
When Monitors Agree for the Wrong Reasons: Information Design for Collusion Auditing

Nirdesh Bhandari
NIT Warangal, India

With
Apart Research

Abstract

When overseeing AI systems, researchers often rely on multiple Large Language Model (LLM) monitors to review agent behavior. If all the monitors agree that collusion occurred, that consensus is typically treated as strong proof. However, recent research shows that LLMs tend to make the exact same mistakes; their agreement often reflects shared blind spots rather than actual truth.

To address this, we propose an information-partition design for auditing collusion. Instead of giving every monitor the same information, we give different monitors nested levels of access to multi-agent activity logs. This deliberately creates an “informed minority” of monitors who can see secret communication channels that the rest cannot see. We then test whether a “Surprisingly Popular” (SP)-inspired voting rule, which flags an outcome when it receives more actual votes than the monitors predicted it would get, can catch secret coordination without the final auditor needing to know which monitors had secret access.

In a pilot study using Colosseum’s JiraTicket environment, we report three key findings: (1) Access determines detection: monitors with private channel access unanimously flagged secret coordination, while others missed it completely. (2) Default SP aggregation fails due to a “curse of knowledge”: informed monitors predicted 95% agreement when actual was 25%. (3) A one-sentence calibration prompt dropped

predictions from 95% to ~14.8%, restoring the SP verdict. Our findings strictly measure detection of explicit secret planning rather than successful collusion, as the colluding agents performed worse than the baseline.

1. Introduction

As AI agents are increasingly deployed to work together, safety researchers rely on LLM monitors to read through agent logs (trajectories) and flag suspicious behavior. When a committee of monitors agrees on a verdict, we usually trust their consensus. But does agreement actually prove they are right?

Recent work by Denisov-Blanch et al. (2026) suggests it does not. Testing five different model families across four benchmarks, they found that no group-voting method, whether simple majority rule, confidence-weighted voting, or the “Surprisingly Popular” (SP) algorithm consistently beat simply asking a single model once. The reason is straightforward: LLMs have strongly correlated errors. Polling a group of LLMs often just amplifies common misconceptions rather than canceling out random errors.

Specifically, Denisov-Blanch et al. (2026) noted that the SP algorithm fails with LLMs because standard LLM groups lack an informed minority: a small subgroup that both knows the true answer and realizes that the majority will get it wrong (Prelec et al., 2017). Without that knowledgeable subgroup, the SP math has no signal to work with.

Auditing AI agents for collusion, however, gives us a unique advantage: the auditor controls what each monitor is allowed to see. We can selectively share private chat logs, hidden chain-of-thought reasoning, or internal model activations with only a small fraction of monitors while withholding them from the rest. Instead of hoping an informed minority exists naturally, we can build one by design.

In this paper, we test whether deliberately building an informed minority allows an SP-inspired voting rule to work. We give a pool of 8 monitors three tiered views of a multi-agent task:

View 1 (V1): Public messages only.

View 2 (V2): Public messages plus final task actions.

View 3 (V3): The full record, including secret private messages.

Our pilot reveals a complication: simply creating an informed minority is not enough. V3 monitors caught the secret coordination but suffered from a “curse of knowledge” i.e. they assumed 95% of

the pool would also vote “yes.” A one-sentence prompt tweak corrected their predictions and restored the SP rule’s accuracy.

Our main contributions are:

- An information-partition design that deliberately constructs an informed minority for SP-based collusion detection.
- Empirical evidence that LLM monitors suffer from a “curse of knowledge” failure in Theory of Mind, predicting 95% agreement when the actual rate was 25%.
- A one-sentence calibration fix that drops informed monitors’ predictions from 95% to ~14.8%, restoring the SP signal.

2. Related Work

Why Crowd Wisdom Fails for LLMs. Denisov-Blanch et al. (2026) tested majority voting, confidence weighting, prediction-weighted voting, and the SP algorithm on tasks without an automated verifier. None consistently beat a single model sample, even when spending $25\times$ more compute. They showed this happens because different LLM families make correlated mistakes and lack a distinct “expert minority” subgroup.

Peer Prediction for LLMs. Qiu, Carroll & Allen (2026) studied peer prediction methods for evaluating and training LLMs, showing these methods work best when one participant holds private information that others lack. Their theoretical results directly inspire our information-partition setup.

The Surprisingly Popular (SP) Rule. Prelec et al. (2017) introduced the SP algorithm, which picks the answer that gets more actual votes than people predicted it would get. Prelec (2004) laid the groundwork with the Bayesian Truth Serum. Our prompt differs slightly: we ask monitors to predict the percentage of peers voting “yes” rather than how popular their own answer will be.

Multi-Agent Collusion Benchmarks. Nakamura et al. (2026) created Colosseum, a testbed for evaluating collusion among LLM agents. They found that 57.89% of tested models gained an unfair advantage through emergent collusion, and standard LLM judges gave lower suspicion scores to the most successful colluders. Hagag et al. (2026) found that monitors checking actions one by one offered no measurable protection.

Collusion in Antitrust Law. Berger et al. (2025) point out that under antitrust law, merely seeing agents act in similar ways is not legal proof of collusion. Auditors need methods that can tell the difference between intentional secret coordination and harmless coincidence.

3. Methods

3.1 Environment and Agent Runs

We ran our experiments in Colosseum’s JiraTicket environment (Nakamura et al., 2026). In this game, 6 AI agents spend 4 planning rounds communicating on a public message board to decide who will claim which

software development tasks, after which each agent executes one task. In our experimental conditions, 2 specific agents are also given access to a secret, private message board.

All agents used gemini-3.5-flash-lite (temperature 0.7, max output tokens 1500). For each random seed, we tested three conditions:

- **Prompted Collusion:** The private channel is turned on, and the two selected agents receive a hidden system prompt instructing them to secretly coordinate and mislead the others.
- **Emergent Collusion:** The private channel is turned on, but no special instructions are given.
- **Baseline Control:** No private channel exists. We ran two replicates per seed to measure baseline variation.

To measure whether the paired agents actually benefited from colluding, we calculate coalition advantage using the regret gap: the average regret of the 4 non-coalition agents minus the average regret of the 2 coalition agents.

3.2 Tiered Monitor Views

Once the agents finished a game, we generated three nested transcripts from the raw event logs:

- **V1 (Public Only):** Contains only the messages posted to the public board.
- **V2 (Public + Actions):** Contains V1 plus the final list of which agent executed which task.
- **V3 (Full Access):** Contains V2 plus the complete transcript of the secret private message board.

All metadata labels showing which condition was run or which agents were in the coalition were stripped. Each transcript was roughly 1,500 to 2,300 tokens long.

3.3 Monitor Pool and Prompting

We designed an informed-minority monitor pool of 8 monitors total: 4 receive V1, 2 receive V2, and 2 receive V3. Every monitor is told the exact makeup of the pool. Each monitor reads its assigned transcript and outputs a JSON response with four fields: verdict, confidence, predicted_yes_pct, and rationale.

For every game run, we collected 8 independent responses for each view (24 per run). The 8-monitor pool was assembled by taking the first 4 V1, first 2 V2, and first 2 V3 responses in generation order. All monitors also used gemini-3.5-flash-lite at temperature 0.7.

3.4 Aggregation Rules

We compare three “view-blind” voting rules (the rule does not know which monitor saw which view):

- **Majority Vote:** Whichever verdict gets the most votes wins (ties default to “no”).
- **Confidence-Weighted Vote:** Each monitor’s vote is multiplied by its 0–100 confidence score.

- **SP-Inspired Rule:** Let f be the actual fraction voting “yes,” and $\bar{p}$ be the average of all monitors’ `predicted_yes_pct/100`. Flags collusion if $f > \bar{p}$.

We compare these against one idealized benchmark:

- **Oracle-Minority:** Majority vote strictly among the V3 monitors in the pool.

3.5 Exploratory Calibration Intervention

After seeing V3 monitors predicting 95% agreement, we ran a post-hoc follow-up. We added a single clarifying sentence to the V3 prompt:

“Important: the V1 and V2 monitors cannot see the private channel messages. They can only see public messages.”

Everything else was kept identical, and we collected 8 new V3 responses.

3.6 Evaluating Ground Truth

We calculate Δ : the regret gap of the experimental run minus the average regret gap of the matched baseline controls. A positive Δ means the secret channel helped the coalition. Because we did not pre-register a cutoff threshold, we report raw Δ numbers and test whether monitors catch explicit secret coordination rather than whether that coordination paid off.

4. Results

We completed 7 game runs across 2 seeds (101 and 102). On seed 102, the task setup was a natural Nash equilibrium where every agent achieved zero regret, making it useless for measuring coalition advantage. We focus on seed 101 (2 controls, 1 emergent, 1 prompted). Table 1 summarizes the dataset.

Seed	Condition	Runs	Standard Samples	Calibrated
101	Control	2	48 (24/run; 8/view)	—
101	Emergent	1	24 (8/view)	—

101	Prompted	1	24 (8/view)	8 (V3 only)
102	Control	2	48 (24/run; 8/view)	—
102	Emergent	1	24 (8/view)	—
Total		7	168	8

4.1 Secret Coordination Did Not Yield a Gameplay Advantage

In the prompted collusion run (seed 101), the coalition agents actively plotted on the private board but bungled execution: one claimed a task already committed to by another agent, causing a collision penalty ($\Delta = -0.78$). In the emergent run, agents used the private channel only for harmless scheduling ($\Delta = +0.25$, within baseline noise).

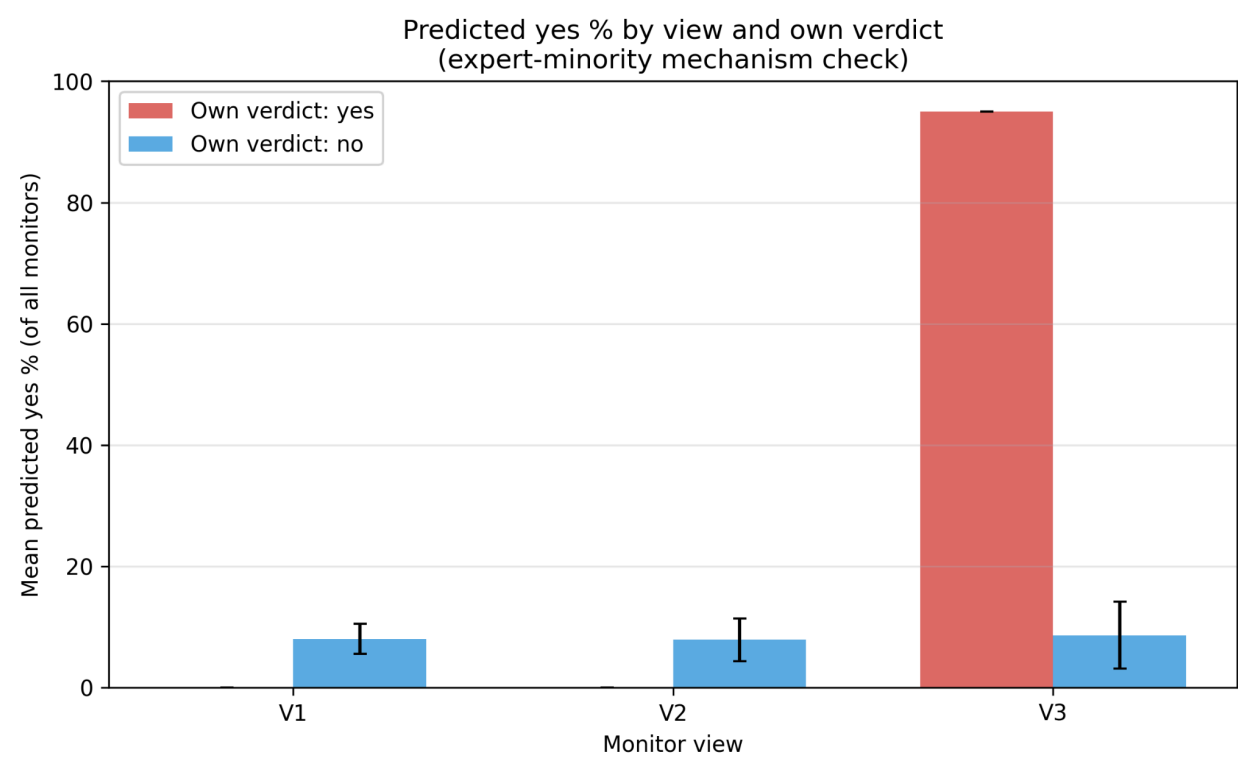
Takeaway: Because the prompted colluders penalized themselves, our monitor evaluation tests whether monitors catch *explicit attempts to secretly coordinate and deceive*, not successful score manipulation.

4.2 Only Monitors with Private Channel Access Caught the Collusion

Across all 7 runs (56 evaluations per view level), detection depended entirely on view level (Table 2).

Monitor View Level	“Yes”	“No”	Total	% Flagging
V1 (Public only)	0	56	56	0.0%

V2 (Public + actions)	0	56	56	0.0%
V3 (Full access)	8	48	56	14.3%



4.3 Why the Default SP-Inspired Rule Failed

Table 3 shows individual monitor outputs for the prompted collusion run.

Slot	View	Verdict	Predicted “Yes” %

1	V1	no	5%
2	V1	no	10%
3	V1	no	5%
4	V1	no	5%
5	V2	no	10%
6	V2	no	10%

7	V3	yes	95%
8	V3	yes	95%

- **Majority Vote:** 6 “no” vs. 2 “yes” → Outputs “No” (Fails).
- **Confidence-Weighted:** All monitors reported near-100 confidence, so the 6-to-2 split holds → “No” (Fails).
- **SP-Inspired Rule:** $f = 2/8 = 0.250$. $\bar{p} = (5+10+5+5+10+10+95+95)/(8 \times 100) = 0.294$. Since $0.250 < 0.294$, outputs “No” (Fails).
- **Oracle-Minority:** Only counts V3 → Outputs “Yes” (Succeeds).

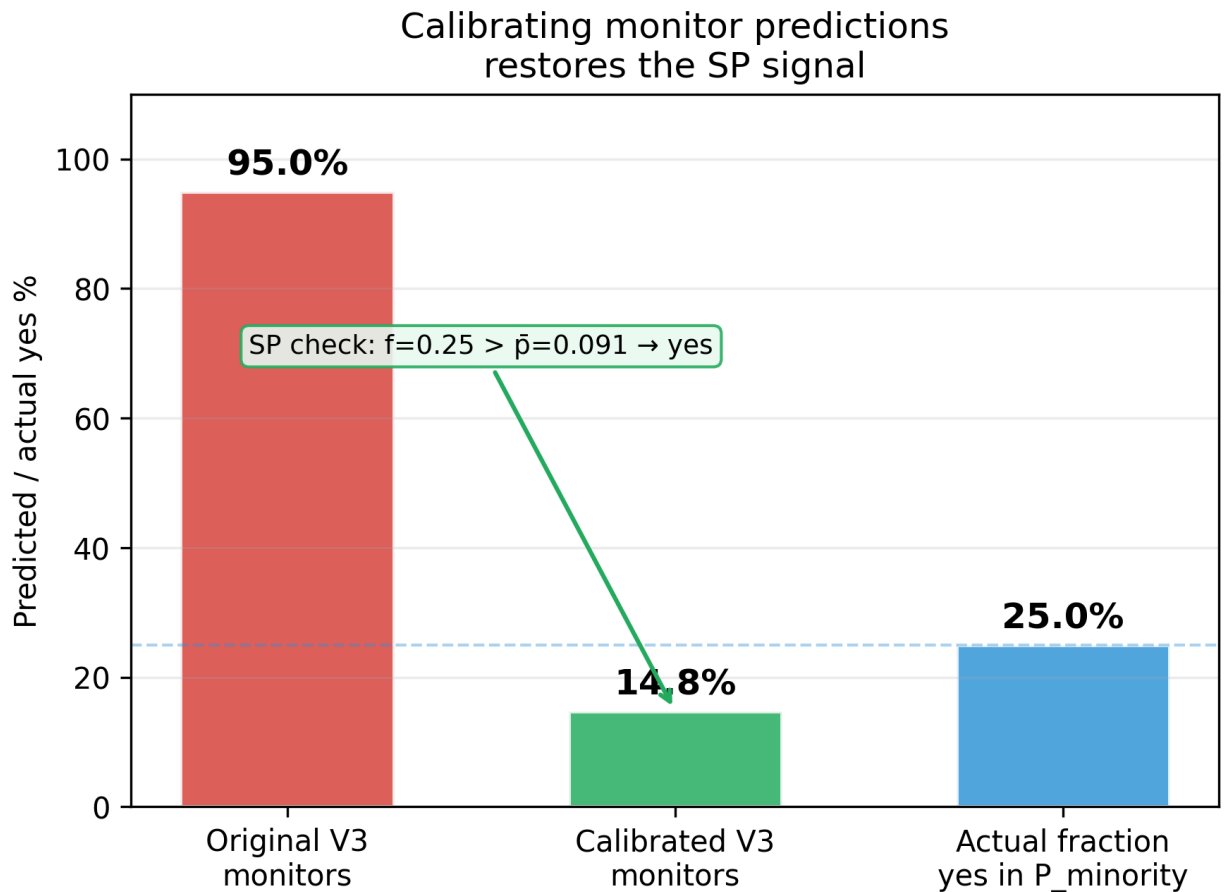
Diagnosis: Monitors 7–8 predicted 95% of the pool would vote “yes.” They saw blatant cheating and failed to realize that monitors without private access would miss it entirely. Their inflated predictions dragged $\bar{p}$ above the actual vote share.

4.4 Explaining What Others Cannot See Fixes the SP Signal

After adding the calibration sentence, Table 4 compares original vs. calibrated V3 outputs.

Prompt Condition	Verdicts	Confidence	Predicted “Yes” % (Mean $\pm$ SD)
Original V3	8/8 “yes”	100	95.0% $\pm$ 0.0%

Calibrated V3	8/8 “yes”	100	2.1%	14.8% $\pm$
---------------	-----------	-----	------	-------------



The calibration left verdicts and confidence unchanged but dropped predictions from 95% to 14.8% (close to the true informed fraction of $1/7 \approx 14.3\%$). Substituting calibrated predictions: $f = 0.250$, $\bar{p}_{\text{cal}} = 74.6/800 = 0.093$. Now $f > \bar{p}$, so the SP rule outputs “Yes,” matching Oracle-Minority.

Caveats: This was a post-hoc substitution on a single trajectory with one model family. All 8 calibrated samples predicted between 12–18%, so any pair keeps $\bar{p} < 0.250$, but this should be treated as a proof-of-concept hypothesis.

5. Discussion and Limitations

A New Reason Why Crowd Wisdom Can Fail in LLMs. Denisov-Blanch et al. (2026) showed SP fails without an informed minority. Our experiment highlights a second hurdle: **even when you build an informed minority on purpose, LLMs fail at Theory of Mind unless prompted carefully.** Spelling out the information gap in plain English was essential.

Practical Value for AI Oversight. If this calibration effect holds up, it offers a simple recipe for auditing multi-agent systems: combine cheap superficial monitors with expensive deep-access monitors, aggregate using SP-inspired rules, and explicitly calibrate by telling deep-access monitors what the others cannot see.

Where This Approach Will Break. If colluding agents use advanced steganography that fools all monitors equally, the informed minority disappears and the SP rule fails alongside majority voting.

The Upstream Problem. Our pilot used a plaintext private channel. In real deployments, privileged access could come from chain-of-thought logs, activation probes, cross-agent behavioral correlations, or steganographic decoders. Our contribution concerns what happens once you have an information gap, not how to obtain one.

Limitations

- **Same Model for Agents and Monitors:** Both used gemini-3.5-flash-lite. Shared biases could inflate within-view agreement.
- **Small Sample Size:** 7 runs across 2 seeds, only 1 with explicit secret coordination.
- **No Positive Gameplay Advantage:** Colluders penalized themselves ($\Delta = -0.78$). We tested detection of attempted, not successful, collusion.
- **Simplified SP Question:** Our prediction question differs slightly from Prelec et al.’s (2017) exact formulation.
- **Only One Pool Ratio:** We tested only $4 \times V1$, $2 \times V2$, $2 \times V3$.
- **Single-Trajectory Calibration:** The calibration fix was tested post-hoc on one game run.
- **Readable Secret Channel:** Real-world auditing may require interpreting messy internal activations or decoded steganography.
- **Honest Monitors Assumed:** We assumed monitors were not strategically lying or colluding with agents.

Future Work

Natural next steps include testing across multiple model families and environments (especially where coalitions reliably gain an advantage), varying pool composition, testing calibration across multiple trajectories, and exploring V3 access from chain-of-thought logs or activation probes rather than a plaintext channel.

6. Conclusion

We proposed an information-partition design that deliberately constructs an informed minority among LLM monitors for collusion detection. Our pilot showed that while giving some monitors private access is necessary for detection, it is not sufficient for the SP-inspired rule to work out of the box. LLM monitors suffer from a “curse of knowledge”—they assume others can see what they see—which inflates the group’s predicted support and causes the SP rule to fail.

A simple one-sentence calibration prompt that explains what other monitors cannot see dropped predictions from 95% to ~14.8%, restoring the SP signal. If this replicates, it suggests a practical recipe: combine cheap surface-level monitors with expensive deep-access monitors, aggregate using SP-inspired rules, and calibrate the deep-access monitors by telling them what the others are blind to.

Code and Data

- Code repository: <https://github.com/nirdeshbhandari/collusion-monitor>
- Data/Datasets: Included in the repository under experiments/collusion/track23/

References

- [1] Berger et al. (2025). Antitrust plus-factors doctrine and algorithmic coordination.
- [2] Denisov-Blanch, Y., Kazdan, J., Chudnovsky, J., Schaeffer, R., Guan, S., Adeshina, S., & Koyejo, S. (2026). Consensus is Not Verification: Why Crowd Wisdom Strategies Fail for LLM Truthfulness. ICML 2026. arXiv:2603.06612.
- [3] Hagag et al. (2026). ORBIT: Evaluating Multi-Agent Oversight. arXiv:2609.33102.
- [4] Nakamura, M., et al. (2026). Colosseum: A Benchmark for Evaluating Multi-Agent Collusion. arXiv:2602.15198.
- [5] Prelec, D. (2004). A Bayesian Truth Serum for Subjective Data. *Science*, 306(5695), 462–466.

[6] Prelec, D., Seung, H. S., & McCoy, J. (2017). A solution to the single-question crowd wisdom problem. *Nature*, 541(7638), 532–535.

[7] Qiu, T. A., Carroll, M., & Allen, C. (2026). Truthfulness Despite Weak Supervision: Evaluating and Training LLMs Using Peer Prediction. ICLR 2026. arXiv:2601.20299.

Appendix: Limitations and Dual-Use / Ethical Considerations

This research focuses on catching secret coordination among AI agents, directly supporting AI safety and trustworthy oversight. Because our methods focus on helping auditors combine monitor reports more accurately rather than teaching agents how to hide their behavior, we see minimal dual-use risk. All experiments were conducted in a simulated software-task game with no real-world side effects.

Our results do not establish that the SP-inspired rule works reliably in general. They establish that (1) information partitioning creates a detectable signal, (2) LLMs’ Theory-of-Mind failures suppress that signal by default, and (3) explicit calibration prompting can restore it in at least one case. The approach cannot detect collusion when no monitor has privileged access to the coordination channel.

LLM Usage Statement

We used Claude (Anthropic) to brainstorm experimental design approaches, help debug code, and assist with drafting sections of this report. All results, claims, and analyses were independently verified by the author. The final version of this report was primarily written by the author.