From Forecast Scores to Auditable Benchmarks: WorldFork for LLM Forecasting Evaluation

Anonymous Authors[1]

Abstract

Foundation-model forecasting benchmarks often report aggregate scores without specifying how uncertainty, leakage, endpoint semantics, or extraction choices affect whether a result should generalize. We introduce WorldFork as a benchmark-design case study for LLM forecasting agents: a public event card is converted into branching timelines with actor state, endpoint ledgers, path mass, unresolved mass, provenance, and a scoring-rule-compatible extraction rule. The central object is therefore not only a forecast probability, but an auditable record of how uncertainty moves through decomposition, branch policy, endpoint settlement, and report generation. On 24 masked retrospective resolved-event cards, unconditional branching reduces WorldFork Brier score from 0.282 to 0.214 and log score from 0.725 to 0.581; a fixed 50/50 blend with a direct JSON forecast reaches Brier 0.205. We treat these numbers as descriptive stress-test evidence, not a guarantee: retrospective masking only partially controls leakage, the exact sign test is suggestive but not significant ($p = 0.064$), the paired bootstrap interval includes zero, and multiple comparisons were explored. The contribution is a guarantee-oriented benchmark protocol that makes pre-registration, leakage audit, uncertainty composition, and trace-level failure analysis explicit for future locked evaluations.

1. Introduction

LLM forecasting is a useful stress test for foundation-model evaluation because it combines calibration, temporal reasoning, uncertain evidence, and real-world distribution shift.[1]

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

[1]https://anonymous.4open.science/r/worldfork-icml.

Yet a benchmark that reports only a mean Brier score leaves several evaluation questions underspecified. Did the model see resolution-adjacent information? Was the scored endpoint separated from auxiliary mechanism claims? How much probability mass remained unresolved? Which module or prompt choice moved the final probability? Without answers to these questions, an aggregate score is hard to interpret as evidence about generalization or reliability.

WorldFork reframes forecasting evaluation around an auditable benchmark object. A public event card is converted into actors and cohorts, constraints, binary endpoints, branch hypotheses, path masses, endpoint ledgers, and report provenance. The resulting trace exposes how uncertainty composes across the initialization, branch policy, endpoint-settlement rule, and probability-extraction rule. This makes the protocol compatible with proper scoring rules while preserving the process information needed for leakage audits, calibration diagnostics, and failure analysis.

This paper is written as a benchmark-methodology note. We contribute: (1) a trace-level forecast object that separates scored endpoint mass from unresolved and mechanism mass; (2) a reproducibility checklist for locked LLM forecasting evaluations; and (3) a 24-card retrospective pilot used as a stress test for the protocol. The empirical result is encouraging but deliberately non-confirmatory. Branching improves descriptive aggregate scores in the available artifacts, but the sample is small, retrospective masking is incomplete, and the observed branch losses show that unconditional branching is a benchmark design lever rather than a guaranteed improvement.

2. Benchmark Object

WorldFork operationalizes a forecast as a structured multiverse rollout with explicit endpoint semantics. A multiverse is a tree of timelines rooted in the same public scenario: each timeline records state, evidence, endpoint ledger, and path mass, while child timelines inherit the parent history up to the fork point and then evolve independently. Generated futures are treated as hypotheses, not observations. Appendix A gives the end-to-end process.

Decomposition. Initialization converts the public card into

typed actors and cohorts, scenario constraints, candidate endpoints, graph layers, and an initial social state. The benchmark distinction is that binary deadline endpoints are the scored targets, while auxiliary mechanism endpoints are audit targets. This prevents a mechanism claim, such as a regulator acting for a particular reason, from being silently mixed with the yes/no event label.

Uncertainty composition. Branch depth, active-timeline caps, branch thresholds, and unresolved-mass treatment are benchmark parameters. They determine how uncertainty propagates from local actor updates to path mass and then to the final binary probability. These parameters are useful because they create controlled hardness axes, but they also create degrees of freedom that must be locked before labels are opened.

Scoring extraction. Endpoint ledgers track whether evidence supports, contradicts, or leaves open each candidate outcome. For scoring, WorldFork converts path-level settlements into p_{yes} and reports unresolved mass separately. Brier is $(p_{\text{yes}} - y)^2$; log score uses the same binary probability clipped to $[0.01, 0.99]$. Unresolved mass can contribute neutrally during extraction, but it remains an audit diagnostic rather than a third scored class.

Trace audit. A final forecast report may contain mechanism claims about public attention, trust, institutional response, or coalition formation. WorldFork makes those claims auditable: a claim should point to a case, branch, tick, actor or cohort, source text, state update, confidence, and endpoint-ledger entry. This audit layer is not scored by Brier, but it exposes whether an apparently calibrated forecast is supported by the trace.

OASIS by CAMEL-AI is related background for LLM/rule-based social simulation; WorldFork's distinction is forecast-specific audit semantics: branch lineage, endpoint ledgers, path mass, unresolved mass, and deadline-aware scoring (Yang et al., 2024).

3. Evaluation Protocol

The benchmark contains 108 public cards: 72 synthetic stress prompts and 36 forecasting cards. The forecasting set contains 24 retrospective resolved forecast cards, 8 long-form dossiers, and 4 adversarial/calibration cards; only the 24 resolved cards are scored with proper scoring rules. Public inputs mask salient entity names and withhold private resolutions until predictions are frozen.

Leakage and labels. Retrospective evaluation creates forward-looking bias risk because resolved 2024–2026 events may appear in training data or retrieval corpora. Masking reduces exact-name matching but does not eliminate latent memorization or pattern recognition from public context. WorldFork decomposition limits some full-card exposure, but that design choice is not sufficient to justify a lower-leakage claim without future locked cards and pre-registered extraction rules.

Table 1. Benchmark checklist for locked LLM forecasting evaluations.

Field	Required record
Event card	As-of date, deadline, endpoint wording, public context, private label.
Pre-registration	Primary metric, baseline set, extraction rule, blend policy, statistical test.
Leakage audit	Masking rule, forbidden strings, auxiliary endpoint audit, rejected endpoints.
Run metadata	Provider, requested model, resolved snapshot, prompt version, response ID.
Uncertainty report	Brier/log score, calibration check, bootstrap interval, paired test, unresolved mass.
Endpoint hygiene	Binary settlement separated from mechanisms and open-ended closure.

Baselines and metadata. We compare uniform binary prediction, direct JSON API calls, an `ai-prophet` package baseline (AI Prophet Team, 2026), no-branch WorldFork, branching WorldFork, and a fixed blend. The WorldFork rows use a mixed model policy: DeepSeek v4 Flash for cohort agents and GPT-5.4-Codex/GPT-5.4 for initialization, global review, and other non-cohort roles. Because model routing, prompt versions, extraction rules, and blend policy can all move the result, a locked benchmark should record each of them before scoring.

Table 1 summarizes the CTB-oriented benchmark contract. The checklist converts broad concerns about reproducibility and guarantees into concrete fields: event-card provenance, pre-registered primary metric, leakage audit, model/run metadata, uncertainty report, and endpoint hygiene. The protocol does not prove a formal guarantee, but it makes clear which assumptions a future guarantee or confidence statement would have to condition on.

4. Results

All 108 public cards have live initialization evidence: 108/108 successful initializations, mean actor count 8.82, mean graph edge count 34.21, mean rubric score 3.884 on a 0–4 scale, and no critical failures. This structural proxy tests runtime coverage rather than forecast accuracy. Proper-score results are limited to the 24 resolved cards.

Paired branch/no-branch effect. Branching improves mean Brier from 0.282 to 0.214, giving $\Delta_{\text{Brier}} = 0.068$ and a 24.1% relative reduction. It has lower per-card Brier loss on 17 of 24 cards under an exact zero-difference comparison. The exact two-sided sign-test value is $p = 0.064$, and the paired card-bootstrap 95% interval is $[-0.065, 0.189]$. The result is directionally encouraging, but the interval includes zero and the test is not significant at 0.05.

Table 2. Resolved-card scores computed from the per-card probabilities in Appendix J. WF mixed denotes DeepSeek v4 Flash for cohort agents and GPT-5.4-Codex/GPT-5.4 for initialization, global review, and other non-cohort roles. Exploratory rows are not confirmatory comparisons.

Condition	Model	n	Brier	Log	Role
Uniform binary	fixed	24	0.250	0.693	reference
Direct JSON	GPT-5.4	24	0.236	0.677	baseline
Direct JSON	DeepSeek v4 Flash	24	0.260	0.741	exploratory
AI Prophet package	GPT-5.4	24	0.245	0.715	exploratory
No-branch WF	WF mixed	24	0.282	0.725	pilot
Branching WF	WF mixed	24	0.214	0.581	pilot
Fixed 50/50 blend	GPT-5.4+WF	24	0.205	0.527	exploratory

Trace-level uncertainty diagnostics. The seven no-branch lower-loss cards are not homogeneous. Two are numerical near-ties: cases 022 and 024 differ by only 1.68×10^{-5} and 9.22×10^{-5} in Brier loss. Four substantive losses share a common pattern: no-branch was already confident and correct, while branching diluted the endpoint probability toward uncertainty. In cases 006, 012, and 016, the true label is yes and no-branch assigns $p_{\text{yes}} = 0.9842$, 0.9233, and 1.0000, while branching assigns 0.4872, 0.2622, and 0.0680. In case 020, the true label is no and no-branch assigns $p_{\text{yes}} = 0.0972$, while branching assigns 0.4496. The branch trace localizes this as an uncertainty-composition failure rather than a generic score drop.

Complementarity and unresolved mass. Branching has lower reported Brier than direct JSON GPT-5.4, 0.214 versus 0.236, and a fixed 50/50 blend reports Brier 0.205. This blend is exploratory because multiple baselines and extraction choices were inspected. Branching also carries average unresolved mass 0.27, which is useful audit information but can hide weak endpoint settlement and pull a probability toward 0.5. A locked follow-up should therefore report unresolved mass next to proper scores and pre-register any blend or tie threshold.

5. Guarantee-Oriented Reporting

WorldFork does not provide a theorem that branching improves forecasting, but it makes the conditioning set for future guarantees explicit. A useful guarantee for a forecasting agent would need to state which event-card distribution, model route, extraction rule, leakage controls, endpoint semantics, and branch policy it assumes. The trace object gives each of those assumptions a logged representation. This is a practical step toward benchmark results that can be interpreted as controlled evidence rather than as one-off leaderboard numbers.

Instance-wise uncertainty. The natural unit of analysis is the card, not only the aggregate score. For each card, a locked report should include the final p_{yes}, unresolved mass, path-mass concentration, endpoint-settlement status, and whether the card falls under a pre-specified failure tag such as endpoint ambiguity, branch dilution, or leakage-adjacent auxiliary endpoints. These fields make it possible to ask whether high loss concentrates in identifiable benchmark regimes. They also support calibrated error bars at the instance or class level, for example by stratifying cards by deadline length, source type, branch depth, or unresolved-mass bucket before any outcome labels are inspected.

Module-wise uncertainty. Branching rollouts create a multi-stage system, so uncertainty can enter before the final probability is computed. Initialization may omit an actor or constraint, branch generation may allocate too much mass to speculative futures, endpoint ledgers may mix deadline settlement with open-ended simulation closure, and final reports may turn weak trace evidence into confident mechanism claims. Reporting only Brier score collapses these modules into one loss value. A CTB-style report should instead publish a small audit vector beside each score: initialization validity, endpoint hygiene, path-mass conservation, unresolved mass, extraction-rule version, and report-grounding status.

Controlled extrapolation. The 24-card pilot should be read as a development stress test for this audit vector. It suggests concrete hypotheses for a locked benchmark: adaptive branching may help when the no-branch trace has low endpoint support, while unconditional branching may hurt when the no-branch trace is already decisive and correct. That claim can be tested without changing the endpoint labels if the trigger rule is pre-registered from trace features alone. In this sense, the pilot identifies a benchmark design question rather than a deployment rule.

Reproducibility package. A complete CTB submission should make the reported score reproducible from frozen public cards, private labels, predictions, and extraction code. The minimum artifact is a per-card table with method label, model route, prompt version, response identifier, raw yes probability, extracted yes probability, unresolved mass, binary label, Brier loss, clipped log loss, and any excluded or rejected auxiliary endpoints. Such a table lets reviewers recompute aggregate scores, inspect practical ties, rerun bootstrap intervals, and verify that no post-label edit changed the extraction rule.

6. Discussion and Limitations

The central lesson is that a branching trace is not automatically a better forecast distribution. Proper scoring requires a clean probability over scored endpoints, not semantically

mixed ledger rows. The current result is modest: it improves over no-branch in the available per-card table, does not reach nominal sign-test significance, has a bootstrap interval that includes zero, and yields an exploratory blend improvement that is not a corrected multi-comparison result. For CTB-style evaluation, these caveats are part of the contribution because they expose which design choices must be fixed before performance claims become reliable.

The protocol suggests several benchmark-design requirements. Binary deadline settlement must be separate from open-ended simulation closure; scoring must be path-level; auxiliary mechanisms should not be scored as primary endpoints; child branches should inherit already-settled parent endpoints unless they fork before decisive evidence; and event summaries must preserve the official text needed to settle the card. Branch depth, branch threshold, active-timeline caps, and unresolved-mass handling should be treated as benchmark hardness parameters rather than implementation details.

Retrospective masking still cannot eliminate leakage risk. Future locked evaluations should release public cards before resolution, freeze predictions, pre-register extraction rules, pre-register any practical-tie threshold for paired lower-loss counts, report exact per-card predictions, and include paired confidence intervals plus a multiplicity plan. The next protocol should also report provider, requested model, resolved snapshot, prompt version, and response identifier for every scored call so model routing cannot become an untracked degree of freedom.

7. Conclusion

WorldFork is best framed as an auditable benchmark substrate, not as a finished guarantee of better forecasting. On 24 matched resolved cards, branching reduces WorldFork Brier from 0.282 to 0.214 and achieves log score 0.581, but the statistical evidence remains descriptive. The value of the protocol is that it makes uncertainty composition, leakage exposure, endpoint semantics, and mechanism claims inspectable. Future locked evaluations should test adaptive branching policies under pre-registered extraction and reporting rules.

Impact Statement

This paper studies auditable forecasting benchmarks. More transparent forecasting systems could help users inspect uncertainty, evidence, and assumptions before making decisions. The same systems could cause harm if retrospective pilot results are over-interpreted, if simulated branches are treated as evidence, or if scalar probabilities are used without domain expertise in high-stakes settings. We therefore emphasize leakage controls, disclosure of unresolved mass, grounding of mechanism claims, and human review. The work is intended as an evaluation protocol and research substrate, not as a deployed decision system.

References

Brier, G. W. Verification of forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3, 1950.

Gneiting, T. and Raftery, A. E. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378, 2007.

Tetlock, P. E. and Gardner, D. *Superforecasting: The Art and Science of Prediction*. Crown, 2015.

AI Prophet Team. `ai-prophet`: AI Prophet ecosystem CLI and Prophet Arena trade benchmark runner. PyPI package, 2026.

Yang, Z., Zhang, Z., Zheng, Z., Jiang, Y., Gan, Z., Wang, Z., Ling, Z., Chen, J., Ma, M., Dong, B., Gupta, P., Hu, S., Yin, Z., Li, G., Jia, X., Wang, L., Ghanem, B., Lu, H., Lu, C., Ouyang, W., Qiao, Y., Torr, P., and Shao, J. OASIS: Open Agent Social Interaction Simulations with One Million Agents. arXiv:2411.11581, 2024.

A. How WorldFork Works

WorldFork treats forecasting as constructing and scoring a tree of possible futures rather than asking a model for a single probability in one pass. The input is a public scenario card with an as-of date, a forecast deadline, and one or more candidate endpoints. Initialization creates the root scenario state, which we call the Big Bang: it stores the scenario text, forecast contract, actors and cohorts, constraints, graph and social variables, seed events, branch hypotheses, and an initial endpoint ledger. The first timeline in the multiverse starts from this root state.

The rollout then advances each active timeline through discrete ticks. During a tick, actor and cohort agents receive local state, propose decisions or events, and update social or institutional variables. A governance/review step audits the provisional tick bundle, applies endpoint-ledger updates, executes structured tool calls, and decides whether the timeline should continue, terminate, or fork. A fork creates a child timeline that inherits parent ticks, future queued events, actor/cohort state, graph context, and relevant prompt influences up to the fork point. After the fork, the child has its own executable state, so two timelines can preserve different hypotheses about the same unresolved event.

Branching is constrained by a policy rather than left open-ended. The policy caps branch depth, the number of active timelines, branches per tick, and branch-score thresholds. Each retained timeline carries path mass, which records how much probability remains attached to that timeline after splits and termination. At scoring time, endpoint ledgers answer whether each timeline supports yes, no, or unresolved for the binary forecast contract, while path mass determines how much of the retained multiverse supports each status. Unresolved mass is reported as an audit quantity and is handled neutrally in the binary probability extraction. Final reports compare terminal timelines, summarize the outcome distribution, and attach evidence so reviewers can inspect which branch assumptions and mechanism claims carried the forecast.

B. Reported Score Values

This appendix repeats the rounded score values used in Table 2. Values are computed from the per-card probabilities in Appendix J; exploratory rows are included for transparency rather than confirmatory testing.

C. Resolved-card Construction

Each resolved card was created from a real-world event between 2024 and 2026. We first wrote a public model-facing card containing only pre-resolution information, an as-of date, a deadline, and binary candidate endpoints. We then wrote a private evaluation object with the true resolution, resolution date, masked-entity map, and resolution source URLs. Entity names were masked before model input to reduce exact memorization and answer lookup. The private resolution file was not shown to forecasting systems until predictions were frozen.

Table 3. Rounded score values reported in Table 2. All scored rows use the same 24 resolved cards. Exploratory rows are reported for transparency but are not confirmatory tests.

Condition	Model	n	Brier	Log	Role
Uniform binary	fixed	24	0.250	0.693	reference
Direct JSON	GPT-5.4	24	0.236	0.677	baseline
Direct JSON	DeepSeek v4 Flash	24	0.260	0.741	exploratory
AI Prophet package	GPT-5.4	24	0.245	0.715	exploratory
No-branch WF	WF mixed	24	0.282	0.725	pilot
Branching WF	WF mixed	24	0.214	0.581	pilot
Fixed 50/50 blend	GPT-5.4+WF	24	0.205	0.527	exploratory

Table 4. Resolved-card construction summary, cases 001–012. Source/risk abbreviates the private resolution source class and leakage-risk label.

ID	y	As-of	Deadline	Domain; source/risk
001	yes	2025-09-15	2025-10-10	institutions/awards; Nobel/med
002	yes	2025-09-15	2025-10-08	science/awards; Nobel/med
003	yes	2025-11-15	2025-12-10	macro policy; Fed/low
004	no	2026-01-10	2026-01-28	macro policy; Fed/low
005	no	2026-01-23	2026-03-15	culture/awards; Academy/med
006	yes	2025-11-10	2026-02-01	culture/awards; Grammy/med
007	no	2025-10-24	2025-11-01	sports; MLB/low
008	no	2026-02-06	2026-02-08	sports; team/league/low
009	yes	2026-01-28	2026-02-01	sports; AusOpen/low
010	no	2026-01-29	2026-01-31	sports; AusOpen/low
011	yes	2025-10-01	2025-11-04	election; news/med
012	yes	2025-10-01	2025-11-04	election; AP/NPR/med

D. Statistical Checks

This section documents the scoring and statistical rules used for the rounded values in the main paper. All proper-score rows use the same 24 resolved forecast cards. Cards are not reweighted by topic, source type, or confidence. The added 36-card file is not a 36-case proper-scoring set: it contains 24 resolved forecast cards, 8 longform dossiers, and 4 adversarial/calibration cards. The latter 12 do not have binary yes/no resolutions and should not be added to the Brier/log-score denominator.

For a method m, the reported Brier score is

$$\mathrm{Brier}(m) = \frac{1}{24}\sum_{i=1}^{24}(p_{m,i} - y_i)^2, \tag{1}$$

where $p_{m,i}$ is the frozen yes probability for card i and $y_i = 1$ for a private yes resolution and $y_i = 0$ for a private no

Table 5. Resolved-card construction summary, cases 013–024. Public masked IDs and exact resolution URLs are stored in the private resolution-source artifact.

ID	y	As-of	Deadline	Domain; source/risk
013	no	2025-10-01	2025-11-04	election; AP/NPR/med
014	no	2025-04-01	2025-04-28	election; parliament/med
015	yes	2025-01-17	2025-06-30	product launch; company/med
016	yes	2025-09-10	2025-09-30	product launch; company/low
017	yes	2025-07-01	2025-10-14	tech policy; vendor/low
018	yes	2025-04-01	2025-04-30	regulation; regulator/med
019	yes	2025-07-25	2025-08-31	merger; company/low
020	no	2024-12-01	2024-12-31	antitrust merger; FTC/company/high
021	no	2026-02-20	2026-03-31	space launch; agency/low
022	no	2025-12-15	2026-01-31	climate record; NOAA/NCEI/low
023	no	2025-11-10	2025-11-22	climate negotiation; UN/low
024	no	2025-08-15	2025-09-30	antitrust remedy; court/news/low

resolution. The reported log score is

$$\mathrm{Log}(m) = \frac{1}{24}\sum_{i=1}^{24}\big[-y_i\log(\tilde{p}_{m,i}) - (1-y_i)\log(1-\tilde{p}_{m,i})\big], \quad (2)$$

where $\tilde{p}_{m,i} = \min(0.99, \max(0.01, p_{m,i}))$ is the clipped probability used to avoid infinite loss at exactly 0 or 1. The per-card table in Appendix J recomputes the rounded main values: no-branch WorldFork Brier/log $0.282/0.725$, branching WorldFork $0.214/0.581$, and fixed 50/50 blend $0.205/0.527$.

WorldFork prediction extraction first converts branch/path evidence into the binary probability consumed by these formulas. For candidate-status mass extraction, yes endpoint rows contribute realized mass to yes and eliminated mass to no; no endpoint rows contribute realized mass to no and eliminated mass to yes. If unresolved candidate mass remains, the extracted probability is

$$p_{\mathrm{yes}} = \frac{M_{\mathrm{yes}} + 0.5M_{\mathrm{unresolved}}}{M_{\mathrm{yes}} + M_{\mathrm{no}} + M_{\mathrm{unresolved}}}, \quad (3)$$

with $p_{\mathrm{yes}} = 0.5$ when no binary endpoint mass is available. For path-ledger extraction, each included path contributes its normalized weight times that path's extracted p_{yes}; uncertain paths use the same neutral 0.5 default unless a deadline-forced value is explicitly supplied. The resulting scalar p_{yes} is what Brier and log score use. The unresolved mass is retained in the reported predictions and tables as an audit quantity, not as an additional outcome class.

The paired branch/no-branch lower-loss count is computed per card from Brier losses: branching is lower-loss on card i if $(p_{\mathrm{branch},i} - y_i)^2 < (p_{\text{no-branch},i} - y_i)^2$. The primary descriptive count in this pilot uses an exact zero-difference comparison because no practical-equivalence threshold was pre-registered. In future runs, ties and practical ties should be excluded from the sign-test denominator or reported separately before computing the binomial test. In the current 24-card pilot there are 17 branching lower-loss cards out of 24 matched cards, giving an exact-comparison lower-loss rate of $17/24 = 70.8\%$.

The exact sign-test calculation treats the number of branching lower-loss cards L as $\mathrm{Binomial}(n = 24, q = 0.5)$ under the null hypothesis that branching and no-branch are equally likely to have lower loss on a card. The two-sided p-value for the observed favorable count is

$$p_{\mathrm{sign}} = 2\sum_{k=17}^{24}\binom{24}{k}2^{-24} = 0.063914656639, \quad (4)$$

reported as 0.064 after rounding. This is suggestive but not statistically significant at the conventional 0.05 level, so the result should be treated as an initial pilot evaluation rather than a final statistical claim.

Cases 022 and 024 illustrate why future runs should pre-register a practical-tie rule. Their no-branch advantages are 1.68×10^{-5} and 9.22×10^{-5} in Brier loss, respectively. Under a post-hoc practical-equivalence threshold of $\epsilon = 0.001$ in per-card Brier-loss difference, these two cards would be counted as ties, leaving 17 branching lower-loss cards among 22 non-tied comparisons and giving a two-sided sign-test value of $p = 0.017$. This sensitivity is diagnostic only and is not used as the primary statistical result.

Table 6. Diagnostic sensitivity of the paired lower-loss count to practical-tie thresholds. Thresholds were not pre-registered, so the exact zero-difference row remains the primary descriptive count.

ϵ	Branch	No-branch	Ties	p_{sign}
0	17	7	0	0.064
0.0001	17	5	2	0.017
0.001	17	5	2	0.017
0.01	15	5	4	0.041
0.05	12	5	7	0.143

A paired nonparametric bootstrap was computed by resampling the 24 matched cards with replacement and recomputing the mean per-card Brier difference Δ_{Brier} for each draw. Using the per-card probabilities in Appendix J, the observed mean difference is 0.0679 and the 95% percentile interval is $[-0.065, 0.189]$. The interval includes zero, reinforcing that the pilot is not a confirmatory result.

Multiplicity and analysis degrees of freedom. The benchmark development process considered several baselines, two proper scoring rules, multiple WorldFork extraction variants, and at least one blend. The paper therefore does not treat the smallest observed Brier value or the fixed-blend comparison as a multiplicity-corrected discovery. A locked follow-up should pre-register the primary endpoint, extraction rule, baseline set, blend policy if any, and statistical test before seeing private labels.

Leakage and endpoint-audit scope. Retrospective masking lowers exact-name exposure but does not eliminate

latent model memory or resolution-pattern recognition. WorldFork decomposition should be evaluated as a process design rather than a proven lower-leakage intervention. Auxiliary endpoints are useful for mechanism audit, but they can leak information if they name mechanisms that are only obvious after resolution. Future runs should audit auxiliary endpoints for resolution-adjacent wording before scoring and preserve the rejected endpoint list.

E. Post-hoc Branch-Loss Anatomy

This appendix classifies the seven cards on which no-branch has lower Brier loss under the exact comparison. The categories are descriptive and were assigned after inspecting the frozen probabilities in Appendix J; they should be treated as diagnostics rather than pre-registered subgroup results.

Table 7. No-branch lower-loss cards. Here $\Delta_i = (p_{\mathrm{NoWF},i} - y_i)^2 - (p_{\mathrm{BrWF},i} - y_i)^2$, so negative values favor no-branch.

Case	y	NoWF	BrWF	Δ_i	Interpretation
006	1	0.9842	0.4872	-0.263	Confident correct forecast diluted by branching
012	1	0.9233	0.2622	-0.538	Confident correct forecast diluted by branching
015	1	0.2530	0.1803	-0.114	Both methods assign minority mass to the true outcome; no-branch is less wrong
016	1	1.0000	0.0680	-0.869	Confident correct forecast diluted by branching
020	0	0.0972	0.4496	-0.193	Confident correct forecast diluted by branching
022	0	0.0079	0.0089	-1.68×10^{-5}	Numerical near-tie
024	0	0.0000	0.0096	-9.22×10^{-5}	Numerical near-tie

The substantive pattern is concentrated in cases 006, 012, 016, and 020. In those cards, the no-branch rollout is already on the correct side with high confidence, and branching pulls the final probability back toward uncertainty. Case 015 is a different pattern: both methods are on the wrong side of the binary outcome, with no-branch merely less wrong. Cases 022 and 024 are better treated as practical ties unless a future protocol pre-registers an exact zero-difference rule. This motivates an adaptive follow-up in which branching is triggered by uncertainty or weak ledger support rather than applied to every card.

F. Baseline Prompts and Run Settings

The direct baselines used one full public-card call per resolved case, no retrieval, temperature 0.2, max tokens 4096, and JSON-schema output validation. GPT-5.4 calls used the `openai-codex` provider route; DeepSeek calls reused the same prompt families through OpenRouter. The shared system prompt was:

```
You are a calibrated forecasting
assistant.
You are evaluating a
resolved-but-hidden event, but you
do not know the resolution.
Use only the public card text
provided by the user.
Do not use web search.
Do not infer from real-world memory
if the card is entity-masked.
Return valid JSON only.
```

The direct JSON user payload contained the public card, asked for one calibrated yes/no forecast and short rationale, and required an identifier, condition label, yes/no probabilities, confidence, main drivers, main uncertainties, and a leakage warning. The structured variant used the same public card but separated evidence for and against the event, base-rate analogies from the card only, key uncertainties, a calibration note, yes/no probabilities, and a leakage warning. Both prompt families enforced: probabilities sum to one, do not mention or imply the true resolution, and do not use web or private evaluation data.

For WorldFork short rollouts, the no-branch policy disabled branching by allowing one active timeline and using a high branch threshold of 0.999. The branching policy allowed depth 2, up to four active timelines, up to two branches per tick, a branch threshold of 0.75, at least two runway ticks, and candidate-endpoint branches only. Deadline-aware runs used the public as-of date and forecast deadline to set the tick duration and horizon.

G. Model and Run Identifiers

Exact model IDs are not uniformly locked across all rows. Table 9 records the identifiers available from the reported runs and the remaining reproducibility gap. Future locked evaluations should report provider, requested model, resolved model snapshot, route policy, prompt version, and response identifier for every scored call.

H. AI Prophet Baseline Details

The `ai-prophet` baseline used the package's FORECAST-stage style: an expert forecaster prompt returns a JSON yes probability and a short rationale. For this static benchmark, SEARCH and ACTION were disabled; each public masked resolved card was sent once to OpenAI GPT-5.4, and only the returned yes probability was scored. The run produced 24 predictions, averaged 3784.5 ms per call at concurrency four, and used 16,669 total tokens.

The artifact manifest records provider `openai`, requested model `gpt-5.4`, resolved model `gpt-5.4-2026-03-05`, 13,454 input tokens, 3,215 output tokens, and 16,669 total tokens. The package version and exact shell command were not captured in the manifest, so they remain reproducibility gaps. The adaptation was deliberately limited

Table 8. Prompt and run-setting summary for the reported forecast rows. WorldFork mixed route denotes DeepSeek v4 Flash for cohort work and GPT-5.4-Codex/GPT-5.4 for initialization, global review, endpoint-ledger, and report work.

Method	Provider / route	Model label	Temp.	Retrieval	Calls	Output schema
Direct JSON GPT-5.4	OpenAI Codex	GPT-5.4	0.2	no	24	direct yes/no JSON plus rationale
Structured direct GPT-5.4	OpenAI Codex	GPT-5.4	0.2	no	24	structured evidence plus yes probability
Structured direct DeepSeek	OpenRouter	DeepSeek v4 Flash	0.2	no	24	same structured schema
`ai-prophet` GPT-5.4	OpenAI	GPT-5.4 (2026-03-05)	default	no	24	FORECAST-stage JSON
WorldFork no-branch	WF mixed route	mixed	route	no	24 rollouts	endpoint-ledger/path-mass extraction
WorldFork branching	WF mixed route	mixed	route	no	24 rollouts	endpoint-ledger/path-mass extraction

Table 9. Available model and run identifiers. Rows without exact resolved snapshots should be treated as run labels, not immutable model definitions.

Row	Provider / route	Requested model	Resolved model string	Reproducibility note
Direct JSON GPT-5.4	OpenAI Codex	GPT-5.4	GPT-5.4	Snapshot date not available in the reported run label.
Direct JSON DeepSeek	OpenRouter	DeepSeek v4 Flash	deepseek-v4-flash-20260423	Resolved model string is available.
`ai-prophet` GPT-5.4	OpenAI	GPT-5.4	gpt-5.4-2026-03-05	Resolved model string is available.
WorldFork rollouts	OpenRouter + OpenAI Codex	DeepSeek v4 Flash cohorts; GPT-5.4-Codex non-cohort	WF mixed	Cohort agents used DeepSeek v4 Flash; initialization, global review, and other non-cohort roles used GPT-5.4-Codex/GPT-5.4.

Table 10. Leakage exposure summary for the reported rows. Masking reduces direct entity recall but cannot eliminate retrospective leakage.

Method	Full	IDs	Ret.	Local	Risk
Direct JSON GPT-5.4	yes	yes	no	no	med/high
Direct JSON DeepSeek	yes	yes	no	no	med/high
`ai-prophet` GPT-5.4	yes	yes	no	no	med/high
WF no-branch	decomp.	yes	no	yes	med
WF branching	decomp.	yes	no	yes	med

to the FORECAST stage: REVIEW, SEARCH, and ACTION were not used for the scored static cards, and the same Brier/log extraction consumed only the returned yes probability.

I. Leakage and Scoring Notes

All reported Brier scores use $(p_{\text{yes}} - y)^2$, where $y = 1$ for a private yes resolution and $y = 0$ for a private no resolution. Log score is $-\log(\tilde{p}_{\text{yes}})$ for yes labels and $-\log(1 - \tilde{p}_{\text{yes}})$ for no labels, where $\tilde{p}_{\text{yes}}$ is clipped to $[0.01, 0.99]$. For WorldFork rows, unresolved mass can contribute neutrally to the extracted binary probability, but it remains an audit quantity and should be reported separately from the binary score. The scoring loop is:

```
for each resolved case:
  read the frozen public prediction
  read the private label y in {0, 1}
  extract p_yes for the method
  compute Brier = (p_yes - y)^2
  compute clipped binary log score
  for WorldFork:
    settle each path as yes, no, or unclear
    p_yes = yes_mass + 0.5 * unclear_mass
    report unresolved_mass separately
```

J. Per-card Predictions

Table 11 reports the per-card probabilities available from the direct baseline, DeepSeek structured baseline, `ai-prophet`, and WorldFork prediction artifacts. The WorldFork columns are assembled from available terminal path-mass exports and are used for the consistency checks in this draft. Values are rounded to four decimals; exact-looking entries such as 0.0000 and 1.0000 should be read together with the clipped log-score rule in Appendix D. A locked follow-up should export this table automatically before labels are opened. The blend column is the fixed equal blend of the GPT-5.4 structured direct probability and the WorldFork branch probability in this table.

Table 11. Per-card probability table, cases 001–012. Direct is the structured GPT-5.4 direct row; DS is DeepSeek v4 Flash; Proph. is `ai-prophet`.

Case	y	Dir.	DS	Proph.	NoWF	BrWF	Blend
001	1	0.1035	0.1200	0.0800	0.2579	0.5591	0.3313
002	1	0.0709	0.2000	0.1200	0.2735	0.5579	0.3144
003	1	0.3854	0.3500	0.4200	0.2523	0.3630	0.3742
004	0	0.0151	0.3500	0.2800	0.7287	0.0001	0.0076
005	0	0.0133	0.3500	0.2400	0.1872	0.0007	0.0070
006	1	0.1616	0.1500	0.1600	0.9842	0.4872	0.3244
007	0	0.0659	0.4000	0.4400	0.1872	0.0000	0.0330
008	0	0.0857	0.5000	0.4900	0.3480	0.0002	0.0430
009	1	0.9776	0.5500	0.4300	0.2629	0.9989	0.9883
010	0	0.6473	0.6500	0.6100	0.7136	0.6407	0.6440
011	1	0.5177	0.6500	0.7200	0.2449	0.3103	0.4140
012	1	0.5717	0.3500	0.5800	0.9233	0.2622	0.4169

Table 12. Per-card probability table, cases 013–024. NoWF and BrWF are the no-branch and branching WorldFork probabilities.

Case	y	Dir.	DS	Proph.	NoWF	BrWF	Blend
013	0	0.0107	0.3500	0.4100	0.0475	0.0011	0.0059
014	0	0.8116	0.5500	0.6200	0.7262	0.5118	0.6617
015	1	0.4678	0.5500	0.5800	0.2530	0.1803	0.3240
016	1	0.8317	0.6500	0.8700	1.0000	0.0680	0.4499
017	1	0.9891	0.9500	0.9000	0.2622	0.9984	0.9938
018	1	0.0695	0.1500	0.2800	0.2652	0.5531	0.3113
019	1	0.5285	0.6500	0.7400	0.2522	0.3342	0.4314
020	0	0.0055	0.1500	0.1200	0.0972	0.4496	0.2275
021	0	0.0093	0.2500	0.3800	0.0345	0.0074	0.0083
022	0	0.0889	0.3500	0.3400	0.0079	0.0089	0.0489
023	0	0.0250	0.1500	0.1400	0.1872	0.0002	0.0126
024	0	0.0111	0.2000	0.2400	0.0000	0.0096	0.0103

K. Report and Audit Examples

These examples illustrate why audit metrics are reported separately from Brier. A forecast can be accurate but ungrounded, or inaccurate but diagnostically useful because the trace reveals which assumption failed.

Branching helps: resolved_001. The public card asks whether Candidate A wins a 2025 Nobel Prize. The true label is yes. In the available paired artifact, no-branch reports $p_{\text{yes}} = 0.2579$ while branching reports $p_{\text{yes}} = 0.5591$, reducing Brier by 0.3563. The useful branch behavior is not that the model knows the winner, but that alternate award-mechanism paths preserve more yes mass under the public source-packet evidence. A grounded report claim should cite the award endpoint, nomination/base-rate evidence, and the branch path that keeps the category-specific upset route alive; a claim that the result was already known before the deadline would be unsupported.

Branching hurts: resolved_016. The public card asks whether Phone A is available to customers by September 30, 2025. The true label is yes. The no-branch artifact reports $p_{\text{yes}} = 1.0000$, while the branching artifact reports $p_{\text{yes}} = 0.0680$. This is a useful audit failure: branching introduced or preserved too much contrary/uncertain mass despite a short-horizon company availability signal. Together with cases 006, 012, and 020, it illustrates the confident-correct dilution pattern in Appendix E. The trace should expose whether the error came from preorder-vs-availability confusion, over-weighted launch-risk branches, or missing authoritative source-packet settlement.

Blend helps: resolved_009. The public card asks whether Player C wins the 2026 Australian Open men's singles title. The true label is yes. The structured direct GPT-5.4 probability is 0.9776 and the WorldFork branch probability is 0.9989, yielding a fixed blend of 0.9883. This example shows complementarity: the scalar direct call is slightly more cautious, while branch mass gives more weight to a path in which the favorite/opponent-risk balance resolves positively. The report remains auditable only if the path summary and endpoint evidence state which tournament-path assumptions carried the additional yes mass.

Fixed-blend caveat. These examples are explanatory, not confirmatory. They were selected after inspecting the run artifacts, and the fixed blend was one of several explored comparisons. A locked follow-up should pre-register example-selection rules, primary metrics, and whether any blend is allowed before labels are opened.