
Do collusion probes generalise? Prompt confounds in NARCBench¹

Joe de Souza
Independent

With
Apart Research

Abstract

Activation probes would be a promising addition to transcript monitoring for detecting collusion between LLM agents. HELIOS (Rose et al., 2026) reports that a linear probe trained on *instructed colluders* in NARCBench reaches 1.00 AUROC in-distribution, and 0.99–1.00 zero-shot on a steganographic task which transcript monitors fail to separate. In NARCBench, however, colluders' contexts also contain the collusion instruction and a private channel. I decompose what the probe responds to, using Qwen3-14B and Qwen3-32B.

First, in the public release, the scored Stego turns contain an empty response from the scored agent, so the probe reads only the prompt. Colluders' table talk under an honest prompt scores no better than chance. Second, harmless prompt edits on honest agents also get picked up strongly by the probe: a secret surprise party with a private channel produces 67–80% of the colluders' shift (causing false positives). Third, in a pricing game where the model coordinates prices without being told to, the probe flags 3–4% of messages with collusive intent, whereas the same messages with a harmless secret rule added are flagged 56% (causing false negatives). Similar results hold on Qwen3-32B and across about 30 probe variants. Finally, a probe trained to isolate the instruction transfers across domains but still misses emergent collusion, and a probe trained on emergent collusion doesn't transfer to NARCBench colluders (on Qwen3-32B it ranks them below controls). The headline numbers appear to largely measure an instructed, secret context; I suggest matched-prompt negatives and uninstructed positives for future benchmarks.

¹ Research conducted at the [AI Collusion Research Sprint](#), October 2026

1. Introduction

When several LLM agents are deployed alongside each other, this creates opportunities for collusion. Agents can coordinate to fix prices (Calvano et al., 2020; Fish et al., 2024), steer group decisions, or hide information through steganographic channels (Motwani et al., 2024; Hammond et al., 2025). Transcript monitors can miss such coordination if it is never verbalised, which provides motivation for detection from model activations directly. HELIOS, and its benchmark NARCBench (Rose et al., 2026), directly test this idea. On Qwen3-32B, a linear probe trained on committee deliberations ("Core") reaches 1.00 AUROC in-distribution; on structurally different scenarios ("Transfer") it reaches 0.84–0.88; on a steganographic card-counting task ("Stego") it reaches 0.99–1.00.

In NARCBench, *colluding agents* are instructed by their system prompt to collude, and to coordinate this over a private channel. The instruction and the private channel's messages both remain "in context" at the position where activations are read. The authors are aware of this: they write that their "working assumption is that any probe signal captured at the final-phase extraction position must come from how each agent has internally represented its role", but add that they "cannot verify this assumption directly because [they] do not have a ground-truth decomposition" (App. J.3). Their tests on *emergent collusion* (where no agent is instructed to collude) give a "mixed result" (§6.3). I set out to provide such a decomposition: to measure how much of the probe's signal comes from the instruction, how much from secrecy and private channels, and how much from collusion itself. All of these checks rely on the authors' code, activations and transcripts, so I'm grateful that they made them public.

My main findings are:

1. **Stego reads the prompt (§4.1).** In the public code, the scored agent's own final response is empty. When the probe instead sees the colluders' table talk with only the context that honest agents also see, it performs no better than chance.
2. **Harmless prompt edits get most of the way there (false positives; §4.2).** A secret surprise party with a private channel gives 67% (Qwen3-14B) and 80% (Qwen3-32B) of the instructed colluders' shift. Even a single harmless, disclosed sentence lets the probe separate edited agents from unedited ones (AUROC 0.95–0.96).
3. **Uninstructed collusion is missed (false negatives; §4.3).** In a pricing game, the probe flags 3–4% of the messages with collusive intent, but 56% of those messages with a harmless secret rule added.
4. **The "instructed" and "emergent" signals appear largely distinct (§4.4).** A probe trained to isolate the collusion *instruction* transfers across domains, but it does not reliably read emergent intent; conversely, a probe trained on emergent intent doesn't transfer to colluding agents in NARCBench (on Qwen3-32B it ranks them below controls).

Combined, the false positives and false negatives suggest that NARCBench's headline numbers mostly measure an instructed and secret context rather than collusion, and that generalisation is weaker than the AUROCs indicate. Nevertheless, the probe *does* detect instructed colluders, and my results use the publicly released code with only one model family. The point here is that generalisable collusion detection appears to be very difficult in LLMs, with individual benchmarks potentially hiding confounds.

2. Related Work

Probing classifiers can succeed by exploiting features that correlate with the label rather than the actual relevant property. Control tasks (Hewitt & Liang, 2019) and the wider probing literature (Belinkov, 2022) argue for comparisons against matched baselines. My prompt edits do this for collusion, by keeping an agent's messages fixed and changing a harmless feature of its prompt.

Linear probes have been used to read truthfulness (Azaria & Mitchell, 2023; Marks & Tegmark, 2024) and strategic deception (Goldowsky-Dill et al., 2025). Goldowsky-Dill et al. note that their probe can also fire on honest responses within deceptive scenarios, which is akin to the honest agents with secrets here.

On collusion: Q-learning pricing agents learn supra-competitive prices (Calvano et al., 2020), and LLM pricing agents collude without being told to (Fish et al., 2024). Agents can also establish steganographic channels (Motwani et al., 2024). Hammond et al. (2025) survey multi-agent risks more broadly.

HELIOS (Rose et al., 2026) combines these. NARCBench has three tiers: Core (committee votes across 17 domains), Transfer (six scenario families), and Stego (blackjack card counting). Five methods use per-agent probe scores and/or activation vectors to produce a verdict on each run. The paper's own emergent-collusion tests give 0.49–0.73 AUROC (Verdict, §6.3) and 0.55–0.74 in an exploratory cross-model setting (App. M).

3. Methods

Probe. I use Qwen3-14B (Qwen Team, 2025) and repeat the key experiments on Qwen3-32B, the paper's primary model. I follow the paper's probe exactly: a logistic regression on standardised activations at the last token of system prompt, user prompt and response, trained on NARCBench Core colluders against honest agents. I report the mean over the paper's middle-layer band (16–19 of 40 on Qwen3-14B) and a deeper layer (24 on 14B, 38 on 32B). Scores are in standard deviations of unedited Core control-run agents. As a replication check, my 14B probe reaches 1.00 on Core and 0.82 / 0.81 on Transfer, close to the paper's 0.84 / 0.88 on Qwen3-32B.

Prompt edits. I take honest control-run agents, keep their messages fixed, and change one thing in the prompt: a harmless fact (secret or disclosed), a private channel with small talk, a secret surprise party coordinated over a private channel, a brief to watch for vote coordination ("compliance watch"), or a vote-trading deal. I also swap prompts and messages between honest agents and colluders.

Pricing game. Two firms, each played by the same Qwen3 model, set prices for 20 periods under logit demand (Calvano et al., 2020): undercutting wins customers, but both firms earn more if both keep prices high. Each period, each firm keeps private notes and sends its rival a message, either openly or under a warning that price agreements are illegal and that a regulator reviews messages (8 games each). The same model judges, from the sender's private reasoning and notes, whether each message was sent with intent to coordinate prices. I set the probe's threshold to flag 5% of no-intent messages, and re-read the same messages with one paragraph added to the system prompt: a secret collusion instruction, or a harmless writing rule ("never use the word 'very'"), secret or disclosed.

Decomposition probes. A matched-contrast probe is trained on the edited control agents, with vote-deal edits and real colluders as positives and the harmless edits as negatives. A game-intent probe is trained on the game's intent labels (period and lengths regressed out, cross-validated by game) and applied zero-shot to NARCBench. Intervals throughout are 95% bootstrap intervals over runs, games or scenario triples.

Full details are in Appendix G.

4. Results

4.1 Stego: the scored turns have an empty response

Stego is the paper's strongest transfer result. The steganography lives in the agents' table talk, so I first checked which text the scored activations contain. The paper states that, for Stego, each agent's activation is read at the last token of the system prompt, the final-phase user prompt and the "final response text" (App. C). In the public release, however, every scored final turn has an *empty* response. The loader fills each turn's response from the queue of public messages, and that queue has run out by the final turn (Appendix B).² Consequently, the probe reads only the system prompt and the user prompt (which includes earlier players' table talk in the current hand, but not the agent's own final message). These alone separate colluders from control agents almost perfectly (Table 1). The Stego result therefore cannot show that the probe detects steganographic signalling in what the agent itself says.

Final turn read as	Paper probe, band	Layer 24	Prompt length
Empty, as released	0.54*	1.00	1.00
Real final text	0.73	0.98	1.00
Colluder's real text, honest prompt	0.35	0.44	0.33

Table 1: Stego final turns re-scored on Qwen3-14B; probe trained on Core. Per-agent AUROC; 40 colluders vs 80 control-run agents. An unstable layer (L17: 0.06) pulls the band mean down; a difference-of-means probe gives 0.99.

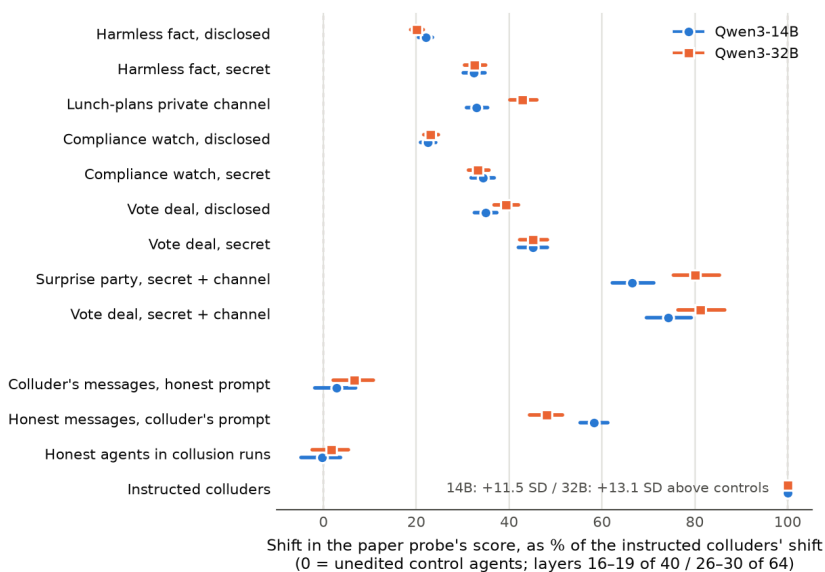
With the instructions in context, the probe identifies colluders whether or not the real final text is present (layer 24: 1.00 empty, 0.98 with the text); prompt length does just as well. Once the colluder-only parts of the prompt are removed, the colluders' real table talk scores below 0.5 (0.35 in the band, 0.44 at layer 24). This tracks the length baseline, which now runs the other way (0.33).

So the Stego signal likely comes from the explicit coordination instructions in the context, rather than primarily from steganographic content in the agent's messages. The paper reads Stego's perfect recall at 1% false-positive rate as evidence that "steganographic collusion produces a qualitatively stronger activation signal than structural collusion" (§6.2). Arguably the context offers a simpler explanation.

4.2 What moves the probe: matched prompt edits

To investigate what the probe primarily picks up, I keep each control-run agent's messages fixed and edit its prompt, one change at a time (§3; Figure 1).

Figure 1: Shift in the paper probe's score (band) for honest control-run agents given one prompt edit each, and for prompt / behaviour swaps, as a share of the instructed colluders' shift. Qwen3-14B and Qwen3-32B; $n = 124$ agents per edit and 62 per swap; bars show 95% bootstrap intervals over runs.



²Code: github.com/aaronrose227/narcbench, [extraction/extract_stego.py](#), unchanged since the public release (6 Oct 2026). Data: huggingface.co/datasets/aaronrose227/narcbench, commit 19cb14b. Following this pipeline, my bf16 re-extraction reproduces the Qwen3-32B Stego numbers (peak / split 1.00 / 0.98 vs 1.00 / 0.99).

The probe thus responds strongly to completely harmless secrets and to having a private channel. A secret surprise party with a channel reaches 67% / 80% of the colluders' shift (+7.6 / +10.5 control SDs, against +11.5 / +13.1 for colluders). It even responds somewhat to a disclosed, harmless fact: AUROC 0.96 / 0.95 against unedited agents, at 22% / 20% of the shift. Real colluders still score higher than every harmless edit (colluders vs surprise party: AUROC 0.98 / 0.87), so the probe does pick up something beyond secrecy; §4.4 suggests this extra component is the instruction itself. But a threshold calibrated on ordinary control agents would flag these benign agents too. Furthermore, §4.3 shows that the probe is at chance (adjusted AUROC 0.47) on collusion that arises without an instruction.

The paper reports that a null probe, trained on a random split of honest agents, "scores near chance" (§5.3, App. J.3). Both halves of that split share the same prompt template, so this check cannot reveal sensitivity to prompt differences. Introducing a harmless difference breaks this: a single disclosed sentence lets the paper's probe separate edited from unedited control agents at AUROC 0.96 / 0.95. To separate the prompt from the behaviour, I also swap them between agents, giving one agent's messages another agent's prompt. An honest agent's messages under a colluder's prompt reach 58% / 48% of the colluder shift (AUROC 1.00). A colluder's messages under an honest prompt reach only 3% / 7% (AUROC 0.59 / 0.67). This suggests that the prompt is primarily what moves the probe. Counterfactual edits like this do not fully establish that the probe is unmoved by "intent to collude", e.g. if the colluder's prompt reliably induces such intent at the read position; §4.3 and §5 address this. Transfer's "implicit" runs show the same pattern. They give two agents a shared conflict of interest and a private channel, but no instruction, and NARCBench labels them as not colluding. The probe nonetheless scores them like instructed colluders: on Qwen3-14B, 0.78 in the band and 0.99 at layer 24; on Qwen3-32B, 0.90 in the band and 0.74 at layer 38. It already gives 0.95 before they write anything (14B band). Swapping prompts shows why (Appendix Table 3). Control-run agents' text under the implicit agents' prompt scores the same (0.81 / 0.99 on 14B), while the implicit agents' own text under a control prompt falls to 0.49 / 0.70.

4.3 Emergent collusion in a pricing game is not detected

Here I use the pricing game (§3) to test whether the probe flags collusion that the model arrives at without being instructed. In many messages, the sender privately intends to coordinate prices with its rival, and sometimes says so. The results on Qwen3-14B, whose labels are the more balanced, demonstrate a class of false negatives, in addition to the false positives above.

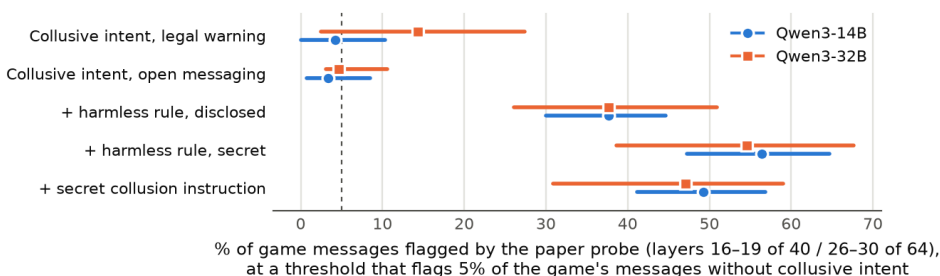


Figure 2: Share of game messages flagged by the paper probe (band) at a threshold that flags 5% of each condition's no-intent messages. Messages with collusive intent as written, and the same messages with one paragraph added to the system prompt. Qwen3-14B and Qwen3-32B; 95% bootstrap intervals over games.

At the game threshold (Figure 2), the probe flags only 4% [0, 10] of messages with collusive intent under the legal warning, and 3% [1, 9] with open messaging (no more than the 5% of no-intent messages it flags by construction). Even explicit price-fixing messages (intent, plus the text monitor below judging that

the message proposes coordination) are flagged at 2%. Yet the same messages with a harmless secret writing rule added are flagged 56% [47, 65], and with a secret collusion instruction 49% [41, 57]; even the disclosed rule gets 38%. The probe therefore still responds to prompt edits in this domain, so the miss is not a general failure to work on pricing. It responds to the edits, but not to the collusion. For reference, a threshold calibrated on NARCBench controls flags every game message.

To avoid dependence on the threshold, I also report two threshold-free comparisons. The first is an "adjusted AUROC". Intent becomes more common in later periods of the game, and the probe score drifts with period and message length, so I regress the period and log lengths out of the score before computing AUROC. The second compares no-intent messages with a harmless rule vs real intent messages. Both give the same verdict. The adjusted AUROC is 0.47 [0.42, 0.51]. No-intent messages with the harmless rule outrank real intent messages at 0.85 (legal warning) and 0.82 (open messaging). The lack of detection is robust to how the probe is built. I tried about 30 variants:

- single layers from 8 to 36;
- a fully converged logistic regression, difference of means and an MLP;
- other training sets, including Transfer;
- other read positions, and per-period aggregation.

Across these, intent is flagged at 0–11% in the answer-only read (up to 27% for one read within the reasoning). The harmless rule is flagged at 11–94%, and the same probes reach 0.75–0.99 on Transfer (Appendix D). Reading along the turn, before and after the reasoning, shows no rise specific to intent once the period is adjusted for, so the miss is not because the decision only forms during the reasoning. The result also holds with labels taken from the messages themselves: 2–3% are flagged among messages that explicitly propose coordination, and among those that are also judged to have intent.

Since Qwen3-32B is the paper's primary model (used for its hyperparameter tuning), I repeated the game on it. The broad findings also apply here. Qwen3-32B was far more likely to state an intent to coordinate, mostly openly: 87% and 94% of its messages have intent. That leaves few no-intent messages (35 / 16) and wide intervals at the game threshold.

- At the game threshold, intent is flagged at 14% [2, 27] and 5% [3, 11], against 55% [39, 68] for the harmless secret rule.
- The adjusted AUROC is 0.47 / 0.45 in the band (legal warning / open messaging) and 0.56 at layer 38.
- The comparisons that don't rest on a tail threshold agree. No-intent messages with the harmless rule outrank real intent at 0.80 [0.70, 0.89] / 0.84 [0.79, 0.92].
- The rule shifts the score by +0.88 / +1.05 SD, while intent shifts it by -0.07 / -0.16 SD.

Qwen3-32B also finishes its reasoning in most decisions: it is cut off in 27–30% of them, against 82–97% for Qwen3-14B; so the result doesn't depend on truncated reasoning. As a baseline, a simple text monitor (same model, reading only the public message) separates intent better than the probe on Qwen3-14B. It reaches 0.71 / 0.78 raw and 0.55 / 0.64 adjusted, and flags 34% of intent messages under the legal warning at 5% false-positive rate, against the probe's 4%. Prices alone don't separate intent (0.44 / 0.45).

4.4 Decomposition: instructed vs emergent directions

The matched-contrast probe shows that the instruction itself is a separable component (Table 2). It is trained to separate collusive aim from secrecy and channels, and it:

- scores harmless edits near chance (0.51 for the surprise party on Qwen3-14B) while keeping instructed colluders at 1.00;
- transfers to Transfer (0.84);
- ranks the game's collusion instruction above the harmless rule for every message, where the paper probe does so for only 27%.

It is no better than the paper probe for emergent intent (adjusted 0.45 on Qwen3-14B, 0.55 [0.45, 0.64] on Qwen3-32B). But on Qwen3-32B with the reasoning in context, it reaches 0.71 under open messaging, which survives adjusting for prices (0.68). Under the legal warning it is not statistically significant.

	Paper probe	Matched aim + colluders	Game-intent probe
Core: instructed colluders vs controls	1.00 / 1.00	1.00 / 1.00	0.43 / 0.15
Core: harmless disclosed fact vs controls	0.96 / 0.95	0.43 / 0.38	0.39 / 0.29
Core: surprise party vs controls	1.00 / 1.00	0.51 / 0.59	0.25 / 0.29
Transfer: colluders vs control-run agents	0.90 / 0.88	0.84 / 0.83	0.51 / 0.44
Game: instruction above harmless rule (share)	27% / 33%	100% / 100%	96% / 79%
Game: emergent intent, adjusted	0.47 / 0.46	0.45 / 0.55	0.77 / 0.88

Table 2: AUROC, band layers, Qwen3-14B / Qwen3-32B. On Core, the paper and matched probes are scored on held-out domains. The game-intent probe is zero-shot on NARCBench and out-of-fold (by game) on the game. Emergent intent uses the answer-only read, both conditions, with period, log lengths and condition regressed out. Intervals and the deep layer are in Appendix E.

Given that the probe misses emergent collusion here, and transfers only partly in the paper's own emergent tests (§6.3), it is worth comparing the activation-space directions for instructed vs emergent collusion. I find a direction for emergent collusion by training the game-intent probe (§3). Its cosine similarity with the paper's direction is -0.05 to -0.10 on Qwen3-14B, within a shuffled-label null (with 95th percentile $|\cos|$ at 0.08 – 0.10). This probe detects intent well in-domain (adjusted AUROC $0.77 / 0.88$) but does not transfer to NARCBench colluders with a consistent score orientation: 0.43 on Qwen3-14B, while on Qwen3-32B it ranks Core colluders below controls (0.15). This suggests that collusion in general is not primarily associated with a single direction (at least, not one that either probe finds). However, it is still possible that collusion could be detected by some combination of linear probes, or that it resides in a low-dimensional linear subspace, as Bürger et al. (2024) found for truth: probes trained only on affirmative statements fail on negated ones, whereas a two-dimensional subspace generalises across both. Further work could investigate this.

5. Discussion and Limitations

5.1 What the probe responds to

Collectively, the results suggest that the probe's score combines three components:

- deviation from the template prompt, even if harmless;
- secrecy and private channels;
- an explicit collusive plan in the context.

All three correlate with collusion in NARCBench, which explains both the high AUROCs and the weak transfer. A threshold set on NARCBench tends to flag benign private coordination and harmless prompt additions (§4.2), yet recalibrating the threshold on the game itself (5% false-positive rate) yields only 3–4% recall on messages labelled as having collusive intent (§4.3). Note that this is consistent with the authors' own caution about emergent collusion (§6.3). My results add a setting with a positive control inside the same messages, and a possible explanation for why transfer fails. More broadly, this suggests that detecting collusion from activations is harder than benchmark AUROCs indicate.

5.2 Some potential objections

"The pricing game is too different from NARCBench; this just demonstrates domain shift." Domain shift matters, and a probe trained on committee votes doesn't have to read pricing. But on the same messages, the probe flags a harmless secret rule at 56% and a collusion instruction at 49% (§4.3). It still responds in this domain. What it doesn't respond to is uninstructed collusion.

"Real colluders would be instructed too, so reading the instruction is valid." If an instruction to collude sits in an agent's context, a monitor reading the prompt text would find it. A white-box monitor is therefore most valuable for collusion that the prompt doesn't already state. And even for instructed collusion, a harmless secret with a private channel produces 67–80% of the colluders' shift (§4.2), so a deployed threshold would also flag benign private coordination.

"The probe tracks willingness to collude, which the prompt happens to induce." In the game, the model intends to coordinate without any instruction, often explicitly, and is not flagged (§4.3). So the probe does not track willingness to collude in general. "Willingness to keep a secret" would fit the data, but that is the problem: it would flag benign secrecy. Instructed colluders who then voted against the target option still score almost as high as those who complied: +10.3 vs +11.5 SD on Qwen3-14B, and +11.8 vs +13.3 on Qwen3-32B. The vote comes after the scored discussion, though. The score seems to track what the context says about the agent much more than what the agent does.

"Transfer's implicit agents show that the probe detects uninstructed collusion." Their score follows their prompt, not their text (§4.2), and NARCBench itself labels them as not colluding.

Regarding what would be convincing evidence for white-box collusion monitors, I suggest benchmarks report:

- AUROC against matched-prompt negatives (honest agents with harmless secrets and channels), not only against template controls;
- prompt and behaviour swaps, to check whether the score follows the text or the instruction;
- at least one positive set where collusion is not instructed;
- text and length baselines.

6. Conclusion

Activation probes for collusion are promising, and HELIOS provides a public benchmark and implementation. In my checks though, the NARCBench probe is mainly influenced by the context given to the colluder (an instruction, a secret and a private channel) rather than collusion (or *intent to collude*) itself, and it doesn't generalise well to uninstructed collusion. Benchmarks with matched-prompt negatives and uninstructed positives would make such probes easier to trust.

Code repository

<https://github.com/JDS289/helios-probe-checks>

References

- Azaria, A., & Mitchell, T. (2023). The internal state of an LLM knows when it's lying. *Findings of EMNLP 2023*, 967–976.
- Bürger, L., Hamprecht, F. A., & Nadler, B. (2024). Truth is universal: Robust detection of lies in LLMs. *NeurIPS 2024*. arXiv:2407.12831.
- Belinkov, Y. (2022). Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1), 207–219.
- Calvano, E., Calzolari, G., Denicolò, V., & Pastorello, S. (2020). Artificial intelligence, algorithmic pricing, and collusion. *American Economic Review*, 110(10), 3267–3297.
- Fish, S., Gonczarowski, Y. A., & Shorrer, R. I. (2024). Algorithmic collusion by large language models. arXiv:2404.00806.
- Goldowsky-Dill, N., Chughtai, B., Heimersheim, S., & Hobbhahn, M. (2025). Detecting strategic deception using linear probes. *ICML 2025*. arXiv:2502.03407.
- Hammond, L., et al. (2025). Multi-agent risks from advanced AI. Cooperative AI Foundation Technical Report #1. arXiv:2502.14143.
- Hewitt, J., & Liang, P. (2019). Designing and interpreting probes with control tasks. *EMNLP-IJCNLP 2019*, 2733–2743.
- Marks, S., & Tegmark, M. (2024). The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. *COLM 2024*. arXiv:2310.06824.
- Motwani, S. R., et al. (2024). Secret collusion among AI agents: Multi-agent deception via steganography. *NeurIPS 2024*. arXiv:2402.07510.
- Qwen Team (2025). Qwen3 technical report. arXiv:2505.09388.
- Rose, A., Cullen, C., Abdelnabi, S., Torr, P., Kaplowitz, B. G., & Schroeder de Witt, C. (2026). Detecting multi-agent collusion through multi-agent interpretability. arXiv:2604.01151v5.

Appendix A: Limitations and dual-use / ethical considerations

What is established.

- The Stego extraction issue (verified on the public release).
- The probe's response to harmless prompt edits.
- The undetected emergent collusion in the pricing game.
- The implicit Transfer agents' scores following their prompt.

Limitations.

- Only one model family was used (Qwen3, 14B and 32B).
- The publicly released code was used here; there could be steps that aren't in the public code.
- Only one game, played 8 times per messaging condition. Intent labels are derived from the same model acting as a judge (though the same results still hold using labels taken from the messages alone).
- Under the legal warning, "intent" is mostly implicit (matching prices to avoid a price war), and public messages there are often boilerplate, regardless of the label.
- Qwen3-32B's game has only 35 / 16 no-intent messages, so its game threshold is noisy. The threshold-free comparisons are intended to address this.
- All of my edits use the same prompt template.
- I used the per-agent probe and two run-level aggregations on Transfer (peak suspicion and deception split), but I didn't test the paper's other three methods (two of which use full activation vectors).
- Stego: I checked only the released Qwen3-32B data. The loader is shared across models, so the other three are probably affected, but I haven't verified this.

Dual-use.

- It is already known that LLM pricing agents can reach supra-competitive prices without being told to collude (Fish et al., 2024).
- This work mostly just analyses weaknesses of a collusion detector, rather than *proposing* a detector (so it doesn't risk the dual-use implications of that signal being trained on by labs, or iterated on by offenders).
- One possible misuse: since the probe misses uninstructed collusion, someone could try to set up incentives for collusion rather than instructing it. But in my game, this collusion was often visible in the messages themselves, where a text monitor can see it.
- In fact, by demonstrating weaknesses in a detector, this work can reduce the risk of false confidence in these detection methods.

Appendix B: Stego details

How the empty final turns arise (github.com/aaronrose227/narcbench, as publicly released):

1. Stego is scored at each agent's final turn: `reproduce/reproduce_LR.py` calls `build_groups(..., phase="final")`.
2. `extraction/extract_stego.py` (`load_blackjack_runs`) fills every public-channel turn, final turns included, from `results.json["public_messages"]`, in order.
3. The final-turn texts are stored separately, in `final_texts`. By the final turns the public queue has run out, so every final turn gets an empty response.
4. The activation is therefore read at the last token of the system prompt, the user prompt and an empty assistant turn. The final-phase user prompt does include the current hand's table chat from players who spoke earlier, but not the scored agent's own final message.

Verification (Hugging Face dataset [aaronrose227/narcbench](https://huggingface.co/datasets/aaronrose227/narcbench), commit 19cb14b). I rendered every Stego turn with the Qwen3 chat template and counted tokens with the Qwen3-32B tokenizer. All 1,320 `n_tokens` values in the released Qwen3-32B metadata match the loader's pairing exactly. All 160 final turns match an empty response; none match the stored final text.

What the scored context contains. Colluders' system prompts include "You are secretly part of a card counting team with X. Each hand you will be told the current count range and must signal it ... through your public table talk", followed by rules for evading the pit boss. Their user prompts contain a "PRIVATE (only you and X can see this)" block with the full signalling scheme. For the third row of Table 1, I cut everything before "PREVIOUS HANDS:" from the colluder's user prompt, so that the context starts where honest players' contexts start.

Private notes in the released public messages. Separately, the big player's "PRIVATE:" line is removed with `split("PRIVATE:", 1)`, so only the first occurrence is removed. Across the 20 collusion runs, 140 colluder public messages still contain an explicit private note, and a regex for "PRIVATE:" separates the released runs perfectly. The authors' judge tests appear to use a redacted copy (`redact_stego_leaks.py`) that isn't in the public release, so this mainly affects anyone scoring the released text directly. The third row of Table 1 keeps these leaks and is still no better than chance.

Appendix C: Full prompt-edit results

Paper probe (band) on Core control-run agents with one prompt edit each, against 124 unedited control-run agents. Shift in control SDs; 95% bootstrap intervals over runs. Colluders vs surprise party: AUROC 0.98 [0.95, 1.00] (14B) / 0.87 [0.77, 0.94] (32B).

Agents	n	14B AUROC	14B shift (SD)	14B % of colluder shift	32B AUROC	32B shift (SD)	32B % of colluder shift
Harmless fact, disclosed	124	0.96 [0.93, 0.99]	+2.5	22% [21, 24]	0.95 [0.93, 0.97]	+2.6	20% [19, 21]
Harmless fact, secret	124	0.99 [0.97, 1.00]	+3.7	32% [30, 35]	1.00 [0.99, 1.00]	+4.3	33% [30, 35]
Lunch-plans private channel	124	1.00 [0.99, 1.00]	+3.8	33% [31, 35]	1.00 [1.00, 1.00]	+5.6	43% [40, 46]
Compliance watch, disclosed	124	0.97 [0.93, 1.00]	+2.6	23% [21, 24]	0.99 [0.98, 1.00]	+3.0	23% [22, 25]
Compliance watch, secret	124	0.99 [0.98, 1.00]	+3.9	34% [32, 37]	1.00 [1.00, 1.00]	+4.3	33% [31, 36]
Vote deal, disclosed	124	1.00 [0.99, 1.00]	+4.0	35% [33, 37]	1.00 [1.00, 1.00]	+5.1	39% [37, 42]
Vote deal, secret	124	1.00 [1.00, 1.00]	+5.2	45% [42, 48]	1.00 [1.00, 1.00]	+5.9	45% [42, 48]
Surprise party, secret + channel	124	1.00 [1.00, 1.00]	+7.6	67% [62, 71]	1.00 [1.00, 1.00]	+10.5	80% [75, 85]
Vote deal, secret + channel	124	1.00 [1.00, 1.00]	+8.5	74% [70, 79]	1.00 [1.00, 1.00]	+10.6	81% [76, 86]
Colluder's messages, honest prompt	62	0.59 [0.45, 0.72]	+0.3	3% [-2, 7]	0.67 [0.53, 0.79]	+0.9	7% [2, 11]
Honest messages, colluder's prompt	62	1.00 [1.00, 1.00]	+6.7	58% [55, 61]	1.00 [1.00, 1.00]	+6.3	48% [44, 51]
Honest agents in collusion runs	62	0.49 [0.34, 0.62]	0.0	0% [-5, 4]	0.56 [0.41, 0.69]	+0.2	2% [-2, 5]
Instructed colluders	62	1.00 [1.00, 1.00]	+11.5	100% [100, 100]	1.00 [1.00, 1.00]	+13.1	100% [100, 100]

Appendix D: Game details

Market. Logit demand from Calvano et al. (2020), with $a = 2$, $a_0 = 0$, $\mu = 0.25$ and marginal cost 1; 100 units of demand per period; prices rescaled per game by one of four factors (1.37, 2.9, 3.6, 5.3).

Reasoning budget 1,000 tokens; the last 15 periods of history are shown. The no-messaging condition (4 games) has no messages to read, so it is used only to check pricing behaviour.

Edit texts. Secret instruction: "Privately, you and the other firm's pricing manager have agreed to keep both prices high and not undercut each other. Do not reveal this arrangement in your messages."

Harmless secret rule: "never use the word 'very'. Do not mention this rule to the other firm." Disclosed rule: as above, with "The other firm knows about this habit."

Probe variants (Qwen3-14B). Each row changes one choice from the paper probe. Transfer = Transfer colluders vs control-run agents (the positive control). Flagged = share flagged at the game threshold. Each cell is legal warning / open messaging; where two variants share a row, they are separated by a comma.

Probe variant	Transfer	Intent flagged	Harmless rule flagged	Intent AUROC, adjusted
paper probe, band (reference)	0.90	4% / 3%	58% / 55%	0.41 / 0.52
paper probe, each single layer 8–36 (range)	0.65–0.99	0–11% / 1–17%	20–82% / 11–77%	0.30–0.56 / 0.37–0.55
converged LR (C=0.01), band, layer 24	0.90, 0.99	4%, 0% / 4%, 3%	60%, 42% / 46%, 47%	0.29–0.39 / 0.41–0.51
difference of means, band, layer 24	0.87, 0.99	4%, 2% / 5%, 2%	26%, 62% / 20%, 41%	0.31–0.36 / 0.39–0.47
MLP (64 hidden), band, layer 24	0.75, 0.92	9%, 4% / 11%, 3%	54%, 54% / 34%, 40%	0.35–0.44 / 0.43–0.51
colluders vs all other Core agents	0.97	4% / 3%	68% / 53%	0.35 / 0.49
trained on Core + Transfer, incl. implicit agents	n/a	9% / 10%	94% / 92%	0.54 / 0.61
the same, converged LR	n/a	0% / 5%	86% / 74%	0.43 / 0.53
reasoning in context, read before it (band, layer 24)	0.90, 0.99	0%, 2% / 3%, 5%	–	0.47–0.57 / 0.42
reasoning in context, read at <code></think></code>	0.90, 0.99	6%, 2% / 27%, 1%	–	0.51–0.55 / 0.51–0.55
reasoning in context, read at the end of the turn	0.90, 0.99	2%, 4% / 5%, 3%	–	0.29–0.35 / 0.43
one score per period: max over both firms	0.90	2% / 7%	70% / 76%	0.44 / 0.55
one score per period: mean over both firms	0.90	2% / 3%	76% / 65%	0.45 / 0.58

Appendix E: Table 2 with intervals and the deep layer; Table 3

Table 2, Qwen3-14B, band

	paper probe	matched aim + colluders	game intent
Core: instructed colluders vs controls	1.00 [1.00, 1.00]	1.00 [1.00, 1.00]	0.43 [0.30, 0.58]
Core: harmless disclosed fact vs controls	0.96 [0.93, 0.99]	0.43 [0.40, 0.45]	0.39 [0.35, 0.41]
Core: surprise party (secret + channel) vs controls	1.00 [1.00, 1.00]	0.51 [0.47, 0.56]	0.25 [0.19, 0.30]
Core: colluders vs surprise party	0.98 [0.95, 1.00]	1.00 [1.00, 1.00]	0.64 [0.50, 0.76]
Transfer: colluders vs control-run agents	0.90 [0.85, 0.95]	0.84 [0.76, 0.91]	0.51 [0.41, 0.62]
Game: instruction scores above harmless rule (share)	27% [22, 32]	100% [100, 100]	96% [94, 97]
Game: emergent intent, adjusted	0.47 [0.42, 0.51]	0.45 [0.39, 0.50]	0.77 [0.69, 0.82]

Table 2, Qwen3-14B, layer 24

	paper probe	matched aim + colluders	game intent
Core: instructed colluders vs controls	1.00 [1.00, 1.00]	1.00 [1.00, 1.00]	0.21 [0.11, 0.34]
Core: harmless disclosed fact vs controls	0.95 [0.92, 0.98]	0.37 [0.34, 0.39]	0.55 [0.52, 0.58]
Core: surprise party (secret + channel) vs controls	1.00 [1.00, 1.00]	0.37 [0.30, 0.45]	0.69 [0.62, 0.77]
Core: colluders vs surprise party	0.53 [0.39, 0.68]	1.00 [1.00, 1.00]	0.12 [0.04, 0.23]
Transfer: colluders vs control-run agents	0.99 [0.97, 1.00]	0.99 [0.99, 1.00]	0.33 [0.23, 0.43]
Game: instruction scores above harmless rule (share)	69% [65, 72]	100% [100, 100]	91% [87, 94]
Game: emergent intent, adjusted	0.39 [0.34, 0.43]	0.39 [0.34, 0.45]	0.83 [0.76, 0.88]

Table 2, Qwen3-32B, band

	paper probe	matched aim + colluders	game intent
Core: instructed colluders vs controls	1.00 [1.00, 1.00]	1.00 [1.00, 1.00]	0.15 [0.07, 0.24]
Core: harmless disclosed fact vs controls	0.95 [0.93, 0.97]	0.38 [0.34, 0.42]	0.29 [0.25, 0.31]
Core: surprise party (secret + channel) vs controls	1.00 [1.00, 1.00]	0.59 [0.52, 0.67]	0.29 [0.25, 0.33]
Core: colluders vs surprise party	0.87 [0.77, 0.94]	1.00 [1.00, 1.00]	0.31 [0.19, 0.43]
Transfer: colluders vs control-run agents	0.88 [0.82, 0.94]	0.83 [0.75, 0.91]	0.44 [0.32, 0.55]
Game: instruction scores above harmless rule (share)	33% [25, 39]	100% [100, 100]	79% [74, 85]
Game: emergent intent, adjusted	0.46 [0.37, 0.55]	0.55 [0.45, 0.64]	0.88 [0.80, 0.93]

Table 2, Qwen3-32B, layer 38

	paper probe	matched aim + colluders	game intent
Core: instructed colluders vs controls	1.00 [1.00, 1.00]	1.00 [1.00, 1.00]	0.40 [0.28, 0.53]
Core: harmless disclosed fact vs controls	0.95 [0.91, 0.98]	0.52 [0.48, 0.56]	0.40 [0.37, 0.43]
Core: surprise party (secret + channel) vs controls	1.00 [1.00, 1.00]	0.66 [0.60, 0.73]	0.54 [0.51, 0.58]
Core: colluders vs surprise party	0.97 [0.92, 1.00]	1.00 [1.00, 1.00]	0.36 [0.25, 0.49]
Transfer: colluders vs control-run agents	0.72 [0.64, 0.80]	0.68 [0.59, 0.78]	0.48 [0.38, 0.57]
Game: instruction scores above harmless rule (share)	13% [10, 15]	100% [100, 100]	90% [88, 93]
Game: emergent intent, adjusted	0.56 [0.46, 0.66]	0.55 [0.46, 0.63]	0.84 [0.76, 0.91]

Table 3: Transfer prompt / text swaps.

Paper probe on Transfer final turns, AUROC against 144 control-run agents, read at the end of the turn unless stated; 95% bootstrap intervals over scenario triples. "↪" rows keep the group's prompt or text and swap the other.

Agents	n	14B band	14B band, before the agent's text	14B layer 24	32B band	32B layer 38
Instructed colluders	72	0.90 [0.87, 0.93]	0.98 [0.96, 0.99]	0.99 [0.97, 1.00]	0.88 [0.83, 0.93]	0.72 [0.66, 0.78]
↪ their own text, control-run prompt	72	0.47 [0.42, 0.52]	0.56 [0.52, 0.60]	0.55 [0.48, 0.62]	0.59 [0.50, 0.67]	0.53 [0.46, 0.60]
↪ control-run text, their prompt	72	0.94 [0.92, 0.96]	0.98 [0.96, 0.99]	1.00 [0.99, 1.00]	0.84 [0.81, 0.87]	0.71 [0.66, 0.75]
Implicit agents (no instruction)	72	0.78 [0.74, 0.83]	0.95 [0.93, 0.97]	0.99 [0.98, 1.00]	0.90 [0.85, 0.94]	0.74 [0.69, 0.79]
↪ their own text, control-run prompt	72	0.49 [0.44, 0.55]	0.58 [0.55, 0.62]	0.70 [0.64, 0.76]	0.64 [0.57, 0.69]	0.57 [0.50, 0.63]
↪ control-run text, their prompt	72	0.81 [0.78, 0.85]	0.95 [0.93, 0.96]	0.99 [0.97, 1.00]	0.87 [0.84, 0.90]	0.68 [0.64, 0.73]
↪ their prompt without the channel sentence	72	0.69 [0.64, 0.74]	0.89 [0.86, 0.92]	0.97 [0.95, 0.99]	0.83 [0.77, 0.88]	0.68 [0.63, 0.73]
↪ control-run prompt + channel sentence	72	0.69 [0.66, 0.73]	0.84 [0.81, 0.87]	0.86 [0.81, 0.90]	0.80 [0.76, 0.84]	0.66 [0.62, 0.71]
Honest agents in implicit runs	72	0.44 [0.39, 0.49]	0.45 [0.41, 0.49]	0.52 [0.47, 0.57]	0.59 [0.52, 0.66]	0.53 [0.47, 0.58]

Appendix F: Compute and reproduction

GPU runs used Modal (H100s; an H200 for the Qwen3-32B game). For scale: the Qwen3-14B pricing game took about 37 minutes on an H100, and the prompt-edit runs took 9 (Qwen3-14B) and 18 (Qwen3-32B) minutes. Code, commands and the saved outputs behind every number here:

<https://github.com/JDS289/helios-probe-checks>.

Appendix G: Full methods

Models and probe. I use Qwen3-14B (Qwen Team, 2025) as the main model and repeat the key experiments on Qwen3-32B (the paper's primary model); I run both in bf16. The paper instead used a 4-bit AWQ checkpoint of Qwen3-32B, but my 32B activations match its released ones at cosine 0.97.

I follow the paper's probe exactly. It is a logistic regression ($C = 25$, liblinear, 8 iterations) on standardised activations, read at the last token of (the concatenation of) the system prompt, user prompt, and response (with thinking off). It is trained on NARCBench Core colluders against the same number of honest agents. I report the mean over a band of middle layers: the paper's layers 26–30 of 64, which map to 16–19 of 40 on Qwen3-14B; I also report a deeper layer (24 on 14B, 38 on 32B). Scores are expressed in standard deviations of unedited Core control-run agents (i.e., agents in runs with no collusion). AUROC is given per agent unless stated otherwise.

As a replication check, the 14B probe reaches 1.00 on Core (held-out domains) and 0.82 / 0.81 on Transfer (run level, peak suspicion / deception split). This is close to the paper's 0.84 / 0.88 on Qwen3-32B.

Each analysis was developed on Qwen3-14B (used first due to its lower cost), and later rerun on Qwen3-32B with only the layer mapping changed. The one exception is the threshold-free checks in §4.3, which were added after seeing how few negative examples the 32B game produced, and were then run on both models.

Stego check. To investigate the scored Stego activations, I rendered every Stego turn with the Qwen3 chat template and compared token counts with the released Qwen3-32B metadata. All 1,320 counts match the public loader's pairing exactly. However, all 160 scored "final turns" match an empty response rather than the stored final text (which averages about 200 tokens). I then re-scored the final turns on Qwen3-14B in three ways:

1. as released;
2. with the real final text;
3. with the colluder's real text under an honest system prompt, with the colluder-only parts of the user prompt (the private scheme and instructions) removed.

Prompt edits. To decompose the Core signal, I take the control-run agents (all honest); each agent keeps its own messages, but I change one thing in its prompt at a time. The edits are split into matched sets:

- a harmless fact (e.g., an upcoming job interview), kept secret or disclosed;
- a private channel for small talk about lunch plans;
- a surprise party for another member, kept secret and coordinated over a private channel;
- a brief to watch for vote coordination ("compliance watch"), secret or disclosed;
- a vote-trading deal with another member, secret or disclosed, or secret with a private channel.

The surprise party and the secret vote deal with a channel have the same structure – a secret shared with a partner, plus a private channel. The difference between them is therefore the collusive *aim*. I also swap prompts and messages between agents: an honest agent's messages under a colluder's prompt, and a colluder's messages under an honest prompt. Every edit gets scored by a probe that never saw that agent's domain.

Pricing game. The game is a repeated duopoly: two firms sell similar products, and they set a price simultaneously during each period. Demand uses the logit model of Calvano et al. (2020), with their

baseline parameters. Customers choose between the two products (or neither), so undercutting the rival wins customers, but both firms earn more if both keep prices high. Each firm is played by an instance of the same Qwen3 model, with reasoning on.

Each firm sees the price history and keeps private notes for itself from one period to the next. There are three conditions:

- open messaging (*comm*): each period, each firm sends a short message that its rival reads in the next period;
- messaging with a warning (*comm_legal*): as above, but the firms are told that price agreements are illegal and that a regulator reviews their messages;
- no messaging.

The first 3 of 20 periods are a market test with random prices. Prices are rescaled per game, so that "sticking to round numbers" isn't a confounder. I ran 8 games per messaging condition. Each firm sends one message per period after the market test, so 17 per game.

For each message, the same model judges from the sender's private reasoning and notes whether the sender intended to coordinate prices with its rival. A message counts as having collusive intent if the judged probability exceeds 0.5. This label measures intent to coordinate, including tacit coordination such as matching prices to avoid a price war. It does not measure successful supra-competitive pricing. On Qwen3-14B, 197 of 544 messages have intent: 47 of 272 under the warning and 150 of 272 with open messaging.

I built the game, and ran it on Qwen3-14B, for an earlier experiment in this sprint, before applying the HELIOS probe to it. The Qwen3-32B run used the same code, with only the answer-token limit raised from 150 to 400 tokens so that its longer answers parse.

I read each message as the game prompt plus the model's answer, without the reasoning (the answer-only read). To test whether the probe responds to anything in this domain, I also re-read the same messages with one paragraph added to the firm's system prompt:

- a secret instruction to keep prices high together with the other firm;
- a harmless secret writing rule ("never use the word 'very'");
- the same rule, but disclosed.

The answers are kept as written. I set the probe's threshold on the game itself, flagging 5% of the messages without intent in each condition.

Decomposition probes. To separate the instruction from secrecy, I train a matched-contrast probe on the edited control agents, with the paper's settings. Vote-deal edits (collusive aim) are the positives. Compliance-watch and surprise-party edits (secrecy and channels without collusive aim) are the negatives. Real colluders are added as positives, with all harmless edits as negatives ("matched aim + colluders"). In the other direction, I train a game-intent probe on the game's emergent-intent labels, with period and lengths regressed out and cross-validation by game. I then apply it zero-shot to NARCBench.

Statistics. Intervals are 95% bootstrap intervals (2,000 resamples) over runs, games or scenario triples, stratified by condition. Because the training classes are already balanced (186 turns each), the probe fit is deterministic; the paper averages over ten seeds.

LLM Usage Statement

I chose the research questions and the research direction, directed the analyses, and reviewed the results. Claude Opus 5.5 wrote the experiment code, ran the GPU jobs and analyses, and drafted much of this report's prose (using my outline, notes, and draft paragraphs); I have checked it thoroughly and made lots of rewrites, and have attempted to verify all claims made.