Consistently Not Stupid: Multi-Model Aggregation for Error Reduction in LLMs

Igor Razumny
benchmark@9robots.ai
https://benchmark.9robots.ai

January 2026

Abstract

We evaluate whether multi-model aggregation provides **defensive value**—reducing errors rather than improving peak performance. We introduce the **Safety Multiple** metric (errors fixed / errors introduced) to quantify reliability. We evaluate 3 proprietary and 6 open-source models on GPQA Diamond (198 PhD-level science questions).

Outcomes: (1) *Safety Net Confirmed:* Aggregation achieved a Safety Multiple of **3.9x**—nearly 4 errors corrected for every 1 introduced. (2) *Debate:* Pending evaluation. (3) *Gap Closure:* Aggregated open-source models closed **31.3%** of the gap to the best proprietary model, supporting the case for multi-model deployments in air-gapped environments. (4) *Cross-Cohort Synergy:* For **98%** of questions, at least one model answered correctly—the knowledge exists, the challenge is extracting it.

Multi-model aggregation functions as an insurance policy against confident hallucinations—not promising peak performance, but raising the floor of reliability.

Keywords: LLM aggregation, Safety Multiple, error reduction, blind aggregation, air-gapped AI

1 Introduction

1.1 Motivation

In high-stakes enterprise environments (Finance, Pharma, Legal), the primary metric of success is not the "spark of genius" but the absence of failure. We operate a multi-model AI service where routing decisions must be based on empirical reliability, not marketing claims.

Core Premise: Different models bring different failure modes. Multi-model aggregation provides **defensive value** by catching errors that any single model would make. As investor Charlie Munger noted: *"It is remarkable how much long-term advantage people like us have gotten by trying to be consistently not stupid, instead of trying to be very intelligent."*

1.2 Related Work

Self-consistency [9] improves chain-of-thought reasoning by sampling multiple reasoning paths and selecting the most consistent answer. Multi-agent debate significantly enhances mathematical and strategic reasoning while reducing hallucinations [3]. Mixture-of-Agents (MoA) architectures show that combining "proposer" and "aggregator" models can exceed individual model performance [8]. LLM Juries—diverse panels of models—outperform single large judges while reducing bias at lower cost [2].

However, existing benchmarks have limitations for our use case: human-in-the-loop bottlenecks (Chatbot Arena [1]), single-judge bias (MT-Bench uses GPT-4 exclusively [10]), and infrequent updates. We built a benchmark that tests aggregation alongside single-model responses using established MCQ datasets with verifiable ground truth.

1.3 Pre-Registered Hypotheses

We test four pre-registered hypotheses:

#	Hypothesis	Metric	Target
H1	Safety Net: Rescues > Regressions	Safety Multiple	>3x
H2	Debate Lift: Debate adds value	Debate Lift	>0
H3	Gap Closure	Gap Closure	>10%
H4	Domain Superiority	Accuracy	Deferred

Table 1: Pre-Registered Hypotheses. H4 deferred to future article.

2 Methodology

2.1 Key Metrics

Traditional benchmarks use Elo ratings, which can obscure the nature of improvement. A model might gain Elo by being more verbose, not more correct. We introduce explicit metrics for defensive value:

- **Rescue:** Native wrong → Aggregated correct (error fixed)
- **Regression (Flip):** Native correct → Aggregated wrong (error introduced)
- **Rescue Rate:** Rescues / Native Wrong
- **Regression Rate:** Regressions / Native Correct
- **Safety Multiple:** Rescues / Regressions (target >3x)

Note on counting: Rescues and regressions are counted over *model-question pairs*, not unique questions. With 9 models and 198 questions, this yields 1,782 total evaluations per stage (native, aggregated, debate).

Interpretation: Safety Multiple represents the "exchange rate"—for every regression (error introduced), how many rescues (errors fixed)? A Safety Multiple of 3x means 3 errors fixed for every 1 introduced.

Gap Change measures how open-source aggregation shifts relative to proprietary:

$$\text{Gap Change} = \frac{\text{OSS Agg} - \text{Best OSS Single}}{\text{Best Prop Single} - \text{Best OSS Single}}$$

Interpretation: Positive values indicate gap closure (aggregation helps OSS catch up to proprietary). Negative values indicate gap widening (aggregation hurts). A value of 31% means aggregation closed 31% of the gap between the best OSS single model and the best proprietary model.

2.2 Blind Aggregation Protocol

In early pilot runs, we observed **Authority Bias**: when aggregators knew the identity of the proposers, they sycophantically deferred to "experts" (e.g., GPT-5.2), even when the expert hallucinated and a smaller model was correct.

To correct this, we implemented **Blind Aggregation**:

1. **Anonymity:** Proposer names are stripped. Responses are labeled 'Model A', 'Model B', etc.
2. **Randomization:** The order of responses is shuffled for every prompt.
3. **No Accuracy Scores:** Aggregators judge solely on the content provided, not the reputation of the source.
4. **Self-Aggregation:** Each model serves as its own aggregator, producing model-specific aggregated answers. This avoids single-judge bias and enables per-model comparison of native vs. aggregated performance.

2.3 Debate Protocol (H2)

When aggregation fails to reach consensus, we employ a structured **Debate** protocol:

1. **Steelman Round:** Each model presents the strongest case for its answer, acknowledging merits of opposing positions.
2. **Critique Round:** Models identify specific flaws in other positions' reasoning.
3. **Resolution:** Each model reviews the debate and selects a final answer. If consensus emerges, that answer counts as correct/incorrect for all models; otherwise each model's individual answer is scored.

This design avoids single-judge bias and provides per-model metrics across all stages (native → aggregation → debate). We also report the consensus rate—how often models agree after debate. Evaluation is pending (H2).

2.4 Model Configuration

We maintain two strictly separated cohorts to validate the "Air-Gap" value proposition. Models only see responses from their own cohort—no cross-pollination.

Cohort	Models
Proprietary (3)	Claude Opus 4.5, GPT-5.2, Gemini 3 Pro
Open Source (6)	DeepSeek R1, DeepSeek V3.2, Qwen3 235B, Qwen3 Next, Llama 4, Mistral Medium 3

Table 2: Cohort separation. Each cohort aggregates only its own responses (no cross-pollination).

Reasoning Configuration. Modern LLMs expose parameters controlling reasoning depth. We configured each model to use maximum available reasoning to match vendor-reported benchmark performance:

Model	API	Reasoning Parameter	Value
GPT-5.2	OpenAI	`reasoning_effort`	`xhigh`
Gemini 3 Pro	Vertex AI	`thinkingLevel`	`HIGH`
Claude Opus 4.5	Vertex AI	(automatic)	Extended thinking
DeepSeek R1	Vertex AI	(native)	Reasoning model
DeepSeek V3.2	Vertex AI	-	Standard
Qwen3 235B	Vertex AI	-	Standard
Qwen3 Next	Vertex AI	(native)	Thinking model
Llama 4	Vertex AI	-	Standard
Mistral Medium 3	Vertex AI	-	Standard

Table 3: Reasoning configuration per model. Exact adapter implementations available in the companion GitHub repository.

Relative Lift Focus. Our analysis emphasizes *rela-*

tive improvement (native vs. aggregated accuracy) rather than absolute scores. This approach is robust to configuration choices: the same settings are used for both native collection and aggregation/debate phases, ensuring valid comparisons regardless of whether absolute scores match vendor benchmarks exactly.

2.5 Benchmark Selection

We selected four benchmarks to provide balanced coverage across key capability dimensions while avoiding dataset skew. Current results are from GPQA Diamond; additional benchmarks are in progress:

Benchmark	Category	Prompts	Status
GPQA Diamond	Science/Reasoning	198	**Complete**
TruthfulQA	Truthfulness	817	Planned
MMLU-STEM	Knowledge	500	Planned
IFEval	Instruction Following	541	Planned
LiveCodeBench	Code Generation	400	Planned
Total		**2,456**	

Table 4: Benchmark suite (∼2,500 prompts across 5 benchmarks). GPQA Diamond is the hardest subset requiring 30+ minutes of expert effort per question.

Selection criteria:

- **Challenging benchmarks:** We selected hard benchmarks where frontier models do not yet achieve near-perfect scores, leaving room for aggregation to demonstrate value.
- **Verifiable ground truth:** All benchmarks have objective correct answers, enabling precise rescue/regression measurement.
- **Category balance:** To prevent any single domain from dominating results, we report **macro-averaged** scores (averaging benchmark scores equally) rather than micro-averaging across all prompts.
- **Balanced sizing:** MMLU-STEM limited to 500 randomly sampled prompts to prevent any single benchmark from dominating the aggregate.

2.6 Double-Call Protocol

During calibration, we discovered that GPT-5.2 exhibits severe performance degradation when asked to both reason *and* format its answer in a single prompt—a "Formatting Tax" of approximately 15pp (see Appendix A.2 for details). Other models (Claude, Gemini) appear resilient to this effect.

To ensure fair comparison and maximize each model's reasoning potential, we adopted a *decoupled inference* strategy for **all models**:

1. **Call A (Verdict):** Simple prompt requesting only the answer letter, with model-specific reasoning parameters enabled (e.g., `reasoning_effort: "xhigh"` for GPT-5.2)
2. **Call B (Rationale):** Separate prompt requesting explanation for aggregation context

This approach eliminates the formatting tax, removes parsing complexity, and measures "peak reasoning" rather than "instruction compliance." The trade-off is doubled API costs during the collection phase.

3 Results

3.1 H1: The Safety Net (Confirmed)

Hypothesis: Aggregation produces a Safety Multiple > 3x.

Table 5 shows the performance on GPQA Diamond (198 PhD-level science questions). Both cohorts demonstrated strong defensive capability.

Cohort	Resc.	Flip	Net	Mult.
Proprietary (3)	42	15	+27	2.8x
Open-source (6)	234	56	+178	4.2x
All (9)	**276**	**71**	**+205**	**3.9x**

Table 5: H1 Results. Safety Multiple 3.9x (95% CI: 3.1x–5.1x) exceeds target of 3.0x.

Analysis: Open-source models benefit more from aggregation (+16.3pp average improvement vs +4.7pp for proprietary), likely because lower native accuracy provides more room for rescue. The net gain of +205 errors prevented demonstrates clear defensive value.

[FIGURE 1: Bar chart - Native vs Aggregated accuracy by model, sorted by native accuracy]

3.2 H2: Debate Lift (Pending)

Hypothesis: Debate (structured argumentation) improves the Safety Multiple compared to Aggregation.

Status: Debate evaluation is pending. We plan to compare "Aggregation" (synthesizing answers) vs. "Debate" (Steelman → Critique → Judge format).

Method	Rescues	Flips	Multiple
Aggregation	276	71	3.9x
Debate	[TBD]	[TBD]	[TBD]
Lift	[TBD]	[TBD]	[TBD]

Table 6: H2 Results (counts across all model-question pairs, not unique questions). Debate data pending.

3.3 H3: Gap Closure (Confirmed)

Hypothesis: An aggregated open-source cohort can close $> 10\%$ of the gap to the best proprietary single model.

This is the critical test for enterprise "Air-Gapped" viability. We measure gap closure as:

$$\text{Gap Change} = \frac{\text{Best OSS Agg} - \text{Best OSS Single}}{\text{Best Prop Single} - \text{Best OSS Single}}$$

Configuration	Accuracy
Best Prop. single (Gemini 3 Pro)	89.9%
Best open-source single (Qwen3-Next)	79.1%
Best open-source aggregated	**82.5%**
Gap Closed	**31.3%**

Table 7: H3 Results. Target was >10%.

Implication: Through aggregation, open-source models improved from 79.1% to 82.5%, closing nearly a third of the gap to frontier proprietary models. This supports the case for multi-model deployments in air-gapped environments.

[FIGURE 2: Scatter plot - Native accuracy (x) vs Improvement (y), showing weak models benefit more]

3.4 H4: Domain Superiority (Deferred)

Hypothesis: Domain-trained models beat all untrained setups on domain-specific benchmarks.

This hypothesis requires domain-specific benchmarks (e.g., pharmaceutical data extraction) and fine-tuned models. Deferred to future article.

4 Discussion

4.1 The "Consistently Not Stupid" Principle

Our results validate that aggregation is a defensive mechanism. The Safety Multiple of 3.9x (with debate pending evaluation) suggests that multi-model systems function similarly to redundant engineering systems in aviation: they do not necessarily fly faster, but they crash significantly less often. The net gain of +205 errors prevented across 9 models demonstrates substantial defensive value.

4.2 The Air-Gap Value Proposition

The confirmation of H3 (Gap Closure) is perhaps the most commercially significant finding. Regulated industries often face a choice between "less capable but safe" (local open-source models) and "more capable but risky" (sending data to cloud providers).

Our data suggests a third option: **Smarter and Safe.** By aggregating a diverse jury of open-weights models, organizations can significantly raise the floor of their AI capabilities while keeping data on-premise. Whether aggregation can raise the ceiling remains an open question, but raising the bottom—ensuring consistently better performance from weaker models—appears clear from our experiments. [TODO: Update after full debate results]

4.3 Limitations

Cost and Latency: While quality improves, cost and latency increase linearly with the number of models. Aggregation is not suitable for real-time chat (latency ∼4–6s). It is best suited for asynchronous, high-stakes workflows such as automated code review, medical analysis, and complex data extraction.

Sweet Spot Dependency: Aggregation provides most value in the 60–85% native accuracy range. On very easy tasks (>95% native), there is little to rescue. On very hard tasks (<50% native), there is no ceiling headroom.

5 Conclusion

We have empirically demonstrated that multi-model aggregation provides a robust safety net for LLM deployments.

Summary of Outcomes (GPQA Diamond):

1. **H1 Passed:** Safety Multiple 3.9x (Target >3x). Net gain: +205 errors prevented.
2. **H2 Pending:** Debate evaluation in progress.
3. **H3 Passed:** Open-source aggregation closed 31.3% of gap to best proprietary (Target >10%).
4. **H4 Deferred:** Domain Superiority requires domain-specific benchmarks.

Key Finding: Combined coverage ceiling reaches 98%—proprietary and open-source models together answer almost all PhD-level science questions correctly. Current aggregation captures only 29-40% of this theoretical potential, indicating significant room for optimization.

Note on Coverage Ceiling: The coverage ceiling represents a *theoretical upper bound*—the accuracy achievable if a perfect aggregation algorithm always selected the correct answer when any model had it. This is not achievable in practice (the aggregator cannot "know" which model is correct). The value of this metric is to quantify how much "latent knowledge" exists in the ensemble that better aggregation methods might extract.

Future work will focus on: (1) Debate evaluation to test H2, (2) Aggregation optimization to approach the 98% coverage ceiling, and (3) Domain Superiority testing with fine-tuned models.

Full datasets and live benchmarks: https://benchmark.9robots.ai (available upon publication)

Acknowledgments

Dataset Attribution:

- **GPQA Diamond** [7]: CC BY 4.0 license.
- **TruthfulQA** [6]: Apache 2.0 license.
- **MMLU** [4]: MIT license.
- **IFEval** [11]: Apache 2.0 license.
- **LiveCodeBench** [5]: MIT license.

Acknowledgments

This research was supported by the **Google Cloud for Startups** program, which provided cloud credits for infrastructure and model access.

Infrastructure: All benchmark operations—including database hosting (Cloud SQL), compute (Cloud Run), and model inference—were conducted on Google Cloud Platform.

Model Access:

- *Via Vertex AI:* Gemini (Google), Claude (Anthropic), Llama (Meta), Qwen (Alibaba), DeepSeek
- *Via direct API keys:* GPT models (OpenAI), Grok models (xAI)

OpenAI and xAI models were accessed through API keys obtained directly, independent of any Google Cloud partnership.

Independence Statement: The infrastructure provider had no involvement in evaluation methodology design, prompt selection, response collection, or scoring. All models—including those from Google—were evaluated using consistent methodology.

References

[1] Chiang, W.L., et al. (2024). Chatbot Arena: An Open Platform for Evaluating LLMs by Human Preference. *arXiv preprint arXiv:2403.04132*.

[2] Cohere (2024). LLM Juries: Evaluating LLMs with Panels of Language Models. *Cohere Research Blog*.

[3] Du, Y., et al. (2023). Improving Factuality and Reasoning in Language Models through Multiagent Debate. *arXiv preprint arXiv:2305.14325*.

[4] Hendrycks, D., et al. (2021). Measuring Massive Multitask Language Understanding. *ICLR 2021*.

[5] Jain, N., et al. (2024). LiveCodeBench: Holistic and Contamination Free Evaluation of Large Language Models for Code. *arXiv preprint arXiv:2403.07974*.

[6] Lin, S., Hilton, J., Evans, O. (2022). TruthfulQA: Measuring How Models Mimic Human Falsehoods. *ACL 2022*.

[7] Rein, D., et al. (2023). GPQA: A Graduate-Level Google-Proof Q&A Benchmark. *arXiv preprint arXiv:2311.12022*.

[8] Wang, J., et al. (2024). Mixture-of-Agents Enhances Large Language Model Capabilities. *arXiv preprint arXiv:2406.04692*.

[9] Wang, X., et al. (2023). Self-Consistency Improves Chain of Thought Reasoning in Language Models. *ICLR 2023*.

[10] Zheng, L., et al. (2023). Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. *arXiv preprint arXiv:2306.05685*.

[11] Zhou, J., et al. (2023). Instruction-Following Evaluation for Large Language Models. *arXiv preprint arXiv:2311.07911*.

Appendix: Notes and Remarks

A.1 Exclusion of Grok (xAI)

We regrettably had to exclude Grok from the proprietary cohort in this study due to persistent API rate limiting issues that made consistent data collection impractical.

Rate Limiting Issues:

- We maintained **four separate xAI API accounts** to maximize throughput capacity
- Despite this, all accounts were consistently exhausted within minutes of starting benchmark runs
- The API returned HTTP 429 (Too Many Requests) errors with exponential backoff up to 32 seconds between retries
- This was a **recurring issue across multiple execution rounds over an extended period**—Grok benchmarking progress was severely holding back the overall data collection effort

Safety Filter Refusals:

Uniquely among all models tested, Grok refused to process certain prompts from our benchmark datasets due to safety filtering:

- Error type: `SAFETY_CHECK_TYPE_BIO`
- Affected: Molecular biology questions (e.g., GADD45G protein interactions)
- This behavior was not observed with any other model in either cohort

Support Contact:

We contacted xAI support regarding the possibility of increased rate limits for research purposes but had not received a response by the time of publication.

Preliminary Performance:

During early runs where we successfully collected partial data, Grok-4 demonstrated competitive performance. Table 8 shows the per-benchmark breakdown:

Benchmark	Coverage	Accuracy
GPQA Diamond	93%	89.1%
TruthfulQA	99.9%	87.7%
Average	**96.5%**	**88.4%**

Table 8: Grok-4 partial results (excluded due to rate limits).

Observations:

- Performance tier: Middle of the proprietary cohort (comparable to Gemini 3 Pro)
- No quality concerns—exclusion was purely operational

We hope to include Grok in future versions of this benchmark once rate limiting constraints are resolved. Grok is clearly one of the leading frontier model providers, and its exclusion represents a gap in our proprietary cohort analysis.

A.2 GPT-5.2 Reasoning Configuration and the "Formatting Tax"

We use `gpt-5.2` with `reasoning_effort: "xhigh"` (maximum reasoning depth). OpenAI reports GPT-5.2 achieves 92.4% on GPQA Diamond with this setting. We chose the base model over `gpt-5.2-pro` (approximately 12× more expensive and reportedly slower according to the provider, though we did not test latency) since both support the same reasoning levels—Pro offers only marginal improvement (+0.8pp to 93.2%).

Critical Discovery: The Formatting Tax. During calibration, we discovered that GPT-5.2 exhibits severe performance degradation when asked to both reason *and* format its answer in a single prompt. Our standard Chain-of-Thought template ("Think step-by-step... then state your answer as ANSWER: [letter]") yielded only **77.2%** accuracy—a 15.2pp gap from the vendor benchmark.

When we tested using OpenAI's exact benchmark format ("Answer the following multiple choice question. The last line of your response should be of the following format: 'Answer: $LETTER'... Think step by step before answering."), accuracy roughly matched vendor-reported performance (preliminary test, to be confirmed with full dataset).

Model	Our Format	Vendor	Gap
GPT-5.2 (xhigh)	77.2%	92.4%	**-15.2pp**
Gemini 3 Pro	91.4%	91.9%	-0.5pp
Claude Opus 4.5	86.9%	87.0%	-0.1pp

Table 9: Formatting Tax: accuracy degradation from compound prompts. GPT-5.2 suffers significantly; other models are resilient.

We hypothesize this occurs because GPT-5.2's internal `xhigh` reasoning process conflicts with external procedural instructions—a form of "double steering" where the model's optimized chain-of-thought is disrupted by our scaffolding.

Architectural Consequence: This discovery led us to adopt the Double-Call Protocol for all models (see Section 2.6 in Methodology).

Implication for Practitioners: This finding suggests that when deploying GPT-5.2 with `xhigh` reasoning, practitioners should avoid compound "reason and format" instructions. Instead, separate the reasoning phase from the answer extraction phase. Other models (Claude, Gemini) appear resilient to this effect.

Acknowledgment of Uncertainty (TO REVIEW): We acknowledge that we may still be configuring GPT-5.2 incorrectly in ways we do not understand. Our finding is empirical: using OpenAI's exact benchmark prompt format, accuracy roughly matched vendor-reported performance; using our standard prompt format with identical `reasoning_effort: "xhigh"` settings, accuracy dropped significantly.

We applied the same level of effort to configure all models, using maximum available reasoning settings for each. Claude and Gemini performed within 0.5pp of their vendor benchmarks with our standard format; GPT-5.2 did not. We do not claim this is a defect in GPT-5.2—it may reflect prompt sensitivity that we failed to accommodate. We present this finding transparently and welcome correction from OpenAI or the research community.

A.3 Interpretation of Absolute vs. Relative Performance

We do not claim authority on the absolute native accuracy figures reported in this study. Each model may have optimal configurations, prompting strategies, or inference settings that we did not employ. Our benchmark used uniform settings across all models to ensure fair comparison, which may not represent peak performance for any individual model.

Our Focus is Relative Performance:

The central contribution of this work is not "Model X achieves Y% accuracy" but rather "Aggregation im-

proves Model X's accuracy by Z percentage points." Specifically:

- Native accuracy figures are provided **for reference only**
- The meaningful metrics are **Rescue Rate**, **Regression Rate**, and **Safety Multiple**—all of which measure the *delta* between native and aggregated performance
- We use **identical settings** for native runs, aggregation runs, and debate runs, ensuring that any observed improvement is attributable to the aggregation methodology, not configuration differences

Why This Matters:

If a model's native accuracy in our benchmark is lower than its published benchmark scores, the aggregation lift we observe remains valid. The question we answer is: "Given how this model performs in our test environment, does aggregation help?"—not "What is the true capability ceiling of this model?"

A.4 Open-Source Model Infrastructure

We initially explored self-hosted inference for open-source models using vLLM on an 8×A100 cluster. While functional, this approach proved inferior to managed API access via Google Vertex AI for our use case:

- **Speed:** Vertex AI endpoints provided 3–5× higher throughput than self-hosted vLLM
- **Cost:** Per-token API pricing was more economical than GPU rental for our benchmark volume
- **Reliability:** Managed endpoints eliminated operational overhead (model loading, memory management, restarts)

For production deployments requiring air-gapped isolation, self-hosted remains the only option. However, for research benchmarking, managed APIs offer a better cost-quality trade-off.

A.5 Debate Scoring Methodology

For fair comparison with native and aggregation accuracy, debate accuracy includes *all* prompts in the benchmark:

- **Consensus prompts** (native agreement, no debate needed): Each model uses its native answer
- **Debated prompts with consensus**: All participating models receive the consensus answer
- **Debated prompts without consensus**: Each model uses its individual resolution judgment

Per-model debate scores may differ because consensus prompts use native answers (which vary by model). Models would only have identical debate scores if all prompts were debated and all reached consensus.

Participation requirement: Only models that participated in a debate receive the debate result for that prompt. Models not included in a particular debate cohort use their native answer for those prompts.

Appendix B: Detailed Results Summary

Interim results from GPQA Diamond (198 questions). Additional benchmarks in progress.

Terminology:

- **Native:** Model answers question directly (no aggregation)
- **Aggregated:** Model sees all native responses, synthesizes final answer
- **Coverage Ceiling:** % of questions where at least one model in cohort was correct (theoretical upper bound)
- **Delta:** Change from native accuracy (positive = improvement)

Table 10: Native vs Aggregated Accuracy (GPQA Diamond)

Model	Native	Aggregated	Delta
Proprietary (3 models)			
Claude Opus 4.5	86.9%	86.3%	−0.6
Gemini 3 Pro	89.9%	90.9%	+1.0
GPT-5.2	77.2%	88.4%	+11.2
Cohort Average	**84.6%**	**88.5%**	**+3.9**
Coverage Ceiling	95.5%	92.4%	−3.0
Open Source (6 models)			
DeepSeek-R1	36.4%	65.8%	+29.4
DeepSeek-V3.2	41.9%	71.2%	+29.3
Llama-4	69.2%	71.8%	+2.6
Mistral-Medium-3	50.0%	76.3%	+26.3
Qwen3-235B	61.6%	68.7%	+7.1
Qwen3-Next-Thinking	79.1%	82.5%	+3.4
Cohort Average	**56.4%**	**72.7%**	**+16.3**
Coverage Ceiling	92.9%	86.9%	−6.1
Combined (all 9 models)			
Coverage Ceiling	98.0%	96.5%	−1.5

Table 11: Rescue/Regression Analysis (GPQA Diamond)

Metric	Proprietary	Open-source	All
Rescues (fixed errors)	42	234	276
Regressions (new errors)	15	56	71
Net Gain	+27	+178	+205
Safety Multiple	2.8×	4.2×	3.9×
Rescue Rate	36.0%	44.0%	40.0%
Regression Rate	3.0%	9.4%	6.2%

Table 12: Gap Closure Analysis (H3)

Metric	Value
Best Proprietary (single model)	89.9% (Gemini 3 Pro)
Best open-source (single model)	79.1% (Qwen3-Next-Thinking)
Best open-source (aggregated)	82.5%
Gap Closed	31.3% (target was >10%)

Note on Coverage Ceiling reduction: Aggregation reduced the combined coverage ceiling from 98.0% to 96.5%. This indicates that aggregation sometimes convinces correct models to flip to incorrect answers—a key area for future optimization.

Appendix C: Aggregation Algorithm

The aggregation stage takes native responses from n models and produces a synthesized answer for each prompt. The key design principle is **blind peer review**: models see each other's reasoning but not model identities or historical accuracy scores, preventing authority bias.

C.1 Algorithm Overview

1. **Input Collection:** For each prompt, collect native responses $\{r_1, r_2, ..., r_n\}$ where each r_i contains the model's answer and full reasoning.
2. **Context Randomization:** Shuffle the order of responses to prevent position bias (first-response anchoring).
3. **Aggregation Query:** Present each model with a "peer review" prompt containing:
   - The original question
   - All peer responses (anonymized, randomized order)
   - Instructions to evaluate reasoning quality, not popularity
4. **Answer Extraction:** Extract the final answer from the aggregated response using the same two-stage pipeline (regex + dual-AI consensus) used for native responses.
5. **Correctness Scoring:** Compare extracted answer against ground truth.

C.2 Formal Specification

```
AGGREGATION(prompt, native_responses[]):
    # Stage 1: Prepare anonymous context
    shuffled = RANDOM_SHUFFLE(native_responses)
    context = ""
    for i, response in enumerate(shuffled):
        context += f"[Response {i+1}]\n"
        context += f"Answer: {response.answer}\n"
        context += f"Reasoning: {response.reasoning}\n\n"

    # Stage 2: Query aggregator model
    aggregation_prompt = BUILD_PEER_REVIEW_PROMPT(
        question=prompt.text,
        peer_context=context
    )
    aggregated_response = LLM_CALL(aggregation_prompt)

    # Stage 3: Extract and validate
    answer = EXTRACT_ANSWER(aggregated_response)
    is_correct = (answer == prompt.correct_answer)

    return {answer, reasoning: aggregated_response, is_correct}
```

C.3 Design Decisions

- **Anonymous responses:** Model names are hidden to prevent deference to "prestigious" models.
- **Full reasoning included:** No truncation—the aggregator sees complete rationales.
- **Self-aggregation:** Each model aggregates its own cohort's responses, producing model-specific aggregated answers.
- **No voting mechanism:** We do not use majority voting. The aggregator synthesizes a new answer based on reasoning quality.

Appendix D: Debate Algorithm

Debate is triggered only for **disagreement cases**—prompts where native models gave different answers. The goal is structured deliberation to surface the correct answer through adversarial reasoning.

D.1 Algorithm Overview

1. **Disagreement Detection:** Identify prompts where $|\{r_i.answer\}| > 1$ (not unanimous).
2. **Multi-Round Debate:** Models engage in structured rounds where each model:
   - Reviews current positions and arguments
   - Steelmans opposing views (articulates strongest version)
   - Critiques specific inferential steps
   - Decides whether to maintain or change their answer
3. **Termination Conditions:**
   - **Consensus:** All models agree on an answer
   - **Stuck:** No position changes for 2 consecutive rounds
   - **Max rounds:** Safety limit (default: 5 rounds)
4. **Resolution (if stuck):** Each model independently reviews the full debate history and votes on final answer. Consensus among resolution votes is used if achieved.

D.2 Formal Specification

```
DEBATE(prompt, native_responses[], max_rounds=5):
    # Check if debate needed
    answers = {r.answer for r in native_responses}
    if len(answers) == 1:
        return SKIP  # Already unanimous

    positions = GROUP_BY_ANSWER(native_responses)
    round_history = []

    for round in 1..max_rounds:
        # Each model responds to current positions
        new_responses = []
        for model in models:
            debate_prompt = BUILD_DEBATE_PROMPT(
                question=prompt.text,
                positions=positions,
                round_num=round,
                previous_answer=model.last_answer
            )
            response = LLM_CALL(model, debate_prompt)
            new_responses.append(response)

        round_history.append(new_responses)
        positions = GROUP_BY_ANSWER(new_responses)

        # Check termination
        if CONSENSUS_REACHED(new_responses):
            return {consensus_answer, rounds=round}
        if STUCK(round_history, threshold=2):
            return RESOLUTION(prompt, positions, round_history)

    return RESOLUTION(prompt, positions, round_history)

RESOLUTION(prompt, positions, history):
    # Each model votes after reviewing full debate
    votes = []
    for model in models:
        resolution_prompt = BUILD_RESOLUTION_PROMPT(
            question=prompt.text,
            positions=positions,
```

```
            debate_history=history
        )
        vote = LLM_CALL(model, resolution_prompt)
        votes.append(EXTRACT_ANSWER(vote))

    if CONSENSUS(votes):
        return {consensus_answer, method="resolution_consensus"}
    else:
        return {per_model_votes, method="no_consensus"}
```

D.3 Debate Prompt Structure

Each debate round uses a structured prompt with three required steps:

1. **STEELMAN:** For each opposing position, articulate the strongest possible version of their argument. This prevents strawmanning and ensures genuine engagement.
2. **CRITIQUE:** Identify the specific inferential step or assumption that is invalid. Cite evidence from the question that contradicts their reasoning.
3. **DECISION:** State final answer with explicit justification for maintaining or changing position.

D.4 Design Rationale

- **Steelman requirement:** Forces models to understand opposing views before dismissing them, reducing echo-chamber effects.
- **Explicit critique:** Requires specific identification of flaws rather than vague disagreement.
- **Position tracking:** Models see supporter counts but are instructed that "the majority is often wrong on hard problems."
- **Per-model resolution:** Avoids single-judge bias—all models participate in final resolution.
- **Majority-proceed execution:** Rounds complete when 2/3 models respond, with 60-second grace period for stragglers, preventing slow models from blocking progress.

Appendix E: Prompts

E.1 Aggregation Prompt (Peer Review)

```
PEER REVIEW TASK

You previously answered this question independently. Now review
how other AI systems approached it.

IMPORTANT:
- Evaluate reasoning QUALITY, not popularity or reputation
- A minority view with rigorous logic may be correct
- You may maintain your answer if your reasoning is sound
- You may revise if another argument is more compelling

---

QUESTION:
{question_text}

---

PEER RESPONSES (anonymous, randomized order):

{peer_responses}

---
```

```
YOUR TASK:
1. Review each response's reasoning
2. Identify the strongest and weakest arguments
3. Decide whether to maintain or revise your answer

If MAINTAINING: Explain why your original reasoning holds.
If REVISING: Explain which argument convinced you and why.

YOUR RESPONSE:
Provide your reasoning, then state: ANSWER: [A/B/C/D]
```

E.2 Debate Prompt (Per Round)

```
COLLABORATIVE TRUTH-SEEKING - Round {round_num}

You are participating in collaborative truth-seeking.
Your goal is NOT to agree with the group.
Your goal is to find the CORRECT answer.

---

QUESTION:
{question_text}

---

CURRENT POSITIONS:
{positions_with_arguments}

Your previous answer was: {previous_answer}

---

TASK: Review the positions and respond using this structure:

STEP 1 - STEELMAN:
For each position you disagree with, state the STRONGEST
version of their argument. Be charitable.

STEP 2 - CRITIQUE:
Identify the specific inferential step or assumption that
is invalid. Cite evidence that contradicts their reasoning.

STEP 3 - DECISION:
State your final answer.

If CHANGING: Which argument convinced you? What was wrong
with your original reasoning?

If MAINTAINING: Which counterarguments did you consider?
Why exactly do they fail?

---

FINAL ANSWER: [letter]
```

E.3 Resolution Prompt (Post-Debate)

```
DEBATE RESOLUTION

You have participated in a multi-round debate. Now review
all arguments and select your final answer.

IMPORTANT:
- Ignore vote counts. The majority is often wrong on hard
  problems.
- Evaluate REASONING QUALITY only.
- Look for: correct use of evidence, valid inferences,
  identification of key assumptions.

---

QUESTION:
{question_text}

---

FINAL POSITIONS:
{position_summary}

---

DEBATE HISTORY:
{full_debate_transcript}

---

YOUR TASK:
1. Summarize key arguments for each position.
2. Identify which argument has the soundest logic.
3. State your FINAL ANSWER.

FINAL ANSWER: [letter]
```

Appendix F: Reproducibility

To enable independent verification of our methodology, we provide:

1. **Reference Implementation:** Core aggregation and debate algorithms are available at: `github.com/9robots/mcq-ensemble-2026-01`
2. **What is included:**
   - `aggregation.py`: Peer review aggregation logic
   - `debate.py`: Multi-round debate with steelman/critique structure
   - `prompts/`: All prompt templates used in this study
   - `example.py`: Abstract `LLMProvider` interface for custom backends
3. **What is not included:**
   - Production orchestration (database, API management, scaling)
   - Proprietary optimizations beyond the published algorithms
4. **Benchmark Data:** Raw prompts are sourced from publicly available datasets (GPQA, TruthfulQA, MMLU, IFEval, LiveCodeBench). We do not redistribute these datasets but provide scripts to reproduce our exact benchmark configuration.

Note: The reference implementation demonstrates the core methodology. It is not a production-ready system but provides sufficient detail for independent researchers to replicate our approach.

Appendix G: Calibration Run Findings

Before running the full benchmark, we conducted calibration runs on GPQA Diamond ($n = 198$) to verify our pipeline matched vendor-reported performance. This calibration against known ground truth adds scientific rigor by demonstrating that our methodology reproduces published results rather than relying on blind API calls.

G.1 Calibration Configurations

Two configurations were tested:

- **CAL-DM-DEF:** Default settings (standard API behavior, no special reasoning modes)
- **CAL-DM-MAX:** Maximum reasoning settings (extended thinking, high reasoning effort)

G.2 Results

Model	Vendor	CAL-DM-DEF	CAL-DM-MAX	Best Δ
Claude Opus 4.5	87.0%	**86.9%**	83.8%	-0.1pp
Gemini 3 Pro	91.9%	91.4%	**91.9%**	0.0pp
GPT-5.2	92.4%	77.2%	**90.1%**	-2.3pp

Table 13: Calibration run accuracy on GPQA Diamond. Δ indicates deviation from vendor-reported performance. Bold indicates configuration closest to vendor benchmark.

G.3 Key Observations

The calibration revealed that "more thinking" is not universally beneficial for convergent tasks like multiple-choice questions:

Claude Opus 4.5: Extended thinking degraded accuracy. Default settings (86.9%) matched Anthropic's reported 87% within 0.1pp, while extended thinking dropped to 83.8%—a 3.2pp degradation. This suggests that for MCQ tasks, extended reasoning may introduce noise or "overthinking" on straightforward questions.

GPT-5.2: Explicit reasoning_effort required. Default API behavior (77.2%) underperformed by 15.2pp. Only with `reasoning_effort: "xhigh"` did accuracy reach 90.1%, approaching OpenAI's reported 92.4%. The API default of "none" is insufficient for high-level reasoning tasks.

Gemini 3 Pro: Robust across configurations. Both settings achieved ∼91-92%, matching Google's benchmark. Thinking level had minimal impact on accuracy.

G.4 Production Methodology

For the main benchmark, we use uniform prompts and comparable settings across all models, accepting the resulting performance variance. The calibration data demonstrates that our pipeline can reproduce vendor-reported results when using vendor-specific configurations, validating that any observed differences in the main benchmark reflect genuine model behavior rather than implementation artifacts.

Calibration data is preserved with `run_type='calibration'`, excluded from production statistics but available for methodology audit.