Unraveling the ARC Puzzle: Mimicking Human Solutions with Object-Centric Decision Transformer

Jaehyun Park Jaegyun Im Sanha Hwang Mintaek Lim
Sabina Ualibekova Sejin Kim Sundong Kim

Gwangju Institute of Science and Technology

Abstract

Various strategies exist for developing AI that can emulate human-like behavior. One such approach relies on self-training the model on minimal data, as exemplified by systems like Dreamcoder. In contrast, a second approach uses expansive models and large amounts of data to learn intelligent behaviors, a method typified by Large Language Models (LLMs) using transformers. For our exploration of human-like problem-solving, we favor this second approach. Specifically, we are interested in the application of the transformer structure used in LLMs to facilitate learning that emulates human behavior. Building upon this concept, we focus on offline reinforcement learning : Decision Transformer (DT) and Behavior Cloning (BC) . Consequently, we propose a compelling question: could supervised learning techniques that mimic human behavior, using these tools, also solve complex datasets such as the ARC?

Research objectives

- **Objective 1:** Applying a Decision Transformer to assess the potential for solving ARC problems.
- **Objective 2:** Analyzing the impact of incorporating object information into Decision Transformer.

Architecture: Decision Transformer

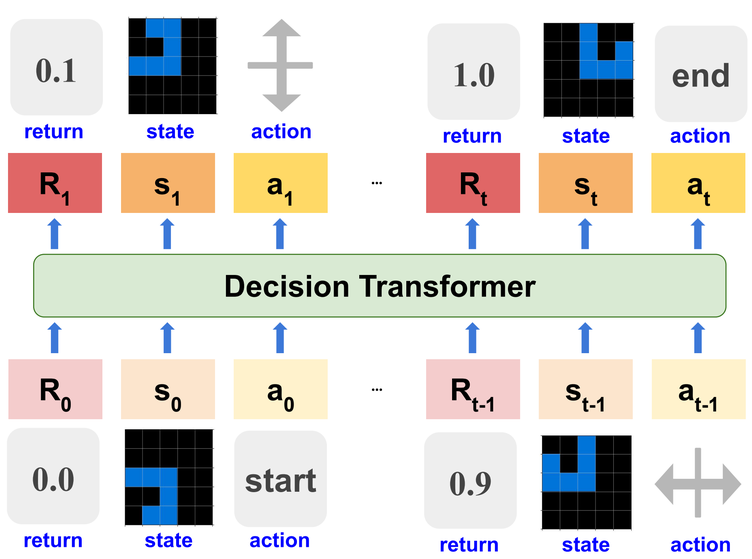

Figure 1. Training a Decision Transformer with Mini-ARC trace. A Decision Transformer utilizes the return-to-go, state, and action at time t as input and generates a prediction of the following time step, $t+1$. The return-to-go is calculated by initiating the state at 0, designating the final state as 1, and partitioning the interval between these states into equal segments. The state is represented using a 5×5 input grid, same with ARC problems. Each of these three inputs are converted to embedding vectors through their corresponding embedding layers.

Data Augmentation

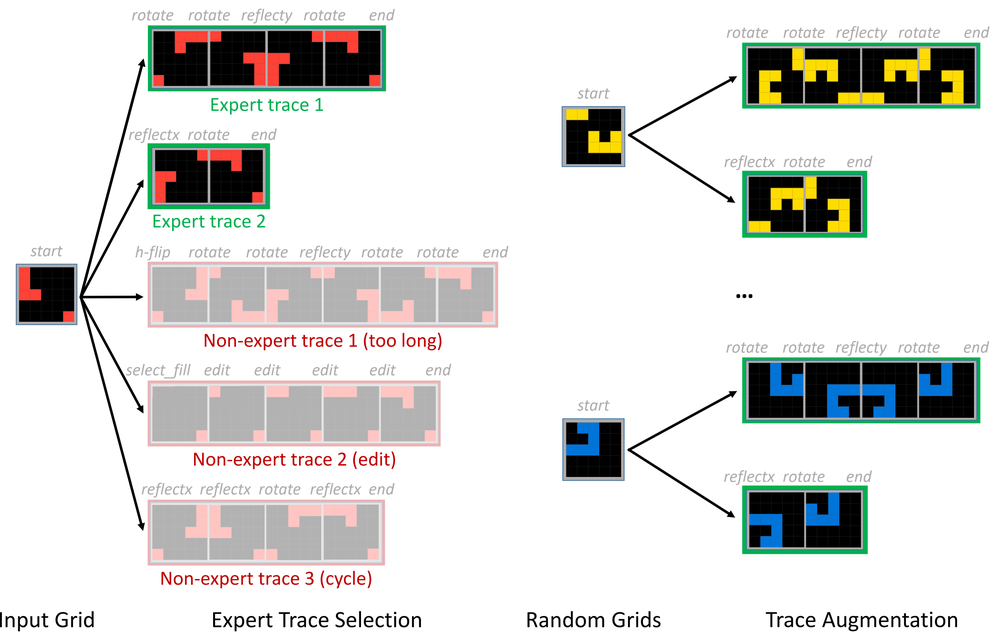

Figure 2. Trace augmentation process for the *diagonal flip* problem: Firstly, from the human solution processes for a particular input grid of the *diagonal flip* problem, we select the expert traces. To be classified as an expert trace, the length of the trace must not be too long, the `edit` operation should not be used, and there must be no cycles in the solution process. The selected expert traces are then applied identically to each randomly generated grid, which is how we perform trace augmentation.

Object Detection: PnP Clustering Algorithm

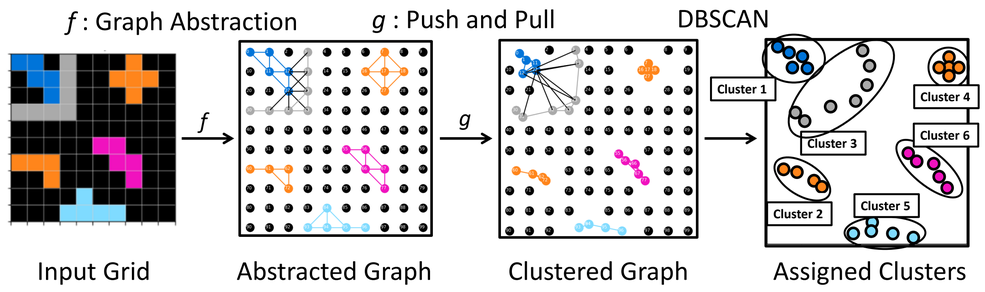

Figure 3. Demonstration of the PnP Clustering Algorithm. Function f abstracts the ARC problem into a graph, then function g applies the push and pull operation to form object clusters. The PnP algorithm enables effective object detection while being computationally efficient compared to other methods.

Decision Transformer with PnP Algorithm

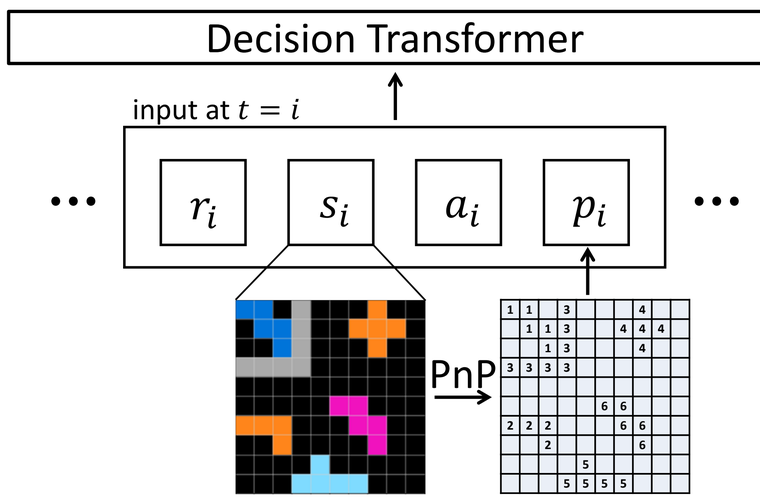

Figure 4. Diagram illustrating a method in which the PnP algorithm provides additional information to the Decision Transformer. For every input at $t = i$, we extract current state from s_i and apply PnP algorithm to generate p_i. The 2-dimensional grid p_i includes object information.

Results

Table 1. Task-wise accuracy of the Decision Transformer and its variations. Noticeable performance improvements are observed when the Decision Transformer is supplemented with additional object information. BC refers to Behavioral Cloning, excluding r_i from the input, while No DT baseline with transformer backbone, does not use any trace information.

	Diagonal Flip	*Tetris*	*Gravity*	*Stretch*
No DT	0.00	0.00	0.00	0.00
BC	30.37 ± 0.31	80.85 ± 0.61	50.01 ± 0.91	64.43 ± 1.12
BC (+PnP)	72.37 ± 0.91	**91.28** ± 0.47	**59.15** ± 0.93	<u>79.26</u> ± 0.58
DT	<u>76.51</u> ± 0.75	71.51 ± 0.66	46.72 ± 1.11	69.98 ± 1.12
DT (+PnP)	**89.96** ± 0.72	<u>83.80</u> ± 0.47	<u>59.00</u> ± 1.00	**86.41** ± 0.61

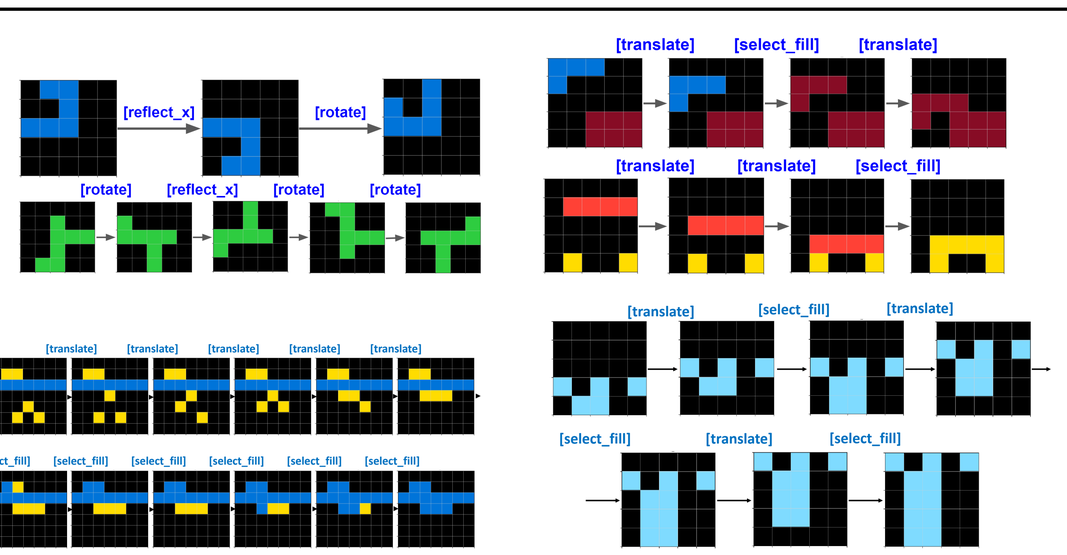

Figure 5. Comparison of the Decision Transformer's predictions for four different tasks (*Diagonal_flip*, *Tetris*, *Gravity*, *Stretch*). Decision Transformer learns the patterns and selects the correct action following the established rules.

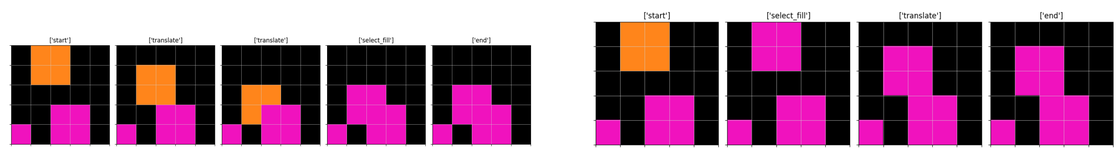

Figure 6. Traces regenerated by Decision Transformer - A comparison of state spaces generated from the standalone DT model (Left), and the DT+PnP model (Right). When augmented with the PnP algorithm, the model identifies and interacts with objects more effectively, avoiding unnecessary downward movement of partial blocks, thereby adhering to the fundamental rules of Tetris.

Discussions

- Applying traditional data augmentation methods in ARC problems can alter the essence of the problem.
- We anticipate that the development of cognition-driven, sequence-preserving augmentation techniques could enhance the learning capacity of the Decision Transformer, enabling it to solve a broader range of problems.
- By specifying objects with the PnP algorithm, state predictions could become more accurate.
- Decision Transformers rely on offline datasets to inform policy training, resulting in a potential adaptability gap when faced with unseen inputs.

Conclusions

- Our employment of the Decision Transformer to replicate human imitation learning demonstrated promising results on the four representative ARC problems, in average of 66.18%, suggesting its applicability to other ARC problems given sufficient data.
- The accuracy significantly improved to an average of 13.61% when we combined the PnP algorithm with the Decision Transformer.
- We anticipate this object-clarifying advantage of the PnP algorithm will be beneficial across various ARC approaches, potentially enhancing problem-solving strategies.

jaehyun00518@gmail.com