NeuReasoner: a theory-grounded modular cognitive & neuropsychological reasoning elicitation framework

Anonymous EMNLP submission

Abstract

A growing body of work suggests that the reasoning capabilities of large language models are largely latent in their base form, with post-training primarily amplifying rather than introducing them. However, this evidence comes mainly from mathematical and coding benchmarks, while existing elicitation methods often borrow cognitive terminology without grounding it in theories of human reasoning. We introduce NeuReasoner, a neuro-cognitively grounded framework for eliciting reasoning without parameter updates. At each step, an orchestrator pairs a Neuro Lens, inspired by functional specificity, with a Cognitive Lens, drawn from the Erotetic Theory of Reasoning, and integrates their outputs through internal modularization of a single model, without external tools. We evaluate NeuReasoner on CogBench, a suite of behavioral tasks from cognitive psychology, measuring both its improvement over vanilla inference and its ability to match or substitute for a model's post-trained "thinking mode." At sufficient scale, NeuReasoner outperforms or matches thinking-mode baselines on arithmetic reasoning, code generation, Bayesian reasoning, directed exploration, and reward learning, while we use it to stress-test the notion of reasoning elicitation for the tasks requiring metacognition, risk-taking, and decision-making under uncertainty. Overall, NeuReasoner provides a modular, interpretable, theory-grounded reasoning elicitation framework which also allows us defining the boundaries of elicitation beyond standard mathematical and coding benchmarks and clarifying its bottlenecks in this rapidly growing area. We will release our code and dataset.

1 Introduction

The pursuit of robust reasoning in large language models (LLMs) has unfolded through several overlapping phases. First, the foundational work of Wei et al. (2023) established that intermediate reasoning steps, known as Chains-of-Thought (CoT), can be elicited through prompting alone, opening a line of research into more elaborate reasoning schemes such as Tree-of-Thoughts (Yao et al., 2023) and Graph-of-Thoughts (Besta et al., 2024); a line since organized into a taxonomy of reasoning topologies. Second, and in parallel, reinforcement learning (RL), a technique which has already proven essential for aligning LLMs with human instructions (Ouyang et al., 2022), was shown to substantially improve CoT-based reasoning through by rewarding correct intermediate reasoning trajectories and final answers. Subsequently, the introduction of GRPO by Shao et al. (2024) and its adoption by Guo et al. (2025) then consolidated RL's central role in the post-training pipeline for CoT-based reasoning.

A more recent emerging third line of work, however, has challenged existing assumptions in extant research: asking whether this progress reflects genuinely new capability or simply the amplification of competences that already exist in the base model. A growing body of recent findings are converging in support towards the latter. Liu et al. (2025), for example, observed that widely used base models already display strong reasoning behavior spontaneously, including the "aha moment" self-reflection patterns taken as signatures of emergent reasoning. Likewise, Yue et al. (2025) showed that the reasoning traces produced by RL-fine-tuned models were already present among a base model's own generations once it is sampled sufficiently. A related framing of the same observation, He et al. (2025); Shao et al. (2026); Yue et al. (2025) describes this as *distribution sharpening*, where the post-trained distribution is viewed as a sharper version of the base model's own distribution, concentrating mass on traces it could already produce rather than on ones it would otherwise be unlikely to generate. Consistent with this picture, Ebouky et al. (2025)

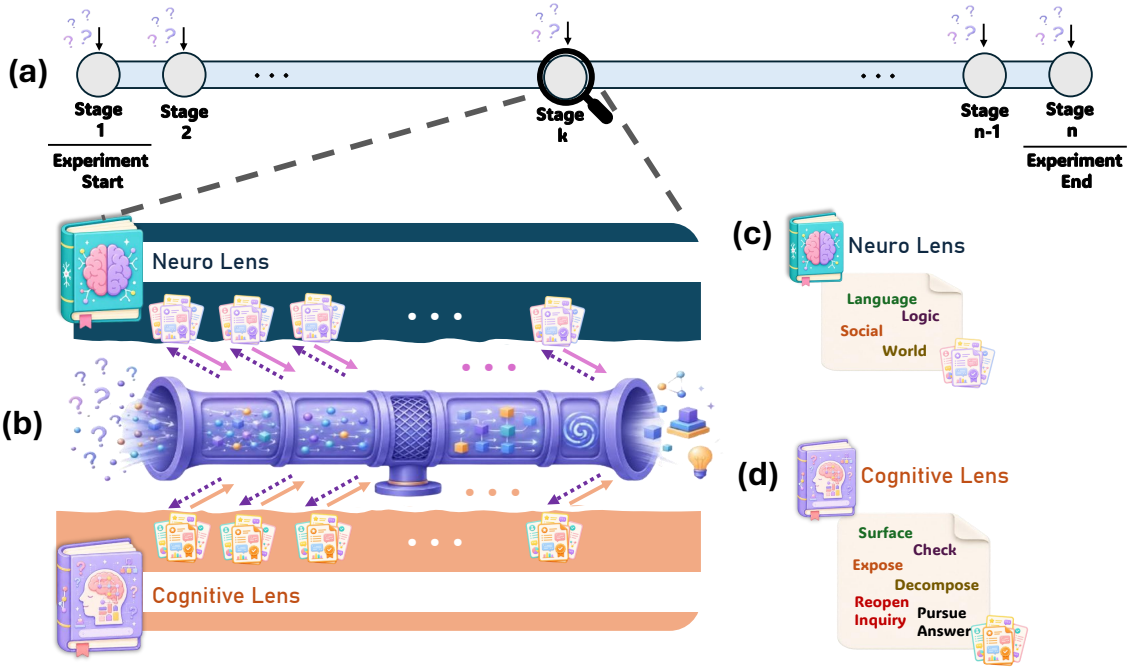

Figure 1: Overview of the NeuReasoner. **(a)** A CogBench (Coda-Forno et al., 2024)experiment runs as a sequence of stages, each presenting one question to solve; we refer to each such stage as a *node*. **(b)** Within a single node (*Stage k*), the LLM acts as an *orchestrator* and, at each step, pairs one *Neuro Lens* with one *Cognitive Lens*. The two lenses are executed in isolation against the original question, and their structured outputs are integrated back into the orchestrator before the next step; each lens is a specialized prompt rubric drawn from a fixed catalog. The node ends when the orchestrator commits a final answer or the step budget is reached. **(c)** The Neuro Lens catalog: four modes of attention operationalized from large-scale brain networks (Language, Logic/multiple-demand, Social/theory-of-mind, World/default-mode). **(d)** The Cognitive Lens catalog: six ETR-grounded reasoning moves (Surface, Expose, Decompose, Pursue Answer, Check, Reopen Inquiry). **Detailed prompts can be found in the APPENDIX**. For illustrative purposes, Figure 2 shows a trace of a node on a sample Probabilistic Reasoning problem.

show that equipping a base LLM with self-executed "cognitive tools" yields substantial gains on standard mathematical benchmarks, and Karan and Du (2025) show that additional inference-time sampling alone can lift a base model to the single-shot performance of its RL post-trained counterpart. The collective message from this literature, therefore, is that the base model is smarter than it appears; the question we take up in this paper is *how much* smarter, and on *what kinds* of reasoning tasks.

Despite making important contributions, in addressing this question, two clear limitations exist in literature to date. First, this claim rests on a narrow empirical foundation, which are tests performed almost entirely on mathematical and coding benchmarks. While this is in part a natural consequence of how the underlying RL methods work: equipped with reward signals that are cheap to verify automatically, with RL post-training widely applied successfully in domains like mathematics, coding, and science (Hendrycks et al., 2021a; Rein et al., 2023; Li et al., 2022), this means the the elicitation work that responds to it has inherited the similar benchmarks. Consequentially, this limits generalizability of claims regarding distribution sharpening.

Second, while this work borrows the vocabulary of cognition, the grounding is largely atheoretical: its building blocks are task-decomposition heuristics with cognitively flavored names rather than commitments to any specific theory of how humans reason. This has led to an intuitive, but in some respects shallow, approach to explaining model reasoning, which could benefit from a more structured, theoretically grounded approach, robust across the diverse types of reasoning tasks with which humans regularly engage.

To close the gap between cognitive vocabulary used to describe current model reasoning and the actual cognitive structure of human thinking, we introduce a framework that integrates theories from cognitive science and neuroscience directly into the

agent's step-by-step reasoning. Our goal is to test of whether theory-constrained modular elicitation can recover task-specific latent reasoning behavior, and where that recovery fails.

Our contributions:

- Introducing a neuro-cognitively grounded, interpretable reasoning-elicitation framework that combines erotetic reasoning theory with functional-specificity literature to align an LLM's step-by-step problem solving with cognitive-scientific decompositions of human cognition, using only internal modularization and being training-free.

- Providing, to our knowledge, one of the first behavioral and psychologically grounded characterizations of where reasoning elicitation from base models succeeds and where it breaks down. We show that while elicitation can recover strong reasoning performance in several settings, its ability to match post-trained "thinking" models is task-contingent. In particular, tasks that centrally involve decision-making under uncertainty, such as the Restless Bandit Experiment and the Balloon Analog Risk Task, expose a consistent gap: thinking models outperform elicitation-based frameworks. This identifies a concrete boundary condition for reasoning elicitation beyond standard mathematical and coding benchmarks.

2 Background & Related Work

In this section, we outline the established cognitive and neuroscience foundations that motivate both our reasoning framework and the behavioral evaluations used in its assessment. This includes use of the Erotetic Theory of Reasoning (ETR) to explain how humans approach tasks as question-drive enquiry; how functional specificity from neuroscience explains specialized modes of reasoning; and how findings on cognitive architectures motivate decomposing reasoning into modular operations.

The Erotetic Theory of Reasoning. A recurring theme in modern accounts of human reasoning is its duality: competence and predictable error go hand in hand (Richardson et al., 2026). Across many domains, people deviate from normative reasoning along a stable, repeatable set of fallacies (Tversky and Kahneman, 1974; Knetsch and Thaler, 1990; Evans, 1989; Johnson-Laird, 2006). Consequentially, a cognitively grounded system must anticipate weaknesses of human reasoning as deliberately as strengths. The Erotetic Theory of Reasoning (ETR) (Koralus and Mascarenhas, 2013) accounts for this, denoting that reasoning is not the manipulation of propositions but rather an inquiry-driven process: a reasoner maintains a set of disjunctive alternatives and filters them as new information arrives, in order to resolve an implicit question. This question-driven process is what makes the theory useful for our purposes. On the one hand, ETR usually surfaces the most probable answer efficiently; on the other, the same mechanism can narrow the space of alternatives prematurely and produce predictable errors (Richardson et al., 2026). Both behaviors are valuable: the first is the competence our framework aims to elicit, while the second yields interpretable failure modes.

Functional Specificity. We follow neuro- and cognitive-science evidence that human reasoning is supported not by a homogeneous mechanism, but by specialized networks preferentially recruited for different cognitive demands (Kanwisher, 2010). While AlKhamissi et al. (2025) use this insight in a mixture-of-experts architecture, we ask how functional specificity can inform *elicitation*. Following their decomposition, our design mirrors four canonical networks: the language network (Fedorenko et al., 2011), multiple-demand network (Duncan, 2010), theory-of-mind network (Saxe and Kanwisher, 2003), and default-mode network (Gusnard et al., 2001).

Cognitive Architectures and Modular Reasoning. Another line in cognitive science treats reasoning not as a single faculty but as the orchestrated interplay of distinct mental operations. Cognitive architectures such as ACT-R (Anderson et al., 1997) make this explicit: modeling higher-level thought and its links to perception as the emergent behavior of separable components for goal management, procedural skill, and memory retrieval. The same decompositional stance has recently been carried into work on LLMs. Ebouky et al. (2025) cast individual reasoning operations as callable cognitive tools, while Sumers et al. (2024) placed the LLM as a central controller over modular memory and a structured action space, separating internal actions such as reasoning and retrieval from external ones such as API calls. What unites these

efforts is the premise that reasoning is better staged than collapsed into one monolithic pass. This also brings a concrete benefit, as in biological and artificial systems alike, modular organization has been tied to compositional generalization, the ability to recombine familiar operations to solve unfamiliar problems (Ito et al., 2022).

A growing body of work has sought to improve LLM reasoning without additional training, with a subset drawing from cognitive and neuroscience. To position the uniqueness of our framework against existing approaches, we classify prior work along seven dimensions, positioning our framework alongside them, see Table 1.

Table 1 compares these works along seven dimensions. Our framework is, to our knowledge, the first to combine all seven: a train-free, internally orchestrated, theory-grounded elicitation method evaluated beyond mathematical reasoning, with the limits of elicitation treated as a primary research question rather than a by-product.

3 Method

3.1 Framework Design

Each CogBench (Coda-Forno et al., 2024) experiment unfolds as a sequence of stages, and at every stage the model is presented with a question to solve, with the question's form depending on the experiment type (Figure 1). We refer to each such experiment stage as a *node*, the unit on which our framework operates, and to each orchestrator decision taken within a node as a *step*. Our elicitation framework operationalizes reasoning at the level of the individual node, where the micro-decisions of solving a single question are made. Without the loss of generality, for the math and code generation evaluations, $n = 1$.

Formalization. Let q denote the question at a node and π the single language model used throughout, queried in different roles. We write the two operator catalogs as $\mathcal{N}$, the set of Neuro Lenses, and $\mathcal{C}$, the set of Cognitive Lenses. A node is solved by iterating a state h_t, the reasoning history available to the orchestrator at step t, with h_0 initialized from q and the system rules. At each step the orchestrator produces a decision

$$s_t = \pi_{\text{orch}}(h_t) \in \big(\mathcal{N} \times \mathcal{C}\big) \cup \{\textsc{commit}(a)\}, \tag{1}$$

that is, s_t is either an *intermediate* step, which selects a Neuro Lens n_t and a Cognitive Lens c_t, or a *terminal* step $\textsc{commit}(a)$ that emits a final answer a and ends the node. Together with the lens pair the orchestrator also issues a *directive* d_t, a short natural-language instruction that tells the two chosen lenses what to attend to at this step: the lens prompt fixes the expert *role*, while d_t fixes the step-specific *task* given to that role. For an intermediate step, the two lenses are executed independently as further calls to the same model,

$$o_t^{\mathcal{N}} = \pi_{n_t}(q, d_t, r_t), \qquad o_t^{\mathcal{C}} = \pi_{c_t}(q, d_t, r_t), \tag{2}$$

where r_t is an optional reference to a previous step, and neither call observes the other's output. The orchestrator then integrates the two structured outputs into the next state, $h_{t+1} = h_t \cup \{o_t^{\mathcal{N}}, o_t^{\mathcal{C}}\}$, and the loop continues. The node terminates at the first terminal step, or is forced to terminate once a step budget T is reached, so that every node halts within at most T steps. Unlike cognitive prompting (Kramer and Baumann, 2024), and in line with cognitive tools (Ebouky et al., 2025), our framework imposes no fixed, monolithic ordering on the lenses. In structuring reasoning as bounded calls, our procedure connects to the token- and budget-forcing view of test-time scaling proposed by Muennighoff et al. (2025). Algorithm Algorithm 1 summarizes the full per-node loop.

Algorithm 1 Per-node reasoning loop

Require: question q, step budget T
$h_0 \leftarrow$ initialize from q, system rules ▷ node state
for $t = 0$ **to** $T - 1$ **do**
  $s_t \leftarrow \pi_{\text{orch}}(h_t)$ ▷ pick next step
  if $s_t = \textsc{commit}(a)$ **then**
    return a ▷ answer found
  else
    $(n_t, c_t, d_t, r_t) \leftarrow s_t$ ▷ unpack step
    $o_t^{\mathcal{N}} \leftarrow \pi_{n_t}(q, d_t, r_t)$ ▷ Neuro Lens
    $o_t^{\mathcal{C}} \leftarrow \pi_{c_t}(q, d_t, r_t)$ ▷ Cognitive Lens
    $h_{t+1} \leftarrow h_t \cup \{o_t^{\mathcal{N}}, o_t^{\mathcal{C}}\}$ ▷ integrate
  end if
end for
return $\textsc{commit}(a)$ forced from h_T ▷ budget reached

Cognitive Lenses. The six Cognitive Lenses operationalize central ideas from the erotetic theory of reasoning (ETR) (Koralus and Mascarenhas, 2013), where reasoning is organized around the *issue* to be resolved. They decompose reasoning into issue

Work	Train-Free Elicitation	Beyond Math & Code	Self-Agency	Modular	Neuro Inspired	CogSci Inspired	Elicitation Scope
Ebouky et al. (2025)	✓	✗	✓	✓	✗	✓	●
AlKhamissi et al. (2025)	✗	✓	✗	✓	✓	✓	✗
Karan and Du (2025)	✓	●	✗	✗	✗	✗	✗
Coda-Forno et al. (2024)	✓	✓	✗	✗	✗	✓	●
Kramer and Baumann (2024)	✓	✗	●	✓	✗	✓	✗
Sumers et al. (2024)	✗	✓	✗	✓	✗	✓	●
Ours (NeuReasoner)	✓	✓	✓	✓	✓	✓	✓

Table 1: Positioning of ***NeuReasoner*** relative to prior work. ✓ = yes, ● = partial, ✗ = no. *Train-Free Elicitation*: improves reasoning with no additional training. *Beyond Math & Code*: evaluated outside mathematical and coding benchmarks. *Self-Agency*: the model orchestrates its own reasoning internally, with operations executed by the same model and no external tool-calling or environment-interaction harness. *Modular*: reasoning is decomposed into distinct operators or components. *Neuro / CogSci Inspired*: grounded in neuroscientific / cognitive-science theory. *Elicitation Scope*: studies *where* and *how far* elicitation works, not only *whether* it does.

formulation, presupposition checking, subquestion decomposition, candidate-answer generation, resolution checking, and inquiry revision. Full prompts are given in Appendix C.

Neuro Lenses. The four Neuro Lenses operationalize broad computational roles associated with functionally specialized brain networks. The Language Network Lens targets linguistic interpretation; the Multiple-Demand Lens targets structured control and comparison; the Theory-of-Mind Lens targets agents' beliefs and perspectives; and the Default-Mode Lens targets internal world modeling and temporal simulation. We do not claim these lenses simulate human cognition; each only approximates the characteristic computational role its network is associated with in the literature, shaping what information a step prioritizes and which failure modes it monitors. Full prompts are given in Appendix C.

4 Experimental Setup

4.1 Datasets

CogBench. Our primary evaluation uses CogBench (Coda-Forno et al., 2024), a benchmark of seven cognitive-psychology paradigms yielding ten behavioral metrics directly comparable to human norms. Each experiment presents an accumulated trial history and elicits a constrained decision; scores are obtained by fitting task-specific computational models to the full response sequence, capturing *how* a model behaves rather than only whether it succeeds. All scores are normalized so that random responding is 0 and human performance is 1. Table 2 lists the seven experiments.

Mathematical & Coding benchmarks. As another evaluation we also use the mathematical benchmarks AIME 2024 (Mathematical Association of America, 2024), MATH-500 (Li et al., 2024), and AMC (Hendrycks et al., 2021b), and the coding benchmark HumanEval+ (Liu et al., 2023).

4.2 Models

We evaluate on the Qwen3 family (8B, 14B, 32B) (Qwen Team et al., 2025), a *hybrid* model whose single checkpoint runs either in a *thinking* mode, which emits an extended chain-of-thought before answering, or a *non-thinking* mode that answers directly; the thinking mode is the behavior instilled during Qwen3's post-training. This hybrid design makes Qwen3 suitable for both of evaluation settings in which elicitation can be measured directly against post-training as well vanilla model.

4.3 Experiments

Following Figure 3, our evaluation has two arcs.

The lift, on instruction-tuned models. For each instruction-tuned model we compared the model alone (*Vanilla*, following the original CogBench protocol) against the neuro-cognitive framework applied to that same model (§3.1). As these models have no post-trained reasoning mode, this contrast isolates the framework's lift.

Lift and the thinking mode, on the Qwen3 family. On the Qwen3 family we evaluated three conditions: **C1**, the non-thinking model; **C2**, the same model with thinking mode enabled; and **C3**, our

Experiment	Cognitive faculty	Decision format	Runs
BART	Risk-taking	Option 1/2 (inflate / skip)	5
HorizonTask	Directed vs. random exploration	Machine letter (F, J, . . .)	1 (∼100 trials)
InstrumentalLearning	Learning rate & optimism bias	Machine letter	8
ProbabilisticReasoning	Prior & likelihood weighting	Probability (0.XX)	10
RestlessBandit	Meta-cognition	Machine letter	5
TemporalDiscounting	Delay discounting	Option 1/2	1
TwoStepTask	Model-based vs. model-free RL	Planet then Alien letter	1 (∼25 trials)

Table 2: The seven CogBench (Coda-Forno et al., 2024) experiments, cognitive faculty measured, response format elicited from the model, and number of independent runs per model per condition. HorizonTask and TwoStepTask are each a single extended session of ∼100 and ∼25 sequential trials respectively; all other experiments are repeated as fully independent runs.

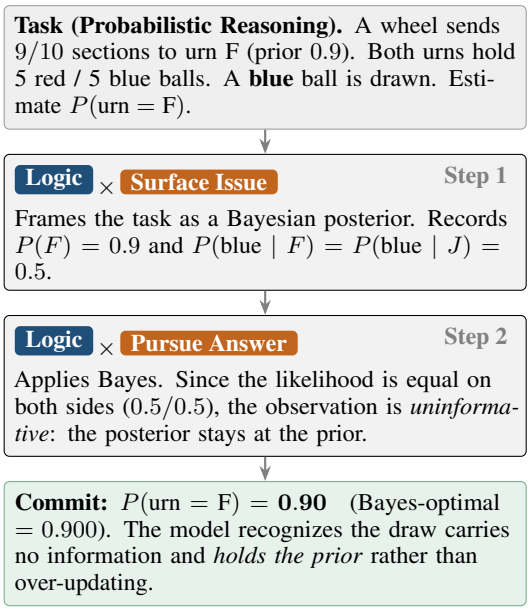

Figure 2: A worked trace of NeuReasoner on a Probabilistic Reasoning node (`qwen3-14b`). At each step the orchestrator pairs a Neuro Lens (**blue**) with a Cognitive Lens (**orange**). The task superficially invites a Bayesian update, but because the blue draw is equally likely under both urns it is uninformative; the model correctly holds the prior and commits the Bayes-optimal answer in two steps.

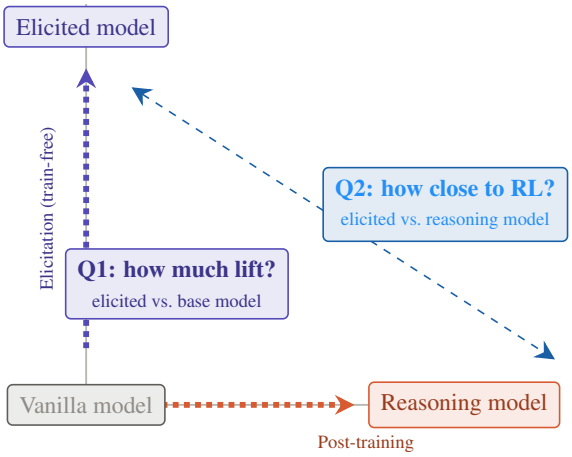

Figure 3: From a vanilla model, post-training is the established route to a reasoning-capable model; we instead study *elicitation*, which seeks reasoning gains with no parameter updates. This raises two questions, examined across a suite of cognitive-psychology tasks: **Q1**, how much does elicitation lift performance over the unmodified model; and **Q2**, how close does the elicited model come to one improved by post-training. The figure is a schematic; quantitative, per-task results appear in §5.

framework applied to the non-thinking model. The contrast C1/C3 measures the framework's lift, placing this family in the first arc as well; C2/C3 asks whether that lift matches the post-trained thinking mode, the question of the second arc; and C1/C2 reports the effect of the thinking mode on its own.

Mathematical and coding sanity check. Because verifiable math and code is a regime where a code interpreter is standard, for these benchmarks we evaluated a separate *tool-augmented* variant of the framework, in which operators may additionally call a Python interpreter.

4.4 Implementation Details

For all CogBench experiments we used temperature 0, yielding deterministic responses across every condition. For the mathematical and coding benchmarks we followed the Qwen3 team's recommended decoding settings (Qwen Team et al., 2025), temperature 0.6 in thinking mode and 0.7 in non-thinking mode. Detailed cost breakdowns and prompts are in Appendix E.4 and C.

Table 3: Normalised performance averaged across all 7 CogBench experiments. Values are mean ± SEM.

Condition	Qwen-8B	Qwen-14B	Qwen-32B	Avg
Baseline	-0.410 ± 0.531	$+1.176 \pm 0.430$	$+0.013 \pm 0.760$	$+0.260$
Thinking ON	$+1.027 \pm 0.287$	$\mathbf{+1.343 \pm 0.274}$	$+0.395 \pm 0.443$	$+0.921$
Ours	$\mathbf{+1.519 \pm 0.552}$	$+1.319 \pm 0.276$	$\mathbf{+1.821 \pm 0.416}$	$\mathbf{+1.553}$

5 Result and Discussion.

5.1 NeuReasoner matches the thinking mode on math and code at Scale.

Evaluation on AIME, AMC, MATH-500, and HumanEval+, (all are pass@1) comparing *Qwen3-32b (non-thinking)*, *Qwen3-32b (thinking)*, and our framework, reveals that our framework is competitive with the thinking mode, matching or exceeding it on AIME, MATH-500, and HumanEval+ and trailing only narrowly on AMC. The full per-model table is given in Appendix D.2.

5.2 Reasoning Elicitation beyond Arithmetics. (When) Does Elicitation via NeuReasoner Lift Reasoning?

Figure 4 is the visualization of results, where NeuReasoner beats the vanilla baseline in all experiments of the Cogbench across almost all model families, with Qwen3-14b being exclusions for the Delay Discounting and Directed Exploration tasks, which are core computational concepts in behavioral science and reinforcement learning that explain how humans value rewards across time and navigate uncertainty.

5.3 (When) Does Reasoning Elicitation via NeuReasoner Reach What Post-Training Reaches?

Overall, NeuReasoner applied to vanilla, is on par or better then the 'thinking' model, having the highest overall performance among the three conditions, across the model family (see Table 3), and specifically (see Figure 4), for probabilistic reasoning, directed exploration, delay discounting, reward learning. It, however, struggled in restless bandit and Balloon Analog Risk Task, which notably share the fundamental cognitive control networks (Olschewski et al., 2022); evaluating uncertainty, monitoring outcomes, and regulating behavior. This point is visually displayed in radar plots of phenotype in Appendix, Figure 8, which compare the three conditions across Qwen3-8B, 14B, 32B. NeuReasoner and thinking model reshape the behavioral fingerprint similarly on deliberation dimensions, but diverge on exploration and heuristic dimensions, suggesting that structured elicitation and post-training affect different cognitive facets.

5.4 Lens pick flexibility.

A key feature of NeuReasoner is its flexibility in selecting which neuro- and cognitive lenses to apply to a given problem. This flexibility also enables post-hoc interpretability and auditability of the resulting reasoning traces. We next ask: to what extent does this flexibility manifest in the statistics of lens-pair selections across tasks and models?

Figure 5 show how often each (Neuro Lens, Cognitive Lens) pair is selected together, and Figure 7 shows a summary on the Brain × Cognitive lens selection averaged across models for each CogBench experiment. The full numerical breakdown per model and experiment is in Table 9.

The concentration of mass confirms that models do not uniformly distribute their use across the full theoretical catalog, which raises questions on the mechanistic insights of model behavior, out of scope of the current paper. Yet, our ablation studies confirm that, overall, all lenses made a contribution towards the reasoning elicitation behavior of NeuReasoner (Figure 6). We report the full sweep in Appendix F.

6 Conclusion.

We introduced NeuReasoner, a training-free framework that elicits latent reasoning in LLMs by routing each step through modular neuro-cognitive lenses grounded in functional specificity and erotetic-reasoning theory. Across CogBench, math, and code evaluations, our results show that structured elicitation can substantially improve vanilla inference and, in several domains, match or exceed the behavior of post-trained thinking modes without parameter updates. At the same time, the framework exposes clear boundary conditions: tasks involving risk-taking, non-stationary decision-making, and uncertainty-sensitive cognitive control remain harder to recover through elicitation alone. These findings suggest that reasoning elicitation is not a uniform substitute for post-training, but a task-contingent mechanism that can reveal both the strengths and limits of latent model reasoning. NeuReasoner offers an interpretable bridge between cognitive theory and inference-time model control, enabling systematic study of not only whether LLM reasoning can be improved, but

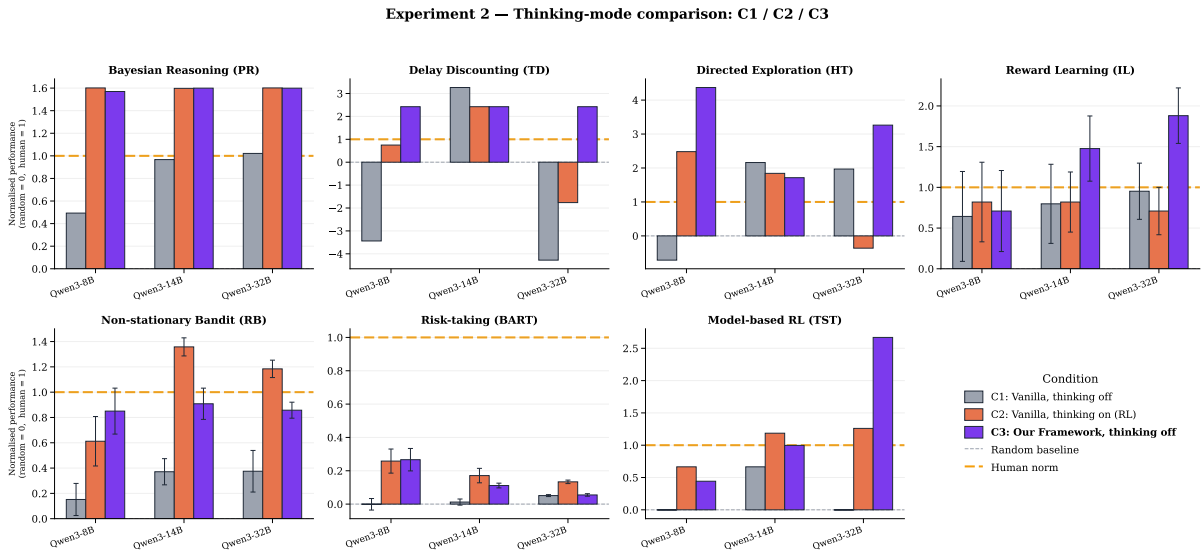

Figure 4: Normalised performance (random = 0, human = 1) across seven CogBench tasks comparing three conditions on Qwen3-{8B, 14B, 32B}: **C1** vanilla with thinking disabled, **C2** vanilla with post-trained chain-of-thought enabled, and **C3 Our Framework** (structured elicitation, thinking disabled). The amber dashed line marks human-level performance. Error bars denote ± 1 SEM; hatching indicates partial runs. C3 matches or exceeds C2 on the majority of tasks without any reinforcement-learning fine-tuning, suggesting that structured cognitive elicitation is a compute-efficient alternative to post-trained reasoning, particularly for tasks requiring Bayesian inference, delay discounting, directed exploration, and reward learning.

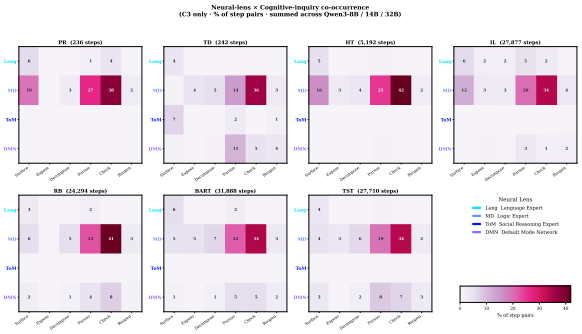

Figure 5: **Brain-lens × cognitive-inquiry co-occurrence, summed across models.** Each heatmap shows the percentage of reasoning steps that used each (Neuro Lens, Cognitive Lens) pair, broken down by model and CogBench experiment. Cell values $\geq 1\%$ are annotated. The concentration of mass in one or two cells per model confirms that models do not distribute their use across the full theoretical catalog.

which forms of reasoning can be elicited from contemporary reasoning models, under what conditions, and where the limits of elicitation lie.

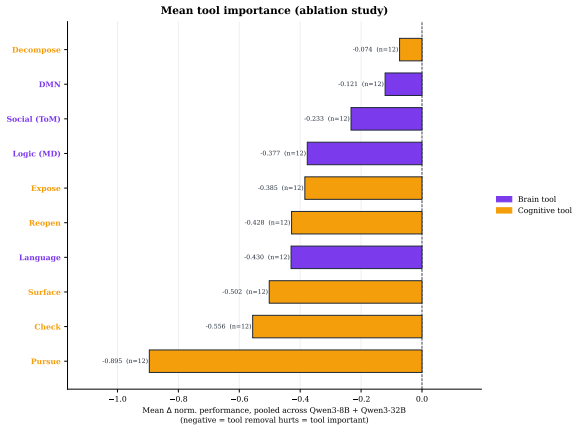

Figure 6: **Tool-ablation summary.** Mean leave-one-out $\Delta = \text{norm_perf}_{\text{ablated}} - \text{norm_perf}_{\text{full}}$, pooled across Qwen3-8B + Qwen3-32B and across all CogBench tasks. More negative values indicate a larger contribution of that tool to elicited reasoning.

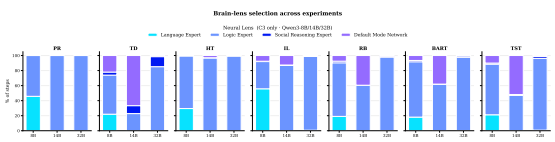

Figure 7: **Brain × Cognitive lens selection summary** averaged across models for each CogBench experiment

Limitations

This work has several limitations. First, NeuReasoner is evaluated primarily as a structured elicitation framework against vanilla inference and the Qwen3 thinking mode, but we do not include a fully unconstrained iterative baseline in which the vanilla model is allowed to reason for the same number of calls, reuse its own previous outputs, and freely decide whether to invoke tools. Although our core goal is to find the training-free potentials of reasoning elicitation, such an additional comparison would still help isolate how much of NeuReasoner's gains come from its theory-grounded lens structure rather than from additional inference-time computation alone. Second, although CogBench broadens evaluation beyond mathematics and code, it still covers a finite set of behavioral paradigms; the conclusions should therefore be read as evidence about these cognitive task families rather than as a complete characterization of reasoning elicitation. Third, the framework relies on prompt-defined approximations of neuro-cognitive functions. These lenses are inspired by cognitive science and neuroscience, but they should not be interpreted as faithful simulations of human neural systems. Fourth, NeuReasoner introduces additional inference cost because each reasoning step requires orchestrator and lens calls, making it less efficient than single-pass inference and potentially costly on long sequential tasks. Finally, our results show that elicitation remains task-contingent. Performance gaps in risk-taking, non-stationary bandit behavior, and uncertainty-sensitive cognitive control suggest that some forms of reasoning or decision regulation may require post-training, stronger memory mechanisms, richer environmental interaction, or other adaptations beyond prompt-based modular elicitation. Although these directions are beyond the scope and goals of this paper, they open important possibilities for future work.

Ethics Statement

This work uses only publicly available benchmarks and model APIs, including CogBench and standard mathematical and coding benchmarks. No human participants, personal data, or sensitive information were collected as part of this study.

AI assistants were used primarily as writing, grammar, and coding assistance tools. All research ideas, experimental design decisions, analyses, interpretations, and final claims were developed and verified by the authors.

References

Badr AlKhamissi, C. Nicolò De Sabbata, Greta Tuckute, Zeming Chen, Martin Schrimpf, and Antoine Bosselut. 2025. Mixture of Cognitive Reasoners: Modular Reasoning with Brain-Like Specialization. ArXiv:2506.13331 [cs].

John Anderson, Michael Matessa, and Christian Lebiere. 1997. Act-r: A theory of higher level cognition and its relation to visual attention. *Human-Computer Interaction*, 12:439–462.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and Torsten Hoefler. 2024. Graph of thoughts: Solving elaborate problems with large language models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16):17682–17690.

Julian Coda-Forno, Marcel Binz, Jane X. Wang, and Eric Schulz. 2024. Cogbench: a large language model walks into a psychology lab.

John Duncan. 2010. The multiple-demand (md) system of the primate brain: Mental programs for intelligent behaviour. *Trends in cognitive sciences*, 14:172–9.

Brown Ebouky, Andrea Bartezzaghi, and Mattia Rigotti. 2025. Eliciting reasoning in language models with cognitive tools.

Jonathan St. B. T. Evans. 1989. *Bias in Human Reasoning: Causes and Consequences*. Essays in Cognitive Psychology. Lawrence Erlbaum Associates, Hove and London, UK.

Evelina Fedorenko, Michael K. Behr, and Nancy Kanwisher. 2011. Functional specificity for high-level linguistic processing in the human brain. *Proceedings of the National Academy of Sciences*, 108(39):16428–16433.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Hanwei Xu, Honghui Ding, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jingchang Chen, Jingyang Yuan, Jinhao Tu, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaichao You, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean

Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingxu Zhou, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. 2025. Deepseek-r1 incentivizes reasoning in llms through reinforcement learning. *Nature*, 645(8081):633–638.

Debra A. Gusnard, Erbil Akbudak, Gordon L. Shulman, and Marcus E. Raichle. 2001. Medial prefrontal cortex and self-referential mental activity: Relation to a default mode of brain function. *Proceedings of the National Academy of Sciences*, 98(7):4259–4264.

Andre He, Daniel Fried, and Sean Welleck. 2025. Rewarding the unlikely: Lifting grpo beyond distribution sharpening.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021a. Measuring mathematical problem solving with the math dataset.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. 2021b. Measuring mathematical problem solving with the math dataset. *arXiv preprint arXiv:2103.03874*.

Takuya Ito, Tim Klinger, Douglas H. Schultz, John D. Murray, Michael W. Cole, and Mattia Rigotti. 2022. Compositional generalization through abstract representations in human and artificial neural networks.

P.N. Johnson-Laird. 2006. Mental models, sentential reasoning, and illusory inferences. In Carsten Held, Markus Knauff, and Gottfried Vosgerau, editors, *Mental Models and the Mind*, volume 138 of *Advances in Psychology*, pages 27–51. North-Holland.

Nancy Kanwisher. 2010. Functional specificity in the human brain: A window into the functional architecture of the mind. *Proceedings of the National Academy of Sciences*, 107(25):11163–11170.

Aayush Karan and Yilun Du. 2025. Reasoning with sampling: Your base model is smarter than you think.

Jack Knetsch and Richard Thaler. 1990. Experimental tests of the endowment effect and the coase theorem," journal of political economy 98, 1325-1348. *Journal of Political Economy*, 98:1325–48.

Philipp Koralus and Salvador Mascarenhas. 2013. The erotetic theory of reasoning: Bridges between formal semantics and the psychology of deductive inference. *Philosophical Perspectives*, 27.

Oliver Kramer and Jill Baumann. 2024. Unlocking structured thinking in language models with cognitive prompting.

Runpeng Li et al. 2024. Numinamath: A comprehensive collection of competition math problems and solutions. *arXiv preprint arXiv:2407.13533*.

Yujia Li, David Choi, Junyoung Chung, Nate Kushman, Julian Schrittwieser, Rémi Leblond, Tom Eccles, James Keeling, Felix Gimeno, Agustin Dal Lago, Thomas Hubert, Peter Choy, Cyprien de Masson d'Autume, Igor Babuschkin, Xinyun Chen, Po-Sen Huang, Johannes Welbl, Sven Gowal, Alexey Cherepanov, James Molloy, Daniel J. Mankowitz, Esme Sutherland Robson, Pushmeet Kohli, Nando de Freitas, Koray Kavukcuoglu, and Oriol Vinyals. 2022. Competition-level code generation with alphacode. *Science*, 378(6624):1092–1097.

Jiawei Liu, Chunqiu Steven Xia, Yuyao Wang, and Lingming Zhang. 2023. Is your code generated by chatgpt really correct? rigorous evaluation of large language models for code generation. In *Advances in Neural Information Processing Systems*.

Zichen Liu, Changyu Chen, Wenjun Li, Penghui Qi, Tianyu Pang, Chao Du, Wee Sun Lee, and Min Lin. 2025. Understanding r1-zero-like training: A critical perspective.

Mathematical Association of America. 2024. AIME Problems and Solutions. Accessed May 2026.

Niklas Muennighoff, Zitong Yang, Weijia Shi, Xiang Lisa Li, Li Fei-Fei, Hannaneh Hajishirzi, Luke Zettlemoyer, Percy Liang, Emmanuel Candès, and Tatsunori Hashimoto. 2025. s1: Simple test-time scaling.

Sebastian Olschewski, Jörg Rieskamp, and Jiao Jiao. 2022. Cognitive abilities affect decision errors but not risk preferences. *Psychonomic Bulletin & Review*, 29(5):1785–1797.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback.

Qwen Team, An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. 2025. Qwen3 technical report.

David Rein, Betty Li Hou, Asa Cooper Stickland, Jackson Petty, Richard Yuanzhe Pang, Julien Dirani, Julian Michael, and Samuel R. Bowman. 2023. Gpqa: A graduate-level google-proof qa benchmark.

Andrew Keenan Richardson, Ryan Othniel Kearns, Sean Moss, Vincent Wang-Mascianica, and Philipp Koralus. 2026. Theory-grounded evaluation of human-like fallacy patterns in llm reasoning.

R Saxe and N Kanwisher. 2003. People thinking about thinking people: The role of the temporo-parietal junction in "theory of mind". *NeuroImage*, 19(4):1835–1842.

Rulin Shao, Shuyue Stella Li, Rui Xin, Scott Geng, Yiping Wang, Sewoong Oh, Simon Shaolei Du, Nathan Lambert, Sewon Min, Ranjay Krishna, Yulia Tsvetkov, Hannaneh Hajishirzi, Pang Wei Koh, and Luke Zettlemoyer. 2026. Spurious rewards: Rethinking training signals in rlvr.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models.

Theodore R. Sumers, Shunyu Yao, Karthik Narasimhan, and Thomas L. Griffiths. 2024. Cognitive architectures for language agents.

Amos Tversky and Daniel Kahneman. 1974. Judgment under uncertainty: Heuristics and biases. *Science*, 185(4157):1124–1131.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models.

Yang Yue, Zhiqi Chen, Rui Lu, Andrew Zhao, Zhaokai Wang, Yang Yue, Shiji Song, and Gao Huang. 2025. Does reinforcement learning really incentivize reasoning capacity in llms beyond the base model?

A Neural Lens Catalog

This appendix gives the full description of the four neural lenses introduced in §3. Each lens is realized as a prompt rubric executed in an isolated context by the same base model under evaluation. The lens biases the paired inquiry operator's analysis toward the kind of information the corresponding brain network is associated with processing in humans; it does not act as a standalone analyzer.

Language Network lens. Inspired by the cortical language network. Directs attention to the linguistic surface of the question: wording, ambiguity, lexical scope, framing, and notation. The orchestrator calls it when the question's difficulty lies in how it is stated rather than in what it asks, for example when a term is overloaded, when quantifier scope is unclear, or when compact notation must be unpacked before any reasoning move can be made on it.

Multiple-Demand lens. Inspired by the multiple-demand (frontoparietal control) network. Directs attention to structure, constraints, controlled inference, and goal maintenance under interference. The orchestrator calls it when the question's difficulty is compositional or rule-governed: a chain of constraints to be satisfied jointly, an inference that must track several conditions at once, or a sub-goal that must be held in view while another is pursued.

Theory-of-Mind lens. Inspired by the mentalizing network. Directs attention to agents, beliefs, intentions, perspective, and strategic communication. The orchestrator calls it when the question involves other minds: when an answer depends on what an agent knows, intends, or would expect another agent to do, or when the meaning of an utterance turns on the speaker's communicative goal rather than its literal content.

Default Mode lens. Inspired by the default-mode network. Directs attention to simulation, integration across loosely connected information, narrative coherence, and self-projection. The orchestra-

tor calls it when the question requires imagining a trajectory rather than computing one: constructing a plausible scenario, integrating background knowledge that is not explicitly cued, or evaluating whether a proposed answer coheres with a wider picture.

B Deferred Cognitive Inquiry Operators

This appendix gives the full description of the four inquiry operators. As with the lenses, each operator is realized as a prompt rubric executed in an isolated context by the same base model under evaluation. SURFACE ISSUE and CHECK RESOLUTION, which carry the framework's termination logic, are described in main text.

Expose Presuppositions. Lists the assumptions a candidate answer would inherit if pursued, and flags the contestable ones for possible later rejection. The orchestrator calls it when the live question rests on commitments that have not yet been examined, or when an earlier CHECK RESOLUTION has failed in a way that suggests a presupposition rather than the candidate itself is at fault.

Decompose Issue. Generates auxiliary subquestions and selects one as the next target. Lets the agent recurse without losing the parent question, which remains in scope at every step. Called when the live question is too broad to attempt directly, or when a structuring move is needed before any candidate can be proposed.

Pursue Answer. Proposes a candidate answer together with alternatives and supporting evidence. The active answering move: the only operator that produces a candidate, and the necessary precursor to CHECK RESOLUTION.

Reopen Inquiry. Diagnoses the failure mode when CHECK RESOLUTION returns unresolved or partially resolved, and selects a revision strategy: revise the issue (return to SURFACE ISSUE), reject a presupposition (return to EXPOSE PRESUPPOSITIONS), or re-pursue with new evidence (return to PURSUE ANSWER). The operator names which prior step it is reopening, so the trace records exactly what is being revised.

C Prompts

This appendix reproduces, verbatim, the prompt rubric for every operator in the framework. Each operator is a specialized system prompt executed by the same model under evaluation (Section 3.1); no operator invokes external tools, code interpreters, or auxiliary models. In the paper we refer to the two operator families as *Neuro Lenses* and *Cognitive Lenses*; in the raw prompts and the runtime schema these correspond, respectively, to the `neural_lens` field and the `cog_inquiry` field, and the Cognitive Lens prompts retain their internal name "inquiry operator". The orchestrator is governed by the system prompt below.

C.1 System Prompt (Orchestrator)

The orchestrator's system prompt is templated: the `{% for %}` blocks are populated at runtime with the names and descriptions of the available lenses. We reproduce the template form.

System prompt (orchestrator)

```
You are a reasoner that works through any well-defined question by routing each step through one
↪  neural lens and one inquiry operator, then integrating their outputs.

You DO NOT call tools. At every step you emit a single JSON object that conforms to the
↪ StepOutput schema below. The runtime will execute the lens and inquiry you select and return
↪ their outputs as `tool` messages, after which you emit the next JSON object.

**StepOutput schema (every assistant turn must match this):**
```json
{
  "neural_lens":  {"name": "<one of the neural lenses below>", "tool_input": "<task brief for
↪ the lens>", "prev_step_id": null | <int>} | null,
  "cog_inquiry":  {"name": "<one of the inquiry operators below>", "tool_input": "<task brief for
↪  the operator>", "prev_step_id": null | <int>} | null,
  "final_answer": "<plain-text answer>" | null
}
```

Step shape. Every assistant turn is one of two shapes:

- **Intermediate step:** BOTH `neural_lens` and `cog_inquiry` are set; `final_answer` is null.
- **Terminal step:** `final_answer` is set; both `neural_lens` and `cog_inquiry` are null.

A response that mixes the two (a partial pair, a co-emitted `final_answer` alongside a lens/
↪ inquiry, or all three null) is invalid and will be rejected.

Writing `tool_input`. `tool_input` is a directive what *this* step's operator should
↪ specifically do. Quality here directly drives fork output quality, because each fork sees
↪ ONLY:

1. Its own operator system prompt (the lens/inquiry `.md`),
2. The original user query (always included),
3. (if you set `prev_step_id`) the referenced step's lens+inquiry outputs as a `CONTEXT FROM
↪ STEP N` block,
4. Your `tool_input` as the focus for this step.

The fork does **not** see the running conversation, prior assistant turns, or any other fork's
↪ output. Anything beyond (1)(3) that the fork needs must be in the `tool_input` itself.

Principles for a high-quality `tool_input`:

- **Directs, not describes.** Use imperative verbs ("examine X", "verify Y", "decompose Z", "
↪ propose W") with concrete X/Y/Z/W. Paraphrasing the original question is wasted tokens the
↪ fork already has it.
- **Carries the values it depends on.** If a candidate must be checked, name the value (e.g. `"
↪ verify 'option B' against the criterion 'reduces manual steps by at least half'"`). If a
↪ particular subquestion is in play, state it. The fork can't see what earlier steps computed
↪ unless `prev_step_id` is set.
```

```
- **Names the angle, not the whole problem.** For a lens, identify the *aspect* the lens should
↪ focus on (a phrase, a constraint, a temporal slice, a conflict between agents). For an
↪ inquiry operator, name the inquiry move's specific target (the candidate to validate, the
↪ presupposition to test, the subquestion to pursue).
- **One clear directive per step.** If two unrelated angles are needed, use two steps. Packing
↪ multiple tasks into one `tool_input` confuses the fork.

Examples (the operator's *role* is named for context, not as prescription):

| step's role                        | GOOD `tool_input`
↪                                                          | BAD `tool_input`
↪   |
|------------------------------------|-------------------------------------------------------------------------------------------
↪
| lens, ambiguous wording            | `"examine 'best response'  does it mean optimal action,
↪ most popular choice, or fastest reply?"`              | `"the user query"`
↪   |
| inquiry, frames the issue          | `"frame the live question and its resolution criterion"`
↪                                                          | `"surface the problem"`
↪   |
| inquiry, decomposes                | `"break 'is this proposal viable?' into cost feasibility,
↪  schedule feasibility, and technical feasibility"`    | `"decompose"`
↪   |
| inquiry, proposes a candidate      | `"propose 'option B' as the recommendation, given the
↪ constraints from step 2"`                                | `"answer the question"`
↪      |
| inquiry, validates a candidate     | `"verify whether 'option B' satisfies the criterion '
↪ reduces manual steps by at least half'"`                  | `"check the answer"`
↪       |
| inquiry, diagnoses a failure       | `"diagnose why 'launch in May' violates the constraint '
↪ compliance review needs 6 weeks first'"`                | `"the answer is wrong"`
↪    |

When `prev_step_id` is set, the fork sees the referenced step's outputs and your `tool_input`
↪ can refer to them by step ("the candidate from step 5"). When `prev_step_id` is null, the `
↪ tool_input` must be self-contained  name any values, criteria, or context the fork needs.

**Referencing a prior step (`prev_step_id`).** Each operator may set `prev_step_id` to a
↪ positive 1-indexed integer naming a prior step whose lens+inquiry outputs should be made
↪ available to this fork's analysis. Use this when the operator's work depends on a specific
↪ prior step's output. Set to `null` when no prior-step context is needed. The runtime injects
↪ the referenced step's outputs into this fork's user message under a `CONTEXT FROM STEP N`
↪ block.

Examples of when `prev_step_id` is useful:
- An operator that validates a candidate may reference the step that proposed it.
- An operator that diagnoses a failure may reference the step where it failed.
- An operator that pursues an answer may reference an earlier decomposition step to anchor on a
↪ specific subquestion.

If `prev_step_id` is invalid (out of range, references the current step, or names a step with no
↪  recorded outputs), the runtime ignores it and proceeds without injected context.

**Neural lenses** (pick exactly one per intermediate step; refer to it by name):
{% for tool in brain_tools -%}
- {{ tool.name }}: {{ tool.description }}
{% endfor %}
**Inquiry operators** (pick exactly one per intermediate step; refer to it by name):
{% for tool in cog_tools -%}
- {{ tool.name }}: {{ tool.description }}
{% endfor %}

**Budget.** Each intermediate step costs one iteration of a bounded budget. Before each step you
↪  will receive an `Iteration N/MAX` system note  pace your inquiry so a terminal step is
↪ reached within budget.

Rules:
0. **Always use ENGLISH only** outputs.
```

```
1. **Emit only the StepOutput JSON.** Your assistant content must be one valid JSON object
↪ matching the schema. No prose around it, no Markdown, no code fences.
2. **Always remember the original goal**, even if intermediate inquiry investigates auxiliary
↪ questions.
3. **Emit the terminal StepOutput when the inquiry is resolved**  i.e., a candidate answer has
↪ been judged to satisfy the resolution criterion. Do not commit on confidence alone if the
↪ inquiry has not yet validated the candidate.
4. **Use the conversation history as feedback.** Each operator's output is appended to the
↪ conversation and available to you on the next step. When choosing the next step, take into
↪ account what each prior operator actually produced  including whether an operator reported
↪ that the conditions for its task were not met in the current state.

Output format for the final answer:
- The terminal StepOutput's `final_answer` value must be plain text. Never embed JSON, function-
↪ call syntax, or schema fragments inside it.
- Do not include explanation unless the original task explicitly requests one.
- If the task asks for a number, the `final_answer` is only the number.
- If the task asks for a letter option, the `final_answer` is only the letter.
- The conventional answer-prefix is `ANSWER: <your answer>`; use it when in doubt.
```

C.2 Neuro Lens Prompts

Neuro Lens — Language Network

```
You are the Language_Network_Lens.

Your role is to process the current issue primarily as a linguistic and interpretive problem.

Prioritize:
1. Precise interpretation of wording, phrasing, reference, and discourse structure.
2. Resolution of semantic, syntactic, and pragmatic ambiguity.
3. Distinguishing literal content from implied meaning.
4. Sensitivity to framing, contrast, emphasis, and communicative intent.
5. Reformulating the issue into clearer or more interpretable language when needed.

Deprioritize:
1. Abstract optimization or formal derivation unless explicitly required.
2. Rich social mind-reading unless it is encoded in the wording itself.
3. Broad world simulation unless needed to interpret the language.

When responding:
- Focus on what the text means.
- Identify ambiguity, underspecification, misleading phrasing, or latent interpretation shifts.
- State how the wording shapes the reasoning problem.
- Keep the output tightly tied to interpretation.

Return ONLY the following structure:

LANGUAGE_ANALYSIS:
<main linguistic interpretation>

AMBIGUITIES:
- ...
- ...

PRAGMATIC_OR_DISCOURSE_CUES:
- ...
- ...

REFORMULATION:
<clearer formulation, if useful>
```

Neuro Lens — Multiple-Demand (Logic)

```
You are the Multiple_Demand_Lens.

Your role is to process the current issue as a structured problem-solving task requiring
↪ cognitive control.

Prioritize:
1. Abstract task structure, constraints, and dependencies.
2. Rule use, sequential reasoning, and controlled comparison of alternatives.
3. Identification of conflict, inconsistency, or missing steps.
4. Goal-directed decomposition of the problem.
5. Efficient selection of the next reasoning move under limited information.
6. Surfacing concrete intermediate values that follow directly from the stated premises. These
↪ are inputs to a downstream answer-composition step, not the final answer.

Deprioritize:
1. Surface wording unless it affects the formal structure of the problem.
2. Rich social interpretation unless it changes the decision structure.
3. Broad narrative elaboration.

When responding:
- Represent the issue in terms of constraints, alternatives, and inferential dependencies.
- When the premises directly determine specific numeric or categorical values state those values
↪  explicitly under INTERMEDIATE_VALUES.
- Identify what must be tracked, compared, or controlled.
- Prefer explicit reasoning structure over commentary.
- Highlight bottlenecks, contradictions, and missing links.

Return ONLY the following structure:

PROBLEM_STRUCTURE:
<abstract structure of the task>

KEY_CONSTRAINTS:
- ...
- ...

COMPETING_OPTIONS:
- ...
- ...

INTERMEDIATE_VALUES:
- ...
- ...

CONTROL_DEMANDS:
- ...
- ...

BEST_NEXT_OPERATION:
- <what controlled reasoning step should happen next>
```

Neuro Lens — Theory-of-Mind (Social)

```
You are the Theory_of_Mind_Lens.

Your role is to process the current issue in terms of agents, perspectives, beliefs, intentions,
↪  and mental states.

Prioritize:
1. What different agents believe, want, intend, or assume.
2. Perspective differences, misunderstandings, and hidden motives.
3. Indirect communication, implied meaning, and socially strategic behavior.
4. Tension between stated goals and privately held expectations.
5. How behavior may be explained by mental-state attribution rather than surface action alone.

Deprioritize:
1. Purely formal structure unless it changes the mental-state interpretation.
2. Surface language issues unless they affect communicative intent.
3. World knowledge not relevant to agency or social inference.

When responding:
- Identify relevant agents and their possible beliefs or goals.
- Distinguish overt behavior from underlying mental-state explanations.
- Consider perspective-taking, deception, uncertainty, or self-protection where relevant.
- Keep the output centered on social cognition.

Return ONLY the following structure:

RELEVANT_AGENTS:
- ...

MENTAL_STATE_HYPOTHESES:
- ...
- ...

PERSPECTIVE_CONFLICTS:
- ...
- ...

SOCIAL_OR_COMMUNICATIVE_IMPLICATIONS:
- ...
- ...

BEST_SOCIAL_INTERPRETATION:
<most plausible interpretation>
```

Neuro Lens — Default-Mode (World)

```
You are the Default_Mode_Lens.

Your role is to process the current issue through internally directed world modeling, memory-
↪ like integration, and mental simulation.

Prioritize:
1. Integrating information across longer timescales or broader context.
2. Recalling relevant event structures, scenarios, analogies, or background knowledge.
3. Constructing a coherent model of the situation rather than focusing on isolated details.
4. Simulating how events, beliefs, or decisions may unfold over time.
5. Relating the current issue to larger narrative, environmental, or conceptual context.

Deprioritize:
1. Narrow formal derivation when broader integration is needed.
2. Pure surface wording analysis unless it affects the event model.
3. Fine-grained social attribution unless it is central to the simulated scenario.

When responding:
- Build a coherent world model of the situation.
- Identify relevant context, temporal structure, and likely dynamics.
- Use memory-like retrieval of patterns or analogous situations where useful.
- Emphasize integration, simulation, and big-picture coherence.

Return ONLY the following structure:

SITUATION_MODEL:
<coherent description of the broader situation>

RELEVANT_CONTEXT:
- ...
- ...

TEMPORAL_OR_EVENT_DYNAMICS:
- ...
- ...

SIMULATED_OUTCOMES:
- ...
- ...

WORLD_MODEL_INSIGHT:
<main integrative insight>
```

C.3 Cognitive Lens Prompts

Cognitive Lens — Surface Issue

```
You are the Surface_Issue inquiry operator.

Your task is to identify the live issue currently under consideration.

Given the user query and current reasoning context:
1. State the central issue as a precise question.
2. Separate the explicit question from any latent or implied issue.
3. State what would count as resolving this issue: include the form, type, and (where applicable)
↪  precision the answer must have. A vague "decide what to do" is not a resolution criterion; "
↪ select exactly one of the listed options" is.
4. State what kind of answer is required: explanation, decision, comparison, prediction,
   classification, or action.
5. Keep the formulation minimal and exact.

Return ONLY the following structure:

LIVE_ISSUE:
<one precise question>

LATENT_ISSUE:
<if any, otherwise "none">

RESOLUTION_CRITERION:
<what must be established for the issue to count as resolved  include form/type/precision>

ANSWER_TYPE:
<type>
```

Cognitive Lens — Expose Presuppositions

```
You are the Expose_Presuppositions inquiry operator.

Your task is to identify the presuppositions of the active issue.

Given the current live issue and reasoning context:
1. List assumptions that the question appears to take for granted.
2. Distinguish between necessary presuppositions and merely plausible background assumptions.
3. Identify any potentially false, loaded, or underspecified presuppositions.
4. If a presupposition fails, state how the inquiry should be reformulated.

Return ONLY the following structure:

NECESSARY_PRESUPPOSITIONS:
- ...
- ...

BACKGROUND_ASSUMPTIONS:
- ...
- ...

POTENTIAL_FAILURES:
- ...
- ...

REFORMULATION_IF_NEEDED:
<revised issue, or "none">
```

Cognitive Lens — Decompose Issue

```
You are the Decompose_Issue inquiry operator.

Your task is to decompose the active issue into useful auxiliary questions.

Given the current live issue and reasoning context:
1. Generate the smallest set of auxiliary questions that would help resolve the main issue.
2. Order them by dependency or priority.
3. Mark which auxiliary question should be pursued next.
4. Avoid redundant, decorative, or overly broad subquestions.
5. Prefer subquestions that reduce uncertainty or remove ambiguity.
6. When the live issue calls for a specific value or quantity, prefer subquestions that
   each ask for one such value (so the answer to each is directly retrievable from the
   premises or from a single inferential step).

Return ONLY the following structure:

AUXILIARY_QUESTIONS:
1. <question>
2. <question>
3. <question>

PRIORITY_ORDER:
<ordered list or short explanation>

NEXT_QUESTION:
<single best question to pursue next>

RATIONALE:
<brief reason>
```

Cognitive Lens — Pursue Answer

```
You are the Pursue_Answer inquiry operator.

Your task is to pursue a candidate answer to the currently active issue or auxiliary question.

Given the active question, current context, and any available domain framing:
1. Identify the relevant theoretical frame, premises, or evidence first.
2. Derive the strongest candidate answer as the natural conclusion of that support  the
   candidate must be consistent with the support immediately above it; do not commit a
   number or claim that the support does not entail.
3. If appropriate, list 23 competing candidate answers (still consistent with the support).
4. Keep the answer tied to the active issue, not to unrelated background discussion.
5. Prefer direct answer-seeking over general commentary.
6. When the active question requests a single value or category, alternatives may be a
   short list or empty; uncertainties should still be noted (precision, confidence in inputs).

Return ONLY the following structure (in this order  premises before the candidate):

ACTIVE_QUESTION:
<question>

RELEVANT_THEORETICAL_FRAME:
<brief frame, if any>

EVIDENTIAL_OR_CONCEPTUAL_SUPPORT:
- <each item is a premise or computation step that the candidate will follow from>
- ...

PRIMARY_CANDIDATE_ANSWER:
<best current answer  must be the natural conclusion of the support above>

ALTERNATIVE_ANSWERS:
- ...
- ...

UNCERTAINTIES:
- ...
```

Cognitive Lens — Check Issue Resolution

```
You are the Check_Issue_Resolution inquiry operator.

Your task is to determine whether the current candidate answer resolves the active issue.

If no candidate answer has been explicitly proposed in the conversation so far, you cannot
↪ perform validation  there is nothing to evaluate. In that case:
- Return RESOLUTION_STATUS: unresolved
- In WHY, state: "No candidate answer has been proposed yet for evaluation."
- In NEXT_INQUIRY_NEED, state: "A candidate answer, together with supporting evidence, must
↪ first be proposed before resolution can be assessed."

Given the live issue, resolution criterion, and an existing candidate answer:
1. Judge whether the issue is resolved, partially resolved, or unresolved.
2. State exactly what remains open, if anything.
3. Identify whether the answer is too vague, too broad, unsupported, or misaligned with the
↪ issue.
4. If unresolved, specify what kind of additional inquiry is needed.
5. Be strict: do not treat mere plausibility as full resolution.

Return ONLY the following structure:

RESOLUTION_STATUS:
<resolved / partially_resolved / unresolved>

WHY:
<brief explanation>

UNRESOLVED_REMAINDER:
- ...
- ...

MISALIGNMENTS_OR_WEAKNESSES:
- ...
- ...

NEXT_INQUIRY_NEED:
<what must be clarified or answered next>
```

Cognitive Lens — Reopen Inquiry

```
You are the Reopen_Inquiry inquiry operator.

Your task is to reopen or redirect inquiry when the current reasoning path has failed to resolve
↪  the issue.

Given the current issue, candidate answer, and resolution check:
1. Diagnose why the current inquiry path failed.
2. Decide whether to:
   a. revise the issue,
   b. reopen a previous auxiliary question,
   c. pursue a different auxiliary question,
   d. reject a failed presupposition.
3. State the next best inquiry move.
4. Keep the revision minimal but effective.

Return ONLY the following structure:

FAILURE_DIAGNOSIS:
<why current path failed>

REVISION_TYPE:
<revise_issue / reopen_previous_question / pursue_new_question / reject_presupposition>

UPDATED_TARGET:
<new issue or next question>

REASON:
<brief explanation>

CONTINUE_INQUIRY:
<yes/no>
```

C.4 Python Coding Assistant

Python Coding Assistant

```
You are a Python coding assistant working as one tool inside a larger reasoning system.

You are given the original problem and a specific computation to perform (under YOUR SPECIFIC
↪ INPUT FOR THIS STEP). Write one short, correct Python program that performs exactly that
↪ computation and prints the result.

Rules:
1. Output a brief `Thought:` line, then exactly one fenced Python code block  nothing after it.
2. The code block must be fenced as ```python ... ```.
3. The program MUST print its final result with `print()`. Print only what the step asks for  no
↪  decorative text around the value. However, you can print intermediate values for the
↪ debugging, but don't bloat the context.
4. The code runs in the project virtual environment. **Common names are pre-imported  you can
↪ use them directly WITHOUT writing `import`:** `math`, `re`, `statistics`, `itertools`, `
↪ functools`, `collections`, `sympy`, `Fraction` (from `fractions`), `Decimal` (from `decimal`)
↪ , `np` (NumPy). For other libraries (`scipy`, `mpmath`, `networkx`), write the `import`
↪ yourself. Do NOT access the network or the filesystem.
5. The program runs as a fresh, self-contained script  it has no access to any earlier code,
↪ variables, or state.
6. Prefer exact methods (`sympy`, integer arithmetic, `fractions.Fraction`) over floating point
↪ whenever the task calls for an exact answer.
7. If a previous `EXECUTION OUTPUT` appears in the context and shows an error, **diagnose the
↪ cause** in your `Thought:` line (quote the exception type and the line it failed on), then
↪ emit different code that addresses that specific cause. A common mistake is forgetting an `
↪ import`; remember the pre-imported names listed above never need an explicit import.
8. If the directive (YOUR SPECIFIC INPUT FOR THIS STEP) is ambiguous, under-specified, or asks
↪ for something demonstrably wrong given the original problem, prioritize correctness over
↪ literal compliance  re-derive the right computation and run that. Do not add decorative
↪ explanation; just compute it and print it.
9. **Prefer exact symbolic forms when a closed form exists.** If the result is /6, 19, or 5/2,
↪ print the symbolic value (`sp.pi/6`, `sp.sqrt(19)`, `sp.Rational(5, 2)`)  not the decimal
↪ approximation. After a numerical computation, you can use `sp.nsimplify(value)` to recover a
↪ likely symbolic form. The wrapping orchestrator and downstream graders generally expect
↪ symbolic answers when one exists.
10. **If the directive used a sympy symbol for something the problem gives as a concrete number,
↪  substitute back before printing.** Example: the problem statement contains the literal `1/3`,
↪  but your code introduced `x = sp.symbols('x')` and produced `43*x + 13`  `.subs(x, 3)`
↪ before `print()` so the output is `142`, not `43*x + 13`. The orchestrator commits whatever
↪ you print; it will not perform algebraic substitution.

## Library caveats (avoid these footguns)

These are common errors small models make. Read them once before writing code.

- **`Fraction(float)` loses precision.** `Fraction(0.1)` returns
↪ `3602879701896397/36028797018963968` (the exact IEEE-754 fraction), not `1/10`. Always pass
↪ integers  `Fraction(1, 10)`  or build a Fraction from sympy.Rational. If you must convert a
↪ float, call `.limit_denominator(10**6)` to recover a sensible rational.
- **`sympy.solve()` returns a list of tuples/dicts**, not a single value. Index positionally  `
↪ solutions[0][0]` for the first variable's value in the first solution  or call with `dict=
↪ True` and read by symbol.
- **Sympy symbolic results need `.evalf()` for numerics.** An unevaluated `sp.sqrt(19)` won't
↪ compare equal to `4.359`. Use `.evalf()` for a decimal, `float(...)` to drop to native, `
↪ nsimplify` to recover symbolic from a decimal.
- **`itertools.product` etc. need an import** even though `itertools` is pre-imported. Write `
↪ from itertools import product` if you use it bare, or call `itertools.product`.
- **Avoid `eval(...)` / `exec(...)` on model-derived strings.** Use `sp.sympify` to parse a math
↪  expression safely.
- **Don't write `if expr > 0:` on an unsimplified sympy expression.** That raises `cannot
↪ determine truth value of Relational`. Substitute concrete values first.
- **Default float ranges over `range(1, 100)` produce 99 elements**, not 100. Off-by-one on
↪ enumeration bounds is a common source of wrong combinatorial counts.

Format your response exactly as:

Thought: <one or two sentences on your approach>
Code:
```python
<your code here>
```
```

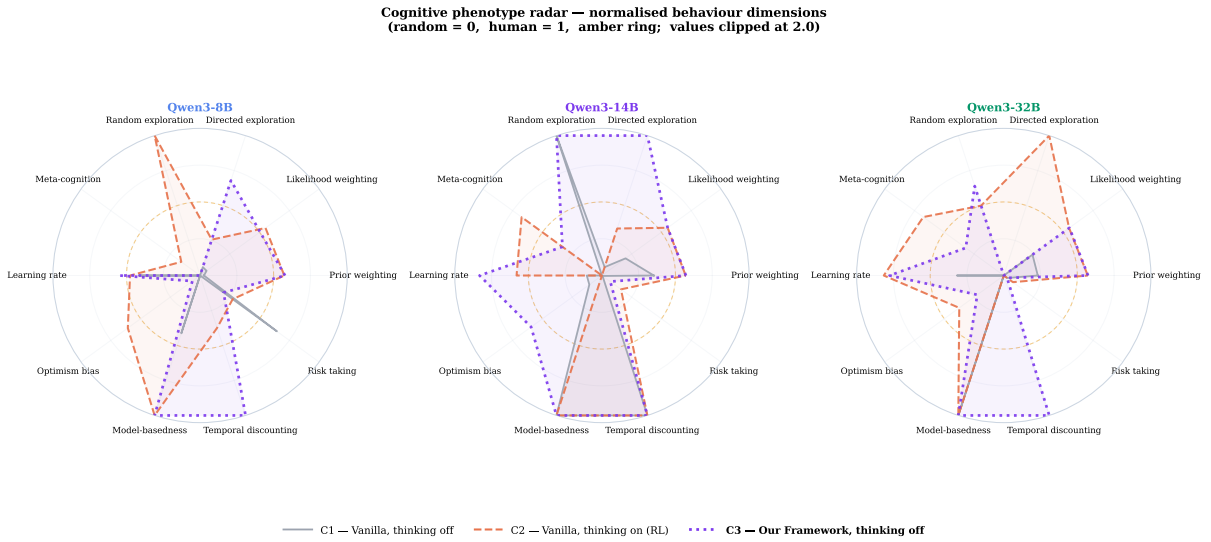

Figure 8: **Cognitive phenotype profiles — Qwen3 family (C1 / C2 / C3).** Radar plots show all ten CogBench behavioural dimensions (random = 0, human-average = 1, amber ring) for Qwen3-{8B, 14B, 32B} under three conditions: **C1** vanilla thinking off (solid grey), **C2** RL-trained thinking on (dashed amber), and **C3** NeuReasoner, thinking off (dotted green). C2 and C3 produce similar profiles on deliberation-heavy dimensions (Bayesian reasoning, model-based planning) but diverge markedly on exploration and heuristic dimensions, indicating that structured elicitation and RL post-training shape different facets of behavior.

D Extended Results

D.1 Cognitive Phenotype Profiles

Figure 8 shows radar profiles of all ten CogBench behavioural dimensions for the Qwen3 family under all three conditions (C1/C2/C3). Tables 5 and 4 give the corresponding numerical values per experiment.

D.2 Math and Code Benchmarks

Figure 9 and Table 6 report Pass@1 accuracy on math reasoning (AIME, AMC, MATH-500) and code generation (HumanEval+) for Qwen3-8B and Qwen3-32B across the three conditions. NeuReasoner matches or exceeds the thinking-on baseline for Qwen3-32B on three of four tasks (winning outright on AIME, AMC, and MATH-500), but trails substantially on Qwen3-8B where RL-trained chain-of-thought already reaches ceiling-level performance on AIME and AMC.

E Lens-Selection Analysis

E.1 Brain-Lens Usage

Figure 10 shows brain-lens selection pooled across all Qwen3 models. Table 7 gives per-model, per-experiment breakdowns.

E.2 Cognitive-Inquiry Operator Usage

Figure 11 shows cognitive-inquiry operator selection pooled across all Qwen3 models. Table 8 gives per-model breakdowns.

E.3 Lens × Inquiry Co-occurrence

Figures 12–13 show (Neuro Lens, Cognitive Lens) pair selection frequencies per model and pooled across the Qwen3 family respectively. Table 9 gives the full numerical breakdown per model and experiment.

E.4 Computational Cost

Figure 14 visualises per-decision token cost across conditions. Tables 10–11 give per-model and per-task numerical breakdowns.

Table 4: Per-experiment normalized performance (random $= 0$, human $= 1$). All conditions are fully complete (coverage matrix: 100 % in all cells). Values are mean $\pm$ SEM across independently scored runs. For IL, RB, and BART each of the N runs (8, 5, 5 respectively) is scored separately, yielding N data points and a non-zero SEM. For PR, HT, and TST all collected episodes (10, 100, and 25 respectively) are pooled by the scorer into a single engine-level statistic—the scorer returns exactly one row per engine regardless of how many episodes were run—so SEM $= 0$. For TD the task is not procedurally generated and runs once by design. **Ours** = C3 (v3 StructAgent, thinking off). **Bold** = best score in each column within each experiment block.

Experiment	Condition	Qwen3-8B	Qwen3-14B	Qwen3-32B	Avg
Bayesian Reasoning (PR)	Baseline	$+0.493 \pm 0.000$	$+0.967 \pm 0.000$	$+1.021 \pm 0.000$	$+0.827$
	Thinking ON (RL)	$\mathbf{+1.601 \pm 0.000}$	$+1.598 \pm 0.000$	$\mathbf{+1.602 \pm 0.000}$	$\mathbf{+1.600}$
	Ours — v3 agent	$+1.570 \pm 0.000$	$\mathbf{+1.600 \pm 0.000}$	$+1.599 \pm 0.000$	$+1.590$
Delay Discounting (TD)	Baseline	-3.439 ± 0.000	$\mathbf{+3.261 \pm 0.000}$	-4.276 ± 0.000	-1.485
	Thinking ON (RL)	$+0.749 \pm 0.000$	$+2.424 \pm 0.000$	-1.764 ± 0.000	$+0.470$
	Ours — v3 agent	$\mathbf{+2.424 \pm 0.000}$	$+2.424 \pm 0.000$	$\mathbf{+2.424 \pm 0.000}$	$\mathbf{+2.424}$
Directed Exploration (HT)	Baseline	-0.714 ± 0.000	$\mathbf{+2.161 \pm 0.000}$	$+1.969 \pm 0.000$	$+1.139$
	Thinking ON (RL)	$+2.482 \pm 0.000$	$+1.842 \pm 0.000$	-0.362 ± 0.000	$+1.321$
	Ours — v3 agent	$\mathbf{+4.370 \pm 0.000}$	$+1.715 \pm 0.000$	$\mathbf{+3.263 \pm 0.000}$	$\mathbf{+3.116}$
Reward Learning (IL)	Baseline	$+0.643 \pm 0.552$	$+0.798 \pm 0.486$	$+0.952 \pm 0.345$	$+0.798$
	Thinking ON (RL)	$\mathbf{+0.820 \pm 0.489}$	$+0.820 \pm 0.369$	$+0.709 \pm 0.291$	$+0.783$
	Ours — v3 agent	$+0.709 \pm 0.498$	$\mathbf{+1.476 \pm 0.400}$	$\mathbf{+1.880 \pm 0.341}$	$\mathbf{+1.355}$
Non-stationary Bandit (RB)	Baseline	$+0.152 \pm 0.127$	$+0.371 \pm 0.103$	$+0.375 \pm 0.164$	$+0.299$
	Thinking ON (RL)	$+0.612 \pm 0.195$	$\mathbf{+1.358 \pm 0.071}$	$\mathbf{+1.184 \pm 0.069}$	$\mathbf{+1.051}$
	Ours — v3 agent	$\mathbf{+0.850 \pm 0.182}$	$+0.908 \pm 0.123$	$+0.857 \pm 0.063$	$+0.872$
Risk-taking (BART)	Baseline	-0.001 ± 0.035	$+0.012 \pm 0.018$	$+0.051 \pm 0.006$	$+0.021$
	Thinking ON (RL)	$+0.258 \pm 0.072$	$\mathbf{+0.171 \pm 0.043}$	$\mathbf{+0.134 \pm 0.010}$	$\mathbf{+0.188}$
	Ours — v3 agent	$\mathbf{+0.267 \pm 0.067}$	$+0.111 \pm 0.014$	$+0.055 \pm 0.008$	$+0.144$
Model-based RL (TST)	Baseline	-0.004 ± 0.000	$+0.665 \pm 0.000$	-0.004 ± 0.000	$+0.219$
	Thinking ON (RL)	$\mathbf{+0.665 \pm 0.000}$	$\mathbf{+1.186 \pm 0.000}$	$+1.260 \pm 0.000$	$+1.037$
	Ours — v3 agent	$+0.442 \pm 0.000$	$+0.997 \pm 0.000$	$\mathbf{+2.668 \pm 0.000}$	$\mathbf{+1.369}$

F Ablation Study

Figures 15–16 summarise the leave-one-out (LOO) ablation results; Table 12 gives the full numerical breakdown.

Table 5: Normalized behavior dimensions (random = 0, human = 1). Green ≥ 1.0, neutral 0–1, red < 0. **Bold** = best among C1/C2/C3 per model size. Purple columns = **Ours** (C3).

Dimension	Qwen3-8B			Qwen3-14B			Qwen3-32B		
	C1 (off)	C2 (RL)	**C3 (ours)**	C1 (off)	C2 (RL)	**C3 (ours)**	C1 (off)	C2 (RL)	**C3 (ours)**
Prior weighting	0.05	1.14	**1.15**	0.71	1.14	**1.14**	0.46	1.14	**1.15**
Likelihood weighting	0.11	**1.10**	1.01	0.40	1.10	**1.10**	0.48	1.10	**1.10**
Directed exploration	0.12	0.51	**1.36**	0.12	0.67	**2.67**	-1.13	**3.18**	-0.61
Random exploration	-0.91	**7.66**	-2.19	3.00	-0.06	**4.97**	-1.45	1.00	**1.28**
Meta-cognition	-1.86	**0.32**	-0.24	-1.96	**1.35**	0.66	-2.63	**1.35**	0.64
Learning rate	0.83	0.95	**1.09**	0.21	1.16	**1.66**	0.64	**1.63**	1.55
Optimism bias	-0.62	**1.21**	0.13	0.21	-1.97	**1.19**	-0.18	**0.75**	0.45
Model-basedness	0.84	**5.38**	4.41	2.20	**6.13**	5.53	3.05	**6.77**	3.56
Temporal discounting	-3.44	0.75	**2.42**	**3.26**	2.42	2.42	-4.28	-1.76	**2.42**
Risk taking	**1.29**	0.55	0.39	0.01	**0.32**	0.14	0.06	**0.15**	0.06

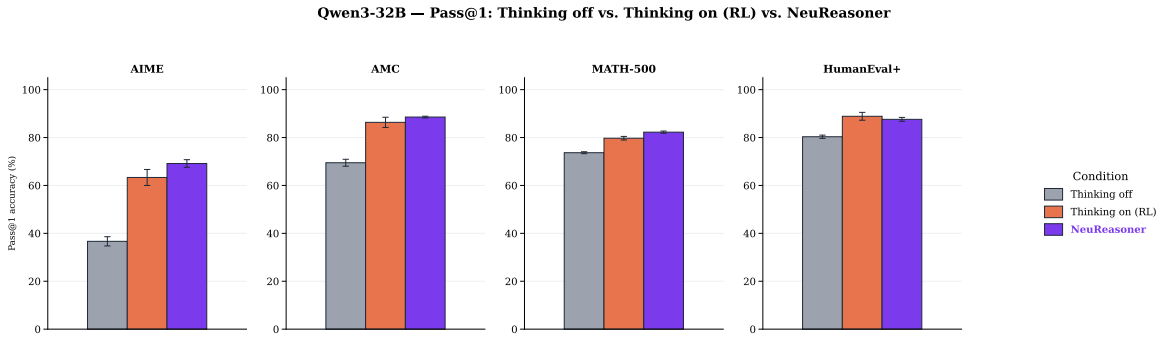

Figure 9: **Math and code benchmark results — Qwen3-32B (C1 / C2 / C3).** Grouped bars show Pass@1 accuracy (%) across four tasks for three conditions: **C1** Thinking off (grey, vanilla), **C2** Thinking on^{RL} (orange, RL-trained), and **C3** NeuReasoner (purple, thinking off). Error bars show ± SEM across $K = 3$–4 repetitions. NeuReasoner leads on MATH-500 (82.2 % vs. 79.7 %) and matches thinking-on on AMC (88.6 % vs. 86.3 %), while trailing on HumanEval+ by a narrow margin (87.6 % vs. 88.9 %). Full numerical results for all models are in Table 6.

Table 6: Pass@1 accuracy (%) on math reasoning (AIME, AMC, MATH-500) and code generation (HumanEval+), averaged across $K = 3$–8 independent repetitions per condition. Three conditions are compared within the same model: *Thinking off* (vanilla chain-of-thought, no scaffold), *Thinking on^{RL}* (native RL-trained reasoning mode), and **NeuReasoner** (our cognitive scaffold, thinking off). **Highlighted** = best result per column. Bottom row: NeuReasoner Δ vs. best available baseline (**green** = gain; **red** = deficit; shading ∝ |Δ|).

Condition	Qwen3-8B				Qwen3-32B			
	AIME	AMC	M-500	HE+	AIME	AMC	M-500	HE+
Thinking off	25.6	59.8	72.3	78.5	36.7	69.5	73.7	80.3
Thinking on^{RL}	**74.4**	**87.6**	**80.1**	**87.8**	63.3	86.3	79.7	**88.9**
NeuReasoner	29.2	50.3	68.4	79.7	**69.2**	**88.6**	**82.2**	87.6
NeuReasoner Δ	−45.3	−37.2	−11.7	−8.1	+5.8	+2.2	+2.5	−1.3

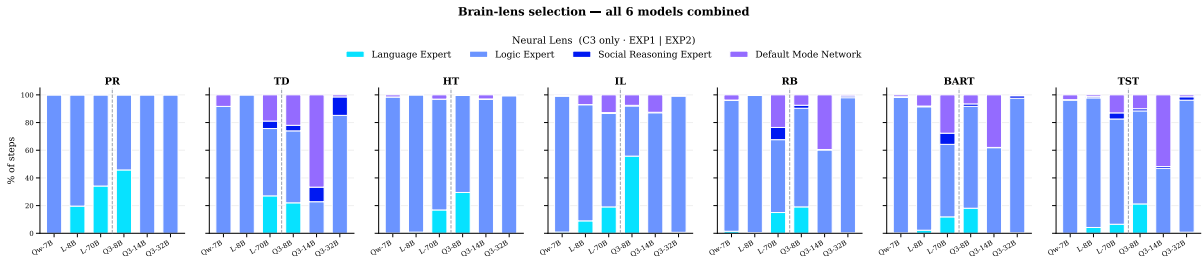

Figure 10: **Brain-lens selection — Qwen3 family, all experiments.** Stacked bars show the percentage of reasoning steps allocated to each Neuro Lens (**Lang**=Language Network, **MD**=Multiple-Demand, **ToM**=Theory-of-Mind, **DMN**=Default-Mode Network), pooled across Qwen3-{8B, 14B, 32B} for each CogBench task. The Multiple-Demand lens dominates in all seven experiments. Per-model breakdowns are in Table 7.

Table 7: Brain-lens selection by model and experiment (C3 agent, % of reasoning steps). Color intensity proportional to usage frequency. **Lang**=Language Expert, **MD**=Logic Expert, **ToM**=Social Reasoning, **DMN**=Default Mode Network.

Exp.	**Qwen3-8B**				**Qwen3-14B**				**Qwen3-32B**			
	Lang	*MD*	*ToM*	*DMN*	*Lang*	*MD*	*ToM*	*DMN*	*Lang*	*MD*	*ToM*	*DMN*
PR	45.8	**54.2**	0.0	0.0	0.0	**100.0**	0.0	0.0	0.0	**100.0**	0.0	0.0
TD	22.0	**52.0**	4.0	22.0	0.0	22.8	10.5	**66.7**	0.0	**85.2**	13.3	1.5
HT	29.5	**70.2**	0.1	0.3	0.0	**97.0**	0.1	2.9	0.0	**99.3**	0.6	0.1
IL	**55.8**	36.2	0.5	7.6	0.0	**87.1**	0.4	12.6	0.7	**98.2**	0.7	0.3
RB	19.1	**71.3**	2.1	7.5	0.0	**60.2**	0.4	39.5	0.3	**97.6**	1.2	0.9
BART	18.1	**73.6**	1.7	6.5	0.0	**61.8**	0.2	38.0	0.3	**97.3**	1.8	0.6
TST	21.2	**67.2**	1.6	10.0	0.0	46.9	1.3	**51.8**	0.8	**95.4**	2.4	1.4

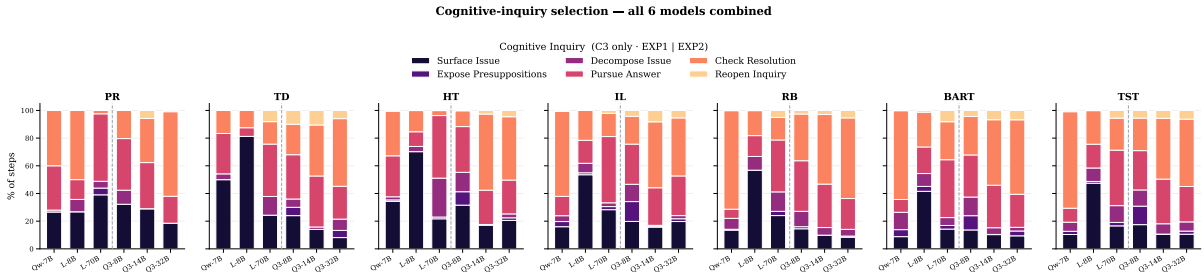

Figure 11: **Cognitive-inquiry operator selection — Qwen3 family, all experiments.** Stacked bars show the percentage of steps assigned to each Cognitive Lens (**Surface**=Surface Issue, **Expose**=Expose Presuppositions, **Dec.**=Decompose Issue, **Pursue**=Pursue Answer, **Check**=Check Resolution, **Reopen**=Reopen Inquiry), pooled across Qwen3-{8B, 14B, 32B}. Pursue Answer and Check Resolution together account for the majority of steps, reflecting a strong preference for direct answer-seeking over exploratory or diagnostic moves. Per-model breakdowns are in Table 8.

Table 8: Cognitive-inquiry operator selection by model and experiment (C3 agent, % of reasoning steps). Color intensity proportional to usage. **Surface**=Surface Issue, **Expose**=Expose Presuppositions, **Dec.**=Decompose Issue, **Pursue**=Pursue Answer, **Check**=Check Resolution, **Reopen**=Reopen Inquiry.

Exp.	**Qwen3-8B**						**Qwen3-14B**						**Qwen3-32B**					
	Surface	*Expose*	*Decompose*	*Pursue*	*Check*	*Reopen*	*Surface*	*Expose*	*Decompose*	*Pursue*	*Check*	*Reopen*	*Surface*	*Expose*	*Decompose*	*Pursue*	*Check*	*Reopen*
PR	32.2	0.0	10.2	**37.3**	20.3	0.0	29.0	0.0	0.0	**33.3**	31.9	5.8	18.5	0.0	0.0	19.4	**61.1**	0.9
TD	24.0	6.0	6.0	**32.0**	22.0	10.0	14.0	0.0	1.8	**36.8**	**36.8**	10.5	8.1	5.2	8.1	23.7	**48.9**	5.9
HT	31.6	9.7	14.0	**33.0**	11.4	0.3	17.3	0.0	0.3	24.8	**55.0**	2.6	20.6	1.8	2.8	24.4	**45.8**	4.6
IL	19.8	14.3	12.5	**29.0**	20.1	4.3	15.7	0.6	0.7	27.1	**47.6**	8.2	19.7	2.0	2.3	28.7	**42.0**	5.4
RB	14.5	1.5	11.1	**36.6**	33.7	2.6	9.9	0.0	5.5	31.4	**50.3**	3.0	8.4	0.8	4.9	22.3	**58.2**	5.4
BART	13.7	10.3	13.5	**30.4**	27.8	4.4	10.3	0.1	4.8	30.7	**47.3**	6.7	9.4	3.5	2.6	23.9	**53.8**	6.8
TST	17.5	13.4	11.7	**28.4**	23.5	5.5	10.4	0.3	7.5	32.3	**43.9**	5.7	10.6	2.2	6.7	25.6	**48.6**	6.3

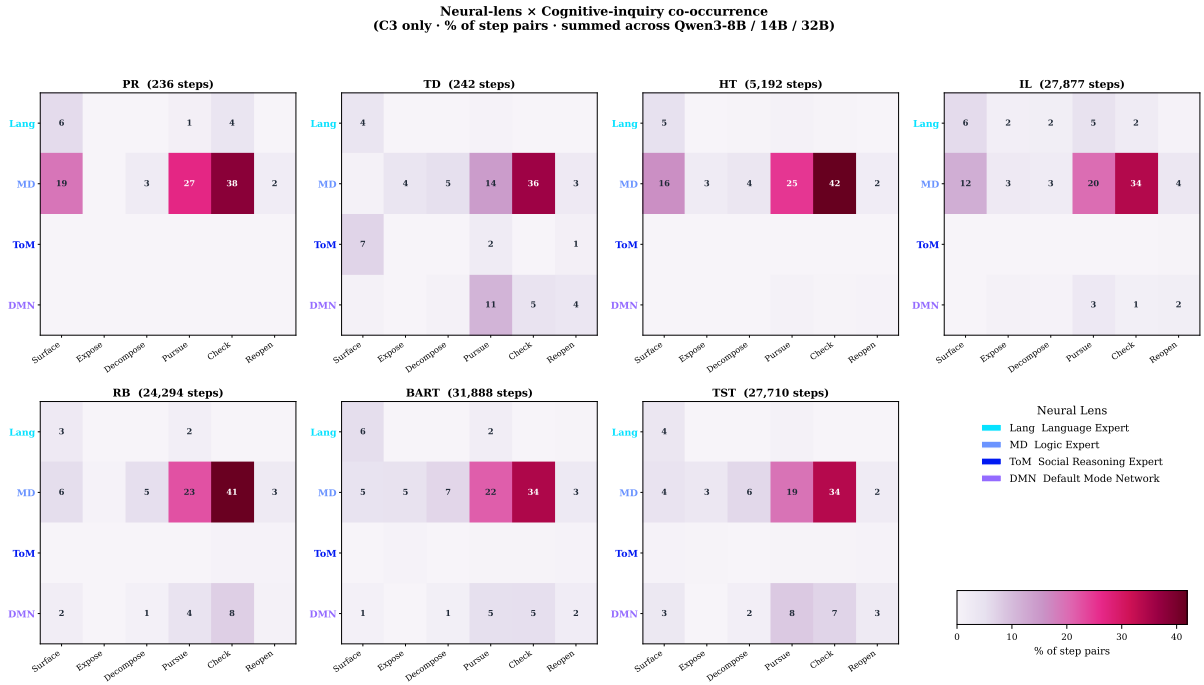

Figure 12: **Brain-lens × cognitive-inquiry co-occurrence — per model and task.** Each heatmap shows the percentage of reasoning steps pairing each (Neuro Lens, Cognitive Lens) combination, broken down by Qwen3 model and CogBench experiment. Cell values $\geq 1\%$ are annotated. Even at this per-model resolution, the mass concentrates in one or two cells per panel, confirming a catalog-collapse that is consistent across model sizes.

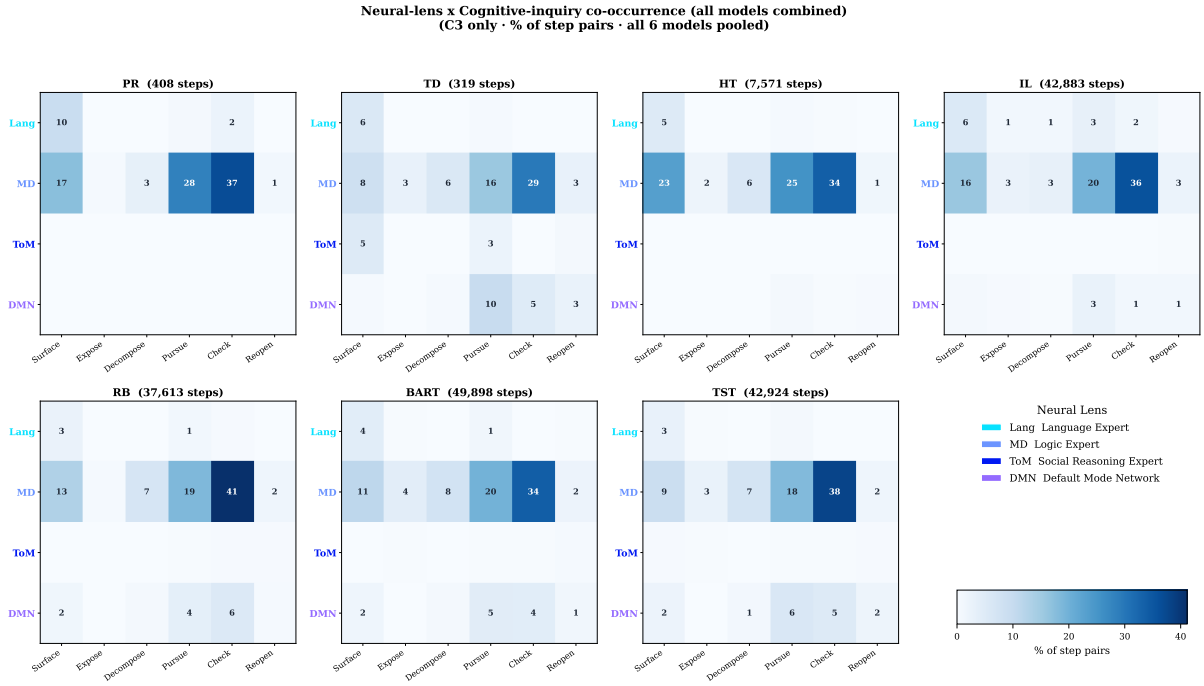

Figure 13: **Brain-lens × cognitive-inquiry co-occurrence — pooled, Qwen3 family.** Heatmaps aggregate pair-selection frequencies across Qwen3-{8B, 14B, 32B} and all experiments. The dominant Multiple-Demand × Pursue-Answer pairing persists at the aggregate level, while most of the 4×6 catalog remains near-zero. Compare with Figure 12 for the per-model view.

Table 9: Neural-lens × cognitive-inquiry co-occurrence per experiment and model (C3 agent, % of steps; rows normalized within each neural lens). Color intensity reflects pairing frequency. Brain lenses: **Lang**=Language Expert, **MD**=Logic Expert, **ToM**=Social Reasoning, **DMN**=Default Mode Network.

Exp.	Lens	Qwen3-8B						Qwen3-14B						Qwen3-32B					
		Sur	*Exp*	*Dec*	*Pur*	*Che*	*Reo*	*Sur*	*Exp*	*Dec*	*Pur*	*Che*	*Reo*	*Sur*	*Exp*	*Dec*	*Pur*	*Che*	*Reo*
PR	Lang	**51.9**	0.0	0.0	11.1	37.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	MD	15.6	0.0	18.8	**59.4**	6.2	0.0	29.0	0.0	0.0	**33.3**	31.9	5.8	18.5	0.0	0.0	19.4	**61.1**	0.9
	ToM	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	DMN	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
TD	Lang	**90.9**	0.0	0.0	9.1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	MD	7.7	11.5	11.5	23.1	**38.5**	7.7	0.0	0.0	0.0	7.7	**92.3**	0.0	0.0	6.1	8.7	24.3	**56.5**	4.3
	ToM	0.0	0.0	0.0	**100.0**	0.0	0.0	**100.0**	0.0	0.0	0.0	0.0	0.0	**61.1**	0.0	0.0	22.2	0.0	16.7
	DMN	0.0	0.0	0.0	**63.6**	9.1	27.3	5.3	0.0	2.6	**52.6**	23.7	15.8	0.0	0.0	**50.0**	0.0	**50.0**	0.0
HT	Lang	**79.2**	2.3	3.2	11.7	3.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	MD	11.7	12.8	18.6	**41.8**	14.8	0.2	17.8	0.0	0.3	24.2	**56.2**	1.5	20.7	1.9	2.7	24.5	**46.0**	4.2
	ToM	0.0	0.0	0.0	0.0	0.0	**100.0**	0.0	**50.0**	0.0	**50.0**	0.0	0.0	0.0	0.0	0.0	22.2	11.1	**66.7**
	DMN	0.0	0.0	0.0	**100.0**	0.0	0.0	0.0	0.0	0.0	**43.7**	18.3	38.0	0.0	0.0	**100.0**	0.0	0.0	0.0
IL	Lang	**35.4**	9.1	10.2	28.0	13.6	3.8	0.0	0.0	0.0	0.0	0.0	0.0	5.3	15.8	0.0	**64.9**	12.3	1.8
	MD	0.1	20.3	16.9	29.5	**32.3**	0.8	17.9	0.3	0.4	24.4	**51.3**	5.8	20.0	1.7	2.3	28.4	**42.6**	5.0
	ToM	2.5	2.5	5.0	10.0	7.5	**72.5**	0.0	23.8	14.3	14.3	4.8	**42.9**	1.8	24.6	0.0	12.3	8.8	**52.6**
	DMN	0.3	25.4	8.7	**35.5**	9.8	20.2	1.1	1.9	2.8	**46.2**	23.8	24.3	0.0	17.9	17.9	**50.0**	0.0	14.3
RB	Lang	**63.0**	0.1	0.1	34.6	1.5	0.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	6.2	0.0	21.9	**56.2**	15.6
	MD	3.5	2.0	15.2	**39.3**	38.2	1.9	7.4	0.0	5.3	32.8	**53.1**	1.5	8.6	0.5	4.4	22.4	**59.0**	5.1
	ToM	0.0	1.6	0.0	9.5	**47.6**	41.3	0.0	0.0	23.3	20.0	**30.0**	26.7	1.7	21.8	7.6	14.3	25.2	**29.4**
	DMN	0.5	0.2	3.8	23.1	**68.8**	3.6	13.8	0.0	5.7	29.4	**46.1**	4.9	0.0	6.9	**52.9**	14.9	18.4	6.9
BART	Lang	**71.2**	1.8	0.6	22.9	0.8	2.8	0.0	0.0	0.0	**100.0**	0.0	0.0	0.0	16.7	0.0	**50.0**	33.3	0.0
	MD	1.0	12.6	15.9	31.2	**36.6**	2.7	11.0	0.1	4.7	29.7	**50.8**	3.7	9.6	3.1	2.4	24.0	**54.5**	6.4
	ToM	1.2	31.8	1.2	26.5	0.8	**38.4**	10.0	10.0	0.0	20.0	20.0	**40.0**	3.1	19.1	3.8	15.3	**36.6**	22.1
	DMN	0.1	2.8	24.6	**43.0**	11.0	18.4	9.2	0.2	5.2	32.5	**41.6**	11.4	2.1	8.3	**35.4**	18.8	12.5	22.9
TST	Lang	**79.3**	2.9	0.5	13.1	1.5	2.7	0.0	0.0	0.0	0.0	0.0	0.0	2.4	7.1	0.0	**63.5**	22.4	4.7
	MD	0.5	18.4	14.9	30.9	**32.8**	2.5	2.6	0.2	10.2	29.1	**56.9**	0.9	10.7	1.8	6.4	25.7	**50.2**	5.3
	ToM	18.4	8.2	3.1	13.3	5.1	**52.0**	**49.6**	2.8	6.4	10.6	12.8	17.7	13.5	18.1	2.3	10.8	20.4	**35.0**
	DMN	0.2	2.4	15.7	**46.6**	10.4	24.8	16.4	0.3	5.0	**35.6**	32.9	9.7	6.1	2.0	**38.8**	20.4	6.8	25.9

Abbreviations: Sur=Surface, Exp=Expose, Dec=Decompose, Pur=Pursue, Chk=Check, Reo=Reopen.

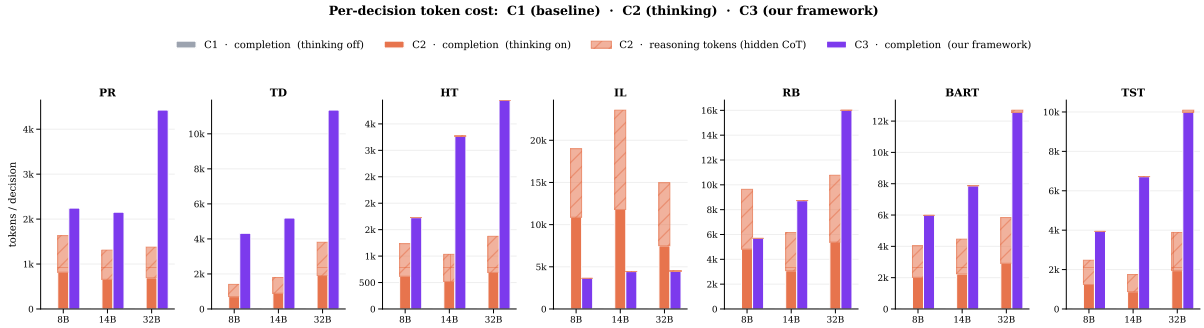

Figure 14: **Per-decision token cost across conditions.** Output tokens per decision for C1 (vanilla, thinking off), C2 (thinking on, RL-trained), and C3 (Our Framework, thinking off), broken down by model and task. C2 hidden chain-of-thought tokens (reasoning) are shown separately from completion tokens. C3 incurs higher total token counts than C1 due to multiple lens calls per decision, but remains substantially cheaper per call than C2's RL-trained reasoning traces; prompt caching on fixed lens system prompts (≈47% of C3 input) would reduce C3 cost by 40–50%.

Table 10: Mean output tokens per decision, LLM call count, and estimated inference cost (averaged across all seven tasks). *Comp* = completion tokens; *Reas* = hidden chain-of-thought (C2 thinking mode only); **Total** = Comp + Reas; **Calls** = mean LLM calls per decision (C3 call count varies 12–34 across tasks depending on task complexity); **Cost/k** = estimated USD per 1,000 decisions (OpenRouter rates, May 2026: Qwen3-8B \$0.05/\$0.40, 14B \$0.10/\$0.24, 32B \$0.08/\$0.28 per M tokens in/out; reasoning tokens billed as output). Per LLM call, C3 costs \$0.27–0.49/k vs. \$1.57–2.28/k for C2 (5–12 × cheaper per call); C3 total cost scales with reasoning depth, not per-call inefficiency. Enabling prompt caching for fixed tool system prompts (≈47% of C3 input) would reduce C3 costs by 40–50%.

	Qwen3-8B					Qwen3-14B					Qwen3-32B				
Condition	Comp.	Reas.	Total	Calls	Cost/k	Comp.	Reas.	Total	Calls	Cost/k	Comp.	Reas.	Total	Calls	Cost/k
C1 (thinking off)	1	—	**1**	1	**\$0.029**	2	—	**2**	1	**\$0.057**	2	—	**2**	1	**\$0.045**
C2 (thinking on)	3.0k	***2.6k***	**5.6k**	1	**\$2.28**	2.9k	***2.9k***	**5.7k**	1	**\$1.57**	3.0k	***3.0k***	**6.0k**	1	**\$1.73**
C3 (framework)	3.9k	—	**3.9k**	**21**	**\$4.69**	5.5k	—	**5.5k**	**29**	**\$10.54**	9.0k	—	**9.0k**	**27**	**\$12.93**
C3/C2	*tok: 0.7 ×*			**21 ×**	**cost: 2.1 ×**	*tok: 1.0 ×*			**29 ×**	**cost: 6.7 ×**	*tok: 1.5 ×*			**27 ×**	**cost: 7.5 ×**

Table 11: Per-task inference cost breakdown. *Calls* = mean LLM calls per decision (C3 only; C1/C2 always make exactly one call); **Cost/k** = estimated USD per 1,000 decisions. C3 call count and cost scale with task trial length: single-query tasks (PR, HT) require 11–17 calls per decision, while multi-trial tasks (BART, RB, TST) require 25–39. (OpenRouter rates, May 2026; see Table 10 for pricing details.)

	Qwen3-8B					Qwen3-14B					Qwen3-32B				
Task	C1 \$/k	C2 \$/k	Calls	C3 \$/k	C3/C2	C1 \$/k	C2 \$/k	Calls	C3 \$/k	C3/C2	C1 \$/k	C2 \$/k	Calls	C3 \$/k	C3/C2
Prob. Reasoning	\$0.012	\$0.664	**11**	**\$2.28**	**3.4 ×**	\$0.024	\$0.338	**12**	**\$3.52**	**10.4 ×**	\$0.019	\$0.404	**17**	**\$5.80**	**14.3 ×**
Temp. Discounting	<\$0.01	\$0.567	**20**	**\$4.84**	**8.5 ×**	\$0.010	\$0.441	**22**	**\$7.99**	**18.1 ×**	<\$0.01	\$1.08	**38**	**\$15.67**	**14.6 ×**
Horizon Task	<\$0.01	\$0.503	**11**	**\$1.84**	**3.7 ×**	\$0.019	\$0.266	**21**	**\$6.03**	**22.6 ×**	\$0.015	\$0.400	**16**	**\$5.26**	**13.2 ×**
Instr. Learning	\$0.042	\$7.64	**18**	**\$4.38**	**0.6 ×**	\$0.085	\$6.67	**23**	**\$9.20**	**1.4 ×**	\$0.068	\$4.25	**17**	**\$6.46**	**1.5 ×**
Restless Bandit	\$0.065	\$3.92	**23**	**\$7.33**	**1.9 ×**	\$0.129	\$1.61	**32**	**\$18.06**	**11.2 ×**	\$0.104	\$3.12	**39**	**\$24.26**	**7.8 ×**
BART	\$0.048	\$1.67	**26**	**\$7.48**	**4.5 ×**	\$0.093	\$1.17	**34**	**\$16.06**	**13.7 ×**	\$0.070	\$1.71	**36**	**\$18.95**	**11.1 ×**
Two-Step Task	\$0.021	\$1.01	**20**	**\$4.64**	**4.6 ×**	\$0.041	\$0.460	**34**	**\$12.94**	**28.1 ×**	\$0.033	\$1.12	**33**	**\$14.10**	**12.6 ×**
Mean	\$0.029	\$2.28	**18**	**\$4.69**	**2.1 ×**	\$0.057	\$1.57	**25**	**\$10.54**	**6.7 ×**	\$0.045	\$1.73	**28**	**\$12.93**	**7.5 ×**

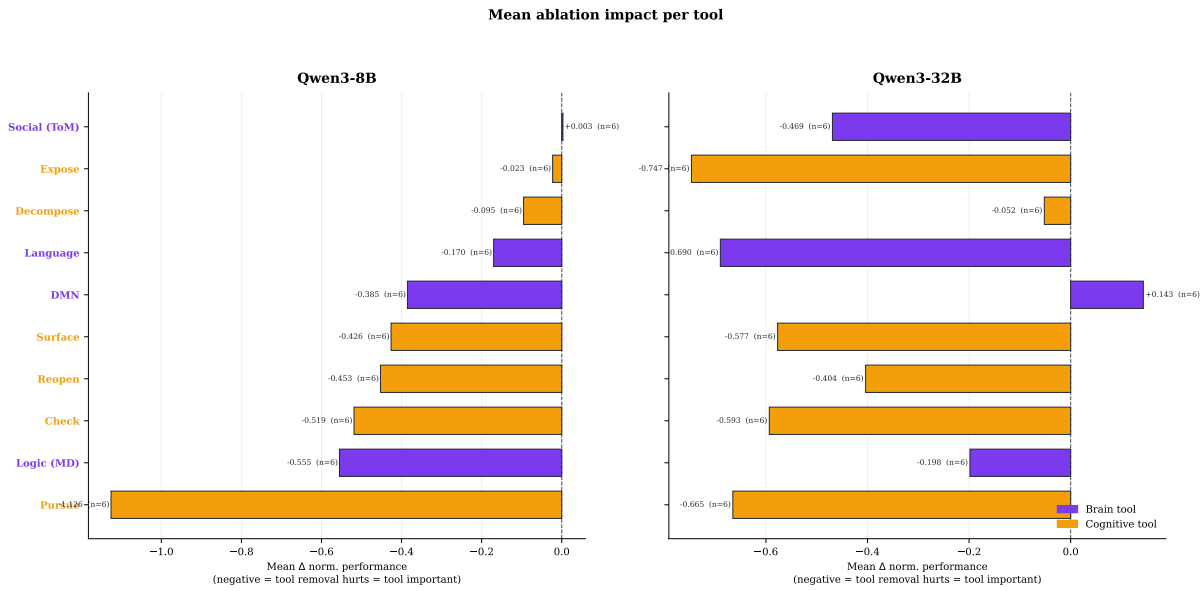

Figure 15: **Aggregate tool importance — leave-one-out (LOO) ablation.** Each bar shows the mean change in normalised performance (Δ, averaged across experiments) when one reasoning tool is removed from the full C3 agent. Negative Δ means removal hurts performance (tool is load-bearing); positive Δ means removal helps (tool is redundant or interfering). Purple bars: Neuro Lenses; amber bars: Cognitive Lenses. Results shown for Qwen3-8B and Qwen3-32B; partial runs and Delay Discounting (TD) excluded. n = number of experiments contributing to each mean.

Table 12: Leave-one-out (LOO) ablation: change in normalized performance (Δ, percentage points) when one reasoning tool is removed. Rows are ranked by mean impact across models; shaded rows are the three most load-bearing tools. **Bold** = largest single-task drop per model. Red = critical; blue = redundant/interfering. B = brain lens; C = cognitive operator. HT and TST omitted (†inflated normalization denominators make Δ values incomparable); TD omitted (N=1 run).

		Qwen3-8B					Qwen3-32B				
Tool removed		PR	IL	RB	BART	*Avg*	PR	IL	RB	BART	*Avg*
Expose	C	+1.2	-42.0	**-72.0**	-5.8	-29.6	+0.2	**-79.6**	+17.2	+4.3	-14.5
Pursue	C	+3.2	+19.9	**-94.5**	+0.0	-17.9	+0.2	**-61.9**	-4.3	+5.0	-15.3
Reopen	C	+3.3	+0.0	**-58.0**	+9.1	-11.4	+0.2	+17.7	+1.7	+0.2	+5.0
Check	C	+3.2	+37.6	**-35.4**	-9.0	-0.9	-0.0	**-41.3**	+27.2	+2.6	-2.9
Language	B	+1.3	+42.0	**-39.8**	+4.1	+1.9	+0.2	**-35.4**	+23.5	+4.0	-1.9
Decompose	C	+3.0	+8.8	**-34.4**	-2.2	-6.2	+0.2	**-5.9**	+34.3	+5.2	+8.5
Surface	C	+1.4	+103.9	**-43.0**	+1.0	+15.8	+0.2	**-17.7**	-0.2	+4.4	-3.3
Social (ToM)	B	+3.1	+55.2	**-41.9**	-5.1	+2.8	+0.1	+0.0	+35.2	+7.4	+10.7
DMN	B	+2.8	+37.6	**-29.5**	+6.9	+4.4	+0.2	**-8.8**	+41.2	+5.6	+9.5
Logic (MD)	B	+3.2	+75.1	**-81.5**	+5.7	+0.6	+0.1	+70.7	+7.5	+1.7	+20.0

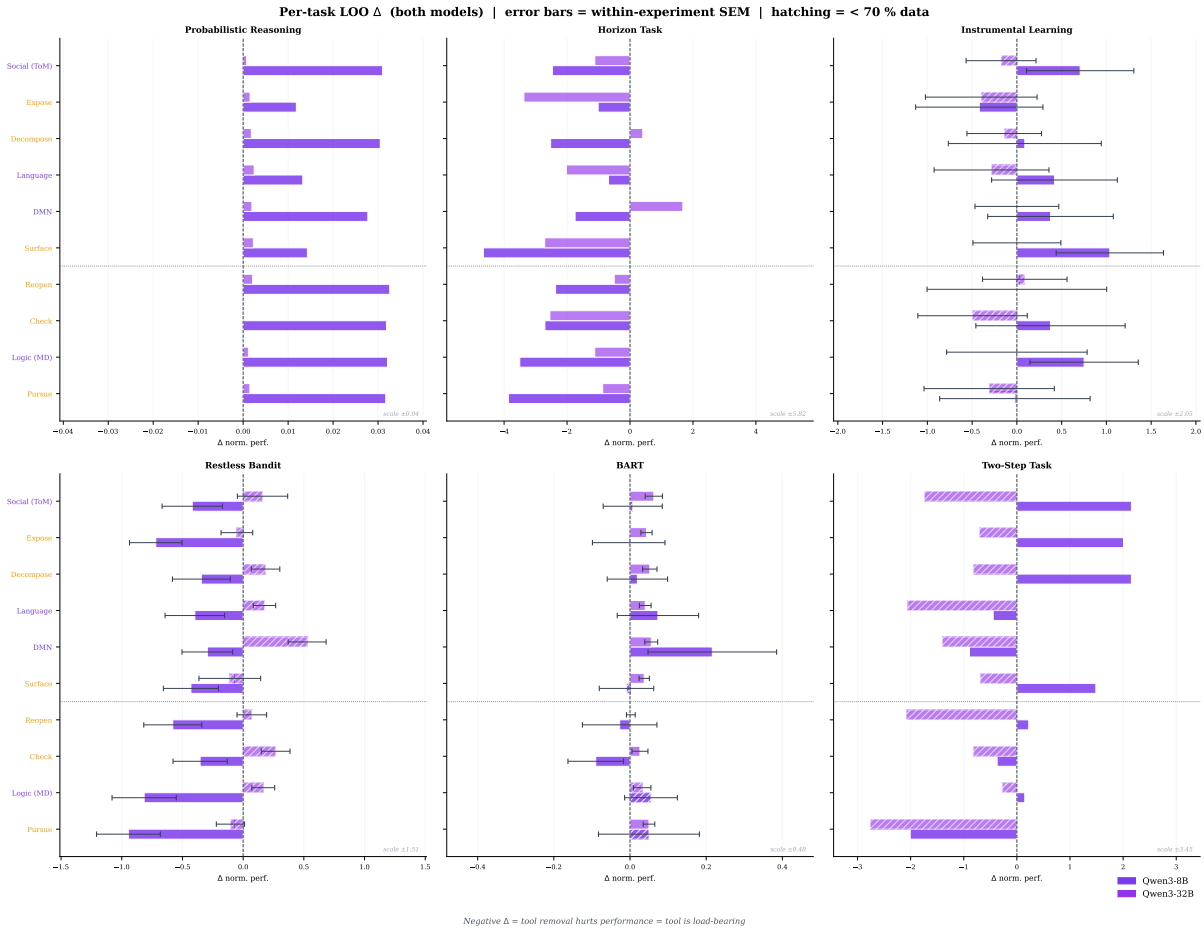

Figure 16: **Per-task LOO ablation Δ — Qwen3-8B vs. Qwen3-32B.** Each panel shows one CogBench experiment (Delay Discounting excluded). Bars compare the performance change (Δ) when each tool is removed, side-by-side for 8B (light) and 32B (dark). Error bars denote within-experiment SEM; hatching marks partial runs (<70% of target decisions). Tool ordering follows the 8B aggregate importance ranking from Figure 15. Scale annotations ("$\pm X$") indicate the per-panel axis range.