MASR: Multi-Agent System with Reflection for the Abstraction and Reasoning Corpus

Kiril Bikov[1], **Mikel Bober-Irizar**[1], **Soumya Banerjee**[1,*]
[1]University of Cambridge
Cambridge
United Kingdom
[*]Corresponding author: sb2333@cam.ac.uk

Abstract

The Abstraction and Reasoning Corpus (ARC) benchmarks general artificial intelligence, presenting a significant challenge to existing machine learning models and program synthesis solvers due to its focus on broad generalization. In this work, we introduce a Multi-Agent System with Reflection (MASR) for ARC. MASR combines Large Language Models (LLMs) and a program synthesis solver based on a Domain Specific Language (DSL). We analyse the accuracy of LLMs on ARC and demonstrate unsatisfactory results. We create AugARC, an augmented ARC benchmark, which consistently improves the performance of LLMs compared to the normal ARC benchmark. Using augmented ARC data, we fine-tune LLMs and observe a significant gain in ARC accuracy after training. By utilizing reflection, we combine LLMs and a previous DSL solver into our MASR approach for abstraction and reasoning. Our experiments show that MASR outperforms the previous publicly available ARC systems that consist solely of LLMs or DSL solvers, demonstrating the effectiveness of multi-agent systems on ARC. MASR motivates research to advance previous ARC attempts by combining the advantages of LLMs and program synthesis solvers into multi-agent systems.

Introduction

Incorporating abstract reasoning into machines has been an active research topic since the 1955 Dartmouth AI workshop (McCarthy et al. 2006). Despite significant progress in machine learning, today's AI systems still lack human-level abstract reasoning (Korteling et al. 2021). Studies have shown that digital systems are significantly inferior to humans in terms of abstract cognitive abilities (Boden et al. 2017; Shneiderman 2020).

To address the gap between human intelligence and AI models, François Chollet created the Abstract Reasoning Corpus (ARC) (Chollet 2019). ARC consists of 1000 visual tasks, that capture essential aspects of abstraction and analogy. The ARC tasks are split into 400 for training, 400 for evaluation and hidden 200 tasks for testing. A Program Synthesis approach from 2020 solved 40% of the complete evaluation set (Icecuber 2023), and a voting ensemble from 2024 achieved 40.25% (Bober-Irizar and Banerjee 2024).

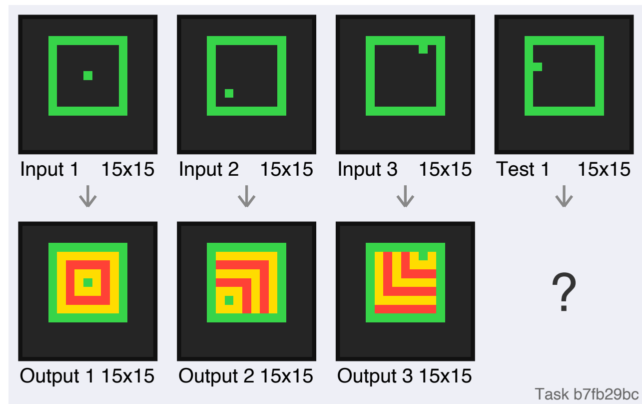
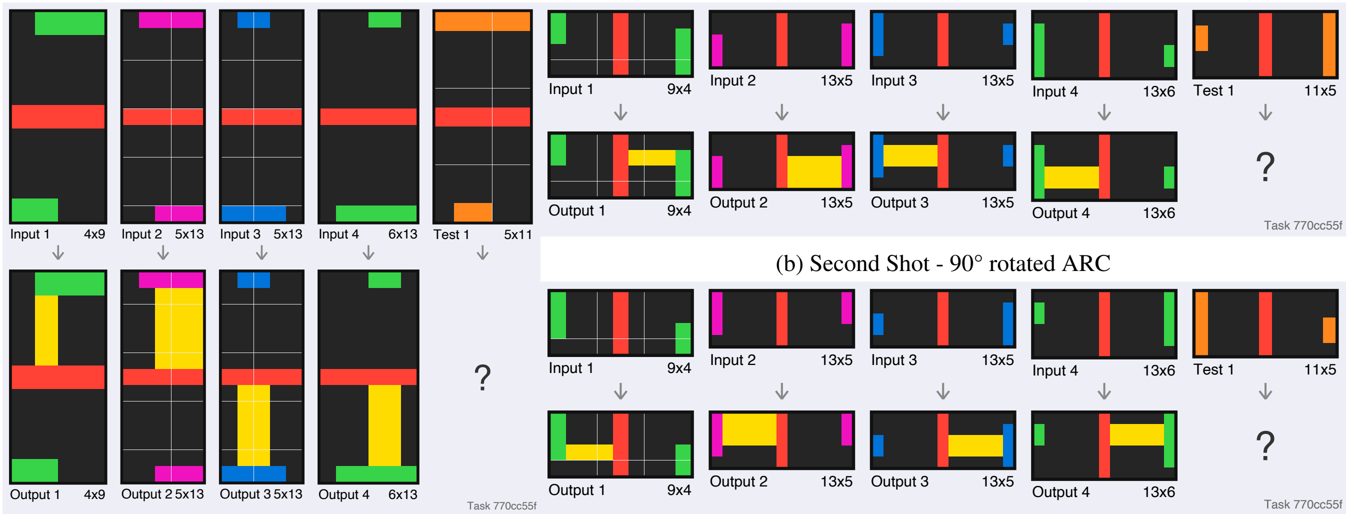


Figure 1: Visualisation of an ARC task. The test-taker is provided with some input-output pairs as examples. The objective is to recognise the transformation used in the given input-output pairs and apply it to the test input grid to obtain the test output grid.

State-of-the-art systems that attempt to solve ARC use heuristic search. Such models are heavily handcrafted and designed entirely with the goal of solving ARC. Also many of these approaches have only been tested on a subset of the ARC evaluation data. Some attempts have been made to use Large Language Models (LLMs) to solve ARC (Xu et al. 2023; Min 2023; Mitchell, Palmarini, and Moskvichev 2023), with some of the previous publications testing LLMs on the ARC evaluation set (Bober-Irizar and Banerjee 2024; Opiełka et al. 2024; Gendron et al. 2023; Lee et al. 2024b). Nevertheless, there have no attempts to build more advanced multi-agent systems based on several LLMs.

Hence, we aim to fully explore the abilities of base LLMs on ARC and how those can be combined in multi-agent systems. We introduce a new augmented ARC (AugARC) benchmark tailored for LLMs, which shows consistently improved performance across all tested LLMs compared to normal ARC. We show the benefit of fine-tuning

LLMs on augmented ARC data. Finally, we built a Multi-agent System with Reflection (MASR) that solves 5 more evaluation tasks than a previous systems that combines multiple ARC solvers (Bober-Irizar and Banerjee 2024).



(a) First Shot - Normal ARC

(b) Second Shot - 90° rotated ARC

(c) Third Shot - 270° rotated ARC

Figure 2: Evaluation Task of the 3-shot AugARC Benchmark. The first shot is a normal ARC evaluation task, while the second and third shots are 90° and 270° rotated. All three shots are represented as a 2D matrices of numbers, each one representing a different colour. The figure showcases the three shots as coloured grids for demonstration purposes.

AugARC: Augmented ARC for LLMs

The ARC training data can be utilized for fine-tuning LLMs and improving their performance on the evaluation and test sets. One potential issue with this approach is the size of the training set - it contains only 400 samples. Since LLMs have billions of parameters, they usually cannot be effectively trained on smaller datasets and instead require more samples. Therefore, due to its small size, the ARC training dataset limits the ability to fine-tune LLMs for improved broad generalization and reasoning.

Augmented Training Data

To overcome the limited number of ARC training tasks, we propose an augmentation procedure that can significantly extend the training dataset. Our approach expands the ARC training set by applying the following transformations:

- **Rotation:** clockwise rotation of each ARC grid for a given task by 90° or 270°.
- **Flipping:** flips each ARC grid of a task horizontally (along the y-axis) and vertically (along the x-axis).
- **Permutations:** rearranges the sequence of demonstration input-output pairs before the test input grid. We set a threshold for the maximum number of permutations per task to produce datasets of various sizes.

Depending on the transformations applied and the maximum number permutations applied, the augmented ARC training datasets vary from 2000 up to over 18 million tasks. The AugARC data is available from the following repository: https://github.com/kiril-bikov/AugARC

3-Shot AugARC Benchmark

A key reason for the relatively scarce ARC research on LLMs and Multi-agent systems is the lack of textual version of the benchmark. The only benchmark suitable for LLMs that resembles Chollet's visual ARC (Chollet 2019) is the AI2 Reasoning Challenge (Clark et al. 2018; Pătras et al. 2022). AI2 is a multi-choice question answering benchmark that focuses on assessing reasoning. Although AI2 is a more popular and well-established reasoning benchmark for LLMs compared to Chollet's ARC (Chollet 2019), the latter is more effective at evaluating broad generalization abilities due to its hand-crafted abstract logic.

Identifying that the lack of a textual ARC benchmark is a significant barrier for evaluating LLMs and multi-agent systems, we create the AugARC Benchmark. The AugARC Benchmark provides a unified, easy way to evaluate LLMs on 3-shot accuracy over reasoning tasks. In AugARC, each ARC task starts with a textual description explaining the format of the problem. Each ARC grid is represented as a 2D matrix of numbers.

Dataset Size	Max Permutations
2 000 tasks	-
4 000 tasks	2
5 715 tasks	3
7 430 tasks	4
9 145 tasks	5
18 668 610 tasks	All

Table 1: Size of the augmented ARC training datasets according to the maximum number of permutations. All datasets include 90° and 270° rotations, horizontal and vertical flipping. The augmented datasets range from 2000 to 18 million tasks.

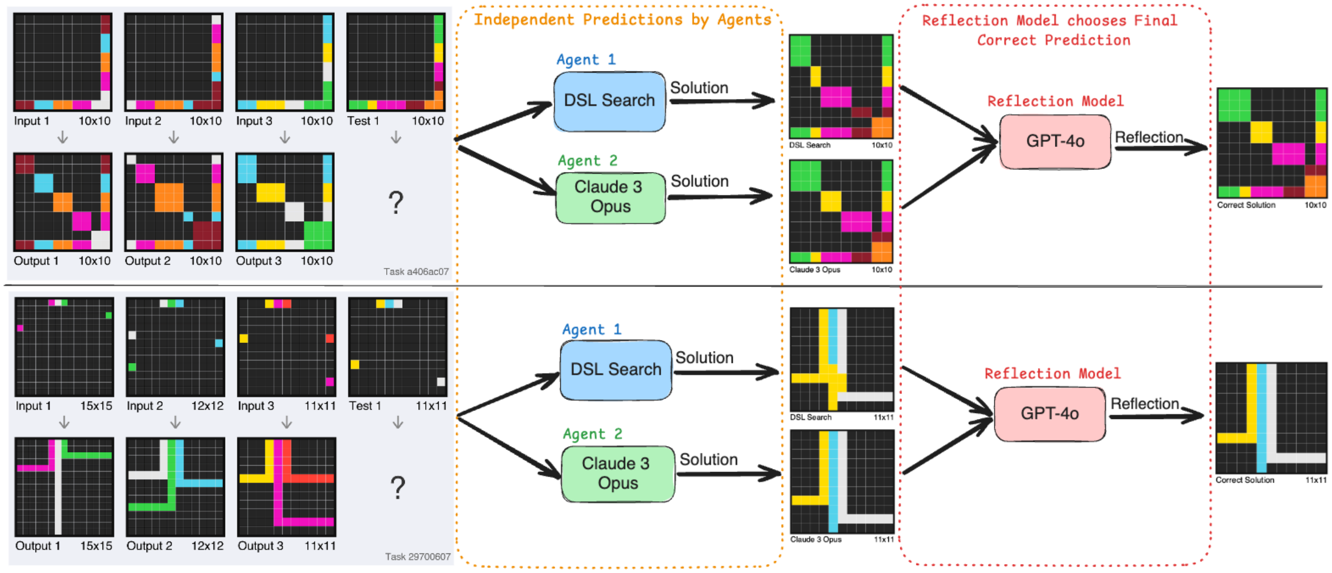


Figure 3: Multi-Agent System with Reflection (MASR) - execution on two ARC evaluation tasks. Initially, multiple agents make independent predictions on the task. Then, the task and the prediction are presented to the reflection model, which chooses the correct final prediction. In the example, agent 1 is based on program synthesis (DSL Search) and agent 2 is an LLM (Claude 3 Opus). The reflection model is an LLM (GPT-4o). Both task flows are actual demonstrations of how MASR perform on ARC evaluation tasks. In both cases, MASR produces the correct final solution.

AugARC Input to LLM The first prediction is based on a normal ARC task, whereas the second and the third ones are 90° and 270° clockwise rotated versions of the same task. The AugARC Benchmark is tailored for LLMs, as these models process inputs in an auto-regressive, sequential manner. By rotating the ARC tasks, LLMs are presented with a different sequence of numbers (2D matrices) which contain the same abstract logic.

Reproducing ARC Solutions from AugARC Outputs Although the second and third shot in AugARC are based on rotated ARC tasks, the output of the LLMs can easily be transformed back to a solution to the original ARC problem. Once an output is generated by the LLM, it is simply rotated back in an anticlockwise direction. In this way, AugARC only changes the input representation of the ARC problems, but the outputs by the models are then rotated to valid ARC solutions. This process ensures that the results with our approach are directly comparable with previous ARC attempts.

Fine-tuning LLMs on ARC tasks

Although LLMs have shown impressive capabilities, they can sometimes hallucinate and are therefore regarded as unreliable in reasoning tasks. One potential way to reduce such hallucinations and improve performance on abstract logical tasks is to fine-tune LLMs. Due to the limited size of the ARC training dataset (400 tasks), previous studies have not attempted to train LLMs on ARC. Our proposed augmentation of ARC allows us to overcome this limitation and have sufficient ARC data to fine-tune LLMs.

For efficient training of LLMs, we use Quantized Low-Rank Adaptation (QLoRA) with 4-bit NormalFloat (NF4) quantization (Dettmers et al. 2024). Low-Rank Adaptation constrains the update of a pre-trained weight matrix $W_0 \in \mathbb{R}^{d\times k}$ with a low-rank decomposition $W_0 + \Delta W = W_0 + BA$, where $B \in \mathbb{R}^{d\times r}$, $A \in \mathbb{R}^{r\times k}$, and the rank $r \ll \min(d, k)$ (Hu et al. 2021). During training, W_0 is frozen and does not receive gradient updates, while A and B contain trainable parameters. Both W_0 and $\Delta W = BA$ are multiplied with the same input, and their respective output vectors are summed coordinate-wise (Hu et al. 2021).

Using QLoRA, we fine-tune LLMs on an augmented ARC training dataset consisting of 2000 tasks 1. Due to a significant increase in computational complexity, we avoid fine-tuning the models on some of the bigger augmented ARC training sets from Table 1. For the same reason, we only train LLMs with number of parameters ranging from 7 to 13 billion.

MASR: Multi-Agent System with Reflection

A previous promising approach which solves 40.25% of the ARC evaluation tasks combines solutions from different ARC solvers (Bober-Irizar and Banerjee 2024). This approach utilizes a voting ensemble of systems that “votes” for the predictions of an LLM, a Program Synthesis solver and a Neuro-symbolic model (Bober-Irizar and Banerjee 2024). The voting ensemble outperforms systems that are solely based either on LLMs or Program Synthesis (such as DSL Search (Icecuber 2023)).

The encouraging result of the voting ensemble motivates further research into combining different architectures into a complex system for enhanced ARC performance. Although the voting ensemble achieves promising ARC accuracy, it lacks any "intelligent" analysis of potential solutions and instead uses a weighting algorithm (Bober-Irizar and Banerjee 2024). In order to build upon this limitation and combine multiple previous attempts into a new complete approach, we propose a Multi-Agent System with Reflection (MASR).

MASR relies on agents that could have various architectures - for example, LLMs or Domain Specific Languages using Program Synthesis (Icecuber 2023). When predicting the correct solution to an ARC task, MASR executes in two main stages, as visualised in Figure 3.

Predictions by Agents

In the first stage, each agent makes a prediction on the given ARC task. Each agent works independently and cannot access the outputs of other agents. Once an agent has produced a prediction for the ARC task, it passes the solution to the reflection model.

Reflection over all Prediction

The second stage of MASR is inspired by previous studies on self-reflection (Lee et al. 2024a; Renze and Guven 2024), in which LLMs refine their responses based on feedback against previous outputs and achieve more accurate predictions. In MASR, the reflection model processes all the predictions generated by the agents, together with the initial task. Conditioned on the given ARC task, the reflection model chooses the prediction from the agents that is most likely to be correct.

Flexibility of MASR

In MASR, an agent can be any ARC solver, including LLMs, program synthesis approaches or neuro-symbolic models. MASR allows any number of agents to be used, as the reflection model can easily process the outputs of various agents. This makes MASR a highly flexible and customisable architecture, as each of its components - the agents and the reflection model, can easily be changed. This architectural design allows MASR to be easily tested with various ARC solvers to find the optimal ARC configuration.

Experiments

We perform all experiments on the ARC evaluation set which consist of 400 tasks. By design, the ARC evaluation set is significantly more challenging than the training set (Chollet 2019). The creator of ARC, François Chollet, emphasised that the performance of intelligent systems should be measured by the fraction of solved tasks on the evaluation set (Chollet 2019). Therefore, we perform our experiments on the evaluation set and use 3 shots per task, as set out in the ARC design (Chollet 2019).

To present fully reproducible results, all experiments are executed on the complete evaluation set. Some previous solvers have been evaluated on a subset of the ARC evaluation data, making it difficult to understand the true performance of the solver. Our testing approach ensures that future studies could easily use our results for direct comparison with new ARC solvers.

Performance on base ARC and AugARC

We start our experiments with LLMs on the base ARC benchmark, shown in Table 2. The ARC accuracy across 7-13 billion models ranges from 5 to 9. Bigger LLMs solve slightly more ARC tasks, from 7 to 20, with Gemini Pro achieving the highest accuracy (20).

Model	ARC	AugARC	Increase
Llama-2 7B	5/400	7/400	29%
Mistral 7B	9/400	15/400	67%
Llama-2 13B	5/400	8/400	100%
Llama-2 70B	7/400	14/400	100%
Mixtral 8x7B	9/400	18/400	**125%**
Gemini Pro	20/400	**33/400**	65%

Table 2: Performance of LLMs on ARC and AugARC (on the evaluation set). There is a consistent increase of the accuracy of LLMs when using the AugARC inputs compared to using the base ARC ones (29-125%).

Using the same LLMs, we evaluate the performance on AugARC. For all LLMs, there is a clear accuracy improvement on AugARC compared to the base ARC. The increase varies from 29% for Llama-2 7B up to 125% for Mixtral 8x7B, with the majority of models achieving at least 60%.

The significant improvement in all LLMs on AugARC compared to ARC suggests that changing the grid structure of the tasks for the second and third shot leads to enhanced accuracy. LLMs process the ARC tasks sequentially, and thus are directly influenced by the exact order of the grids. Based on the results, we conclude that the proposed AugARC benchmark is well-suited for the testing LLMs.

Since AugARC results are directly comparable to ARC, we proceed to use AugARC for the remainder of our experiments.

ARC accuracy across LLMs

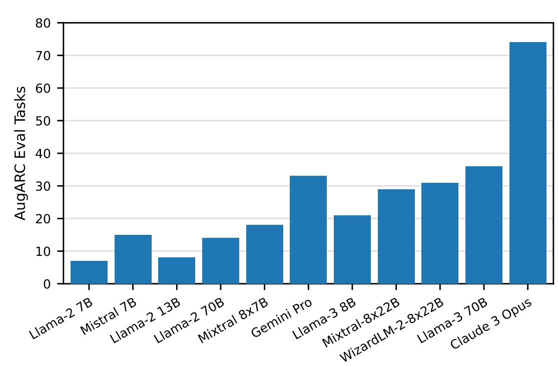


Figure 4: ARC evaluation tasks solved by LLMs. Claude 3 Opus solves the most ARC tasks (74).

The ARC accuracy of LLMs ranges between 7 up to 74 solved tasks, as visualised in Figure 4. The best performance by a smaller 7B model is achieved by Llama-3 8B

(21). Some bigger open-source LLMs can solve more than 30 ARC tasks, with Llama-3 70B achieving 36. The highest number of solved ARC task, 74, is by Claude 3 Opus.

The ARC results demonstrated some variability in performance across LLMs. Bigger models appear to be more accurate on ARC compared to smaller ones. Most LLMs achieve an accuracy in the range of 10-35 tasks, with the only exception being Claude 3 Opus with 74.

Performance of Fine-tuned LLMs on ARC

To observe whether we can reduce the performance gap between smaller and bigger LLMs on ARC, we fine-tune the 7 and 13B models. The models are fine-tuned on the training set only using a single Nvidia A100 80GB GPU.

The results in Table 3 show that the fine-tuned LLMs solve between 18 and 34 ARC evaluation tasks. Training benefited all the models substantially - the small fine-tuned Llama-2 7B and 13B achieved a performance on par with the base versions of the significantly bigger models such as Llama-2 70B. After fine-tuning, Mistral 7B outperforms the standard Mixtral 8x7B by 5 correct tasks. The Llama-3 8B model achieved the best result with 34 correct solutions after fine-tuning, surpassing the performance of Gemini Pro.

Model	Base	Fine-tuned	Increase
Llama-2 7B	7/400	21/400	**200%**
Mistral 7B	15/400	23/400	53%
Llama-2 13B	8/400	18/400	125%
Llama-3 8B	21/400	**34/400**	62%

Table 3: Results of base and fine-tuned LLMs on the ARC evaluation set. The increase column shows the improvement in accuracy from a base LLM compared to its fine-tuned version. All LLMs consistently show improved ARC performance after fine-tuning, ranging from 62% to 200%.

The results in Table 3 demonstrate a significant increase in ARC performance across all fine-tuned LLMs compared to their base versions. The accuracy improvement after training varies between 53% in Mistral 7B up to 200% in Llama-2 7B. While Llama-2 7B and 13B both achieve more than 100% improvement - 125% and 200% respectively, Mistral 7B and Llama-3 8B improved in the range of 50% to 65%.

Based on our results, we conclude that training small LLMs on an AugARC dataset consistently improves their performance. Notably, fine-tuning smaller LLMs (7-13B parameters) is so effective that it can lead to better ARC performance than significantly bigger base LLMs.

Solutions Overlap and Gain Measure

To motivate our multi-agent approach to ARC, we show the benefit of combining ARC solutions from base and fine-tuned LLMs together with Program Synthesis solvers.

The ratio of overlapping solutions between different ARC solvers is visualised in Figure 5a. The numbers in Figure 5a refer to the proportion of overlapping tasks solved by the systems on the left and on the bottom. For each pair of LLMs, there is an overlap between 0.5 and 0.9 in their correct ARC solutions. A lower overlap can be observed between the base LLMs and the fine-tuned ones. For example, a fine-tuned Mistral 7B has an overlap of only 0.52 with standard models such as Mixtral 8x22B and Llama-3 70B. The low overlap in the solutions between fine-tuned and base LLMs indicates that training the models leads to correct solutions to new ARC tasks, which have previously not been solved by the base LLMs. The solution overlap between LLMs and a Program Synthesis solver, DSL Search (Icecuber 2023), ranges between 0.69 and 0.9.

We also measure the gain from adding a second system when testing the models on ARC. Figure 5b shows how much the base systems on the left would benefit from adding the solutions from the models at the bottom. We visualise how much the base systems can gain from utilizing new correct solutions from the second systems. For most LLMs, the gain of adding the solutions from another LLM is between 3 and 24. The gain between every two LLMs is slightly skewed by Claude 3 Opus due to its substantially better performance than any other LLM.

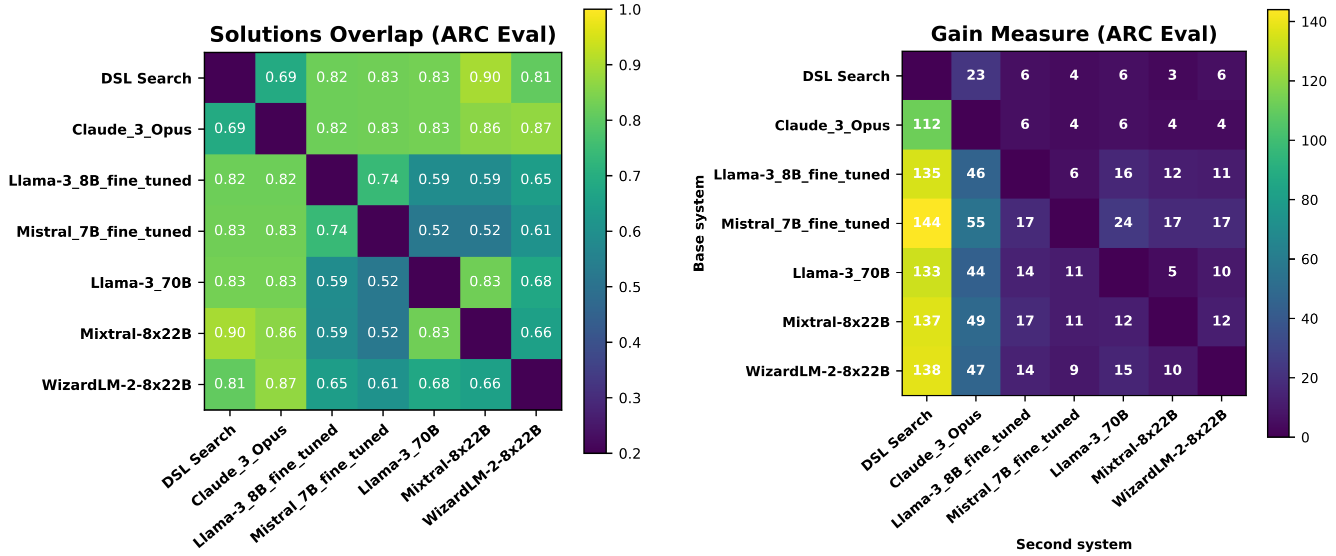
Most LLMs only solve 3 to 6 new tasks compared to DSL Search (Icecuber 2023). Importantly, Claude 3 Opus could contribute with 23 new correct solutions to the DSL Search, leading to a substantial improvement in ARC accuracy. This encouraging result motivates a new approach, which can effectively combine solutions from LLMs such as Claude 3 Opus with ARC outputs by Program Synthesis solvers (such as DSL Search (Icecuber 2023)).

Performance of MASR configurations

We experiment with MASR configurations based on two or three agents and with different reflection models. Since the current best publicly available ARC performance is by a Program Synthesis solver (DSL Search (Icecuber 2023)), we always include it as an agent in all of our MASR experiments. We also always include the LLM with highest ARC accuracy as an agent (Claude 3 Opus). We experiment with base and fine-tuned LLMs for the reflection models and a potential third agent to find the MASR configurations which achieve the highest ARC accuracy.

Agent 1	Agent 2	Agent 3	Reflection Model	ARC Correct
DSL Search	Claude 3 Opus	-	Llama-3 70B	133/400
DSL Search	Claude 3 Opus	-	GPT-4-turbo	165/400
DSL Search	Claude 3 Opus	-	GPT-4o	**166/400**
DSL Search	Claude 3 Opus	Fine-Tuned Llama-3 8B	Claude 3.5 Sonnet	163/400

Table 4: Correctly solved ARC evaluation tasks in a 3-shot setting by different MASR configurations. The best 2-agent performance is with DSL Search and Claude 3 Opus as agents and GPT-4o as a reflection model (166). The highest 3-agent accuracy adds a fine-tuned Llama-3 8B model (163).



(a) Overlapping ARC tasks between the system on the left and the one on the bottom.

(b) Number of new solutions from adding a second system (bottom) to the first one (left).

Figure 5: Overlap of solutions and gain measure between systems. The systems are ordered by how much gain they add. In (a), the overlap ranges from 0.5 up to 0.9, with lower values between fine-tuned and base LLMs. In (b), Claude 3 Opus could add 23 new solutions to the DSL Search, while the remaining LLMs could add between 3 and 6.

Table 4 shows that the ARC performance by different MASR configurations varies between 133 and 166 solved evaluation tasks. In a 2-agent setting, with DSL Search and Claude 3 Opus, Llama-3 70B struggles as a reflection model, solving only 133 tasks. GPT-4-turbo and GPT-4o perform significantly better as reflection models, achieving 165 and 166 ARC tasks. When adding a fine-tuned Llama-3 8B as a third agent, MASR solves 163 ARC tasks.

Our best 2-agent and 3-agent MASR configurations both outperform the best single LLM, Claude 3 Opus (74), and the best Program Synthesis approach, DSL Search (160). Based on the results, we argue that MASR is an effective approach for combining LLMs and Program Synthesis solvers into multi-agent systems for enhanced ARC performance.

Comparison of MASR to Previous Approaches

To demonstrate the effectiveness of MASR, we compare the ARC performance of our optimal configuration with previously publicly available approaches. We present systems that have been tested on the complete ARC evaluation dataset and split the categories into LLMs, Neuro-symbolic models and Program Synthesis solvers. We also include a previous attempt at combining different ARC solvers with a voting ensemble (Bober-Irizar and Banerjee 2024).

Table 5 shows that MASR outperforms all previous publicly available systems on the ARC evaluation dataset. MASR achieves a substantially higher accuracy than any previous LLM or Neuro-symbolic model. Furthermore, it outperforms the best available Program Synthesis solver (DSL Search) by 6 ARC tasks. MASR also surpasses the accuracy of the voting ensemble (Bober-Irizar and Banerjee 2024), solving 5 more ARC tasks.

System Type	Method	ARC Correct
Neuro-Symbolic	DreamCoder (Bober-Irizar and Banerjee 2024)	18/400
	CodeIt (Butt et al. 2024)	59/400
LLM	GPT-4 (Bober-Irizar and Banerjee 2024)	32/400
	Fine-Tuned Llama-3 8B	34/400
	Llama-3 70B	36/400
	Claude 3 Opus	74/400
Program Synthesis	Brute Force (Ainooson et al. 2023a)	26/400
	Neurodiversity solver (Ainooson et al. 2023b)	45/400
	DSL Search (Icecuber 2023)	160/400
Ensemble	Voting (Bober-Irizar and Banerjee 2024)	161/400
Multi-agent	**MASR** (Agents: DSL Search, Claude 3 Opus; Reflection: GPT-4o)	**166/400**

Table 5: Number of correctly solved ARC evaluation tasks across different system types. The highest ARC accuracy is achieved by MASR (166).

The results in Table 5 show that MASR achieves a new best performance compared to all previous publicly available systems that have been tested on the complete ARC evaluation dataset. The accuracy of MASR demonstrates the effectiveness of combining LLMs and Program Synthesis solvers in multi-agent systems with reflection on the ARC benchmark.

Related Work

Most of the previous ARC attempts can be split into two categories: Program Synthesis solvers and methods that rely on machine learning. The most successful publicly available attempt relies on efficient search implementations (Icecuber 2023). In contrast, machine learning implementations vary from Neuro-symbolic models to the LLMs.

Program Synthesis Solvers

A popular Program Synthesis solver for ARC is the DSL Search implementation by IceCuber which achieves 40% accuracy on the complete ARC evaluation dataset. The DSL solution is based on brute-force search. It applies transformations of varying depth in parallel and greedily stacking them to fit training samples (Icecuber 2023). The final prediction is ensembled based on the most solved training samples and least depth.

Another promising Program Synthesis approach is the the Generalized Planning for Abstract Reasoning (GPAR) solver (Lei, Lipovetzky, and Ehinger 2024). It casts an ARC problem as a generalized planning (GP) problem, where a solution is formalized as a planning program with pointers (Lei, Lipovetzky, and Ehinger 2024). On 160 of the 400 ARC evaluation tasks, GPAR outperforms the DSL Search by 10% (Lei, Lipovetzky, and Ehinger 2024). Nevertheless, due to the lack of GPAR results on the complete ARC evaluation set, in this work, we refer to the DSL Search as the current best Program Synthesis solver.

Neuro-symbolic Models

Neurosymbolic models have emerged as promising AI systems that aim to integrate the ability to learn from experience and the ability to reason from what has been learnt (Garcez et al. 2019). In neuro-symbolic computing, knowledge is represented in symbolic form, whereas learning and reasoning are performed by a neural network (Garcez et al. 2019).

The first Neuro-symbolic approach to solving ARC is DreamCoder (Alford 2021). It uses neural networks to guide its ability to write programs (Bober-Irizar and Banerjee 2024). An initial implementation of DreamCoder (Alford 2021) solves 2 ARC evaluation tasks, and an updated version with a Perceptual Abstraction & Reasoning Language (PeARL) achieves 18 (Bober-Irizar and Banerjee 2024).

Code Iteration (CodeIt) is a recent Neuro-symbolic that approaches ARC (Butt et al. 2024) as a programming-by-examples problem by training a policy to produce programs when shown demonstration examples (Butt et al. 2024). Experiments on the complete ARC evaluation set show that CodeIt solves 59 tasks, significantly outperforming previous Neuro-symbolic approaches.

Large Language Models

Previous research exploring LLMs in ARC has focused primarily on OpenAI's GPT models (Mitchell, Palmarini, and Moskvichev 2023; Mirchandani et al. 2023; Moskvichev, Odouard, and Mitchell 2023). Many published studies evaluate LLMs on a subset of the ARC evaluation dataset and make high-level observations (Opiełka et al. 2024; Xu et al. 2023; Qiu et al. 2023; Wang et al. 2023).

Complete experiments on the 400 ARC evaluation tasks show that the best overall LLM is GPT-4 with 32 correct tasks (Bober-Irizar and Banerjee 2024), while the best open-source LLM is LLaMa-65B with 13 correct solutions (Bober-Irizar and Banerjee 2024). A general conclusion from existing work is that LLMs fail on simple ARC tasks (Xu et al. 2023).

Ensembling Different System Types

A promising approach to ARC is to use a voting ensemble of systems: each system can propose an ARC solution which they "vote" for (Bober-Irizar and Banerjee 2024). Using this voting ensemble to combine the DSL Search (Icecuber 2023), DreamCoder and GPT-4 solutions achieves 161 correct tasks on the ARC evaluation set (Bober-Irizar and Banerjee 2024).

Limitations

Since we did not have access to the data used for pre-training the LLMs, we cannot exclude the hypothesis that some models might have been pre-trained either on ARC tasks or on other very similar abstract problems. Additionally, our multi-agent approach lacked communication and collaboration between the agents. The independence between the agents in MASR can limit its flexibility. Another potential limitation is that most of the correct solutions generated by MASR are produced by the DSL search (160 out of 166).

Conclusion

We implement a Multi-Agent System with Reflection (MASR) system, which effectively combines ARC solutions from LLMs and a Program Synthesis solver. We demonstrated that MASR can easily be configured to work with different number of agents. The architecture of MASR also allows the reflection model to be easily changed. On the complete ARC evaluation dataset, MASR configurations with 2 and 3 agents outperformed previous approaches such as LLMs, Neuro-symbolic models, Program Synthesis, and a voting ensemble.

In future work, MASR can be extended with more than 3 agents, potentially including another Program Synthesis solver such as GPAR (Lei, Lipovetzky, and Ehinger 2024). The MASR architecture could be improved by allowing the agents to collaborate and communicate when solving the ARC tasks. Future studies can improve the MASR architecture by making agents support each other dynamically during computation on more complex reasoning tasks.

References

Ainooson, J.; Sanyal, D.; Michelson, J. P.; Yang, Y.; and Kunda, M. 2023a. An approach for solving tasks on the Abstract Reasoning Corpus. *arXiv preprint arXiv:2302.09425*.

Ainooson, J.; Sanyal, D.; Michelson, J. P.; Yang, Y.; and Kunda, M. 2023b. A Neurodiversity-Inspired Solver for the Abstraction\& Reasoning Corpus (ARC) Using Visual Imagery and Program Synthesis. *arXiv preprint arXiv:2302.09425*.

Alford, S. 2021. *A Neurosymbolic Approach to Abstraction and Reasoning*. Ph.D. thesis, Massachusetts Institute of Technology.

Bober-Irizar, M.; and Banerjee, S. 2024. Neural networks for abstraction and reasoning: Towards broad generalization in machines. *arXiv preprint arXiv:2402.03507*.

Boden, M.; Bryson, J.; Caldwell, D.; Dautenhahn, K.; Edwards, L.; Kember, S.; Newman, P.; Parry, V.; Pegman, G.; Rodden, T.; et al. 2017. Principles of robotics: regulating robots in the real world. *Connection Science*, 29(2): 124–129.

Butt, N.; Manczak, B.; Wiggers, A.; Rainone, C.; Zhang, D.; Defferrard, M.; and Cohen, T. 2024. Codeit: Self-improving language models with prioritized hindsight replay. *arXiv preprint arXiv:2402.04858*.

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

Clark, P.; Cowhey, I.; Etzioni, O.; Khot, T.; Sabharwal, A.; Schoenick, C.; and Tafjord, O. 2018. Think you have solved question answering? try arc, the ai2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Dettmers, T.; Pagnoni, A.; Holtzman, A.; and Zettlemoyer, L. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*, 36.

Garcez, A. d.; Gori, M.; Lamb, L. C.; Serafini, L.; Spranger, M.; and Tran, S. N. 2019. Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning. *arXiv preprint arXiv:1905.06088*.

Gendron, G.; Bao, Q.; Witbrock, M.; and Dobbie, G. 2023. Large language models are not strong abstract reasoners. *arXiv preprint arXiv:2305.19555*.

Hu, E. J.; Shen, Y.; Wallis, P.; Allen-Zhu, Z.; Li, Y.; Wang, S.; Wang, L.; and Chen, W. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

Icecuber. 2023. ARC: 1st Place Solution. Available at: https://www.kaggle.com/code/icecuber/arc-1st-place-solution/execution. Accessed: 01 August 2024.

Korteling, J. H.; van de Boer-Visschedijk, G. C.; Blankendaal, R. A.; Boonekamp, R. C.; and Eikelboom, A. R. 2021. Human-versus artificial intelligence. *Frontiers in artificial intelligence*, 4: 622364.

Lee, K.; Hwang, D.; Park, S.; Jang, Y.; and Lee, M. 2024a. Reinforcement Learning from Reflective Feedback (RLRF): Aligning and Improving LLMs via Fine-Grained Self-Reflection. *arXiv preprint arXiv:2403.14238*.

Lee, S.; Sim, W.; Shin, D.; Hwang, S.; Seo, W.; Park, J.; Lee, S.; Kim, S.; and Kim, S. 2024b. Reasoning Abilities of Large Language Models: In-Depth Analysis on the Abstraction and Reasoning Corpus. *arXiv preprint arXiv:2403.11793*.

Lei, C.; Lipovetzky, N.; and Ehinger, K. A. 2024. Generalized planning for the abstraction and reasoning corpus. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 20168–20175.

McCarthy, J.; Minsky, M. L.; Rochester, N.; and Shannon, C. E. 2006. A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955. *AI magazine*, 27(4): 12–12.

Min, T. J. C. 2023. An Approach to Solving the Abstraction and Reasoning Corpus (ARC) Challenge. *arXiv preprint arXiv:2306.03553*.

Mirchandani, S.; Xia, F.; Florence, P.; Ichter, B.; Driess, D.; Arenas, M. G.; Rao, K.; Sadigh, D.; and Zeng, A. 2023. Large language models as general pattern machines. *arXiv preprint arXiv:2307.04721*.

Mitchell, M.; Palmarini, A. B.; and Moskvichev, A. 2023. Comparing humans, gpt-4, and gpt-4v on abstraction and reasoning tasks. *arXiv preprint arXiv:2311.09247*.

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

Opiełka, G.; Rosenbusch, H.; Vijverberg, V.; and Stevenson, C. E. 2024. Do large language models solve ARC visual analogies like people do? *arXiv preprint arXiv:2403.09734*.

Pătras, C.-B.; Pîrtoacă, G.-S.; Rebedea, T.; Ruşeţi, Ş.; et al. 2022. More with Less: ZeroQA and Relevant Subset Selection for AI2 Reasoning Challenge. *Procedia Computer Science*, 207: 2757–2766.

Qiu, L.; Jiang, L.; Lu, X.; Sclar, M.; Pyatkin, V.; Bhagavatula, C.; Wang, B.; Kim, Y.; Choi, Y.; Dziri, N.; et al. 2023. Phenomenal yet puzzling: Testing inductive reasoning capabilities of language models with hypothesis refinement. *arXiv preprint arXiv:2310.08559*.

Renze, M.; and Guven, E. 2024. Self-Reflection in LLM Agents: Effects on Problem-Solving Performance. *arXiv preprint arXiv:2405.06682*.

Shneiderman, B. 2020. Human-centered artificial intelligence: Three fresh ideas. *AIS Transactions on Human-Computer Interaction*, 12(3): 109–124.

Wang, R.; Zelikman, E.; Poesia, G.; Pu, Y.; Haber, N.; and Goodman, N. D. 2023. Hypothesis search: Inductive reasoning with language models. *arXiv preprint arXiv:2309.05660*.

Xu, Y.; Li, W.; Vaezipoor, P.; Sanner, S.; and Khalil, E. B. 2023. Llms and the abstraction and reasoning corpus: Successes, failures, and the importance of object-based representations. *arXiv preprint arXiv:2305.18354*.