

Can AI-Agent Benchmarks Favor Firms over Consumers?

Joy Zheyun Yang*

Georgetown University

Abstract

Agent benchmarks reward whatever their reference answers encode, even when those answers enforce company policies that restrict consumer rights. These restrictive policies are hidden in the grading criteria, only affecting an agent’s score once it is capable enough to complete the other steps of the task. We examine τ^3 -Banking, a benchmark reported on several public leaderboards [1, 15, 23] where developers test AI agents in a simulated fictional bank. Although the bank is fictional, it invokes real consumer-protection law, specifically Regulation E (a consumer-protection rule for electronic transfers). We test three scenarios where the benchmark expects agent actions that give less protection than Regulation E under our stated assumptions by 1) leaving a reported claim unfiled, 2) assigning contested liability, and 3) misclassifying a lost PIN as voluntarily shared. When we take a passing answer and change just one decision to an alternative the regulation supports under our stated assumptions, the benchmark changes the grade from a pass to a fail. This penalty remains hidden in industry leaderboards because today’s models fail other parts of the task first. However, looking at the agents’ raw actions shows higher-scoring submissions leaving the capped claims unfiled more often, a pattern consistent with closer adherence to the restrictive policy. A benchmark can therefore demand bank-friendly decisions long before models are advanced enough for those choices to impact their final scores. Developers should audit a benchmark’s internal grading criteria before trusting it as a valid measure of customer service. Note that because we targeted these selected scenarios to test this mechanism, our findings highlight a hidden structural vulnerability not necessarily a general bias across the dataset.

1 Introduction

Imagine a banking agent receiving the simple request that a customer wants the bank to investigate a \$275 transfer she does not recognize. The agent is capable of filing the claim, and the banking software allows it. However, doing so will cost the agent its score on the benchmark. The reason is that the fictional bank’s internal policy limits customers to two open disputes. Because an earlier claim already occupies one slot, and the benchmark’s reference answer prioritizes a larger transaction for the second, the \$275 claim exceeds the cap. If the agent completes every other step of the workflow but decides to file this third claim anyway, the benchmark’s grader turns what would have been a passing result into a failure. There is an obvious defense for this design. Benchmarks in the τ -bench family test whether AI agents can follow company policy while handling user requests [25]. If a customer asks for something a company does not permit, refusing that request is what the benchmark is supposed to reward. Our concern begins where the fictional bank invokes real law. The bank’s dispute guide instructs agents to follow Regulation E, but then enforces claim-filing

*Code and replication materials accompany this paper at <https://github.com/JOYY/tau3-policy-as-reward>.

caps that the actual regulation does not impose. Testing policy adherence is a valid goal. But a benchmark that presents a policy as an application of consumer-protection law rewards agents for enforcing legally questionable limits, and makes a claim about whether that policy gets the law right.

This matters because benchmark scores are published and compared. Organizations like Sierra publish submissions from developers, including model providers [23], Artificial Analysis runs its own banking evaluation and leaderboard [1], and Induction compares inference cost and pass rates with and without its context-management system [15]. These scores are useful for measuring an agent’s ability to navigate tools and retrieve information, but they are dangerous to use as proof that a model is ready for regulated customer service. Before deploying these models, we must ask, what is the agent being forced to do to earn that high score?

We examine this question in two ways. First, we audit the benchmark’s knowledge base. Out of 698 bank documents, we analyze the four that explicitly cite Regulation E, categorizing their rules based on whether they match the law, offer extra protections, or give less protection than Regulation E. Second, we conduct targeted tests using three specific dispute scenarios to see how these rules impact an agent’s score. Two scenarios involve the cap on disputes, while the reference answer in the third records a written PIN kept in a lost wallet as "voluntary sharing."

Our findings demonstrate that this benchmark penalizes alternatives the regulation supports under our stated assumptions in these selected cases. This penalty remains temporarily hidden on industry leaderboards due to models failing other tasks. An analysis of raw agent actions shows that higher-scoring submissions leave the capped claims unfilled more often. We detail these mechanisms and their implications across the following sections. Section 3 explains the document sweep, legal comparisons, and grading tests. Section 4 moves from policy findings to scores and agent behavior, and Section 5 considers what maintainers can fix before these requirements begin to move the leaderboard.

2 Background and related work

2.1 Banking agents and consumer protection

The Consumer Financial Protection Bureau highlighted in June 2023 how banking chatbots often provide inaccurate answers and create barriers to exercising consumer rights [7]. Benchmark designers face the same problem. An AI agent may apply a company’s policy efficiently even when that policy leaves out a customer’s rights. To understand this tension, we look to actual consumer law. Regulation E treats coverage, liability, and investigations as separate legal questions. For example, liability depends heavily on whether a device was lost or stolen and when the bank was notified [11]. The official regulatory commentary distinguishes between simple customer negligence (like writing a PIN on paper kept with the card) and voluntarily granting someone else authority to use an account [10, 11]. The benchmark translates these complex legal questions into simple database fields. A generated dispute record indicates the agent filed a claim, while a "maximum-liability" field records the maximum amount the customer could have to pay. In our study, we use Regulation E to populate these fields with alternatives the regulation supports under our stated assumptions, which we call the alternative records, and then test how the benchmark’s grading criteria evaluate them.

2.2 Research on legal compliance and benchmark quality

The emerging field of law-following AI explores how agents can serve their principals within the bounds of the law [20]. In the context of evaluation, legal compliance must start with the benchmark’s

reference answers. Designers must ensure that the bank policy they reward aligns with the consumer law it cites.

Prior work has examined policy loopholes, ambiguous instructions, or agent behavior under European law. Cao and colleagues study whether agents follow required procedures [4], Cao’s *Policy Loopholes* examines ambiguous policies in two τ -bench domains and argues that evaluation quality depends on the quality of the policy specification [3], and EU-Agent-Bench builds scenarios that test agent behavior under EU law [17]. Here, we specifically compare a benchmark’s internal policy against the external legal standards it claims to enforce.

We also build on literature identifying structural flaws in AI evaluations. Northcutt, Athalye, and Mueller show that test-set errors can affect model comparisons [19], Gema and colleagues identify errors in MMLU and show how correcting them changes evaluation [14], and Zhu and colleagues’ Agentic Benchmark Checklist addresses errors in task setup, reward design, and reporting [26]. Because agent benchmarks often grade an entire sequence of actions or a final database state, a single restrictive policy embedded deep in the reference answer can cause an otherwise passing response to fail.

Uncovering these mechanics requires open access to the benchmark’s underlying code. Casper and colleagues explain why output-only testing cannot support many important audit questions [5]. Access to the benchmark’s code lets us replay the answer key, change one record, and check the score. Incident-analysis work identifies the records that investigators need to understand agent failures [13]. We use that approach to examine the benchmark itself.

2.3 Prior work found missing information for liability decisions

Our analysis builds on an earlier public bug report that identified a structural issue in the benchmark. Its grading criteria demand precise liability calculations that are impossible to derive from the provided information. Specifically, the report examines a customer request involving several disputed debit-card transactions. For each claim, the agent must record the maximum amount the customer could have to pay, and the bank’s own guidance calculates that amount using the date of the customer’s account statement, but neither the task nor the tools supply that date. The benchmark nevertheless requires specific amounts for a passing score, so an agent could follow the bank’s written rule and still disagree with the expected answer because it must guess a missing fact [2].¹

The report then examined 110 publicly available attempts by agents to complete this customer request, and none of them had passed the whole test. Ten attempts matched the liability amounts but failed other grading criteria, which proves that the missing statement date is not the sole reason models currently score zero on this task [2].² That prior report established a mechanical inconsistency between the bank policy and the available facts. Our work investigates a deeper legal inconsistency. We expand the analysis to examine reported claims the benchmark leaves unfilled, and cases where a lost PIN is recorded as voluntarily shared. By testing controlled changes against the grading criteria, we isolate which contested records an agent must produce to earn a passing score, and whether fixing those decisions would alter current rankings.

¹The report appears as issue 592 in the benchmark’s GitHub repository. It focuses on task 086 and also flags five other dispute tasks that use the same liability rule, tasks 082, 084, 085, 087, and 088. These numbers identify customer scenarios in the benchmark.

²The report excludes one additional attempt that ended in an infrastructure failure. Our separately collected archive contains one more scored attempt at this task and also finds no passes.

3 Methodology

Our methodology bridges legal analysis and computational evaluation. First, we compare the fictional bank’s internal policy against actual consumer law. Next, we use controlled tests to determine whether the benchmark’s grading criteria force agents to enforce restrictive policies to earn a passing score. We analyze public leaderboard submissions to see how these grading requirements impact model behavior in practice.

3.1 Benchmark data and case selection

We audit benchmark release v1.0.1, which contains 97 tasks and 698 policy documents [24]. After inventorying all 955 reference actions, we focused on eight tasks containing 23 expected debit-dispute filings. Rather than sampling these tasks randomly, we deliberately selected three core scenarios, two involving caps on disputes and one involving a lost wallet. This purposive selection allows us to directly test the specific mechanisms by which a reference answer might conflate a restrictive bank policy with legal compliance. For the text-based rules agents read before acting, we conducted a separate document sweep as a distinct unit of analysis.

We evaluate the individual task files loaded by the benchmark rather than its aggregated `tasks.json` file, which contains discrepancies across 13 tasks. The underlying banking data, tools, and grading criteria align with the repository snapshot used to freeze our leaderboard metadata. We downloaded 32 of the 33 advertised public submissions (one archive returned an HTTP 404 error), retaining all source URLs and file hashes for full reproducibility (detailed in the Appendix).

Throughout our analysis, we evaluate "submissions" rather than just base models. A submission represents a specific pairing of a model with its evaluation setup. This matters because retrieval settings and environmental conditions can alter an agent’s behavior, even if the underlying model remains the same. Each archived run represents a single attempt at a given task.

Before downloading the agent trajectories, we documented our selected cases, alternative grading rules, and sampling seeds in a local analysis plan. Because our case selection was informed by prior exposure, this plan is exploratory rather than confirmatory. Following our initial results, we expanded our analysis to include trial-count restrictions, tool-sequence checks, date verifications, and the full policy document sweep. We preserved all archives and documented any schema repairs, infrastructure exclusions, or score discrepancies. We validated our testing pipeline, which covers scoring, record extraction, and tool-call ordering, by comparing 60 controlled replays against the benchmark’s upstream evaluator, and we investigated 15 disputed scores.

3.2 A complete sweep of documents that cite Regulation E

To understand the knowledge base provided to the agents, we systematically audited the fictional bank’s policy documents. We searched the entire document library for explicit references to consumer law ("Regulation E," "Reg E," or section "1005"), isolating four key guides. We reviewed these selected documents comprehensively, covering every line to capture rules that omit direct legal citations but operate within the covered context (such as the dispute cap).

We divided these documents into distinct rules, splitting compound statements whenever different clauses required separate legal comparisons. A coverage check accounted for every non-empty line, including headings, and we grouped interface definitions and routine security instructions into operational blocks. Repeated rules across different documents were treated as separate occurrences. For each unit, our public ledger records the exact source text, line numbers, legal provision, interpretation, and direction of the comparison.

To evaluate these documents, we first separated introductory and operational instructions from substantive claims. We then compared each substantive rule against actual consumer law, specifically relying on regulatory text, official commentary, and CFPB guidance, to classify its directional impact [6, 8–11]. A rule is classified as offering *less protection* than Regulation E if it imposes an explicit exclusion, an added barrier, or an excessive liability amount or deadline. Conversely, we record rules that offer *more protection* or remain *consistent* with the law. If a statement leaves its trigger, scope, or direction unresolved, such as a clock error that could benefit either party, or a missing qualification with an unknown practical effect, we classify it as *unclear*.

These labels describe the isolated legal comparison of a specific clause, rather than weighing the overall benefit or harm to a customer. For example, a bank policy might promise advance warning of a credit reversal (offering *more protection*) while simultaneously omitting another required protection in the same paragraph. We evaluate these claims independently.

We conducted this comprehensive sweep after our initial targeted experiments. This sequence broadens our evidence of the bank’s policy environment without retroactively turning our three core scenarios into a random sample. We identified an edge case in a separate customer guide stating that a police report may be required for some claims, while CFPB guidance bars requiring a police report before starting an investigation [6]. We retain this comparison as contextual evidence of restrictive bank policy, but exclude it from the formal sweep’s counts.

3.3 Controlled tests of the grader

To determine whether the benchmark enforces the restrictive bank policies found in its text, we traced the credit guide’s two clearest errors directly into the implementation. We inspected all reference debit-dispute filings and their resulting credit records for a liability deduction, searched every task’s grading criteria, and inspected the filing tool for an investigation deadline. This step distinguishes a poorly written bank policy in the knowledge base from a restrictive requirement that the benchmark’s grader enforces.

Next, we conducted controlled replays to isolate what the benchmark rewards. We first executed the benchmark’s reference answers in the official environment to secure a passing grade. We then modified a single decision to match the alternative records. Specifically, we filed an omitted claim, lowered a liability amount, or changed the lost-wallet PIN classification. Keeping the initialization and all other reference actions fixed, we fed these modified answers back through the benchmark to see if this single change caused the grading criteria to fail the agent.

To understand why these failures occur, we must look at how the benchmark evaluates success. For most tasks, the grader compares the agent’s final database and tool-use logs against those produced by the reference answer. (Nine tasks instead verify whether the agent made specific expected calls, and one combines a database comparison with a language judgment). Although the database grader tolerates minor formatting differences and certain read operations, an extra write operation, such as filing an omitted claim, causes a strict mismatch. We ran controls to verify the grader’s precision, confirming that it accepts equivalent numeric formats but correctly rejects substantive changes to unrelated fields, such as a police-report record.

3.4 Alternative grading rules

We then designed alternative grading criteria to test what would happen to public leaderboards if the benchmark accepted the alternative records in the two capped-claim scenarios and the lost-wallet scenario. To earn a passing score under these alternative criteria, an agent must still complete the conversation and match every unaffected field in the reference answer. Equation 1 defines this

combination.

$$a_{it} = \mathbf{1}\{N_{it}\} \mathbf{1}\{P_t(D_{it})\} \mathbf{1}\{C_t(D_{it}) = C_t(G_t)\}. \quad (1)$$

Here D_{it} is the database the agent leaves after run i on task t , and G_t is the reference database. The grader first verifies a normal conversation ending N_{it} , then validates our alternative records P_t . It ensures all other records match the reference database after removing the specific differences we permit, C_t . Because the agent and user tools share a single database, one comparison suffices. Premature stops earn a zero score, while infrastructure failures are excluded from the scoring denominator.

For the capped-claim scenarios, our alternative requires the agent to file the omitted transaction and record an alternative liability amount (between \$0 and \$50) for TechWorld, the merchant named in the contested card-fraud claim. We also validate the claim’s identity, amount, dates, status, and allowed classifications. For the lost-wallet scenario, the alternative requires rejecting the "voluntary sharing" classification while keeping the provisional-credit fields intact (as they raise a separate timing question). As a control, the alternative grader rejects missing or malformed claims, fabricated identities, and invalid liability values.

We implemented three additional variations of the grading criteria. First, we tested an *accept-either* policy, awarding credit if a response satisfies either the original reference answer or our alternative. Second, we removed the three affected tasks entirely to observe how they influence the overall denominator of a model’s score. Third, we applied a stricter zero-liability rule across seven tasks to test how a tighter interpretation of consumer law impacts accepted answers.

3.5 Analysis of public evaluations

We analyzed 111 scored public attempts for each of our three targeted scenarios to see how the benchmark’s requirements influence actual model behavior. We included early stops but excluded one infrastructure failure per scenario. Rather than relying solely on the final task score, we extracted the raw database records left by the agents to observe their recorded decisions: whether they filed the omitted claim, what liability they recorded for TechWorld, and which PIN category they chose for the lost wallet. Missing records were tracked to fully capture what the agents left in the database.

To determine if a model’s general capability correlates with its likelihood of enforcing a restrictive bank policy, we established a baseline comparison score for each submission using 81 unaffected tasks. We excluded any tasks involved in our grading changes or the broken-call control, isolating a common set of usable results across all submissions. Each submission’s baseline score is its average pass rate across this unaffected set. For each disputed outcome, we calculated its frequency across all scored attempts for that scenario, including runs that failed before ever reaching the filing step.

We then used Spearman correlation to measure the relationship between this baseline capability and the frequency of disputed outcomes. A positive correlation indicates that higher-scoring submissions record the outcome *more* often (e.g., filing the omitted claim). To check sensitivity to model lineage, we applied two checks across the nine model families in our sample. We subtracted each family’s mean from the ranked values, and we separately estimated uncertainty by resampling whole families. Because the sample contains only nine families, these remain exploratory comparisons (with further robustness checks detailed in the Appendix).

To understand *how* agents executed these omissions, we audited their tool-call sequences. For an agent to receive the dispute history, it must query the relevant customer’s account history without errors. Our strictest test required this successful history response to arrive *before* the agent’s first filing request on the capped account. Simultaneous requests or merely unlocking the tool did not

qualify. This check establishes the sequence of actions, confirming the agent had access to the history before omitting the claim. It cannot prove the agent’s internal reasoning for the omission.

4 Results

Our results proceed in three stages. First, our document sweep reveals that the fictional bank’s policy contains a mix of restrictive rules and additional customer protections. Second, our controlled replays demonstrate which of these restrictive policies the benchmark enforces through its grading criteria. Third, we analyze public leaderboards to explain why these penalties currently remain hidden and expose what AI agents are doing behind their zero scores.

4.1 The policy sweep finds restrictions and additional protections

τ^3 -Banking supplies a library of product descriptions, procedures, and customer guidance [21]. Across the four documents that explicitly cite consumer law (Regulation E), we identify 51 distinct review units. Of these, 14 contain a bank policy that gives less protection than Regulation E. Conversely, four offer additional protections, 12 align with the law, and 11 require further conditions to clarify their impact. The remaining ten units cover operational instructions (9) or introduce the legal framework (1). Table 1 summarizes the ledger. These counts represent individual occurrences of rules in the text, including repetitions, rather than 51 independent legal tests.

Comparison with the legal rule	Clause occurrences
Less protection	14
More protection	4
Consistent	12
Conditional or unclear	11
Operational	9
Context	1

Table 1. A complete review of the four documents that explicitly cite Regulation E. Counts describe the units in the published ledger, including repeated rules. They do not estimate a benchmark-wide error rate. Operational units cover interface and security procedures.

Some of these clauses conflict with the regulation on their face. For instance, the dispute-filing guide outright rejects disputes below \$1 and refuses claims on closed accounts, whereas Regulation E imposes no minimum dollar amount and preserves error-resolution rights after an account is closed, under comment 11(a)-4 [8]. Similarly, the provisional-credit guide gives the bank 45 *business* days to investigate a claim (exceeding the regulatory 45 *calendar* days) and permits a \$500 deduction from temporary credit, exceeding the conditional withholding limit of \$50 [8]. In Table 2, the deduction and investigation-period rules state the wrong amount and unit of time. Applying the other clauses to a particular claim requires checking the conditions shown in the table.

However, the sweep also reveals bank policies that help the customer. The bank promises written confirmation for every resolution, exceeding the law’s baseline allowance for oral notice. Two documents promise advance warning before reversing temporary credit, a benefit compared with notice upon debiting. Document 032 promises at least three business days and document 037 gives no interval. Neither specifies the five-business-day notice that can replace the separate payment protection under comment 11(d)(2)-1 [8]. The bank’s internal list of cases eligible for a 90-day extended investigation omits domestic point-of-sale transactions. By defaulting to a shorter

timeframe for these cases, the bank promises customers a faster resolution than the law mandates [8].

Other policies resist a strict directional label. For example, the balance guide broadly states that consumer law "requires opt-in for overdraft coverage." In reality, the law only conditions *fees* for ATM and one-time debit-card overdrafts on consent. A bank can legally cover an overdraft without consent as long as it does not charge the customer [9]. This legal distinction is significant, but the bank policy's phrasing alone does not prove how the bank would treat a customer in practice. We classify such cases as *unclear* rather than forcing them into a binary category.

Bank policy	External rule	Why the difference matters
The transaction must be no more than 60 days old to file a claim (doc. 031).	Section 1005.11(b)(1)(i) starts the notice window when the bank sends the statement that first shows the error.	A transaction-date cutoff can reject a claim that the customer reports on time under the statement-based rule [6].
The customer must contact the merchant to qualify for provisional credit on a non-fraud dispute (doc. 032).	Section 1005.11(c)(2) sets conditions for credit during an investigation. It does not require prior merchant contact.	Lack of merchant contact alone cannot remove the conditional credit duty for a covered error. Some merchant-service complaints fall outside this rule. FAQ 3 separately rejects merchant contact as a condition of starting an investigation [6].
The bank may deduct \$500 of customer liability from provisional credit (doc. 032).	Section 1005.11(c)(2)(i) permits withholding at most \$50 when its conditions apply.	The rule confuses potential final liability with the smaller amount the bank may withhold during investigation.
The bank has 45 business days to investigate after issuing provisional credit (doc. 032).	Section 1005.11(c)(2) and comment 11(c)-5 specify 45 calendar days for the ordinary extended period.	Counting business days gives the bank extra time beyond that period. Separate exceptions permit 90 days.
Incorrect amounts and uncredited ATM deposits never require provisional credit (doc. 032).	Section 1005.11(a)(1)(ii), (iii), and (iv) includes incorrect transfers, omissions, and bookkeeping errors. The credit procedure in (c)(2) can apply.	The type of error alone cannot justify withholding credit when the conditions apply. The rule does not require immediate credit in every case.

Table 2. We compare five clauses within the policy sweep from the pinned benchmark release [24] with Regulation E and official commentary [8]. Provisional credit means temporary credit while the bank investigates. Document numbers abbreviate `doc_bank_accounts_bank_accounts_(general)_NNN`.

The two clearest written errors are not enforced by the grading criteria. When we replayed all 23 reference debit-dispute filings across the eight customer scenarios, we observed zero instances of the \$500 deduction from temporary credit. Of the 13 claims that recorded temporary credit, all 13 credited the full disputed amount. Even when the reference answer forced a \$500 customer liability for a \$625 purchase, it still credited the full \$625. Because the agent’s filing tool stores liability separately and never subtracts it from the credit amount, the benchmark’s reference answer ignores its own written deduction rule.

Similarly, the incorrect 45-business-day investigation rule is never enforced in the workflow. The filing tool lacks any input field for an investigation deadline, and none of the 97 tasks evaluate the agent on calculating this date. We therefore find no secondary scoring mechanism for either of these rules in the current benchmark release. The bank’s documents feed agents legally inaccurate guidance, but the benchmark’s grading criteria do not penalize an agent for ignoring these specific textual errors. This check verifies the required database records. It does not measure whether conversational agents repeat the wrong deadline to customers in the chat.

4.2 Controlled tests of claims, liability, and PIN records

Dispute filings. The benchmark’s reference answer leaves a \$275 transfer unfiled in task 084 and a \$350 charge unfiled in task 086, enforcing the restrictive bank policy that caps open disputes. However, consumer law requires an investigation after qualifying notice of a covered error. It does not condition this duty on how many other disputes a customer has open, under Section 1005.11(b)-(c) [8]. In the benchmark’s simulation, using the dispute-filing tool is the only way to record a customer’s claim. A real-world bank might initiate an investigation through an alternative route, but the benchmark provides no other mechanism for these claims. Even though the tool technically accepts these additional filings, the benchmark’s grading criteria will fail an otherwise passing run if an agent records them.

Customer liability. The reference answer also assigns the customer full liability for a \$412.88 TechWorld charge in one scenario and \$500 of a \$625 TechWorld charge in the other. Both customers report they still possess their physical cards. Under consumer law, however, retaining the physical card does not settle liability, as an "access device" can also include a code under section 1005.2(a) [10].

The lower amounts we tested depend on assumptions about how and when the customer discovered the fraud. If the customer first learns of any relevant device loss or theft on the reporting date, section 1005.6(b)(1) limits liability to at most \$50, apart from any liability for later transfers under the separate statement-notice rule in (b)(3). The higher tier in (b)(2) requires delayed notice after discovery.

Our zero-liability alternative assumes that the transfer did not involve an access device. Comment 6(b)(3)-2 excludes the first two liability tiers in that case, subject to the statement-notice rule [11]. Possession of the physical card alone does not establish this assumption. The tasks leave statement timing and the nature of the compromise unresolved. Under these assumptions, changing either reference amount to zero or \$50 causes the original grader to fail the run.

The \$350 scenario also leaves uncertainty about coverage and dates. The bank lists both accounts, Green Fee-Free and Evergreen, in its section on personal checking and savings pairings in document 012 [24]. That provides the strongest available evidence of consumer coverage. The customer also uses the accounts for restaurant expenses, and the task does not state their primary purpose at establishment, which determines whether the accounts meet the consumer-account definition in

section 1005.2(b)(1) [10]. Its script gives conflicting January and November discovery dates. Our alternatives use the fixed November task date and assume reporting on discovery.

PIN classification. The lost-wallet scenario (task 082) tests whether a customer granted another person access to their account. The customer admits their lost wallet contained a written PIN, prompting the reference answer to officially record this as "voluntary sharing." However, consumer law (comment 6(b)-2) distinguishes between simple negligence (writing a PIN on paper kept with the card) and intentionally granting someone authority to use an access device [10, 11]. CFPB guidance also treats a thief using a card from a stolen wallet as an unauthorized transfer [6]. Losing a written PIN provides no legal evidence of granted authority. When we change this classification from "shared" to "unknown," while keeping every other part of the reference answer identical, the grading criteria fail the run. The scenario’s dates also conflict. The customer lost the wallet *after* the disputed transaction occurred, meaning the loss cannot logically explain the charge. In contrast, a separate scenario (task 083) features a customer intentionally lending a nephew their card and PIN, correctly supplying the explicit grant of authority that the lost-wallet case lacks.

Scenario	Controlled sequence	Passes	New filings
\$275 claim (084)	Reference	Yes	3
	Add the omitted claim	No	4
	Change TechWorld liability to 0 or 50	No	3
\$350 claim (086)	Reference	Yes	5
	Add the omitted claim	No	6
	Change TechWorld liability to 0 or 50	No	5
Lost wallet (082)	Reference	Yes	4
	Change PIN category to unknown	No	4
Nephew case (083)	Broken expected calls	Yes	0
	Repaired calls only	No	4
	Broken calls followed by repairs	Yes	4

Table 3. One changed decision can turn a passing reference into a failing run. The original evaluator supplies the pass result. New filings count all disputes that the sequence opens, including claims outside the targeted change.

4.3 Invalid tool calls can earn credit

The nephew scenario (task 083) reveals a second, more mechanical way the benchmark rewards errors. In this scenario, the reference answer expects the agent to make four separate dispute-filing calls. However, all four of these expected calls omit a required liability argument. Because they are malformed, all four calls fail to execute, meaning the reference answer opens zero disputes for the customer.

Because this specific task evaluates the agent based on its expected tool calls rather than the final database state, the grading criteria reward this failure. Table 3 shows this result. If an agent repeats the invalid calls exactly as expected, it earns a passing score. Conversely, if an agent detects the missing argument and repairs the calls to open the disputes, it loses credit because its repaired calls no longer match the broken reference answer. An agent can achieve both goals, earning credit *and* opening the disputes, by repeating the calls. The agent must make the broken calls first to satisfy the grader, and then repeat them with the missing argument added to file the claims. Because the grading criteria tolerate extra calls as long as the expected ones are present, this redundant

sequence passes. A passing score on this task merely certifies that the agent repeated the reference answer’s mistakes, even if it failed to open a single reported claim for the customer. To check for other mechanical defects, we swept every reference answer in the benchmark. We found only one other tool error (a duplicate request in task 051), confirming that this missing-argument defect is unique to the nephew scenario.

4.4 Effects on scores and rankings

We replayed 10,880 public runs across 32 submissions and reproduced published scores for 26 of the 32. Following the benchmark’s official metrics utility, we excluded 161 infrastructure failures (though some published leaderboards count these as outright failures). This provided a valid result for every remaining run, allowing us to evaluate all accessible submissions for our main analysis and isolate a reproduced-score subset for verification.

We found that none of the 111 scored attempts at each capped-claim or lost-wallet scenario passed the overall task under *either* set of grading criteria. Because today’s agents fail other unaffected steps in the workflow, applying our "accept-either" grading criteria, which accept the alternative records, changes no final scores or leaderboard rankings. The original criteria, our main alternative, and the accept-either rules all yield the exact same score for every public submission. This holds true even when restricting our analysis to the specific submissions whose published scores we reproduced. We investigated the remaining score discrepancies by tracing how infrastructure failures were handled, documented corrections to an unrelated ATM-refund task, and metadata mismatches between archived outcomes and published records, leaving three causes unconfirmed, as detailed in Appendix A.3.

We tested broader scoring adjustments. Removing the three selected scenarios entirely raises overall pass rates slightly (by 0.31 to 1.76 percentage points) because submissions face fewer total tasks, but this adjustment reverses no strict rankings. Conversely, applying our stricter zero-liability rule across seven tasks lowers scores by at most 0.52 points and reverses three pairwise rankings, as it demands a tighter interpretation of consumer law and accepts fewer answers.

4.5 What agents record in customer accounts

Figure 1 details the claims agents filed, the liability amounts they recorded, and the PIN classifications they selected. Analyzing these raw database records allows us to distinguish between fundamentally different agent decisions that all receive the same failing task score.

We found that agents filed the omitted \$275 transfer in 34 of 111 attempts (roughly 30%), and the \$350 charge in 23 attempts (roughly 20%). In the remaining runs, agents left the reported claims unfilled, either because they enforced the dispute cap or because they failed to complete the workflow. When recording customer liability for the TechWorld charges, agent records varied. Many copied the reference answer’s liability, some recorded \$50 or less, others recorded unlimited liability, and some left no record at all, as seen in Figure 1 and Table 6.

In the lost-wallet scenario, 54 runs recorded the written PIN as "shared." However, 37 runs avoided the reference answer’s category, logging the PIN compromise as "observed" or "unknown." The remaining 20 runs created no record. These database categories describe the agent’s data entry. They do not establish how the third party obtained the PIN.

Fifty runs across our scenarios recorded the alternatives we targeted, yet still failed the overall task. This occurred in 37 lost-wallet runs, eight \$275-claim runs, and five \$350-claim runs. For the claim scenarios, a match required the agent to both file the omitted claim *and* record a lower

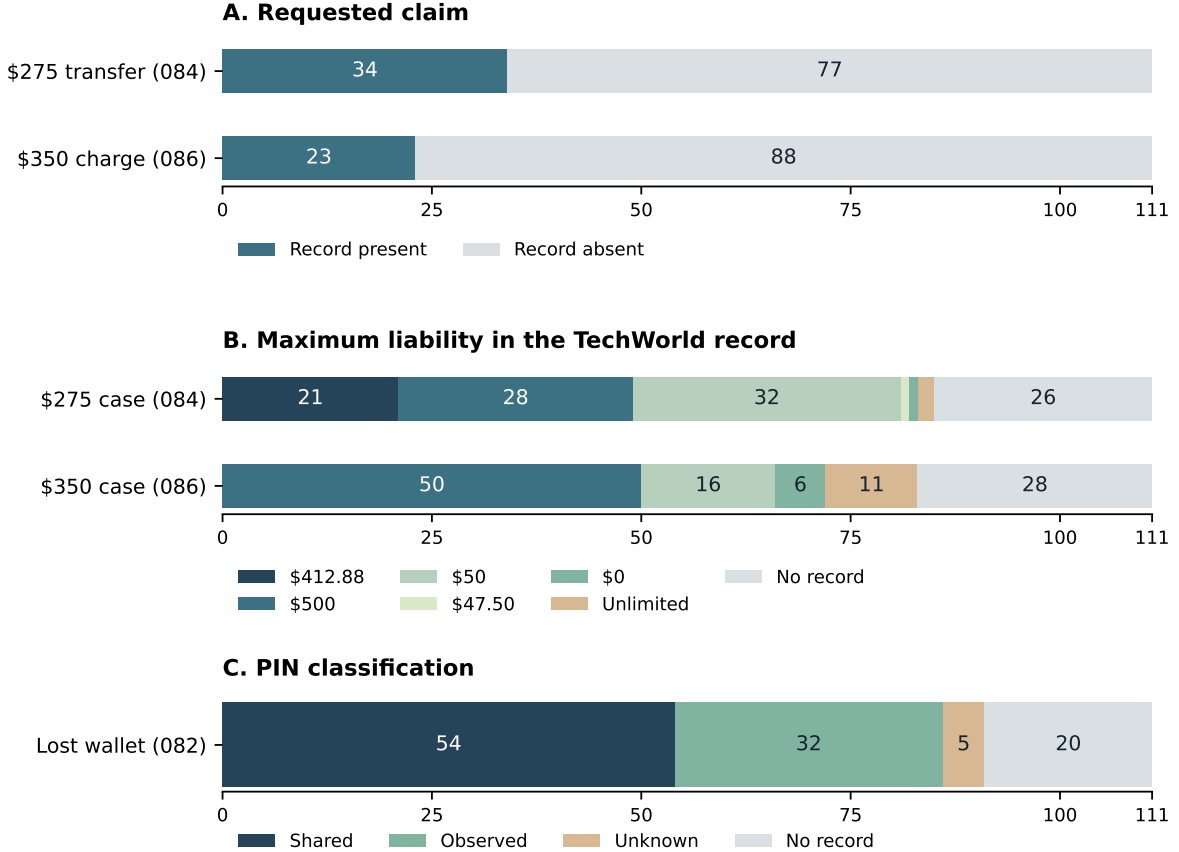


Figure 1. Agents make different decisions even when they all fail the task. Each row counts 111 attempts at the named scenario, including early stops. Green segments in the liability panel span zero to \$50. The tool uses -1 to represent unlimited liability.

liability. For instance, one of the eight \$275-case matches specifically recorded a liability of \$47.50, falling within our alternative \$0-to-\$50 range.

These 50 runs concretely illustrate the gap between recording an alternative and earning a passing score. For example, an Opus 4.5 run in the \$275 scenario filed the claim and lowered the liability, but the grading criteria still failed it because the agent’s database mismatched the reference answer on unrelated fields like dispute identity, transaction type, and card orders. Similarly, a Fable 5 run in the wallet scenario avoided the "shared" classification but failed due to discrepancies in provisional credit and card closures. These mismatches explain why the benchmark rejected the runs, even though the agents met the targeted record conditions.

4.6 Higher-scoring agents file fewer capped claims

When comparing these outcomes against an agent’s baseline capability, we see a clear pattern. Higher-scoring submissions leave the capped claims unfilled more often. The correlation between performance on the 81 unaffected tasks and claim recording is $\rho = -0.38$ for the \$275 transfer and -0.56 for the \$350 charge. These negative correlations remain robust even after adjusting for model-family averages or systematically leaving out one family at a time (as detailed in Appendix A.4).

Higher-scoring submissions are more likely to enforce the dispute cap, but they are also *more* likely to assign lower liability amounts (at most \$50) for the TechWorld charges. Their choices in

Recorded outcome	Spearman ρ	Family-centered rank correlation	Family bootstrap interval for ρ
\$275 claim recorded (084)	-0.38	-0.43	[-0.63, -0.06]
\$350 claim recorded (086)	-0.56	-0.57	[-0.76, -0.23]
TechWorld 0–50, \$275 case	0.27	0.21	[-0.08, 0.71]
TechWorld 0–50, \$350 case	0.34	0.58	[0.19, 0.72]
Lost-wallet PIN not shared	0.19	0.26	[-0.21, 0.56]

Table 4. How each recorded decision relates to performance on 81 other banking tasks. Positive correlations mean higher-scoring submissions record the outcome more often. Negative correlations mean they do so less often. Each submission contributes one rate, using all its scored attempts. We obtain the 95% intervals by resampling the nine model families. They describe ordinary Spearman correlation.

the lost-wallet PIN scenario show a weaker association with broader performance. A structural flaw in the benchmark offers a possible explanation for this divergence. The bank’s knowledge base gives agents explicit information to apply the dispute cap, but it omits the facts they need to calculate the reference answer’s liability (as identified in issue 592). Higher-scoring submissions may follow the restrictive policies that are documented, but choose lower liability amounts when the policy facts are unclear. See Figure 2.

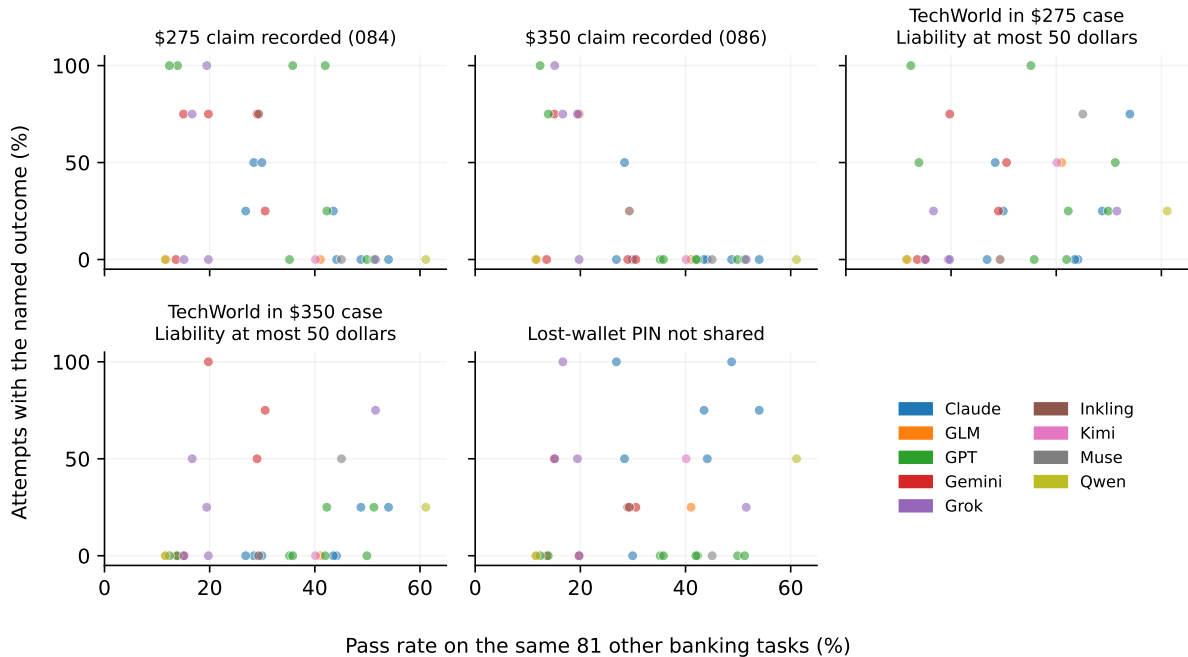


Figure 2. Each point represents a submission. The horizontal axis shows its pass rate on the same 81 other tasks. The vertical axis shows how often it records the named outcome in the disputed scenario. Colors identify model families. Some points overlap at zero and 100%. Missing records count as absence of the outcome. The panels use all 111 attempts per scenario, grouped by submission.

To check sensitivity to submissions with very few trials, we restricted our sample to the 27 submissions that attempted at least three runs per main scenario. Under this restriction, the negative association between capability and claim filing grows even stronger, reaching $\rho = -0.54$ for the \$275 claim and -0.61 for the \$350 claim (with both family-resampling intervals remaining below zero). The positive association for lower liability also holds. All five excluded submissions used

voice interactions and gave the agent access to every tool simultaneously, meaning this restriction altered the evaluation setup as well as the trial count, as fully detailed in Appendix A.4.

To test whether omissions followed a history check, we audited their internal tool sequences. Among the 77 runs that omitted the \$275 claim, 59 retrieved the customer’s dispute history before making their first successful filing on the capped account, and each one finished with exactly two open disputes. Similarly, of the 88 runs that omitted the \$350 claim, 70 checked the history beforehand, with 69 finishing at the cap. Together, 128 omissions followed the same sequence: the agent checked the existing disputes, filed a claim to fill the last available slot, and left the extra claim unfilled.

Omitted claim	Total	Check before filing	Other completed check	No completed check
\$275 (084)	77	59	6	12
\$350 (086)	88	70	0	18

Table 5. What agents do before leaving a claim unfilled. A prior check requires the history response to arrive before the first successful filing request on the capped account. A completed check alone cannot establish why the agent omits the claim.

Conversely, agents that filed the extra claims checked the customer’s history far less often (only 9 out of 34 runs for the \$275 claim, and 11 out of 23 for the \$350 claim). We cannot establish internal intent or observe the effect of training on benchmark rewards. Most omissions followed a completed history check, so an early stop before that check cannot explain them.

4.7 Conflicting dates and unlimited liability

The \$350-claim scenario reveals another defect. Contradictory timelines in the prompt may lead agents to record higher liability. In our analysis, 11 runs recorded unlimited liability for the TechWorld charge. Each one of these records logged January 12, 2026, as the discovery date, a date mentioned in the customer’s script, but which falls 100 days after the disputed purchase and after the benchmark’s fixed internal clock of November 14, 2025.

Three additional runs used this contradictory January date to assign \$500 liability. In contrast, no agent that used a November discovery date assigned unlimited liability. This pattern suggests that agents are pulling the future date from the script and using it to enforce a late-notice penalty under consumer law. Confirming this would require fixing the scenario’s timeline and rerunning the agents.

5 Discussion

5.1 What a higher score would reward

The benchmark’s grading criteria matter long before agents are capable of satisfying them. Currently, because today’s agents fail other steps in the capped-claim scenarios, the overall task score is a poor indicator of whether a model enforces the restrictive bank policy. However, as models improve and begin completing the rest of the workflow, this dormant requirement could become the deciding factor between passing and failing. If future agents complete the surrounding steps while the reference answers remain fixed, recording the alternative will cost them credit.

Waiting for a shift in public rankings would mean postponing correction until the benchmark is penalizing otherwise successful models. Our controlled replays prove this penalty already exists

in the code, and our behavioral analysis shows higher-scoring submissions leaving capped claims unfilled more often.

Consider a future developer whose agent completes every step but files the extra claim. Seeing a failed score, the developer might treat that alternative record as a "bug" to be fixed. The benchmark would then reward the developer for patching the agent to enforce the restrictive bank policy. Our archived data cannot prove whether developers are already training models to chase these specific rewards. Our experiments show the benchmark provides the structural incentive to do so.

5.2 Recommendations for benchmark maintainers

Benchmark maintainers can implement three practical changes to prevent these hidden penalties.

First, document the origin of every scoring requirement. Fictional bank policies, actual statutes, official commentary, and agency guidance all carry different legal weight. If a task cites consumer law (such as Regulation E), the documentation must identify the specific provision and define how it applies to the expected outcome.

Second, report individual agent decisions alongside the final task scores. Readers should be able to see whether an agent filed a claim, assigned a specific liability amount, or stopped before creating a record. If a maintainer proposes an alternative grading rule, they should state which database fields change, which remain fixed, and why, allowing researchers to verify what the alternative accepts.

Third, proactively test alternative outcomes before publishing or revising a reference answer. A maintainer can execute the reference, alter one contested record, and verify how the grading criteria evaluate it. Implementing "accept-either" grading can recognize the alternative records while still crediting the original path. Furthermore, if an agent finds a valid, separate escalation route to satisfy the customer's request, the grading criteria must be flexible enough to recognize that result.

5.3 Correction and disclosure in practice

The benchmark ecosystem is capable of correcting these flaws when they are properly reported. For example, a previous public issue (Issue 374) identified an ATM-fee refund in task 074 that contradicted the bank's own internal allowances. The reference answer refunded \$8.00 when the policy implied \$14.50. A project contributor accepted the correction for v1.0.1 [18], and the release we audit now records the larger refund [24]. A well-documented, reproducible report changed the reference answer.

Future audits comparing a benchmark's requirements against actual consumer law should follow this same path. An effective bug report should detail the restrictive bank policy clause, the external legal source, the affected database record, an executable example of the failure, and the strongest counterargument defending the current setup. Following this standard, we posted our own findings and full reproduction materials to issue 592 on 7 October 2026 [16]. Transparency should extend beyond benchmark developers. Evaluators and enterprise procurement teams should question the leaderboards they rely on. Before trusting a score as proof of readiness, they should ask which reference answers produced it, whether the grading criteria accept alternative records, and which external legal sources justify the rules the agent is forced to enforce.

6 Limitations

The document sweep evaluates all explicit Regulation E citations within a single, frozen library. Because this boundary excludes policies that never explicitly name the regulation, it provides a

comprehensive audit within that specific scope rather than an estimate of errors across all banking benchmarks. Furthermore, we evaluate the rules exactly as stated (including repetitions) rather than cataloguing every duty the documents fail to mention. We conducted this sweep after reviewing our initial cases to provide broader context, not as a randomized sample.

Our three replay scenarios remain deliberately selected cases. We traced the credit deduction and investigation-period clauses into the benchmark’s implementation, but we have not traced every newly discovered policy from the document sweep into the grading criteria.

We developed our alternative records using official legal sources and model assistance. However, missing context within the benchmark, such as account purpose, statement timing, and conflicting prompt dates, limits definitive legal conclusions. Additionally, we rely on current CFPB indexes because we could not directly retrieve historical eCFR text from before the scenario dates [12]. Our policy table carefully distinguishes between formal regulatory text, official commentary, and FAQ guidance.

Our alternative grading criteria modify a narrow set of records while retaining all other requirements, and therefore, they might still reject a defensible run based on an unrelated, contested liability or credit decision. Furthermore, a correct database record only proves successful data entry and cannot confirm whether the agent accurately explained the customer’s rights in the chat. This study measures simulated actions and benchmark scoring, not real-world consumer harm.

Public submissions vary widely in retrieval settings, interaction modes (voice vs. text), model lineage, and trial coverage. Although we control for this by using unaffected tasks as a baseline and checking several subsets, our correlational analysis cannot prove a strict causal relationship between model capability and policy enforcement. Resampling across only nine model families provides a limited view of compositional effects, and future benchmark releases or model updates may differ from our frozen snapshot.

7 Conclusion

This case demonstrates that AI-agent benchmarks can favor firms over consumers through their reference answers. In the scenarios we tested, an agent must make contested decisions that favor the bank to earn a passing score. If an agent files an additional reported claim, lowers a customer’s recorded liability under our stated assumptions, or refuses to misclassify a lost PIN as "voluntary sharing," an otherwise passing run fails. This evidence establishes this mechanism within a single benchmark. It does not prove a universal bias across all evaluations, especially given that the fictional bank’s own documents occasionally offer extra customer protections.

Today’s industry rankings hide this penalty because models fail other steps in the workflow. Consequently, accepting our alternative records leaves every current leaderboard score unchanged. Nevertheless, raw database records show higher-scoring submissions leaving capped claims unfiled more often. Once agents become capable enough to finish the rest of the workflow, these contested requirements could become the threshold that separates a passing agent from a failing one.

Benchmark maintainers have the power to act before this happens. By checking their cited consumer law, repairing flawed policies, and testing their grading criteria against alternative records, they can eliminate these hidden traps. An AI agent’s ability to rigidly follow a company’s policy is a useful metric. However, whether that policy respects a customer’s legal rights is a question the benchmark must answer as well.

References

- [1] Artificial Analysis. τ^3 -banking benchmark leaderboard, 2026. URL <https://artificialanalysis.ai/evaluations/tau3-banking>. Accessed 7 October 2026.
- [2] bhavishya-pohani. [banking_knowledge] task_086: gold liability amounts need a statement date that the task never gives, 2026. URL <https://github.com/sierra-research/tau2-bench/issues/592>. GitHub issue 592, opened 3 October 2026.
- [3] Hongliu Cao. Policy loopholes in agent evaluation: When policy ambiguity masquerades as agent error. arXiv:2609.14400, 2026. URL <https://arxiv.org/abs/2609.14400>.
- [4] Hongliu Cao, Ilias Driouich, and Eoin Thomas. Beyond task completion: Revealing corrupt success in LLM agents through procedure-aware evaluation. arXiv:2603.03116, 2026. URL <https://arxiv.org/abs/2603.03116>.
- [5] Stephen Casper, Carson Ezell, Charlotte Siegmann, Noam Kolt, Taylor Lynn Curtis, Benjamin Bucknall, Andreas Haupt, Kevin Wei, J  r  my Scheurer, Marius Hobbhahn, Lee Sharkey, Satyapriya Krishna, Marvin Von Hagen, Silas Alberti, Alan Chan, Qinyi Sun, Michael Gerovitch, David Bau, Max Tegmark, David Krueger, and Dylan Hadfield-Menell. Black-box access is insufficient for rigorous AI audits. In *The 2024 ACM Conference on Fairness, Accountability, and Transparency*, pages 2254–2272. Association for Computing Machinery, 2024. doi: 10.1145/3630106.3659037. URL <https://doi.org/10.1145/3630106.3659037>.
- [6] Consumer Financial Protection Bureau. Electronic fund transfers FAQs, 2021. URL <https://www.consumerfinance.gov/compliance/compliance-resources/deposit-accounts-resources/electronic-fund-transfers/electronic-fund-transfers-faqs/>. Error Resolution questions 3 and 4, updated 4 June 2021, and Unauthorized EFTs question 4, updated 13 December 2021. Accessed 13 October 2026. Agency guidance.
- [7] Consumer Financial Protection Bureau. Chatbots in consumer finance, June 2023. URL <https://www.consumerfinance.gov/data-research/research-reports/chatbots-in-consumer-finance/chatbots-in-consumer-finance/>.
- [8] Consumer Financial Protection Bureau. 12 CFR 1005.11: Procedures for resolving errors and official interpretations, 2026. URL <https://www.consumerfinance.gov/rules-policy/regulations/1005/11/>. Current text accessed 7 October 2026.
- [9] Consumer Financial Protection Bureau. 12 CFR 1005.17: Requirements for overdraft services and official interpretations, 2026. URL <https://www.consumerfinance.gov/rules-policy/regulations/1005/17/>. Current text accessed 8 October 2026.
- [10] Consumer Financial Protection Bureau. 12 CFR 1005.2: Definitions and official interpretations, 2026. URL <https://www.consumerfinance.gov/rules-policy/regulations/1005/2/>. Current text accessed 7 October 2026.
- [11] Consumer Financial Protection Bureau. 12 CFR 1005.6: Liability of consumer for unauthorized transfers and official interpretations, 2026. URL <https://www.consumerfinance.gov/rules-policy/regulations/1005/6/>. Current text accessed 7 October 2026.
- [12] Consumer Financial Protection Bureau. Version history of 12 CFR 1005.6 (Regulation E), 2026. URL <https://www.consumerfinance.gov/rules-policy/regulations/1005/versions/6/>. CFPB version index lists 19 April 2023 as the current version. Accessed 7 October 2026.
- [13] Carson Ezell, Xavier Roberts-Gaal, and Alan Chan. Incident analysis for AI agents. *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, 8(1):865–878, 2025. doi: 10.1609/aies.v8i1.36596. URL <https://ojs.aaai.org/index.php/AIES/article/view/36596>.
- [14] Aryo Pradipta Gema, Joshua Ong Jun Leang, Giwon Hong, Alessio Devoto, Alberto Carlo Maria Mancino, Rohit Saxena, Xuanli He, Yu Zhao, Xiaotang Du, Mohammad Reza Ghasemi Madani, Claire Barale, Robert McHardy, Joshua Harris, Jean Kaddour, Emile van Krieken, and Pasquale Minervini. Are we done with MMLU? In *Proceedings of the 2025 Conference of the Nations of the Americas*

- Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 5069–5096. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.naacl-long.262. URL <https://aclanthology.org/2025.naacl-long.262/>.
- [15] Induction. tau3-banking, 2026. URL <https://www.induction.ai/docs/benchmarks/tau3>. Updated 2 October 2026. Accessed 7 October 2026.
 - [16] JOYY. Follow-up audit and executable reproduction on banking issue 592. GitHub issue comment, 2026. URL <https://github.com/sierra-research/tau2-bench/issues/592#issuecomment-6042790694>. Posted 7 October 2026. Public visibility verified 13 October 2026.
 - [17] Ilija Lichkovski, Alexander Müller, Mariam Ibrahim, and Tiwai Mhundwa. EU-Agent-Bench: Measuring illegal behavior of LLM agents under EU law. arXiv:2510.21524, 2025. URL <https://arxiv.org/abs/2510.21524>.
 - [18] mohitunplugged. Banking task 74 / T074: gold under-refunds Light Blue ATM fees (\$8.00 vs \$14.50) against the 2-free-per-type policy, 2026. URL <https://github.com/sierra-research/tau2-bench/issues/374>. GitHub issue 374, opened 28 June 2026. Contributor benshi34 accepted the correction for v1.0.1 on 15 July 2026.
 - [19] Curtis G. Northcutt, Anish Athalye, and Jonas Mueller. Pervasive label errors in test sets destabilize machine learning benchmarks. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021. URL <https://datasets-benchmarks-proceedings.neurips.cc/paper/2021/hash/f2217062e9a397a1dca429e7d70bc6ca-Abstract-round1.html>.
 - [20] Cullen O’Keefe, Ketan Ramakrishnan, Janna Tay, and Christoph Winter. Law-following AI: Designing AI agents to obey human laws. *Fordham Law Review*, 94(1):57–129, 2025. URL <https://ir.lawnet.fordham.edu/flr/vol94/iss1/2/>.
 - [21] Quan Shi, Alexandra Zyttek, Pedram Razavi, Karthik Narasimhan, and Victor Barres. τ -knowledge: Evaluating conversational agents over unstructured knowledge. arXiv:2603.04370, 2026. URL <https://arxiv.org/abs/2603.04370>.
 - [22] Sierra Research. Add v1.0.1 frontier banking results. GitHub commit 0f67910830d0b0168c43637de9ed0d5a355cd56c, pull request 452, 2026. URL <https://github.com/sierra-research/tau2-bench/commit/0f67910830d0b0168c43637de9ed0d5a355cd56c>. Metadata history accessed 8 October 2026.
 - [23] Sierra Research. τ -bench public submission metadata, 2026. URL <https://github.com/sierra-research/tau2-bench/tree/4ce7c0397c1eb65c9bbe59aeacfe1ca44a1cd699/web/leaderboard/public/submissions>. Frozen commit 4ce7c0397c1eb65c9bbe59aeacfe1ca44a1cd699, includes developer submissions.
 - [24] Sierra Research. τ -bench, release v1.0.1. GitHub repository, commit fc0055dc4e0a316c3f83133267fbd6faaa770992, 2026. URL <https://github.com/sierra-research/tau2-bench/tree/fc0055dc4e0a316c3f83133267fbd6faaa770992>.
 - [25] Shunyu Yao, Noah Shinn, Pedram Razavi, and Karthik Narasimhan. τ -bench: A benchmark for tool-agent-user interaction in real-world domains. arXiv:2406.12045, 2024. URL <https://arxiv.org/abs/2406.12045>.
 - [26] Yuxuan Zhu, Tengjun Jin, Yada Pruksachatkun, Andy Zhang, Shu Liu, Sasha Cui, Sayash Kapoor, Shayne Longpre, Kevin Meng, Rebecca Weiss, Fazl Barez, Rahul Gupta, Jwala Dhamala, Jacob Merizian, Mario Giulianelli, Harry Coppock, Cozmin Ududec, Jasjeet Sekhon, Jacob Steinhardt, Antony Kellermann, Sarah Schwetzmänn, Matei Zaharia, Ion Stoica, Percy Liang, and Daniel Kang. Establishing best practices for building rigorous agentic benchmarks. arXiv:2507.02825, 2025. URL <https://arxiv.org/abs/2507.02825>.

A Reproduction details

We provide the complete code, source and download hashes, corpus inventories, and per-run results in the public repository <https://github.com/JOYY/tau3-policy-as-reward>. The accompanying README details how to rebuild all figures and tables, as well as how to download and replay the public trajectories. Because our experiments rely entirely on CPU replay, they require no new model inference. Our supplementary trace `results/credit_rule_trace.json` records all 23 reference debit-dispute calls, their resulting records, the filing-tool implementation, and the deadline search across all evaluation criteria, regenerated via `python -m scripts.trace_credit_rules`.

We use benchmark commit `fc0055dc4e0a316c3f83133267fbd6f6aaa770992` and freeze leaderboard metadata at `4ce7c0397c1eb65c9bbe59aeacfe1ca44a1cd699`. Source URLs and SHA-256 hashes identify all external inputs, and our formal disclosure provides a minimal executable reproduction [16].

For scoring, we evaluate each retained task t , letting n_t represent the observed trials and c_t the successful trials. We use the upstream estimator

$$\widehat{\text{pass}}^k = \frac{1}{|T|} \sum_{t \in T} \frac{\binom{c_t}{k}}{\binom{n_t}{k}}, \quad k \leq \min_{t \in T} n_t. \quad (2)$$

This formula estimates how often an agent succeeds on all k attempts, averaged over tasks. We report higher- k results only when every retained task has enough trials. Every run we retain includes the original reward components we need. For example, if an archive lacked a required language-judge result, the replay would leave that score unresolved. For older archives missing response IDs, our adapter pairs calls and responses only when adjacent messages make the match unambiguous. Throughout our data processing, we preserve the raw archives, exclude infrastructure failures, and retain premature stops as zero scores. Our state-extraction code recovered nine early-stop runs that our initial cache had omitted.

Our main figures rely on 111 scored trials per scenario. When restricting this sample to reproduced submissions, 88 trials remain. Within that subset, 30 lost-wallet runs, eight \$275-claim runs, and five \$350-claim runs meet our targeted record conditions. None of these runs pass the overall alternative task, leaving the main figure’s denominator entirely unchanged.

A.1 Exact alternative requirements

To enforce our alternative grading criteria, the added-claim checks require the correct transaction, account, card, customer, amount, dates, open status, allowed category, and consistent derived fields. For liability records, the value must be numeric, finite, non-boolean, and at least zero (we reject the tool’s -1 code for unlimited liability). Only after an agent’s records pass these checks do we remove the permitted record differences and compare the remaining database records and tool logs against the benchmark’s reference answers.

Furthermore, our stronger zero-liability robustness check extends beyond the three primary scenarios to cover four additional dispute cases (tasks 085, 087, 088, and 091). In task 082, it requires zero liability for the ATM cash discrepancy and the card-not-present claim with the card in hand, while retaining the alternative PIN check for the lost-wallet claim. This allows us to test the impact of alternative liability amounts across a broader set of tasks. For full reproducibility, all machine-readable outputs retain their original grading identifiers, while the text of this paper uses descriptive names for clarity.

Scenario	\$412.88	\$500	\$50	\$47.50	\$0	Unlimited	Missing
\$275 case (084)	21	28	32	1	1	2	26
\$350 case (086)	0	50	16	0	6	11	28

Table 6. How much liability agents assign to the TechWorld fraud claim in each capped-claim scenario. Every row sums to 111 runs. Missing means the agent leaves no matching dispute record.

A.2 Counts and robustness checks

In the lost-wallet scenario, agents choose `yes_shared` in 54 runs, `yes_observed` in 32, and `unknown` in five. Twenty runs leave no record. The observed and unknown categories account for the 37 runs that avoid voluntary sharing.

To understand why the 50 runs matching the alternative records still failed the overall task, we tracked their overlapping database mismatches. All 50 runs that met our targeted record conditions mismatched the reference answer’s dispute records elsewhere. Specifically, 46 also differed in debit-card records, 48 contained other liability or provisional-credit discrepancies, 35 mismatched on tool logs, and 29 differed in card orders. These overlapping counts pinpoint the exact grading criteria that triggered the failures, though they do not guarantee that the benchmark’s remaining requirements are legally supported.

For our statistical robustness checks, we resampled tasks 10,000 times in paired samples (using seed 20261007) to measure how sensitive our score comparisons are to the specific mix of tasks. For capability comparisons, we used the same seed to resample whole model families, calculating the ordinary Spearman correlation. We separately centered ranks within each family.

Our sample comprises 32 submissions across nine model families: eight GPT, seven Claude, five Gemini, five Grok (including xAI realtime), two GLM, two Qwen, and one each of Kimi, Muse, and Inkling. Because the three singleton families contribute no variation after centering, our centered statistic draws on 29 submissions across six families. We assigned equal correlation weight to each submission, 25 of which provided four trials per main scenario, two provided three, and five provided one. We omitted constant resamples because their correlation is undefined, leaving at least 9,993 valid resamples for every measured outcome.

Under our stricter zero-liability rule, Kendall’s rank agreement remains extremely high at $\tau_b = 0.975$, with only three pairs reversing order among the reproduced submissions (Muse Spark 1.1 and Opus 4.7, Opus 4.8 and GPT-5.4, Sonnet 4.5 and Inkling). As a baseline correction, counting infrastructure failures as zeros fully explains the score discrepancies for Gemini 2.5 Pro and Qwen3.5-397B-A17B. The remaining four score gaps are investigated below.

A.3 Tracing the four remaining score gaps

To trace the remaining discrepancies, we compared three distinct records for each submission: the published leaderboard metadata, the rewards stored in its public archive, and our own replay against the pinned benchmark release. A deviation from the published leaderboard does not necessarily mean our replay altered an archived reward.

Our replay matches every stored reward for Claude Fable 5, GPT-5.5, and GPT-5.6-sol. Their archives contain 53, two, and 41 infrastructure failures, respectively. Much of the Fable discrepancy arises from how infrastructure failures are counted. For Fable 5, excluding its 53 infrastructure failures yields a 44.89% pass rate, while counting them as zeros yields 38.66%, falling just below its published 39.69% rate. The denominator choice explains most of the gap but cannot close it entirely. We also investigated the metadata itself. Fable 5 and Opus 4.8 publish the exact same single-attempt

Submission	Published	Replay	Missing as zero
Claude Fable 5	39.69	44.89	38.66
Gemini 3.1 Pro Preview	26.03	26.29	26.29
GPT-5.5	44.59	44.50	44.33
GPT-5.6-sol	46.91	47.05	42.53

Table 7. Pass rates in percent. Replay excludes infrastructure failures. The last column counts those failures as zero. Neither convention reproduces these four published rates.

rate (154 successes out of 388 attempts) despite their other reliability metrics differing, and both metadata files entered the repository in the same commit. Opus 4.8’s archive reproduces all four published values, but Fable 5’s does not. By mathematically assigning successes to four missing Fable attempts, we can reconstruct its published reliability metrics (GPT-5.5 would need one such success, and GPT-5.6-sol would need 17). These hypothetical completions prove mathematical compatibility but do not confirm that the missing attempts were rerun, so we leave the root cause unconfirmed [22, 23].

Gemini 3.1 Pro Preview exhibits a genuinely traceable grading difference. Its older archive stores 15 failures that the current official evaluator now scores as passes. One of these involves the ATM-fee refund correction (from \$8 to \$14.50) discussed in Section 5.3. The benchmark’s release history confirms one Gemini trial changed from failure to success following this correction, and our replay identifies this trial [24]. When we restore the earlier \$8 reference answer and re-evaluate all four ATM attempts, the single successful attempt reverts to a failure. By then counting infrastructure failures as zero, we reproduce all four of Gemini 3.1 Pro Preview’s published metrics, including its 26.03% single-attempt rate. This reconciles the gap with a documented grading change.

Our public ledger records all source hashes, simulation identifiers, metadata history, evaluator checks, and hypothetical completions. Throughout our main analysis, we consistently evaluate against the corrected release.

A.4 Trial-count sensitivity

The filing correlations stay negative when we leave out one family at a time. For the \$275 claim they range from -0.45 to -0.27. For the \$350 claim they range from -0.63 to -0.47.

We repeated the comparison using only submissions with at least three scored attempts at each main scenario. This leaves 27 submissions and 106 trials per scenario. All five submissions we exclude use voice and give the agent access to every tool. The restriction therefore changes the evaluation setup as well as trial count. We keep the same 81 comparison tasks and give each submission equal weight. Both claim-filing correlations become more negative, while the lower-liability correlations stay positive.

Recorded outcome	Spearman ρ	Family-centered rank correlation	Family bootstrap interval for ρ
\$275 claim recorded (084)	-0.54	-0.59	[-0.81, -0.15]
\$350 claim recorded (086)	-0.61	-0.63	[-0.85, -0.26]
TechWorld 0–50, \$275 case	0.36	0.34	[0.09, 0.68]
TechWorld 0–50, \$350 case	0.30	0.60	[0.08, 0.73]
Lost-wallet PIN not shared	0.09	0.19	[-0.39, 0.52]

Table 8. The pattern persists among submissions with at least three attempts per main scenario. We resample model families to obtain the intervals.

B Per-submission scores

Submission	Published	Replay [†]	Drop tasks	Stricter liability
Qwen 3.8 Max	55.15	55.15	56.91	55.15
Claude Opus 5	48.71	48.71	50.27	48.71
Grok 4.5	47.94	47.94	49.47	47.68
GPT-5.6-sol *	46.91	47.05	48.57	47.05
Claude Fable 5 *	39.69	44.89	46.47	44.89
GPT-5.5 *	44.59	44.50	45.92	44.50
Muse Spark 1.1	40.46	40.46	41.76	39.95
Claude Opus 4.7	40.21	40.21	41.49	40.21
Claude Opus 4.8	39.69	39.69	40.96	39.18
GPT-5.4	39.43	39.43	40.69	39.43
GLM-5.2	37.11	37.11	38.30	36.60
GPT-live-1 (Astra, high)	37.11	37.11	38.30	37.11
Kimi K3	37.11	37.11	38.30	36.86
GPT-5.2 (high)	32.22	32.22	33.24	32.22
GPT-live-1 (Diamond)	31.96	31.96	32.98	31.96
Claude Opus 4.6	27.32	27.32	28.19	26.80
Gemini 3 Flash	27.32	27.32	28.19	27.32
Gemini 3.1 Pro Preview *	26.03	26.29	27.13	26.29
Claude Sonnet 4.5	25.26	25.26	26.06	24.74
Inkling	25.00	25.00	25.80	25.00
Claude Opus 4.5	24.74	24.74	25.53	24.23
Gemini 3 Pro	18.04	18.04	18.62	18.04
Grok 4.2	18.04	18.04	18.62	18.04
xai-realtime	16.49	16.49	17.02	16.49
Grok 4 fast	15.72	15.72	16.22	15.46
Gemini 2.5 Pro *	13.66	13.83	14.27	13.83
Grok 4.1 fast	13.14	13.14	13.56	13.14
GPT-5.2 (none)	12.63	12.63	13.03	12.63
Gemini 3.1 Flash Live (high)	11.34	11.34	11.70	11.34
gpt-realtime-2	10.31	10.31	10.64	10.31
Qwen3.5-397B-A17B *	9.79	9.88	10.20	9.88
GLM-5	9.79	9.79	10.11	9.79

[†] Original, primary alternative, and accept-either scores are identical for every submission and share this column. * Replay does not reproduce the published score. All values are percentages. Higher- k scores, trial coverage, and bootstrap intervals appear in the replication tables.