Reasoning Like Humans

Yue Liang [1] **Zhenyao Ma** [2]

Abstract

While recent reasoning models have achieved increasingly strong task-solving performance on ARC benchmarks, final correctness alone does not show that they reason in a human-like way. In our preliminary ConceptARC evaluations, current high-performing models remain much less reliable at intermediate concept-level reasoning. This gap motivates a process-level view of ARC evaluation: models should be assessed not only by whether they produce the correct output grid, but also by how they arrive at the answer. We formalize human-like ARC reasoning as a staged process of representation formation, problem-space selection, and heuristic search. Based on this view, we propose a Cognitive Reasoning Model composed of an Inductive Encoder, a Decision Decoder, and a Heuristic Searcher. By introducing a task-level concept prior before rule search, the model offers a concrete way to study structured reasoning in ARC beyond final-answer accuracy.

1. Introduction

The Abstraction and Reasoning Corpus (ARC) was designed to test broad generalization, abstraction, and rule induction from very few demonstrations (Chollet, 2019). In the standard ARC task, a system observes several input-output demonstration pairs and must infer the hidden transformation that maps a new input grid to the correct output.

This setting is important, but it leaves a central question unanswered: are current models solving ARC tasks through structured reasoning, or are they merely producing correct answers through opaque pattern matching, memorized priors, or brute-force search? ConceptARC provides a way to examine this question. By organizing ARC tasks into concept families, it supports evaluation of both output accuracy and task-level concept identification (Moskvichev et al., 2023). Additional background on ARC-AGI and recent reasoning approaches is provided in Appendix A.

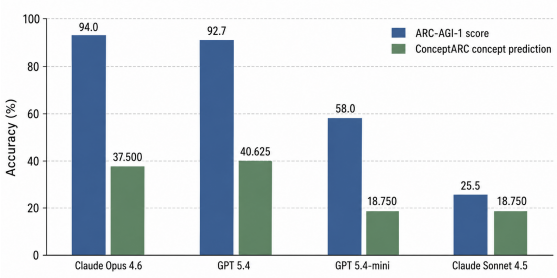

Figure 1. Comparison of LLMs on ARC-AGI-1 scores reported on the ARC Prize Leaderboard and ConceptARC task-level concept prediction accuracy.

Figure 1 illustrates this concern. Selected large language models achieve strong reported performance on ARC-AGI-1, yet their task-level ConceptARC concept prediction accuracy remains much lower under our evaluation. This gap suggests that final-answer accuracy alone may overestimate a model's ability to identify the relevant reasoning space.

We argue that final output accuracy is necessary but insufficient for evaluating human-like abstraction. Human ARC solving appears to involve intermediate structure: first constructing a relational representation of the demonstrations, then selecting a plausible problem space, and finally searching for a rule within that space. A human solver is therefore unlikely to search uniformly over all possible grid programs.

In this paper, we make three contributions. First, we introduce a cognitive decomposition of ARC task solving into representation formation, problem-space selection, and heuristic search. Second, we propose a Cognitive Reasoning Model aligned with this decomposition: an inductive encoder, a decision decoder, and a heuristic searcher. Third, we present a series of preliminary experiments showing that a high performance score on ARC-AGI can coexist with much weaker task-level concept prediction, motivating evaluation of structured abstraction beyond final accuracy.

2. A Cognitive View of ARC Reasoning

We view ARC solving as a staged cognitive process rather than a single input-output mapping. The first stage is *rep-*

[1]University of Tübingen, Tübingen, Germany [2]Xiamen University, Xiamen, China. Correspondence to: Yue Liang <yue.liang@student.uni-tuebingen.de>.

Accepted at the ICML 2026 Workshop on Failure Modes in Agentic AI (FAGEN).

resentation formation. Humans compare multiple demonstrations, extract common relational structure, and represent task-relevant properties such as objects, positions, and spatial relations. This stage is analogical: the solver must align different examples and infer what remains invariant between them (Gentner, 1983).

The second stage is *problem-space selection*. Humans do not usually enumerate all possible programs from the beginning. Instead, they first decide what kind of transformation is likely to explain the demonstrations. This problem-space decision strongly constrains the set of candidate operations that should be considered. In cognitive terms, this resembles constructing the space of operators and states before the search begins (Newell & Simon, 1972).

The third stage is *heuristic search*. After selecting a plausible problem space, humans generate candidate rules, test them against the demonstrations, revise inconsistent hypotheses, and apply the best rule to the query input. This process is not an exhaustive search over all possible transformations, but a guided generate-test search. The quality of reasoning therefore depends not only on final output correctness, but also on whether the model searches in the right conceptual space.

This decomposition motivates our main critique of current reasoning systems. Some models may solve ARC tasks successfully, but they can remain unstable in the middle stage: selecting the correct reasoning space. A model that solves a task without exposing this intermediate decision is difficult to interpret, debug, or compare with human reasoning.

3. Method

We propose a Cognitive Reasoning Model with three components: an Inductive Encoder, a Decision Decoder, and a Heuristic Searcher (Figure 2). Demonstrations are first encoded into a task representation, the decision decoder predicts the task concept as an explicit problem-space decision, and the heuristic searcher uses this decision to restrict symbolic rule search. This design allows us to evaluate not only the final output accuracy, but also whether the model selects a plausible reasoning space before search.

3.1. Preliminaries

Our Decision Decoder is implemented as a BL-gated mixture-of-experts (MoE) module. Behavior Learning (BL) is a structured modeling framework motivated by utility maximization problem (UMP) (Ma et al., 2026). BL represents functions by composing modular B-blocks. In the original formulation, a B-block is parameterized as

$$\mathcal{B}(\mathbf{x}, \mathbf{y}; \theta) := \boldsymbol{\lambda}_0^\top \phi(U_{\theta_U}(\mathbf{x}, \mathbf{y})) - \boldsymbol{\lambda}_1^\top \rho(C_{\theta_C}(\mathbf{x}, \mathbf{y})) - \boldsymbol{\lambda}_2^\top \psi(T_{\theta_T}(\mathbf{x}, \mathbf{y})).$$

where $\theta := (\boldsymbol{\lambda}_0, \boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2, \theta_U, \theta_C, \theta_T)$ denotes the learnable parameters. Following Ma et al. (2026), ϕ is an increasing transformation, ρ penalizes inequality violations, and ψ captures symmetric deviations. Multiple B-blocks can be arranged in parallel to form a BL layer, and BL layers can be composed hierarchically to produce structured decision functions.

In this paper, we use BL blocks as structured function modules. Specifically, BL replaces the standard MLP parameterization of the MoE gate and expert transformations, so that routing and concept prediction depend on structured transformations of the task representation.

MoE models combine multiple expert modules using a learned gating network (Jacobs et al., 1991). Each expert can specialize in a different region of the input or task space, while the gate assigns weights to experts and combines their outputs. Sparse MoE variants restrict computation to a small subset of experts selected by the gate (Shazeer et al., 2017). In our model, MoE provides the modular decision structure, while BL provides a structured bias for routing and concept prediction.

3.2. Inductive Encoder

The Inductive Encoder constructs a task-level representation from the input-output demonstration pairs. Each pair is independently processed by a CNN-based spatial encoder to produce a token that captures grid structure and input-output differences. The resulting token set is contextualized through self-attention without set-level positional encodings. Multi-head attention pooling then aggregates the contextualized tokens into a task embedding.

A separate CNN-based spatial encoder maps the query input to a query token. Cross-attention then conditions this token on the contextualized demonstration pair tokens. The resulting query-conditioned embedding is fused with the task embedding and passed to the Decision Decoder.

The goal of this stage is to represent the abstract transformation shared across demonstrations.

3.3. Decision Decoder

The Decision Decoder maps the unified representation to a concept prior over ARC problem spaces. We implement the decoder as a BL-based mixture-of-experts module with 16 experts.

Let $\mathbf{z}$ denote the unified representation. The BL gating network first computes dense routing weights over experts,

$$\boldsymbol{\pi} = \mathrm{softmax}(g_{\mathrm{BL}}(\mathbf{z})),$$

where $\boldsymbol{\pi} \in \mathbb{R}^{16}$. We then apply top-4 sparse gating by retaining only the four largest routing weights and renormalizing

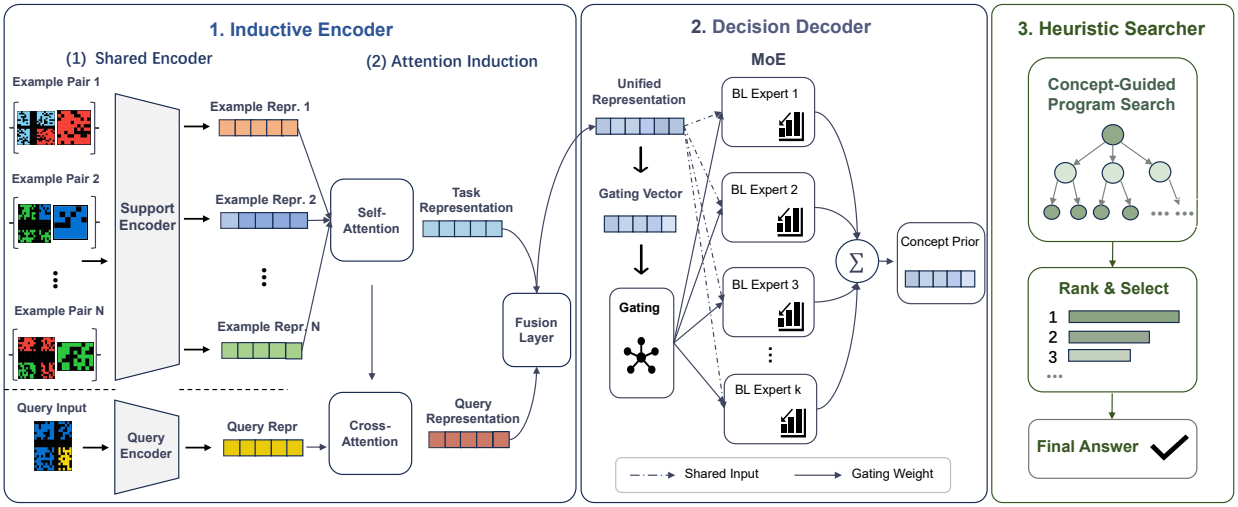

Figure 2. Overview of the Cognitive Reasoning Model. The model encodes demonstration pairs and the query input, predicts a concept prior through a decision decoder, and uses this prior to guide symbolic program search.

them:

$$\tilde{\pi}_k = \frac{\pi_k \mathbb{I}[k \in \mathrm{Top4}(\boldsymbol{\pi})]}{\sum_{j \in \mathrm{Top4}(\boldsymbol{\pi})} \pi_j}.$$

Each BL expert produces concept logits

$$\mathbf{o}_k = f_{\mathrm{BL}}^{(k)}(\mathbf{z}),$$

and the final concept logits are computed as

$$\mathbf{o} = \sum_{k=1}^{16} \tilde{\pi}_k \mathbf{o}_k.$$

Applying a softmax to $\mathbf{o}$ yields a concept prior over problem spaces, which is evaluated using ConceptARC labels and used to guide the subsequent symbolic search.

3.4. Heuristic Searcher

The Heuristic Searcher performs concept-guided program search within the selected problem space. Given the concept prior from the Decision Decoder, it restricts symbolic search over the IceCuber DSL (Wind, 2020). Primitives are grouped by concept family, so the searcher only considers primitive subsets associated with the predicted top-2 concepts. Candidate programs are evaluated on the demonstration pairs, ranked, and applied to the test input.

4. Experiments

This section evaluates our Cognitive Reasoning Model on ConceptARC. We focus on two questions: whether the model can predict task-level concept families as an intermediate problem-space selection step, and whether the predicted concept prior can guide symbolic search efficiently. The evaluation is structured around two findings: (i) our model achieves the highest task-level concept prediction accuracy among the evaluated models; and (ii) concept-guided IceCuber search achieves accuracy comparable to full IceCuber search while reducing search time.

4.1. Setting

We evaluate task-level concept prediction on ConceptARC. The benchmark contains 16 concept families, with 10 tasks in each family. We evaluate on 32 held-out target tasks, corresponding to 2 target tasks per concept family. The remaining tasks in each concept family are used as labeled reference tasks for concept prediction. For each task, the demonstration pairs are provided as input, and the concept family serves as the task-level label. Given a held-out target task, the model predicts its concept family from its demonstration pairs. We compare our model against LLM baselines under the same held-out concept-prediction setting. This evaluation measures whether a system can select the appropriate reasoning space before heuristic search, complementing final output accuracy.

Separately, we evaluate whether the predicted concepts can guide symbolic search. For this experiment, we compare the concept-guided IceCuber search with the IceCuber baseline, allowing at most two candidate outputs per test input for both methods.

4.2. Concept Prediction Results

Figure 3 summarizes task-level ConceptARC concept prediction results. The proposed model achieves 0.4375 accuracy on the target tasks, compared with 0.4062 for ChatGPT 5.4. These preliminary results suggest that our comparatively small model can outperform the strongest evaluated

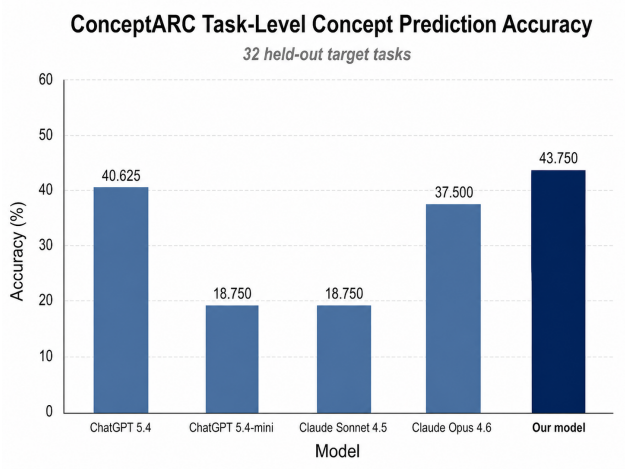

Figure 3. Task-level ConceptARC concept prediction accuracy on 32 held-out target tasks. The proposed model achieves the highest accuracy among the evaluated models.

LLM baseline on task-level concept prediction.

Table 1 breaks down task-level concept prediction by concept family. Our model improves over ChatGPT 5.4 in several categories, but both models remain weak on concepts AboveBelow, Center, HorizontalVertical, and Order.

4.3. Concept-Guided Search

We also evaluate whether predicted concepts can guide heuristic program search. Table 2 compares concept-guided IceCuber search with the IceCuber baseline under the same pass@2 output budget. Our concept-guided search achieves 0.419 overall accuracy, close to the IceCuber baseline at 0.433. Although it does not improve final output accuracy, it retains most of the baseline performance while constraining search through a predicted concept prior. This result suggests that task-level concept prediction can serve as a useful problem-space selection mechanism for heuristic search.

5. Discussion and Future Work

In this work, we introduce the Cognitive Reasoning Model, a process-level framework for ARC reasoning. Our goal is to make the reasoning process explicit by decomposing ARC solving into three stages: inductive representation, problem-space decision, and heuristic search.

The preliminary results support this framework. Our model achieves the strongest task-level concept prediction accuracy among the evaluated models, and the concept-guided IceCuber search achieves accuracy comparable to the IceCuber baseline while using a constrained search space.

More broadly, ARC evaluation should move from asking only whether a model gets the answer right toward asking whether it solves the problem through structured and interpretable reasoning.

For future work, we will analyze failure modes in both concept prediction and concept-guided search, and compare our framework with a broader range of ARC models, including recent visual reasoning, recursive refinement, and program-synthesis approaches.

Table 1. Per-concept task-level ConceptARC accuracy on 32 held-out target tasks, comparing ChatGPT 5.4 with our model.

Concept	ChatGPT 5.4	Our model
AboveBelow	0.0	0.0
Center	0.0	0.0
CleanUp	0.5	0.5
CompleteShape	0.5	0.0
Copy	1.0	0.5
Count	1.0	1.0
ExtendToBoundary	0.5	0.0
ExtractObjects	1.0	1.0
FilledNotFilled	0.0	1.0
HorizontalVertical	0.0	0.0
InsideOutside	0.5	1.0
MoveToBoundary	1.0	0.5
Order	0.0	0.0
SameDifferent	0.5	0.5
TopBottom2D	0.0	0.5
TopBottom3D	0.0	0.5
Overall	0.4062	**0.4375**

Table 2. Per-concept pass@2 accuracy on ConceptARC for the IceCuber baseline and our concept-guided IceCuber search.

Concept	IceCuber	Our Model
AboveBelow	0.700	0.700
Center	**0.400**	0.367
CleanUp	**0.433**	0.267
CompleteShape	**0.400**	0.200
Copy	0.167	0.167
Count	0.467	**0.500**
ExtendToBoundary	0.600	**0.733**
ExtractObjects	0.267	**0.467**
FilledNotFilled	0.700	0.700
HorizontalVertical	**0.300**	0.267
InsideOutside	0.467	**0.667**
MoveToBoundary	0.333	0.333
Order	**0.200**	0.167
SameDifferent	**0.433**	0.100
TopBottom2D	0.567	**0.633**
TopBottom3D	**0.500**	0.433
Overall	0.433	0.419

Impact Statement

This paper presents work whose goal is to advance the study of machine learning systems for abstract reasoning. We do not identify specific societal risks beyond those generally

associated with improved reasoning and program-synthesis systems.

References

Akyürek, E., Damani, M., Zweiger, A., Qiu, L., Guo, H., Pari, J., Kim, Y., and Andreas, J. The surprising effectiveness of test-time training for few-shot learning. *arXiv preprint arXiv:2411.07279*, 2024.

Chollet, F. On the measure of intelligence. *arXiv preprint arXiv:1911.01547*, 2019.

Chollet, F., Knoop, M., Kamradt, G., and Landers, B. Arc prize 2024: Technical report. *arXiv preprint arXiv:2412.04604*, 2024.

Gentner, D. Structure-mapping: A theoretical framework for analogy. *Cognitive science*, 7(2):155–170, 1983.

Hodel, M. Domain specific language for the abstraction and reasoning corpus (arc-dsl), 2023.

Hodel, M. re-arc: Reverse-engineering the abstraction and reasoning corpus, 2024.

Hu, K., Cy, A., Qiu, L., Ding, X. D., Wang, R., Zhu, Y. E., Andreas, J., and He, K. Arc is a vision problem! *arXiv preprint arXiv:2511.14761*, 2025.

Jacobs, R. A., Jordan, M. I., Nowlan, S. J., and Hinton, G. E. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

Jolicoeur-Martineau, A. Less is more: Recursive reasoning with tiny networks. *arXiv preprint arXiv:2510.04871*, 2025.

Li, W.-D., Hu, K., Larsen, C., Wu, Y., Alford, S., Woo, C., Dunn, S. M., Tang, H., Naim, M., Nguyen, D., et al. Combining induction and transduction for abstract reasoning. *arXiv preprint arXiv:2411.02272*, 2024.

Liao, I. and Gu, A. Arc-agi without pretraining. *arXiv preprint arXiv:2512.06104*, 2025.

Ma, Z., Liang, Y., and Li, D. Behavior learning (BL). In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=bbAN9PPcI1.

Moskvichev, A., Odouard, V. V., and Mitchell, M. The conceptarc benchmark: Evaluating understanding and generalization in the arc domain. *arXiv preprint arXiv:2305.07141*, 2023.

Newell, A. and Simon, H. A. *Human Problem Solving*. Prentice-Hall, 1972.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

Shu, W.-J., Qiu, X., Zhu, R.-J., Chen, H. H., Liu, Y., and Yang, H. Loopvit: Scaling visual arc with looped transformers. *arXiv preprint arXiv:2602.02156*, 2026.

Wang, G., Li, J., Sun, Y., Chen, X., Liu, C., Wu, Y., Lu, M., Song, S., and Yadkori, Y. A. Hierarchical reasoning model. *arXiv preprint arXiv:2506.21734*, 2025.

Wind, J. S. Dsl solution to the arc challenge, 2020. URL https://github.com/top-quarks/ARC-solution.

A. ARC-AGI and Recent Reasoning Approaches

ARC-AGI. The Abstraction and Reasoning Corpus (ARC), now commonly known as ARC-AGI, was introduced as a benchmark to measure few-shot generalization in novel tasks (Chollet, 2019). Each ARC-AGI task consists of a small number of input-output demonstration pairs and one or more test inputs, all represented as small two-dimensional grids with discrete cell values. The goal is to infer the underlying transformation from the demonstrations and construct the corresponding output grid for each test input. This setting is designed to evaluate abstract rule discovery from only a few examples.

A central property of ARC-AGI is that each task follows a different underlying logic. The benchmark is therefore designed so that systems cannot solve it by memorizing a fixed training distribution or preparing for specific tasks in advance. ARC-AGI also avoids specialized world knowledge and language dependence: the intended prior knowledge is limited to core knowledge such as objectness, topology, spatial relations, and elementary arithmetic. This makes ARC-AGI particularly relevant for studying abstraction, compositional generalization, and efficient adaptation from few examples.

Recent ARC-AGI approaches. Recent ARC-AGI systems can be broadly understood through the interaction between search, transduction, and refinement. Early successful systems relied on program search in domain-specific languages, where candidate grid transformations are generated and tested against the demonstrations (Wind, 2020; Hodel, 2023). Such approaches are attractive because they produce explicit rules, but they also face a large combinatorial search space.

The ARC Prize 2024 technical report identifies three main directions behind recent progress: deep-learning-guided program synthesis, test-time training for transductive models, and combinations of induction and transduction (Chollet et al., 2024). In this terminology, induction searches for an explicit program or function that maps demonstration inputs to outputs, whereas transduction directly predicts the output grid for a given test input and task specification. Li et al. (2024) study this distinction directly and show that the two paradigms solve partially different sets of tasks. Test-time training (TTT) is a representative transductive strategy, adapting a model to each task instance before predicting the output grid (Akyürek et al., 2024).

The ARC Prize 2025 technical report further identifies refinement loops as a central theme in recent progress. In such systems, a candidate program, model state, latent state, or output is iteratively improved using feedback from the demonstrations or a verifier. These refinement loops may operate in symbolic program space, natural language program space, or neural weight space. Recent recursive and zero-pretraining models instantiate this idea in different ways: HRM uses recurrent latent state computation (Wang et al., 2025), TRM recursively refines the answer and latent states in a compact model (Jolicoeur-Martineau, 2025), and CompressARC optimizes a small equivariant neural network on the target puzzle itself using an MDL-inspired objective (Liao & Gu, 2025). Together, these works suggest that per-task adaptation and iterative refinement have become central mechanisms in recent ARC-AGI progress.

Visual reasoning models provided another important development recently. VARC argues that ARC reasoning should be treated primarily as a visual reasoning problem, using visual architectures as the main inductive bias (Hu et al., 2025). LoopViT follows a related direction by introducing a looped visual transformer computation, allowing the model to iteratively refine its representation in multiple recurrent steps (Shu et al., 2026). Both approaches use Re-ARC to generate data for training or evaluation. Re-ARC procedurally generates variants of ARC tasks, providing a scalable source of grid-transformation examples (Hodel, 2024). These works show that visual models can benefit from visual inductive bias, recurrence, and refinement. Nevertheless, their evaluation is still usually based on the final output prediction.

Position of our work. Our work follows the broader trend of combining learned inference with symbolic search, but focuses on a different intermediate question: whether the system selects the correct reasoning space before search. The standard ARC-AGI evaluation measures whether the final output grid is correct. However, a correct final output does not necessarily reveal whether the model formed an appropriate task representation, identified the relevant problem type, or searched within a human-plausible space of rules.

We use ConceptARC to evaluate both task solving and intermediate problem-space selection. The benchmark organizes ARC tasks into concept groups, allowing us to evaluate whether a system identifies the relevant concept family in addition to whether it solves the task (Moskvichev et al., 2023). In our model, the Inductive Encoder forms a representation from demonstrations, the Decision Decoder predicts a concept prior, and the Heuristic Searcher uses this prior to restrict symbolic search. This places our work between unrestricted program search and direct transduction: the final answer is produced by symbolic search, but the search space is first constrained by an explicit, learned problem-space decision.