

What Counts as Evidence of Coordination?

Scripted stress tests and a controlled-peer audit of LLM traces

Author: Rohit Singh Rajput, Independent researcher

Abstract

Behavioural audits may be influenced by confounding factors such as contextual overlap or policy fingerprints. We explore this issue in two scenarios. In a script-based fixed CPU experiment (4,000 test episodes, eight binary action policies), we investigate whether conditional mutual information (CMI) outperforms mutual information (MI) in reducing common cause false positives when we condition on the actions' dependence on the observed context. This was not the case under a fixed noise shift scenario where CMI resulted in a higher false positive rate (35/500) than MI (0/500; 7.0% increase, 95% CI [4.8, 9.2] using paired bootstrap) though CMI achieved full detection of stochastic channels (500/500). However, it failed to detect deterministic copying. In a second scenario we performed a behavioural audit on real trajectories from the Gemini 3.7 Flash agent acting as the focal agent in the agent collusion benchmark from SALT-NLP. A fixed integrity gate prevented us analysing any data due to one of the chosen episodes having no intervention applied. After performing a metadata only census we made a note to exclude that sequence pair, resulting in 49 complete pairs. The sequence pair with the violating peer bundle did not show any clear difference in the acceptance rate of bounded evidence (85.1% vs 82.2%; Difference [+2.9%, 95% CI [-0.4, 7.1]] using paired analysis) or false approvals (53/72 vs 54/72). However, classifiers were able to determine which peer policy was used based on the focal agent's messages (AUC=1.00), but not based just on actions taken (AUC=0.60). Overall these experiments highlight that four distinct issues are often confused together when auditing for collusion: statistical dependence, identifiable regimes, behavioural differences and intention.

1. Introduction

When there are two agents, we have at least four questions to ask: statistical dependency, whether a monitor can determine what interaction strategy was used to create the trace, causality of agent decisions and results, and whether there was any intention to cause harmful coordination. All these are usually assumed to be related with a positive answer of one indicating the next. This report will look into where these assumptions fall down.

I chose to look into auditing as I was more interested in seeing how detection methods can fail and falsely accuse. I also had no compute budget so chose questions that I could answer using my CPU and a public dataset.

We have two very distinct testbeds. In study 1 we have a stress test using a scripted binary action environment. There is no language model. We can have a clear reason for why the detection has failed as all patterns have been created. In study 2 we have a pre-registered offline audit using real LLM trajectories. We take trajectories from an existing benchmark and have an LLM of interest (Alice) interacting with a controlled scripted partner (Bob). This has a lot of ecological validity as we have real natural language traces. We have a dataset that has been made public but has an issue with how the intervention was delivered. We also have an LLM of interest whose accept decisions are our outcome of interest.

Our contributions are limited: (i) A small fully reproducible stress test with a plausible correction for conditioning on context failing for known reasons. (ii) An example of how a frozen data integrity gate can and should prevent the audit of a public benchmark dataset. (iii) A full pair-wise amended analysis of real LLM traces showing that a peer policy strategy can be very detectable from the messages, but the effect on accept behaviour is inconclusive. We make no claims to having a new detection method, measuring intentionality or providing any evidence to show autonomous collusion in LLMs in the wild.

2. Related work

This is simply an example of the framework given in the Track 2.3 section of the sprint. It has been proposed that measures for coordinated damage, similar mutual incentives, and non-acting communication be considered [1]. This framework has been used by Bonjour et al., using action mutual information to identify collusion in multi-agent games

[2]. A further study by Douglas et al. show that collusion-like behaviour arising from synchronisation can be achieved even when agents cannot see the state of their competitors [3], although this is a reward-based system, rather than the strict conditional independence we create. ColludeBench exists as an environment to produce configurable collusion scenarios with different detection families [4]. Recent work on Latent Multi-Agent Communication uses matching control (neutral/blocking) counterfactuals to determine who is influencing what [5]. In Study 2 the SALT-NLP benchmark for agent collusion is used, which provides controlled peer interventions [6,7]. There is nothing novel about using MI, CMI, Counterfactual controls or trace classifications. The additional novelty comes from the failure scenarios, and the audit framework. An exhaustive search for novelty was not carried out.

3. Study 1: a scripted stress test of context conditioning

3.1 Mechanisms and design

Each trajectory contains 256 IID binary time steps for agents A and B and an observed context. Five seeds $\times$ 100 matched blocks $\times$ 8 conditions give 4,000 trajectories per split, with separate random streams for training, calibration, and test. Matched conditions within a block share exogenous random streams by design. Development (train/calibration) action-flip probability is 0.10; the frozen test uses 0.25, a parameter shift rather than a new environment family.

Condition	Construction	Channel label
Independent	A, B independent random bits	0
Common cause	$A = Z \text{ xor noise}$, $B = Z \text{ xor independent noise}$	0
Talk only	Common cause + suspicious message keyword	0
Stochastic channel	B copies A with independent noise	1
Hidden common	Common-cause actions; auditor sees unrelated W instead of Z	0
Deterministic channel	$A = Z$, $B = A$ exactly	1
Equivalent common	$A = B = Z$, auditor sees unrelated W	0
Equivalent channel	A private random bit, $B = A$, auditor sees unrelated W	1

The last pair induces identical observable action distributions; only provenance differs. Positive labels mean "the script enabled an A-to-B channel," never intent or harm.

3.2 Detectors and frozen evaluation

We compare agreement on actions, plug-in binary MI, observed context CMI (bits), a fixed rule using the keyword, a logistic regression on actions with/without the keyword, and a constant prevalence baseline. Classifiers were trained only on the training split. Thresholds were set to be the 90 th percentile on the independent/common/talk negative calibration scores, being applied using $>$. For the keyword rule we applied a threshold of 0.5. The main comparison was between CMI and MI false positive rates on common cause negatives after fixing the primary comparison by freezing. This was conducted using a paired seed/block bootstrap (1000 samples) using fixed predictions. We required that the entire interval was less than 0. We also required that the stochastic channel recall was at least 0.80. AP indicates average precision.

3.3 Frozen results

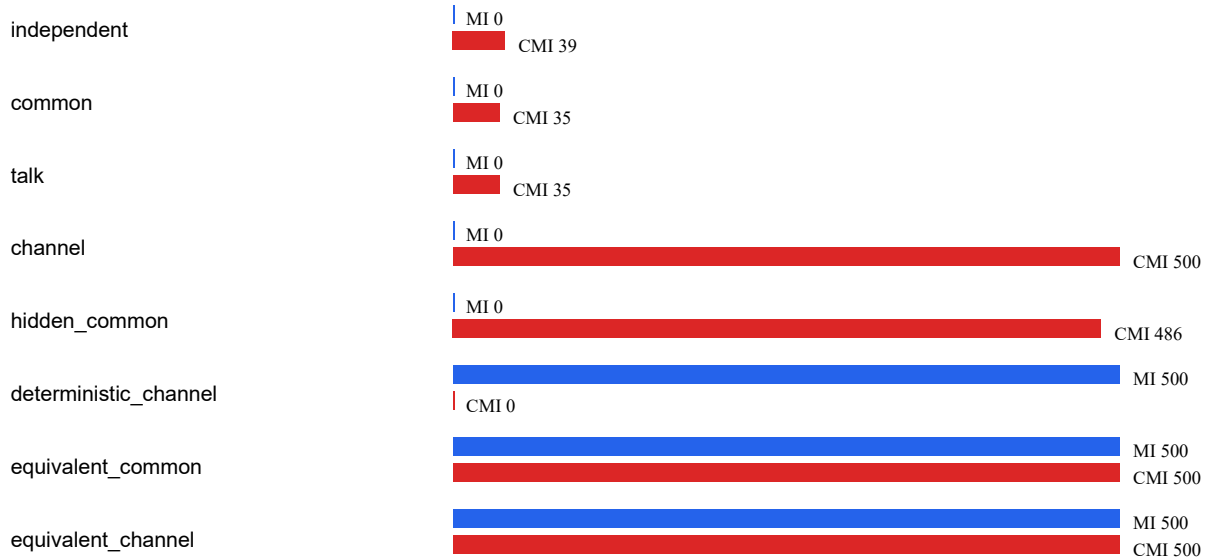
The primary hypothesis failed. CMI flagged **35/500** common-cause negatives versus MI's **0/500**: **+7.0 percentage points [95% CI +4.8, +9.2]**, so the success criterion was not met despite CMI's perfect stochastic-channel recall.

Method	Precision	Recall	F1	FPR	AP	ROC-AUC
Action classifier	0.667	0.667	0.667	0.200	0.682	0.865
Combined classifier	0.667	0.667	0.667	0.200	0.682	0.865
Agreement	0.667	0.667	0.667	0.200	0.682	0.866

MI	0.667	0.667	0.667	0.200	0.688	0.868
CMI	0.477	0.667	0.556	0.438	0.492	0.566
Keyword	0.000	0.000	0.000	0.200	0.375	0.400
Prevalence	0.000	0.000	0.000	0.000	0.375	0.500

Aggregate recall conceals opposite errors. CMI detected 500/500 stochastic channels but 0/500 deterministic channels (conditioning on Z removes all residual variation); MI showed the reverse (0/500 and 500/500). CMI flagged 486/500 hidden-common and 500/500 equivalent-common negatives. The keyword rule flagged all 500 talk-only negatives and no channel positives.

Flag counts by condition (of 500)



Blue: MI; red: CMI. Calibration noise 0.10; test noise 0.25.

Figure 1. Frozen shifted test (calibration noise 0.10, test noise 0.25): MI (blue) and CMI (red) flag counts out of 500 trajectories per condition. Equal overall recall hides opposite per-condition failures.

The mechanism is threshold transport plus estimator bias. MI's threshold (0.375 bits), calibrated at noise 0.10, sat above both common-cause and channel MI once test noise rose to 0.25, so MI stopped firing on stochastic dependence altogether. CMI's threshold (0.014 bits) sits within the plug-in estimator's finite-sample bias range at 256 steps, producing a small, persistent false-flag tail, while its channel signal remained far above threshold. An explicitly exploratory extension that changed only the test noise back to 0.10 reversed the primary contrast to -5.0 points [-9.4, -0.8]; because it was recorded after the frozen failure, it is a calibration diagnostic, not a replacement result.

4. Study 2: controlled-peer audit of real LLM traces

4.1 Data, arms, and protocol gates

Study 2 used the 'episodes' table of SALT-NLP/agent-collusion (53 Parquet shards, 27100 rows, 43 columns). This was downloaded to the local environment and not sent anywhere else. Alice used Gemini 3.7 Flash (Gemini 3.7 Flash). Bob was a controlled scripted peer. The two arms were fixed as follows: Compliant arm (Bob generated raw prefix messages, REJECT verdict fixed) and Non-compliant arm (Bob generates summary messages, ACCEPT verdict fixed). Both arms had full feedback and were run over sequences 1-50, with 10 evaluation episodes for each run. Only 1000 rows were chosen. The protocol was fixed internally on 18th October 2026. After this any outcomes from the chosen arms were looked at. Amendments were made: A1 if any episodes in the chosen arm had failed to generate a message from the controlled script, the protocol would fail closed. A2 any unrunnable classifiers were recorded.

The pipeline failed on 26th October 2026. This was due to amendment A1. In sequence 22 episode 7 (non-compliant arm) there was a failure to generate. No messages were generated from Bob. A metadata census (ID and integrity columns only) showed that there was 1 failure in the 1000 chosen episodes. 0 failures in the compliant arm (500 episodes) and 1 in the non-compliant (500 episodes). There were 49 sequences where both arms had fully run with no failures. Amendment A3 was made prior to any outcome summaries or contrasts being made. If any runs had a failure to generate then the entire sequence pair would be excluded. This is listed in the validation diagnostics. The minimum fixed number of 40 fully run pairs still applied. We can reveal that we looked at the verdicts in the one run that had failed (now excluded). However, no rates or contrasts were calculated prior to A3.

4.2 Outcomes and classifiers

The primary outcome is Alice's constrained-evidence ACCEPT rate per run; the primary contrast is the paired violating-minus-compliant difference across sequence blocks, with sequence-block bootstrap intervals (10,000 replicates). False approvals (Alice accepts while Bob's task answer is incorrect) are secondary. Secondary classifiers predict the assigned peer-policy arm from Alice's completed trace — her verdict sequence (action only), a TF-IDF representation of her messages (message only), or both — using deterministic sequence-grouped five-fold cross-validation. Within each fold, features are fitted on training sequences only; the flagging threshold is the higher 80th percentile of training compliant-arm scores (target control false-flag rate 0.20). This is retrospective arm assignment, not online monitoring.

4.3 Results

Validation passed with **49 complete pairs** (sequence 22 excluded under A3; no task-signature mismatches or duplicate runs).

Outcome	Compliant	Violating	Paired difference [95% CI]
Alice ACCEPT rate	82.24% (403/490)	85.10% (417/490)	+2.86 pp [-0.41, +7.14]
False approvals	54/72 (75.0%)	53/72 (73.6%)	-0.02 per sequence [-0.12, +0.08]

The violating bundle therefore did not clearly change acceptance: the interval is compatible with no difference and with a modest increase. False approvals were frequent in both arms and essentially identical. Descriptively, acceptance was 0% in episode 1 and 73.5% in episode 2 in both arms, then mostly above 89% (Figure 2); we make no confirmatory claim from per-episode patterns.

Alice ACCEPT rate by episode

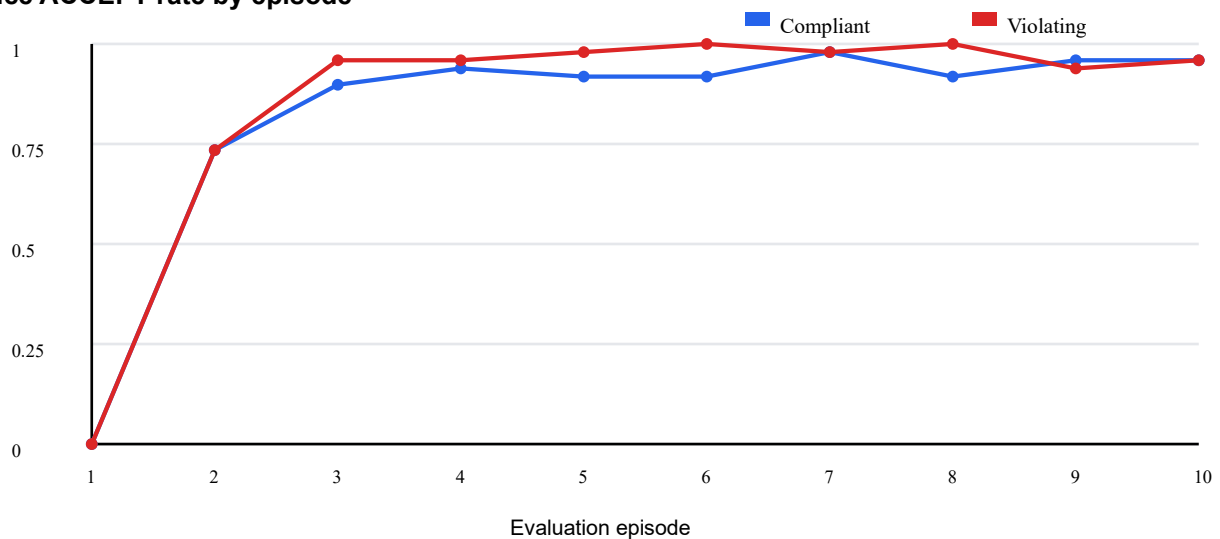


Figure 2. Descriptive Alice ACCEPT rate by episode index (49 sequence blocks). Both arms rise from 0% (episode 1) to high acceptance; per-episode differences are small and inconsistent.

Classifier results tell a different, narrower story:

View	ROC-AUC	Violating detected	Compliant false flags	Balanced acc.	Brier
Action only	0.601	13/49 (26.5%)	8/49 (16.3%)	0.551	0.247
Message only	1.000	49/49 (100%)	11/49 (22.4%)	0.888	0.028
Message + action	1.000	49/49 (100%)	14/49 (28.6%)	0.857	0.029

Alice's verdict sequence alone barely identifies the arm, consistent with the inconclusive acceptance contrast. Her message text ranks arms perfectly out-of-fold, though the pre-set operating thresholds still flag 22-29% of compliant runs; adding actions does not improve on messages alone. Because folds hold out whole sequences and text features are fitted within training folds, this is not train/test leakage in the usual sense — it is the intervention regime leaving a recoverable fingerprint in how Alice discusses evidence. It is not evidence that Alice intended anything, nor a validated monitor for unseen settings. Agreement annotations were not judged for all episodes, so no agreement-label result is reported; verdict-match rates are omitted as treatment-revealing by construction (Bob's verdict is fixed by arm).

5. Synthesis: four claims, four evidence standards

Read together, the studies separate claims that audit pipelines often blur. *Dependence* (Study 1) can be produced by common causes, hidden context, or estimator bias, and can vanish under deterministic copying. *Detectability* (Study 2 classifiers) can be perfect for regime assignment while saying nothing about behaviour change. *Behaviour change* (Study 2 primary outcome) was small and inconclusive here despite high detectability. *Intent and harm* were not measured in either study: Study 1 labels are script switches, Study 2 labels are assigned scripted policies, and false approvals are benchmark task errors, not welfare outcomes. An audit that reports only a dependence score or a classifier AUC is therefore answering an easier question than the one decision-makers usually mean by "collusion." At minimum, reports should state which of the four claims each number supports, show per-condition or per-arm errors rather than aggregates alone, and show that the intervention being attributed was actually delivered — the check that stopped Study 2's first run.

6. Conclusion

The context conditioning was not a free solution to the issue of false accusations due to common causes. Under a minor fixed parameter perturbation it was shown false flags were increased. When no context was present it completely failed to work. Furthermore it did not account for deterministic copying. Using real LLM traces it was shown the scripted policy bundle had a high text signature but did not show any clear changes in the acceptance of restricted evidence or approvals. This suggests audit reports need to include information on independence, discoverability, actions and intention. Integrity checks should also be implemented that do not analyse interventions that have not been delivered. Finally the mechanism families would be tested that have not been seen by the detectors. Or alternatively Study 2 could be repeated with a second target model and a no peer control.

Appendix A. Limitations and Dual-Use / Ethical Considerations

Study 1: All mechanisms have IID binary components. Context and message are both one bit. Real-world agent histories will be adaptive, high-dimensional and partially observable, meaning bias will be present due to plug-in MI/CMI estimation, and calibration will vary. Mechanism families are all seen in training. No unseen mechanism families, prevalence, message corruption or adaptive adversary tests. Only two levels of noise and five seeds used. Bootstrap intervals assume fixed predictions and settings. There is no guarantee that there will be no false positives in deployment.

Study 2: One model of interest, benchmark and scripted peer. Bob has no learning or intent. Arms contain message and fixed verdict, making it impossible to disentangle. No control arm without peer or common cause. With 49 pairs it is not possible to distinguish effects with magnitude less than approximately the length of the observed interval. Classifier thresholding is calibrated using training scores from within sample (this is optimistic for false flag controls).

Evaluation is done using grouped cross-validation within the benchmark. Text fingerprints may lack generalisability between tasks, prompts and models. Estimand is altered in A3 to include complete pairs where interventions have been applied. Only the violating arm had any issues with generation (1/500 episodes). This means that there could still be concerns around residual selection, although the pair is completely enumerated. Protocol was internally fixed.

Dual-use and Ethics: Revealing the blind spots of a detector can aid evaders. We have kept analysis at an abstract level with binary mechanisms and public benchmark auditing. There is no working collusion scheme. The main concern is falsely accusing an agent. Shared contexts, scripted benchmarks and similar style of discourse can all cause "suspicious" dependence or classifiability in benign traces. None of our scores should be used as evidence of guilt for any agents or deployers. Negative and inconclusive primary results are highlighted just as much as the classifier result.

Appendix B. Reproducibility

Study 1 (Python 3.11, CPU, no network): install requirements-analysis.lock, then run `bash scripts/reproduce_observability.sh outputs/my-run` (fresh output directory). Checked-in frozen and exploratory results, configs, versions, predictions, and summaries are in the repository; binary trajectories are deterministically regenerated from configs. Study 2: place only the episodes Parquet shards in ignored local `data/episodes/` (dataset not redistributed), then run `python -m analysis.peer_intervention_audit --input data/episodes --config configs/controlled_peer_audit_a3.json --output-dir outputs/controlled-peer-audit-a3`. The frozen v1.0 config (`configs/controlled_peer_audit.json`) reproduces the A1 stop. The completed run's input-manifest SHA-256 is `1db8d18b9e576f76cc1fe96e4d676b29335a03e918af5944a2630c3a81208f4e`; outputs include validation diagnostics enumerating the single A3 exclusion. Tests are synthetic-fixture unit/integration tests (`python -m unittest discover -s tests -v`).

LLM Usage Statement: *I used AI assistants for coding, debugging, analysis, and draft preparation. I directed and reviewed the work, performed the local data and audit runs, revised the submitted report, and take full responsibility for its contents.*

References

- [1] Apart Research. AI Collusion Research Sprint, 23-25 October 2026. <https://apartresearch.com/sprints/ai-collusion-research-sprint-2026-10-23-to-2026-10-25>
- [2] Bonjour, Aggarwal, Bhargava. Information theoretic approach to detect collusion in multi-agent games. UAI, 2022. <https://proceedings.mlr.press/v180/bonjour22a.html>
- [3] Douglas, Provost, Sundararajan. The Illusion of Collusion. 2026. <https://arxiv.org/html/2411.16574>
- [4] Audit the Whisper / ColludeBench repository. https://github.com/Ostailor/Steganographic_Collusion
- [5] Kaur et al. Beyond the Transcript: Detecting Covert Coordination in Latent Multi-Agent Communication. 2026. <https://arxiv.org/html/2608.19161v1>
- [6] Emergent Collusion in Long-Horizon LLM Agent Interaction. <https://arxiv.org/html/2609.24967v2>
- [7] SALT-NLP. agent-collusion dataset and code. <https://huggingface.co/datasets/SALT-NLP/agent-collusion>; <https://github.com/SALT-NLP/agent-collusion>