusted figures from the target when using them as evidence.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

firm-quarter, we build a *schema* from historical filings that identifies the quantities to forecast, which we call *drivers*, and the accounting identities relating them. Agents work inside a fixed information environment built from filings, earnings-call transcripts, and news, all restricted to what was published before the results were released. They estimate these quantities, support their estimates with cited evidence, and submit them to a shared accounting calculator. The calculator returns the implied EPS, and the agent reports a prediction interval to express uncertainty about the outcome, as Figure 1 shows. We score the submitted EPS and its interval, and because the arithmetic is fixed, each score stays attached to the line-item estimates and cited evidence that produced it.

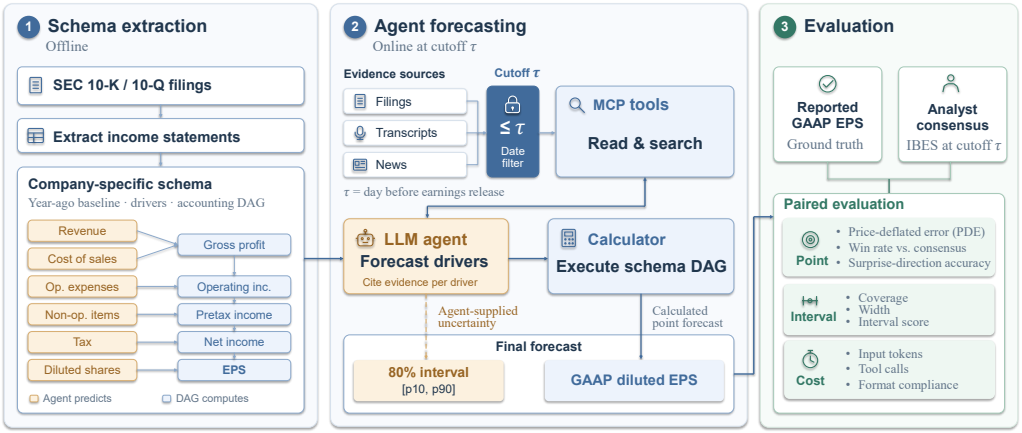

Figure 1: **EPSBench workflow.** Left: historical filings are used to construct company-specific accounting schemas comprising a year-ago baseline, forecast drivers, and a directed acyclic graph of accounting identities. The DAG shown is illustrative; identities vary by firm-quarter. Center: agents use filings, earnings-call transcripts, and news available by the day before the earnings release to estimate drivers with cited evidence. A shared accounting calculator converts these estimates into GAAP diluted EPS, and agents provide an 80% prediction interval. Right: forecasts are evaluated against reported EPS, using date-matched analyst consensus as a reference, alongside measures of interval calibration, resource use, and format compliance.

We evaluate six models from four vendors under a common harness that manages the model's tool calls and context, scoring forecasts against reported EPS and using date-matched analyst consensus as a professional reference. Three models, termed the leading models, show no detectable paired difference in point forecast error from consensus, and the other three have higher error. All six fall short of 80% interval coverage, covering only 37–70% of outcomes among runs that return an interval, and most misses fall above the upper bound consistent with low point forecasts, as consensus forecasts also do. The gap between the two groups opens on quarters whose earnings move far from the year-ago level, where the leading models more often adjust far enough and in the right direction, and it lies almost entirely in the operating lines of the income statement. Sections 5.4 and 5.5 compare information sources, harnesses, and reasoning effort: news and filings carry partly different information, while harness and effort change coverage more than point forecast error.

Our contributions are **(1)** company-specific accounting schemas that define the forecasting task and map driver estimates to EPS; **(2)** a common evaluation protocol linking dated evidence and driver estimates to point and interval scores; and **(3)** an empirical analysis that locates model differences in operating-income forecasts and assesses reported uncertainty. We release the schemas, tools, prompts, forecast records, scores, and trajectories to support reproducible comparisons.

2 Related Work

Forecasting with language model agents. Forecasting benchmarks increasingly evaluate how models use current information to predict unresolved outcomes. Retrieval-based systems combine dated evidence with probabilistic forecasts (Halawi et al., 2024; Zou et al., 2022); ForecastBench continuously elicits forecasts on unresolved questions, while FutureX and FinDeepForecast extend evalua-

tion to search-enabled agents and numerical financial targets (Karger et al., 2025; Zeng et al., 2026; Li et al., 2026). EPSBench focuses on a common representation of an earnings forecast: agents estimate the same company-specific drivers, whose accounting relationships determine EPS. The resulting records support comparisons of quantitative assumptions, while outcome-based metrics assess the point forecast and interval.

Financial prediction and research. Financial benchmarks distinguish forecasting an unreported outcome from reasoning over information already disclosed. FinQA studies numerical reasoning over supplied reports, while FinAgentBench evaluates financial document retrieval (Chen et al., 2021; Choi et al., 2025). Prior earnings work predicts earnings direction or surprises and, in some cases, compares model forecasts with professional analysts (Ou & Penman, 1989; Shu et al., 2026; Chen et al., 2022; Yin & Guo, 2026; An & Lee, 2026). FinanceHarness evaluates financial research reports under controlled model and harness configurations (Xiao et al., 2026). EPSBench instead evaluates future earnings from point-in-time evidence through structured income-statement forecasts that determine GAAP diluted EPS.

Uncertainty and evaluation validity. Forecast quality is not captured by point accuracy alone: useful forecasts must also represent uncertainty, motivating calibration measures and proper scoring rules (Gneiting et al., 2007; Gneiting & Raftery, 2007). Agent evaluation further raises questions of execution cost, reliability, and intermediate progress (Kapoor et al., 2025; Yao et al., 2025; Ma et al., 2024). Retrospective forecasting introduces an additional validity challenge, because apparent performance can be inflated by training exposure, revised webpages, or misdated information (Paleka et al., 2026; Sarkar & Vafa, 2024; Lopez-Lira et al., 2025). EPSBench addresses these concerns jointly by combining point and interval evaluation with execution metrics, tool-enforced point-in-time access, and a memorization test. Appendix A discusses earnings forecasting, financial research agents, and these evaluation issues in more detail.

3 EPSBENCH

Each EPSBench instance asks an agent to forecast a company's GAAP diluted EPS for one quarter, using only documents available the day before the results are released. The agent reads filings, earnings-call transcripts, and news, submits a value for each driver in the company's schema with the evidence behind it, and reports an 80% prediction interval for the EPS that the schema's accounting identities compute. We score the EPS and the interval against the outcome the company reported. Appendix B gives implementation details.

3.1 TASK INSTANCES AND INFORMATION CUTOFF

Each instance pairs a company c with a target fiscal quarter q, which we call a *firm-quarter*. The target is GAAP diluted EPS, the measure reported in the company's filed income statement. We define the release date as $d^{\text{rel}}_{c,q} = \min(d^{\text{call}}_{c,q}, d^{\text{ann}}_{c,q})$, taking the earlier of the earnings-call and official-announcement dates from the Institutional Brokers' Estimate System (IBES). This accounts for companies that publish their results before holding the call. The information cutoff and the date used for analyst consensus are

$$\tau = d^{\text{rel}}_{c,q} - 1 \text{ day}.$$

Because companies announce results weeks after a quarter closes, the target quarter has always ended by τ: the task is to estimate a result that already exists but has not yet been published, as analysts do immediately before an announcement.

We select large, widely followed US companies across eight sectors for which historical filings, a prior-quarter transcript, and an IBES GAAP consensus are available. The benchmark contains 149 firm-quarters from 86 companies with announcements between 2 December 2025 and 24 June 2026, and this set is the common evaluation sample for every model, information-source, and harness comparison. Appendix C provides the full list of companies, and Section 4 describes the separation between the announcements and the models' training cutoffs.

3.2 COMPANY-SPECIFIC ACCOUNTING SCHEMAS

Income statements differ across companies, so no single template of forecast quantities fits them all. A bank reports net interest income, for example, whereas a technology company may report separate product and service revenues. We therefore construct a schema for each firm-quarter from the company's own historical statements. The schema has three parts. The baseline holds the line-item values of the year-ago quarter. The drivers mostly state how a baseline line item changes, as a growth rate for a revenue line or a change in a cost ratio; the rest, such as other expense, tax, and share count, give an amount or rate directly. A directed acyclic graph (DAG) of accounting identities combines the resulting line items into diluted EPS, with the drivers setting its leaves. In Microsoft's fiscal 2026Q3 schema the drivers are product and service revenue growth, the change in cost of revenue as a share of total revenue, operating-expense growth, other expense, income tax expense, and diluted shares. Appendix I gives the complete example.

We build each schema in several model stages. An LLM extractor first retrieves income-statement tables from the filings, and a reasoning model uses the four preceding quarterly statements to construct the DAG around the year-ago quarter, $q-4$. A further model extracts the historical tax and share-count references used as driver defaults. We then verify the schema by executing its DAG on the extracted year-ago values: 146 of 149 reproduce reported EPS within one cent, and the remaining three within four cents. Appendix B.1 details the schema construction and validation.

3.3 AGENT FORECASTING

An agent forecasts from a fixed information environment holding three kinds of source, each covering a different part of the evidence. Annual and quarterly filings (10-K and 10-Q) supply historical statements and detailed disclosures, the prior earnings call usually contains management's outlook for the target quarter, and news covers later developments while sometimes repeating the same guidance. We collect filings from SEC EDGAR, transcripts from Seeking Alpha, and news from CC-News, and expose them through six tools on a Model Context Protocol (MCP) server. The prompt directs agents to the $q-1$ call and the $q-1$ and $q-4$ filings, which provide the recent outlook and the statement underlying the schema; earlier documents remain available on request. Removing either news tools or filing and transcript tools creates the information conditions compared in Section 5.4.

Every source stops at τ. The server blocks the target-quarter filing and call, and news search covers $[\tau_0, \tau]$, where τ_0 is the date of the $q-1$ call. The news tools apply the cutoff to the index shards and to each article they return, rejecting any article whose recorded effective date is later than τ or missing; Appendix B.2 describes how that date is determined and gives the retrieval configuration. Agents cannot read the stored scoring outcome, and the harness configurations in Section 4 block shell, file, and web access. We fix this collection because live search would return pages that postdate the announcement, and a released index lets others re-run the evaluation.

For each driver, the agent writes an *evidence line* identifying its source and explaining the estimate, then submits the driver values to the calculator. The accounting calculator executes the schema and returns the income statement and EPS. The agent keeps that EPS as its point forecast and reports p10 and p90 bounds rather than moving the point to the interval midpoint. Each run is a single attempt without restarts; a final message that cannot be parsed into the required forecast counts as a format failure. Appendix B.3 provides details of the tools and run protocol.

Each run leaves a full trajectory: the agent's messages and tool calls, the documents each call returned, and the final driver values with their evidence lines. Because the calculator performs the arithmetic, the record also holds every intermediate node of the DAG.

3.4 EVALUATION MEASURES

Reported EPS and reference forecasts. We score every forecast against reported GAAP diluted EPS from the target-quarter income statement, verified against that filing. Three reference forecasts give the comparison a scale: year-ago EPS and last-quarter EPS measure simple historical extrapolation, and the analyst consensus, the IBES GAAP EPS consensus at τ, provides a dated professional reference. The consensus comparison is not controlled for information access: analysts have access to sources outside the benchmark, while agents may encounter analyst expectations in retrieved news even though the tools do not return the IBES series. The prompt allows the agent to use an

encountered consensus figure only as a sanity check after it has set its drivers. This mirrors the analyst's own position, since analysts also see one another's estimates; EPSBench instead requires every driver to be supported by cited evidence.

Point forecast. Our primary measure is price-deflated error, $\mathrm{PDE} = |\hat{y} - y|/P$, where $\hat{y}$ is the forecast EPS, y is reported EPS, and P is the closing share price on the last trading day before release. Scaling by price avoids unstable percentage errors when EPS is close to zero and follows common practice in analyst-forecast evaluation (Hong & Kubik, 2003). The win rate is the fraction of instances on which a forecast has lower PDE than consensus, with an unparseable run counted as a loss. Surprise-direction accuracy is the fraction of instances on which the forecast correctly predicts whether reported EPS lands above or below consensus.

Interval forecast. We assess the reported p10–p90 interval by its coverage, the fraction of outcomes it contains, together with the two tail rates, its width, and its interval score. The score adds the width to a penalty for outcomes outside the interval; it is proper, so misreporting p10 or p90 cannot improve the expected score (Gneiting & Raftery, 2007). Like PDE, it is scaled by share price, as defined in Appendix B.4. We elicit the 10th and 90th percentiles, the P10–P90 range commonly used to report forecast uncertainty; with 447 runs per model it leaves about 45 expected misses on each side, enough to estimate the two tail rates separately.

Aggregation and uncertainty. We run each configuration three times per instance. For point forecasts, we take the median PDE over each instance's scoreable runs and report the median across the 149 instances. For interval forecasts, we pool all runs that return an interval across instances, reporting coverage and tail rates as shares of those runs, the median width, and the mean interval score (Gneiting & Raftery, 2007); Table 10 in Appendix G reports these run counts by model. To compare two forecasters, we pair them on each instance, take the difference in their PDEs, and report the median of the 149 differences with a bootstrap 90% confidence interval. Pairing removes the variation in difficulty across instances, which is much larger than the differences between forecasters. The bootstrap resamples instances with replacement 10,000 times and takes the 5th and 95th percentiles of the medians. An interval containing zero means that the data do not resolve the difference at this sample size; it does not establish equivalence. We also record model requests, tool calls, input, output, and reasoning tokens, and format compliance; Appendix B.4 gives their aggregation rules.

4 Experimental Setup

The main experiment compares six models from four vendors, each at its provider's default reasoning effort, on the same 149 tasks, accounting schemas, information environment, and evaluation protocol. Table 6 in Appendix D lists their release dates and published knowledge cutoffs, and we abbreviate them as Opus 4.6, GPT-5.5, GPT-5.4, GPT-5-mini, Qwen 3.5, and GLM 4.5 Air. Additional experiments hold the model fixed while changing the information available to GPT-5.5 or the harness and reasoning effort used by GPT-5.5 and Opus 4.6. We ran all configurations three times per firm-quarter, yielding 447 runs per configuration.

Harnesses and model settings. The primary harness is OpenCode SDK 1.17.9 with shell, file, and web tools disabled, leaving the six MCP tools as the agent's interface. We also test a vanilla harness built on the OpenAI SDK for GPT-5.5, and Claude Code 2.1.269 for Opus 4.6. Section 5.5 additionally runs Opus 4.6 under both harnesses and GPT-5.5 under OpenCode at maximum effort. Appendix E details the harness configurations and isolation controls. All harnesses receive the same prompts, provided in Appendix J, apart from the stated changes required when an information source is removed. Source ablations remove news, filings and transcripts, or both, and update the prompts to name the remaining tools.

Temporal separation from training data. All benchmark announcements follow the published knowledge cutoffs of four models. Of the two that do not disclose a cutoff, GLM 4.5 Air was released before the evaluation window, whereas Qwen 3.5 was released within it. We additionally conduct a memorization test on Qwen 3.5, finding no detectable recall of benchmark outcomes under the tested protocol. Appendix F provides the test details and results.

5 RESULTS

5.1 POINT FORECASTS

Against analyst consensus, the six models fall into two groups, which we call the leading models (Opus 4.6, GPT-5.5, and GPT-5.4) and the trailing models (GPT-5-mini, Qwen 3.5, and GLM 4.5 Air). Table 1 shows that Opus 4.6, GPT-5.5, and GPT-5.4 have median PDEs of 0.113–0.131%, close to the 0.111% of consensus, and each beats consensus on 50% of instances; the other three reach 0.190–0.215% and win on 28–38%. Surprise-direction accuracy follows the same split, from 63% for GPT-5.5 to 45% for GLM 4.5 Air. All six improve on copying year-ago or last-quarter EPS (0.238% and 0.226%). Figure 2 tests these differences with paired comparisons. The intervals for Opus 4.6, GPT-5.5, and GPT-5.4 all include zero, and Opus 4.6 has the lowest median paired difference, -0.002 percentage points (pp) of PDE with a 90% interval of $[-0.011, +0.006]$. The other three have higher paired error than consensus.

Table 1: **Main results.** Six models and three reference forecasters on 149 firm-quarters under OpenCode with the full information environment. PDE is the median price-deflated error; Win is the share of instances with lower PDE than analyst consensus; Surprise is surprise-direction accuracy. Interval metrics pool runs returning a p10–p90 interval; width and score are scaled by share price, in percentage points (pp). Shading marks the leading models; bold marks each column's best model value, with ties, except width, which has no single best value. Section 3.4 details the aggregation.

	Point forecast			80% interval forecast		
Forecaster	PDE (%) ↓	Win (%) ↑	Surprise (%) ↑	Coverage (%)	Median Width (pp)	Mean interval Score (pp) ↓
Analyst consensus	0.111	–	–	–	–	–
Year-ago EPS	0.238	21	36	–	–	–
Last-quarter EPS	0.226	27	51	–	–	–
Opus 4.6	**0.113**	**50**	62	**70**	0.41	**2.87**
GPT-5.5	0.131	**50**	**63**	61	0.33	3.08
GPT-5.4	0.126	**50**	60	52	0.25	4.29
GPT-5-mini	0.191	38	51	50	0.39	19.61
Qwen 3.5	0.190	32	47	37	0.26	4.87
GLM 4.5 Air	0.215	28	45	44	0.33	6.48

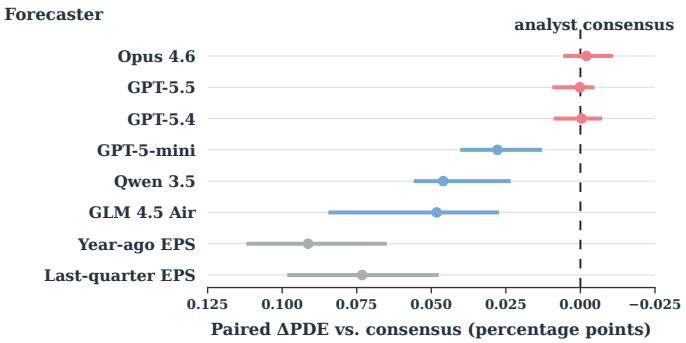

Figure 2: **Paired point forecast error comparisons with analyst consensus.** Dots show the median paired difference in PDE between each forecaster and analyst consensus on 149 firm-quarters; horizontal lines show 90% bootstrap confidence intervals. Gray rows are the two historical reference forecasts. Negative values, to the right of the dashed line, indicate lower error than consensus.

The pairwise comparisons in Table 13 support this separation. Each leading model has lower point forecast error than each trailing model, while no two leading models differ significantly from each other. The trailing models also have more dispersed errors: their interquartile ranges are 0.34–0.44

pp, against 0.23–0.26 pp for the leading models, and Table 11 shows that each trailing model has a PDE above 1% on 13–23 instances, compared with 8–11 instances for each leading model.

5.2 Interval Forecasts

An 80% prediction interval should contain reported EPS in 80% of outcomes, with 10% falling on each side. Table 1 shows that no model reaches this coverage: it ranges from 37% for Qwen 3.5 to 70% for Opus 4.6. Figure 3 shows that the misses are lopsided. For every model, reported EPS lands above the upper bound in 20–38% of outcomes, well over the nominal 10%, while below the lower bound Opus 4.6 and GPT-5.5 stay at 10% and the other four reach 16–25%.

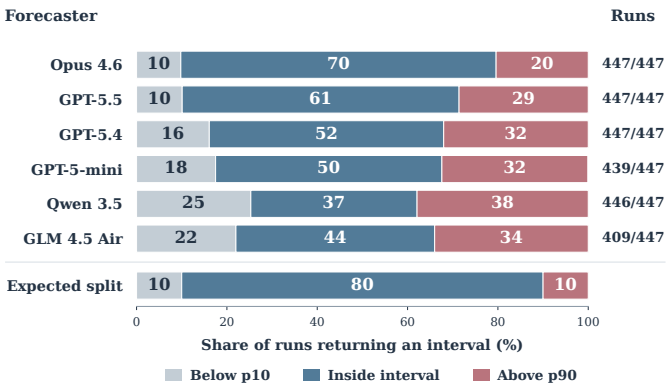

Figure 3: **Prediction interval coverage and tail rates.** Stacked bars show the percentages of reported EPS outcomes below, inside, and above each model's p10–p90 prediction interval, pooling runs that return an interval. The Runs column lists the number of returned intervals over total runs. The expected split is 10% below, 80% inside, and 10% above the interval.

The misses lean upward because the point forecasts themselves tend to be too low. Table 10 in Appendix G reports the median signed error $(y - \hat{y})/P$, which is positive when a forecast falls below reported EPS: it ranges from $+0.021$ to $+0.048$ pp across models, and its confidence interval excludes zero for every model. Each interval is nearly symmetric around its point forecast, with midpoint displacements within 0.004 pp, so a low forecast pulls the whole interval down with it. Consensus is also too low, at $+0.028$ pp with a confidence interval of $[+0.008, +0.051]$. Prior work documents the walk-down of analyst forecasts to beatable targets (Richardson et al., 2004) and the rewards to meeting or beating expectations and their rise over time (Bartov et al., 2002; Brown, 2001). Expectation management and exposure to analyst expectations could both produce the agents' bias.

Table 1 also shows that coverage alone does not rank the intervals. GPT-5.4 and GPT-5.5 have similar point forecast errors, but GPT-5.4's intervals are narrower (median width 0.25 against 0.33 pp), cover fewer outcomes (52% against 61%), and receive a worse mean interval score (4.29 against 3.08). GPT-5-mini shows the opposite case: its coverage of 50% is close to GPT-5.4's, yet its mean score of 19.61 is three to seven times that of every other model (2.87–6.48). A few of its intervals are narrow and far from reported EPS, and because the score charges ten times the distance of each miss, those runs dominate the mean.

5.3 Analysis of Forecast Differences

Model differences are largest on quarters where earnings move far from their year-ago level. For each instance, we measure how much reported EPS changed from the year-ago quarter, scaled by share price, and compare the third of instances with the smallest changes (near) against the third with the largest (far). On near quarters last year's EPS is already a good forecast, so the question is whether a forecaster can anticipate a large change. Table 2 shows that on the 49 near instances, model median PDEs range from 0.048% to 0.082%, compared with 0.036% for consensus. On the 50 far instances, Opus 4.6, GPT-5.5, and GPT-5.4 reach 0.392–0.398%, compared with 0.599–0.716% for the other three models and 0.409% for consensus.

Table 2: **Forecast performance by departure from year-ago EPS.** Near and far groups are the thirds of instances with the smallest and largest price-scaled EPS changes from the year-ago quarter, defined from reported outcomes; the year-ago EPS row gives the median change. Median PDE is aggregated as in Section 3.4. Numeric evidence is the share of far-instance runs that cite a number for their most-adjusted driver. Error halved counts far instances whose median error across the three runs is at most half that of year-ago EPS, and Wrong direction counts far instances where at least two of the three runs move away from year-ago EPS opposite to the realized change.

	Median PDE (%)		Far instances only ($n = 50$)		
Forecaster	Near ($n = 49$)	Far ($n = 50$)	Numeric evidence (%)	Error halved (n)	Wrong direction (n)
Year-ago EPS	0.080	0.975	–	–	–
Analyst consensus	0.036	0.409	–	–	–
Opus 4.6	0.048	0.396	95	30	5
GPT-5.5	0.053	0.392	96	30	6
GPT-5.4	0.050	0.398	95	29	5
GPT-5-mini	0.082	0.614	62	19	12
Qwen 3.5	0.071	0.599	72	17	12
GLM 4.5 Air	0.074	0.716	35	12	12

Table 2 also shows how the two groups handle the far instances. The leading models more often move their forecasts far enough from last year's EPS to cut its error at least in half, on 29–30 of the 50 instances against 12–19 for the trailing models, and they move in the wrong direction less often, on 5–6 instances against 12. To see what supports these adjustments, we read the evidence line for the driver that each run adjusts most, since a large departure from last year needs a concrete basis such as management guidance or a disclosed transaction amount. The leading models cite a number there in 95–96% of runs, against 35–72% for the trailing models; Appendix G gives the coding rules.

The records also show where in the income statement the two groups' errors differ. On the 125 instances with an operating-income subtotal, the gap lies almost entirely in operating income: median operating error is 0.089–0.098 pp for the leading models versus 0.149–0.166 pp for the trailing models, while the non-operating increments are similar at 0.007–0.020 pp. Appendix G reports the full decomposition in Table 14. The records can further trace these errors to individual drivers: in Boeing's fourth quarter of 2025, much of the gap comes from the gain on a divestiture that Boeing reports within operating income. Appendix H gives this case in detail.

5.4 Contributions of Filings, Transcripts, and News

We test how much GPT-5.5's error depends on news and on filings and transcripts. Table 3 shows that, with the schema and accounting calculator held fixed, we detect no paired difference when only news is kept: the median change is -0.000 pp with a 90% interval of $[-0.010, +0.009]$, while input use falls to about one quarter. Removing news and keeping filings and transcripts makes forecasts slightly worse, raising error by $+0.009$ pp $[+0.002, +0.016]$. Removing both makes them substantially worse: error rises by $+0.033$ pp $[+0.019, +0.047]$ and the win rate against consensus falls from 50% to 31%.

To see which instances drive this result, we count an error increase as harmful when it exceeds 0.057 pp, the median spread of PDE across an instance's three runs. Table 12 in Appendix G sorts the 149 instances into three groups. On 67, no removal is harmful. On 32, neither single removal is harmful but removing both is. The remaining 50 depend on a particular source: 17 are hurt only by removing filings and transcripts, 24 only by removing news, and 9 by either removal. Dependence on the two sources is uncorrelated: their error increases have a rank correlation of -0.01.

On the 32 instances where only removing both is harmful, the sources carry overlapping information: management's guidance appears in the earnings call and reappears in news and earnings previews, and in inspected trajectories agents choose similar driver values while citing the transcript in one condition and an article in another. The 50 instances that depend on a particular source show that

Table 3: **Information-source ablation** for GPT-5.5 under OpenCode on 149 firm-quarters. Full includes filings, earnings-call transcripts, and news; PDE and Win are as in Table 1. Paired ΔPDE is relative to Full, and positive values indicate higher error. Tools and Input are per-run medians, with Input including cached tokens.

Information	PDE (%) ↓	Win (%) ↑	Paired ΔPDE, pp vs. full (90% CI)	Tools	Input (K)
Full	0.131	50	–	47	837
News only	0.141	40	-0.000 $[-0.010, +0.009]$	28	196
Filings + transcripts	0.138	42	$+0.009$ $[+0.002, +0.016]$	25	376
Neither	0.210	31	$+0.033$ $[+0.019, +0.047]$	1	8

each source also carries information the other lacks: news supplies post-filing developments and current market prices, whereas filings supply detailed accounting quantities such as the book value of a divested business or the magnitude of an equity-method loss. Removing either source alone therefore barely moves the median error, yet for a third of the instances a particular source cannot be replaced.

Table 4: **Harness and reasoning-effort comparisons** on 149 firm-quarters. Paired ΔPDE is relative to each model's first row, OpenCode at the default effort, and coverage pools runs returning an interval. Tools, Input, and Output are per-run medians; Input includes cached tokens and Output includes reasoning tokens.

Model	Harness	Effort	PDE (%)↓	Paired ΔPDE, pp (90% CI)	Coverage (%)	Tools	Input (K)	Output (K)
Opus 4.6	OpenCode	default	0.113	–	70	35	1056	9.7
	Claude Code	default	0.117	$+0.001$ $[-0.005, +0.004]$	65	34	1029	36.4
	OpenCode	max	0.106	-0.002 $[-0.006, +0.001]$	65	34	1011	40.5
	Claude Code	max	0.114	-0.004 $[-0.008, +0.002]$	65	34	1032	40.4
GPT-5.5	OpenCode	default	0.131	–	61	47	837	6.2
	Vanilla	default	0.110	-0.003 $[-0.007, +0.002]$	54	38	561	8.6
	OpenCode	max	0.135	-0.000 $[-0.003, +0.003]$	54	48	821	12.6

5.5 Sensitivity to the Agent Harness

Finally, we ask whether these results depend on the agent harness and reasoning effort. Table 4 shows that changing either barely moves point forecast error: relative to OpenCode at the default effort, median paired PDE changes by -0.004 to $+0.001$ pp, and all five intervals include zero. Resource use, by contrast, changes considerably. Maximum effort raises median output tokens per run, including reasoning, from 9.7K to 40.5K for Opus 4.6 and from 6.2K to 12.6K for GPT-5.5 under OpenCode. Claude Code already produces 36.4K at the default effort, close to its maximum-effort level, and for GPT-5.5 the vanilla harness uses 33% fewer input tokens. Interval coverage also varies despite similar point forecast error: for Opus 4.6, it is 70% under OpenCode at the default effort and 65% in the other three configurations, while for GPT-5.5 it ranges from 54% to 61%.

6 Conclusion

EPSBench provides an accounting task for studying how agents turn financial evidence into earnings forecasts. The leading agents show no detectable difference in point forecast error from date-matched analyst consensus, yet all six produce intervals with coverage below 80%; more outcomes fall above the upper bound than below the lower bound. Model differences are largest when earnings move sharply from the year-ago level, with most of the gap concentrated at the operating-income stage. Information access affects point forecast error and resource use, while harness and reasoning effort change resource use and interval coverage more than point forecast error.

AI USE STATEMENT

We used generative AI in schema extraction, benchmark implementation, and manuscript drafting and revision. The evaluated systems are themselves language model agents, and all reported experimental results come from recorded runs and evaluation scripts. The authors retain responsibility for validating the implementation and ensuring that the manuscript accurately represents the underlying results.

ETHICS STATEMENT

This study evaluates company financial information and model outputs without recruiting human participants. We obtain public filings from the SEC, licensed earnings-call transcripts from Seeking Alpha, news from CC-News, analyst consensus through an academic IBES/WRDS license, and share prices from Yahoo Finance. We use the IBES records for evaluation and report the forecasts as research outputs; this benchmark does not establish their suitability for investment decisions. The authors declare no conflict of interest.

REPRODUCIBILITY STATEMENT

Sections 3.1–3.4 define the task and evaluation protocol, Section 4 specifies the model and harness configurations, and Appendix F describes the recall probes. We release the extracted schemas, allowing others to reproduce the evaluation without regenerating them. The original schema-generation model, `deepseek-reasoner`, which we used in June and August 2026, is no longer served; extending the benchmark to new quarters will require another extractor.

The release, available at `https://anonymous.4open.science/r/epsbench-7388`, includes the tool server, harness adapters, prompts, evaluation scripts, share prices, per-instance forecasts, reported EPS, and scores. We provide trajectories with transcript text replaced by document identifiers to respect licensing restrictions, along with fetch scripts for public filings and the filtered CC-News index. We do not release IBES consensus values: we report aggregate comparisons and, for each instance, whether the agent outperformed consensus.

REFERENCES

Hyuk An and Seohyun Lee. Simulating Analyst Forecast Behavior using LLMs, jun 2026. URL https://papers.ssrn.com/abstract=6940898. KDI School of Public Policy and Management Paper No. WP26-14.

Ryan T. Ball and Eric Ghysels. Automated Earnings Forecasts: Beat Analysts or Combine and Conquer? *Management Science*, 64(10):4936–4952, 2018. URL https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2972185.

Eli Bartov, Dan Givoly, and Carla Hayn. The rewards to meeting or beating earnings expectations. *Journal of Accounting and Economics*, 33(2):173–204, 2002. doi: 10.1016/S0165-4101(02)00045-9. URL https://doi.org/10.1016/S0165-4101(02)00045-9.

Lawrence D. Brown. A temporal analysis of earnings surprises: Profits versus losses. *Journal of Accounting Research*, 39(2):221–241, 2001. doi: 10.1111/1475-679X.00010. URL https://doi.org/10.1111/1475-679X.00010.

Lawrence D. Brown and Michael S. Rozeff. Univariate Time-Series Models of Quarterly Accounting Earnings per Share: A Proposed Model. *Journal of Accounting Research*, 17(1):179–189, 1979. doi: 10.2307/2490312. URL https://www.jstor.org/stable/2490312.

Lawrence D. Brown, Robert L. Hagerman, Paul A. Griffin, and Mark E. Zmijewski. Security analyst superiority relative to univariate time-series models in forecasting quarterly earnings. *Journal of Accounting and Economics*, 9(1):61–87, apr 1987. doi: 10.1016/0165-4101(87)90017-6. URL https://www.sciencedirect.com/science/article/pii/0165410187900176.

Xi Chen, Yang Ha (Tony) Cho, Yiwei Dou, and Baruch Lev. Predicting Future Earnings Changes Using Machine Learning and Detailed Financial Data. *Journal of Accounting Research*, 60(2): 467–515, may 2022. doi: 10.1111/1475-679X.12429. URL https://onlinelibrary.wiley.com/doi/10.1111/1475-679X.12429.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang. FinQA: A Dataset of Numerical Reasoning over Financial Data. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih (eds.), *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 3697–3711, Online and Punta Cana, Dominican Republic, nov 2021. doi: 10.18653/v1/2021.emnlp-main.300. URL https://aclanthology.org/2021.emnlp-main.300/.

Chanyeol Choi, Jihoon Kwon, Alejandro Lopez-Lira, Chaewoon Kim, Minjae Kim, Juneha Hwang, Jaeseon Ha, Hojun Choi, Suyeol Yun, Yongjin Kim, and Yongjae Lee. FinAgentBench: A Benchmark Dataset for Agentic Retrieval in Financial Question Answering. In *Proceedings of the 6th ACM International Conference on AI in Finance (ICAIF)*. ACM, 2025. doi: 10.1145/3768292.3770362.

Lars Elend, Sebastian A. Tideman, Kerstin Lopatta, and Oliver Kramer. Earnings Prediction with Deep Learning. In *KI 2020: Advances in Artificial Intelligence*, Lecture Notes in Computer Science, pp. 267–274. Springer, 2020. doi: 10.1007/978-3-030-58285-2_22.

George Foster. Quarterly Accounting Data: Time-Series Properties and Predictive-Ability Results. *The Accounting Review*, 52(1):1–21, jan 1977. doi: 10.2308/TAR-4493420. URL https://publications.aaahq.org/accounting-review/article/52/1/1/16459/Quarterly-Accounting-Data-Time-Series-Properties.

Tilmann Gneiting and Adrian E Raftery. Strictly Proper Scoring Rules, Prediction, and Estimation. *Journal of the American Statistical Association*, 102(477):359–378, mar 2007. doi: 10.1198/016214506000001437. URL https://doi.org/10.1198/016214506000001437.

Tilmann Gneiting, Fadoua Balabdaoui, and Adrian E. Raftery. Probabilistic Forecasts, Calibration and Sharpness. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 69(2): 243–268, apr 2007. doi: 10.1111/j.1467-9868.2007.00587.x. URL https://doi.org/10.1111/j.1467-9868.2007.00587.x.

Danny Halawi, Fred Zhang, Yueh-Han Chen, and Jacob Steinhardt. Approaching Human-Level Forecasting with Language Models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 50426–50468, 2024. doi: 10.52202/079017-1598. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/5a5acfd0876c940d81619c1dc60e7748-Abstract-Conference.html.

Harrison Hong and Jeffrey D. Kubik. Analyzing the analysts: Career concerns and biased earnings forecasts. *The Journal of Finance*, 58(1):313–351, 2003. doi: 10.1111/1540-6261.00526. URL https://doi.org/10.1111/1540-6261.00526.

Sayash Kapoor, Benedikt Stroebl, Zachary S. Siegel, Nitya Nadgir, and Arvind Narayanan. AI Agents That Matter. *Transactions on Machine Learning Research*, 2025.

Sayash Kapoor, Benedikt Stroebl, Peter Kirgis, Nitya Nadgir, Zachary S. Siegel, Boyi Wei, Tianci Xue, Ziru Chen, Felix Chen, Saiteja Utpala, Franck Ndzomga, Dheeraj Oruganty, Sophie Luskin, Kangheng Liu, Botao Yu, Amit Arora, Dongyoon Hahm, Harsh Trivedi, Huan Sun, Juyong Lee, Tengjun Jin, Yifan Mai, Yifei Zhou, Yuxuan Zhu, Rishi Bommasani, Daniel Kang, Dawn Song, Peter Henderson, Yu Su, Percy Liang, and Arvind Narayanan. Holistic Agent Leaderboard: The Missing Infrastructure for AI Agent Evaluation. In *International Conference on Learning Representations*, 2026.

Ezra Karger, Houtan Bastani, Yueh-Han Chen, Zachary Jacobs, Danny Halawi, Fred Zhang, and Philip E. Tetlock. ForecastBench: A Dynamic Benchmark of AI Forecasting Capabilities. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=lfPkGWXLLf.

Xiangyu Li, Xuan Yao, Guohao Qi, Fengbin Zhu, Kelvin J. L. Koa, Xiang Yao Ng, Ziyang Liu, Xingyu Ni, Chang Liu, Yonghui Yang, Yang Zhang, Wenjie Wang, Fuli Feng, Chao Wang, Huanbo Luan, Xiaofen Xing, Xiangmin Xu, Tat-Seng Chua, and Ke-Wei Huang. FinDeep-Forecast: A Live Multi-Agent System for Benchmarking Deep Research Agents in Financial Forecasting, jan 2026. URL https://arxiv.org/abs/2601.05039.

Xiao Liu, Hao Yu, Hanchen Zhang, Yifan Xu, Xuanyu Lei, Hanyu Lai, Yu Gu, Hangliang Ding, Kaiwen Men, Kejuan Yang, Shudan Zhang, Xiang Deng, Aohan Zeng, Zhengxiao Du, Chenhui Zhang, Sheng Shen, Tianjun Zhang, Yu Su, Huan Sun, Minlie Huang, Yuxiao Dong, and Jie Tang. AgentBench: Evaluating LLMs as Agents. In *International Conference on Learning Representations*, 2024.

Alejandro Lopez-Lira and Yuehua Tang. Can ChatGPT Forecast Stock Price Movements? Return Predictability and Large Language Models. *Journal of Financial Economics*, 184:104335, 2026.

Alejandro Lopez-Lira, Yuehua Tang, and Mingyin Zhu. The Memorization Problem: Can We Trust LLMs' Economic Forecasts?, 2025. URL https://arxiv.org/abs/2504.14765.

Qing Lyu, Shreya Havaldar, Adam Stein, Li Zhang, Delip Rao, Eric Wong, Marianna Apidianaki, and Chris Callison-Burch. Faithful Chain-of-Thought Reasoning. In Jong C. Park, Yuki Arase, Baotian Hu, Wei Lu, Derry Wijaya, Ayu Purwarianti, and Adila Alfa Krisnadhi (eds.), *Proceedings of the 13th International Joint Conference on Natural Language Processing and the 3rd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 305–329, Nusa Dua, Bali, nov 2023. doi: 10.18653/v1/2023.ijcnlp-main.20. URL https://aclanthology.org/2023.ijcnlp-main.20/.

Chang Ma, Junlei Zhang, Zhihao Zhu, Cheng Yang, Yujiu Yang, Yaohui Jin, Zhenzhong Lan, Lingpeng Kong, and Junxian He. AgentBoard: An Analytical Evaluation Board of Multi-turn LLM Agents. In *Advances in Neural Information Processing Systems*, volume 37, pp. 74325–74362, 2024. doi: 10.52202/079017-2365. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/877b40688e330a0e2a3fc24084208dfa-Abstract-Datasets_and_Benchmarks_Track.html.

Jane A. Ou and Stephen H. Penman. Financial statement analysis and the prediction of stock returns. *Journal of Accounting and Economics*, 11(4):295–329, nov 1989. doi: 10.1016/0165-4101(89)90017-7. URL https://www.sciencedirect.com/science/article/pii/0165410189900177.

Daniel Paleka, Shashwat Goel, Jonas Geiping, and Florian Tramèr. Pitfalls in Evaluating Language Model Forecasters. In *International Conference on Learning Representations*, 2026.

Guilherme Penedo, Hynek Kydlíček, Loubna Ben Allal, Anton Lozhkov, Margaret Mitchell, Colin Raffel, Leandro von Werra, and Thomas Wolf. The FineWeb datasets: Decanting the web for the finest text data at scale. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-0970. URL https://papers.nips.cc/paper_files/paper/2024/hash/370df50ccfdf8bde18f8f9c2d9151bda-Abstract-Datasets_and_Benchmarks_Track.html.

Archiki Prasad, Swarnadeep Saha, Xiang Zhou, and Mohit Bansal. ReCEval: Evaluating Reasoning Chains via Correctness and Informativeness. In Houda Bouamor, Juan Pino, and Kalika Bali (eds.), *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 10066–10086, Singapore, dec 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.emnlp-main.622. URL https://aclanthology.org/2023.emnlp-main.622/.

Scott Richardson, Siew Hong Teoh, and Peter D. Wysocki. The walk-down to beatable analyst forecasts: The role of equity issuance and insider trading incentives. *Contemporary Accounting Research*, 21(4):885–924, 2004. doi: 10.1506/KHNW-PJYL-ADUB-0RP6. URL https://doi.org/10.1506/KHNW-PJYL-ADUB-0RP6.

Suproteem K. Sarkar and Keyon Vafa. Lookahead Bias in Pretrained Language Models, jun 2024. URL https://papers.ssrn.com/abstract=4754678.

Dong Shu, Yanguang Liu, Huopu Zhang, and Mengnan Du. FinCall-Surprise: A Large Scale Multi-modal Benchmark for Earning Surprise Prediction. In Maria Liakata, Viviane P. Moreira, Jiajun Zhang, and David Jurgens (eds.), *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13357–13370, San Diego, California, United States, jul 2026. doi: 10.18653/v1/2026.acl-long.610. URL https://aclanthology.org/2026.acl-long.610/.

Colin White, Samuel Dooley, Manley Roberts, Arka Pal, Ben Feuer, Siddhartha Jain, Ravid Shwartz-Ziv, Neel Jain, Khalid Saifullah, Sreemanti Dey, Shubh-Agrawal, Sandeep Singh Sandha, Siddartha Naidu, Chinmay Hegde, Yann LeCun, Tom Goldstein, Willie Neiswanger, and Micah Goldblum. LiveBench: A Challenging, Contamination-Limited LLM Benchmark. In *International Conference on Learning Representations*, apr 2025.

Yijia Xiao, Rujun Han, Yanfei Chen, Zifeng Wang, Ke Jiang, Zhongying CuiZhu, Vishy Tirumalashetty, Wei Wang, Burak Gokturk, Tomas Pfister, and Chen-Yu Lee. FinanceHarness: Autonomous Financial Deep Research Framework, aug 2026. URL https://arxiv.org/abs/2607.27853.

Miao Xiong, Zhiyuan Hu, Xinyang Lu, Yifei Li, Jie Fu, Junxian He, and Bryan Hooi. Can LLMs Express Their Uncertainty? An Empirical Evaluation of Confidence Elicitation in LLMs. In *International Conference on Learning Representations*, pp. 23650–23678, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/6733cf15e10e2cd1d59af033c3bb8507-Abstract-Conference.html.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. SWE-agent: Agent-Computer Interfaces Enable Automated Software Engineering. In *Advances in Neural Information Processing Systems*, nov 2024.

Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ-bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains. In *International Conference on Learning Representations*, 2025.

Meiqun Yin and Mengzhu Guo. Complex forecasting and investment strategy optimization via chain-of-thought of large language models. *Expert Systems with Applications*, 298:129913, mar 2026. doi: 10.1016/j.eswa.2025.129913. URL https://www.sciencedirect.com/science/article/pii/S0957417425035286.

Zhiyuan Zeng, Jiashuo Liu, Siyuan Chen, Tianci He, Yali Liao, Yixiao Tian, Jinpeng Wang, Zaiyuan Wang, Yang Yang, Lingyue Yin, Mingren Yin, Zhenwei Zhu, Tianle Cai, Xinjie Chen, Zehui Chen, Jiecao Chen, Yantao Du, Xiang Gao, Jiacheng Guo, Liang Hu, Jianpeng Jiao, Xiangsheng Li, Jingkai Liu, Shuang Ni, Zhoufutu Wen, Ge Zhang, Kaiyuan Zhang, Xin Zhou, Jose Blanchet, Xipeng Qiu, Mengdi Wang, and Wenhao Huang. FutureX: An Advanced Live Benchmark for LLM Agents in Future Prediction. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=z28PLIEj6l.

Andy K. Zhang, Neil Perry, Riya Dulepet, Joey Ji, Celeste Menders, Justin W. Lin, Eliot Jones, Gashon Hussein, Samantha Liu, Donovan Jasper, Pura Peetathawatchai, Ari Glenn, Vikram Sivashankar, Daniel Zamoshchin, Leo Glikbarg, Derek Askaryar, Mike Yang, Teddy Zhang, Rishi Alluri, Nathan Tran, Rinnara Sangpisit, Polycarpos Yiorkadjis, Kenny Osele, Gautham Raghupathi, Dan Boneh, Daniel E. Ho, and Percy Liang. Cybench: A Framework for Evaluating Cybersecurity Capabilities and Risks of Language Models. In *International Conference on Learning Representations*, 2025.

Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. WebArena: A Realistic Web Environment for Building Autonomous Agents. In *International Conference on Learning Representations*, apr 2024.

Andy Zou, Tristan Xiao, Ryan Jia, Joe Kwon, Mantas Mazeika, Richard Li, Dawn Song, Jacob Steinhardt, Owain Evans, and Dan Hendrycks. Forecasting future world events with neural networks. In *Advances in Neural Information Processing Systems*, volume 35, pp. 27293–27305, 2022. doi: 10.52202/068431-1979.

Appendix for EPSBench

A Additional Related Work **14**

B Environment and Protocol Details **15**

B.1 Schema Construction and Validation 15

B.2 Information Environment 16

B.3 Tools and Run Protocol 16

B.4 Evaluation and Aggregation 17

C Companies in the Benchmark **17**

D Model Details **17**

E Harness Settings **18**

F Memorization Test for Qwen 3.5 **18**

G Additional Results **20**

H Case Study: Boeing's Divestiture **22**

I Complete Example of a Generated Schema **23**

J Prompts **25**

A Additional Related Work

Earnings forecasting. Comparisons between automated earnings forecasts and analyst forecasts depend on what information the methods receive and what they are asked to predict. Early models used adjacent-quarter and seasonal patterns, while analysts benefited in part from information that arrived before they issued their forecasts (Foster, 1977; Brown & Rozeff, 1979; Brown et al., 1987). Subsequent work used financial-statement information to predict the direction of earnings changes and construct stock trading strategies (Ou & Penman, 1989). Comparisons with analyst forecasts show that relative performance depends on the forecasting method and evaluation setting (Chen et al., 2022; Ball & Ghysels, 2018; Elend et al., 2020). LLM-based studies extend these approaches to direction and earnings-surprise prediction from statements and calls, and to direct EPS prediction from structured inputs that include analyst forecasts (Yin & Guo, 2026; Shu et al., 2026; An & Lee, 2026). Search-enabled forecasting systems further allow models to gather evidence when predicting financial quantities (Zeng et al., 2026; Li et al., 2026). EPSBench follows this numerical forecasting tradition while specifying a common accounting task: agents estimate company-specific income-statement drivers from retrieved documents, and we evaluate the resulting quarterly GAAP EPS forecasts and intervals.

Financial research and agent environments. Interactive benchmarks extend evaluation from reasoning over supplied information to acquiring that information and acting on it through tools. FinQA studies numerical reasoning over provided financial reports, whereas FinAgentBench evaluates retrieval of relevant financial documents and passages (Chen et al., 2021; Choi et al., 2025). More general environments such as AgentBench and WebArena assess multi-step interaction, and studies of agent interfaces and scaffolds show how the surrounding harness can affect performance (Liu et al., 2024; Zhou et al., 2024; Yang et al., 2024; Zhang et al., 2025; Kapoor et al., 2026). FinanceHarness brings controlled model and harness comparisons to financial research reports using

point-in-time data and expert-validated rubrics (Xiao et al., 2026). EPSBench applies a related experimental approach to a numerical outcome: the main comparison holds the task, tools, accounting calculator, and harness fixed, while additional comparisons change the information sources or harness for selected models.

Temporal validity of forecasting evaluations. A forecasting evaluation must distinguish information available at the forecast date from information that becomes available afterward. ForecastBench collects predictions before questions resolve, while retrieval-based forecasting systems restrict evidence by publication date (Karger et al., 2025; Halawi et al., 2024; Zou et al., 2022). Retrospective studies face additional difficulties because pretrained models may have encountered the outcomes and web pages may have been updated after their recorded dates (Paleka et al., 2026; Sarkar & Vafa, 2024; Lopez-Lira et al., 2025). Benchmarks built from recent sources and studies using news after model cutoffs reduce some of these risks, but recency alone does not establish that a particular evaluation is free of exposure (White et al., 2025; Lopez-Lira & Tang, 2026). EPSBench uses document-date restrictions and examines the separation between model training and outcome announcements.

Uncertainty and interpretation of agent forecasts. Forecast evaluation and agent evaluation provide complementary criteria for assessing uncertainty, resource use, and the steps leading to an estimate. In probabilistic forecasting, calibration and sharpness jointly characterize uncertainty, and proper scoring rules provide incentives for truthful quantile reports (Gneiting et al., 2007; Gneiting & Raftery, 2007). Studies of verbalized LLM confidence document overconfidence, while agent evaluations also measure execution cost and reliability across runs (Xiong et al., 2024; Kapoor et al., 2025; Yao et al., 2025). AgentBoard and ReCEval examine intermediate progress and reasoning steps, while structured representations connect numerical answers to recorded computations (Ma et al., 2024; Prasad et al., 2023; Lyu et al., 2023). EPSBench brings these forms of assessment to the same forecasting task: interval metrics evaluate stated uncertainty, and driver records support descriptive comparisons of the assumptions behind EPS estimates. The accounting calculator fixes the relationship between submitted drivers and EPS; it does not establish that the recorded explanations faithfully describe the model's internal reasoning.

B Environment and Protocol Details

We provide the implementation details for the environment and evaluation protocol introduced in Section 3. We first describe how we build the forecasting schemas and prepare the documents, then specify the tools, run protocol, and evaluation measures.

B.1 Schema Construction and Validation

Schema extraction. We use o4-mini to extract income-statement tables from each 10-K and 10-Q. DeepSeek's `deepseek-reasoner` then reads the four quarterly statements preceding the target, $q-1$ through $q-4$, and generates the baseline leaves, formula DAG, and drivers described in Section 3.2. When $q-4$ is a fiscal fourth quarter without a standalone statement, we subtract the first nine months from the annual figures in the 10-K. We obtain those nine-month figures from the cumulative column of the third-quarter 10-Q when available; otherwise, we sum the first three quarterly statements. We use GPT-5.4 to extract the tax and share-count references.

Driver forms. Let V_q be a line item in the target quarter and V_{q-4} its value in the year-ago baseline quarter. Let A_q and A_{q-4} denote the corresponding values of an anchor line item, typically revenue. The agent predicts a growth rate g, a margin change δ in basis points, an absolute value, or an effective tax rate, and each driver takes one of four forms:

- **Year-over-year growth** (`yoy`), for scale line items such as segment revenues and R&D: $V_q = V_{q-4}\,(1+g)$, with g a decimal fraction.
- **Margin change in basis points** (`bps`), for cost items that scale with the anchor: $V_q = A_q\left(V_{q-4}/A_{q-4} + \delta/10{,}000\right)$, with δ an integer number of basis points, so that $\delta = -20$ lowers the cost ratio by 0.20 percentage points.

- **Absolute value** (`absolute`), for non-operating and capital-structure items such as interest, other income, income tax expense, the diluted share count, non-controlling interest, and preferred dividends.
- **Rate** (`rate_override`), for an effective tax rate applied to pretax income.

Dividing by 10,000 converts the integer basis-point inputs requested by the prompts into decimal changes in margins, as the prompts in Appendix J specify.

Tax and share-count references. To convert pretax income into diluted EPS, the agent must also forecast tax expense and the diluted share count. Unlike revenue and cost lines, neither follows from how the business is performing: the tax rate depends on tax law and one-off tax events, and the share count on buybacks and issuance, so the documents that describe the business say little about either. GAAP tax rates can vary sharply when a company records a one-off benefit or earns little pretax income. Share counts usually change more gradually through buybacks and issuance, making recent values useful references. The schema represents tax as an effective rate in 138 of the 149 instances, as an absolute expense in 10, and as both in one. The Microsoft example in Appendix I uses the absolute form.

Three historical references accompany the share-count driver in every schema, the tax-rate driver in all 139 schemas that use one, and the absolute tax driver in 4 of the 11 that use one: the most recent quarter (`last_q`), the trailing twelve months (`ttm`), and the year-ago quarter (`q4`), taken from the company's GAAP statements. We use their median as the default to reduce sensitivity to a single unusual quarter, and allow the agent to select a reference or provide its own value. The other seven absolute tax drivers default to the year-ago expense. We exclude adjusted tax-rate guidance from these references because it omits items that enter the GAAP earnings we evaluate.

Schema verification. We execute each schema's DAG on its extracted $q-4$ leaf values and compare the result with the EPS reported for that quarter. Of the 149 instances, 146 reproduce EPS within one cent and the other three within four cents. Rounding individual line items can accumulate in the final EPS calculation. We release the schemas together with the per-instance verification results, so that the evaluation can be reproduced without regenerating them.

B.2 Information Environment

News corpus and effective dates. We download WARC archives from CC-News and apply a seven-stage quality filter based on FineWeb (Penedo et al., 2024). The pipeline combines text cleaning, FastText language identification, Gopher repetition and quality heuristics, C4 rules, and a FineWeb quality pass. We then deduplicate the aggregate corpus by URL, title, and MinHash, rather than deduplicating each file separately. We embed the retained articles and store them in daily FAISS shards. The index used in the reported experiments contains 516 daily shards spanning February 2025 to July 2026. We embed with Qwen3-Embedding-8B and search an inner-product index; a query returns ten candidate articles by default, although the agent may request more.

We assign each article an effective date by taking the latest date from its structured markup, body, and URL, substituting the crawl date when the body date is missing. This conservative rule can move an article to a later date, reducing the risk that information becomes available to an instance prematurely. Nevertheless, historical availability cannot be reconstructed perfectly from web data: publication timestamps are not always complete or consistently represented across sources, and some pages may be updated after their recorded dates. We therefore treat the effective date as a conservative approximation of when an article could have been available.

Documents offered. For target quarter q, we direct the agent to the $q-1$ earnings-call transcript and the $q-1$ and $q-4$ filings. We also suggest the $q-2$ and $q-3$ filings as optional reading on trends and seasonality. The agent can request other filings or transcripts published on or before τ, but the server blocks the target-quarter filing and call and any document dated after τ. We expose filings through a table of contents so agents can read selected sections of a long filing.

B.3 Tools and Run Protocol

Tool suite. The MCP server of Section 3.3 exposes six tools:

- `list_financial_report_sections` / `read_financial_report_section`: enumerate and read individual sections of historical filings (balance sheet, MD&A, segment tables), so that long documents can be navigated selectively instead of loaded whole.
- `read_earnings_call_transcript`: retrieve the transcript of a past earnings call, where management discusses the outlook for coming quarters.
- `financial_search`: semantic retrieval over the date-sharded news index, subject to the temporal filtering described in Appendix B.2.
- `read_news_article`: fetch the full text of a retrieved article.
- `financial_calculate`: compute the income statement and EPS from the agent's driver values. The calculator applies these inputs to the schema and executes the accounting DAG in topological order.

Run protocol. Each run is a single attempt that ends when the agent sends its final message after calling `financial_calculate`. We impose no turn limit beyond the model's context window and do not retry runs. When the agent requests a missing document or one dated after τ, the server returns an error and the run continues. We log and count these refusals. If we cannot parse the final message into the required JSON, we record a format failure and no forecast. These rules apply to all models, harnesses, and information conditions, and failures are reported alongside point forecast error, as Section 3.4 describes.

B.4 Evaluation and Aggregation

Interval score. For a reported p10 l, p90 u, and reported EPS y, the interval score of Gneiting & Raftery (2007) for a central $(1-\alpha)$ interval with $\alpha = 0.2$ is

$$\mathrm{IS} = \frac{1}{P}\Big[(u-l) + \tfrac{2}{\alpha}\,(l-y)^{+} + \tfrac{2}{\alpha}\,(y-u)^{+}\Big],$$

where $(x)^{+} = \max(x, 0)$ and P is the share price used for PDE, so IS is in the same price-deflated units as PDE, with lower scores indicating better performance. The score penalizes each miss by ten times its distance from the nearest bound and charges for interval width. It is proper for the elicited p10 and p90 quantiles, balancing the penalties for missed outcomes against the cost of wider intervals. We report the mean over runs that return an interval because propriety holds for the expected score. A median would fall among covered runs, where the score equals the width, and would therefore reward narrow intervals.

Efficiency measures. We record model requests, tool calls, output and reasoning tokens, and total input tokens for each run. Total input sums the tokens carried by every request, including cached tokens and conversation history sent again in later requests. We report medians over runs and use token counts because providers price cached input differently. Format compliance is the share of runs whose final message parses into the required forecast.

Company-level resampling. Two quarters from the same company may not be independent. Resampling companies instead of instances, and keeping all of a company's instances together, moves the endpoints of the intervals against consensus in Table 1 by at most 0.002 pp and changes no conclusion.

C Companies in the Benchmark

Table 5 lists the 86 companies by sector. To describe the sample, we use sector labels and market capitalizations from Yahoo Finance as of September 2026. The median market capitalization is \$212B, and 85 of the 86 companies exceed \$10B.

D Model Details

Table 6 summarizes the models evaluated in our experiments. We report release dates and knowledge cutoffs from vendor disclosures where available; "undisclosed" indicates that the vendor has not disclosed a value for that field. Four models have published knowledge cutoffs preceding all benchmark announcements. Qwen 3.5 and GLM 4.5 Air do not disclose knowledge cutoffs.

Table 5: **Benchmark companies by sector.** The 86 companies across eight sectors, with company counts (n) and ticker symbols. Sector classifications follow Yahoo Finance as of September 2026.

Sector	n	Tickers
Technology	20	ADBE, AMAT, AMD, AVGO, CRM, CRWD, CSCO, DELL, IBM, INTC, INTU, MSFT, MU, NVDA, ORCL, PANW, PLTR, QCOM, SMCI, UBER
Consumer Cyclical	15	AMZN, BBY, CCL, CMG, CVNA, GM, HD, LOW, LULU, M, MCD, MELI, NKE, SBUX, TSLA
Financial Services	14	AXP, BAC, BLK, C, COIN, GS, HOOD, JPM, MA, MS, PYPL, SPGI, V, WFC
Industrials	11	BA, CAT, DAL, GE, HON, LMT, PLUG, RTX, UAL, UNP, UPS
Healthcare	9	ABBV, AMGN, ISRG, JNJ, LLY, MRK, PFE, TMO, UNH
Consumer Defensive	7	COST, KHC, KO, PEP, PG, TGT, WMT
Communication Services	6	CMCSA, DIS, GOOGL, META, NFLX, WBD
Energy	4	COP, CVX, SLB, XOM

Table 6: **Evaluated model specifications.** Model identifiers, total and active parameter counts, release dates, and knowledge cutoffs from vendor disclosures. Entries marked "undisclosed" were not provided by the vendor.

Model identifier	Parameters total / active	Released	Knowledge cutoff
claude-opus-4.6	undisclosed	2026-02-05	2025-08
gpt-5.5	undisclosed	2026-04-23	2025-12-01
gpt-5.4	undisclosed	2026-03-05	2025-08-31
gpt-5-mini	undisclosed	2025-08-07	2024-05-31
qwen3.5-122b-a10b	122B / 10B	2026-02-24	undisclosed
glm-4.5-air	106B / 12B	2025-07-28	undisclosed

E Harness Settings

An agent harness is the framework between the model and the tools: it assembles the prompt sent at each turn, routes the model's tool calls to the MCP server, and decides what to keep in the context as the conversation grows. The harness experiments in Section 5.5 hold everything else fixed, namely the 149 instances, the schemas, the MCP server and its six tools described in Appendix B.3, and the system and user prompts given in Appendix J, and vary only the harness and the reasoning effort.

Table 7 lists the three harnesses. OpenCode is the primary harness for all six models. We drive it through its SDK and disable its built-in shell, file, and web tools, so the six MCP tools are the agent's only interface. Claude Code, Anthropic's agent harness, runs Opus 4.6 under the same restriction. The vanilla harness is a plain tool-calling loop over the OpenAI SDK that we wrote for this comparison. It resends the full conversation each turn and adds no system prompt beyond the benchmark's, no context management, and no tool orchestration, so it shows how GPT-5.5 behaves without the scaffolding the other harnesses provide.

All harnesses send the same benchmark prompts and call the same MCP server. Every model runs at the provider's default reasoning effort. To separate the effect of effort from that of the harness, we also run Opus 4.6 at maximum effort under both OpenCode and Claude Code, and GPT-5.5 at maximum effort under OpenCode. No configuration sets a temperature.

F Memorization Test for Qwen 3.5

The vendor released Qwen 3.5 on 24 February 2026, during the benchmark window, without publishing a knowledge cutoff. Of the 149 instances, 79 have announcements after that date and 70

Table 7: **Harness configurations.** Every harness runs at the provider's default reasoning effort; the maximum-effort configurations for Opus 4.6 and GPT-5.5 are listed in Table 4. Temperature is left unset throughout.

Harness	Version	Models	Configuration
OpenCode	SDK 1.17.9	all six	Driven through the SDK; built-in shell, file, and web tools disabled by the agent definition. Primary harness.
Claude Code	2.1.269	Opus 4.6	Driven through the SDK; built-in shell, file, and web tools disabled.
Vanilla harness	ours	GPT-5.5	Plain tool-calling loop over the OpenAI SDK; resends the full conversation every turn; no system prompt beyond Listing 2, no context management, no tool orchestration.

on or before it. We therefore probe whether the model can recall reported EPS across successive announcement periods, including the benchmark window.

Probe. We ask the model for a named company's GAAP diluted EPS in a specified fiscal quarter, without tools or documents, and instruct it to answer UNKNOWN if it does not know. We use an unguided prompt that asks directly and a guided prompt that encourages recall from training data. We query each firm-quarter three times with each prompt and count an answer as a *hit* when its PDE against reported EPS, defined in Section 3.4, is at most 0.02%.

To assess incidental matches, we compare four strategies that use no memory of the target quarter at three candidate thresholds, which Table 8 reports. Copying analyst consensus, the strongest tested strategy at 0.02%, achieves a 9% hit rate; its rate rises to 22% at 0.05%. We use the tighter threshold to reduce such matches, while recognizing that a hit need not imply memorization. Each reference strategy receives one attempt per firm-quarter, compared with six for the model.

Table 8: **Reference hit rates for the memorization test.** Hit rates of non-memorizing strategies on 149 firm-quarters at three PDE thresholds. A hit occurs when the forecast's PDE does not exceed the threshold.

Strategy	0.02%	0.05%	0.10%
Copy analyst consensus	9%	22%	37%
Copy previous quarter ($q-1$)	4%	14%	27%
Copy same quarter last year ($q-4$)	4%	8%	22%
Guess zero	1%	2%	5%

Firm-quarters probed. We construct a historical sample from the same 86 companies, using announcements from June 2024 to June 2026. We group these into nine calendar-quarter buckets, from 2024Q2 to 2026Q2, and sample 30 firm-quarters per bucket, using announcement dates estimated at sampling time for some older quarters. The first bucket contains only seven available cases, giving 247 in total. After we replaced the estimated dates with actual announcement dates from IBES, one firm-quarter moved from the third bucket to the fourth, so Table 9 lists 29 and 31 for these two periods. This sample lets us compare elicited recall across announcement periods. We also probe all 149 benchmark instances and report the results in Table 9.

Results. We count a firm-quarter as a hit if at least one of its six responses meets the threshold. Table 9 shows hits for announcements in 2024 and none from January 2025 onward, including every benchmark instance. Across six responses per instance, none of the 149 benchmark instances yields a hit at the selected threshold, indicating no detectable recall under the tested protocol. When asked about its own cutoff, the model answers "December 2024". This response is consistent with the observed pattern but provides only weak additional evidence.

Table 9: **Qwen 3.5 memorization test.** Recall results by announcement period and on the benchmark instances. A firm-quarter counts as a hit if at least one of six responses achieves PDE $\leq 0.02\%$.

Announcement dates	Firm-quarters	Hits	Hit rate
June 2024	7	1	14%
11 July to 9 September 2024	29	10	34%
1 October to 19 December 2024	31	6	19%
10 January 2025 to June 2026 (six buckets)	180	0	0%
Benchmark instances (December 2025 to June 2026)	149	0	0%

G Additional Results

Detailed evaluation results. Tables 10–13 supplement the evaluation in Section 5. Table 10 reports interval coverage, tail rates, width, and score. These statistics include only runs that return an interval, with the denominator reported for each model. Table 11 reports resource use, format compliance, and the number of instances with large errors. Peak context is the largest input-token count in any single request within a run. Table 13 reports all pairwise model comparisons underlying the point-forecast analysis in Section 5.1.

Resource use and compliance. The resource measurements show that processing more input does not necessarily accompany lower error. The three leading models make 35–49 tool calls per run, compared with 20–27 for the others, but GPT-5-mini processes the most input at 1.35M tokens per run. Its 28 model requests repeatedly carry the conversation history, whereas Table 11 shows 12–21 requests for the other models. Opus 4.6 and GPT-5.5 therefore achieve lower error while processing fewer input tokens than GPT-5-mini. All models return parseable forecasts in at least 98.2% of runs. Eight of the nine unparseable runs come from GPT-5-mini, while GLM 4.5 Air, the model with the highest error, parses every run, so compliance does not track point forecast error.

Table 10: **Prediction interval calibration and forecast bias.** Interval metrics pool runs returning a p10–p90 interval. Below, Inside, and Above report outcome percentages relative to the interval; Inside is empirical coverage. Width is the median interval width, and Score is the mean interval score. Signed error is the median of $(y-\hat{y})/P$ over scoreable runs, with a 90% bootstrap confidence interval that resamples instances; positive values indicate underprediction.

		Outcome location (%)					
Model	Intervals returned	Below	Inside	Above	Width (pp)	Score (pp)	Signed error, pp (90% CI)
Opus 4.6	447 / 447	10	70	20	0.41	2.87	$+0.021$ $[+0.007, +0.044]$
GPT-5.5	447 / 447	10	61	29	0.33	3.08	$+0.038$ $[+0.016, +0.068]$
GPT-5.4	447 / 447	16	52	32	0.25	4.29	$+0.037$ $[+0.015, +0.061]$
GPT-5-mini	439 / 447	18	50	32	0.39	19.61	$+0.047$ $[+0.010, +0.086]$
Qwen 3.5	446 / 447	25	37	38	0.26	4.87	$+0.039$ $[+0.006, +0.072]$
GLM 4.5 Air	409 / 447[1]	22	44	34	0.33	6.48	$+0.048$ $[+0.021, +0.080]$
Analyst consensus	–	–	–	–	–	–	$+0.028$ $[+0.008, +0.051]$
Expected split	–	10	80	10	–	–	–

Source-condition interaction. To complement the information-source comparisons in Section 5.4 and Table 3, we compute an interaction estimate for each instance. Let d_{N}, d_{F}, and d_{B} be the changes in the instance's PDE, the median over its three runs, when news, filings and transcripts, or both are removed, relative to the full environment. The interaction estimate is $d_{\mathrm{B}} - (d_{\mathrm{N}} + d_{\mathrm{F}})$, which is positive when removing both sources hurts more than the two single removals combined.

[1] Context compaction inside the harness was triggered on 49 of GLM 4.5 Air's 447 runs (its 128K window); these runs are scored as delivered and account for all of its missing intervals.

Table 11: **Resource use, format compliance, and large errors.** Panel (a) reports medians over runs; input includes cached tokens. The APIs for Opus 4.6 and GLM 4.5 Air do not report reasoning tokens separately, so their reasoning is counted in Output and the Reasoning column is n/a. Panel (b) reports format compliance, the share of runs whose final message parses into the required forecast, and the number of the 149 instances whose median PDE over runs exceeds 1%.

(a) Resource use per run

Model	Requests	Tools	Input (K)	Output (K)	Reasoning (K)	Peak context (K)
Opus 4.6	15	35	1056	9.7	n/a	112
GPT-5.5	12	47	837	3.6	2.6	119
GPT-5.4	13	49	949	3.7	7.2	125
GPT-5-mini	28	27	1354	2.7	3.6	90
Qwen 3.5	14	23	542	3.0	5.1	61
GLM 4.5 Air	21	20	738	3.4	n/a	59

(b) Format compliance and error frequency

Model	Compliance (%)	Instances with PDE $> 1\%$
Opus 4.6	100	8
GPT-5.5	100	11
GPT-5.4	100	11
GPT-5-mini	98.2	13
Qwen 3.5	99.8	14
GLM 4.5 Air	100	23

For GPT-5.5, its median over the 149 instances is $+0.037$ pp, with a bootstrap 90% interval of $[+0.020, +0.066]$ computed as in Section 3.4, and it is positive on 100 instances. A median of instance-level differences need not equal the difference between the medians in Table 3, so this estimate is not the gap between the displayed values.

Source-removal groups. For each instance and condition, we take the median PDE over the three runs; a removal effect is the difference between this value in the reduced condition and in the full environment. An effect is harmful when it exceeds 0.057 pp, the median over the 149 instances and four conditions of the range between the highest and lowest PDE among an instance's three runs. Table 12 places an instance in the first group when no removal is harmful, in the second when only the joint removal is harmful, and in the third when at least one single removal is harmful. The rank correlation of -0.01 is computed over all 149 instances between the two single-removal effects.

Table 12: **Instance-level effects of information-source removal.** GPT-5.5 results on 149 firm-quarters. A removal is harmful when its error increase exceeds 0.057 pp, the median spread of PDE across an instance's three runs. Error increases from removing filings and transcripts versus removing news have a Spearman correlation of -0.01.

Group	Instances	Share (%)
No removal is harmful	67	45
Only removing both is harmful	32	21
A single removal is harmful	50	34
only removing filings and transcripts	17	11
only removing news	24	16
either removal	9	6

Accounting-stage decomposition. To complement the analysis in Section 5.3, we decompose forecast error on the 125 instances with an operating-income subtotal. For each run, we measure absolute forecast error at operating income, pretax income, and EPS, placing all three on the PDE scale by dividing the two income errors by the target quarter's reported diluted share count and then by share

Table 13: **Pairwise model comparisons of point forecast error.** Median paired ΔPDE (A − B) on 149 firm-quarters, with 90% bootstrap confidence intervals and counts of instances where each model has lower PDE. Negative differences indicate lower error for model A.

Model A	Model B	ΔPDE (A − B), pp	90% CI	A wins / B wins
Opus 4.6	GPT-5.5	-0.004	$[-0.013, +0.002]$	81 / 68
Opus 4.6	GPT-5.4	-0.001	$[-0.009, +0.005]$	79 / 70
Opus 4.6	GPT-5-mini	-0.032	$[-0.042, -0.026]$	105 / 44
Opus 4.6	Qwen 3.5	-0.045	$[-0.054, -0.031]$	110 / 39
Opus 4.6	GLM 4.5 Air	-0.048	$[-0.063, -0.031]$	107 / 42
GPT-5.5	GPT-5.4	$+0.002$	$[-0.000, +0.006]$	67 / 82
GPT-5.5	GPT-5-mini	-0.029	$[-0.038, -0.020]$	107 / 42
GPT-5.5	Qwen 3.5	-0.035	$[-0.046, -0.019]$	105 / 44
GPT-5.5	GLM 4.5 Air	-0.042	$[-0.063, -0.032]$	105 / 44
GPT-5.4	GPT-5-mini	-0.028	$[-0.042, -0.019]$	100 / 49
GPT-5.4	Qwen 3.5	-0.038	$[-0.057, -0.021]$	108 / 41
GPT-5.4	GLM 4.5 Air	-0.038	$[-0.063, -0.026]$	106 / 43
GPT-5-mini	Qwen 3.5	-0.007	$[-0.017, +0.003]$	83 / 66
GPT-5-mini	GLM 4.5 Air	-0.009	$[-0.016, +0.002]$	83 / 66
Qwen 3.5	GLM 4.5 Air	-0.003	$[-0.015, +0.008]$	77 / 72

price P. Writing e_{op}, e_{pt}, and e_{EPS} for these errors, the three components are e_{op}, $e_{\text{pt}} - e_{\text{op}}$, and $e_{\text{EPS}} - e_{\text{pt}}$. Thus they sum to PDE, and a negative increment means that error shrinks at that stage. We take the median of each component over an instance's three runs and report medians across instances. The last column of Table 14 counts the instances on which each component has the largest absolute magnitude.

Evidence coding for Table 2. For each run on a far instance, we reset each driver in turn to its year-ago value, with zero growth or margin change for growth and margin drivers, recompute EPS with the calculator, and take the driver whose reset changes EPS the most as the run's most-adjusted driver. We then read that driver's evidence line in the agent's final message and count the run as citing a specific number if the part of the line before any arrow to the computed value contains a dollar amount, a percentage, a figure with thousands separators, or a number with a unit such as basis points, million, or billion. A run whose most-adjusted driver has no evidence line counts as not citing a number, so the denominator is all runs on the 50 far instances.

H Case Study: Boeing's Divestiture

Boeing's fourth quarter of 2025 shows how the driver records locate differences between forecasts. Boeing completed the \$10.55B sale of Jeppesen and ForeFlight on 3 November 2025, about twelve weeks before its earnings announcement. The sale price is not earnings; the forecasting question is how large a gain Boeing would recognize in the quarter. Boeing's schema has a driver for this gain, which enters operating income. In every run, Opus 4.6, GPT-5.5, and GPT-5.4 entered \$8.0–9.6B, working from the sale price and the net book value of the assets held for sale disclosed in the third-quarter filing; with their other drivers unchanged, this entry adds \$9.0–12.4 to their EPS forecasts. Qwen 3.5 found the transaction but concluded that no gain would be recognized in the quarter, and it and GLM 4.5 Air entered a gain close to zero in every run. GPT-5-mini entered about \$9.6B in two of its three runs, but its other drivers put operating income \$4–5B below the leading models' forecasts. Boeing reported GAAP diluted EPS of \$10.23, against a consensus of $-\$0.31$. The median forecasts over three runs were \$8.78 for Opus 4.6, \$9.16 for GPT-5.5, \$10.48 for GPT-5.4, \$5.06 for GPT-5-mini, $-\$7.44$ for Qwen 3.5, and $-\$7.57$ for GLM 4.5 Air. The gain driver accounts for \$10–12 of the roughly \$17 gap between the leading models and Qwen 3.5 and GLM 4.5 Air, and the records attribute the rest to revenue and cost drivers.

Table 14: **Forecast error decomposition by accounting stage.** Results on 125 instances with an operating-income subtotal. Operating reports the median error at operating income; Non-operating and Tax and shares report the median error increments from operating to pretax income and from pretax income to EPS, respectively, all on the PDE scale. Negative increments indicate a reduction in error. The last column counts instances where each component is largest in absolute magnitude (O: operating; N: non-operating; T: tax and shares).

Model	Operating (pp)	Non-operating (pp)	Tax and shares (pp)	Largest component O / N / T
Opus 4.6	0.092	0.007	−0.017	80 / 26 / 19
GPT-5.5	0.089	0.013	−0.016	83 / 20 / 22
GPT-5.4	0.098	0.013	−0.018	88 / 22 / 15
GPT-5-mini	0.149	0.020	−0.031	84 / 27 / 14
Qwen 3.5	0.166	0.012	−0.025	91 / 23 / 11
GLM 4.5 Air	0.166	0.012	−0.032	86 / 26 / 13

I Complete Example of a Generated Schema

Listing 1 reproduces the complete, unedited schema that our extraction pipeline generated for Microsoft (`MSFT`), fiscal 2026Q3, following Section 3.2. The `variables_baseline` block stores leaf values for 2025Q3, the year-ago quarter, while `formulas` defines the DAG and `prediction_schema` lists the agent's input fields. A field can set several leaves: `cogs_margin_delta_bps`, for example, controls both cost-of-revenue lines.

This schema represents income tax as an absolute value, `income_taxes_abs`, with the year-ago expense as its default. The share-count field carries the three references described in Appendix B.1: `last_q`, `ttm`, and `q4`. We use the reported EPS values in `_validation` to verify the schema and score forecasts, and, as Section 3.3 describes, keep this block inaccessible to the agent.

Listing 1: The complete schema `MSFT_2026Q3.json`.

```
{
  "ticker": "MSFT",
  "target_fq": "2026Q3",
  "baseline_fq": "2025Q3",
  "variables_baseline": {
    "product_revenue": 15319.0,
    "service_and_other_revenue": 54747.0,
    "product_cogs": 3037.0,
    "service_and_other_cogs": 18882.0,
    "rd": 8198.0,
    "sales_and_marketing": 6212.0,
    "general_and_administrative": 1737.0,
    "other_expense_net": 623.0,
    "income_taxes": 5553.0,
    "shares": 7461.0
  },
  "formulas": {
    "total_revenue": "product_revenue + service_and_other_revenue",
    "total_cogs": "product_cogs + service_and_other_cogs",
    "gross_profit": "total_revenue - total_cogs",
    "total_opex": "rd + sales_and_marketing + general_and_administrative"
        ,
    "operating_income": "gross_profit - total_opex",
    "pretax_income": "operating_income - other_expense_net",
    "net_income": "pretax_income - income_taxes",
    "earnings_to_common": "net_income",
    "eps": "earnings_to_common / shares"
  },
  "prediction_schema": {
    "product_revenue_yoy": {
```

```
      "variable": "product_revenue",
      "type": "yoy",
      "constituents": [
        "product_revenue"
      ],
      "description": "Product revenue YoY growth rate vs Q-4"
    },
    "service_and_other_revenue_yoy": {
      "variable": "service_and_other_revenue",
      "type": "yoy",
      "constituents": [
        "service_and_other_revenue"
      ],
      "description": "Service and other revenue YoY growth rate vs Q-4"
    },
    "cogs_margin_delta_bps": {
      "variable": "total_cogs",
      "type": "bps",
      "anchor": "total_revenue",
      "constituents": [
        "product_cogs",
        "service_and_other_cogs"
      ],
      "description": "Combined cost of revenue as % of total revenue,
          delta vs Q-4"
    },
    "rd_yoy": {
      "variable": "rd",
      "type": "yoy",
      "constituents": [
        "rd"
      ],
      "description": "R&D expense YoY growth rate vs Q-4"
    },
    "sales_and_marketing_yoy": {
      "variable": "sales_and_marketing",
      "type": "yoy",
      "constituents": [
        "sales_and_marketing"
      ],
      "description": "Sales and marketing expense YoY growth rate vs Q-4"
    },
    "general_and_administrative_yoy": {
      "variable": "general_and_administrative",
      "type": "yoy",
      "constituents": [
        "general_and_administrative"
      ],
      "description": "General and administrative expense YoY growth rate
          vs Q-4"
    },
    "other_expense_net_abs": {
      "variable": "other_expense_net",
      "type": "absolute",
      "description": "other_expense_net absolute dollar value for the
          target quarter (predict the amount directly; 0 if none expected
          )"
    },
    "income_taxes_abs": {
      "variable": "income_taxes",
      "type": "absolute",
      "default": 5553.0,
      "description": "income_taxes absolute dollar value for the target
          quarter (below-pretax line; predict directly, 0 if none
          expected)."
```

```
    },
    "shares_abs": {
      "variable": "shares",
      "type": "absolute",
      "default": 7461.0,
      "references": {
        "last_q": 7460.0,
        "ttm": 7462.0,
        "q4": 7461.0
      },
      "description": "shares: diluted weighted-average share count for
          the target quarter (predict the count directly; drifts slowly
          with buybacks/issuance)."
    }
  },
  "_validation": {
    "baseline_fq_eps": 3.46,
    "target_fq_eps": 4.27,
    "target_fq_pretax": 39340.0
  }
}
```


J PROMPTS

For a given instance and information condition, all models and harnesses receive the same system and user prompts. The system prompt in Listing 2 is static. The user prompt is generated per instance from the schema and the cutoff; Listing 3 is the one issued for Microsoft, fiscal 2026Q3, the instance whose schema Appendix I shows. The reduced information conditions of Section 5.4 change only the text that refers to removed tools. When news is removed, the news tools are not registered, and the user prompt drops the news window and the instruction to search news; the system prompt is unchanged, and no run attempts a news call. When filings and transcripts are removed, the three document tools are not registered, the user prompt replaces the required reading with a news search and names the year-ago baseline and the tax and share references as the only historical figures, and the system prompt states that the document tools are deliberately unavailable. When both are removed, no research tool is registered, and both prompts tell the agent to work from the year-ago baseline and the references alone. The release includes the prompts for every condition.

Listing 2: The system prompt.

```
# Financial Forecaster (Fundamental Update)

Run ONE complete forecasting iteration for a specified target ticker and fiscal quarter. Your
task is to act as a quantitative analyst updating a financial model. Start by carefully
reading the user message to identify the exact variables and units you must (and can only)
predict, and to obtain the target ticker, fiscal quarter, and the exact Q-4 Baseline Data.
From there, use the available tools at your own discretion --- in whatever order and
combination makes sense --- to read the baseline filings and recent qualitative text (Reports,
 ECT, News), predict the relative changes for specific core drivers, and offload the actual
computation entirely to the calculator tool.

## CRITICAL CONSTRAINTS

- **No Numerical Hallucination:** You MUST NOT predict absolute values for `Total Revenue` or
`EPS` directly. You ONLY predict the relative changes (YoY growth rates, margin bps deltas)
for the exact fields requested in the prompt.
- **No Manual Calculation:** Do not attempt to compute the final absolute values yourself. You
 MUST use the `financial_calculate` tool.
- **Strict Input Fields:** When calling `financial_calculate`, the arguments must strictly
match the required prediction fields (e.g., `products_revenue_yoy`). Never invent or
hallucinate new parameters.
- **Maximize Information Retrieval:** Before predicting, you MUST maximize your use of the
search tool and read multiple relevant news articles. Do not jump to conclusions after reading
 only one or two articles.
- **Comprehensive News Coverage Before Predicting:** Do not call `financial_calculate` until
you have built a comprehensive, multi-dimensional picture of the company's recent situation.
This means actively searching across multiple angles --- revenue trends by segment, margin
pressures, macro and regulatory factors, competitive dynamics, and management signals. Only
when further reading would no longer materially change your estimates should you stop and move
 to prediction.
```

```
- **Recency Bias:** When reading news, assign higher analytical weight to articles published closer to the cutoff date.
- **No Control Over the Information Cutoff:** The news search tool is restricted server-side to a fixed information window for this run. There is no parameter to widen it, query past it, or ask for a different date. Do not assume a "current date" --- rely only on what the tools return.
- **Never Fabricate on Tool Failure:** If any tool errors, times out, or becomes unavailable --- including `financial_calculate` --- do not retry indefinitely and do not invent a final JSON from memory. After a few failed attempts, output a single line starting with `ERROR:` describing what failed, and stop.
- **No Reverse-Engineering From Consensus:** If you encounter analyst consensus estimates for total EPS or total revenue (e.g. in an earnings-preview article), do NOT treat that aggregate figure as a target and back-solve your driver values to match it. Build each driver estimate independently and bottom-up from segment-specific evidence (unit sales, pricing, regional trends, cost data) gathered during your research. You may use a consensus figure only as a sanity check AFTER you have already derived your estimates --- a large gap from consensus is not, by itself, evidence that your estimate is wrong, as long as it is grounded in specific evidence you found. This is enforced procedurally: see the mandatory per-field evidence block in OUTPUT FORMAT below --- every field's value must trace to its own cited evidence, not to a total you picked first.

### Anti-Hallucination Rules

The most common failure mode is attempting to act as a calculator or hallucinating variables that do not exist in the schema.
- If the target requires `products_revenue_yoy`, NEVER attempt to provide `total_revenue_yoy`.
- If a qualitative news article mentions a $1 billion fine or a one-time gain, but there is no corresponding node in the prediction list, NEVER attempt to manually adjust any numbers to deduct or add it.
- Trust that the Python backend will use Kahn's Algorithm for correct topological sorting. Fully trust the calculated results summary returned by the tool.

## RESEARCH TOOLS

You have two categories of information sources, available in any order and combination you see fit:

- **Filings:** `read_earnings_call_transcript` and `list_financial_report_sections` -> `read_financial_report_section`, covering the Q-1 and Q-4 quarters, for historical performance, segment breakdown, and management guidance. Large sections are returned in chunks --- if a response has `"has_more": true`, call `read_financial_report_section` again with `offset` set to the returned `next_offset` to keep reading the same section; do not treat a single chunked call as the complete section.
- **News:** `financial_search` with distinct queries covering different angles --- revenue outlook by segment, cost pressures/tariffs/supply chain, macro and regulatory developments, competitive dynamics --- followed by `read_news_article` on the most relevant results.

There is no required reading order or fixed number of calls. Use your judgment about what to read and when, subject only to the constraints above (comprehensive coverage before predicting, recency weighting, and never fabricating on tool failure).

## INTERFACE DEFINITIONS

**Available Tools** (ticker and fiscal quarter for this run are fixed by the environment --- do not pass them):
- `read_earnings_call_transcript(quarter)`
- `list_financial_report_sections(quarter)`
- `read_financial_report_section(quarter, section_name)`
- `financial_search(query, top_k?)`
- `read_news_article(article_id)`
- `financial_calculate(**kwargs)`

## OUTPUT FORMAT

Once you have read enough to satisfy the constraints above, write a short evidence block --- **one line per required field** --- in this exact format, BEFORE calling `financial_calculate`:

'''
<field_name>: <specific evidence found, with a number where possible> -> <chosen value>
'''

Each line's evidence must be specific to that field's own segment/driver (unit sales, pricing, regional trends, cost data, management guidance) --- it must NOT be a residual you derived by picking a total revenue or EPS figure first and back-solving this field to match it. If you genuinely found no segment-specific evidence for a field, say so explicitly and state your fallback (e.g. "no segment-specific news found; carrying forward Q-4 trend") rather than silently reverse-engineering a number from a total.

Then formulate exact numerical estimates for the required fields, consistent with the evidence block above:
```

```
- `_yoy` fields must be decimals (e.g., `0.05` for +5%).
- `_bps` fields must be integers (e.g., `-20` for a 20 basis point change).

Call `financial_calculate` with these exact estimates. As soon as it returns successfully,
immediately output the final structured JSON below --- do not output any further analytical
text, and do not call any further tools. Assemble it as follows:
- `fundamental_predictions` = the exact arguments you passed to `financial_calculate`
- `calculated_results_summary` = the `income_statement` field returned by the tool

Do NOT output any other text or explanations outside of this JSON.

'''json
{
  "fundamental_predictions": {
    "<key_1>": <value>,
    "<key_2>": <value>
  },
  "calculated_results_summary": {
    "key_metrics": {
      "eps": <value_from_tool>,
      "total_revenue": <value_from_tool>
    },
    "detail": {
      "products_revenue": <value_from_tool>,
      "gross_profit": <value_from_tool>,
      "...": "..."
    }
  }
}
'''
```

Listing 3: The user prompt for `MSFT`, fiscal 2026Q3.

```
Ticker: MSFT
Forecast target fiscal quarter: 2026Q3
News window: 2026-01-28 to 2026-04-28. financial_search will only return articles with
pub_date in this range.
Required reading:
  - Earnings call transcript (Q-1): quarter=Q2_2026
  - Financial report sections (Q-1): quarter=Q2_2026
  - Financial report sections (Q-4): quarter=Q3_2025
You MAY also read the intervening quarters (Q-2=Q1_2026, Q-3=Q4_2025) --- their financial
reports --- to judge trend and seasonality. This matters here: you must forecast the WHOLE
income statement yourself --- every operating and non-operating line down to pretax income,
PLUS the below-pretax lines (GAAP effective tax rate and diluted share count). Use the last
four quarters to judge each line's trend/seasonality, and the GAAP tax/share history shown
with the prediction fields to anchor the tax rate and share count.

All YoY values are relative to the same fiscal quarter one year ago (Q-4 = year-ago baseline).

=== Q-4 BASELINE (from most recent 10-Q filing) ===
  product_revenue: 15,319.0000
  service_and_other_revenue: 54,747.0000
  product_cogs: 3,037.0000
  service_and_other_cogs: 18,882.0000
  rd: 8,198.0000
  sales_and_marketing: 6,212.0000
  general_and_administrative: 1,737.0000
  other_expense_net: 623.0000
  income_taxes: 5,553.0000
  shares: 7,461.0000

Read the required filings above first (ECT and financial report sections), then use
financial_search to discover relevant recent news. When ready, call financial_calculate with
your predicted driver values to see the full computed income statement.

PREDICTION FIELDS --- you must predict EXACTLY these fields, no more and no fewer.
Only these fields are valid inputs to financial_calculate; do not invent or add others:
  product_revenue_yoy: <float>  // Product revenue YoY growth rate vs Q-4. Decimal fraction. e
.g. 0.05 = +5% YoY, -0.03 = -3% YoY, 0.0 = flat.
  service_and_other_revenue_yoy: <float>  // Service and other revenue YoY growth rate vs Q-4.
 Decimal fraction. e.g. 0.05 = +5% YoY, -0.03 = -3% YoY, 0.0 = flat.
  cogs_margin_delta_bps: <float>  // Combined cost of revenue as % of total revenue, delta vs
Q-4. Basis points. e.g. -20 = margin improved 20 bps, 50 = cost ratio rose 50 bps, 0 =
unchanged.
  rd_yoy: <float>  // R&D expense YoY growth rate vs Q-4. Decimal fraction. e.g. 0.05 = +5%
YoY, -0.03 = -3% YoY, 0.0 = flat.
  sales_and_marketing_yoy: <float>  // Sales and marketing expense YoY growth rate vs Q-4.
Decimal fraction. e.g. 0.05 = +5% YoY, -0.03 = -3% YoY, 0.0 = flat.
```

```
  general_and_administrative_yoy: <float>  // General and administrative expense YoY growth
rate vs Q-4. Decimal fraction. e.g. 0.05 = +5% YoY, -0.03 = -3% YoY, 0.0 = flat.
  other_expense_net_abs: <float>  // other_expense_net absolute dollar value for the target
quarter (predict the amount directly; 0 if none expected). Dollar amount for the target
quarter, SAME units/scale as the Q-4 baseline above (not a growth rate). Predict the value
directly; use 0 if none is expected. Sign matters (a loss/expense may be negative).
  income_taxes_abs: <float>  // income_taxes absolute dollar value for the target quarter (
below-pretax line; predict directly, 0 if none expected).. Dollar amount for the target
quarter, SAME units/scale as the Q-4 baseline above (not a growth rate). Predict the value
directly; use 0 if none is expected. Sign matters (a loss/expense may be negative).
  shares_abs: <float>  // shares: diluted weighted-average share count for the target quarter
(predict the count directly; drifts slowly with buybacks/issuance).. Dollar amount for the
target quarter, SAME units/scale as the Q-4 baseline above (not a growth rate). Predict the
value directly; use 0 if none is expected. Sign matters (a loss/expense may be negative).  [
GAAP history --- last_q=7460, ttm=7462, q4=7461]

=== UNCERTAINTY (additional required output) ===
After financial_calculate returns, you must ALSO state an 80% prediction interval for the GAAP
 diluted EPS it computed. Add ONE extra key to the final JSON, alongside the keys the system
prompt specifies:

  "eps_interval": { "p10": <float>, "p90": <float> }

Semantics: there should be a 10% chance the actual reported GAAP diluted EPS comes in BELOW
p10, and a 10% chance it comes in ABOVE p90.
The point forecast from financial_calculate stays exactly as computed --- do NOT revise it to
sit at the interval's midpoint, and do not call the calculator again.
Both under- and over-confidence are penalised: the interval is scored on how often the actual
lands inside it AND on its width, so a deliberately wide interval scores no better than a well
-judged one.
```