ARCTraj: A Dataset and Benchmark of Human Reasoning Trajectories for Abstract Problem Solving

Sejin Kim
GIST

Hayan Choi
VTOV

Seokki Lee
GIST

Sundong Kim
GIST

Abstract

We present ARCTraj, a large-scale dataset of human reasoning trajectories collected from interactive sessions on the Abstraction and Reasoning Corpus (ARC), a visual reasoning benchmark that challenges solvers to induce patterns from input-output grid pairs. While ARC provides only static examples, ARCTraj captures the whole sequence of high-level, object-centric actions humans take to solve these tasks, revealing intermediate steps typically hidden in conventional datasets. Collected via the O2ARC web interface, the dataset includes over 10,000 trajectories aligned with a Markov Decision Process (MDP) structure and enriched with metadata such as timestamps, user IDs, and success labels. ARCTraj has enabled diverse applications across reinforcement learning, sequence modeling, and generative planning, powering models such as PPO, World Models, Decision Transformers, GFlowNets, and diffusion agents. We further analyze the dataset to uncover behavioral patterns in spatial selection, color attribution, and strategy convergence. Together, these contributions position ARCTraj as a structured and interpretable resource for studying human-like reasoning and building cognitively informed learning systems.

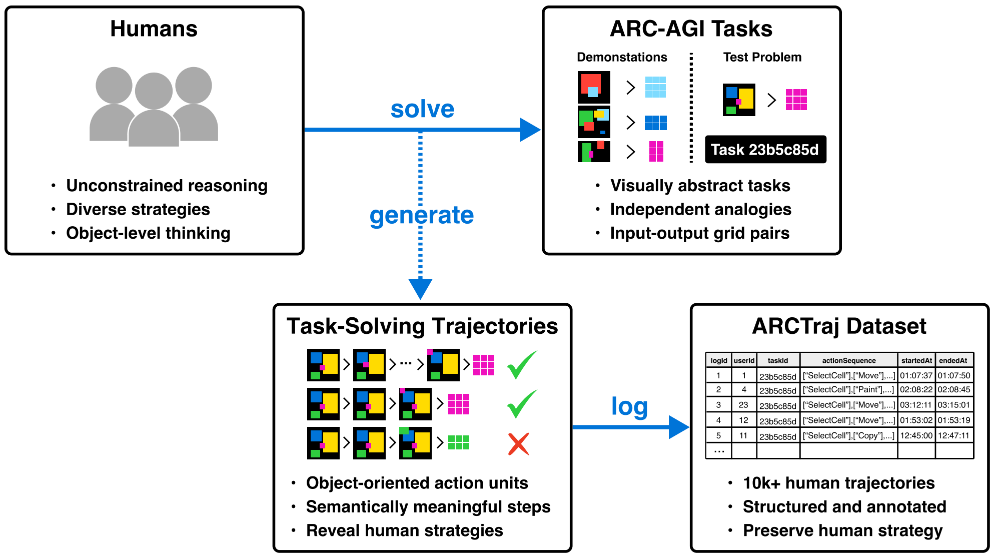
1 Introduction

Understanding how humans reason and solve problems is a longstanding goal in artificial intelligence (AI). Human problem-solving often involves conceptual abstraction, attention shifts, and flexible strategy use,abilities that remain difficult for machines to emulate. The Abstraction and Reasoning Corpus (ARC) [7] was introduced to benchmark such capabilities through grid-based tasks where solvers must infer and apply rules from a small set of input-output examples. While ARC has inspired a wide range of approaches, including program synthesis [3], neuro-symbolic models [19], and test-time learning [2], it only provides static examples, making it difficult to analyze or model the dynamic reasoning processes humans use to solve these tasks.

To address this limitation, we present **ARCTraj**, a large-scale dataset of human reasoning trajectories collected while solving ARC tasks. Each trajectory captures a temporally ordered sequence of object-level actions (i.e., moving objects, rotating objects, and flipping objects) that transform an input grid into its correct output. These logs were collected through the O2ARC web interface [21], designed to support natural human interaction with ARC problems. Each trajectory is annotated with metadata such as timestamps, task identifiers, and success labels, enabling both learning and analysis.

Compared to existing human ARC datasets, ARCTraj offers several advantages. Whereas H-ARC [17] captures pixel-level edit logs and the ARC-Interactive-History-Dataset [24] records low-level cell and operation sequences, ARCTraj provides object-centric actions with consistent formatting across all 400 ARC training tasks. In addition, its public platform (O2ARC) supports visual inspection, replay, and future data expansion, making the dataset extensible and accessible.

Preprint. Under review.



logId	userId	taskId	actionSequence	startedAt	endedAt
1	1	23b5c85d	[“SelectCell”],[“Move”],...]	01:07:37	01:07:50
2	4	23b5c85d	[“SelectCell”],[“Paint”],...]	02:08:22	02:08:45
3	23	23b5c85d	[“SelectCell”],[“Move”],...]	03:12:11	03:15:01
4	12	23b5c85d	[“SelectCell”],[“Move”],...]	01:53:02	01:53:19
5	11	23b5c85d	[“SelectCell”],[“Copy”],...]	12:45:00	12:47:11
…					

Figure 1: Overview of the ARCTraj data collection process. Users solve ARC tasks through the O2ARC platform by interacting with grid-based objects. Their actions are recorded step-by-step to form semantically rich, temporally ordered trajectories.

ARCTraj enables two main lines of research. First, it has been used to train learning agents in various paradigms, including reinforcement learning (e.g., PPO in ARCLE [14]), offline diffusion models [13], world models [15], and generative policies [10, 20]. Second, it allows analysis of human behavior in ARC tasks. In this work, we investigate spatial selection preferences, color source attribution, and strategy variation across users, highlighting the diversity and structure in human reasoning.

In summary, ARCTraj bridges a gap in the ARC ecosystem by providing dynamic, interpretable records of human reasoning. It supports both behavioral studies and cognitively inspired model development, making it a versatile resource for researchers interested in abstraction, planning, and generalization.

2 Related Work

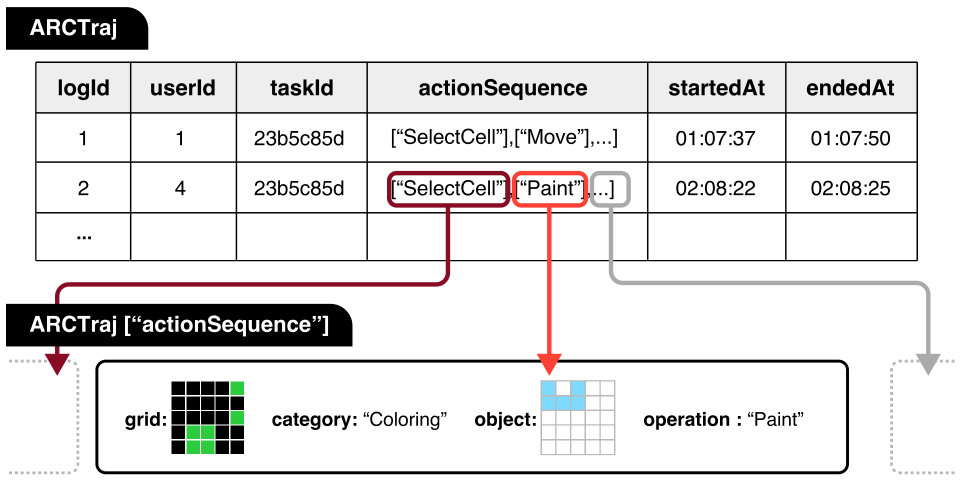
Abstraction and Reasoning Corpus (ARC) The Abstraction and Reasoning Corpus (ARC) [7] is a benchmark to test human-like generalization in abstract reasoning tasks. Each ARC task comprises a few input-output grid pairs, requiring solvers to induce and apply conceptual transformations. It has motivated research across various paradigms, including program synthesis [3, 5, 6], neuro-symbolic reasoning [4, 19, 25], and test-time training [2, 9, 18], many of which were featured in the ARC Prize 2024 Technical Report [8]. However, ARC only provides input-output pairs, limiting insight into the dynamic reasoning steps that underlie human problem solving. In this work, we use the term ARC to refer specifically to the 400 training tasks from the ARC-AGI-1 benchmark, which serves as the basis for all trajectories collected in ARCTraj.

Human Trajectory Datasets for ARC Recent efforts have sought to capture human reasoning on ARC tasks through various forms of interaction data. LARC [1] collects natural language descriptions of task solutions, offering a semantic perspective but lacking action-level granularity. Fast and Flexible [11] and its successor H-ARC [17] describe the same pixel-level edit actions collected during the ARC task-solving dataset. While valuable, these low-level records are difficult to map onto structured reasoning steps or integrate into learning frameworks. The ARC-Interactive-History-Dataset recently introduced logs from the BrainGridGame (BGD) interface [24], capturing cell-level and operation-level actions over time. While this dataset supports MDP-like interpretations and includes many trajectories, it lacks standardized task coverage and object-level abstraction. It has not yet been integrated into downstream learning environments or model training pipelines.

3 The ARCTraj Dataset

ARCTraj is a large-scale dataset that captures human reasoning trajectories for solving abstract visual tasks from the ARC benchmark. Unlike the original ARC dataset, which only provides static input-output pairs, ARCTraj records temporally ordered, semantically grounded action sequences taken by humans while solving ARC tasks. This structure offers a high-resolution view of goal-directed behavior and supports research in sequential decision making, intention inference, and learning from demonstration.

3.1 Task and Collection Protocol



logId	userId	taskId	actionSequence	startedAt	endedAt
1	1	23b5c85d	[“SelectCell”],[“Move”],...]	01:07:37	01:07:50
2	4	23b5c85d	[“SelectCell”],[“Paint”],...]	02:08:22	02:08:25
...					

Figure 2: Example visualization of a single log in ARCTraj. Each action includes its category, operation, and associated grid and object state, forming a structured state-action unit.

All trajectories were collected using **O2ARC 3.0** [21], a custom web interface replicating the core mechanics of ARC tasks in an interactive, user-friendly format. Each task presents at least one input-output example and provides a manipulable input grid. Users interact with the grid through actions applied to *objects*, where each object is an automatically segmented group of adjacent colored pixels. Supported operations include `move`, `color`, `delete`, or `copy`, applied via clicks, keyboard shortcuts, or drag-and-drop. ARCTraj captures user actions at a conceptually coherent level aligned with human reasoning by operating on semantically meaningful objects rather than individual pixels.

Participants were instructed to solve tasks freely, without time limits or strategic constraints. All actions were automatically logged with timestamps and grid states. Each trajectory consists of alternating `selection` and `operation` steps, forming MDP-compatible state-action sequences suitable for downstream learning applications. In total, over 300 users contributed trajectories for all 400 training tasks from ARC-AGI-1, the official training split of the ARC benchmark.

3.2 Dataset Statistics

Table 1: Summary statistics of the ARCTraj dataset.

Metric	Value
Number of trajectories	10,672
Number of unique ARC tasks	400
Number of users	327
Mean trajectory length	9.8
Std. dev. of trajectory length	5.6
Success rate	82.3%
Most frequent action type	`move`
Average time per task	42.7 sec

The dataset includes 10,672 complete trajectories, averaging 9.8 actions per trajectory (std: 5.6). Each trajectory includes metadata such as task ID, user ID, timestamps, action types, and success labels. Participants solved various ARC tasks involving symmetry, object grouping, spatial transformations, and numeric rules. Most actions were completed within 43 seconds on average, indicating natural problem-solving pacing. The `move` operation reflects the design's emphasis on object-level interactions that match human intuitions for manipulating abstract visual elements.

3.3 Comparison with Existing ARC Datasets

Several prior datasets have attempted to capture human reasoning on ARC tasks through logs of natural language, pixel-level interactions, or interface actions. **Fast and Flexible** [11] and its follow-up study **H-ARC** [17] describe the same dataset of pixel-level edit sequences collected from a custom ARC-solving interface. These datasets provide fine-grained logs of human actions at the pixel level and have been used to study planning and temporal patterns. However, the recorded actions are low-level and lack object abstraction, making it difficult to model behavior using structured learning frameworks such as reinforcement learning or sequence modeling.

More recently, the **ARC-Interactive-History-Dataset** [23], collected through the BrainGridGame interface [24], logs cell-level and operation-level actions as humans solve ARC tasks. While BGD provides structured logs with semantic labels and supports MDP-compatible interpretations, it differs from ARCTraj in several important ways: (i) the task set is not aligned with the official ARC benchmark, (ii) object-level operations are inferred rather than explicitly logged, and (iii) no public platform exists for interaction replay or exploration.

In contrast, **ARCTraj** offers a task-aligned, object-centric, and publicly explorable dataset built on the official ARC training split. It features structured state-action sequences with clearly defined object boundaries and operations, captured through a uniform user interface. Moreover, the O2ARC platform [21] enables replay, inspection, and visualization of each trajectory, facilitating both reproducibility and in-depth behavioral analysis.

Overall, ARCTraj complements existing efforts by combining the semantic structure of object-level actions with compatibility for downstream modeling, while providing broader task coverage and an accessible ecosystem for interactive analysis.

4 Human Trajectory Analyses

ARCTraj is not only a dataset of recorded behavior but also a foundation for analyzing strategic diversity and cognitive patterns in human problem-solving. This section presents a series of structured analyses on human trajectories, aiming to reveal both low-level interaction biases and high-level reasoning dynamics embedded in the problem-solving process. We use various methods, including statistical summaries, trajectory clustering, and visualization, to extract insights that may inform the design of human-aligned models.

We organize our analyses around three core research questions (RQs) spanning micro-level behaviors and macro-level reasoning structures. These questions cover a broad spectrum of cognitive aspects, from where humans focus their attention and how they choose colors, to the diversity of problem-solving strategies and the semantic coverage of existing symbolic abstractions.

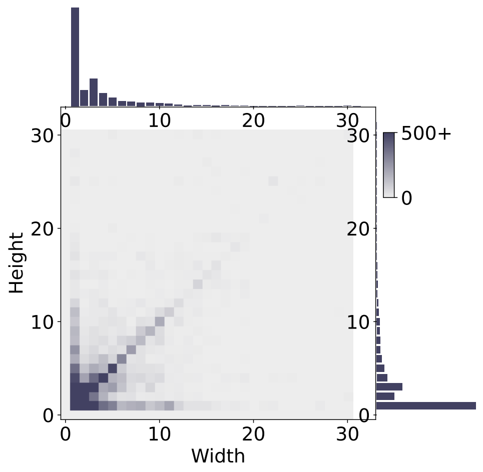
RQ 1. **Do humans exhibit spatial or object-level selection biases when solving ARC tasks?** This question investigates action tendencies such as preferred regions of interaction, the relationship between the number of objects and trajectory length, and whether particular objects are disproportionately selected.

RQ 2. **Where do the colors used in test-time outputs originate, and how do they relate to human color selection strategies?** We examine whether color choices are derived from input grids, reference outputs, or latent reasoning, thereby informing the design of inductive biases in generative models.

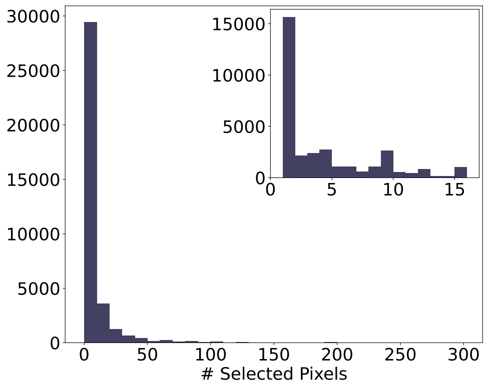
RQ 3. **What strategic patterns and clustering structures emerge across human trajectories solving the same ARC task?** By analyzing variation across solution paths, we uncover how humans approach the same goal differently and how such variations relate to transformation patterns or inferred intentions.

4.1 Biases in Human Grid Selections

To address RQ1, we examine whether humans exhibit systematic selection biases when interacting with ARC grids. In ARCTraj, each *selection* action may correspond to a single pixel, a user-specified region, or an object-level selection that implicitly includes multiple pixels. To unify these heterogeneous selection types, we compute the *bounding box* for each selection—the smallest axis-aligned rectangle that fully contains all selected pixels. We then analyze the distribution of these bounding boxes in terms of their height, width, and area, enabling a consistent characterization of selection scale and shape across the dataset.



(a) Joint distribution of selection height and width.



(b) Histogram of the number of selected pixels.

Figure 3: Distributions of human selection behavior in ARC tasks. Left: Selections are concentrated in compact shapes such as 1×1 to 3×3, with square and bar-shaped regions dominating. Right: Most selections cover fewer than 20 pixels, supporting the preference for local and perceptually salient regions.

As shown in Fig. 3, we identify three dominant tendencies in human selections: (i) selected areas are predominantly small (typically less than 3×3), (ii) selections are often square-shaped ($n \times n$), and (iii) bar-shaped selections ($n \times 1$ or $1 \times m$) also frequently occur. These findings suggest a general preference for local reasoning and perceptual regularity in human problem solving. The left panel of Fig. 3 shows the joint distribution of selection height and width across all selections, revealing an intense concentration in the 1×1 to 3×3 range. A diagonal ridge indicates a square-shape bias, while off-diagonal clusters correspond to frequent horizontal or vertical bar-shaped selections. The marginal histograms reveal firm peaks at width $= 1$ and height $= 1$, indicating a bias toward compact and axis-aligned regions. Complementing this, the right panel shows the distribution of the number of selected pixels per action. Most selections involve no more than 16 pixels. Interestingly, local peaks appear near square numbers (e.g., 1, 4, 9, 16), likely reflecting the high frequency of $n \times n$ square-shaped selections observed in the left panel. This reinforces that humans focus on perceptually salient, compact regions, regardless of task complexity.

Research Direction 1(a): Temporal Dynamics of Selection Behavior. Future work could investigate how selection size and shape evolve during problem-solving. Do humans start with small exploratory selections and later switch to larger ones once a transformation pattern is identified? Or do they first examine the global structure before zooming into specific details? Temporal analysis of selection sequences could provide insight into human attention shifts and inform AI agents' curriculum or phase-based learning strategies.

Research Direction 1(b): Perceptual Features and Selection Probability. Another promising avenue is to examine whether features such as color contrast, spatial isolation, or proximity to grid boundaries influence the likelihood of selection. This could be tested via controlled manipulation of object arrangements and saliency. Modeling this relationship may lead to predictive models of human attention or selection likelihood, which could be incorporated into attention-guided architectures or human-AI collaborative systems.

4.2 Color Source Attribution in Test Outputs

To address RQ2, we examine the origins of colors used in the test output grid and how they relate to human color selection strategies. Color is one of the most critical factors when solving ARC tasks, yet understanding how humans select colors presents unique challenges in our dataset collection methodology.

Analyzing both the task and the collected trajectory reveals that test output colors typically originate from limited sources. Among 400 ARC training tasks, 266 have solutions where colors could be selected exclusively from the color set of the test input grid, while 134 tasks require colors from the union of the color set of the test input grid and the example outputs grid. Interestingly, we found no cases where colors unique to example inputs were necessary for correct solutions, even when considering the complete set of potential sources (test input + example output + example input). This pattern suggests a deliberate task design constraint that limits the search space for potential color sources, focusing primarily on test inputs and secondarily on example outputs while avoiding reliance on example inputs for solution colors.

While our trajectory data does not explicitly record where users sourced their colors, statistical color selection patterns align closely with these potential sources. Users consistently select colors in the test input or example outputs, even without explicit color sampling tools. This suggests humans perform implicit source attribution when reasoning about color transformations, mentally tracking color origins and relationships across different grid examples.

Table 2: Distribution of color sources in ARC tasks. For 66.5% of tasks, all required colors appear in the test input grid. The rest require colors from both the test input and example output grids. No task requires colors exclusive to the example input.

Color Source	**# of Tasks**	**%**
Test Input	266	66.5
Test Input + Example Output	134	33.5
Example Input Only	0	0.0
Others	0	0.0

Research Direction 2(a): Trajectory Logging with Color Origin Capture DSL. Future research would benefit from developing more sophisticated trajectory recording interfaces incorporating explicit color origin tracking mechanisms. By extending current DSLs with formal color sourcing operators (e.g., `sample_color(grid, x, y)` or `apply_color_transformation(rule)`), researchers could create interfaces that allow users to directly sample colors from different grids while precisely logging these conceptual connections. Such integrated systems would document which colors were selected and their relational origins, capturing the mental models humans construct when reasoning about color transformations. These enhanced trajectory capture tools would generate richer datasets that more accurately reflect how humans establish color correspondences across examples, enabling more precise evaluation of computational models against human color selection strategies. Implementing and testing these extended DSLs could significantly advance our understanding of the cognitive processes underlying abstract visual reasoning tasks.

Research Direction 2(b): Generalized Origin Tracking Across Multiple Elements. Building on insights from color origin analysis, future research should explore how humans source and transfer various elements beyond colors when solving ARC tasks. As colors may originate from test inputs or example outputs, other critical problem elements—such as object selection patterns, grid dimensions, transformation sequences, and spatial relationships—likely derive from specific examples within the task. Researchers could develop comprehensive origin tracking frameworks that identify how humans extract and repurpose information across multiple dimensions of the problem space. For instance, when users create specific grid structures, do they primarily reference example output configurations? When selecting objects of particular sizes, are they influenced more by test inputs or example patterns? This multi-dimensional origin analysis would better understand how humans perform cross-example analogical reasoning, revealing which task elements are anchors for different solution development aspects. Such research could significantly advance our understanding of the hierarchical and relational nature of human abstract reasoning, potentially informing more sophisticated computational models that similarly draw information from appropriate sources when building solutions.

4.3 Shared Intentions and Strategy Patterns

To address RQ3, we investigate whether humans solving the same ARC task exhibit converging strategies or follow diverse solution paths. Rather than comparing complete trajectories, we focus on mid-sequence decisions that reflect shared *intentions*, concrete choices about *what* region to act on and *how* to act upon it.

In ARCTraj, each trajectory comprises alternating focus and transformation steps. A focus action highlights a rectangular grid region, which may correspond to an entire object, a shape fragment, or a structured area. After one or more such steps, the user applies a transformation—such as moving, coloring, deleting, or copying—to the focused region. Human solvers do not strictly alternate between focusing and acting; they often examine several areas of succession to inspect the grid or compare subgoals before committing to a transformation. For example, a user might highlight multiple red squares before recoloring a blue one to match, reflecting a search-and-align strategy.

Table 3: Distribution of selection actions preceding each operation. Most operations follow 1–4 selections, suggesting localized exploration before committing to a decision.

Length	Count	%	Cum. %
1	23,632	63.7	63.7
2	5,451	14.7	78.4
3	3,343	9.0	87.4
4	1,379	3.7	91.1
⋮	⋮	⋮	⋮
386	1	0.0	100.0

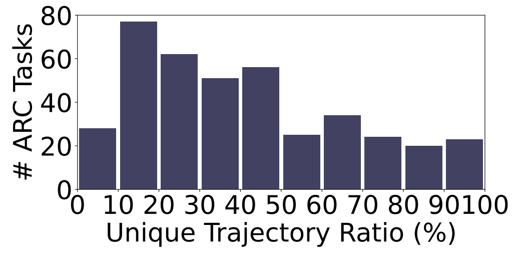


Figure 4: Histogram of task-level trajectory uniqueness. Tasks with low uniqueness (left) exhibit high convergence in human strategies. High uniqueness (right) indicates diverse or ambiguous solution paths.

Our analysis reveals that 63.7% of operations are preceded by a single selection, and over 90% occur within four selections (Table 3). This suggests that a few attentional shifts typically drive human planning before executing a concrete transformation. The short span between selections and operations indicates that humans often perform quick exploratory lookups before committing to a goal-directed action.

To formalize mid-level convergence, we define a shared *intention* as pairing a selection region and an operation type that recurs across different users solving the same task. In other words, if two users both select a 2×2 red square in the lower-left corner and change it to blue, this counts as an instance of shared intention, even if the rest of their actions diverge. This notion captures agreement on *what to do* at a specific point in the problem-solving process, without requiring their entire trajectories to align.

To operationalize this, we extract all (selection, operation) pairs from each trajectory and cluster them within each task based on spatial and semantic similarity. Some tasks exhibit strong convergence, where most users perform the same key transformation on the same grid region. Others show high divergence, with users selecting different substructures or applying varied operations. As shown in Figure 4, tasks with low trajectory uniqueness often correspond to visually salient or intuitively structured solutions. In contrast, ARC tasks with high uniqueness are more ambiguous or allow for multiple valid approaches.

Research Direction 3(a): Strategy Grammar and Diversity Mapping. Future work could formalize intention clusters into a compositional strategy grammar that captures reusable abstraction templates across tasks. Identifying common strategy motifs (e.g., “color and duplicate,” “fold and align”) would enable interpretable models and human-aligned planning systems.

Research Direction 3(b): Intention Prediction and Curriculum Design. Another promising direction is to train models that predict the distribution of human intentions based on task features. This could inform the sequencing of curriculum tasks, scaffold learning from easier to more diverse cases, and support adaptive tutoring systems that anticipate user strategies and provide personalized guidance.

5 Learning with ARCTraj

ARCTraj enables various downstream learning applications by providing structured, object-level, and temporally ordered human trajectories that can be interpreted as state-action sequences. This structure supports both reinforcement learning (RL) and sequence modeling approaches, offering rich supervision derived from human reasoning. This section describes how ARCTraj has been used in interactive and non-interactive learning settings and summarizes empirical results from prior work.

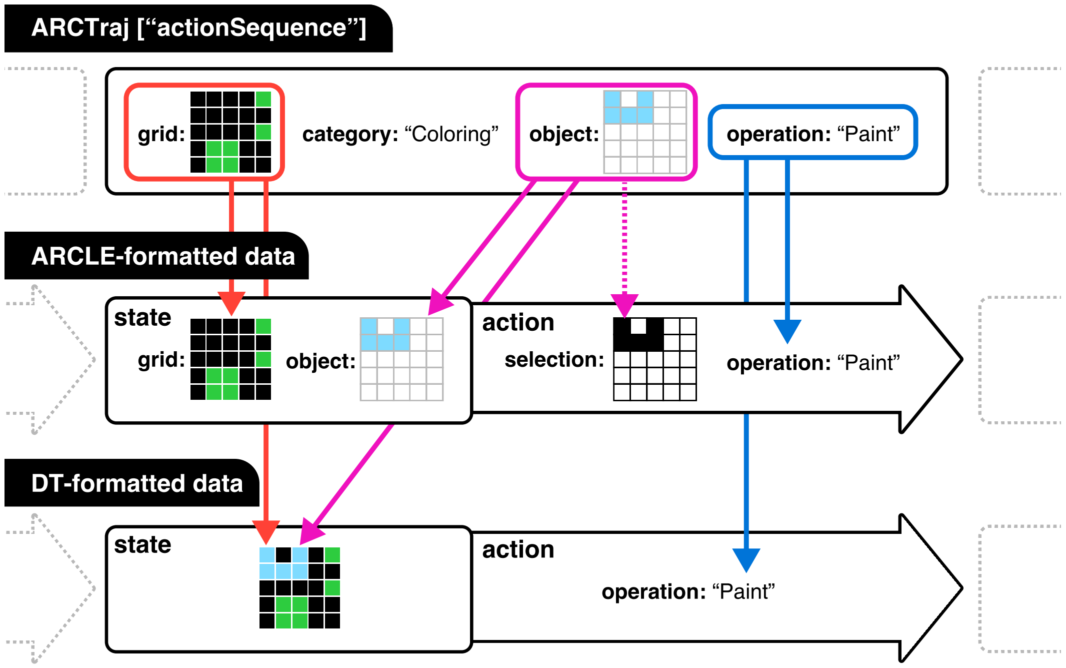


Figure 5: Preprocessing ARCTraj for downstream learning. For RL environments such as ARCLE, ARCTraj is filtered to retain only `operation` actions and is mapped to a Markovian state-action format using `grid`, `object`, and `operation`. For sequence models such as Decision Transformer, only `grid` and `operation` are used, omitting intermediate objects and environment interaction.

5.1 Offline Reinforcement Learning with ARCTraj

ARCTraj can be naturally interpreted as a sequence of (s_t, a_t) pairs, where each state s_t encodes the grid and active object, and each action a_t denotes a high-level transformation applied by a human solver. This structure directly supports offline reinforcement learning (RL) and has been used to construct ARCLE [14], a custom environment aligned with ARCTraj's interaction interface. In ARCLE, trajectories are filtered to retain transformation steps while discarding intermediate selections, enabling agents to learn symbolic manipulation policies grounded in human decisions. PPO-based agents trained in this setting demonstrate the feasibility of learning task-generalizable behaviors from human data.

Several generative planning agents also leverage ARCTraj. LDCQ [13] augments the trajectories by interpolating intermediate grid states between human actions, enabling diffusion models to synthesize coherent, multi-step plans. DreamerV3-based models [15] train world models on ARCTraj to capture latent dynamics and perform analogical generalization across structurally similar tasks. GFlowNet-based methods [10] treat human trajectories as samples from an implicit distribution over valid solutions, training agents to generate diverse, goal-consistent behaviors that mirror the variation seen in human problem solving.

Together, these applications show that ARCTraj supports both imitation-style policy learning and generative behavior modeling. Its structured and semantically grounded trajectories enable agents to reason symbolically, generalize across tasks, and sample solutions in a manner consistent with human strategies.

5.2 Trajectory-Based Learning Without Explicit Environments

ARCTraj also supports learning in non-interactive frameworks without requiring simulation environments. As shown on the right of Fig. 5, trajectories can be simplified by extracting sequences of `grid` states and `operation` actions, omitting the object-level details. This compact format is well-suited for sequence models such as Decision Transformers [20], which treat trajectory length and success as proxy rewards and learn policies directly from demonstrations.

In addition, intention-based approaches further utilize ARCTraj to align low-level actions with latent subgoals. For instance, recent work [12] introduces intention-conditioned supervision to enhance generalization and interpretability. These studies show that incorporating inductive biases, such as latent goal structure, can improve trajectory modeling even in the absence of explicit environments.

5.3 Summary of Learning Outcomes

Table 4 summarizes key methods and findings across interactive and non-interactive paradigms using ARCTraj. These applications demonstrate that ARCTraj is a flexible foundation for training policies, planning agents, and generative models. Its structured and interpretable design supports both symbolic and neural architectures, facilitating comparisons across modeling approaches and enabling progress on problems such as alignment, abstraction, and data-efficient learning.

Table 4: Summary of representative methods and findings from prior work leveraging ARCTraj.

Research	**Setting**	**Model**	**Key Findings**
ARCLE [14]	Online RL	PPO	Demonstrated the feasibility of training in an ARCTraj-compatible MDP environment.
LDCQ [13]	Offline RL	Diffusion	Augmented ARCTraj with intermediate states to enable generative plan synthesis.
DreamerV3 [15]	Offline RL	World Model	Enabled analogical generalization by modeling latent dynamics transferable across tasks.
GFlowNet [10]	Offline RL	GFlowNet	Sampled goal-directed trajectories aligned with expert-like solutions.
Decision Transformer [20]	Offline	Transformer	Learned trajectory-conditioned policies from human demonstrations.
Intention Learning [12]	Offline	Transformer	Enhanced generalization via subgoal alignment and intention-conditioned modeling.

6 Conclusion

ARCTraj captures over 10,000 human trajectories on ARC tasks, offering fine-grained, step-by-step records of people engaging with abstract visual reasoning problems. Unlike conventional datasets that only provide static input-output pairs, ARCTraj logs temporally ordered, object-level actions grounded in human perceptual and symbolic understanding. This structure provides a window into intermediate reasoning processes and enables detailed modeling of how people plan, adapt, and transform problem representations during task solving.

These structured trajectories have already supported a wide range of learning settings. They enable reinforcement learning agents to be trained from human demonstrations, guide generative planners through trajectory-based data augmentation, and support intention-aware modeling that uncovers latent subgoals. Moreover, the dataset has proven compatible with a variety of architectures—including PPO, diffusion models, GFlowNets, and decision transformers—highlighting its versatility across paradigms.

Beyond supporting modeling, ARCTraj enables the analysis of behavioral and cognitive patterns that are difficult to observe in final outputs alone. Our findings reveal consistent regularities in selection behavior, biases in color attribution, and clustered transformation strategies, which offer a new lens on human abstract reasoning and task decomposition. Such insights can inform the design of inductive biases, curriculum structures, and evaluation protocols for cognitively aligned AI.

References

[1] Samuel Acquaviva, Yewen Pu, Marta Kryven, Theodoros Sechopoulos, Catherine Wong, Gabrielle Ecanow, Maxwell Nye, Michael Tessler, and Joshua B. Tenenbaum. Communicating Natural Programs to Humans and Machines. In *NeurIPS Datasets and Benchmarks*, 2022.

[2] Ekin Akyürek, Mehul Damani, Linlu Qiu, Han Guo, Yoon Kim, and Jacob Andreas. The Surprising Effectiveness of Test-Time Training for Abstract Reasoning. *arXiv:2411.07279*, 2024.

[3] Shraddha Barke, Emmanuel Anaya Gonzalez, Saketh Ram Kasibatla, Taylor Berg-Kirkpatrick, and Nadia Polikarpova. HYSYNTH: Context-Free LLM Approximation for Guiding Program Synthesis. *arXiv:2405.15880*, 2024.

[4] Paweł Batorski, Jannik Brinkmann, and Paul Swoboda. NSA: Neuro-Symbolic ARC Challenge. *arXiv:2501.04424*, 2025.

[5] Mikel Bober-Irizar and Soumya Banerjee. Neural Networks for Abstraction and Reasoning. *Scientific Reports*, 14(1):27823, 2024.

[6] Natasha Butt, Blazej Manczak, Auke Wiggers, Corrado Rainone, David Zhang, Michaël Defferrard, and Taco Cohen. CodeIt: Self-Improving Language Models with Prioritized Hindsight Replay. *arXiv:2402.04858*, 2024.

[7] François Chollet. On the Measure of Intelligence. *arXiv:1911.01547*, 2019.

[8] Francois Chollet, Mike Knoop, Gregory Kamradt, and Bryan Landers. Arc Prize 2024: Technical Report. *arXiv:2412.04604*, 2024.

[9] Daniel Franzen, Jan Disselhoff, and David Hartmann. The LLM ARChitect: Solving ARC-AGI Is A Matter of Perspective, 2024.

[10] Sanha Hwang, Sejin Kim, Seungpil Lee, and Sundong Kim. Solution Augmentation for ARC-AGI Problems Using GFlowNet: A Probabilistic Exploration Approach. *Transactions on Machine Learning Research (submitted)*, 2024.

[11] Aysja Johnson, Wai Keen Vong, Brenden M. Lake, and Todd M. Gureckis. Fast and Flexible: Human Program Induction in Abstract Reasoning Tasks. In *CogSci*, 2021.

[12] Sejin Kim, Hosung Lee, and Sundong Kim. Addressing and Visualizing Misalignments in Human Task-Solving Trajectories. In *ICLR Workshop on Bidirectional Human-AI Alignment*, 2025.

[13] Yunho Kim, Jaehyun Park, Heejun Kim, Sejin Kim, Byung-Jun Lee, and Sundong Kim. Diffusion-Based Offline RL for Improved Decision-Making in Augmented ARC Task. *arXiv preprint arXiv:2410.11324*, 2024.

[14] Hosung Lee, Sejin Kim, Seungpil Lee, Sanha Hwang, Jihwan Lee, Byung-Jun Lee, and Sundong Kim. ARCLE: The Abstraction and Reasoning Corpus Learning Environment for Reinforcement Learning. In *CoLLAs*, 2024.

[15] Jihwan Lee, Woochang Sim, Sejin Kim, and Sundong Kim. Enhancing Analogical Reasoning in the Abstraction and Reasoning Corpus via Model-Based RL. In *IJCAI Workshop on Interactions between Analogical Reasoning and Machine Learning*, 2024.

[16] Seungpil Lee, Woochang Sim, Donghyeon Shin, Sanha Hwang, Wongyu Seo, Jiwon Park, Seokki Lee, Sejin Kim, and Sundong Kim. Reasoning Abilities of Large Language Models: In-Depth Analysis on the Abstraction and Reasoning Corpus. *ACM Transactions on Intelligent Systems and Technology*, 2024.

[17] Solim LeGris, Wai Keen Vong, Brenden M Lake, and Todd M Gureckis. H-ARC: A Robust Estimate of Human Performance on the Abstraction and Reasoning Corpus Benchmark. *arXiv:2409.01374*, 2024.

[18] Wen-Ding Li, Keya Hu, Carter Larsen, Yuqing Wu, Simon Alford, Caleb Woo, Spencer M Dunn, Hao Tang, Michelangelo Naim, Dat Nguyen, Wei-Long Zheng, Zenna Tavares, Yewen Pu, and Kevin Ellis. Combining induction and transduction for abstract reasoning. In *ICLR*, 2025.

[19] Isaac Liao and Albert Gu. CompressARC: ARC Solving Without Data Augmentation or Pretraining, 2024.

[20] Jaehyun Park, Jaegyun Im, Sanha Hwang, Mintaek Lim, Sabina Ualibekova, Sejin Kim, and Sundong Kim. Unraveling the ARC Puzzle: Mimicking Human Solutions with Object-Centric Decision Transformer. In *ICML Workshop on Interactive Learning with Implicit Human Feedback*, 2023.

[21] Suyeon Shim, Dohyun Ko, Hosung Lee, Seokki Lee, Doyoon Song, Sanha Hwang, Sejin Kim, and Sundong Kim. O2ARC 3.0: A Platform for Solving and Creating ARC Tasks. In *IJCAI Demo*, 2024.

[22] Woochang Sim, Hyebin Jin, Sejin Kim, and Sundong Kim. The Possibility of Prompt Engineering for ARC Problem Solving. *KIISE Transactions on Computing Practices*, 2024.

[23] Simon Strandgaard. ARC-Interactive-History-Dataset, 2024.

[24] Simon Strandgaard. Brain Grid Game, 2024.

[25] Yudong Xu, Elias B. Khalil, and Scott Sanner. Graphs, Constraints, and Search for the Abstraction and Reasoning Corpus. In *AAAI*, 2023.

Technical Appendices and Supplementary Material

This supplement provides materials that support the paper's main contributions. It includes a conceptual overview of ARC tasks with examples (Appendix A), a description of the O2ARC 3.0 interface and its action-level DSL for trajectory collection (Appendix B), and a walkthrough of the ARCTraj Viewer (Appendix C). Appendix D outlines four analytic directions from ARCTraj: selection behavior, abstraction patterns, reasoning strategies, and model training applications. These directions provide auxiliary knowledge $\mathcal{A}$ that supports the few-shot solver f_θ. Together, these materials enhance transparency, support reproducibility, and offer deeper insight into the dataset and its use.

A ARC Task Examples and Reasoning Strategies

ARC Overview and Problem Format The Abstraction and Reasoning Corpus (ARC) [7] is a benchmark for evaluating abstract reasoning through symbolic grid-based tasks. Each task provides input-output demo pairs that follow an implicit transformation rule, which must be inferred and applied to a new test input. Grids vary in size, structure, and color, with transformations involving color changes, object manipulation, spatial shifts, or logical patterning. As no explicit instructions are given, solving requires generalizing from visual structure alone. ARC evaluates whether a system can extract abstract rules and apply them flexibly based on minimal examples.

Example ARC Tasks Fig. 6 shows four selective ARC tasks used in our study. Each includes a few demonstration pairs and one test input. The selected tasks span a range of reasoning types, from applying local row-wise rules to performing object-level reconfiguration, visual abstraction, and size-dependent pattern generation.

Task 1: The leftmost color in each row determines the fill color for all gray cells in that row.
Task 2: Four identical objects, same shape and color, are moved to the four corners of the grid.
Task 3: The topmost of several overlapping lines is selected, and its color fills the output grid.
Task 4: A blue grid-like pattern is generated, with complexity based on the input grid size.

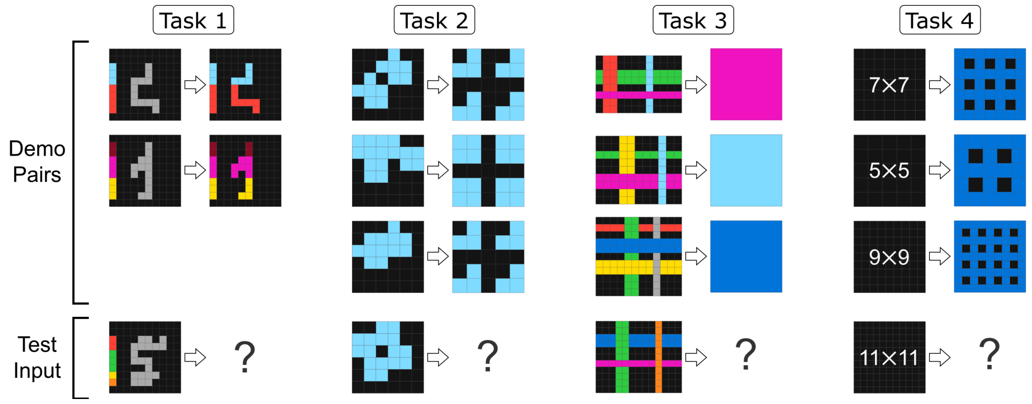


Figure 6: Illustration of sample ARC tasks, each with demo pairs and a test input. Adapted from [22] with permission from the authors.

Common Reasoning Types in ARC The tasks above demonstrate distinct reasoning patterns that reflect the diversity of ARC. Some require recognizing and propagating local rules, while others depend on identifying and relocating objects within the grid. In other cases, the model must resolve visual occlusion or generate patterns conditioned on input size. These variations test a system's ability to compose low-level perception with high-level abstraction using only a few examples.

Task 1:	Local rule execution and partial structure composition	(*local* + *composition*)
Task 2:	Object-based detection and spatial relocation	(*objectness* + *spatial*)
Task 3:	Visual layering and selective abstraction	(*abstraction* + *occlusion-aware*)
Task 4:	Pattern generation with size-dependent recursion	(*generalization* + *recursion*)

B O2ARC 3.0 Interface and Action DSL

Interface and Trajectory Logging O2ARC 3.0 [21] is a web interface for collecting structured user trajectories during ARC task solving. As shown in Fig. 7, the screen consists of three parts: the left panel presents demonstration input-output pairs to illustrate the transformation rule, the center panel shows a fixed test input grid, and the right panel provides an editable output grid, a set of symbolic editing commands, and a "Submit" button. Users solve the task by selecting regions in the output grid and applying grid-editing functions such as coloring, flipping, or copying. We refer to these functions as **symbolic operations**, the predefined actions available through the interface. Each user interaction is recorded in order, forming a symbolic trajectory stored as a JSON array named `actionSequence`. Typically, a solution involves selecting a region, choosing an appropriate operation, applying the change, and repeating this loop until the user clicks "Submit". Correct answers yield a score based on time and number of actions used, while incorrect submissions reduce one of the three visible hearts.

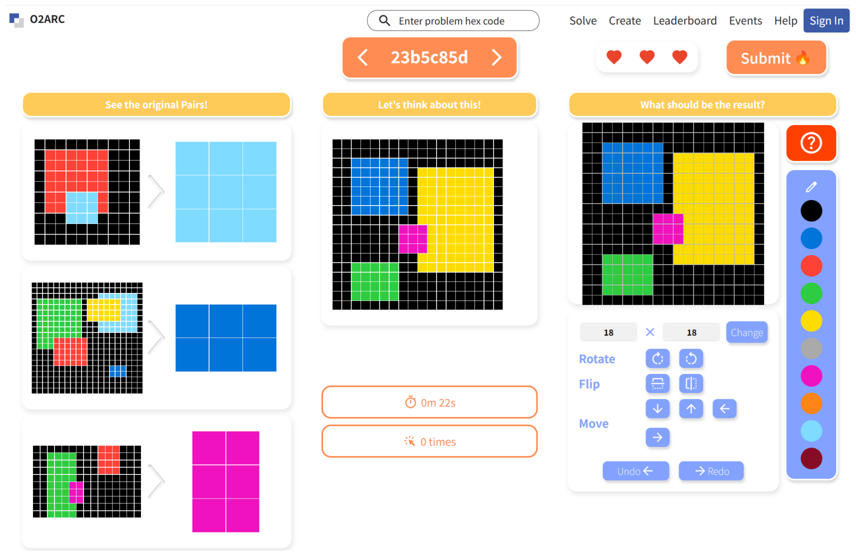


Figure 7: O2ARC 3.0 interface. The left panel shows demonstration pairs, the center panel displays the test input, and the right panel allows symbolic editing of the test output.

Categorization of Symbolic Operations The interface supports 40 symbolic commands grouped into five functional categories: *Coloring*, *Object-Oriented*, *Clipboard*, and *Critical*. Fig. 8 shows each type's full command set and example transformations. *Coloring* commands (`Color0`–`Color9`) apply a specific color to selected pixels. *Object-Oriented* commands such as `Rotate90`, `MoveL`, and `FlipV` apply spatial transformations to selected regions. *Clipboard* commands (`CopyI`, `CopyO`, `Paste`) allow copying and pasting between grids. Finally, *Critical* commands like `Submit`, `ResizeGrid`, or `ResetGrid` govern the overall task flow. This categorization reflects different cognitive operations employed during problem solving and enables modular interpretation of user strategies.

Mapping User Interaction to JSON Format The ARCTraj dataset stores symbolic trajectories generated by users interacting with the O2ARC interface. Each user action, such as selecting a pixel or applying a transformation, is recorded as a structured JSON object. This object includes the action's category (e.g., Selection, Coloring), the specific operation name (e.g., `SelectCell`, `Paint`), and the position of the selected pixel. The whole grid state at the moment is captured in the `grid` field, while the `object` field contains the set of all pixels cumulatively selected so far. The `overlapped` flag indicates whether a symbolic transformation was applied to the selected region. Each action is appended to an `actionSequence` along with a precise `timestamp`, forming a complete, interpretable record of the user's problem-solving trajectory. An example action log is shown in Listing 1.

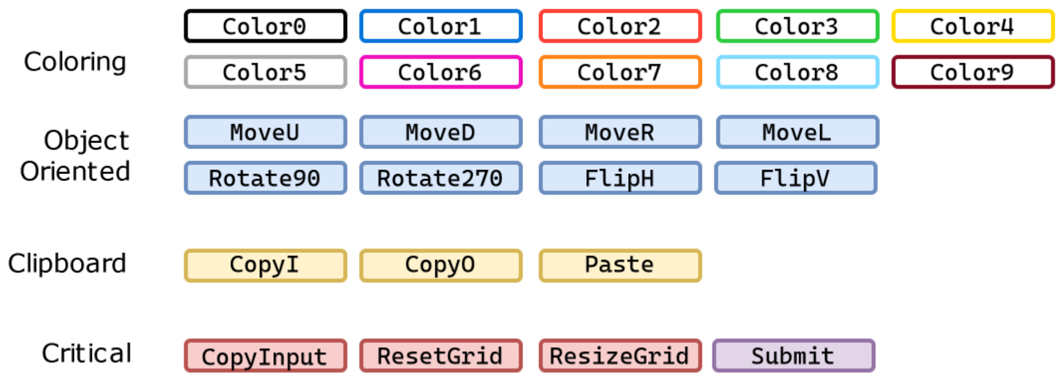


Figure 8: Symbolic operations in O2ARC 3.0 [14], grouped into five categories: *Coloring*, *Object-Oriented*, *Clipboard*, and *Critical*. For clarity, the *Coloring* groups are visualized by color index and affected region. In ARCTraj data, these are stored as a single symbolic operation with a color argument.

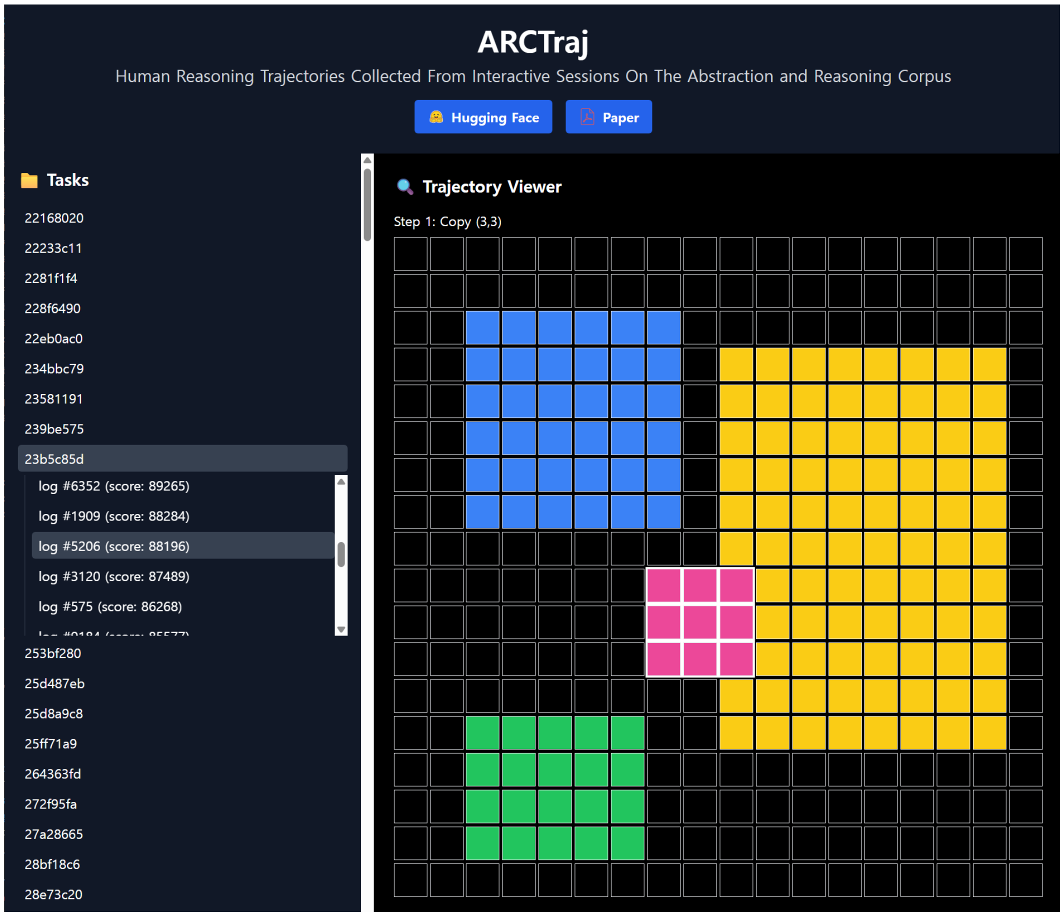
Listing 1: JSON-formatted data from the "actionSequence" column of ARCTraj, recorded when the user clicks the top-left corner pixel of the magenta square object in the output grid of Fig. 7. It logs the operation type (`SelectCell`), selected position, grid state, accumulated object selection, and whether a symbolic transformation has been applied.

```
{
  "category": "Selection",
  "operation": "SelectCell",
  "position": {
    "x": 7,
    "y": 9
  },
  "grid": [
    [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],
    [0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 1, 1, 1, 1, 1, 1, 0, 4, 4, 4, 4, 4, 4, 4, 4, 0],
    [0, 0, 1, 1, 1, 1, 1, 1, 0, 4, 4, 4, 4, 4, 4, 4, 4, 0],
    [0, 0, 1, 1, 1, 1, 1, 1, 0, 4, 4, 4, 4, 4, 4, 4, 4, 0],
    [0, 0, 1, 1, 1, 1, 1, 1, 0, 4, 4, 4, 4, 4, 4, 4, 4, 0],
    [0, 0, 1, 1, 1, 1, 1, 1, 0, 4, 4, 4, 4, 4, 4, 4, 4, 0],
    [0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 4, 4, 4, 4, 4, 4, 4, 0],
    [0, 0, 0, 0, 0, 0, 0, 6, 6, 6, 4, 4, 4, 4, 4, 4, 4, 0],
    [0, 0, 0, 0, 0, 0, 0, 6, 6, 6, 4, 4, 4, 4, 4, 4, 4, 0],
    [0, 0, 0, 0, 0, 0, 0, 6, 6, 6, 4, 4, 4, 4, 4, 4, 4, 0],
    [0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 4, 4, 4, 4, 4, 4, 4, 0],
    [0, 0, 3, 3, 3, 3, 3, 0, 0, 4, 4, 4, 4, 4, 4, 4, 4, 0],
    [0, 0, 3, 3, 3, 3, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 3, 3, 3, 3, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
    [0, 0, 3, 3, 3, 3, 3, 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]
  ],
  "object": [
    {
      "x": 7,
      "y": 9,
      "color": 6
    }
  ],
  "overlapped": true,
  "timestamp": "2024-02-15T01:07:40.537Z"
}
```

C ARCTraj Viewer: Visualization Tool for Trajectory Data

Interactive Web Viewer for ARCTraj To support ARCTraj exploration and human behavior analysis, we developed an interactive viewer at `https://arc-traj-viewer.vercel.app/`. It allows users to browse ARC tasks and inspect symbolic action sequences step-by-step.

Interface Overview The ARCTraj Viewer is a web-based tool for visualizing human trajectories collected from ARC tasks. The top bar includes quick-access links to the Hugging Face dataset and the paper. Users can browse 400 ARC tasks on the left and view associated logs sorted by a custom trajectory quality score. This score, defined over 100,000 points, rewards concise and fast trajectories and assigns a score of zero to any trajectory that includes at least one incorrect submission. After a short initial loading time, selecting a task reveals its logs; selecting a log updates the right panel with a grid and operation display (e.g., `Step 0: SelectGrid (9, 7)`). Users can step through the trajectory using the arrow keys or swipe gestures. Clicking the task again collapses the log list, allowing efficient browsing across multiple tasks.


ARCTraj

Human Reasoning Trajectories Collected From Interactive Sessions On The Abstraction and Reasoning Corpus

22168020
22233c11
2281f1f4
228f6490
22eb0ac0
234bbc79
23581191
239be575
23b5c85d
log #6352 (score: 89265)
log #1909 (score: 88284)
log #5206 (score: 88196)
log #3120 (score: 87489)
log #575 (score: 86268)
253bf280
25d487eb
25d8a9c8
25ff71a9
264363fd
272f95fa
27a28665
28bf18c6
28e73c20

Step 1: Copy (3,3)

Figure 9: Web interface for ARCTraj Viewer. A task is selected on the left panel, a trajectory log is chosen from the center, and the corresponding step-wise grid state and symbolic operation are displayed on the right. The interface allows visual inspection of human problem-solving behavior.

Usage and Applications Fig. 9 shows a sample session in which the user selects a task and a log, resulting in a step-by-step view of the symbolic trajectory. This visual interface enables intuitive inspection of the reasoning process encoded in the `actionSequence`, allowing researchers to replay and analyze human trajectories with step-level precision. The viewer supports diverse use cases such as case studies, strategy comparison, or instructional demonstrations. Its lightweight design makes it suitable for empirical analysis, hypothesis generation, and exploratory examination of human-like reasoning patterns.

D Analytic Pathways from ARCTraj to ARC Solvers

To understand how ARCTraj analysis contributes to solving ARC tasks, we begin by formalizing the ARC objective as a few-shot reasoning problem. Each task provides a small set of demonstration examples:

$$\mathcal{D}_{\text{demo}} = \left\{(x_k^{\text{demo}}, y_k^{\text{demo}})\right\}_{k=1}^{K}$$

where x_k^{demo} and y_k^{demo} denote input-output grid pairs illustrating the transformation pattern. The goal is to predict the output $\hat{y}^{\text{test}}$ for a new input x^{test} using:

$$f_\theta : (x^{\text{test}}, \mathcal{D}_{\text{demo}}, \mathcal{A}) \mapsto \hat{y}^{\text{test}}$$

This can also be expressed as:

$$\hat{y}^{\text{test}} = \arg\max_{y} P(y \mid x^{\text{test}}, \mathcal{D}_{\text{demo}}, \mathcal{A})$$

Here, $\mathcal{A}$ represents auxiliary knowledge that complements the input-output demonstrations. In our setting, $\mathcal{A}$ is derived from ARCTraj, a dataset of symbolic trajectories collected from human solvers. These logs record step-wise interactions, including selection decisions, symbolic operations, and their temporal ordering. While $\mathcal{D}_{\text{demo}}$ offers static input-output pairs, ARCTraj reveals the underlying reasoning processes and intermediate abstractions used to reach the correct output. This information helps reduce ambiguity, guides model learning, and supports generalization.

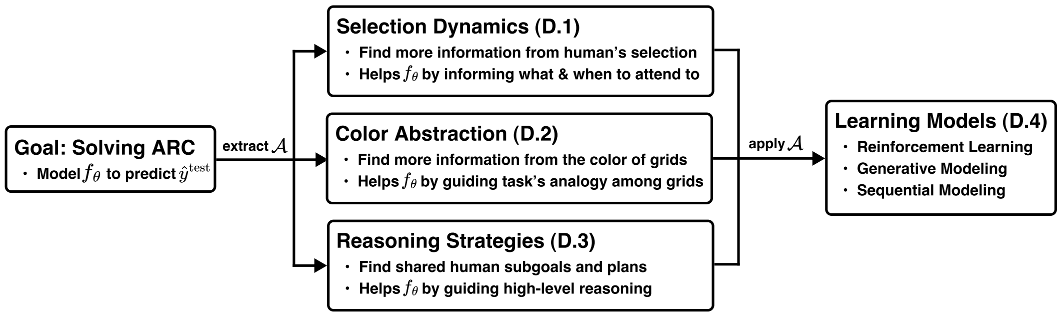

Goal: Solving ARC
• Model f_θ to predict $\hat{y}^{\text{test}}$
extract $\mathcal{A}$
Selection Dynamics (D.1)
• Find more information from human's selection
• Helps f_θ by informing what & when to attend to
Color Abstraction (D.2)
• Find more information from the color of grids
• Helps f_θ by guiding task's analogy among grids
Reasoning Strategies (D.3)
• Find shared human subgoals and plans
• Helps f_θ by guiding high-level reasoning
apply $\mathcal{A}$
Learning Models (D.4)
• Reinforcement Learning
• Generative Modeling
• Sequential Modeling

Figure 10: Overview of how ARCTraj analyses support ARC task solving. The ARC solver f_θ aims to predict $\hat{y}^{\text{test}}$ given x^{test}, demonstration examples $\mathcal{D}_{\text{demo}}$, and auxiliary knowledge $\mathcal{A}$ extracted from ARCTraj. D.1 to D.3 contribute structured signals such as selection dynamics, color abstraction, and reasoning strategies, while D.4 summarizes models that incorporate these signals during training.

Fig. 10 outlines the analytic pipeline from ARCTraj to ARC solving. D.1 through D.3 extract distinct types of structured knowledge that enrich $\mathcal{A}$, each targeting a critical subcomponent of the reasoning process. D.4, in turn, presents examples of learning-based solvers that utilize these signals during training.

- **Selection dynamics** (Appendix D.1) analyzes human visual attention and temporal selection patterns. These inform attention models and guide curriculum construction.
- **Color-based abstraction** (Appendix D.2) explores how color grouping informs object-level reasoning. These abstractions reduce solution space complexity and support symbolic composition.
- **Reasoning strategies and shared intentions** (Appendix D.3) identify common subgoals and modular strategies across solvers. These structures support symbolic reasoning and intention prediction.
- **Use of ARCTraj in learning-based solvers** (Appendix D.4) reviews recent approaches that incorporate ARCTraj in training pipelines via imitation, reward augmentation, or latent modeling.

These analytic directions span from cognitive perception to symbolic planning and implementation. By extracting structured priors from ARCTraj and incorporating them into ARC solvers, they enable learning systems that are more human-aligned, sample-efficient, and interpretable.

D.1 Cognitive and Perceptual Factors in Selection Behavior

To support the few-shot ARC solver f_θ, it is essential to extract structured auxiliary knowledge $\mathcal{A}$ from human solving behavior. One crucial source of such structure is how humans select grid regions, when and where they focus, and what perceptual signals guide those decisions. This section analyzes selection behavior in ARCTraj trajectories to derive generalizable cognitive and perceptual signals, contributing to constructing attention models, curriculum structures, and reasoning scaffolds for f_θ.

Temporal Dynamics of Selection Behavior Human solvers adapt their selection behavior over time. They often begin with exploratory steps, selecting small or uncertain regions, and later shift toward larger, semantically meaningful areas once patterns are recognized. These transitions encode phases of reasoning, such as hypothesis generation and confirmation. Modeling this evolution reveals temporal structure within trajectories and suggests how an ARC solver might emulate similar reasoning progressions.

Let each selection step be represented by $S_t = (x_t, y_t, w_t, h_t)$, the bounding box selected at time t. A trajectory of T steps yields a temporal selection sequence:

$$\mathcal{S} = \{S_1, S_2, \ldots, S_T\}$$

From this, we can derive either a generative model of human attention:

$$h_\phi : \mathcal{S}_{1:t-1} \mapsto S_t$$

or a latent-phase model:

$$P(\text{phase}_t \mid \mathcal{S}_{1:t-1})$$

where phase_t denotes the cognitive mode (e.g., exploratory or exploitative). These phases typically include exploration, where the solver tests unfamiliar regions to generate hypotheses, and exploitation, where it applies known patterns to complete the task efficiently. These models allow us to inject temporal attention priors into f_θ or define adaptive solving curricula based on phase dynamics.

Perceptual Features and Selection Probability Beyond temporal trends, human selections are biased by perceptual cues. Salient features (i.e., color contrast, spatial isolation, or alignment with grid edges) impact selected regions. Modeling these factors enables predicting human attention and designing systems prioritizing perceptually meaningful areas.

Each pixel p can be represented by a perceptual feature vector $\mathbf{v}(p)$, which includes a set of example features such as local contrast and spatial context, defined as below.

$$\mathbf{v}(p) = \begin{bmatrix} v_1(p) \\ v_2(p) \\ v_3(p) \\ \vdots \end{bmatrix} \quad \text{where,}$$

- $v_1(p)$: local color contrast
- $v_2(p)$: distance to nearest edge
- $v_3(p)$: spatial isolation score
- $\vdots$: other perceptual features

The probability that p is selected at any step can be modeled as:

$$P(\text{select } p \mid \mathbf{v}(p)) = \sigma\left(\mathbf{w}^\top \mathbf{v}(p) + b\right)$$

where σ is a logistic function, and parameters $\mathbf{w}, b$ are learned using ARCTraj logs. More advanced approaches could incorporate spatial structure using attention networks or conditional graphical models.

In summary, modeling selection behavior from ARCTraj provides a rich source of auxiliary knowledge $\mathcal{A}$. This knowledge helps the solver f_θ focus attention, adapt learning over time, and prioritize relevant visual information over candidate regions in the grid when solving new ARC tasks.

D.2 Color-Based Abstraction and Origin Reasoning

To support the few-shot ARC solver f_θ, this section explores how color abstraction patterns observed in ARCTraj can serve as structured auxiliary knowledge $\mathcal{A}$. Color usage in ARC tasks often follows implicit analogies across grid regions, and understanding how humans generalize color origins helps solvers infer symbolic patterns and reduce ambiguity during generation. ARCTraj trajectories provide opportunities to analyze how color decisions are made during problem solving, offering insight into both literal provenance and abstract reasoning over color semantics.

We define four candidate color sources: colors from example input grids $\mathcal{C}_1$, demonstration output grids $\mathcal{C}_2$, test input grids $\mathcal{C}_3$, and the union $\mathcal{C}_{1+2+3}$. Although ARCTraj does not explicitly label color origins, symbolic editing trajectories allow us to approximate preferences by analyzing the timing, sequence, and region of color-modifying actions. As visualized in Fig. 11, human decisions frequently deviate from minimal provenance, suggesting abstraction and non-trivial selection criteria.

Color Provenance Modeling To formalize color source attribution, we define a symbolic color action $a_t = (\mathrm{op}_t, c_t)$, where $c_t \in \{0, \ldots, 9\}$ is the color used. We then introduce a source label function:

$$s(c_t) \in \{\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3, \mathrm{None}\},$$

which assigns each color to a provenance set. Based on ARCTraj, we estimate the conditional distribution:

$$P(s(c_t) \mid \text{task metadata, grid context}),$$

where grid context may include local color frequency, positional priors, or recent interaction history. This distribution models how humans implicitly associate colors with their inferred source, reflecting provenance-sensitive decision-making.

We propose integrating $s(c_t)$ into the DSL as a symbolic argument to enable such reasoning in solvers. For instance, `fill(color)` can be extended to `fill(color, source)`. This lets solvers reason about likely origins during planning or decoding, promoting provenance-consistent color selection. Such representations also facilitate counterfactual analysis, interpretability, and human-aligned generation.

Color Abstraction Literal provenance cannot explain all color behavior, especially when humans generalize based on semantic function. To capture this, we define an abstraction function:

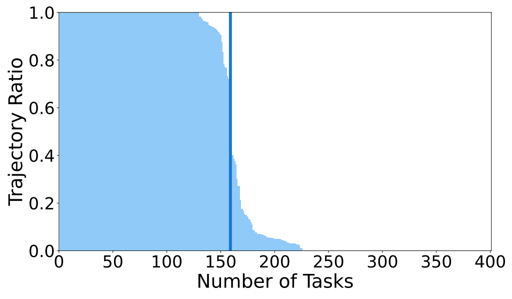
$$\psi : \text{region} \to \text{abstract color group},$$

which maps a selected region to an abstract semantic type (e.g., “border color,'' “main object,'' “mirror fill''). We then model:

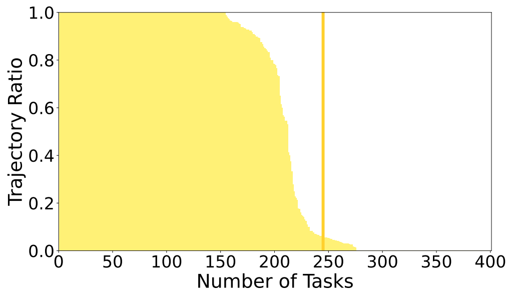
$$P(c_t \mid \psi(\mathrm{region}_t), \mathcal{D}_{\text{demo}}),$$

where $\psi(\mathrm{region}_t)$ informs the likely color based on the function, even if that color was not present in the same literal region.

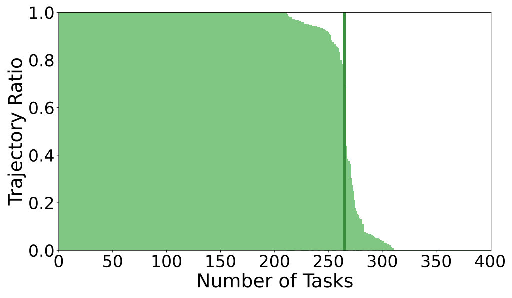
These abstract groupings may be learned from region-level clustering or manually annotated in ARCTraj. Their use allows solvers to generalize symbolic patterns across tasks with novel color distributions or object layouts. Moreover, abstraction-aware prediction encourages high-level reasoning consistency, extending the auxiliary knowledge $\mathcal{A}$ from literal sources to conceptual signals.



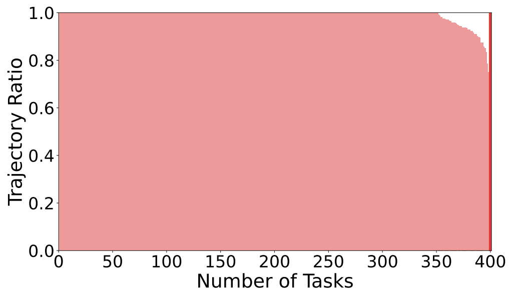
(a) C_1: Colors from example inputs



(b) C_2: Colors from example outputs



(c) C_3: Colors from test input



(d) C_{1+2+3}: Colors from all available sources

Figure 11: Distribution of trajectory ratios across 400 ARC tasks, showing systematic patterns in human color selection that cannot be explained by theoretical color availability alone. Each subplot shows the proportion of human trajectories using colors from specific sources (colored area), while the bold vertical line represents the ground truth number of tasks that require colors from each color source. Across all subplots, four distinct regions emerge based on the intersection of theoretical requirements and actual human behavior: **Region 1** (left of vertical line, colored area): Tasks theoretically requiring colors from the focal color set, and solved using only colors from the focal set. **Region 2** (left of vertical line, white area): Tasks theoretically requiring focal colors, but solved using non-focal colors. **Region 3** (right of vertical line, colored area): Tasks theoretically solvable with the colors from the non-focal color set, but solved using focal colors. **Region 4** (right of vertical line, white area): Tasks theoretically solvable with the colors from the non-focal color set, and solved using non-focal colors. (a) C_1: Colors from example inputs. 159 tasks have the test output color set, a subset of C_1 colors. (b) C_2: Colors from example outputs. The substantial presence of Region 3 indicates systematic deviations from the number 245 tasks. (c) C_3: Colors from test input. The distribution closely aligns with theoretical expectations of 265 tasks. (d) C_{1+2+3}: Colors from all available sources appeared in the task. None of the tasks requires a color that did not appear in the task, which means all 400 tasks are solvable only with the colors that appeared. The contrasting patterns between subplots reveal that human color selection involves complex cognitive strategies that cannot be captured by simple availability-based models, emphasizing the need for explicit color origin tracking in future research.

D.3 Reasoning Strategies and Shared Intentions

To support the few-shot ARC solver f_θ, extracting high-level symbolic reasoning patterns from ARCTraj is valuable. One powerful abstraction is the notion of shared intentions, reusable subgoal structures that reflect how humans decompose and solve tasks. These consist of spatial selections and symbolic operations that frequently co-occur within or across trajectories. This section analyzes how such modular reasoning motifs can be extracted, clustered, and predicted from ARCTraj, enriching auxiliary knowledge $\mathcal{A}$ with intention-level structure.

Clustering Shared Intentions as Reusable Strategy Units Let each intention step be defined as $I_t = (S_t, O_t)$, where S_t is the selected region and O_t is the symbolic operation applied. From ARCTraj, we construct a dataset of observed intentions:

$$\mathcal{I} = \{(S_i, O_i)\}_{i=1}^{N}.$$

The goal is to group these into K clusters:

$$\mathcal{C} = \{C_k \subseteq \mathcal{I}\}_{k=1}^{K},$$

such that intentions in the same cluster share spatial, operational, or temporal characteristics.

The clustering objective is:

$$\min_{\mathcal{C}} \sum_{k=1}^{K} \sum_{(S_i,O_i),(S_j,O_j)\in C_k} d\big((S_i, O_i), (S_j, O_j)\big),$$

where the similarity function $d(\cdot, \cdot)$ considers:

- Spatial overlap between selected regions,
- Semantic similarity between symbolic operations,
- Temporal context within the trajectory.

Each resulting cluster can be viewed as a symbolic production rule or a building block in a *strategy grammar*, encoding reusable steps such as “color fill and replicate” or “rotate and align.” These structures provide modular priors to support interpretable planning, hierarchical reasoning, and few-shot composition of unseen tasks.

Modeling Individual Reasoning Strategies In addition to population-level regularities, ARCTraj includes user identifiers that enable tracking how the same individual solves different tasks. This allows for modeling personalized reasoning styles and decision-making tendencies.

Use the notation $I_t = (S_t, O_t)$ as the intention at step t again, and let $\mathcal{T}^{(u)} = \{\mathcal{I}^{(u,j)}\}_{j=1}^{J_u}$ be the collection of trajectories completed by user u, where each trajectory $\mathcal{I}^{(u,j)} = \{I_1^{(u,j)}, \ldots, I_{T_j}^{(u,j)}\}$ is a sequence of intentions in task j. The objective is to learn a user-conditioned model:

$$P_\psi(I_t \mid \mathbf{x}_t, u)$$

where $\mathbf{x}_t$ encodes the current grid state, prior intentions, and task metadata, and u indexes the user.

This formulation allows the model to account for personal reasoning traits, such as preference for bottom-up visual processing, frequent symmetry-based transformations, or tendency to generalize from small patterns. Learning such user-specific models can:

- Provide personalized suggestions or tutoring aligned with each user’s problem-solving style.
- Enable better simulation or imitation of human trajectories in downstream ARC solvers.
- Reveal inter-task cognitive signatures by analyzing intention patterns shared across tasks.

This personalized modeling augments the auxiliary knowledge $\mathcal{A}$ with human-level variability and individual cognitive profiles, improving the adaptability and interpretability of AI systems learning from ARCTraj.

D.4 Learning from ARCTraj: Applications to Model Training

To support the few-shot ARC solver f_θ, this section reviews how prior studies incorporate ARCTraj into training pipelines. ARCTraj provides structured symbolic trajectories that serve as auxiliary supervision signals $\mathcal{A}$, enhancing sample efficiency, interpretability, and generalization. We summarize existing trials across three modeling paradigms: reinforcement learning, generative modeling, and sequential modeling. Continued research into more effective ways of leveraging auxiliary knowledge could further unlock the potential of ARCTraj as a robust dataset for abstract reasoning tasks.

Reinforcement Learning Agents ARCLE [14] frames ARC solving as a reinforcement learning problem, where symbolic editing actions serve as the action space and grid transformations as the state transitions. The agent interacts with the environment by editing the grid step-by-step until the output matches the target. However, the learning signal in this setting is extremely sparse, as the reward is provided only for exact solutions. Without guidance, exploration often leads to low-probability states with zero reward, making learning slow and unstable.

To address this, ARCLE introduces a two-stage learning process using ARCTraj. First, it uses behavior cloning from human trajectories to pretrain the policy via supervised learning. Each action sequence from ARCTraj serves as a labeled trajectory that allows the agent to mimic expert behavior. This helps the policy avoid unproductive states early in training.

Once initialized, the agent continues training via PPO using environment interactions. A follow-up extension [16] incorporates a World Model using DreamerV3, also trained on ARCTraj. This model learns to predict future latent grid states and actions, enabling planning in latent space. Imitation, RL fine-tuning, and latent dynamics modeling allow the agent to solve novel tasks with improved efficiency and generalization.

Generative Modeling in Latent Space LDCQ [13] proposes a hybrid model that uses latent diffusion and constrained Q-learning. Instead of modeling symbolic actions directly in discrete space, LDCQ first trains an autoencoder on ARCTraj action sequences. This autoencoder learns a continuous latent space where similar symbolic operations lie nearby. The agent can generate smooth and semantically coherent action plans by operating in this latent space.

Diffusion modeling enables stochastic sampling in this space, while constrained Q-learning directs the diffusion process toward regions associated with high returns. ARCTraj data is central to this pipeline: It provides the training corpus for learning semantically meaningful latent embeddings and supplies expert examples with implicit high reward. The result is a generative solver that explores efficiently and generates valid, human-like action sequences.

In parallel, GFlowNet-based methods [10] view ARC solving as learning a flow over action sequences. The model is trained to sample symbolic editing sequences with probability proportional to their reward. However, due to the ample combinatorial space, unguided exploration is ineffective. ARCTraj is used to provide anchor points in this space: expert trajectories with known high rewards are injected into training, helping bias sampling and improving convergence. Flow-matching loss ensures diverse solution coverage, making the model robust to multimodal solution spaces.

Sequential Modeling with Transformers The Decision Transformer (DT) [20] reframes ARC task solving as a return-conditioned sequence generation problem. The agent receives a sequence of 3-element tuples `(return-to-go, state, action)` and learns to predict the following action. ARCTraj supplies the training data: symbolic action sequences, return signals (reward at task completion), and intermediate states. Unlike RL agents, DTs can learn from offline logs, enabling scalable and stable training without environment rollouts.

The DT input is extended to include object-level information, forming 4-element tuples `(return-to-go, state, action, object)`. These richer signals help the model understand spatial context and support object-level reasoning. Further work [12] incorporates intention embeddings derived from ARCTraj logs, resulting in 5-element tuples input structures. These embeddings capture high-level planning intentions (e.g., fill-and-align or mirror-and-reflect), allowing the DT to learn structured and interpretable solution strategies.