Auditing Construct Validity in Agentic Decision Support with Sports Analytics Case Study

Pegah Rahimian
Department of Information Technology, Uppsala University
Uppsala, Sweden
pegah.rahimian@it.uu.se

Abstract

Most agent benchmarks ask whether a task finished. A growing class of agentic systems produces something different: an *analytical claim* for a human decision-maker – a clinical decision-support note, a portfolio risk attribution, a tactical analytics report for a football head coach. For these systems there is no oracle. A recent Brookings–CMU workshop recap with NIST authorship recommends measurement science – specifically *construct validity* – as a foundation for agentic-AI evaluation, but stops short of operationalising it. We propose TRACE, a four-threat audit framework that takes this step. The threats are construct misspecification, verdict unreliability under multiple judges, stochastic instability across re-runs, and regression under model upgrades; each is paired with a measurable audit instrument (APCR, IAJA, SRI, MERS). We instantiate TRACE on **PressureBench**, a 38-proposal seeded-error benchmark for football tactical analytics in which every anti-pattern is anchored to a published methodological error and four hard distractors guard against trivial pattern-matching. Four deployment-relevant findings emerge. (i) In our setting, capability tier was not a reliable proxy for audit precision: the smallest model in our panel (Haiku 4.5) reaches a false-positive rate of 0.17 (95% bootstrap CI $[0.00, 0.50]$) on valid anchors versus 0.67 $[0.33, 1.00]$ for Opus 4.7, and a clean non-overlap on hard distractors (0.00 vs 0.75). (ii) A single producer call is unsafe: only 17% of N=5 producer replication pairs receive the same audit verdict, and this rate *replicates exactly* under a different producer (Sonnet 4.6 also gives 0.17), showing the instability is not Opus-specific. (iii) Per-anti-pattern regression under model upgrade ranges from 0 (E1, evergreen) to 0.35 in F_1 (E3, regression-prone), motivating per-anti-pattern rather than global alert thresholds. (iv) An adversarial-producer stress test in which an Opus producer is instructed to commit each anti-pattern while avoiding obvious trigger words shows the audit pipeline still catches $10/12 = 83\%$ of the adversarial proposals. We are explicit about what TRACE catches and what it does not, and release the benchmark, prompts, judge outputs, event log, and experiment scripts. Total API spend for the full empirical evaluation is $10.90 across 276 LLM calls.

CCS Concepts

• **Computing methodologies** → **Artificial intelligence**; • **Information systems** → *Evaluation of retrieval results*; • **Theory of computation** → Reproducibility.

KDD-WS-AgenticEval '26, Jeju, Korea
2026.

Keywords

agentic AI, evaluation, trustworthiness, construct validity, industrial decision support, sports analytics, benchmark, LLM-as-judge, continuous monitoring

ACM Reference Format:
Pegah Rahimian. 2026. Auditing Construct Validity in Agentic Decision Support with Sports Analytics Case Study. In *Proceedings of KDD Workshop on Evaluation and Trustworthiness of Agentic AI (KDD-WS-AgenticEval '26).* ACM, New York, NY, USA, 11 pages.

1 Introduction

Most agentic-AI benchmarks ask the same question: did the task finish? WebArena [10] checks whether a web flow completed; SWE-bench [12] checks whether code passed tests; τ-bench [14] scores tool-agent-user interaction. The scoring signal is a verifiable outcome.

A different class of agentic system has no such outcome. The deliverable is an *analytical claim*: a clinical decision-support note for a physician, a portfolio risk attribution for a fund manager, a tactical analytics report for a football head coach. The agent's output asserts something about the world – "Our press collapses against opponents who play through the right half-space" – and the consumer must trust both the substance of the claim and the methodology behind it. There is no held-out test set and no ground-truth label. The right question is closer to "Would a domain expert defend this claim?"

Background and recent calls for measurement science. This setting has drawn attention from multiple directions. Producer-side frameworks such as Zimmer et al. [22] and the AI Scientist pipeline of Lu et al. [17] – the latter published in *Nature* and shown to pass workshop peer review at ICLR 2025 – demonstrate that an agent can autonomously generate scientific output, yet both reports list "lack of deep methodological rigour" and hallucination among the residual failure modes. Industry whitepapers inventory continuous-monitoring platforms and component-level checks [24]. Phiri [23] formalises an audit-log substrate via eight axioms for multi-agent integrity. Recent multidimensional taxonomies decompose evaluation into broad axes [25]. Most directly, a Brookings–CMU–NIST workshop recap [21] surveys the field and recommends that *measurement-science concepts – construct validity, content validity, predictive validity – be adapted to agentic AI rather than developing ad-hoc evaluation approaches.* None of the existing work operationalises construct validity end-to-end with a benchmark, a multi-judge calibration, a replication protocol, and a model-evolution monitor.

This paper. We take the operationalisation step that Tabassi et al. [21] recommend. Construct validity originates with Cronbach and Meehl [1] and was extended by Messick [2]; Jacobs and Wallach [3] adapted it for ML fairness. We adapt it one more step, for agentic AI outputs that are themselves measures. The framework, TRACE, identifies four threats specific to agent-emitted claims (Section 3.2) and pairs each with a measurable audit instrument. The measurement theory is not new; the operationalisation for agentic claims is.

We instantiate TRACE on a football case study and release the resulting benchmark, **PressureBench**[1]. The benchmark contains 38 analytical claims about defensive pressure. Twenty-four are seeded with one of six methodological anti-patterns drawn from the published football-analytics literature, six are valid baselines, four are *hard distractors* (surface signal of an anti-pattern but actually sound), and four are generated by Claude Opus 4.7 from neutral analyst-scenario prompts to mitigate the self-grading concern. PressureBench should be viewed as a diagnostic benchmark rather than a statistically representative sample of methodological failures. It is designed for diagnostic clarity rather than statistical power on its own.

Football analytics provides a useful stress-test because concepts such as pressure, compactness, and defensive effectiveness do not have universally accepted operational definitions, making construct-validity questions especially salient.

Contributions. (1) A four-threat audit framework adapting construct validity to agentic-AI outputs, with one measurable instrument per threat, operationalising the call in Tabassi et al. [21]. (2) **PressureBench**: a 38-proposal seeded-error benchmark in football tactical analytics where each anti-pattern is anchored to a published paper. (3) Four deployment-relevant findings: capability tier ≠ precision proxy (Haiku FP 0.17 vs Opus 0.67); a single producer call produced highly variable audit outcomes (only 17% of replication pairs received identical anti-pattern verdict sets, reproducing under a Sonnet producer); per-anti-pattern regression spans 0–0.35 in F_1; adversarial catch rate $10/12 = 83\%$. (4) A failure-analysis taxonomy (Section 6.7). (5) Public release of benchmark, prompts, judge outputs, event log, and scripts (https://github.com/Peggy4444/PressureBench-TRACE).

Scope. One domain (football), one data source (SkillCorner 2024–25). All models Anthropic Claude. The framework is producer-/detector-agnostic; cross-domain and cross-vendor are future work. We do not claim the specific detector implementations are state-of-the-art – the simple rule layer beats the LLM judge on four of six anti-patterns (Section 6.2). The contribution we defend is the measurement methodology and its empirical instantiation.

2 Related Work

We organise prior work by the question each line of work answers, then place TRACE in the resulting landscape (Table 1).

Task-completion benchmarks. WebArena [10], AgentBench [11], SWE-bench [12], GAIA [13] and τ-bench [14] score agents on closed-form outcomes. TRACE addresses the orthogonal setting with no oracle.

LLM-as-judge. Zheng et al. [15] introduced the protocol with MT-Bench; Bavaresco et al. [16] surveyed LLM judges across 20 NLP tasks. TRACE adds calibration against blinded human-expert verdicts (T2), addressing "judges agreeing on a wrong answer".

Agents producing scientific outputs. Lu et al. [17] report The AI Scientist, an end-to-end pipeline whose own Limitations list "lack of deep methodological rigour" and hallucination as residual modes. Zimmer et al. [22] provide a complementary *producer-side* framework of ten prompt-embedded commandments (e.g. "Never Fabricate Citations") that prevent failures inside the agent. ChemCrow [18] is an earlier system. These are about *what an agent can produce*; TRACE audits *what an agent has produced*, and is the complement these producers lack.

Audit substrates and industry validation. Phiri [23] formalises auditability for multi-agent systems via eight axioms with hash-linked event logs; TRACE's log (Section 3.4) targets Coverage, Temporal Coherence and Accessibility without cryptographic tamper-evidence. The Deloitte AI Assurance whitepaper [24] inventories two industry approaches (component-wise: Arize, Orq, IBM; assertion-based: Apple, AWS, NEC Labs) and monitoring platforms (Galileo, LangSmith, Arize Phoenix). None audits methodological validity of the analytical claim itself; TRACE is orthogonal to these.

Multidimensional and policy frameworks. Shukla [25] proposes a five-axis taxonomy with novel goal-drift and harm-reduction metrics. TRACE is narrower: four threats form a dependency chain (claim→verdict→run→version), not axes to balance. Tabassi et al. [21], the Brookings–CMU–NIST recap, explicitly recommends adapting measurement-science methods (construct, content, predictive validity) to agentic AI rather than ad-hoc evaluation. TRACE is one concrete operationalisation of that recommendation.

Reproducibility and model evolution. SRI composes three Jaccard signals (formula, citation, verdict) because construct validity requires the audited claim, not merely the surface text, to be stable. Chen et al. [20] documented ChatGPT temporal drift; MERS extends this into a deployment-time protocol. κ bands per Landis and Koch [19].

2.1 Anti-pattern anchors in football-analytics methodology

Each anti-pattern in PressureBench is anchored to a published football-analytics work that exhibits it: E1 [4], E2 [5], E3 [6], E4 [7], E5 [8], E6 [9]. Appendix A.8 gives the verbatim quote from each anchor showing the error.

2.2 Where TRACE sits

Table 1 places TRACE among the nine recognisable positions in the agentic-AI evaluation landscape we surveyed. TRACE occupies the cell at the intersection of empirical operationalisation and construct validity – the cell that the policy literature [21] identifies as currently unfilled.

[1] https://github.com/Peggy4444/PressureBench-TRACE

Table 1: Where TRACE sits in the agentic-AI evaluation landscape. Stance summarises what the work treats as the evaluation target. CV = construct validity (does the audited measure capture the concept it claims to measure?). The Brookings–CMU–NIST recap recommends construct-validity methods but does not operationalise them; TRACE is one concrete operationalisation.

Line of work	Stance / evaluation target	Empirical?	CV operationalised?
Task-completion benchmarks [10–14]	Did the task finish?	Yes	No
LLM-as-judge protocols [15, 16]	How do judges compare on NLP outputs?	Yes	No
Producer-side discipline [17, 18, 22]	Prevent failure inside the agent	Case studies	No
Audit-log axioms [23]	What is a valid audit log?	Formal proofs	No
Industry monitoring platforms [24]	Continuous component / trajectory health	Vendor reports	No
Multidimensional eval taxonomies [25]	Five axes to balance	Conceptual	No
Policy / research agendas [21]	What should be measured?	—	Recommended
Measurement theory [1–3]	Does the measure capture the concept?	Theoretical	Foundation
TRACE (this paper)	**Audit construct validity of agent-emitted analytical claims**	**Yes (PressureBench)**	**Yes**

3 TRACE: A Construct-Validity Audit Framework

This section defines the framework. We borrow construct-validity vocabulary from psychometrics [1–3] and operationalise it as machine-checkable audit instruments. The contribution is the operationalisation, not the concept.

3.1 Setting and unit of evaluation

A deployed system A receives a brief and returns a deliverable $D = \{c_1, \ldots, c_k\}$. The unit of TRACE evaluation is a single analytical claim c_i (e.g. a metric proposal with its construct and validation procedure).

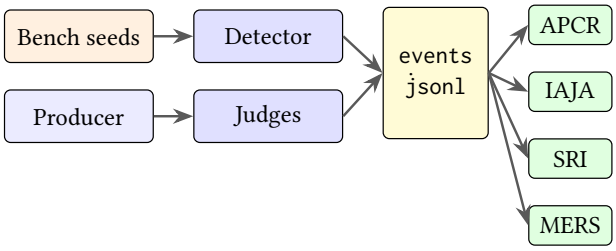

Figure 1: TRACE architecture. Producers emit claims; rule+judge detector and multi-judge panel write to one `events.jsonl`; four pillar readers compute APCR, IAJA, SRI, MERS. Concrete deployment in Figure 2.

3.2 Four threats to construct validity

T1 Construct misspecification. The agent operationalises the wrong target (e.g. provider label as supervised target, E1; pooling across structurally distinct contexts, E2/E6; wrong-confound normalisation, E5; inadequate-consequence validation, E3/E4).

T2 Verdict unreliability. Competent reviewers reach different verdicts on the same claim; a single judge is insufficient.

T3 Stochastic instability. Large variation across producer reruns may reduce confidence that the underlying construct has been operationalised consistently. If the agent proposes one construct on one run and a different one on the next, construct-validity assessment becomes more difficult and less reliable.

T4 Model-evolution regression. The audit pipeline that worked yesterday may not work after an upstream LLM upgrade.

3.3 Audit instruments

APCR (T1) – Anti-Pattern Catch Rate. For benchmark $\mathcal{B}$ with labels $a(c) \in \mathcal{A} \cup \{\text{VALID}\}$ and audit verdict $\hat{a}(c) \in 2^{\mathcal{A}}$, APCR $= |\{c : a(c) \neq \text{VALID} \wedge a(c) \in \hat{a}(c)\}| / |\{c : a(c) \neq \text{VALID}\}|$. Per-AP F_1 and FP rate on hard distractors are reported jointly. *Higher is better but read with FP rate;* carpet-flagging yields APCR= 1.00 trivially. Exact-set APCR $\hat{a}(c) = \{a(c)\}$ (Section 6.2) penalises over-flagging.

IAJA (T2) – Inter-Agent Judge Agreement. Panel $\mathcal{J}$ yields independent verdicts $\hat{a}_j(c)$. We report pairwise Cohen's κ, panel Fleiss' κ, and – when available – per-judge accuracy vs 2-expert consensus on a held-out subset. *Higher κ is better* (≥ 0.61 substantial per Landis & Koch); the expert comparison guards against "judges agreeing on a wrong answer".

SRI (T3) – Scientific Reproducibility Index. The producer is run $N = 5$ times per brief; SRI composes three pairwise Jaccards over $\binom{5}{2}$ pairs: formula-component on the proposed-metric vocabulary; cited-method on named methods; verdict-exact-match (fraction of pairs with identical audit verdicts). *Higher is better;* verdict-match below 0.5 flags an unsafe single producer call.

MERS (T4) – Model-Evolution Regression Score. A frozen subset is replayed under each model version. The matrix is $\text{MERS}_{m,m'}(a) = F_1^{(m)}(\mathcal{B}_a) - F_1^{(m')}(\mathcal{B}_a)$. *Lower* |MERS| *is better;* per-AP, not aggregate, is the operational quantity. Alerts fire when $|\text{MERS}_{m,m'}(a)| > \tau$ for any a.

3.4 Architecture and unified event log

The log is append-only and records every LLM call (model id, response id, token counts, prompt version, full raw response), judge verdict, rule trigger, claim generation, agent invocation, and experiment marker (Appendix A.2). Prompt and model-version tags allow replay of any past audit state – the load-bearing property

for MERS. Of Phiri [23]'s eight auditability axioms our log targets Coverage, Temporal Coherence and Accessibility; cryptographic tamper-evidence is out of scope and would be a natural hardening layer in a regulated deployment.

4 PressureBench

PressureBench is a benchmark of 38 analytical-claim proposals for a Tactical Analytics Report on defensive pressure. The stakeholder brief is "Where does our press break down?". The data substrate is SkillCorner 2024-25 Premier League tracking data (see Section 5). Each proposal reads like a one-page metric specification an analyst would put in front of a head coach.

4.1 Schema and anti-pattern taxonomy

Each proposal has five free-text fields (title; analyst-to-coach context; proposed metric with formula; claimed construct; claimed validation) plus typed metadata (seeded anti-pattern or VALID; hard distractor flag; expected catch; difficulty; provenance; literature anchors). Full schema in Appendix A.1.

The six anti-patterns: **E1** provider label as supervised target; **E2** missing defensive-block stratification (high/medium/low); **E3** turnover-only validation (immediate recovery only, ignoring delay, threat reduction, forced backward passes); **E4** case-study-only evidence; **E5** missing normalisation (raw counts without controlling for possession or build-up volume); **E6** missing phase conditioning (aggregating across structurally distinct opponent phases).

4.2 Composition

Table 2: PressureBench v1.1 composition. 38 proposals: 24 anti-pattern- seeded ($n = 4$ per E1–E6), 6 valid anchors, 4 hard distractors, 4 LLM-authored (UNLABELED; verdicted by independent expert in Section 6.3).

Ground truth	n
VALID anchors	6
VALID hard distractors	4
E1 Provider-label target	4
E2 Missing block stratification	4
E3 Turnover-only validation	4
E4 Case-study-only evidence	4
E5 Missing normalisation	4
E6 Missing phase conditioning	4
UNLABELED (LLM-authored)	4
Total	**38**

The four hard distractors are methodologically sound proposals that superficially resemble an anti-pattern. One mentions a provider label only as a sanity check (E1 surface). One presents case-study clips alongside systematic stratified analysis (E4 surface). One presents per-block league rankings with explicit refusal of cross-block comparisons (E2 surface). One presents a phase-weighted team rating with per-phase sub-scores (E6 surface). Each distractor is paired with a specific anti-pattern surface signal so we can measure per-surface false-positive rates.

4.3 Literature anchoring

Each anti-pattern is anchored to one published football-analytics work that commits the error (Section 2.1 for the full list with verbatim quotes). Two anchors are particularly strong: Forcher et al. [5] explicitly acknowledge the absence of block stratification as a "weakness of this study" in their own Discussion yet publish team-averaged pressure statistics anyway; Morgan [8] reports $r = 0.86$ correlation between his ranking metric and possession share, names the confound as "a likely circular relationship", and publishes the unnormalised league ranking anyway. The anchors guard against the reviewer concern that the seeded errors are strawmen.

4.4 Adversarial subset and release

To break the self-grading concern – the same author designing the framework and the proposals – four proposals (PB-017 to PB-020) are generated by Claude Opus 4.7 from neutral analyst-scenario specs with no mention of TRACE, evaluation, or the anti-pattern taxonomy. The four specs target distinct analyst pressures (player-level ranking, team-level headline, "adapt a published method", one-day quick metric). The generated proposals carry the UNLABELED ground-truth tag; labels are determined by independent expert verdict (Section 6.3). All generation events are logged for replay. The 38 proposals, the schema, the authoring scripts, the expert-validation protocol, and the literature-anchor file are released as supplementary material https://github.com/Peggy4444/PressureBench-TRACE).

5 Industrial Pipeline: Defensive Pressure Analytics

To ground TRACE in a concrete deployment we describe the football tactical-analytics pipeline that PressureBench evaluates. Figure 2 traces a single coaching brief through the producer, the four TRACE audit instruments, the event log, and the audit-annotated Tactical Analytics Report that reaches the head coach.

The pipeline runs on SkillCorner 2024–25 Premier League data: 378 matches with dynamic-event files (~300 columns covering on-ball engagement, pressing chains, inter-player distance and angle, passing options at event start/end, EPV, phase tags) plus 25 Hz tracking. Audit-rubric fields in Appendix A.3; coordinates are normalised so attacking direction is uniform across matches. Given a brief, the producer emits a claim conforming to the `SeededProposal` schema (Appendix A.1); in our evaluation it is a single Opus 4.7 call, but a deployment can substitute multi-agent debate, RAG, or a human-AI draft. The Tactical Analytics Report has a fixed six-part structure (executive summary; per-block E2 audit; per-phase E6 audit; player appendix with E5 warnings; recommendations; explicit non-claims). Every claim is annotated with audit verdict, verdict-exact-match across N=5 replications, and event-log pointer. Flagged claims are not dropped silently; they appear with the AP flag and a remediation note. T4 runs weekly on a frozen subset.

6 Empirical Evaluation

We instantiate TRACE on PressureBench and report results for each of the four audit instruments. Section 6.7 then steps back from the numbers to walk through specific failures the framework catches – and several it does not.

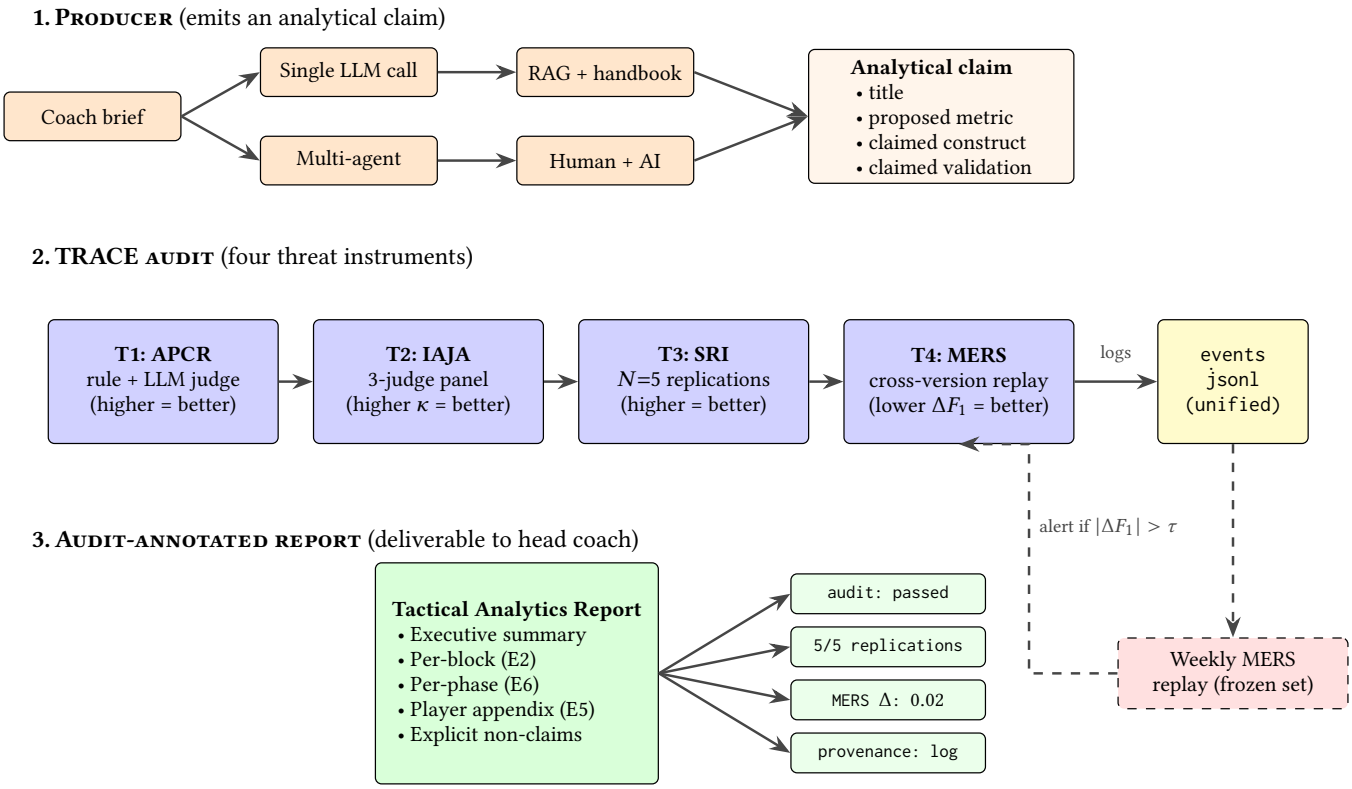

Figure 2: TRACE-instrumented industrial pipeline. The producer (top) emits an analytical claim; four audit instruments (middle) check T1–T4 and write every event to `events.jsonl`; the audit-annotated Tactical Analytics Report (bottom) carries per-claim metadata to the head coach. A weekly MERS replay on a frozen subset fires per-AP alerts. Direction-of-merit (higher / lower = better) is defined in Section 3.3.

Four findings at a glance. (i) *Capability tier ≠ precision proxy:* Haiku 4.5 reaches FP = 0.17 on anchors and 0.00 on distractors; Opus 4.7 reaches 0.67 and 0.75 (CI non-overlap on distractors). (ii) *Single producer call is unsafe:* 17% verdict-match across N=5, reproducing under a Sonnet producer. (iii) *Per-AP regression varies sharply:* ΔF_1 from 0 (E1) to 0.35 (E3). (iv) 83% *adversarial catch rate* (10/12) under evasive producer prompting. Deployment implications in Section 7.2.

6.1 Setup and ablations

The APCR ablation uses four detector modes: *baseline* (Sonnet 4.6 with a generic "is this sound?" prompt; taxonomy hidden); *rule-only* (six deterministic keyword checkers, no LLM); *judge-only* (Sonnet 4.6 with the full six-anti-pattern rubric); and *full* (rule + judge, disjunctively combined). The judge sees only the five free-text fields of each proposal; metadata is hidden. Every API call is recorded in the unified event log so any verdict can be replayed.

6.2 T1: Anti-Pattern Catch Rate

On 34 labelled proposals (Table 3), strict APCR = 1.00 saturates across non-rule conditions – the criterion "ground-truth AP ∈ detected set" is trivially satisfied by broad flagging. The exact-set criterion separates them: baseline collapses from 1.00 to 0.00 (carpet-flagging), `rule_only` reaches the highest exact-match rate (0.39) by abstaining when uncertain, and `full` is worst on both metrics. The rule layer beats the LLM judge on E1 (F_1 = 0.89), E4 (0.75), E5/E6 (0.67 each); the judge wins on E2 (0.38) and E3 (0.50) where negation contexts defeat the rules (per-AP table in Appendix A.4). The per-anti-pattern F_1 in Table 10 contains the more interesting result: at n = 4 per anti-pattern the simple rule layer beats the LLM judge on four of six anti-patterns (E1, E4, E5, E6) because the judge over-flags and accumulates false positives faster than the rule layer's recall deficit costs it. The judge keeps its F_1 advantage on E2 and E3, the two anti-patterns where the rule layer fails on negation contexts ("regardless of phase") and signal cross-talk. The OR- combined `full` pipeline is not best on any cell: recall is already saturated by the judge so OR can only add false positives, and on four of six anti-patterns `full` has the lowest F_1. This is a deliberate negative finding – disjunctive combination is not a free recall enhancer – and we recommend per-anti-pattern dominant-layer selection (Section 7.2).

6.3 T2: Inter-Agent Judge Agreement

The three-judge panel (Sonnet 4.6, Opus 4.7, Haiku 4.5) reaches substantial agreement on E1 (Fleiss κ = 0.89, Appendix A.6) and

Table 3: APCR aggregate results on labelled PressureBench v1.1 ($n = 34$). Strict APCR saturates at 1.00 for any condition that flags broadly; exact-set APCR ($\hat{a}(c) = \{a(c)\}$ on the AP subset) and exact-match labelled (detected set equals ground-truth set on the full labelled subset, empty for VALID) penalise over-flagging. Judge model: Sonnet 4.6.

Condition	Strict APCR	Exact-set APCR	Exact-match labelled	FP anchors	FP distractors
baseline	1.00	0.00	0.05	1.00	0.50
rule_only	0.67	0.29	**0.39**	**0.00**	0.50
judge_only	1.00	0.29	0.29	0.67	0.50
full (rule ∨ judge)	1.00	0.21	0.21	0.67	0.75

Table 4: Per-judge precision on PressureBench v1.1 (n_{ap}=24, n_{anchors}=6, $n_{\text{distractors}}$=4). FP rates carry 95% bootstrap CIs (proposal-id resampling, n_{boot}=2000); CIs are wide because per-group n is small. Strict APCR (recall) saturates at 1.00 across judges. Per-AP Fleiss κ in Appendix A.6; human-expert calibration in Appendix A.7.

Judge	APCR	FP anchors	FP distractors
Sonnet 4.6	1.00	0.67 [0.33, 1.00]	0.50 [0.00, 1.00]
Opus 4.7	1.00	0.67 [0.33, 1.00]	0.75 [0.25, 1.00]
Haiku 4.5	1.00	**0.17** [0.00, 0.50]	**0.00** [0.00, 0.00]

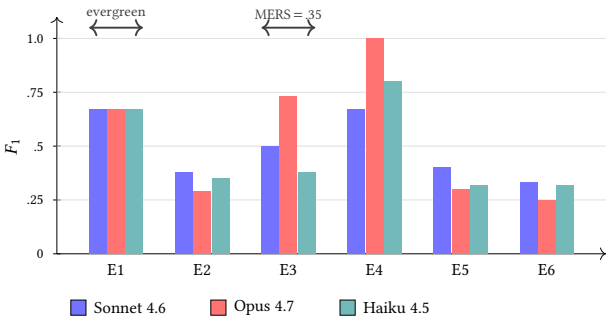

Figure 3: Per-AP F_1 across the three-model panel. MERS is the max–min ΔF_1 per AP. E1 is evergreen (MERS= 0.00); E3 spans the widest range (MERS= 0.35): the Haiku→Opus upgrade gains 35pp on the E3 audit. E2, E5, E6 sit at uniformly modest F_1; their small MERS reflects shared weakness, not stability.

moderate on the rest, lowest on E3 ($\kappa = 0.44$). All three judges hit strict APCR 1.00 (Table 4), but Haiku 4.5 – the smallest – has the lowest FP on anchors (0.17) and a perfect 0.00 on hard distractors, while Opus 4.7 has the worst FP profile on distractors (0.75). The finding held at $n = 16$ in v1.0 and reproduces at $n = 34$ in v1.1. With 95% bootstrap CIs (Table 4), the hard-distractor comparison is unambiguous: Haiku 0.00 vs Opus 0.75 $[0.25, 1.00]$ do not overlap.

Two experts trained in the same methodology verdicted a 16-proposal blinded subset $\mathcal{B}_H$ (Appendix A.7): recall vs the AND-consensus is 0.87–0.93; precision vs OR is 0.30–0.33 (judges over-flag). No judge dominates across all APs (Haiku best on E1/E6, Sonnet on E3/E5, Opus on E4), supporting per-AP judge selection.

6.4 T3: Scientific Reproducibility Index

For each of four specs we run Opus 4.7 $N = 5$ times. Aggregate formula-component Jaccard across 40 within-spec pairs is 0.62; cited-method 0.58; verdict-exact-match only 0.17. The producer draws from a consistent vocabulary but combines elements in methodologically different ways across runs. Spec openness predicts verdict variability: "adapt a published method" has verdict-match 0.00; constrained "player-level metric" has 0.30. A side observation: E1 appears in 12/20 verdict sets and E5 in 13/20 – Opus is a flawed analyst by default, which empirically justifies the audit layer. At $N = 1$, 5 of 6 replications would receive a different verdict; we recommend $N \geq 5$ with aggregation.

Cross-producer generalisation. To check that 0.17 is not Opus-specific, we re-ran SRI with Sonnet 4.6 as producer (20 producer + 20 judge calls). Aggregate verdict-exact-match is 0.17 – identical to Opus 4.7. Formula-component Jaccard drops to 0.48 (vs 0.62 Opus, Sonnet draws from a wider vocabulary). Per-spec breakdown in Appendix A.10; the stochastic-instability finding generalises.

6.5 T4: Model-Evolution Regression Score

Strict APCR is invariant across models (all 1.00); MERS variance lives in precision (Figure 3). E1 is evergreen ($\Delta F_1 = 0.00$); E3 is most regression-prone (Haiku 0.38 → Opus 0.73, a 35-pp gain). E2, E5, E6 sit at $\Delta F_1 \in [0.08, 0.10]$ – shared weakness, not stability. A monitoring deployment should use per-AP thresholds: τ=0.20 on our data flags E3 on Sonnet→Opus and E4 on Sonnet→Haiku, while remaining silent on E1.

6.6 Adversarial-producer stress test

To test the audit against an evasive producer, we prompted Opus 4.7 to design proposals that *structurally commit* each anti-pattern while *linguistically avoiding* obvious trigger words (e.g. for E1, ban "provider label" and "circular"; for E2, ban "block" and "stratification"). Two replications per AP gave 12 adversarial proposals; Sonnet 4.6 then verdicted each (Table 6). The audit catches 10/12 = 83%; E1–E3, E6 caught 2/2; one miss each on E4 and E5 – both flaws of omission (no exposure normalisation; no stratified validation), consistent with the per-AP F_1 pattern. We do not claim adversarial *robustness* (iterative red-teaming against a fixed judge would likely improve the producer); 83% catch under single-shot evasion is nonetheless a stronger self-grading defence than the honest LLM-authored subset alone.

Table 5: SRI results across 4 specs and $N = 5$ replications each ($\binom{5}{2} = 10$ pairs per spec). Mean Jaccard similarity is reported on the formula-component vocabulary (SkillCorner column names plus quantitative-term tokens) and the cited-method vocabulary (named published methods and author surnames). Verdict-exact-match rate is the fraction of pairs where the audit pipeline returns the same detected anti-pattern set. Producer: `claude-opus-4-7`; judge: `claude-sonnet-4-6`.

Spec	Formula J	Citation J	Verdict J	Verdict exact
S017 (player-level)	0.70	0.35	0.53	0.30
S018 (team headline)	0.65	0.87	0.24	0.10
S019 (adapt published)	0.61	0.60	0.13	0.00
S020 (one-day)	0.53	0.52	0.72	0.30
Aggregate	0.62	0.58	0.40	0.17

Table 6: Adversarial-producer audit catch rate. Per anti-pattern, 2 proposals are generated by claude-opus-4-7 with the explicit instruction to commit the named structural flaw while avoiding the obvious surface trigger words a methodological-validity auditor keys on. The audit pipeline (claude-sonnet-4-6 judge) is then asked to verdict the proposals. Catch rate is the fraction of adversarial proposals on which the seeded AP is in the detected set. Overall: 10/12 = 0.83. Wide CIs apply at this n.

AP	Caught / total	Catch rate
E1	2/2	1.00
E2	2/2	1.00
E3	2/2	1.00
E4	1/2	0.50
E5	1/2	0.50
E6	2/2	1.00

6.7 Failure analysis: what TRACE catches and misses

Of eight failure modes we mapped against TRACE (Appendix A.9), three are caught (data leakage → E1; missing normalisation → E5; missing block / phase pooling → E2/E6); two are partially caught (overconfident claims via E2/E5; invalid statistical inference overlapping E3); three require taxonomy extensions left to future work (citation fabrication → bibliographic verification; tactical hallucinations and fake domain concepts → ontological grounding). The most informative concrete case is data leakage: in SRI, 12/20 Opus 4.7 replications train on a SkillCorner provider label, and PB-019 ("adapt a published method") explicitly adapts Bauer and Anzer [4] – recapitulating the published E1 flaw rather than questioning it. Both rule and judge catch this. Conversely, citation fabrication is addressed on the producer side by Commandment III of Zimmer et al. [22]; on the audit side a Semantic-Scholar / OpenAlex check is the natural extension.

6.8 Reproducibility, cost, and artifact release

Every LLM call across the pipeline is recorded in `events.jsonl` with model id, response id, token counts, prompt version, and full raw response. Each experiment script writes a `results.json` and a LaTeX table fragment imported above. A reviewer can replay any LLM call by querying the log for its response identifier, or re-run an entire experiment by filtering the log on event type and purpose tag.

Cost. The complete evaluation cost $10.90 over 276 LLM calls; a Haiku-only judge pass on PressureBench fits under $0.20 (reinforcing R1). Per-role breakdown in Appendix A.11. All artefacts released as supplementary material (https://github.com/Peggy4444/PressureBench-TRACE).

7 Discussion

7.1 Limitations

The atomic unit is an analytical claim, not the deliverable as a whole or the agent's task completion. Construct validity is not new [1–3]; our contribution is the operationalisation for agentic-AI outputs (four-threat pipeline + deployment shape). Whether four threats are *sufficient* is empirical: the failure analysis in Section 6.7 flags tactical hallucinations, citation fabrication and fake domain concepts as requiring extensions (ontological grounding, bibliographic verification). PressureBench v1.1 has $n = 4$ per AP, four hard distractors, six valid anchors; per-AP point estimates are preliminary, and v1.2 ($n \geq 8$ per AP) is the natural next release. All evidence is in football analytics; cross-domain and cross-vendor (GPT, Gemini, open-weights) generalisation is future work. Human ground truth is $|\mathcal{B}_H| = 16$ proposals, small for inter-rater studies [19].

The self-grading defence rests on four moves: literature-anchored anti-patterns, an LLM-authored honest subset (PB-017–PB-020), the expert-validation subset, and the adversarial-producer stress test catching 10/12 = 83%. None is individually decisive at our n; together they shift the burden of proof.

Relation to continuous-monitoring platforms. Platforms inventoried in Deloitte [24] (Galileo, LangSmith, Arize Phoenix; Apple, AWS, NEC Labs assertion-based; Arize, Orq, IBM component-wise) focus on component health and trajectory adherence. TRACE is not a competitor; it addresses construct validity of the claim itself. We expect production deployments to use both, with Phiri [23]'s log substrate underneath when regulated.

7.2 Deployment recommendations

Five recommendations follow from the empirical results. (R1) *Choose judges by precision profile, not capability tier.* Haiku 4.5 is approximately 15× cheaper than Opus 4.7 and beats it on precision (FP 0.17 vs 0.67 on anchors; 0.00 vs 0.75 on distractors). Profile candidate judges on a domain-grounded distractor subset. (R2) *Pick*

per-AP layer; do not OR-combine. The `full` disjunctive pipeline has the lowest F_1 on four of six APs because OR-combining cannot lift saturated recall and only adds false positives. Use rules for E1, E4, E5, E6; the LLM judge for E2, E3. (R3) *Replicate before trusting.* Replicate $N \geq 5$ producer calls and aggregate audit verdicts (flag an AP when at least $\lceil N/2 \rceil$ of N replications flag it). At $N = 1$, 5/6 of replications would give a different verdict. (R4) *Monitor per-AP; alert per-AP.* A single global τ either under-alerts on E3 or over-alerts on E1. Use per-AP thresholds; the unified event log makes every claim replay-able. (R5) *Adopt in a new domain via three artefacts:* a literature- anchored taxonomy, a PressureBench-shape benchmark (valid anchors + anti-pattern seeds + hard distractors), and a `produce(spec)` implementation. The TRACE code, event-log schema, and per-pillar instruments transfer unchanged; only the benchmark must be rebuilt.

8 Conclusion

Agentic AI systems that produce analytical claims for industrial stakeholders need a different evaluation paradigm from task-completion benchmarks. TRACE adapts construct validity from psychometrics as four threats with corresponding audit instruments; PressureBench provides a literature-anchored seeded-error benchmark in football tactical analytics to instantiate them.

Four findings from running TRACE on PressureBench have direct deployment consequences. First, capability tier is not a precision proxy: Haiku 4.5 reaches FP= 0.17 on valid anchors and 0.00 on hard distractors while matching perfect strict recall, so judge selection should be driven by domain calibration rather than capability tier. Second, only 17% of N=5 producer replication pairs receive the same audit verdict from the same brief, and the same 0.17 rate *reproduces under a different producer* (Sonnet 4.6), so a single producer call is not sufficient evidence of construct validity; we recommend $N \geq 5$ with majority-vote aggregation. Third, per-anti-pattern regression under model upgrade ranges from $\Delta F_1 = 0.00$ (E1, evergreen) to 0.35 (E3, regression-prone), motivating per-anti-pattern alert thresholds in continuous monitoring. Fourth, the audit pipeline catches $10/12 = 83\%$ of adversarial proposals designed by an Opus producer instructed to evade detection.

The framework is producer-agnostic; the integration contract (Section 5) is a one-method `produce(spec)`. The instantiations of TRACE that interest us next are in clinical decision support and financial risk attribution – both share the analytical-claim deployment pattern and admit construction of analogous seeded-error benchmarks from their methodological literatures. Several common LLM failure modes – tactical hallucinations, citation fabrication, fake domain concepts – sit outside the current six-anti-pattern taxonomy and are partly addressed on the producer side by frameworks such as Zimmer et al. [22]; on the audit side, we sketch extensions in Section 6.7 and leave their implementation to follow-up work. Tabassi et al. [21] describe agentic-AI evaluation as at a "Wright Brothers" stage; TRACE is one operational instrument among the many a maturing measurement science for agentic AI will require.

References

[1] L. J. Cronbach, P. E. Meehl (1955). Construct validity in psychological tests. *Psychological Bulletin*, 52(4), 281–302. DOI: 10.1037/h0040957.

[2] S. Messick (1995). Validity of psychological assessment: Validation of inferences from persons' responses and performances as scientific inquiry into score meaning. *American Psychologist*, 50(9), 741–749. DOI: 10.1037/0003-066X.50.9.741.

[3] A. Z. Jacobs, H. Wallach (2021). Measurement and Fairness. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency (FAccT)*, 375–385. DOI: 10.1145/3442188.3445901.

[4] P. Bauer, G. Anzer (2021). Data-driven detection of counterpressing in professional football: a supervised machine learning task based on synchronized positional and event data with expert-based feature extraction. *Data Mining and Knowledge Discovery*, 35(5), 2009–2049. DOI: 10.1007/s10618-021-00763-7.

[5] L. Forcher et al. (2022). The keys of pressing to gain the ball – Characteristics of defensive pressure in elite soccer using tracking data. *Science and Medicine in Football*, 8(2), 161–169. DOI: 10.1080/24733938.2022.2158213.

[6] S. Merckx et al. (2021). Measuring the Effectiveness of Pressing in Soccer. In *Machine Learning and Data Mining for Sports Analytics (MLSA 2021), ECML/PKDD Workshop*. URL: https://people.cs.kuleuven.be/ pieter.robberechts/repo/merckx-mlsa21-pressing.pdf.

[7] J. M. Calabuig et al. (2024). A Mathematical Model to Study Defensive Metrics in Football: Individual, Collective and Game Pressures. *Mathematics*, 12(23), 3854. DOI: 10.3390/math12233854.

[8] W. Morgan (2018). How StatsBomb Data Helps Measure Counter-Pressing. StatsBomb Blog, May 20, 2018. URL: https://statsbomb.com/2018/05/how-statsbomb-data-helps-measure-counter-pressing/.

[9] G. Andrienko et al. (2017). Visual Analysis of Pressure in Football. *Data Mining and Knowledge Discovery*, 31(6), 1793–1839. DOI: 10.1007/s10618-017-0513-2.

[10] S. Zhou et al. (2024). WebArena: A Realistic Web Environment for Building Autonomous Agents. In *Proceedings of the 12th International Conference on Learning Representations (ICLR)*.

[11] X. Liu et al. (2024). AgentBench: Evaluating LLMs as Agents. In *ICLR*.

[12] C. E. Jimenez et al. (2024). SWE-bench: Can Language Models Resolve Real-World GitHub Issues?. In *ICLR*.

[13] G. Mialon et al. (2023). GAIA: A Benchmark for General AI Assistants. *arXiv preprint arXiv:2311.12983*.

[14] S. Yao et al. (2024). τ-bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains. *arXiv preprint arXiv:2406.12045*.

[15] L. Zheng et al. (2023). Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. In *NeurIPS Datasets and Benchmarks Track*.

[16] A. Bavaresco et al. (2024). LLMs instead of Human Judges? A Large Scale Empirical Study across 20 NLP Evaluation Tasks. *arXiv preprint arXiv:2406.18403*.

[17] C. Lu et al. (2026). Towards end-to-end automation of AI research. *Nature*, 651, 914–919. DOI: 10.1038/s41586-026-10265-5.

[18] A. M. Bran et al. (2023). ChemCrow: Augmenting Large-Language Models with Chemistry Tools. *arXiv preprint arXiv:2304.05376*.

[19] J. R. Landis, G. G. Koch (1977). The measurement of observer agreement for categorical data. *Biometrics*, 33(1), 159–174. DOI: 10.2307/2529310.

[20] L. Chen, M. Zaharia, J. Zou (2023). How is ChatGPT's behavior changing over time?. *arXiv preprint arXiv:2307.09009*.

[21] E. Tabassi et al. (2026). How can we best evaluate agentic AI? A Brookings–Carnegie Mellon University workshop recap. Brookings Institution, Research Brief. URL: https://www.brookings.edu/articles/how-can-we-best-evaluate-agentic-ai/.

[22] M. Zimmer et al. (2026). The Agentic Researcher: A Practical Guide to AI-Assisted Research in Mathematics and Machine Learning. arXiv preprint arXiv:2603.15914.

[23] C. C. Phiri (2025). Creating Characteristically Auditable Agentic AI Systems. In *Proceedings of Intelligent Robotics (IntRob '25)*. DOI: 10.1145/3759355.3759356.

[24] Deloitte (2025). From hype to control: Validating Agentic AI. Deloitte LLP, AI Assurance Whitepaper.

[25] M. A. Shukla (2025). Evaluating Agentic AI Systems: A Balanced Framework for Performance, Robustness, Safety and Beyond. Preprints.org, 202508.1847.v1. DOI: 10.20944/preprints202508.1847.v1.

A Schema and Vocabulary Reference

This appendix provides the structured reference material for the schema, the event vocabulary, and the data column inventory referenced from Sections 3–6. The paper is self-contained and does not require consulting this appendix to understand the empirical findings.

A.1 Proposal Schema

Each PressureBench proposal is a typed record (Table 7) with five free-text content fields and seven typed metadata fields. Authoring-time validation enforces internal consistency: for instance, a hard

distractor must have ground truth VALID, and a proposal whose ground truth is an anti-pattern must carry at least one literature reference.

Table 7: Proposal record fields. Content fields are free text in analyst-to-coach register; metadata fields are typed.

Field	Role
Content (free text).	
title	one-line metric label
context	analyst-coach scenario, 1–2 sentences
proposed metric	formula and computation procedure
claimed construct	what the analyst says the metric measures
claimed validation	validation procedure the analyst proposes
Metadata (typed).	
ground truth anti-pattern	one of {VALID, E1, …, E6, UNLABELED}
is hard distractor	boolean; only VALID can be `True`
expected catch explanation	what a competent reviewer should say
difficulty	one of {easy, medium, hard}
proposal source	human-authored or LLM-generated
literature references	published anchor(s) per anti-pattern
notes	free-text authoring notes

A.2 Event Log Vocabulary

All cross-cutting events in the TRACE pipeline are appended to a single append-only event log (Table 8). Each record carries a UTC timestamp, an event-type tag, and event-specific fields. The log is the load-bearing property for MERS replay and reproducibility.

Table 8: Event vocabulary. Every record additionally carries a UTC timestamp and an event-type tag.

Event type	Key payload fields
LLM call	model id, response id, token counts, prompt version, prompt hash, stop reason, purpose tag, full raw response
Judge verdict	proposal id, model id, detected anti-patterns, confidence, overall assessment, justification
Rule trigger	proposal id, rule id, triggered (Boolean), confidence, matched evidence
Proposal generation	proposal id, model id, spec text, full raw response
Agent invocation	agent id, tool name, input/output summary, duration
Experiment marker	experiment id, phase (start/checkpoint/complete), notes

A.3 SkillCorner Column Inventory

The football pipeline reads SkillCorner 2024-25 Premier League dynamic event data (378 matches). The columns directly referenced by the audit rubric and the rule-layer detectors are listed in Table 9. Full column documentation is in the SkillCorner data dictionary; we list only the columns the audit pipeline reads.

Table 9: SkillCorner columns referenced by the audit pipeline. Grouped by semantic role.

Role	Columns
Event identity	on-ball-engagement, pressing-chain, pressing-chain length, pressing-chain end type, simultaneous-defensive-engagement same target
Spatial / proximity	interplayer distance (start/end/min), angle of engagement, separation start/end, separation gain, x/y start/end, channel start/end, third start/end
Phase tags	team-in-possession phase type (build-up, create, finish, transition, quick-break), team-out-of-possession phase type (high-block, medium-block, low-block, defending transition, defending quick-break)
Passing options	dangerous-difficult options count, dangerous-not-difficult options count, line-break options count, last-line-break options count, ahead-at-start/end options counts
Defensive shape	last defensive line height (start/end), delta to last defensive line, defensive structure, organised defense, number of defensive lines, inside defensive shape (start/end)
Outcome / danger	EPV start/end, EPV possession start/end, stop possession danger, reduce possession danger, forced backward, beaten by movement/possession, possession danger, xLoss, xShot, first/second-last/last/furthest line break, push/break defensive line, ball recovery

A.4 APCR per-anti-pattern F_1

Table 10 shows Per-AP F_1 on labelled PressureBench v1.1 across the three APCR conditions.

Table 10: Per-AP F_1 on labelled PressureBench v1.1 across the three APCR conditions; baseline omitted (it does not name specific APs).

AP	rule	judge	full
E1	**0.89**	0.67	0.62
E2	0.25	**0.38**	0.33
E3	0.29	**0.50**	0.50
E4	**0.75**	0.67	0.62
E5	**0.67**	0.40	0.38
E6	**0.67**	0.33	0.33

A.5 MERS numerical table

Numerical companion to Figure 3, with bootstrap CIs.

Table 11: MERS: per-AP F_1 per judge model with 95% bootstrap CIs (proposal-id resampling, n_{boot}=2000); ΔF_1 is the max–min across models. *Lower* ΔF_1 *is better*. E1 is evergreen ($\Delta = 0.00$); E3 is regression-prone ($\Delta = 0.35$). Numerical companion to Figure 3.

AP	Sonnet 4.6	Opus 4.7	Haiku 4.5	ΔF_1
E1	0.67 [.25, .92]	0.67 [.25, .92]	0.67 [.25, .93]	**0.00**
E2	0.38 [.11, .62]	0.29 [.08, .50]	0.35 [.10, .58]	0.10
E3	0.50 [.17, .77]	0.73 [.33, 1.0]	0.38 [.11, .63]	**0.35**
E4	0.67 [.22, .92]	1.00 [1.0, 1.0]	0.80 [.33, 1.0]	0.33
E5	0.40 [.10, .64]	0.30 [.08, .50]	0.32 [.08, .53]	0.10
E6	0.33 [.09, .57]	0.25 [.07, .44]	0.32 [.08, .55]	0.08

A.6 Per-AP inter-judge κ matrix

The compact per-judge Table 4 in Section 6.3 reports judge precision. The full per-anti-pattern pairwise Cohen's κ and panel-wide Fleiss' κ are in Table 12.

Table 12: Per-anti-pattern Cohen's κ and Fleiss' κ across the three-judge panel on PressureBench v1.1 ($n = 34$). 95% CIs via 1000-resample bootstrap over proposals. Bands per Landis & Koch (1977).

AP	Son–Op	Son–Ha	Op–Ha	Fleiss
E1	1.00	0.84	0.84	0.89
E2	0.59	0.76	0.69	0.68
E3	0.64	0.47	0.29	0.44
E4	0.60	0.64	0.77	0.66
E5	0.48	0.59	0.74	0.60
E6	0.34	0.69	0.38	0.46

A.7 Human-expert calibration of LLM judges

Two domain experts trained in the same methodology independently verdicted a $|\mathcal{B}_H| = 16$ subset; they agreed on 100% of flag sets, so we treat their verdicts as a single methodology- consistent consensus. Judges' recall vs the AND-consensus is 0.87–0.93; precision vs the OR-consensus is 0.30–0.33 (over- flagging). Per-AP F_1 vs consensus echoes the cross-judge pattern.

A.8 Anti-pattern literature anchors

Each anti-pattern in PressureBench v1.1 is anchored to a published football-analytics work that exhibits the error. The anchors guard against the strawman concern: the methodological flaws TRACE catches are not invented for the benchmark.

E1 (provider label as supervised target): Bauer and Anzer [4] trains a counterpressing classifier on 20,000+ human-annotated labels. **E2** (missing block stratification): Forcher et al. [5] pool 153 Bundesliga matches in a single mixed model and acknowledge the limitation in their own Discussion ("weakness of this study"). **E3** (turnover-only validation): Merckx et al. [6] Equation (1) defines pressing effectiveness as a function of two binary outcomes (recovery, shot conceded) by construction. **E4** (case-study-only evidence): Calabuig et al. [7] empirical section is titled "A Real Case Study" on two matches. **E5** (missing normalisation): Morgan [8] reports $r = 0.86$ confound between his ranking metric and possession share, names this "a likely circular relationship", and publishes the unnormalised ranking anyway. **E6** (missing phase conditioning): Andrienko et al. [9] pools 31,827 "successful pressure positions" across all opponent possession phases without phase conditioning.

Table 13: Human-expert calibration on $\mathcal{B}_H$ ($n = 16$). (a) Aggregate metrics; (b) Per-AP F_1 vs the 2-expert AND-consensus.

(a) Aggregate.

Judge	Exact match	Recall (AND)	Precision (OR)
Sonnet 4.6	0.25	0.93	0.33
Opus 4.7	0.12	0.93	0.30
Haiku 4.5	0.25	0.87	0.30

(b) Per-AP F_1 vs 2-expert AND-consensus.

AP	Sonnet 4.6	Opus 4.7	Haiku 4.5
E1	0.57	0.67	**0.80**
E2	0.20	**0.31**	0.18
E3	**0.60**	0.57	0.36
E4	0.67	**1.00**	0.80
E5	**0.67**	0.53	0.53
E6	0.31	0.25	**0.33**

A.9 Failure-mode coverage

Table 14: Failure-mode coverage by TRACE. "Caught" = at least one of rule or LLM judge reliably flags the mode in our data. "Partial" = sometimes caught, unreliable. "Future work" = not addressed by the current taxonomy.

Failure mode	Anchor / extension	Status
Data leakage (provider label)	E1	Caught
Missing normalisation	E5	Caught
Missing block / phase pooling	E2, E6	Caught
Overconfident tactical claims	via E2/E5	Partial
Invalid statistical inference	overlaps E3	Partial
Citation fabrication	bibliographic check	Future
Tactical hallucinations	ontological grounding	Future
Fake domain concepts	ontological grounding	Future

A.10 Cross-producer SRI breakdown

Per-spec results for the Sonnet 4.6 producer (Section 6.4), for comparison with the Opus producer.

A.11 API cost breakdown

Table 16 summarized Total API spend on the TRACE empirical evaluation, by experiment role and model.

A.12 Reproducibility and code release

PressureBench seeds, the authoring scripts, the judge prompts, the detector code, the unified event log, the four experiment scripts,

Table 15: Cross-producer SRI: same protocol as Table 5 but with claude-sonnet-4-6 as the producer instead of Opus 4.7. The aggregate verdict-exact-match rate is 0.17, for comparison with Opus 4.7's 0.17. The 'single producer call is unsafe' finding is therefore not Opus-specific: a different producer exhibits the same instability shape, though point estimates differ.

Spec	Formula J	Citation J	Verdict J	Verdict exact
S017 (player-level)	0.54	0.73	0.15	0.10
S018 (team headline)	0.44	0.51	0.23	0.10
S019 (adapt published)	0.49	0.50	0.47	0.20
S020 (one-day)	0.44	0.60	0.68	0.30
Aggregate	0.48	0.59	0.38	0.17

Table 16: Total API spend on the TRACE empirical evaluation, by experiment role and model. Token counts come from the unified event log (every LLM call carries model id, response id and token counts). Pricing: Opus 4.7 $15/M input, $75/M output; Sonnet 4.6 $3/M, $15/M; Haiku 4.5 $0.80/M, $4/M (2026-06). Total spend: $10.90 over 276 LLM calls (389K input + 236K output tokens). We report this so that future deployments can size the audit overhead for a comparable benchmark.

Role	Model	Calls	USD
Judge panel (APCR, IAJA, SRI, MERS)	Opus 4.7	38	$3.11
Exp 5 adversarial producer	Opus 4.7	24	$2.51
Producer (SRI, MERS, LLM-authored)	Opus 4.7	22	$2.20
Judge panel (APCR, IAJA, SRI, MERS)	Sonnet 4.6	90	$1.87
proposal_generation_legacy	Opus 4.7	5	$0.49
Producer (SRI, MERS, LLM-authored)	Sonnet 4.6	20	$0.28
APCR baseline judge	Sonnet 4.6	38	$0.26
Judge panel (APCR, IAJA, SRI, MERS)	Haiku 4.5	38	$0.16
judge_taxonomy_legacy	Sonnet 4.6	1	$0.02
Total		**276**	**$10.90**

the expert-validation protocol, and the literature-anchor file are released as a supplementary repository (https://github.com/Peggy4444/PressureBench-TRACE). Each LLM call carries a model-response identifier and a prompt version that lets reviewers replay any single verdict. Each experiment script writes machine-readable results and a LaTeX table fragment that is imported by the corresponding subsection of Section 6.