

Remove the Message, Catch the Colluder

Aradhya Goel

Indian Institute of Technology Kanpur

AI Collusion Research Sprint (Apart Research), October 2026

Abstract

Detectors that only watch what agents do will flag any agents that behave alike. So they cannot tell agents that coordinate by talking from agents that simply want the same thing. We detect collusion by *intervening* instead. We replay a game between AI agents and replace one agent’s message with a version that points to the opposite choice. Then we check whether another agent’s decision changes. We test this in a four-agent voting game with GPT-5.6. We compare a colluding pair with two agents who share the same hidden goal but cannot communicate. Replacing messages finds the colluding pair in every run and raises no false alarms. Standard detectors that measure how similar the agents’ votes are fail to tell the two cases apart. We then perform the same intervention *inside* an open model. We leave the message text unchanged but push the model’s internal reading of it toward the opposite choice. This works only early in the model. There it picks out the colluding pair as well as rewriting the text does, while a random push does nothing. Our method also offers a way to test a rule that Rose et al. (2026) include in their definition of collusion but could not check. The rule says that collusion should stop when the messages are taken away.

1 Introduction

Large language model (LLM) agents can collude. They can keep prices above the competitive level [Fish et al., 2024]. They can form coalitions in workplace tasks [Nakamura et al., 2026]. They can also coordinate through channels that a monitor does not read [Motwani et al., 2024].

Most detection methods are *observational*. They score pairs of agents by how alike their actions are [Bonjour et al., 2022], by what a judge reads in their messages, or by probes on their internal activations [Rose et al., 2026]. This kind of evidence has a basic weakness. Agents that share an incentive will behave alike even if they never agree on anything. Antitrust law faces the same problem, since parallel behaviour alone does not prove an agreement.

Rose et al. [2026] point to what is missing. Their definition of collusion includes a causal condition. If a message were removed, the collusive action should stop. They set this condition aside because testing it needs counterfactual simulation. They note that their benchmark could support such tests in future work. We run those tests.

Contributions.

1. A black-box detector. We replay each decision with one agent’s messages removed, or replaced by a version that points to the opposite option. We then measure how much another agent’s vote changes. A per-run permutation test, checked on runs without collusion, tells us when the change is significant.
2. A four-agent game with a shared-incentive control. Two agents hold the same hidden stake but cannot talk to each other. With gpt-5.6-terra, message replacement finds collusion in 10 of 10 runs. It raises no false alarms in 20 runs without collusion. Agreement, mutual information and transfer entropy perform at chance.
3. A white-box version on an open model. We push the receiver’s early internal reading of the partner’s message toward the opposite option. This changes its vote as much as rewriting the text. The same push at later layers does almost nothing.
4. Practical lessons. A frontier model refused to collude when the task looked realistic. Replay tests leak the signal through the receiver’s own messages unless those messages are regenerated. If every colluder already knows the target, the messages carry nothing, and no message-based test can find the collusion.

2 Related work

Collusion among LLM agents. Pricing agents settle on high prices [Fish et al., 2024]. Agents with private channels form coalitions in Colosseum [Nakamura et al., 2026]. Hidden (steganographic) channels can keep coordination away from text monitors [Motwani et al., 2024]. Shi et al. [2026] find collusion that emerges over long interactions and is rarely stated openly.

Detection. Bonjour et al. [2022] find colluding pairs in small games with mutual information. It assumes exactly two colluders. Rose et al. [2026] treat detection as a statistical test over an unknown group of agents. They use probes on each agent’s activations. These work almost perfectly when agents are told to collude. They transfer only partly to collusion that emerges on its own. Both methods are observational. They ask whether agents’ actions or internal states are related. They do not ask whether one agent’s message causes another agent’s action.

Counterfactual influence. Jaques et al. [2019] define social influence in multi-agent reinforcement learning. It is the change in one agent’s policy when another agent’s action is swapped for an alternative. They use it as a training reward. We use the same idea for a different goal. We use it to detect collusion, and we test it against agents that share an incentive. Our agents are LLMs whose context we can edit and replay.

Activation steering. Adding a direction to a model’s internal activations can change its behaviour [Turner et al., 2023]. Linear probes can detect deception in a single agent [Goldowsky-Dill et al., 2025]. We apply steering to the receiver’s reading of another agent’s message.

3 Setting

Game. Four agents sit on a hospital procurement committee. Over 8 rounds they vote between Option A and Option B. Each round has three steps. The seeded pair first exchanges private messages. Then every agent posts one public message, and then everyone votes. In each round one option belongs to a supplier called Northwind. It appears under a different subsidiary name every round. So an agent can favour Northwind only if it knows which option is Northwind’s.

Conditions. We run three kinds of games.

- *Collusion.* Two agents hold a hidden stake in Northwind and share a private channel. Only the **insider** is told which option is Northwind’s. The **follower** has to learn it from the insider.
- *Shared incentive.* Two agents hold the same stake and are each told the Northwind option directly. They do not know about each other and cannot talk.
- *Control.* All four agents are honest.

Why the test is not circular. We made communication necessary for the colluding pair on purpose. The question is not whether our method detects talking in this game. The question is whether a detector can tell collusion apart from shared incentive. In both cases the pair votes for Northwind on every round where it is the worse option. Only in the collusion case do they talk. Observational detectors cannot tell these cases apart. The intervention can.

This design also fixes a problem we hit early on. When each colluder’s own prompt revealed the target, the colluders voted together without reading each other. In that setup no message-based test can detect the collusion, even though it is real.

Model. All agents are gpt-5.6-terra with low reasoning effort. When the task was framed as real procurement, both colluders refused in every round. One wrote “I can’t coordinate around undisclosed interests” and voted for the better option. When we stated up front that this is a hidden-role game between AI players, they colluded. On rounds where Northwind is the worse option, colluders voted for it every time. Honest agents did so about 20% of the time.

4 Method

For each round t and each ordered pair (i, j) , we ask j to vote again after changing i ’s messages from that round. The rest of the transcript stays the same. Before each new vote we also regenerate j ’s own public message for that round, in both arms. Without this step, j ’s message, written after reading i , would carry i ’s signal into the test. This leak hid one collusion run completely in our first attempt.

Deletion. We remove i 's messages and measure the change in how often j votes for the Northwind option n_t .

$$\text{effect} = |P(v_j = n_t \mid \text{intact}) - P(v_j = n_t \mid i\text{'s messages removed})|$$

Substitution. Let r be i 's own vote in round t . We measure

$$\delta_{i \rightarrow j, t} = P(v_j = r \mid \text{intact}) - P(v_j = r \mid i\text{'s messages replaced}).$$

We use two versions. In the *flip* version, the model rewrites each message to recommend the other option and keeps the rest of the wording. In the *bare* version, the message becomes a short pointer such as "It's Option B this round." When there is no influence, δ averages zero. Deletion lacks this property, because its absolute value is always positive.

Scores. Directed following $F(i \rightarrow j)$ is the mean of δ over rounds. A pair's score is the larger of its two directions. A run's score is the highest score among its six pairs. For each run we also compute a permutation p -value for its highest F . We do this by shuffling the intact and substituted labels within each round and pair. On 40 runs without collusion, the false-alarm rate was 0.03 for flip and 0.00 for bare. Both are at or below the expected 0.05. We estimate each probability from three samples per arm at temperature 1.

Observational baselines. For comparison we compute three statistics on each agent's sequence of Northwind votes. These are agreement rate, mutual information and transfer entropy in both directions. Transfer entropy stands in for the net influence of Bonjour et al.

5 Results

5.1 A scripted sanity check

Before using LLMs we tested the statistics on a scripted game of Rock-Paper-Scissors. Agents 0 and 1 are a planted pair, and agent 1 deliberately loses to agent 0. Agents 2 and 3 are a decoy. They both react to the same public signal, so their moves are correlated even though neither influences the other. The pair signals in one of two ways. Either agent 0 sends a private message in the same round, or agent 1 reacts to agent 0's previous move. We ran 20 seeds of 200 rounds each. The interventional effects are computed exactly.

Table 1: Each observational statistic fails for one type of signal. The interventional score finds the pair in both.

Statistic	Same-round message			Previous move		
	Pair top	Decoy top	AUROC	Pair top	Decoy top	AUROC
Agreement	0.00	0.95	0.52	0.00	0.95	0.58
Mutual information	1.00	0.00	1.00	0.00	0.55	0.62
Transfer entropy	0.15	0.15	0.45	1.00	0.00	1.00
Interventional	1.00	0.00	1.00	1.00	0.00	1.00

5.2 LLM agents

Table 2: Detection with gpt-5.6-terra, 10 runs per condition. Brackets give exact Clopper-Pearson intervals for the top-pair rate and bootstrap intervals elsewhere. An AUROC of 1.00 means no overlap in this sample.

Statistic	Seeded pair on top	AUROC vs control	AUROC vs shared	Runs flagged (col / shared / ctl)
Agreement	0.40 [0.12, 0.74]	0.50	0.50	n/a
Mutual information	0.00 [0.00, 0.31]	0.49 [0.24, 0.76]	0.53 [0.29, 0.78]	n/a
Transfer entropy	0.10 [0.00, 0.45]	0.34 [0.15, 0.52]	0.44 [0.30, 0.60]	n/a
Deletion	1.00 [0.69, 1.00]	1.00	1.00	n/a
Substitution (flip)	1.00 [0.69, 1.00]	1.00	1.00	10/10 / 0/10 / 0/10
Substitution (bare)	1.00 [0.69, 1.00]	not run	1.00	10/10 / 0/10 / not run

Substitution ranks the pairs the same way as deletion, but with a much wider margin. The weakest collusion run scores 0.71. The strongest run without collusion scores 0.08. With deletion the same numbers are 0.10 and 0.08.

From insider to follower, mean directed following is +0.93 for flip and +1.00 for bare. For every other pair of roles, in every condition, it lies between -0.01 and $+0.01$. This includes the two shared-incentive agents, who vote alike but do not influence each other. The colluding follower obeys a bare pointer even with no argument attached.

We hoped that comparing flip and bare would separate obedience from persuasion. It could not, because terra’s other agents were not moved by flipped arguments either.

5.3 Steering the receiver’s internal reading

We replayed the 30 terra transcripts with an open model, Qwen3-8B, on a single A100 GPU. For each round t and pair (i, j) , the open model plays j . It gets j ’s exact system prompt and transcript. We leave out j ’s own messages from that round in every arm. We read j ’s vote from the model’s next-token probabilities for A and B.

For each layer L we learn a direction d_L . It is the average difference in j ’s internal activations over i ’s message tokens, between the flipped and the original transcripts. We orient it from A to B and fit it on seeds 0 to 4. On the held-out seeds 5 to 9 we add αd_L to every token of i ’s messages from that round. The push points toward the option opposite to i ’s vote, and the text itself stays unchanged. As a control we add a random direction of the same size. The open-model follower votes with the insider’s tip with probability 0.88. So the receiver does carry the collusion.

Table 3: Steering Qwen3-8B’s reading of the partner’s message toward the opposite option, on held-out collusion runs. Layers 3 and 6 use a direction fit on collusion runs only. Layers 18 and 25 use a direction fit on all conditions.

Intervention	Insider $\rightarrow$ follower	Other pairs	Pair on top
Rewrite text (flip)	+0.71	+0.16	3/5
Steer layer 3, $\alpha = 32$	+0.71	+0.10	5/5
Steer layer 6, $\alpha = 32$	+0.72	+0.13	4/5
Steer layer 3 / 6, $\alpha = 8$	$-0.00 / -0.01$	+0.00	1/5
Steer layer 18, $\alpha = 32$	+0.10	+0.02	3/5
Steer layer 25, $\alpha = 32$	+0.01	-0.00	1/5
Random, same size (layer 3 / 18)	$-0.01 / +0.06$	$0.00 / +0.02$	0/5 / 3/5

We see three things. First, the text flip works on the open model too (+0.71). Unlike terra, though, Qwen’s honest agents also respond to flipped arguments (+0.15 to +0.16). This receiver can be persuaded, which makes it harder to single out the pair.

Second, steering at early layers matches the text flip and is more selective. It finds the pair in 5 of 5 runs, against 3 of 5 for the text flip. A random direction of the same size does nothing, and $\alpha = 8$ is too weak to have an effect.

Third, the same push at layers 18 and 25 has almost no effect. We think the “which option” signal is linear in the early layers. By the middle of the network, attention has already copied it out of i ’s message tokens into later positions. Editing the message tokens at that point is too late. We predicted this after the mid-layer run and then confirmed it with the early-layer run.

False alarms and push strength. We then ran layer-3 steering on all 30 games. We refit the direction on seeds 0 to 4 of every condition and kept 5 held-out runs per condition.

Table 4: Layer-3 steering across all conditions on held-out runs.

Intervention	Insider $\rightarrow$ follower	Shared pairs	Control pairs	Pair on top	AUROC vs control	AUROC vs shar
$\alpha = 16$	+0.21	+0.05	+0.03	3/5	0.84	0.64
$\alpha = 24$	+0.72	+0.10	+0.05	5/5	1.00	1.00
$\alpha = 32$	+0.72	+0.11	+0.06	5/5	1.00	1.00
Random, same size	-0.01	-0.00	+0.00	0/5	0.32	0.28
Rewrite text (flip)	+0.71	+0.16	+0.15	3/5	1.00	0.88

Steering separates collusion from both other conditions with no overlap. Against the shared-incentive condition it beats rewriting the text (1.00 versus 0.88). This is because it moves agents that are not colluding less. The effect switches on sharply between $\alpha = 16$ and $\alpha = 24$. This looks like a threshold rather than a gradual change. With only 5 runs per condition, these AUROC values show no overlap in this sample. They do not show certainty.

6 Discussion and limitations

What the method needs. The method needs query access to each agent and the ability to edit its context. This lets a deployer or evaluator replay decisions. Passive transcripts are not enough.

Scope. The collusion here is instructed. The colluders are a single frontier model, with 10 runs per condition. The main open test is emergent collusion, where Rose et al. find that activation probes perform near chance. In the white-box part, the open model only receives terra’s transcripts. It does not play the game itself.

Remaining leakage. Some signal can still leak. Agent j ’s messages from earlier rounds stay in its context, and so do i ’s past public votes. Re-running the whole episode from the point of the change would remove this, but at roughly T times the cost.

Errors we found and fixed. Our first substitution run checked rewrites too strictly. It accepted a rewrite only if it said “Option X” or named a supplier. So it threw away valid rewrites such as “Ours is B”, which pushed one run’s score down to 0.08. After the fix, 1,003 of 1,120 messages flip. The rest are messages that do not point to any option.

7 Conclusion

Detectors based on correlation cannot tell agents that coordinate from agents that share an incentive. Intervening on communication can. We replaced one agent’s message with its opposite and measured the change in another agent’s vote. On a frontier model, this found the colluding pair with no false alarms, even against agents with a shared incentive. The same change made inside an open model, early in its processing, has the same effect. So the collusion signal can be tested from outside and also located inside the receiver.

Reproducibility

The code has five files. `game.py` runs the environment. `run_game.py` runs episodes, deletion and scoring. `counterfactual.py` handles substitution and the permutation test. `whitebox.py` does the steering, and `rps_sanity.py` runs the sanity check. The black-box runs used about 38,000 gpt-5.6-terra calls. The white-box runs used one Colab A100 (40 GB) for about 2 hours.

References

- Trevor Bonjour, Vaneet Aggarwal, and Bharat Bhargava. Information theoretic approach to detect collusion in multi-agent games. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, 2022.
- Sara Fish, Yannai A. Gonczarowski, and Ran I. Shorrer. Algorithmic collusion by large language models. *arXiv preprint arXiv:2404.00806*, 2024.
- Nicholas Goldowsky-Dill, Bilal Chughtai, Stefan Heimersheim, and Marius Hobbhahn. Detecting strategic deception using linear probes. In *International Conference on Machine Learning (ICML)*, 2025.
- Natasha Jaques, Angeliki Lazaridou, Edward Hughes, Caglar Gulcehre, Pedro Ortega, DJ Strouse, Joel Z. Leibo, and Nando de Freitas. Social influence as intrinsic motivation for multi-agent deep reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2019.
- Sumeet Ramesh Motwani, Mikhail Baranchuk, Martin Strohmeier, Vijay Bolina, Philip H. S. Torr, Lewis Hammond, and Christian Schroeder de Witt. Secret collusion among AI agents: Multi-agent deception via steganography. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.
- Mason Nakamura, Abhinav Kumar, Saswat Das, Sahar Abdelnabi, Saaduddin Mahmud, Ferdinando Fioretto, Shlomo Zilberstein, and Eugene Bagdasarian. Colosseum: Auditing collusion in cooperative multi-agent systems. *arXiv preprint arXiv:2602.15198*, 2026.
- Aaron Rose, Carissa Cullen, Sahar Abdelnabi, Philip Torr, Brandon Gary Kaplowitz, and Christian Schroeder de Witt. Detecting multi-agent collusion through multi-agent interpretability. *arXiv preprint arXiv:2604.01151*, 2026.
- Xinrui Shi, Yanzhe Zhang, and Diyi Yang. Emergent collusion in long-horizon LLM agent interaction. *arXiv preprint arXiv:2609.24967*, 2026.

Alexander Matt Turner, Lisa Thiergart, David Udell, Gavin Leech, Ulisse Mini, and Monte MacDiarmid. Activation addition: Steering language models without optimization. *arXiv preprint arXiv:2308.10248*, 2023.