Review of Few-shot Learning in ARC Prize 2025

Department of Mathematics, University of Seoul

Jun Beom Park

Few shot Learning

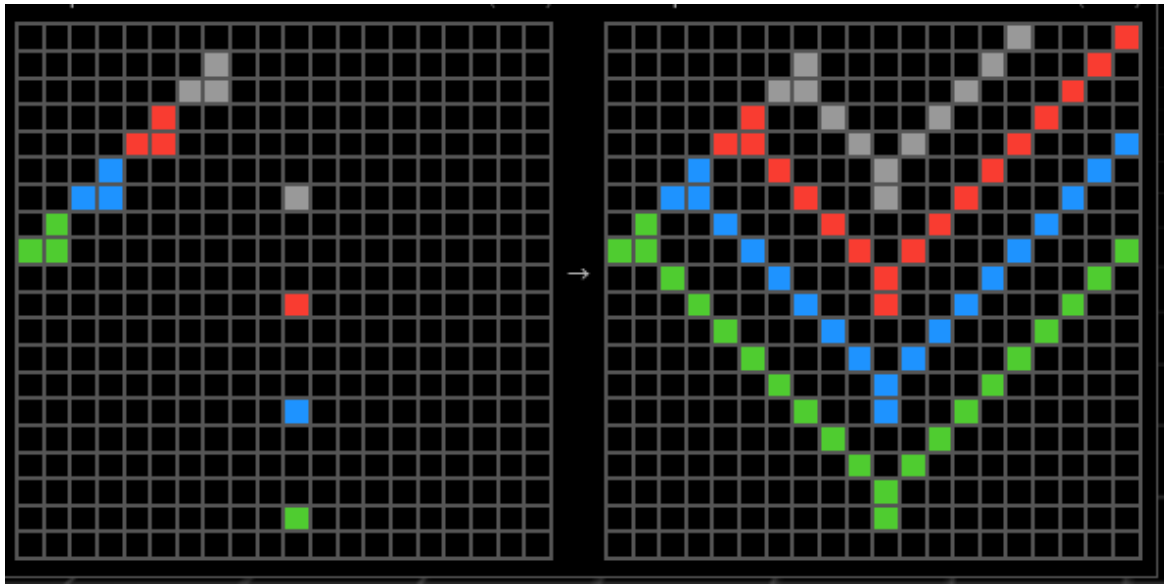
- **Traditional AI:** Needs **a lot** of dog pictures to recognize a dog. (Big Data)
- **Few-Shot AI:** Needs only **3 or 4** examples to understand a new concept

What is ARC?

- **ARC (Abstraction and Reasoning Corpus):** An IQ test for AI.
- **ARC Prize 2025 :**A competition to create an AI capable of novel reasoning

EASY FOR HUMANS, HARD FOR AI

ARC Prize focuses instead on tasks that humans solve effortlessly yet AI finds challenging which highlight fundamental gaps in AI's reasoning and adaptability.



The Challenge: ARC-AGI

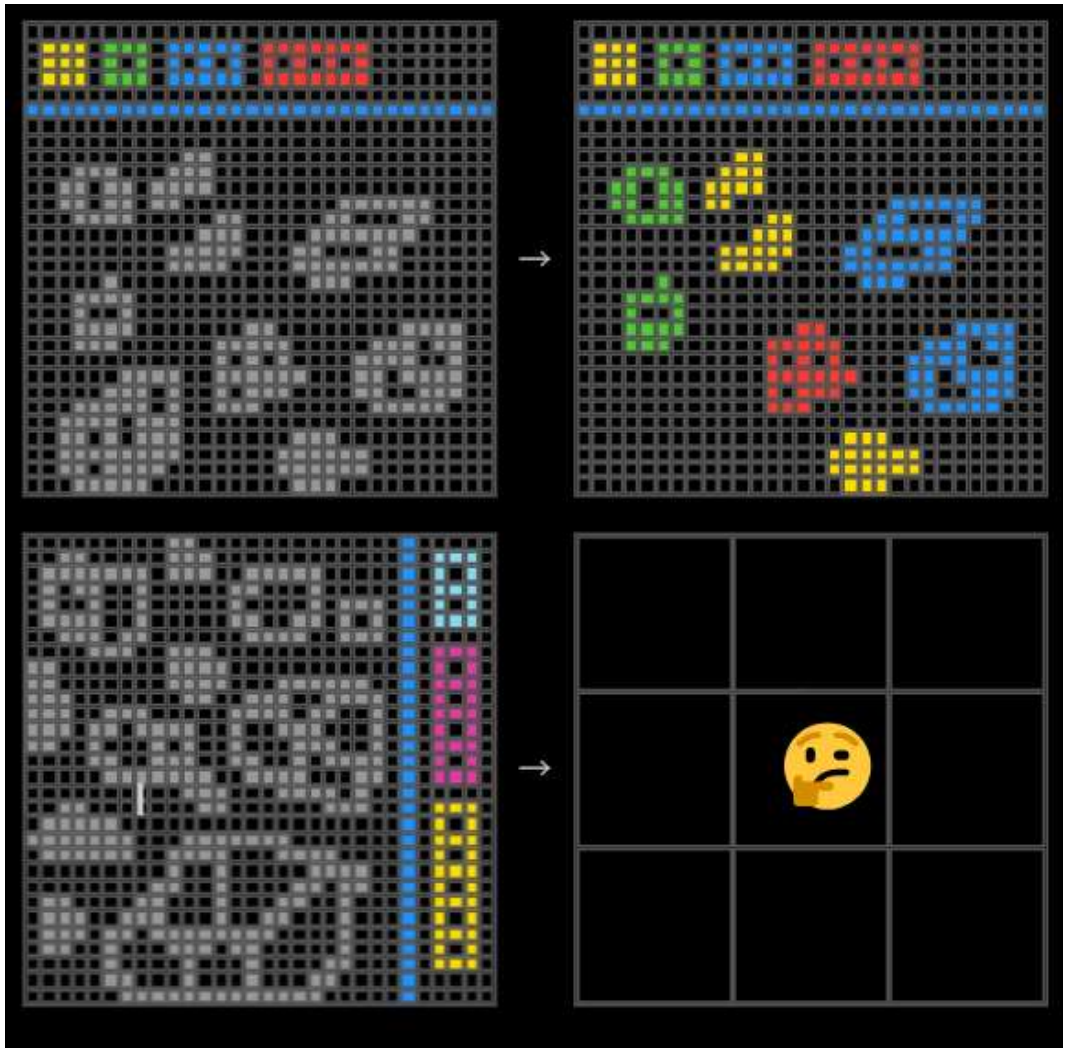
Given

2-5 Example Pairs (Input -> Output)

+1 Test Input

Goal

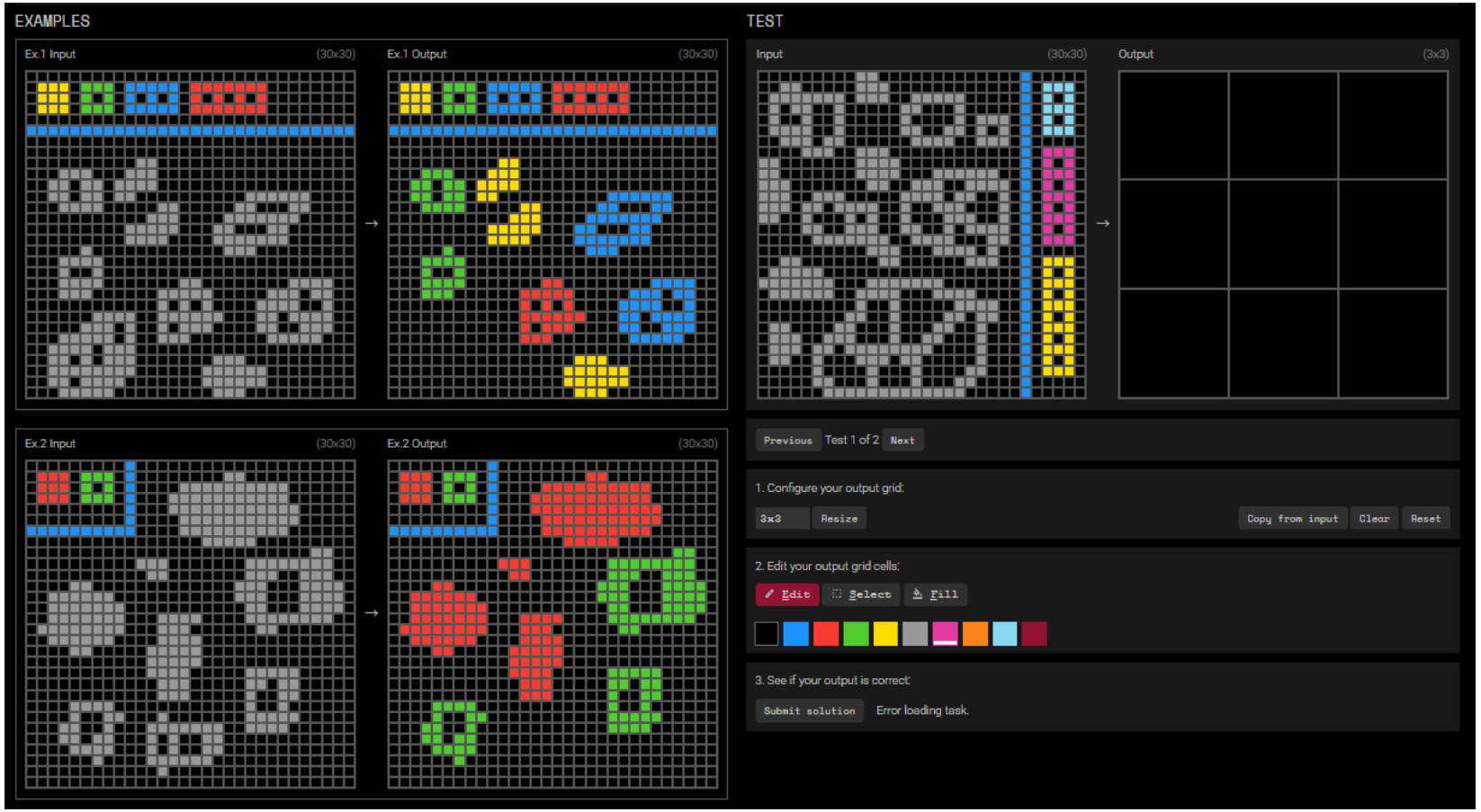
Predict the test Output



Grid Constraints:

10 Colors: Integers 0–9 (Black, Blue, Red, etc.).

Grid Size: Variable dimensions, up to **30x30**



Scoring Rule (Exact Match):

Point: If **every pixel** and the grid dimensions match the ground truth perfectly.

0 Points: If there is even a **single pixel error**.

Requirements & Dataset

Requirements

Using GPU code <= 12 hours run-time

No internet access enabled

External data, freely & publicly available, is allowed, including **pre-trained** models

Dataset

Training set: 1000

Eval set: 120

2020 solutions

Hand-crafted DSL (Domain Specific Language)

Defined ~100 primitive operations (e.g., rotate, crop, recolor, detect_object).

No Deep Learning: Pure algorithmic approach using C++.

Methodology: Brute force search

Found a function f such that f(Input) = Output for all support examples

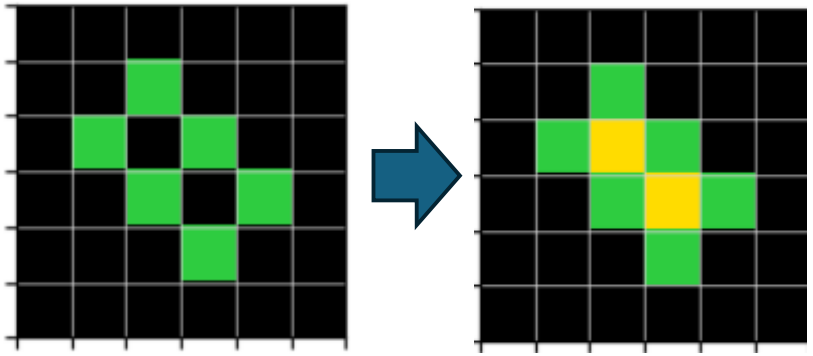
Why did it work?

Strong Inductive Bias: Embedded human knowledge directly into the code.

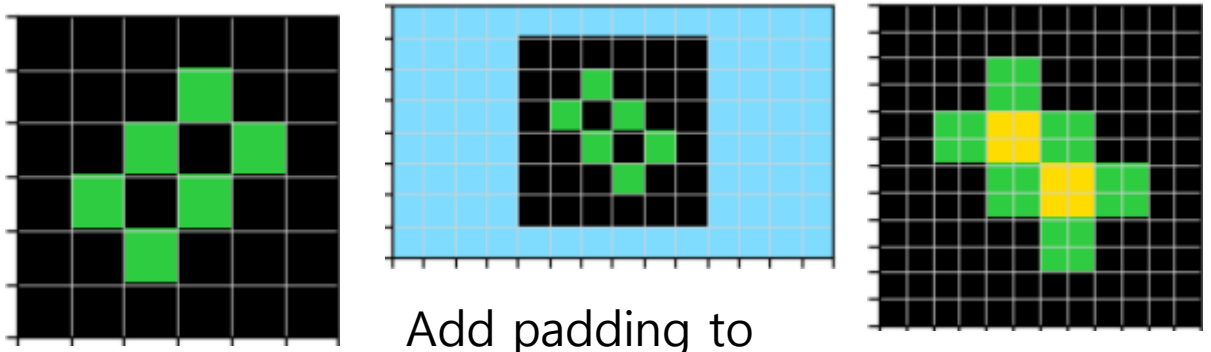
More efficient than Neural Networks for extremely low-data (Few-shot) tasks.

2024 solutions: Data Augmentation

- **Geometric Transformations:** Rotation, Flipping (Horizontal/Vertical), Transposition.
- **Color Permutation :** Randomly mapping colors to different values
- Models need massive train data.



Original Input Original Output



Rotate Input

Add padding to the Input

Upscale the Output

TTFT(Test-Time Fine-Tuning)

Concept of TTFT

Standard models use frozen weights θ during testing.

TTFT updates weights θ -> θ′ specifically for the current task instance using the support set.

➔**Support Set as Training Data:** Treat the few given examples (Input-Output pairs) as a mini-dataset.

➔**Inference:** Use the temporarily updated parameters θ′ to predict the Test input.

TTFT(Test-Time Fine-Tuning)

1. **Augmentation**

We take the given examples and create variants by using data augmentation to build a temporary dataset.

2. **Fine-Tuning**

The model trains on this mini-dataset to adapt its weights to the specific rules of the current task.

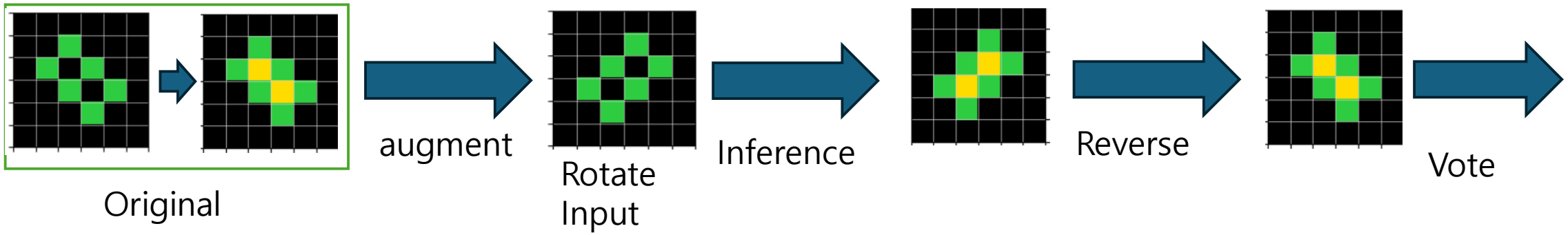
3. **Predict**

The model predicts the answer.

Inference

Using **AIRV** (Augment, Inference, Reverse, and Vote)

1. **Augment**: Apply transformations to the Test Input
2. **Inference**: The model predicts the solution for the modified grid.
3. **Reverse:** Transform the predicted output **back** to the original state (e.g., Rotate -90°).
4. **Vote**



2025 solution

TRM (Tiny Recursive Model)

Concept: A small but smart model (7M params)

Mechanism: The Loop

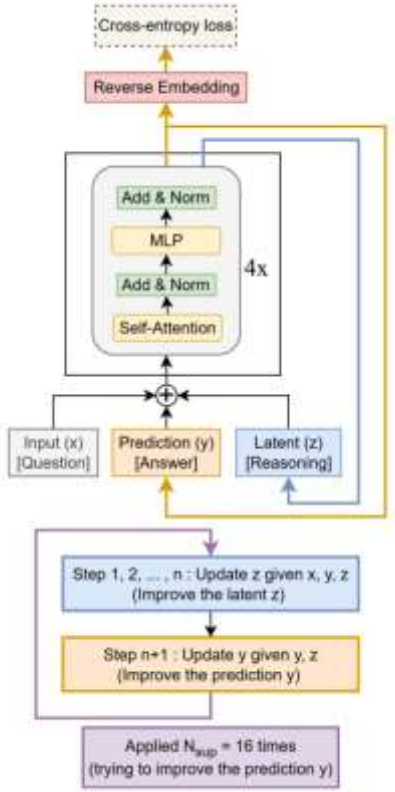
Synthetic Dataset

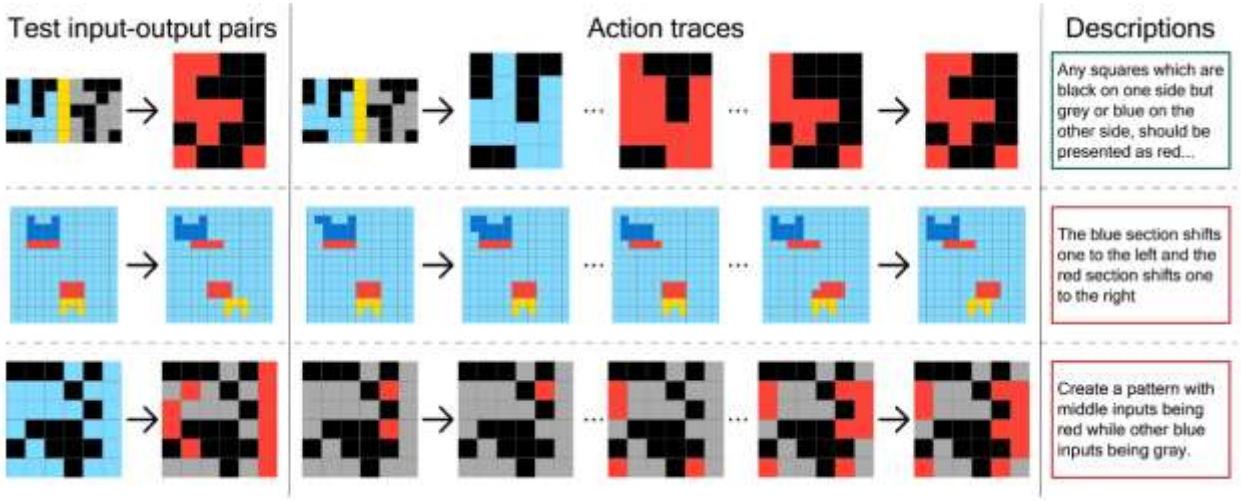
H-ARC:

Describe a rule in text

→LLM (Large Language Model) writes Python code

→Code creates new puzzles.





Implementation

Model:Qwen3-0.6B - Large Language Model

The model was fine-tuned solely using synthetic data generated via **augmentation**.

Evaluated performance using direct inference, bypassing **the Test-Time Fine-Tuning** (TTFT) process.

Prompt

<s>[INST]solve: <example0> input0 123 123 123 output0 123 123 123 </example0>

...

[/INST]

test: [INST] tinput0 12 12 [/INST] toutput0 23 23 </s>

<s>:start sign, [INST]: teacher sign, /:end sign

Analysis

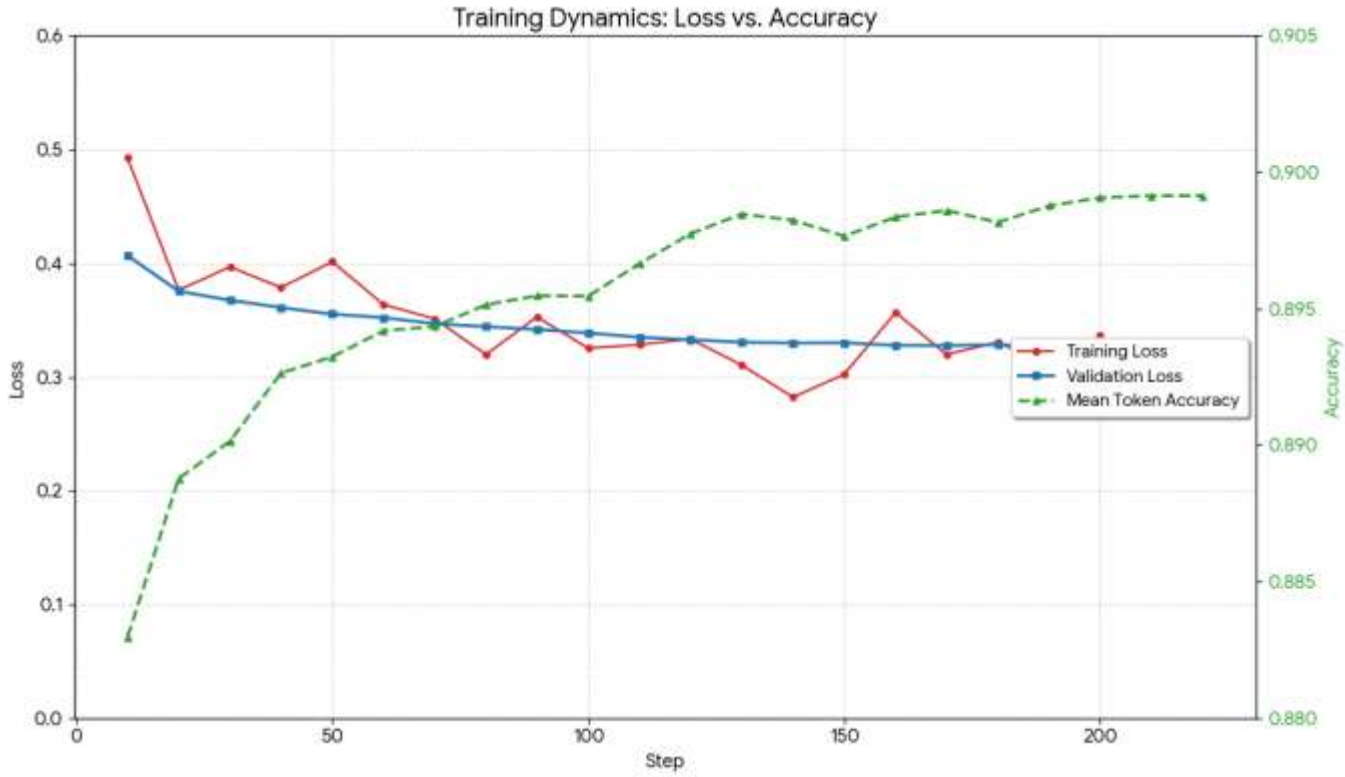
- My Output

[INST]toutput0 323232 787878

<toutput0 70770777077 707

[INST]toutput0 0000000000000

[SOL]000 020 000[/SOL]



- **Training Loss** converges significantly (reaching **0.27**), indicating successful learning of the augmented training set.
- **Validation Loss**, however, **plateaus at ~0.33** after Step 120. The distinct gap between the two curves signals **overfitting** and a lack of generalization to unseen tasks.
- **Mean Token Accuracy** steadily increases, reaching nearly **90%**.

Future work

- Ready for the next ARC-AGI