Team-project: Abstraction and Reasoning Corpus Competition

Chair for Data Science in the Economic and Social Sciences

Supervisor: Maximilian Kreutner

Pablo Hogen
pablo.hogen@students.uni-mannheim.de
University of Mannheim
Mannheim, Germany

Abhay Skaria Thomas
abhay.thomas@students.uni-mannheim.de
University of Mannheim
Mannheim, Germany

Manitarun Sundaram
manitarun.sundaram@students.uni-mannheim.de
University of Mannheim
Mannheim, Germany

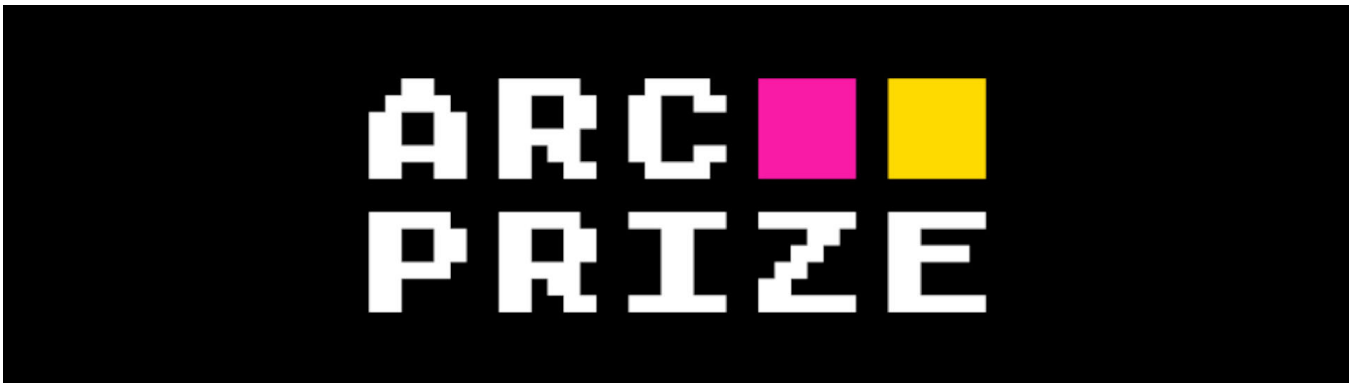


Figure 1: ARC Prize Competition Logo (https://www.ikangai.com/what-is-the-arc-prize-and-why-is-it-important/)

Abstract

The Abstraction and Reasoning Corpus Abstraction and Reasoning Corpus (ARC) Prize Competition aims to evaluate the ability of artificial intelligence Artificial Intelligence (AI) models to perform human-like reasoning and generalization. The competition is grounded in the ARC-Artificial General Intelligence (AGI) benchmark, which challenges models to solve novel reasoning puzzles without prior exposure to the specific tasks. Our team participated in this competition with an approach centered around fine-tuning a large language model, LLaMa-3.1-8B-Instruct, incorporating reasoning-based training methodologies. Initially, we applied instruction tuning and chain-of-thought Chain-of-Thought (CoT) reasoning to encourage step-by-step logical inference. However, due to performance limitations,we designed a series of experiments to test different fine-tuning configurations in which we introduced Baseline without reasoning annotations, candidate sampling and hyperparameter tuning to refine the model's ability to generalize. This report details our methodology, key challenges, findings, results, and potential future improvements in tackling the ARC benchmark.

Keywords

ARC, AGI, Fine-Tuning, LLM, Reasoning, PEFT, CoT

1 Introduction

This report is based on the participation of the authors in the ARC Prize Competition. The competition is publicly accessible and grounded in the ARC-AGI Benchmark. In the context of this benchmark (and therefore in the context of this report) the benchmark creators Chollet et al. [10] defined AGI as follows:

> "... a system capable of efficiently acquiring new skills and solving novel problems for which it was neither explicitly designed nor trained."

Hence, ARC-AGI aims to assess specifically this ability of generalization on novel tasks of State-of-the-Art (SOTA) AI [10]. Although the common definition of AGI is reasoning-centered, it is not fully agreed on how to define it. For instance Legg and Hutter [22] informally defined it as

> "Intelligence measures an agent's ability to achieve goals in a wide range of environments."

which is an outcome-focused interpretation. In this report we will stick to the definition of Chollet et al.
It is far from trivial how the ability of effective adaption to unseen tasks is measured. Chollet et al. decided for reasoning-puzzles which are collected in the dataset ARC-AGI-1 [10]. Since this dataset plays a central role in the benchmark it is discussed in detail in subsection 1.1. Once the tasks the AI is confronted with are understood, subsection 1.2 explains concisely under which general conditions these tasks are to be solved, i.e. the general conditions of the benchmark are clarified. Following, subsection 2.1 deals with reasons why SOTA Large Language Models (LLMs) struggle with solving the ARC-AGI benchmark. Two of the major reasons are

(1) Static Inference
(2) Memorization over Generalization [34]

We expect significant upward leaps in performance w.r.t. the benchmark if these challenges are tackled successfully. This expectation is partially backed by the approaches of the winning teams [23, 1]. These are discussed in detail in section 2.
Section 3 of this report explains our approach of solving the ARC-AGI benchmark in detail. From a birds eye view our approach can be described as fine-tuning the Llama 3.1 8B Instruct LLM [16] via instruction-tuning [6]. We focus on tackling the problem of *Memorization over Generalization* [34]. To be exact, we train the model on verbalizing its strategy in a pre-defined way before applying it and trying to solve the given puzzle. This idea is based on the approach of CoT prompting [32]. Although the paper that introduced CoT focused solely on the art of prompting [32], we explicitly incorporated CoT reasoning into our training data. To be able to do that each of the authors created reasonings to one third of the training dataset via manual labor, this process is described in detail in 1.1.2.
Subsection 3.1 explains the usage of Unsloth for implementing VRAM efficient fine-tuning [17]. Efficiency is not only desired for optimization reasons but also necessary to stay within our boundaries of computational resources. This especially holds for Hyperparameter (HP)-tuning since it is costly time- and computation-wise. Time and computational power are limited due to the usage of bwUniCluster 2.0, which is a joint computer system owned by Baden-Württembergs universities (acknowledgment 7.1). Furthermore, Unsloth does not yet support free to use multi-GPU support [30], which was the second major limiting factor.
Subsection 3.2 touches on the details and results of the grid search HP-tuning implementation. To extend the scope of tested approaches, subsection 3.3 introduces candidate sampling [8] using BigBird-RoBERTa [35]. Since many of the used tools, platforms, techniques, and concepts encountered while implementing the approach were completely new to all the authors, subsection 3.4 concludes the second section by pointing out the key challenges faced during the whole process of training the model.
This concludes the implementation of our approach, leaving us with eight models to evaluate. The experimental setup to evaluate out results is explained in section 4 and the results are reported in section 5. During evaluation the model is confronted with 200 unseen puzzles in the same format as the training tasks (which is elaborated in subsection 1.1). While talking about minor difference in absolute performance, adding reasonings to the training dataset showed a positive impact on the performance of the models during evaluation. Surprisingly the original model with reasoning but without HP-tuning scored the highest accuracy. We believe this is caused by the 80/20 train/test-split performed only for the HP-tuned models. Since this split was not necessary when testing our approach for the first time without HP-tuning the model was trained on 20% more data. We believe that this is the reason for this counterintuitive finding in the results. All findings are discussed in more detail in sections 6 and section 7. The latter discusses alternative approaches and SOTA insights on the performance of OpenAIs o3 model on ARC-AGI tasks [9].

1.1 Enhancing the ARC-AGI-1 Dataset

The ARC-AGI-1 dataset is one cornerstone of ARC-AGI benchmark since it is composed of the challenges which the AI models are

Figure 2: Single Task Example of ARC-AGI-1 https://arcprize.org

trained, evaluated, and finally tested on. Each single task pair has a train and a test part. The train part shows two to ten input/output pairs which all follow the same transformation rule. The test part then shows one to three inputs which the model should produce the respective output to, based on the transformation rule observed from the training pairs. All inputs and outputs are two dimensional nested arrays filled with single digits [10]. To increase the ease of use for humans, the values within the two dimensional grid are commonly visualized as colors (detailed color legend in Appendix B). There is an ongoing discussion about the format of these tasks and if the transformation from color-coded two dimensional grids into nested arrays with single digit values looses information. Therefore many approaches are tested in the digit and the color-coded format [23]. This is out of scope for this report and our approach, but we believe this is an interesting research question for future experiments - if the colors help humans in conceptualizing the transformation rule of these puzzles, why shouldn't it possibly help deep neural networks?
A single task example is shown in Figure 2. On the left half the train input/output pairs are visible which demonstrate the transformation rule aimed to be replicated by the model. In the upper right area of the figure the respective test input is shown. The model is supposed to create the output to that test input, based on the examples seen on the left hand side.
The underlying data structure to that visualization is a two dimensional array wrapped in a JSON-format.

Listing 1: JSON of a Task Example

```
{
  "train": [
    {"input": [[1, 0], [0, 0]],
    "output": [[1, 1], [1, 1]]},
    {"input": [[0, 0], [4, 0]],
    "output": [[4, 4], [4, 4]]},
    {"input": [[0, 0], [6, 0]],
    "output": [[6, 6], [6, 6]]}
  ],
  "test": [
```

```
    {"input": [[0, 0], [0, 8]],
     "output": [[8, 8], [8, 8]]}
  ]
}
```

This is shown in Listing 1 and plays a significant role, since an implicit challenge of the benchmark is not only to understand the transformation rule and apply it properly, furthermore the model is required to represent it syntactically correct to mark a task as successfully solved. Listing 1 is not the JSON translation of figure 2, its only purpose is to demonstrate the general JSON-format of tasks.

1.1.1 The composition of ARC-AGI-1 Dataset. This subsection summarizes the composition of the ARC-AGI-1 dataset concisely and hence allows to clarify which part of the dataset has been enhanced by the authors in 1.1.2.

Table 1: Composition of ARC-AGI-1 Dataset

	Training	Evaluation-1	Evaluation-2
Access Level	Public	Public	Private
No. of Tasks	400	400	100
Difficulty	easy	hard	hard
With Solutions	✓	✓	✗
Usage	Training	Public Score	Final Score

As Table 1 shows, there are 900 benchmark relevant tasks created by manual labor. Therefore the classification made by Chollet et al. into hard and easy tasks are subjective. The training part of the dataset is publicly accessible and commonly used as basis to fine-tune pre-trained models. The public evaluation part of the dataset is also publicly available and defines the score of participants on the public leaderboard. Since high scores can therefore easily be reached by overfitting to the public tasks, the final score is defined by the performance on the unseen evaluation dataset which in theory is free from leakage [10].

1.1.2 Our enhancement - ARC-AGI-1.* To push the Llama 3.1 8B Instruct model [16] towards reasoning we enhance the 400 tasks of the public training dataset introduced in 1.1.1 by manual labored reasonings. These reasonings have a consistent structure and each part is given in natural language

(1) Overall Rule
   - Based on which rules does one get the output from the input?

(2) Dimensions
   - Which implications has the described rule on the dimensions the output grid must have?

(3) Values
   - Which implications has the described rule on the values used in the output?

(4) Changed Values
   - Which implications has the described rule on the entries that have to change their values?

Figure 3: Task example of the public training dataset of ARC-AGI-1. The left third shows the task demonstration pairs, the central third shows the test input, and the right third shows the desired output w.r.t. to the given input.

   - Especially applicable when the starting point of the output creation is copying the input and manipulating it (this is applicable often)

To clarify, an example reasoning follows below which is based on the task example of figure 3. The specific reasoning for that task is:

(1) Overall Rule
   First the input is copied, then we fill for each non-zero value its horizontal and vertical line with the same non-zero value. If two lines cross each other this specific entry gets the value 2 at the intersection.
(2) Dimension
   Therefore the output dimensions are the same as the input dimensions.
(3) Values
   The non-zero values of the input define the values of the added vertical and horizontal lines (except for the case of intersections).
(4) Changed Values
   For each non-zero in the input the whole vertical and horizontal line of that entry will be changed.

Following this strategy the authors created reasoning enhancements for each task in the public training dataset.

1.2 Framework Conditions of AGI

The ARC Prize 2024 imposed strict constraints to ensure models demonstrated genuine reasoning ability rather than relying on memorized patterns or brute-force computation. These framework conditions introduced several challenges, highlighting the limitations of current AI systems in achieving AGI.

1.2.1 ***Compute and Execution Constraints:*** Participants were required to execute their solutions with CPU and GPU within 12 hours on a controlled virtual machine. Unlike standard AI benchmarks that leverage extensive compute, ARC Prize intentionally limited computational power to encourage efficient algorithmic reasoning. However, results showed that higher compute alone did not significantly improve performance, suggesting that AGI progress depends on fundamental algorithmic improvements rather than raw processing power [10].

1.2.2 ***Maximal Runtime Challenge:*** One of the significant challenges in the ARC-AGI competition is the 8-hour maximal runtime

for code execution. Participants are required to execute their models and solutions within this strict time limit on a controlled virtual machine. This constraint is designed to simulate real-world limitations, pushing teams to develop efficient and scalable algorithms that do not rely on extensive computational power or brute-force methods.

1.2.3 ***Transparency and Open-Sourcing:*** To promote reproducibility and transparency, only open-sourced solutions were eligible for prizes. While this condition ensured fairness, it may have discouraged some high-performing participants from fully engaging in the competition [10].

1.2.4 ***Benchmark Target:*** In the ARC-AGI competition, a significant benchmark target is to achieve an 85% accuracy on the private evaluation phase. This private evaluation assesses how well a model generalizes to unseen tasks, which are not included in the training phase. The 85% target serves as a performance threshold to determine how effectively a model can handle abstract reasoning tasks under real-world conditions, free from prior task-specific training. The private evaluation is designed to test models in a setting that closely mirrors how an AGI system would need to perform in unknown, real-world scenarios. Unlike the public evaluation, which might offer more predictable tasks or examples, the private evaluation is more unpredictable, with tasks that challenge the model's generalization ability and problem-solving skills .

2 Related Work

The Abstraction and Reasoning Corpus ARC tests AI models' ability to generalize in a way that is comparable to that of humans by teaching them abstract ideas from a small number of instances. Many strategies, from deep learning-based techniques to program synthesis, have been investigated over time to address ARC challenges. In the sections that follow, we address the main issues in ARC, namely the reasons why large language models LLMhave trouble reaching artificial general intelligence AGI, and we assess other noteworthy methods, such as those used by past ARC competition winners.

2.1 Why LLMs Struggle with AGI

While LLM demonstrate impressive capabilities in natural language tasks, they struggle with key aspects necessary for AGI, particularly in their reliance on static inference and memorization rather than generalization. These limitations prevent them from fully understanding and adapting to new situations in a way that mimics human cognition [21].

2.1.1 ***Static Inference:*** LLM primarily operate through static inference, which means:

- **Limited adaptability**: LLM generate outputs based on learned patterns without adapting in real time to new contexts or environments [4].
- **No real-time learning**: They cannot update or modify their knowledge dynamically as they process new information or experiences [25].
- **Fixed knowledge base**: Once trained, the knowledge of LLM is frozen, which means that they cannot learn from new experiences unless they are re-trained from scratch [31].

2.1.2 ***Memorization Over Generalization:*** LLM often memorizes specific patterns rather than generalizing across a wide range of tasks:

- **Overfitting to training data**: LLM tend to memorize patterns and correlations found in training data, making them prone to overfitting and less capable of generalizing to novel tasks [15].
- **Lack of true understanding**: Memorization-based learning limits their ability to truly "understand" concepts or infer new ideas outside their training data [24].
- **Failure in unseen tasks**: When faced with tasks or scenarios they have not encountered during training, LLM often fail to generalize effectively, leading to incorrect or irrelevant outputs [27].

2.1.3 ***Failure to Perform Complex Reasoning:*** While LLM can mimic human-like responses, they fall short when it comes to:

- **Complex multi-step reasoning**: LLM can struggle to carry out tasks that involve multistep reasoning or require a sequence of logical deductions [11].
- **Planning and decision-making**: LLM do not possess the ability to plan or anticipate the consequences of actions over time, which is necessary for AGI [29].

2.1.4 ***No Conceptual Transfer:*** LLM are often unable to transfer learned concepts from one domain to another:

- **Narrow task-specific learning**: They are good at solving specific problems within the scope of their training data, but they do not transfer knowledge or skills across different domains [6].
- **Inability to infer new contexts**: LLM may not perform well in unfamiliar contexts because they cannot make inferences or connections outside their training data [27].

2.1.5 ***Challenges with Generalization:*** Although LLM can generate outputs based on prior patterns, their generalization abilities are often lacking.

- **Difficulty in handling novel situations**: They are not equipped to handle completely new or unseen scenarios without relying on statistical patterns from previous data [4].
- **Limited contextual understanding**: LLM may not fully grasp the context of complex problems, leading to responses that may seem logical in isolation, but are inappropriate when considering the broader situation [25].

The reliance of LLM on static inference and memorization rather than generalization limits their potential for achieving AGI. To move towards true AGI, models must be able to generalize across a wide range of tasks, adapt in real-time, and perform complex reasoning in dynamic environments, which current LLM struggle to do effectively.

2.2 Top Approaches of ARC Prize Winners:

In the 2024 ARC Prize competition, several teams demonstrated innovative approaches to solving the abstract reasoning challenges posed by the ARC-AGI benchmark. This section outlines the top

three award winning approaches, detailing their methodologies and the performance outcomes of each of them.

- **The ARChitects: Test-Time Training (TTT)** The ARChitects employed TTT as their primary approach to enhance model performance. TTT involves fine-tuning model, during the test phase, adapting it to the specific characteristics of each task. This dynamic adaptation process enables the model to better generalize to new, unseen tasks and refine its output in real time.The ARChitects achieved a score of **53.5%** on the private evaluation set, demonstrating the efficacy of TTT in improving task-solving accuracy for the ARC-AGI challenges. [2].
- **Guillermo Barbadillo: TTT** Guillermo Barbadillo adopted a similar TTT strategy to address the diverse and challenging tasks in the ARC-AGI competition. By applying TTT, Barbadillo's model was able to adapt to each task individually at the test stage, optimizing its performance based on task-specific data. This approach proved effective in improving the model's versatility and ability to solve a wide range of abstract reasoning problems.Barbadillo achieved a score of **40%** on the private evaluation set, securing a competitive position among the top participants. [3].
- **Li et al.: Combining Inductive and Transductive Reasoning** Li et al. introduced an innovative approach that combined inductive reasoning (the process of generalizing from specific examples) with transductive reasoning (the ability to infer outputs for specific inputs without generalization). This hybrid method helped the model to tackle abstract reasoning challenges more effectively by leveraging the strengths of both reasoning techniques. The combination of these two approaches allowed for a more nuanced and accurate handling of the diverse ARC-AGI tasks. Li et al. achieved the highest score in the semi-private evaluation set, underscoring the power of combining inductive and transductive reasoning in abstract reasoning tasks. [14].

These approaches not only contributed to advancing abstract reasoning techniques but also provided valuable insights into the future of AI problem-solving.

3 Fine-Tuning Llama 3.1 8B on Reasoning

The pre-trained Llama 3.1 8B Instruct model [16] was fine-tuned via instruction-tuning [6]. Reinforcement-Learning from Human Feedback (RLHF) [26] has been considered as well but a supervised fine-tuning technique as instruction-tuning seems more natural given the way the ARC-AGI-1 dataset is composed and has been enhanced by the authors.

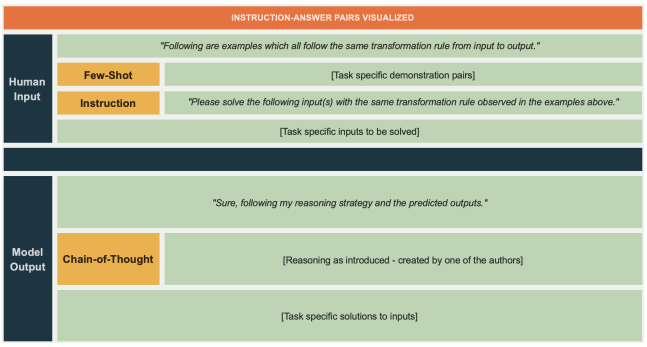


Figure 4: Structure of an Instruction-Answer pair used for the instruction-tuning of the Llama 3.1 8B Instruct model.

Figure 4 shows the implemented structure of an instruction-answer pair the model will be fine-tuned on in a supervised manner. Given that an ARC-AGI-1 task has demonstration pairs by nature (see Subsection 1.1) this results in a few-shot instruction-tuning [6] with encapsulated CoT properties [32]. The few-shot examples are realized by the demonstration pairs, while the push towards a structured CoT approach comes in form of the added reasonings the model is trained on. The major difference to the original introduction of CoT-prompting is that we really train the model on following this reasoning strategy while Wei et al. introduced it as a pure prompting strategy [32]. The idea is to push the model towards this clear reasoning strategy consisting of the overall rule, dimension, values, and changed values (see 1.1.2) since the authors observed these to be the key components defining the solution given the specific puzzle format of the ARC-AGI benchmark. Implicitly we believe that due to the autoregressive nature of LLMs, defining the strategy before executing it helps the model to stay consistent during the production of the final output. This is consistent to what Wei et al. explored when pushing the model towards a CoT-based output [32].

3.1 Efficient Training with Unsloth

The strategy of fine-tuning the Llama 3.1 8B model has been layed out (see Section 3), in this subsection the realization of that strategy is elaborated.
As mentioned in Section 1 the trailblazing limitations of implementing this approach are w.r.t. the resources

(1) Time
(2) Computational Power

The usage of Unsloth helps mainly due to its asynchronous offloading of activation tensors from VRAM to the system RAM [17]. This technique allows the realization of fine-tuning with larger batch-sizes. Although full fine-tuning (i.e. retraining all pre-trained parameters) has been not feasible due to VRAM limitations, Unsloth allowed us to implement Low-Rank Adaptation (LoRA) [19] without further quantization (i.e. 16-bit parameter precision instead of the 4-bit precision common for Quantized Low-Rank Adaptation (QLoRA) [13]). Therefore only $41,943,040$ parameters of the 8B possible are retrained during fine-tuning. This is roughly equal to 0.52% of all accessible parameters. Further rank stabilization is

applied resulting in rank-stabilized Low-Rank Adaptation (rsLoRA) [20]. The enhancement of rank stabilization has proven to help stabilize the training process of fine-tuning LLMs especially with increasing rank of the adapter matrices [20].
The enhanced dataset 1.1.2 is transformed into the instruction answer pairs by using the a pre-defined chat template of the Unsloth documentation in ShareGPT style [17]. Since our approach is not reliant on a multi-turn conversation an Alpaca like chat template would have been more suitable [7]. This has been realized too late but the effect of that inconvenience is expected to be minor, since using a single turn conversation in the ShareGPT style still results in a valid Supervised Fine-Tuning (SFT). The actual SFT is realized via TRL which seemlessly integrates with Unsloth [17, 33]. Because of this natural integration Unsloth is a common choice for Parameter-Efficient Fine-Tuning (PEFT). Because of the hard deadlines we committed to the approach of HP-tuning with rsLoRA, but there are certainly many other possibilites worth exploring. For instance we are still interested in exploring the effect on the performance of the trade-off between our approach (rsLoRA with moderate HP-tuning) and applying QLoRA but tuning HPs excessively.

3.2 Hyperparameter Tuning via Grid Search

Following parameters are HP-tuned via grid search [5] with the respective values in brackets:

- Learning Rate: $[5e-5, 5e-4, 1e-4]$
- Batch Size: $[4, 8, 16]$
- Epochs: $[1, 3, 5]$
- Weight Decay: $[0.01, 0.1]$

Resulting in a total of 54 HP-settings. Since compute time was limited to maximally 4 hours due to the runtime restrictions of the bwUniCluster2.0 the HP-tuning process was regularly interrupted by a dying server. This has been bypassed by using the excel which the results has been logged to also as log file which indicates which HP settings the model been trained on already. Hence, it was possible to run the process in distributed sessions of 4 hour computations without loss of information. The other fixed HPs are:

- rsLoRA
  - Rank of 16
  - $\alpha = 16$
  - No dropout
  - Target Modules: Q,K,V-weight matrices, linear projection weight matrices, attention output weight matrices
- Training
  - Linear Learning Rate Scheduler
  - 10 Warmup Steps
  - Adamw_8bit optimizer

The HP-tuning is performed for the fine-tuning of Llama 3.1 8B Instruct on two different datasets and of each HP-tuning approach the best three models haven been picked for further evaluation (see 5). The two different datasets are the original ARC-AGI-1 dataset (1.1.1) and the enhanced ARC-AGI-1* (1.1.2) dataset. The fine-tuned model which has been only trained on the original dataset is the baseline to our approach, such that we can test if enhancing the dataset by manual labored reasonings has any positive impact on the resulting reasoning ability of the fine-tuned model. We believe that putting equal effort into fine-tuning the baseline is needed to create a baseline which is able to expose if our approach is creating real value. It has been observed in Machine Learning (ML) that many baselines are not as heavily fine-tuned as the presented approach often resulting into pseudo-baselines [28]. We are convinced strong baselines help sorting out the real value of an approach.
The results of the HP-tuning show that the main factor influencing

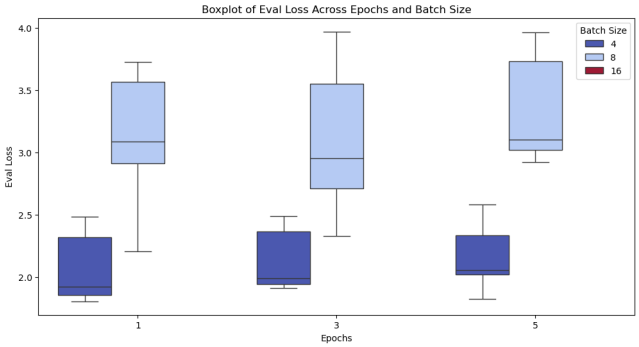


Figure 5: Boxplot of eval loss w.r.t. epochs and batch Size.

the evaluation loss is the batch size (see figure 6). The boxplot of

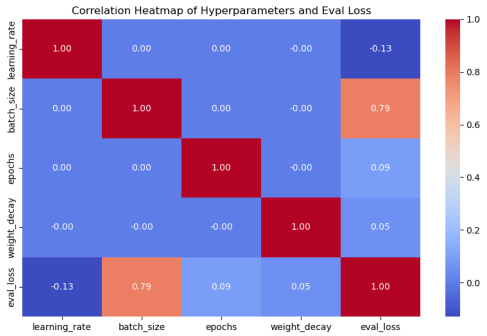


Figure 6: Correlation heatmap of eval loss and HPs.

figure 5 show that a small batch size minimizes evaluation loss on average for training with reasonings. Same holds for training the baseline without reasoning (see appendix section D). Increase in evaluation loss is observable when increasing the batch size. This shows that generalization gets worse which might be due to overfitting to the seen data. All other tuned HPs show almost no correlation with the evaluation loss. The top three models have been uploaded to Hugging Face (HF) and are linked in appendix section C. Further all HP combinations with a batch size of 16 resulted in an Out Of Memory (OOM)-Error, and therefore did not create any evaluation loss they could have been assessed on. This results in a total of six HP-tuned models and one original try where we just used basic HP-settings and didn't tune any of them (see 5). To be exact, the initial unfixed HPs of the first try are:

- Learning Rate: $3e-4$
- Batch Size: 8
- Epochs: 1
- Weight Decay: 0.01

3.3 Candidate Sampling

Candidate Sampling was introduced as an alternative approach to improve model robustness by training it to rank multiple candidate solutions instead of producing a single deterministic output. Instead of learning a direct input-output mapping, the model was trained to assign confidence scores to several possible solutions and select the most likely one.

For this, we used **BigBird-RoBERTa**, a transformer model optimized for long-sequence processing. Unlike traditional transformers, BigBird employs a sparse attention mechanism, allowing it to efficiently handle long input-output sequences in ARC tasks. The dataset was derived from ARC-AGI, where each test input was paired with five candidate solutions: one correct and four incorrect variations. Noise was introduced into incorrect solutions by randomly modifying some elements, ensuring varying levels of deviation from the correct solution. Confidence scores were assigned using a predefined function, which automatically evaluated the level of randomness introduced and assigned values of **1, 0.8, 0.6, 0.4, or 0.2** based on the severity of modification.

Before training, all sequences were tokenized using BigBird's long-sequence tokenization, allowing the model to process up to **4096 tokens**. The model was fine-tuned using a ranking-based objective where it assigned confidence scores to each candidate solution. The training setup included **MSE loss** with a learning rate of **5e-5**, batch size of **2**, and a total of **3 epochs**. Each batch contained a tokenized test input and five candidate solutions, requiring the model to rank them in order of confidence.

Evaluation was conducted by checking whether the correct solution consistently received the highest confidence score. However, initial results indicated that the model struggled with clear ranking differentiation, often assigning nearly uniform confidence scores across all candidates (e.g., 0.57 for each). This suggested that the model was not effectively distinguishing subtle differences between correct and incorrect solutions. To address this, we experimented with increasing the number of training epochs and adjusting the loss function. By extending training duration and refining the ranking objective, the model demonstrated improved confidence differentiation, resulting in a measurable increase in accuracy.

Future improvements include refining the negative sampling process by introducing stronger perturbations to incorrect solutions, experimenting with alternative loss functions such as contrastive loss or pairwise ranking loss, and further HP tuning to enhance ranking precision. While initial results were inconclusive, this approach presents a promising direction for improving model decision-making in ARC-style reasoning tasks.

3.4 Challenges of Fine-Tuning and Optimization

Since this is the first time engaging in fine-tuning a pre-trained model many unexpected challenges have been faced during the process of fine-tuning and optimizing HPs. The most trailblazing challenges are the limitations of compute and time resources (especially VRAM). This led to many failed approaches (full fine-tuning, too large context windows, too large batch sizes) due to OOM-Errors until getting to an approach that runs smoothly while not sacrificing too much precision and performance due to quantization. Further since the competition was time critical the accessibility of bwUniCluster2.0 was another stumbling block. It happened very frequently that the bwUniCluster2.0 did not have the capacity to spawn another server which one can work on. This has been worked around by implementing primitive excel files which also worked as log files during HP-tuning. Still these challenges have been appreciated since we believe that time and compute resource restrictions are relevant right now and will stay relevant in the future [18]. Additionally there have been challenges which cannot be foreseen, e.g. reaching the university-sided file-limit on the bwUniCluster2.0 in a critical stage of our project. The consequence was that it was not possible to setup a conda environment on the cluster. This seems not crucial, but at that time setting up a stable environment for Unsloth via pip was extremely cumbersome. This changes drastically from our experience when using conda and following the installation guide [12].

4 Experiments

4.1 Evaluation of the Original Baseline

The initial baseline model was fine-tuned on the LLaMA-3.1-8B_instruct model using 400 manually curated examples, each paired with a reasoning annotation. The goal was to explicitly train the model to understand and generalize patterns in ARC tasks. This model was evaluated on an unseen dataset of 200 examples, distinct from the dataset used in subsequent experiments, achieving an accuracy of **6.04%**.

A detailed examination of its outputs revealed that in cases where the model generated accurate predictions, the accompanying reasoning was typically aligned with human intuition. However, some examples demonstrated partial success, where the reasoning was logical but contained minor prediction errors. This prompted further investigation into whether fine-tuning strategies, reasoning annotations, or training dynamics played a more significant role in influencing accuracy.

4.2 Experimental Variations

Following the baseline evaluation, we designed a series of experiments to test different fine-tuning configurations:

- **Baselines Without Reasoning**: Fine-tuned models trained without explicit reasoning annotations to assess their impact in generalization.
- **HP-Tuned Models**: Training involved different learning rates, weight decay values, and epoch lengths, all while maintaining reasoning annotations.
- **Candidate Sampling**: This technique introduced noise into correct solutions and trained the model to select the output it was most confident in, aiming to improve robustness.

All models were evaluated on the same dataset of 200 unseen ARC examples.

4.3 Inference and Evaluation Process

Each model was tested using a standardized prompt format that included a set of example input-output pairs followed by a test input matrix for which the model had to generate an output. Given that the models often produced verbose outputs, a structured extraction mechanism was employed to isolate the predicted matrix.

To calculate accuracy, only well-formed matrices were considered valid. Invalid outputs, such as incomplete or non-structured matrices, were excluded from accuracy calculations.

5 Results

The accuracy results from these experiments are summarized in Table 2. The exclusion of invalid outputs ensures that the reported results reflect meaningful model predictions.

Model Configuration	Accuracy (%)
Original Baseline (with reasoning)	**6.04**
Baseline w/o Reasoning (epochs=1, lr=0.0005, wd=0.1, batch=4)	1.43
Baseline w/o Reasoning (epochs=5, lr=0.0005, wd=0.1, batch=4)	2.58
Baseline w/o Reasoning (epochs=1, lr=0.0005, wd=0.01, batch=4)	2.65
HP-Tuned (epochs=1, lr=5e-05, wd=0.1, batch=4)	3.87
HP-Tuned (epochs=5, lr=0.0005, wd=0.1, batch=4)	**4.35**
HP-Tuned (epochs=1, lr=5e-05, wd=0.01, batch=4)	2.66
Candidate Sampling	**3.30**

Table 2: Accuracy of Experimental Models (excluding invalid outputs).

5.1 Observations

- **Explicit reasoning annotations led to higher accuracy**, as demonstrated by the drop in performance in models trained without reasoning (down to 1.43%).
- **HP tuning helped models trained with reasoning perform better**, with the best configuration achieving **4.35% accuracy**. However, despite these optimizations, HP-tuned models did not surpass the original baseline accuracy of **6.04%**.
- **Candidate Sampling showed promise**, achieving **3.30% accuracy**, suggesting that forcing models to rank multiple possible outputs improved robustness.
- **Impact of Training Data Reduction in HP Tuning:** An unexpected finding was that the original baseline, which underwent no HP tuning, outperformed all HP-tuned models. In our HP tuning experiments, we applied an **80-20 training-test split**, reducing the number of training examples available for learning. This reduction may have limited the model's exposure to a sufficient variety of patterns and transformations, affecting its ability to generalize effectively.

  Since ARC tasks require **pattern recognition, abstraction, and concept generalization** across a diverse range of examples, a model's ability to learn effective representations is directly influenced by the variety and quantity of training data available. ARC tasks are particularly challenging because they often require **few-shot generalization**, inferring unseen rules from limited examples. When the training set is restricted, the model sees a smaller subset of potential transformations, reducing its ability to generalize across novel test cases.

  Our findings suggest that while HP tuning optimized model parameters within the available dataset, it did not compensate for the **loss of knowledge diversity** that would have been retained if the full dataset had been used. As a result, despite potential improvements in parameter optimization, the model lacked a sufficiently diverse foundation of learned transformations, weakening its generalization ability.

  This suggests that for reasoning-intensive tasks like ARC, the **breadth of training data** is just as crucial as fine-tuning HPs. Reducing training data may inadvertently constrain the model's ability to discover and apply abstract rules beyond its training distribution. Future work should explore whether **applying HP tuning on a larger dataset rather than reducing the training size could yield more significant improvements**. Additionally, alternative approaches such as **progressive data augmentation** or **transfer learning from broader reasoning datasets** could be explored to mitigate the negative effects of reduced training size while still leveraging the benefits of HP tuning.

6 Conclusion

6.1 Key Takeaways

This work attempted to fine-tune large-scale LLMs toward an intuitive ARC-solving approach, with a focus on explicit reasoning, HP tuning, and candidate selection. While the accuracy results did not reach the levels of SOTA solutions, the experiments provided valuable insights into the broader challenges of generalization in machine reasoning.

Despite the immense capabilities of modern LLMs, these models struggled with fundamental abstraction and pattern recognition in ARC tasks, highlighting key limitations in current AI approaches. The observed performance gaps offered a clearer perspective on where AGI-like reasoning remains an open challenge, emphasizing that even large-scale models that excel at natural language understanding can falter on structured reasoning tasks.

The best-performing models, with HP tuning (4.35%) and candidate sampling (3.30%), demonstrated measurable improvements over the baseline models(without reasonings), yet explicit reasoning annotations remained crucial for higher accuracy. More importantly, this study provided an opportunity to deeply analyze the entire spectrum of where generalization is lacking, offering valuable lessons on the constraints of LLMs in solving abstract reasoning tasks. These findings not only refined our understanding of model limitations but also paves the way for exploring more effective strategies to bridge the gap between statistical learning and true conceptual abstraction.

6.2 Broader Implications for AGI

The ARC challenge serves as a crucial benchmark in the pursuit of AGI, requiring an AI system to exhibit abstraction, generalization which are key traits of human-like intelligence. Unlike traditional

AI benchmarks that focus on well-defined tasks, ARC demands models infer novel transformations from minimal examples, similar to human cognitive generalization.

In many ways, ARC acts as a guiding pathway toward AGI, highlighting the limitations of current AI systems, which rely on pattern recognition rather than true reasoning. This exposes a fundamental gap between specialized AI models and broader general intelligence. Efforts to improve AI performance on ARC contribute to solving broader challenges in developing models capable of reasoning beyond their training distributions.

Advancements in test-time training, hybrid neural-symbolic architectures, and meta-learning demonstrate that progress on ARC is not just about improving task-specific performance but about uncovering principles for more adaptive and robust AI systems. By refining approaches to solving ARC, researchers are addressing core challenges in AGI, working toward models that can think abstractly, adapt dynamically, and generalize across domains.

Despite significant progress in natural language understanding, reasoning-intensive tasks like ARC continue to expose the limitations of even SOTA AI models. The ARC challenge serves as both a diagnostic tool and a roadmap for progress, identifying gaps in current methodologies while guiding research toward more generalizable intelligence.

Ultimately, solving ARC is not just about achieving higher benchmark accuracy, it is about ensuring that AI research progresses toward AGI. Insights gained from ARC are shaping next-generation methodologies, bridging the gap between specialized intelligence and truly generalizable, human-like reasoning.

6.3 Future Work

Moving forward, several promising directions could be explored to enhance model performance on ARC tasks and improve generalization capabilities.

Expanding the training data remains a fundamental challenge due to the limited size of the ARC dataset. Generating additional reasoning-rich examples through **data augmentation techniques** could help address this limitation. By introducing controlled transformations, synthetic variations of ARC tasks can be created, thereby increasing the diversity of training examples without deviating from the core reasoning principles.

Test-Time Training (TTT) is another avenue that has shown promising results in improving model adaptability. By refining model weights during inference using a loss function applied to test inputs, TTT enables models to generalize better to novel ARC tasks. Recent research has demonstrated that applying test-time optimization significantly enhances reasoning capabilities and adaptation to unseen problems.

Another direction involves utilizing **LLMs in a multi-agent framework**. Rather than treating ARC tasks as direct sequence-to-sequence learning problems, LLMs can be used as collaborative agents that decompose problems into modular steps. By converting input images into structured abstraction spaces and allowing multiple expert agents to contribute to a reasoning pipeline, this method could enhance decision-making and improve model robustness.

Each of these advancements holds the potential to meaningfully contribute to solving ARC tasks more efficiently, furthering progress toward **AGI-level abstract reasoning**.

7 Discussion

7.1 OpenAI's o3 Model: Advancing AI Reasoning

OpenAI's **o3 model** represents a major leap in AI reasoning, achieving an **87.5% success rate** on the ARC-AGI benchmark, significantly surpassing prior models. Unlike previous GPT models, o3 incorporates a **simulated reasoning (SR)** mechanism, allowing it to internally generate and evaluate multiple reasoning paths before committing to an answer. This architecture enables a *CoT* approach that mimics human problem-solving by iterating through possible solutions, backtracking, and refining its reasoning dynamically.

Key Architectural Innovations:

- **Internal CoT Search:** Unlike GPT-4o, which primarily predicts tokens sequentially, o3 internally searches through multiple reasoning pathways before producing a response.
- **Extended Context Window:** Supports up to **200k tokens**, allowing it to consider large problem-solving contexts.
- **Monte Carlo Tree Search-Like Mechanism:** Enables o3 to explore multiple solution candidates, akin to AlphaGo's decision-making strategy.
- **RLHF for Reasoning:** O3 was trained to "think before answering" through large-scale RLHF, optimizing reasoning strategies across a diverse set of problems.

Performance on the ARC Challenge: Traditional models like GPT-4 and GPT-4o struggled with ARC, achieving less than **5% accuracy**. In contrast, o3 dynamically **searches for abstract solutions**, identifying transformation rules and synthesizing novel problem-solving strategies, enabling it to surpass human-level performance. This suggests that structured internal reasoning—not just more parameters or training data—is the key to solving complex generalization tasks.

Comparison with Previous Models: GPT-4o excelled in general knowledge and conversational AI but lacked default multi-step reasoning. O3, by contrast, engages in deep reasoning by default, making it significantly better at **mathematical proofs, programming, logical puzzles, and multi-step decision-making**. While GPT-4o relied on pattern recognition, o3 is optimized for **deliberative thinking**.

Future Implications: The success of o3 highlights the importance of integrating **active reasoning** into AI models rather than relying purely on statistical learning. Although o3 is not AGI, it represents a shift toward models that can **plan, reflect, and adapt dynamically**, pushing the boundaries of AI reasoning capabilities.

Acknowledgments

The authors acknowledge support by the state of Baden-Württemberg through bwHPC. We express our gratitude to the computational resources and technical assistance provided, which were essential for conducting large-scale model training and experimentation. We also appreciate the insightful discussions and constructive feedback from Maximilian Kreutner which helped refine our approach.

References

[1] Ekin Akyürek, Mehul Damani, Linlu Qiu, Han Guo, Yoon Kim, and Jacob Andreas. 2024. The surprising effectiveness of test-time training for abstract reasoning. (2024). https://arxiv.org/abs/2411.07279 arXiv: 2411.07279 [cs.AI].

[2] The ARChitects. 2024. The llm architect: solving arc-agi is a matter of perspective. *ARC Prize Proceedings*. https://github.com/da-fr/arc-prize-2024/blob/main/the_architects.pdf.

[3] Guillermo Barbadillo. 2024. Omni-arc. *ARC Prize Proceedings*. https://ironbar.github.io/arc24/05_Solution_Summary/#omni-arc-training-a-single-model-to-do-multiple-arc-related-tasks.

[4] Yoshua Bengio. 2013. Learning deep architectures for ai. *Foundations and Trends® in Machine Learning*. https://arxiv.org/abs/1206.5538.

[5] Yoshua Bengio. 2012. Practical recommendations for gradient-based training of deep architectures. *Neural networks: Tricks of the trade*. https://arxiv.org/abs/1206.5533.

[6] Tom Brown et al. 2020. Language models are few-shot learners. *Advances in Neural Information Processing Systems*. https://arxiv.org/abs/2005.14165.

[7] 2025. Chat templates | unsloth documentation. (Feb. 16, 2025). Retrieved Mar. 11, 2025 from https://docs.unsloth.ai/basics/chat-templates.

[8] Wenqing Chen, Weicheng Wang, Zhixuan Chu, Kui Ren, Zibin Zheng, and Zhichao Lu. 2024. Self-para-consistency: improving reasoning tasks at low cost for large language models. In *Findings of the Association for Computational Linguistics: ACL 2024*. Lun-Wei Ku, Andre Martins, and Vivek Srikumar, (Eds.) Association for Computational Linguistics, Bangkok, Thailand, (Aug. 2024). DOI: 10.18653/v1/2024.findings-acl.842.

[9] François Chollet. [n. d.] OpenAI o3 breakthrough high score on ARC-AGI-pub. ARC Prize. Retrieved Mar. 16, 2025 from https://arcprize.org/blog/oai-o3-pub-breakthrough.

[10] Francois Chollet, Mike Knoop, Gregory Kamradt, and Bryan Landers. 2025. ARC prize 2024: technical report. (Jan. 8, 2025). arXiv: 2412.04604[cs]. DOI: 10.48550/arXiv.2412.04604.

[11] Peter Clark, Tim Rocktäschel, Edward Grefenstette, Karl Moritz Hermann, Phil Blunsom, Stephen Pulman, and Sebastian Riedel. 2020. Combining symbolic and neural approaches to commonsense reasoning. *arXiv preprint arXiv:2005.10188*. https://arxiv.org/abs/2005.10188.

[12] 2025. Conda install | unsloth documentation. (Mar. 3, 2025). Retrieved Mar. 16, 2025 from https://docs.unsloth.ai/get-started/installing-+-updating/conda-install.

[13] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: efficient finetuning of quantized llms. *Advances in neural information processing systems*. https://arxiv.org/abs/2305.14314.

[14] Li et al. 2024. Combining induction and transduction for abstract reasoning. *ARC Prize Proceedings*. https://arxiv.org/abs/2411.02272.

[15] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press. https://www.deeplearningbook.org/.

[16] Aaron Grattafiori et al. 2024. The llama 3 herd of models. (Nov. 23, 2024). arXiv: 2407.21783[cs]. DOI: 10.48550/arXiv.2407.21783.

[17] [SW] Daniel Han, Michael Han, and Unsloth team, Unsloth 2023. URL: http://github.com/unslothai/unsloth.

[18] Jordan Hoffmann et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*. https://arxiv.org/abs/2203.15556.

[19] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. 2022. Lora: low-rank adaptation of large language models. https://arxiv.org/abs/2106.09685.

[20] Damjan Kalajdzievski. 2023. A rank stabilization scaling factor for fine-tuning with LoRA. (Nov. 28, 2023). arXiv: 2312.03732[cs]. DOI: 10.48550/arXiv.2312.03732.

[21] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature*. https://www.nature.com/articles/nature14539.

[22] Shane Legg and Marcus Hutter. 2007. Universal intelligence: a definition of machine intelligence. *Minds and Machines*, 17, 4, (Dec. 1, 2007), 391–444. DOI: 10.1007/s11023-007-9079-x.

[23] Wen-Ding Li et al. 2024. Combining induction and transduction for abstract reasoning. (2024). https://arxiv.org/abs/2411.02272 arXiv: 2411.02272 [cs.LG].

[24] Gary Marcus. 2018. Deep learning: a critical appraisal. *arXiv preprint arXiv:1801.00631*. https://arxiv.org/abs/1801.00631.

[25] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed representations of words and phrases and their compositionality. *Neural Information Processing Systems (NeurIPS)*. https://arxiv.org/abs/1310.4546.

[26] Long Ouyang et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*. https://arxiv.org/abs/2203.02155.

[27] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2019. Improving language understanding by generative pre-training. *OpenAI Blog*. https://openai.com/blog/language-unsupervised.

[28] Faisal Shehzad and Dietmar Jannach. 2023. Everyone's a winner! on hyperparameter tuning of recommendation models. In *Proceedings of the 17th ACM Conference on Recommender Systems*. Association for Computing Machinery, New York, NY, USA. DOI: 10.1145/3604915.3609488.

[29] David Silver, Aja Huang, Chris J. Maddison, and et al. 2017. Mastering the game of go without human knowledge. *Nature*, 550. https://www.nature.com/articles/nature24270.

[30] Unsloth Team and Community. 2025. Unsloth currently does not support multi gpu setups in unsloth-2024.8. Accessed: 2025-03-17. (2025). https://github.com/unslothai/unsloth/issues/859.

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *NeurIPS*. https://arxiv.org/abs/1706.03762.

[32] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models. (Jan. 10, 2023). arXiv: 2201.11903[cs]. DOI: 10.48550/arXiv.2201.11903.

[33] Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. 2020. TRL: transformer reinforcement learning. Publication Title: GitHub repository. (2020). https://github.com/huggingface/trl.

[34] Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. 2024. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Kevin Duh, Helena Gomez, and Steven Bethard, (Eds.) Association for Computational Linguistics, Mexico City, Mexico, (June 2024). DOI: 10.18653/v1/2024.naacl-long.102.

[35] Manzil Zaheer et al. 2020. Big bird: transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33.

A Acronyms

AGI Artificial General Intelligence
AI Artificial Intelligence
ARC Abstraction and Reasoning Corpus
CoT Chain-of-Thought
HF Hugging Face
HP Hyperparameter
LLM Large Language Model
LoRA Low-Rank Adaptation
ML Machine Learning
OOM Out Of Memory
PEFT Parameter-Efficient Fine-Tuning
QLoRA Quantized Low-Rank Adaptation
RLHF Reinforcement-Learning from Human Feedback
rsLoRA rank-stabilized Low-Rank Adaptation
SFT Supervised Fine-Tuning
SOTA State-of-the-Art
TTT Test-Time Training

B Color legend of the ARC Prize Competition

Table 3 illustrates the color legend officially used to visualize the underlying single digits within the datastructure of ARC-AGI-1.

Table 3: Color Legend of ARC-AGI-1 Tasks

Color	Digit
Black	0
Blue	1
Red	2
Green	3
Yellow	4
Gray	5
Pink	6
Orange	7
Light Blue	8
Dark Red	9

C Relevant links

C.1 Fine-tuned models

C.1.1 Original Approach without HP-tuning. :
https://huggingface.co/phogen/FineLlama-3.1-8B_instruct

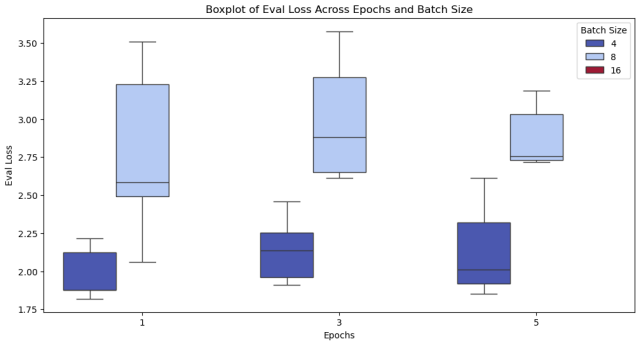
C.1.2 Best three of HP-tuning (enhanced dataset).

(1) https://huggingface.co/phogen/FineLlama-3.1-8B_instruct_eval_lr5e-05_batch4_epochs1_wd0.01
(2) https://huggingface.co/phogen/FineLlama-3.1-8B_instruct_eval_lr0.0005_batch4_epochs5_wd0.1
(3) https://huggingface.co/phogen/FineLlama-3.1-8B_instruct_eval_lr5e-05_batch4_epochs1_wd0.1

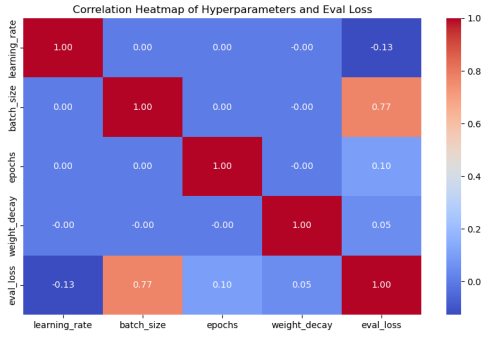
C.1.3 Best three of HP-tuning (original dataset).

(1) https://huggingface.co/phogen/FineLlama-3.1-8B_instruct_eval_lr0.0005_batch4_epochs1_wd0.01_WithoutReasonings
(2) https://huggingface.co/phogen/FineLlama-3.1-8B_instruct_eval_lr0.0005_batch4_epochs5_wd0.1_WithoutReasonings
(3) https://huggingface.co/phogen/FineLlama-3.1-8B_instruct_eval_lr0.0005_batch4_epochs1_wd0.1_WithoutReasonings

D HP-Tuning the Baseline - without reasonings



(a) Boxplot of baseline eval loss w.r.t. epochs and batch Size.



(b) Correlation heatmap of baseline eval loss and HPs.

Figure 7: HP-Plots of the baseline - trained without reasonings.

E Who did what

This sections purpose is to clarify who is responsible for which part of the project.

E.1 Pablo

- **Report Sections**
  - Title Page
  - Introduction Subsections: Part before first subsection (Subsection 1.0 technically), 1.1
  - Fine-Tuning Llama 3.1 8B on Reasoning Subsections: Part before first subsection (Subsection 3.0 technically), 3.1, 3.2, 3.4
  - Appendix Sections: B, C, and D
- **Midterm Presentation Slides**: 1-3, 5-14, and Appendix - see file *Mid-TermPresentationFinal.pptx*
- **Code**: Whole fine-tuning process including HP-tuning - see folder *Fine-Tuning*
- **Conda Environment**: Used environment exported as yml-file for reproducibility - see file *unsloth_env2.yml*

- **Dataset Contribution:** Authored 1/3 of the reasoning annotations - see file *ReasoningAssignment.xlsx* for division and *reasonings_final.csv* for final reasonings
- Final Presentation Slides: 12-22 see file *FinalPresentation.pptx*

E.2 Abhay

- **Report Sections**
  - 3.3 - Candidate Sampling
  - 4 - Experiments
  - 5 - Results
  - 6 - Conclusion
  - 7 - Discussion
- **Midterm Presentation Slides:** 15-39 - see file *Mid-TermPresentationFinal.pptx*
- **Code Contributions:**
  - Trained the Candidate Sampling Model
  - Ran inference on:
    - Original Baseline Model
    - Three Hyperparameter-Tuned Models
    - Three Baseline Models without Reasoning
    - Candidate Sampling Model
- **Dataset Contribution:** Authored **1/3** of the reasoning annotations - see file *ReasoningAssignment.xlsx* for division and *reasonings_final.csv* for final reasonings.
- **Final Presentation Slides:** 23-56 - see file *FinalPresentation.pptx*

E.3 Manitarun

- **Report Sections**
  - Abstract
  - 1 - Introduction subsection 1.2
  - 2 - Related Work
- **Midterm Presentation Slides:** 4, 40-43 - see file *Mid-TermPresentationFinal.pptx*
- **Code Contributions:**
  - Ran inference for the Original Baseline Model
- **Dataset Contribution:** Authored **1/3** of the reasoning annotations - see file *ReasoningAssignment.xlsx* for division and *reasonings_final.csv* for final reasonings.
- **Final Presentation Slides:** 1-11 - see file *FinalPresentation.pptx*