MSc in Artificial Intelligence

Project Report
2024

Evaluations of Large Language Models using the Abstraction and Reasoning Corpus

Grigorios Vaitsas

Supervised by: Dr Eugenio Alberdi

Date of Submission: 1 November 2024

Declaration

By submitting this work, I declare that this work is entirely my own except those parts duly identified and referenced in my submission. It complies with any specified word limits and the requirements and regulations detailed in the assessment instructions and any other relevant programme and module documentation. In submitting this work I acknowledge that I have read and understood the regulations and code regarding academic misconduct, including that relating to plagiarism, as specified in the Programme Handbook. I also acknowledge that this work will be subject to a variety of checks for academic misconduct.

Signed: Grigorios Vaitsas

Abstract

In this study we explore the question of how well Large Language Models (LLMs) can perform in tasks that require reasoning and abstraction abilities. We deploy the Abstraction and Reasoning Corpus (ARC), an artificial intelligence benchmark derived from psychometrics, to evaluate the performance of several LLMs, including some state of the art models, in tasks that require the inference of logical transformations from just a few examples. We investigate various methods of prompting the LLMs in ARC tasks including the use of direct-grid encodings in text as well as visual prompting through the use of images. We then apply an external tool which allows us to generate object-based representations of ARC tasks through the use of graphs, before prompting the LLMs. We find that the type of textual encoding in direct-grid representations affects the performance of the LLMs, with a more structured format providing somewhat better results. We find that visual prompting performs very poorly and that the use of Chain of Thought techniques in our prompts does not induce the expected outcome. On the other hand, the addition of object-based representations greatly boosts the performance of LLMs and we find that with its use, OpenAI's o1-preview model is able to solve 44 out of the 50 tasks in our dataset. The code for this analysis is available at https://github.com/vitsiozo/ARC

Dedicated to Eri Filippou

(1952–2024)

@}–;–

Table of Contents

1 Introduction

1.1 Objectives

One of the fiercest debates in AI these days is whether or not Large Language Models (LLMs) can reason. Well respected AI scientists can be found to advocate on either side of this argument. For example, the recent laureate of the 2024 Nobel Prize in Physics, Geoffrey Hinton, often referred to as the "Godfather of AI", firmly believes that LLMs have reasoning abilities (Heaven 2023) that will only get better with time. On the other hand Yann LeCun, chief AI scientist at Meta and recipient of the Turing award, holds an opposing viewpoint, claiming that LLMs do not at this stage have reasoning capabilities (Fridman 2024). This chasm is very concerning as it leads to a lot of confusion in the field as well as the public opinion and it hampers efforts at implementing this rapidly evolving technology in a safe and ethical way.

As a result, a lot of recent scientific research has focused on clarifying this issue. In 2019 a ground-breaking paper was published by François Chollet (Chollet 2019), an AI scientist working at Google, that purports to offer a methodical approach to answering this question. In this paper Chollet introduces a novel benchmark that can be used to evaluate intelligence in artificial systems called "The Abstraction and Reasoning Corpus" (ARC) which is derived from classic psychometric tests. This benchmark is currently a very hot topic in AI circles with an active competition launched in the summer of 2024 (Chollet et al. 2024).

The motivation behind this study is to try and establish some basic facts around the question of reasoning in LLMs. We will start by presenting a brief discussion on the nature of intelligence as this pertains to artificial systems, outlining some of the main distinctive features that are critical in shaping a meaningful definition. Then, we will introduce the Abstraction and Reasoning Corpus (ARC) as a well-grounded benchmark for evaluating intelligence in artificial systems. We will then be using ARC as a tool to perform a number of experiments. Our goals are to:

1. Test how well LLMs perform on ARC tasks overall and establish whether their performance has been improving with every new model iteration.

2. Investigate different textual encoding methods for presenting ARC tests to LLMs and evaluate how these can affect the result.

3. Study whether prompting techniques like Chain of Thought (CoT) can offer any advantage over simpler prompting approaches.

4. Examine visual prompting using multimodal LLMs and test whether these are better at solving ARC tasks compared to traditional, text-only prompts for LLMs.

5. Deploy an external tool to assist in creating object awareness in ARC tasks and evaluate whether this enhances the performance of LLMs.

6. Perform these tests using a variety of LLMs from various providers including state of the art models.

7. Present previous research in this area and discuss how it compares with our results.

These goals will be addressed in the following order: Chapter 2 discusses the concept of intelligence and the problems around establishing common definitions. It introduces the ARC framework and presents some of the literature around ARC and the best efforts at solving it. Chapter 3 sets forth our plan for this analysis describing in detail the experiments that we carried out. Chapter 4 contains the analytic results of our experiments together with an assessment of their significance. Chapter 5 offers a more general appraisal of our analysis and possible considerations for the future. Finally, Chapter 6 sets out some closing thoughts on the study.

1.2 Background

Artificial Intelligence is currently dominating public discourse. Every day news articles appear that either exalt the technology as a panacea for all of humanity's ills or conversely demonise it as the source of our impending catastrophe. Despite the fact that AI is a field with a long and rich history, it was only in November of 2022, when OpenAI publicly released ChatGPT version 3.5, that the attention of the wider public was patently captivated and the ensuing hype started to proliferate (Mollick December 2022). The reason for this is that applications like ChatGPT, a generative AI chatbot based on the transformer architecture (Vaswani et al. 2023), with enhanced conversational and text

manipulation skills, have only at that time started to be really useful for large numbers of people.

The success of Large Language Models like ChatGPT in processing text, answering questions and generating content have led many to talk of an "intelligence revolution" akin to the industrial revolution that is likely to engulf and transform all aspects of our lives (Suleyman & Bhaskar 2023). Many believe that this progress will continue unfettered and that soon enough AI and LLMs in particular will reach a critical point where their abilities will have surpassed those of their human designers. This singularity point is widely discussed as AI achieving AGI or "Artificial General Intelligence" which denotes the ability of an AI system to expand its knowledge and abilities or "generalise" beyond the limitations of its training instructions and dataset or, as some others put it, achieving human-level intelligence (Talagala November 2023). Some prominent scientists however, like Yann LeCun, one of the leading figures in the development of neural networks, remain much more sceptical with regards to the current abilities of LLMs and caution against thinking that what we currently see with LLMs are true signs of intelligence (Murphy & Criddle May 2024).

There is an evident gulf in opinions, and the question of the extent to which LLMs are able to perform reasoning and abstraction tasks has become a hotly contested issue. Any progress that we make towards answering this question will be instrumental for the future development of AI and the choices that we make with regards to safety and ethical considerations. However, to do this we first of all need to formalise our definition of intelligence so as to have a consensus on what everyone is talking about. This will allow us to create the right tools for measuring and evaluating intelligence in artificial systems and thus be able to track the progress of LLMs and other cutting edge AI models. It is only when such a framework is in existence that any conversation about AGI can assume a meaningful character.

1.3 Audience

We hope that this study will benefit a wider audience than those with a specific interest in generative AI and ARC. The general question, of which we only barely scratch the

surface in this thesis, on whether leading AI systems of our times are able to reason, is very far reaching and of vast importance for the immediate future. In this study we confine ourselves to describing the principles and setting a framework around which a discussion can proceed. In particular we believe that the introductory Context chapter will offer some valuable understanding into questions regarding defining and evaluating intelligence in artificial systems and will shed some light into the difficulties that lie therein.

The consecutive chapters have a more technical focus on how using a modern benchmark can help us to identify elements of reasoning in LLMs and how the framing of these tests can influence LLM performance. We will also be touching upon the question of whether novel approaches, exemplified by the use of custom tools and methods, might be needed in order to help LLMs become better reasoners. In doing this, we will be hopefully creating some insights that will be of value to anyone who is interested in ARC, Large Language Models, prompting techniques and the performance of state of the art LLMs in reasoning problems.

2 Context

2.1 On Intelligence

Intelligence, even though central to the evolution of the human species, has hitherto remained an elusive concept. Many great minds among them artists, philosophers and scientists have attempted to describe, analyse and pinpoint the nature of intelligence but until now there has been no general consensus on what intelligence is. R.J. Stenberg once said that *"Viewed narrowly, there seem to be almost as many definitions of intelligence as there are experts asked to define it"*(Gregory 1998) . This issue has become even more complicated with the advent of computers in the 1940s when questions regarding the nature of biological types of intelligence were further compounded with questions of a new type of intelligence this time artificial (AI). The new conundrum can be summarised as follows: "Since we can create machines that can perform calculations and execute logical operations, is it possible that they too, can display signs of intelligence?"

Questions like this were raised early on in a seminal paper by English mathematician and computer scientist Alan Turing called "Computing Machinery and Intelligence" and published in 1950 (Turing 1950) which opens with the now famous lines "*I PROPOSE to consider the question, 'Can machines think?'*. In that same paper Turing addressed the need to establish methods of evaluation of these "thinking machines" and a conceptual experiment was described under the name "The imitation game". This has come to be known as the "Turing test" and it purportedly can decide whether a machine possesses human-level intelligence. As we shall see however, things are not so simple and despite the fact that the abilities of computers have come a very long way since Turing's time and admittedly the Turing test has now been surpassed, it is still debatable to what degree these machines are in possession of true intelligence.

More recently, a vast focus of attention has been enjoyed by the field of generative Artificial Intelligence, ushered in by the spectacular achievements of Large Language Models (LLMs). Renewed claims have been made that we are now getting close to machines attaining human levels of intelligence (Bubeck et al. 2023), often referred to as AGI or artificial general intelligence, however the truth is that we cannot evaluate such

claims as we do not yet possess the right tools or methods to measure in a precise way intelligence or the progress that we might be making towards AGI. This is a testament to a skewed development of the field which needs to be addressed. It is therefore very important for a greater effort to be concentrated in the establishment of rigorous definitions and benchmarks which can quantify our progress towards increased levels of machine intelligence.

Many notable attempts have been made in the direction of addressing the situation. In the psychological domain, Legg and Hutter have made a survey of no less than 70 definitions of intelligence (Legg & Hutter 2007) and have attempted to consolidate them in a single statement as follows: *"Intelligence measures an agent's ability to achieve goals in a wide range of environments"*. Sternberg and Detterman have also attempted to establish points of general agreement by bringing together and analysing the opinions of two dozen prominent experts in the field of intelligence in their book "What is intelligence" (Sternberg & Detterman 1986).

What seems to emerge from these studies and exemplified in Legg and Hutter's definition, is that there are two distinct elements that are often encountered in isolation, but sometimes also in unison, in most characterisations of intelligence. The first one points to the acquisition of task-specific skills ("ability to achieve goals"). The second one relates to a potential for generalisation and adaptation ("in a wide range of environments"). Under this light an intelligent agent should be able to display problem-solving skills across a range of tasks and under a variety of settings and conditions, even when these tasks are totally novel to them.

Early AI research has relied heavily on the first of these two axioms, since the prevalent view at the time was that intelligent systems could be created by explicitly programming distinct and relatively static functions and routines, that can execute logical operations with access to knowledge stored in a database-like memory. Marvin Minsky, who was a proponent of this thesis, has famously defined AI as *"The science of making machines do things that would require intelligence if done by men"* (Minsky 1968). This clearly puts the emphasis on the performance of the task, neglecting any reference to learning or adaptation. IBM's Deep Blue chess playing computer (Campbell et al. 2002) is probably

the best known example of a system displaying that philosophy by achieving a high level of domain-specific skill. Is that however enough to characterise this system as intelligent?

The resurgence of connectionist theories in the 1980s stimulated by vast improvements in hardware have culminated with deep learning techniques which have come to dominate the field. This development has moved the focus onto the second axiom, the one that favours generality and adaptability to different domains and settings. Neural networks exemplify uniquely this ability of systems to start as a blank-slate (a tabula rasa of randomly initiated weights) and which are "trained" on a skill that can vary, depending on the problem at hand. However, in most cases the outcome is again a system that is overly skill-specific, restricted by the dataset used during training and frozen in time during inference. Regardless, due to its success, this view has become all-pervasive and as a result many AI researchers are now convinced that achieving increased intelligence will be the natural outcome of just scaling this technique to ever greater heights.

2.2 Evaluating Intelligence

Evaluation of intelligence is essential in order to identify and measure progress in the field of AI. The two distinct conceptualizations of intelligence that we have mentioned, shape to a large degree the types of evaluations performed in either biological or artificial systems (Hernández-Orallo 2017*b*). We often make the distinction between narrow AI as opposed to general AI depending on whether a system is constructed to perform well in a narrow and specific task, or whether it is able to undertake more general tasks that it has never encountered before. As a result, evaluations of these systems vary accordingly.

Since narrow AI systems are overly-specialised, this leads to an evaluation approach that needs to be skill or task-oriented. In principle, according to José Hernández-Orallo (Hernández-Orallo 2017*a*), what we actually end up measuring in these systems is performance and not intelligence. In the example of the Deep Blue chess-playing computer mentioned previously, the system has been developed to attain above-human level performance in chess, which is a very narrow and specialised skill. Evaluation of this system was famously done by having it compete against the then world chess champion Garry Kasparov. The system's performance in this task was astounding. However it

would be a mistake to regard Deep Blue's achievement as a sign of intelligence, since what is most likely on display here is the intelligence of the scientists and engineers behind it.

Narrow AI evaluations can take a few different formats with human review, as in the case of Deep Blue, just one of them. The most prominent among them however is benchmarks. Benchmarks have played a very important role in promoting progress in the field because they are for the most part fair (same rules and datasets used by all), reproducible, scalable, and relatively easy to set up. A great deal of progress in AI, especially in recent years, has come from competition among teams trying to improve the performance of systems in a benchmark score (think of MNIST, ImageNet, Kaggle or the DARPA grand challenge for autonomous driving).

On the other hand, evaluations pertaining to general AI measure ability and not performance (Hernández-Orallo 2017*a*). Ability is the property of a system that enables it to perform well in handling situations or tasks that it has not encountered previously. Evaluation in terms of abilities is much more challenging than task-specific, performance evaluation and as a result progress in this area is still at a very early stage. Since the focus of AI has been increasingly directed on designing systems that are capable of broad generalisation abilities, it is imperative that appropriate evaluation methods are devised in this area too.

One promising way of evaluating general AI is through psychometrics. Psychometrics is a well established field in psychology that is concerned with measuring broad cognitive abilities, personality traits and other psychological properties (*Handbook of Intelligence* 2000). Psychometric tests work by assigning a large number of tests on a broad set of tasks and aggregating the results using probabilistic methods. These tests have been used for many years in humans, producing reliable and reproducible results. A joint index that is derived from some of these tests is the well known IQ or Intelligence Quotient number.

Some fundamental aspects of psychometric tests that allow them to produce reliable results are that a) they should not be known to the test taker in advance b) they should incorporate questions with a broad difficulty range and c) they should ideally be agnostic

to the test subject's ethnic and cultural background through the use of abstract exercises. IQ tests are widely accepted and administered in a variety of educational, corporate and government settings. It has variously been suggested that psychometric tests could be used to evaluate AI systems, either in their original format (as suggested for example by Bringsjord et al. under the name "Psychometric AI" (PAI)) (Bringsjord & Schimanski 2003), or extended to any type of intelligent system (as in the case of José Hernández-Orallo's "Universal Psychometrics" (Hernández-Orallo et al. 2014)). A third, very interesting approach to using psychometrics to evaluate AI has been recently proposed and is described below.

2.3 The Abstraction and Reasoning Corpus (ARC)

In a seminal paper called "On the measure of Intelligence" (Chollet 2019) and published in 2019, software engineer and AI researcher François Chollet uses the No Free Lunch theorem (Wolpert & Macready 1997, Wolpert 2013) to infer that "*the very idea of a "valid" measure of intelligence only makes sense within the frame of reference of human values*". This is reminiscent of the famous thesis of Protagoras (490–420 BCE ca) who claimed that "*Of all things the measure is man: of those that are, that they are; and of those that are not, that they are not*" (Bonazzi 2023). Chollet proceeds to argue that any attempts at creating general level intelligence in AI systems should be defined, measured and developed against a human-like standpoint. In this respect, psychometric tests which by their nature derive from and are limited to the sphere of human experience are ideal in providing the right evaluation tools for this purpose.

Chollet uses insights from developmental psychology to deduce that human knowledge priors form a critical part of any measurable form of human-like general intelligence. These knowledge priors correspond to core elements that have been transcribed into our DNA through natural evolution and which constitute the foundations of human cognition. They are part of the developmental science theory of Core Knowledge (Spelke & Kinzler 2007). The four main categories of these innate assumptions are:

- Objectness and elementary physics which relates to how humans perceive objects as discrete elements in their environment and the elementary interactions that are physically possible between them.

- Agentness and goal-directedness which enables humans to recognise which objects in their environment have agency and which are inanimate and to identify actions that indicate specific goals or intentions among the "agents".
- Natural numbers and elementary arithmetic which enables humans to have an inherent representation of small numbers and to perform basic arithmetic operations of addition, subtraction or comparison.
- Elementary geometry and topology which enables a rudimentary facility for orientation in human beings by identifying direction, distance and relative position (in/out, above/below) relationships among objects in our immediate environment.

Based on these broad intuitions, Chollet proceeds to formalise these key concepts into his own definition of intelligence as follows:

> "*The intelligence of a system is a measure of its skill-acquisition efficiency over a scope of tasks, with respect to priors, experience, and generalization difficulty.*"

The advantage of this definition is that it can be formalised mathematically, it is actionable in the sense that it expresses intelligence as an optimization problem and it allows the construction of a solid foundation for the evaluation of general intelligence in AI.

In the same paper, Chollet proceeds to introduce the Abstraction and Reasoning Corpus (ARC, sometimes denoted and as ARC-AGI). ARC is a suite of tests that act as a benchmark for evaluating general artificial intelligence, and which attempt to incorporate most of the considerations and criteria mentioned previously. These tests are similar to Raven's Progressive Matrices (John & Raven 2003), a set of non-verbal tests used to measure general human intelligence and abstract reasoning that date back to the 1930s.

ARC comprises a training set and an evaluation set consisting of 400 and 600 tests respectively. Out of the evaluation set 400 tests are public and the rest 200 are private. Each test was generated manually and is unique. The tests consist of sets of input and output grids. The size of the grids varies for each test, but they range from 1×1 up to 30×30. The grids are made up of pixels which can have any of 10 different colours. The

colours enable the design of various shapes and patterns in the grid with the black colour representing the background. The solver is presented with test examples which are pairs of input and their corresponding output grids. The number of examples offered can vary for each task (3.3 on average). The solver is shown the input grid of the test and is asked to generate the output grid based on the logic that they have deduced by observing the relationships between the inputs and outputs of the examples. Note that no information is provided regarding the test output grid, not even its size. The solver has to decide on the size of the grid and the colouring of all the elements that make it up. Three examples of ARC tasks are shown in Figure 1. In these examples one can fairly easily discern that example (a) relates to the concept of directionality, example (b) relates to basic counting and example (c) relates to the notion of enclosed space.

2.4 Current state of the ARC

After publication of the paper by Chollet in 2019, a Kaggle competition was set up with a total prize of $20,000 which ended in May 2020 (Chollet et al. 2020). The winning submission managed to successfully solve only about 20% of the tasks in the test set. All of the top-3 winners have deployed a similar approach. They created a Domain Specific Language (DSL) which is able to encode into functions the different types of transformations that can be deduced from the training set (these could be things like change colour, move, mirror etc.). Then a heuristic search algorithm is used to go over these transformations and select the series of transformations that produce the best candidates based on the test examples.

Noticing that in the four years since the launch of ARC it had not been possible to somehow game the tests and in this time the state of the art in ARC-solving programs was coming to around 30% (obtained by combining the top solutions), Chollet, this time partnering up with silicon valley entrepreneur Mike Knoop, decided to raise the stakes. The ARC Prize 2024 was announced in June 2024 (Chollet et al. 2024). This time the prize is $100,000 with an extra $500,000 prize unlocked to the team that manages to achieve a score above 85% in the leaderboard. The 85% marker was decided after a study from New York University (Johnson et al. 2021) showed that on average most humans can solve 84% of the tasks in the ARC training set, therefore a

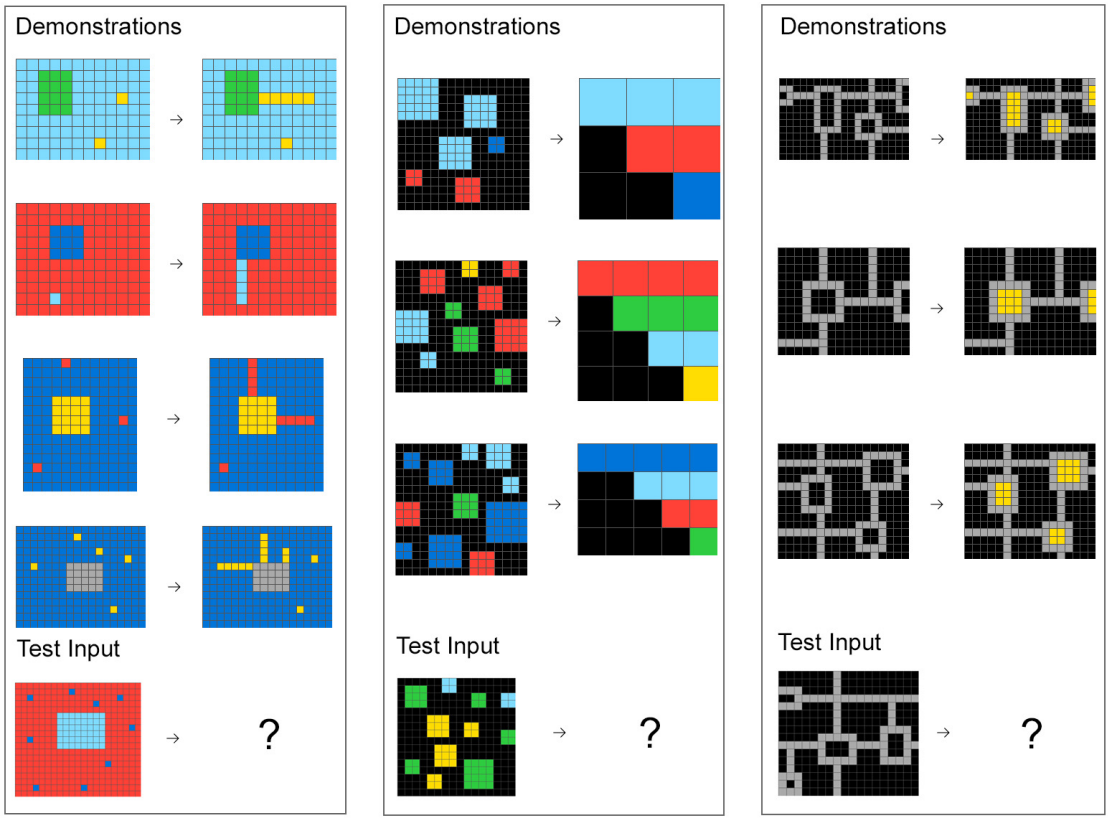


Figure 1: Examples of ARC tasks. Each task has a set of demonstration input-output pairs that illustrate the grid-transformation rule. The number of demonstrations can vary between tasks. A test input is then given. The goal of the solver is to figure out the underlying rule of the transformation used in the demonstrations and then apply it to the test input to produce the resulting test output grid. The size of the output grid is not given.

solution that matches that would approximate human level of general intelligence. The competition deadline is November 2024 but will roll over to the next year if the 85% threshold is not reached. The purpose of the competition is to stimulate research in AGI as Chollet and others believe that the dominance of Deep Learning techniques has in fact stalled progress in this direction. As of the time of this writing (October 2024) the top team in ARC-Prize 2024 has achieved a score of 54.5%. This marks a substantial progress since the launch of the competition just a few months before and this is a testament to how competitions of this kind can stimulate research and experimentation.

The current leading solution (which will become publicly available upon the end of the competition) uses some established methods but then builds on top of them with some novel techniques (Cole 2024). Firstly, a classic Natural Language Processing (NLP) type AI model is trained and fine-tuned specifically on solving ARC tasks. However, they are substantially increasing the size of the training dataset using augmented and synthetic data. The idea here is that an AI model that will be able to solve ARC tasks must receive a strong general grounding in concepts that establish the core knowledge principles around which ARC is built and the number of examples provided in the public ARC training and evaluation dataset is clearly very small. The leading team has found that OpenAI's GPT-4 (OpenAI et al. 2024) can be very useful in writing code that is able to generate synthetic ARC-like puzzles, thus helping a lot in that direction. Augmented data on the other hand is made up of slight variations of the puzzles already found in the public dataset.

In addition, a method akin to test-time fine-tuning is deployed. The way this works is that during test time, the training pairs provided are used to generate a further number of slightly modified examples. These augmented examples are used to perform a short fine-tuning run just before the final prediction is given by the model on the test set. This transforms the problem from a few-shot (just three or four examples) to a many shot task. This has been found to increase the accuracy of the solution provided by the model as in many cases the model can be very close to the right answer, but is just off by a little. Test-time fine-tuning enables the model to have more chances in selecting the right answer out of a few close related possibilities.

It is also worth mentioning another approach (Greenblatt 2024) which has achieved a 50% performance in the public evaluation set of ARC. This approach could not be tested against the private test set and thus be a contender for the official ARC prize because it violates two of the rules that have been set for the competition. First, it requires internet access to use the GPT-4o Application Programming Interface (API) which is not allowed, one of the reason's being that this could leak the private test set and thus invalidate the whole project. The second reason is that this solution requires more than 12 hours of computation time, which is the upper limit set in the competition for any team.

Nevertheless, the reason why this approach is interesting and it is worth mentioning

here, is because it does rely heavily on the use of an LLM (specifically OpenAI's GPT-4o) through prompting. In principle this approach is based in generating very long and detailed prompts describing the example input and output grids as well as supplying an image version and multiple textual representations of them. The LLM is asked to generate numerous (thousands) Python programs that can perform the transformations shown in the example. An intricate method of selecting and vetoing the most promising programs is applied and some further prompting enables the top three programs to be selected and then used on the test tasks. Even though effective, this method could be considered a brute force method as it works by applying huge computational resources to number-crunch the problem by looking at the space of all possible solutions. This is reminiscent of the Deep Blue chess playing program, which relied on a vast look-ahead search of board configurations. As mentioned previously it is contested whether this type of solution could be considered as a true testament of intelligence and for the time being it remains an interesting but not valid approach for solving ARC.

2.5 ARC and LLMs

As we have already mentioned a lot of research interest is currently focused on ARC and LLMs and a significant number of publications have been released in this area. Three of these have been very influential in the direction of the analysis presented in this report. The first one by Xu et al. (2024) tests two LLMs against a subset of ARC tasks, in a similar fashion to what we do. In fact we will be utilizing the same subset of 50 ARC tasks that they have used, so that our results can be directly comparable to theirs. They also look into the question of how best to represent the ARC dataset so that it is suitable to the textual nature of prompts in LLMs. We will be also addressing that question, however we will be going further by expanding on the number and types of encodings that we will be testing on. Additionally we will be relying on a more diverse and up-to-date selection of LLMs for performing our tests.

Some of the contributors in the aforementioned paper also appear in the paper by Xu et al. (2022), which describes a novel method called Abstract Reasoning with Graph Abstractions (ARGA). This is an object-centric framework that first represents images of the ARC grids using graphs and then performs a search for a synthesized program in

a Domain Specific Language (DSL) that is based on the abstracted graph space. Even though this is a complete framework for generating ARC solutions, Xu et al. (2024) as well as we in this present analysis have used just the first part of this tool-kit, the graph abstraction component, in an attempt to boost the performance of LLMs on the ARC tasks by enabling them to identify objects and their relations.

Finally, Mitchell et al. (2023) perform tests on LLMs using text as well as visual prompting through the use of images on one of the first multimodal versions of ChatGPT, called GPT-4V. The idea here is that instead of textual prompts containing the ARC tasks, actual images of the grids are given to the model. This is certainly an interesting development and we will also be performing an analysis using a more recent version of GPT-4V called GPT-4o to see how our findings compare to theirs.

It is also worth noting that a number of ARC-like datasets have been generated in an effort to untangle the complexity of the original ARC dataset. The Mini-ARC (Kim et al. 2022) is a 5×5 compact version of the ARC that offers enhanced usability for model training and experimentation. The ConceptARC dataset organizes tasks around "concept-groups" which focus on specific concepts that vary in complexity and level of abstraction (Moskvichev et al. 2023). The authors performed tests on humans as well as the top ARC solvers from the original Kaggle competition, but also on OpenAI's GPT-4. Finally, Xu et al. (2024) create a simplified version of ARC, the 1D-ARC consisting of one-dimensional (array-like) tasks, in order to investigate how the sequential nature of text in textual encodings of the 2D grids can hamper the ability of LLMs in solving those tasks.

2.6 Multimodal LLMs

In the last couple of years a new development in generative AI has been the introduction of Multimodal Large Language Models or MLLMs. Multimodal models are able to process multiple types of data (modalities) such as text, audio, images and video. This allows them to better mimic human sensory abilities since they can combine and integrate information from different modalities. OpenAI's GPT-4V, where V stands for vision, was one of the first popular models to incorporate image analysis capabilities into a large

language model. It uses an architecture that combines computer vision together with natural language processing (NLP) which allows it to provide answers to questions about image content, perform object detection and even read text within images using Optical Character Recognition (OCR).

Since ARC tasks are primarily visual, it is only natural that with the advent of multimodal LLMs one would be quite interested to find out how they would perform when the tasks are fed to them visually as opposed to textually. The short answer is that they perform badly (for now..). The authors of *"Comparing Humans, GPT-4, and GPT-4V On Abstraction and Reasoning Tasks"* (Mitchell et al. 2023) have found that the multimodal version of GPT-4 (GPT-4V) performs substantially worse than the text-only version with an accuracy of 0.25 compared to 0.69 for the textual version and 0.95 for the human volunteers that attempted to solve a subset of ARC tests. Despite some more promising results obtained when using visual CoT which is an extension of CoT to multimodal LLMS (Singh et al. 2023), it seems that vision models are still at an early stage in their development to be able to provide robust results especially in taks for this kind.

3 Methods

3.1 Dataset

To perform the analysis in this report we first need to access the dataset containing the ARC tasks. The dataset for ARC can be found in two different formats. The first one was created for the original Kaggle competition of 2020 and this has been modified to a newer version for the current ARC Prize 2024. They both use .json files for storing the data. JSON stands for JavaScript Object Notation and it is a lightweight text based data interchange format that has the advantage of being easy for humans to read as well as easy for computers to parse and generate. In JSON files data is stored in key-value pairs and is separated by commas. Curly brackets are used to hold objects and square brackets are used to hold arrays. Each task is identified by a unique hexadecimal number. An example of the JSON encoding of an ARC task can be found in Appendix A.

In our analysis we have relied on the more recent dataset of 2024. In this dataset, all training tasks are found in one .json file (`arc-agi_training_challenges.json`) which contains the training inputs and outputs but only the test input(s), with the corresponding test output(s) found in `arc-agi_training_solutions.json`. Respectively the evaluation tasks are all found in `arc-agi_evaluation_challenges.json` and the evaluation solutions in `arc-agi_evaluation_solutions.json`. The publicly available dataset consists of 400 training and 400 evaluation tests.

We have decided to limit this analysis to a subset of 50 relatively simple ARC tasks for two reasons: Firstly the cost of prompting the LLMs on all tasks would be very high considering the number of experiments that we needed to execute and the number of LLMs we wanted to test against. Secondly, we decided to focus on the exact same subset of tests used in the Xu et al. (2024) paper in order to be able to make a direct comparison against their results. The subset of ARC tests we deployed in this study can be found in the Github repository for this study under the datasets folder and called `50_challenges.json` and `50_solutions.json` for the tests and the solutions respectively.

3.2 Choice of models

The choice of how many and which models to test against was made using three criteria:

1. Establish iterative progress. One of the aims of this study is to identify whether LLMs display signs of improved reasoning skills as these are reflected in their ability to solve ARC tasks. To do this we need to test on a range of models that span over a number of generations. Testing on three successive versions of OpenAI's GPT models has allowed us to achieve this. It should be noted that in current LLM progress terms the distance separating the older generation from the state of the art is in the order of a few months.

2. Include multiple LLM providers. Even though there is a cost associated by adopting more than one LLM providers, we decided that including models from multiple sources would give us a more nuanced estimation of LLM performance. We have settled for using models from OpenAI and Anthropic. More providers would be even better, but cost considerations made this impractical.

3. Experiment on state of the art models. One of the areas where our study goes beyond what was presented in the Xu et al. (2024) paper is by adopting models that are more advanced compared to the ones they used. We are therefore able to obtain improved and more up to date metrics on the performance of LLMs on ARC.

To satisfy all of these considerations and also taking into account cost and time limitations, it was decided to perform the tests on three models from OpenAI (ChatGPT-3.5 turbo, ChatGPT-4o and o1-preview) and one model from Anthropic (Claude Sonnet 3.5).

It is worth mentioning here a few things specifically regarding OpenAI's o1-preview. This model was made available in September of 2024 and it was touted by OpenAI as a model that "significantly advances the state-of-the-art in AI reasoning" (OpenAI 2024). Even though the details of how it works remain undisclosed, it looks like this model represents a paradigm shift in how large language models operate. From what we know, o1-preview is using reinforcement learning combined with Chain of Thought (CoT) methods during inference time (we will be looking closer at CoT in chapter 3.4). This means that the model is able to develop and follow many different reasoning pathways

towards an answer when presented with a prompt and then applies an algorithm to select the answer that seems more probable to be correct. This is reflected in the significantly longer time it takes for the model to produce an answer compared to classic LLMs. According to the company (OpenAI 2024) the performance of the model consistently improves with more computation devoted in train-time and test-time, which is a departure from the classic scaling laws involved in LLM pretraining. It will be very interesting to see how this approach evolves in the future.

3.3 Direct-grid encodings

As we have already mentioned, the ARC tests are in principle visual in nature. Each image is a 2D grid of a size that ranges from 1×1 up to 30×30 and where each pixel or cell has one of 10 different colors with black representing the background. Since LLMs respond to textual input (see Section 2.6 for a discussion of more recent multi-modal LLMs) we need to first convert the images into an appropriate textual representation before sending it to the LLM. There are many possible ways one can represent ARC tasks in a textual manner. However the two main questions that need to be addressed are how to represent the actual pixels and the second is how to represent the ordering of the pixels in the grid and their grouping into distinct rows and columns.

In a manner that is similar to the Xu et al. (2024) analysis we have decided to apply two different methods: one is to represent each pixel or cell by a number corresponding to its assigned colour value (we followed the numbering convention used in the original JSON encoding of 0 to 9 where 0 represents the colour black). The second is to represent each cell by the string value of the colour. To further be able to represent the grid and the ordering of elements within it we have experimented with two distinct ways for the numerical representation and two for the string representation. For the numerical representation we have tried a Compact representation where no delimiter is used between numerical values and each row starts with a new line. We have also used the pipe symbol "|" as a delimiter inserted between each number. For the string representations we have tried separating the string values using either the pipe "|" or comma "," delimiters with different rows identified by a newline character.

Representation	Formatting	Example Grid
Number	Compact	200 200 000
Number	Pipe delimited	2|0|0 2|0|0 0|0|0
Number	JSON	[[2, 0, 0], [2, 0, 0], [0, 0, 0]]
String	Comma delimited	red,black,black red,black,black black,black,black
String	Pipe delimited	red|black|black red|black|black black|black|black
String	JSON	[["red", "black", "black"], ["red", "black", "black"], ["black", "black", "black"]]

Table 1: The six different types of textual encodings of the ARC tasks used in our direct-grid tests. The top three are using numbers for the representation of coloured cells and the bottom three use strings. The number JSON and string JSON representations introduce a structured format that can potentially have advantages in terms of the ability of the LLM to parse and process.

Finally, as an extension to the methods presented in Xu et al. (2024) we have also adopted a JSON encoding that is similar to how the ARC dataset is presented in its original format. We have used the JSON encoding both with numbers and also strings (note that string values in JSON format need to be enclosed in double quotes). Examples of all the different methods used in our encoding of ARC grids are shown in Table 1. Figure 2 shows how an ARC image can be translated into a textual representation that either uses a number compact formatting (left) or a string formatting with the pipe delimiter (right).

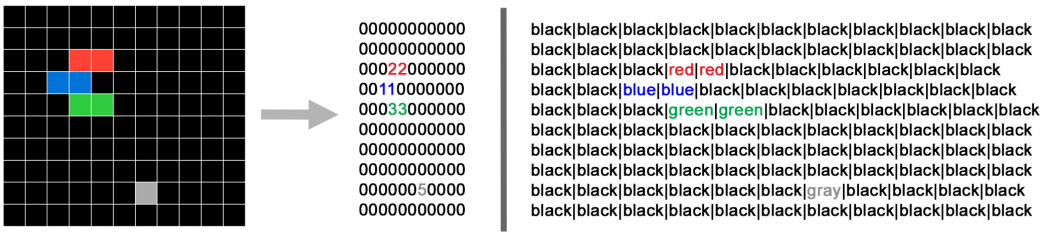


Figure 2: Textual encoding of an ARC grid using numbers or strings. On the left side each pixel is represented by a number from 0 to 9 corresponding to a specific colour and uses the Compact formatting of Table 1. On the right, each pixel is represented by the string name of the colour with each value separated by the pipe delimiter "|". Different rows are indicated by new lines in the text. The colour highlights in the text have been added for easier reading.

3.4 Prompt engineering and CoT

After encoding the ARC images into textual representations, the next step is to incorporate them into prompts which will provide the basis of instructions to the LLM on how to execute the tasks. Prompting in LLMs can take place as a single question-answer interaction (called single turn or standard prompting) or it can have a conversational character where there are a number of exchanges between the user and the LLM and each builds on the context of the previous ones (referred to as multi turn prompting). The latter is more suitable when building interactive agents or chatbots or when the context of a conversation needs to be maintained over multiple interactions. This analysis relies on single turn prompting since the process of posing an ARC task to the LLM and receiving

an answer can be completed as a one full interaction.

Further to this, the structure of the prompt can have a distinct effect on the quality of the response returned by the LLM. Prompts can be either of zero-shot type, where the model is asked a direct question without any prior examples or demonstrations, or they can be of few-shot type, where a few examples of the task or behaviour are provided before the model is asked to respond to a new input. By their nature ARC tasks rely on first presenting a few examples which demonstrate the transformation that the test-taker is asked to identify. Therefore prompting LLMs for ARC tasks falls by definition under the few-shot category. The provision of the examples also allows one to demonstrate to the model the desired format of the model's answer, which can be quite important for the purpose of parsing and further processing of the response. Appendix B contains the prompt template that was used in our direct-grid experiments.

Another technique that has recently garnered a lot of attention is the Chain of Thought (CoT) technique (Wei et al. 2022). CoT encourages step-by-step reasoning by breaking down complex problems into a series of smaller intermediate logical steps, rather than attempting to jump directly to the final answer. This has been shown to work not just for few-shot prompting, but it has yielded positive results in zero-shot reasoning tasks too (Kojima et al. 2023). CoT can also be applied by offering further training examples to the model which will provide it with useful indicators as how to approach a task. CoT can be especially effective for tasks that require multi-step reasoning and where outlining intermediate steps can be conducive to extracting the correct answer to a problem. CoT finally allows us to gain insight into the process leading to certain outcomes and thus enables transparency.

In our analysis we have taken the textual encoding methods that have provided the best results in our direct-grid tests and tried to enhance them by applying the Chain of Thought technique. This was done in various ways, which involved building an extended prompt that contained multiple examples of grid transformations together with the logic of those transformations as well as coercing the model to think in a step by step fashion and to break down the task at hand in smaller actionable parts. Samples of the prompts that were used in our CoT analysis can be found in Appendix C.

It is also worth mentioning one important parameter used when prompting LLMs, called the temperature setting. The temperature of a model determines how random (or creative, depending on how one looks at it) the LLM responses can be. A model temperature can range from 0 (lowest) to 1 or even higher. A temperature close to 0 makes the model behave in a deterministic way, which should translate in the model producing the same output for a specific prompt. For our testing we set the temperature for all models to 0 except for o1-preview which has a different specification for temperature due to the different nature of the model. For the other 3 models our experiments have shown that even setting the temperature to 0 could often produce results with slight variability. To account for this, we have run each experiment three times and used the average value of those runs for our results.

3.5 Visual Prompting

Following in the footsteps of Mitchell et al., we have also attempted an exploration of visual LLMs in our analysis. We have used OpenAI's GPT-4o which is the successor of GPT-4V when it comes to visual models. We generated images containing the example inputs and outputs as well as the test input grid and asked the LLM to analyse the input.

In prompting with images it is quite important that the quality of the image is adequate to convey the details of the objects and text contained within it. We have generated our images using 12×8 inch images with a 300 dpi analysis (3600×2400 pixels) in a PNG format for maintaining accuracy. We have started the tests by specifically concentrating on those tasks that GPT-4o manages to solve in our tests using a textual encoding. We have uploaded the images to the model and asked it to analyse the image, identify the transformation, apply it to the test input and give back the results in a textual format. An example of the image used in our multimodal tests is shown in Figure 3 together with the output form GPT-4o.

The results of this analysis have shown that the performance of GPT-4o in solving ARC tasks in visual prompting is very poor. This is in complete agreement with what is found in Singh et al. (2023) and Mitchell et al. (2023). Out of 10 tasks that GPT-4o had

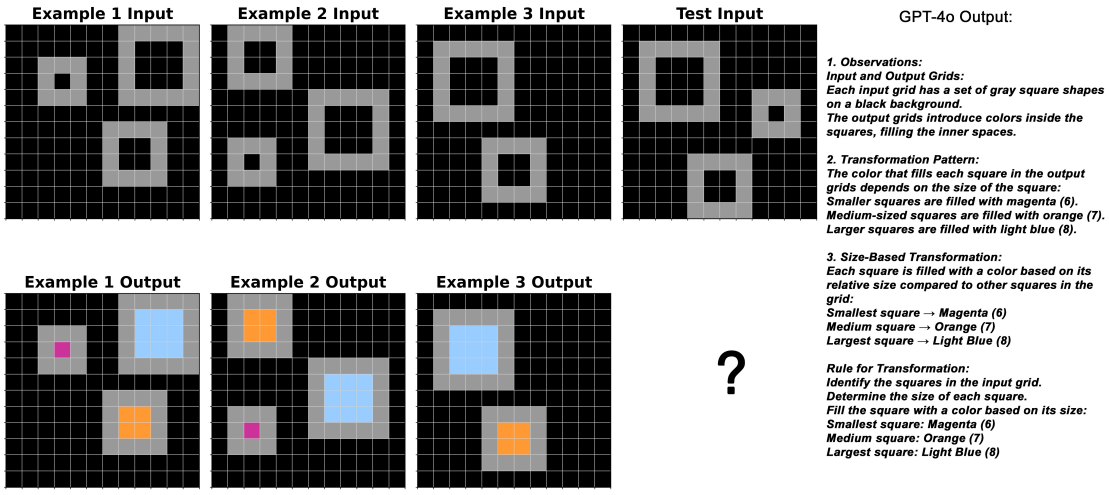


Figure 3: Examples of a visual ARC task given to GPT-4o. The model successfully analyses the image and identifies the transformation pattern as can be seen in its textual output. However, it fails to produce the correct output grid. In this example it produced a 15×8 grid with many incorrect color values

successfully solved in textual mode, the model was unable to provide even one correct solution. In a few cases it could correctly identify the transformation from the examples, but then it would produce the wrong output when asked to give the detailed output grid description. It would either provide the wrong output grid dimensions or the colour values of the pixels would not correspond to what the transformation that it described would actually produce. This indicated to us that this analysis was not worth pursuing any further at this stage and we have therefore refrained from executing this test on all the remaining tasks in our dataset.

3.6 An Object-oriented approach

Despite iterative improvements, LLMs still exhibit a limited ability at solving ARC tasks when these are directly encoded in a textual format as it will also be shown more analytically in our results section 4.1. We have also seen that they currently perform quite poorly when those tasks are presented in a visual format. Recent research has looked into ways of improving the performance of LLMs and addressing some of their shortcomings (Lei et al. 2024, Xu et al. 2024) by resorting to external tools. These tools are aimed

at trying to enhance in artificial systems some aspects of the core knowledge priors that as we discussed in 2.3 are so important in making up what we call human intelligence .

More specifically, the element of objectness, i.e. the ability to perceive individual objects and the relationships between them is fundamental for tasks like ARC, since as Acquaviva et al. have shown, when humans describe in words their mental process of solving ARC tasks to other humans, half of the phrases they use reference objects. An approach that promotes an object-oriented understanding of the ARC tasks can therefore be very beneficial for LLMs. We have decided to explore this path in our analysis and to this end we have deployed the Abstract Reasoning with Graph Abstractions (ARGA) tool (Xu et al. 2022) freely available at https://github.com/khalil-research/ARGA-AAAI23.

ARGA is a framework that has been specifically designed for solving ARC tasks. It consists of a two-stage process. During the first stage ARC tasks are abstracted into a graph format. This was inspired by work on the game of Go (Graepel et al. 2001), where the 2D positions of Go stones are mapped to undirected graph representations that capture information about the objects formed during the play at a higher level of abstraction. Similarly in the case of ARC grids the motivation is to move one level up in abstraction and instead of manipulating individual pixels to work with groups of pixels simultaneously, considering them as part of the same object, thus significantly reducing the number of possible configurations in the grid. This intuition is further enhanced by the observation that in many of the ARC tasks, clusters of same-coloured pixels act as independent objects.

The second stage of ARGA is the implementation of an ARC solving algorithm which shares many common elements with some of the state of the art ARC solvers described in 2.4. It implements a domain specific language (DSL) built upon the objects and relations defined during the abstraction stage that can describe and execute specific graph transformations. It then uses an exhaustive tree search for the sequence of transformations (programs) that can produce the correct outputs for each of the inputs in the task demonstrations thus synthesizing a solution. An example an ARGA synthesized solution to an ARC task is demonstrated in Figure 4.

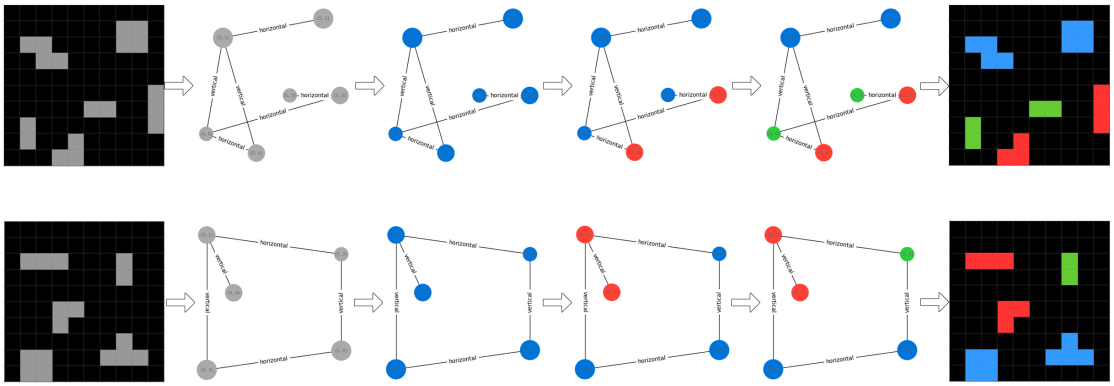


Figure 4: Example solution generated by ARGA. The first step is the abstraction of the grid into an object-oriented graph representation. In this case all adjacent pixels of the same colour except for black (background) are designated as a single object (node). Two nodes form edge relationships between them if and only if any of the pixels in the nodes share the same coordinate value along either axis. The number of pixels that make up a node define its size. The transformation that generates a correct solution in this case comes as a program that has three steps. First, all nodes independent of size are colored in blue, secondly the nodes of size exactly 3 are colored in red and finally the nodes of size 2 are colored in green.

3.7 From grids to graphs

In this study we utilize only the first stage of the ARGA framework, that of object abstraction through the use of graphs. In a graph representation each object is identified as a node and each edge represents relationships between the various objects. Edge relationships can be horizontal and vertical only and are formed when any of the pixels that make up two nodes share the same coordinate value in either axis. In ARGA there are many possible ways to abstract a grid into a graph and the nature of the task dictates which abstraction method is the best to use. Different choice of abstraction methods can lead to different definitions of what constitutes an object in the grid.

Take for example the grid shown in Figure 5. Here we can choose an abstraction method that dictates that "all non-black neighbouring pixels of the same colour form a node" as is exemplified in a). That would lead the two leftmost red vertical bars to be bundled together into a single object which can only be operated upon as a unit. This graph arrangement will not allow the transformations that are required to produce the

correct output grid, shown on the right. However a different abstraction can be selected defined as "all non-black pixels of the same colour *that are vertically connected* form a node". This would lead to a very different set of objects as shown in the graph image in b), where now the two leftmost red vertical bars form distinct objects that can be operated upon independently. This allows the transformation shown on the right to be performed which produces the correct solution.

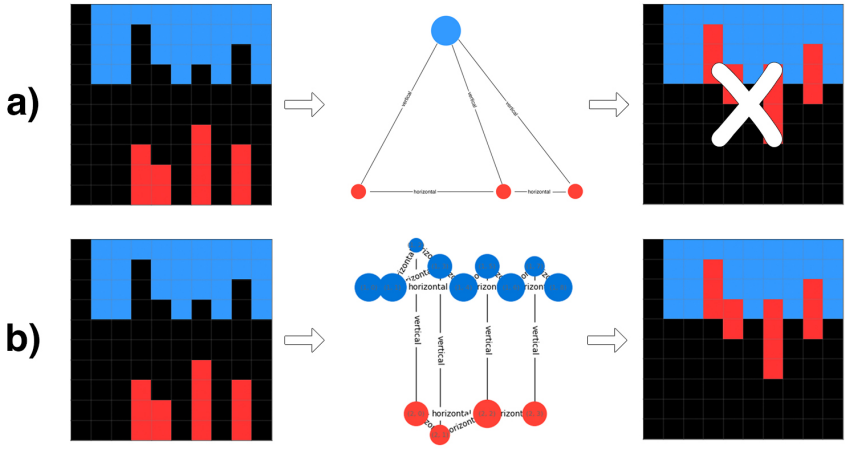


Figure 5: Different graph abstractions define different sets of objects. In the top abstraction a) all adjacent, non-black pixels of the same colour form a single object. This abstraction does not allow the transformations that are required to produce the correct output. In the bottom one b) all the non-black pixels of the same colour that are connected in the vertical axis form an object. This allows the vertical red bars to be acted upon independently of each other, which enables the correct output to be reached

In our analysis we have selected the abstraction to use for each of our tasks based on what the ARGA algorithm deemed to be the optimal choice for synthesizing a solution. As it turned out in 84% of the tasks in our dataset the optimal abstraction to use was the "get_non_black_components_graph", which forms a node for each set of neighbouring pixels of the same colour that are not black. The procedure of generating a graph from a grid is shown in Figure 6. In the first step the grid is translated into a set of coordinates with the origin (0,0) situated at the top left corner. In the second step the selected abstraction method is applied which creates distinct objects depending on its rules. In this case all red pixels form the first object and all cyan pixels the second. The two objects

have an edge relationship since some of the pixels in the two objects share coordinates in the vertical axis. Once the graph is generated, then it can be encoded in a textual format as shown on the right. Note that in the ARGA coordinate system the first number denotes the position in the vertical axis and the second the position in the horizontal axis.

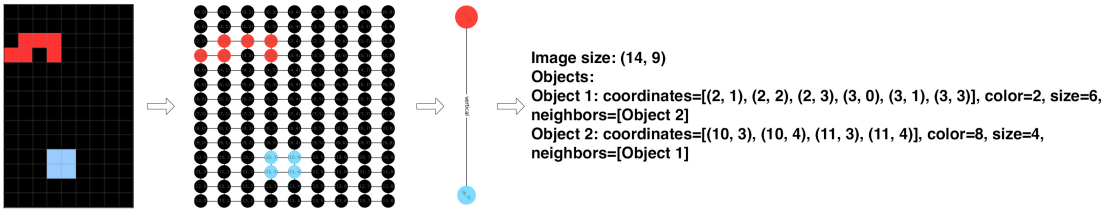


Figure 6: Example of the transformation sequence from a grid to a graph and then to a textual representation of the graph. The textual encoding contains the size of the image, the list of objects in the graph, the coordinates of each cell making up an object, the size of the object (number of cells), its colour and whether it is related to any other objects.

For each of the 50 tasks in our dataset we generated the corresponding graphs using the abstraction method that was optimal at producing a solution in ARGA. These graph representations were then encoded into 5 different types of textual format and these encodings were in turn used in our prompt to the LLMs. The 5 encodings are: plain text encoding with and without edge information, JSON formatted encoding with and without edge information and JSON formatted encoding using strings for the colours. Examples of all the 5 different encodings can be found in Appendix D. The prompt was formatted using the same phrasing as that used for our direct-grid tests, the only thing that was different was the object-based representation as opposed to a direct-grid representation. We run the tests on the same subset of 50 relatively easy ARC tasks and on the same 4 versions of LLMs. The results can be found in section 4.3.

4 Results

All the scripts created for this analysis together with the logs of the outputs for each experiment can be found in https://github.com/vitsiozo/ARC. All the code was written in Python. Code from other sources is referenced within the scripts where applicable.

4.1 Direct-grid encoding

For this test we have deployed the six different types of direct-grid encoding described in 3.3. Three encodings use a number representation and three use a string representation. We have used an API to communicate with the LLM, send out the prompts containing the tasks and receive their response. Once we have obtained a response for all 50 tasks in our dataset we mark each response as correct or incorrect by comparing it to the reference solution. In case of http errors, we make a max of three connection retry attempts for each task to ensure that protocol communication issues do not hamper the testing process. Marks are obtained out of 50. A few tasks have two test inputs instead of one. For those we assigned half point for each correct subtask. The results are shown in Table 2.

Direct-grid encoding test results

Encoding	Formatting	GPT-3.5	GPT-4o	Claude-3.5 Sonnet	o1-Preview
Number	Compact	6.0%	13.0%	24.0%	45.0%
Number	Pipe	20.0%	27.0%	47.0%	48.0%
Number	JSON	12.5%	24.3%	55.0%	59.0%
String	Comma	16.0%	19.0%	36.0%	62.0%
String	Pipe	16.0%	18.0%	38.0%	38.0%
String	JSON	16.3%	29.0%	47.0%	**63.0%**

Table 2: Accuracy on ARC tasks using different types of direct-grid encoding across four different LLMs. The highest score obtained is highlighted in bold.

We can see that results for different models can vary considerably depending on the encoding used. Overall it is not clear whether a number encoding has a clear advantage over a string encoding or vice versa as it is very model dependent. OpenAI's o1-preview

achieved the best score in the direct-grid tests with 63% (31.5 tasks out of 50) using the string encoding in JSON format. What seems to be reflected in the results however is that a JSON formatting of the tasks performs consistently better compared to other textual formats. This suggests that the more structured nature of this format makes it easier for the LLM to parse and process the tests.

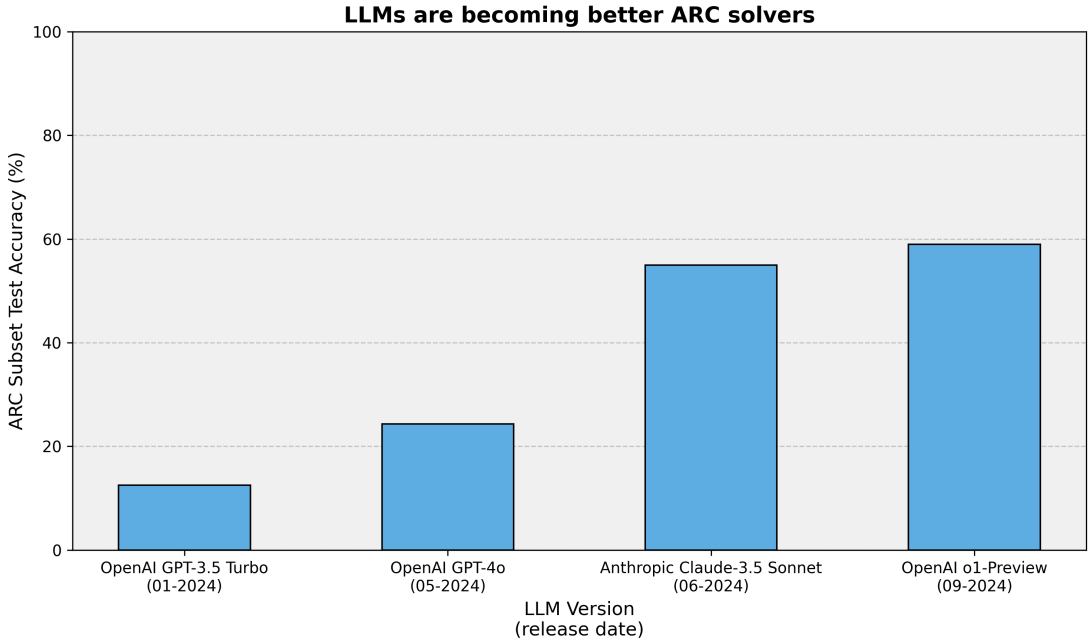


Figure 7: Test Accuracy of direct-grid representations for different LLM versions. This plot depicts the accuracy obtained in the subset of 50 ARC tasks used in our testing from 4 different models using the number JSON encoding. Each new generation of LLM displays an improved ability at solving ARC tasks.

What appears very clearly from this test however is that newer LLMs perform much better compared to older versions and this is true across all choices of task encoding. Based on the results of Table 2, we have created the plot in Figure 7 which depicts the accuracy on the ARC tasks obtained for each LLM version in the numerical JSON encoding. As can be seen in this plot, LLMs are getting better at solving ARC tasks with each new iteration. Averaged over all different encodings, GPT-3.5 Turbo solves $\sim 15\%$ of the tasks in our subset whereas o1-preview solves 52%, marking a significant performance jump. Whether this means that LLMs are becoming better at reasoning is still open to debate and we will present some thoughts on that in the Conclusions section.

4.2 Chain of Thought tests

For our Chain of Thought (CoT) experiments we have confined ourselves to just using the JSON numerical and JSON string encodings. We have also limited our tests to two models, Anthropic's Claude 3.5 Sonnet and OpenAI's ChatGPT-4o. We have deployed three different CoT prompts, each one building on the complexity of the previous one. Examples of all three CoT prompts can be found in Appendix C.

In the first CoT prompt we have simply asked the model to reason step by step about the transformations that it observes in the demonstrations of each task. Then it is asked to apply this transformation to the test input and produce its answer. In the second CoT approach, on top of step by step reasoning, the LLM is presented with an example of a grid transformation together with the logic behind it. In the final CoT test, the example provided to the LLM is extended to include a whole series of grid transformations with the solution and explanation at the end. We have run over the whole subset of 50 ARC tasks using these three methods. The results are presented in Table 3.

Chain of Thought prompting test results

LLM	Encoding	no CoT	basic CoT	CoT example	CoT ext. example
GPT-4o	JSON Nums	**24.3%**	14.0%	22.0%	12.0%
Claude 3.5	JSON Nums	**55.0%**	38.0%	37.0%	34.0%
GPT-4o	JSON String	**29.0%**	22.0%	12.0%	9.0%
Claude 3.5	JSON String	**47.0%**	36.0%	35.0%	30.0%

Table 3: LLM scores on ARC tests using different levels of CoT prompting techniques.

As can be clearly seen from the results, CoT techniques do not improve test accuracy for the LLMs. In fact, adding CoT seems to make LLM performance worse across the board compared to no CoT use. This result does not agree with what is presented in Wei et al. (2022) where CoT is found to improve the ability of LLMs to perform complex reasoning. In our experience adding elements to our prompt led to a detrimental outcome, whereas keeping the prompt simpler worked better. This is certainly an aspect of the study where more thought and research can be devoted in the future.

Another aspect where the CoT testing has yielded some interesting results is the following: By enabling a step-by-step reasoning and asking the model to present its logic in text to us before producing the output, we were able to peer into the "thought process" of the LLM (we use this term in a metaphoric sense). What we have found by doing this was quite interesting. In a significant number of cases (7 out 50 tasks in one instance) the LLM was able to identify and explain the correct reasoning behind the transformation involved in the task, but then it would give an output grid which did not correspond to that reasoning and contained errors. It is also worth pointing out that there was also one instance where the LLM gave the wrong analysis but then produced the correct grid. This discrepancy is interesting and perhaps more research can be devoted also to this area.

4.3 Object-based encoding

For the object-based tests we have deployed the external ARGA tool as described in 3.6. This means that the results presented in this section do not reflect the performance of just the LLMs but that of the combination ARGA+LLM. We have used the abstraction methods from ARGA to convert each of the grids in an ARC task into a graph and then encoded these graphs into object-based textual encodings as described in 3.7. We then incorporated these encodings into a prompt very similar to the one used for the direct-grid tests and executed the tests again via the API. Again the responses were marked out of 50. The results are shown in Table 4.

Object-based encoding test results

Encoding	GPT-3.5	GPT-4o	Claude-3.5 Sonnet	o1-Preview
Object textual	22.0%	42.0%	60.0%	78.0%
Object textual with edge	18.0%	30.0%	58.0%	73.0%
Object JSON	25.0%	42.0%	64.0%	**88.0%**
Object JSON with edge	16.0%	34.0%	67.0%	76.0%
Object JSON String	19.0%	42.0%	62.0%	84.0%

Table 4: LLM scores on ARC tests when using the ARGA object-based tool to generate the task encodings. The highest score obtained is highlighted in bold

We can see right away that the inclusion of object-based information into the prompts boosts the LLM performance in solving the ARC tasks across the board. All versions

of LLM perform significantly better compared to the direct-grid encodings. OpenaAI's o1-preview achieves a remarkable 88.0% accuracy, a 25% improvement over the best direct-grid result. This can be better illustrated in Figure 8 where the direct-grid scores using numerical JSON encoding are plotted against the corresponding object-based JSON encoding (with no edge information). GPT-3.5 Turbo sees the biggest improvement with a doubling of performance and all other LLMs see significant gains too.

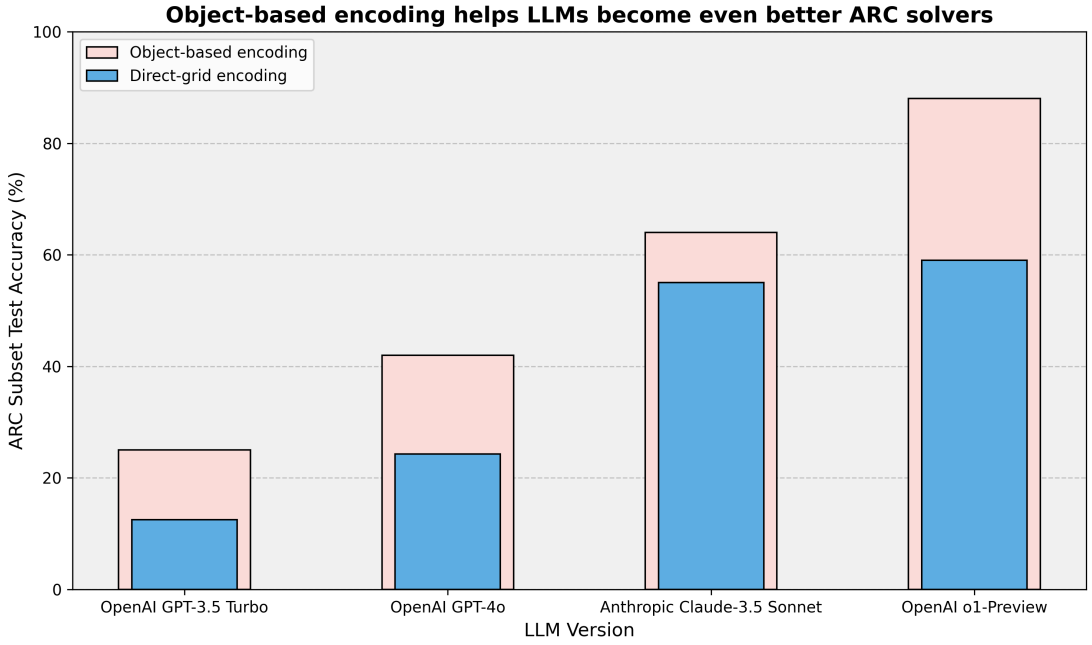


Figure 8: Comparison of LLM performance at solving ARC tasks when using a direct-grid approach and when an object-based abstraction has been applied. The boost in the performance of the LLMs is clear.

What comes as a surprise from these results however is that the addition of edge information (i.e. information about the relative position of objects in the grid) not only does not improve the performance of the LLMs, but it actually makes it worse. Only Anthropic's Claude 3.5 Sonnet sees a 3% improvement in its score when edge information is added to the JSON encoding of the tasks, all other models see a deterioration which in some cases is significant. What this can mean is that the information already contained in the object-based encodings is sufficient for the LLM to have a clear idea about the relationships of objects on the grid (it already has the coordinates of all coloured pixels

on the grid). Adding explicit information about which object shares coordinates with other objects in the grid does not help the LLM's performance but seems to hinder it.

5 Discussion

5.1 Meeting our Objectives

In this study we explored questions around intelligence and reasoning in artificial systems. Specifically we presented ARC, a set of psychometric tests suitable for testing artificial systems that are designed around the principles of core knowledge (Spelke & Kinzler 2007) and which currently seem to generate a lot of research interest. We set out to evaluate how LLMs perform against a subset of ARC tasks. We were able to obtain evidence that their performance, even though not remarkable especially by human standards, has been steadily getting better with every new generation of model.

We performed several tests and collected evidence that shows how the encoding of ARC tasks can affect the performance of LLMs in solving those tasks. We established that a more structured format of the input such as the one offered by a JSON encoding, seems to improve the LLM performance compared to plain textual representations. We believe that the structured format of JSON benefits the parsing and processing of the information by the model.

We looked at the more recent multimodal LLMs such as OpenAI's GPT-4o which can combine multiple types of input such as text, image or audio. We have tested them against ARC tasks as well. We concluded that even though their skills at extracting information from visual inputs is impressive, their overall performance at solving ARC tasks is still much worse than that when relying on text only in agreement to what is found by other researchers.

We applied the Chain of Thought technique which has been reported to significantly improve the ability of LLMs to perform complex reasoning (Wei et al. 2022). However we were unable to replicate this behaviour. In our tests we have found that adding CoT to our prompts led to a deterioration of the LLM performance. This warrants further investigation. On the other hand, our CoT tests have helped us to uncover that there is a significant number of cases where the LLM identifies the transformation and is able to describe it in words, but fails to translate its logic into the correct output grid.

This mismatch between reasoning ability and output generation is also worthy of further investigation.

Finally, we deployed an external tool, ARGA, which enabled us to abstract the ARC tasks by translating grids into graphs. We were then able to encode this information and test the LLMs with this object-based model of representation. We found that combining such a tool with LLMs makes a huge difference in solving ARC tasks. The combination of an object-based abstraction tool with a state of the art LLM like OpenAI's o1-preview has produced a remarkable 88.0% accuracy in our ARC subset, which is almost twice the score achieved in a similar study (Xu et al. 2024).

However, putting things into perspective, we have found that overall LLMs (without external tools) perform poorly at ARC tasks. OpenAI's o1-preview in our tests manages to solve 31.5 out of a set of 50 of what are some of the simplest tests in the ARC training dataset. We were unable to run our tests on larger portions of the dataset due to the costs involved. However, the official ARC prize leaderboard reports that o1-preview scores 21% on the public evaluation dataset (https://arcprize.org/leaderboard), which is probably not surprising. The best score overall in ARC at the time of this writing is 54.5% by a team called *MindsAI*. The deadline for submissions for ARC-prize 2024 is November 10, so it will be really interesting to see what the high score will be at that time.

Regardless, it looks like the ARC benchmark has stood the test of time (see for example Figure 9) and quite possibly represents one of the most reliable ways for measuring reasoning and abstraction abilities in artificial systems at our disposal currently. Other benchmarks which mainly rely on LLMs scoring high on knowledge-based academic tests get outdated quickly and have recently come under fire as it seems that they are liable to variations in performance when small perturbations in the way the questions are presented are introduced (Srivastava et al. 2024).

Finally, we need to also consider the question on whether the improvement that we have established in the performance of LLMs at ARC tasks comes strictly from an actual evolution in their reasoning abilities or whether ARC tests and solutions are slowly finding their way into the set of training data that are used to develop the most recent versions

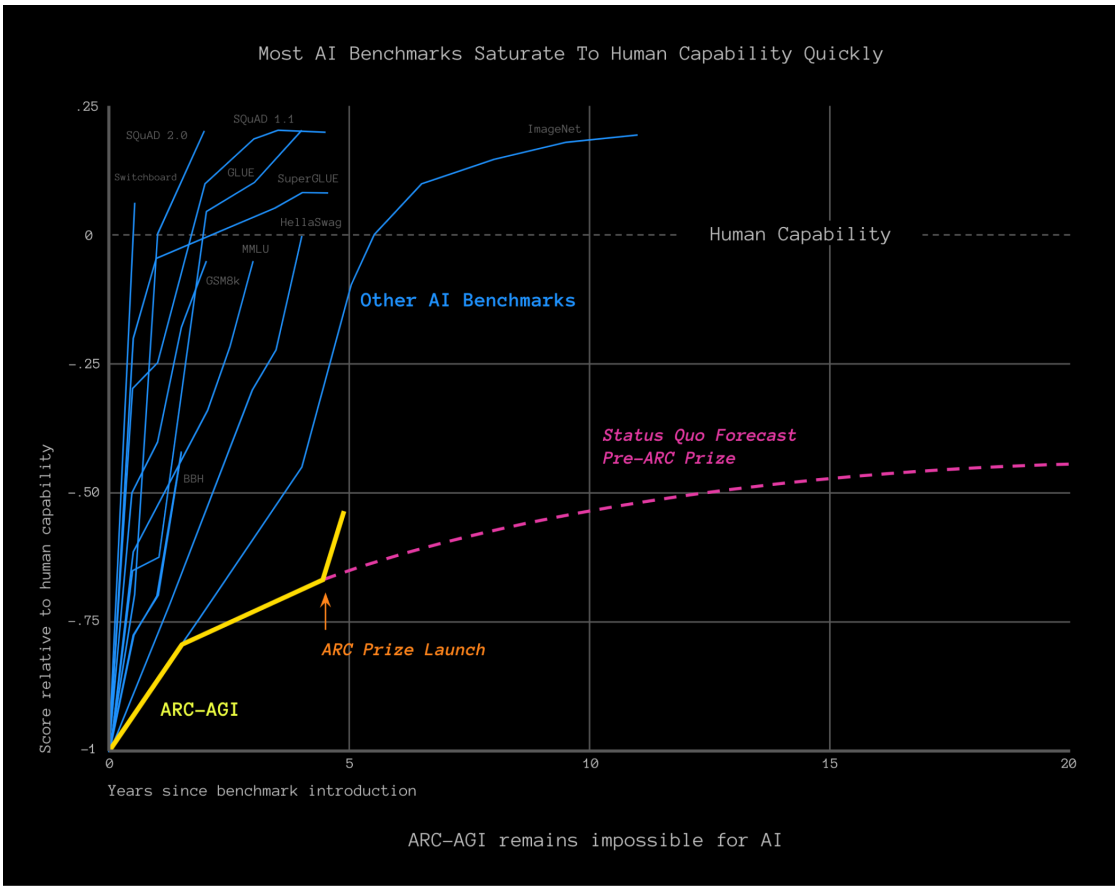


Figure 9: Plot showing AI performance relative to human capability for a number of popular benchmarks as a function of years since the benchmark's introduction. ARC seems to be much more resilient as a benchmark, indicating that a true breakthrough may be needed in order to crack it. It is also remarkable to note how the introduction of ARC-Prize 2024 in June of 2024 has generated such interest that resulted in a dramatic acceleration of progress indicated by the kink in the graph. Plot reproduced from https://arcprize.org (23/10/24).

of LLMs. If the latter is the case then what we observe is only the ability of LLMs to retrieve information that already exists in their vast library of knowledge. The only way to answer this question is of course to test LLMs using ARC tasks that are not in the public domain. The only piece of evidence that we managed to gather in this respect is by looking at the ARC-prize leaderboard (https://arcprize.org/leaderboard), where we see that apparently LLMs have also been tested in what is called a "semi-private eval" set where despite the fact that they perform worse than the public evaluation set, there is still a discernible improvement of newer models over older ones. So in our estimation the assessment that LLM performance is overall improving with time seems valid.

5.2 Comparisons with previous research

Our analysis follows in the footsteps of other research, more specifically with the analysis described in (Xu et al. 2024) and (Mitchell et al. 2023). Specifically, the paper by Xu et al. (2024) has been one of the great inspirations for embarking in this project in the first place and we therefore decided to stay close to its specifications so that we can maintain a solid frame of reference for our own results. In particular we have used in our study the same subset of 50 simpler ARC tasks that they also used so that our results can be directly comparable. However, we have built our code independently and designed our prompting strategy and processing of responses differently to what they have done. We have also run our experiments on a range of different models, including models that were not previously available, thus extending the scope of the study.

In our direct-grid encoding experiments we have developed both a numerical JSON and a string JSON encoding that was not used in the Xu et al. (2024) paper. We have found this encoding to be superior in performance to all other methods of encoding producing better results for all LLMs tested. We believe that this proves that a structured input like that offered by JSON encodings creates a better awareness of the task setup for the LLM compared to a more plain textual representation.

We have not managed to validate the findings of Xu et al. (2024) when it comes to Chain of Thought experiments. Their results demonstrate a (slight) improvement when adding CoT to the prompt (not across all models it has to be said), whereas we have not

noticed an identifiable effect. In fact in our experiments we have observed a decline in performance when adding CoT functionality to the prompts.

We have deployed the object-based encoding in a similar fashion to what they did since both our analyses leverage the same ARGA tool to achieve this. However, our analysis has extended their results by employing a wider variety of LLMs from more providers. We were thus able to record a remarkable improvement by combining ARGA with a state of the art LLM like the o1-preview. In Xu et al. (2024) the best score registered was 23 tasks out of 50 whereas we have achieved a best score of 44 out of 50, an almost doubling of the best result.

In this respect, this report, even though not ground-breaking, we believe makes some substantial contributions to already established research in current publications. With some more time and resources at our disposal we believe that we could expand our research to more novel and interesting directions.

6 Evaluation, Reflections & Conclusions

Undertaking this study has been an enriching and educational experience. Generative AI is a fascinating technology with many promising applications. However the hype that its success has brought about is contributing to a lot of misunderstandings about its nature and our expectations of the technology. Establishing some ground truths about the abilities of generative models like LLMs is paramount especially when a lot of debate is taking place around artificial general intelligence (AGI). This study has attempted an introductory enquiry into this hotly contested space by exploring a benchmark that is well regarded in the scientific community.

We conclude this thesis with a set of mixed feelings regarding the main question we set out to investigate, which was, can LLMs reason? On one hand, the intrinsic design of LLMs as programs that predict the next token by leveraging enormous amounts of data to capture statistical relationships between words, phrases, and concepts should not in principle be conducive to a reasoning ability. At the end of the day LLMs spit out words with no inherent understanding of what these words really mean, they just know the probabilistic relationships between these words. On the other hand we find that their performance in solving tasks that are based purely on reasoning and abstraction abilities, such as the ARC tests, is getting better with each new generation of model. The use of novel techniques like reinforcement learning and chain of though in more recent models seems to further boost this performance to surprising levels. This makes it hard to shrug off these achievements as plain pattern-matching.

What cannot be doubted however is that even if LLMs are relatively poor at reasoning on their own, the use of LLMs in conjunction with other tools can create a powerful combination. This can be seen for example in models like o1-preview which certainly goes beyond the use of just machine learning learning techniques to achieve its impressive performance, but it can also be seen in the top scoring ARC solvers in the ARC competition, which leverage the power of several techniques, including LLMs, to reach their result. Perhaps this is exactly what Françpois Chollet was anticipating to happen by setting forth his competition. And even though the inner workings of models like o1-preview will remain undisclosed for now, we will soon find out how the top teams at ARC-Prize

have built their solutions, since these need to be open sourced at the end of the competition.

In terms of our particular study, certain limiting factors have dictated how we perform the analysis. For example, we could have expanded the breadth of our LLM testing by including even more tests from the large ARC dataset. This would have given us more accurate performance metrics. However prompting LLMs comes at a cost. The total cost of this study in API usage has come to $\$300$, all of it self-funded. The vast majority of the cost has stemmed from the use of the state of the art OpenAI o1-preview model. Each run of the 50 ARC tasks on the o1-preview model cost around $\$20$ which is very expensive. However science requires sacrifices and we were determined to test how the hype around o1-preview as a truly reasoning model stands against reality.

By limiting ourselves to just 50 ARC tasks we could only do a qualitative rather than a quantitative study. Nevertheless, we were still able to establish some useful results. The testing framework that we have developed can be used against more LLMs that are published in the future. One interesting path to explore would be to run our tests with open source models such as Llama 3.2 and see how open-source compares to proprietary models.

It was very curious to see that Chain of Thought techniques did not generate any improvement in our results despite several testaments in the literature that they should do so. In our tests, adding more information to the prompts just seemed to be worse for the LLMs. We are wondering if there are different ways of implementing CoT that we failed to explore. This would definitely be an area of focus in any subsequent investigation.

Finally, It was really interesting to see how a tool that enhances the abstraction capability of the LLM can have such an impact on performance. One question that lingers from this study is why the edge information between objects does not improve test accuracy. One path that we could still explore if we had more time would be to examine different ways of referencing edge information between objects for example by describing *the ways in which* Object A is related to Object B rather than just the fact that Object A is related to Object B. But this is a story for another day..

References

Acquaviva, S., Pu, Y., Kryven, M., Sechopoulos, T., Wong, C., Ecanow, G. E., Nye, M., Tessler, M. H. & Tenenbaum, J. B. (2023), 'Communicating Natural Programs to Humans and Machines'.
URL: https://arxiv.org/abs/2106.07824

Bonazzi, M. (2023), Protagoras, *in* E. N. Zalta & U. Nodelman, eds, 'The Stanford Encyclopedia of Philosophy', Fall 2023 edn, Metaphysics Research Lab, Stanford University.
URL: https://plato.stanford.edu/archives/fall2023/entries/protagoras/

Bringsjord, S. & Schimanski, B. (2003), 'What is Artificial Intelligence? Psychometric AI as an Answer', *Proc. of the 18th International Joint Conference on Artificial Intelligence (IJCAI 2003)* .

Bubeck, S., Chandrasekaran, V., Eldan, R., Gehrke, J., Horvitz, E., Kamar, E., Lee, P., Lee, Y. T., Li, Y., Lundberg, S., Nori, H., Palangi, H., Ribeiro, M. T. & Zhang, Y. (2023), 'Sparks of Artificial General Intelligence: Early experiments with GPT-4'.

Campbell, M., Hoane, A. & hsiung Hsu, F. (2002), 'Deep blue', *Artificial Intelligence* **134**(1), 57–83.
URL: https://www.sciencedirect.com/science/article/pii/S0004370201001291

Chollet, F. (2019), 'On the measure of intelligence', *CoRR* **abs/1911.01547**.
URL: http://arxiv.org/abs/1911.01547

Chollet, F., Knoop, M., Landers, B., Kamradt, G., Jud, H., Reade, W. & Howard, A. (2024), 'ARC Prize 2024'.
URL: https://kaggle.com/competitions/arc-prize-2024

Chollet, F., Tong, K., Reade, W. & Elliott, J. (2020), 'Abstraction and Reasoning Challenge'.
URL: https://kaggle.com/competitions/abstraction-and-reasoning-challenge

Cole, J. (2024), 'Test-Time Augmentation to solve ARC'. Accessed: August 2024.
URL: https://lab42.global/community-interview-jack-cole/

Fridman, L. (2024), 'Can LLMs reason? Yann LeCun and Lex Fridman'. Accessed: 23-10-2024.
URL: https://www.youtube.com/watch?v=N09C6oUQX5M

Graepel, T., Goutrié, M., Krüger, M. & Herbrich, R. (2001), Learning on graphs in the game of go, *in* G. Dorffner, H. Bischof & K. Hornik, eds, 'Artificial Neural Networks — ICANN 2001', Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 347–352.

Greenblatt, R. (2024), 'Getting 50% (SoTA) on ARC-AGI with GPT-4o'. Accessed: August 2024.
URL: https://redwoodresearch.substack.com/p/getting-50-sota-on-arc-agi-with-gpt

Gregory, R. L. (1998), *The Oxford Companion to the Mind*, Oxford University Press, Oxford, UK.

Handbook of Intelligence (2000), Cambridge University Press.

Heaven, W. D. (2023), 'Geoffrey Hinton tells us why he's now scared of the tech he helped build'. Accessed: August 2024.
URL: https://www.technologyreview.com/2023/05/02/1072528/geoffrey-hinton-google-why-scared-ai/

Hernández-Orallo, J. (2017*a*), 'Evaluation in artificial intelligence: from task-oriented to ability-oriented measurement', *Artificial Intelligence Review* **48**, 397–447.
URL: https://api.semanticscholar.org/CorpusID:207079473

Hernández-Orallo, J. (2017*b*), *The Measure of All Minds: Evaluating Natural and Artificial Intelligence*, Cambridge University Press.

Hernández-Orallo, J., Dowe, D. & Hernández-Lloreda, V. (2014), 'Universal psychometrics: Measuring cognitive abilities in the machine kingdom', *Cognitive Systems Research* **27**, 50–74.

John & Raven, J. (2003), *Raven Progressive Matrices*, Springer US, Boston, MA, pp. 223–237.

Johnson, A., Vong, W. K., Lake, B. M. & Gureckis, T. M. (2021), 'Fast and flexible: Human program induction in abstract reasoning tasks'.
URL: https://arxiv.org/abs/2103.05823

Kim, S., Phunyaphibarn, P., Ahn, D. & Kim, S. (2022), Playgrounds for abstraction and reasoning, *in* 'NeurIPS 2022 Workshop on Neuro Causal and Symbolic AI (nCSI)'.
URL: https://openreview.net/forum?id=F4RNpByoqP

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y. & Iwasawa, Y. (2023), 'Large Language Models are Zero-Shot Reasoners'.
URL: https://arxiv.org/abs/2205.11916

Legg, S. & Hutter, M. (2007), 'A collection of definitions of intelligence'.
URL: https://arxiv.org/abs/0706.3639

Lei, C., Lipovetzky, N. & Ehinger, K. A. (2024), 'Generalized Planning for the Abstraction and Reasoning Corpus'.
URL: https://arxiv.org/abs/2401.07426

Minsky, M. L., ed. (1968), *Semantic Information Processing*, MIT Press.

Mitchell, M., Palmarini, A. B. & Moskvichev, A. (2023), 'Comparing Humans, GPT-4, and GPT-4V On Abstraction and Reasoning Tasks'.
URL: https://arxiv.org/abs/2311.09247

Mollick, E. (December 2022), 'ChatGPT Is a Tipping Point for AI'. Accessed: 24-10-2024.
URL: https://hbr.org/2022/12/chatgpt-is-a-tipping-point-for-ai

Moskvichev, A., Odouard, V. V. & Mitchell, M. (2023), 'The ConceptARC Benchmark: Evaluating Understanding and Generalization in the ARC Domain'.
URL: https://arxiv.org/abs/2305.07141

Murphy, H. & Criddle, C. (May 2024), 'Meta AI chief says large language models will not reach human intelligence'. Accessed: September 2024.
URL: https://www.ft.com/content/23fab126-f1d3-4add-a457-207a25730ad9

OpenAI (2024), 'Learning to Reason with LLMs'. Accessed: October 2024.
URL: https://openai.com/index/learning-to-reason-with-llms/

OpenAI, Achiam, J. et al. (2024), 'GPT-4 Technical Report'.
URL: https://arxiv.org/abs/2303.08774

Singh, M., Cambronero, J., Gulwani, S., Le, V. & Verbruggen, G. (2023), 'Assessing GPT4-V on Structured Reasoning Tasks'.
URL: https://arxiv.org/abs/2312.11524

Spelke, E. S. & Kinzler, K. D. (2007), 'Core knowledge', *Developmental Science* **10**(1), 89–96.
URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-7687.2007.00569.x

Srivastava, S., B, A. M., au2, A. P. V., Menon, S., Sukumar, A., T, A. S., Philipose, A., Prince, S. & Thomas, S. (2024), 'Functional benchmarks for robust evaluation of reasoning performance, and the reasoning gap'.
URL: https://arxiv.org/abs/2402.19450

Sternberg, R. & Detterman, D. (1986), *What is Intelligence?: Contemporary Viewpoints on Its Nature and Definition*, Bloomsbury Academic.
URL: https://books.google.co.uk/books?id=04x-AAAAMAAJ

Suleyman, M. & Bhaskar, M. (2023), *The Coming Wave: AI, Power and the Twenty-first Century's Greatest Dilemma*, Bodley Head.
URL: https://books.google.co.uk/books?id=EKyvzwEACAAJ

Talagala, N. (November 2023), 'The OpenAI Drama: What Is AGI And Why Should You Care?'. Accessed: 24-10-2024.
URL: https://www.forbes.com/sites/nishatalagala/2023/11/21/the-open-ai-drama-what-is-agi-and-why-should-you-care

Turing, A. M. (1950), 'Computing machinery and intelligence', *Mind* **LIX**(236), 433–460.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. & Polosukhin, I. (2023), 'Attention is all you need'.
URL: https://arxiv.org/abs/1706.03762

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E. H., Le, Q. & Zhou, D. (2022), 'Chain of Thought Prompting Elicits Reasoning in Large Language Models', *CoRR* **abs/2201.11903**.
URL: https://arxiv.org/abs/2201.11903

Wolpert, D. H. (2013), 'Ubiquity symposium: Evolutionary computation and the processes of life: what the no free lunch theorems really mean: how to improve search algorithms', *Ubiquity* **2013**(December).
URL: https://doi.org/10.1145/2555235.2555237

Wolpert, D. & Macready, W. (1997), 'No free lunch theorems for optimization', *IEEE Transactions on Evolutionary Computation* **1**(1), 67–82.

Xu, Y., Khalil, E. B. & Sanner, S. (2022), 'Graphs, Constraints, and Search for the Abstraction and Reasoning Corpus'.

Xu, Y., Li, W., Vaezipoor, P., Sanner, S. & Khalil, E. B. (2024), 'LLMs and the Abstraction and Reasoning Corpus: Successes, Failures, and the Importance of Object-based Representations'.

Appendix A - JSON format of ARC tasks

{"007bbfb7": {"test": [{"input": [[7, 0, 7], [7, 0, 7], [7, 7, 0]]}], "train": [{"input": [[0, 7, 7], [7, 7, 7], [0, 7, 7]], "output": [[0, 0, 0, 0, 7, 7, 0, 7, 7], [0, 0, 0, 7, 7, 7, 7, 7, 7], [0, 0, 0, 0, 7, 7, 0, 7, 7], [0, 7, 7, 0, 7, 7, 0, 7, 7], [7, 7, 7, 7, 7, 7, 7, 7, 7], [0, 7, 7, 0, 7, 7, 0, 7, 7], [0, 0, 0, 0, 7, 7, 0, 7, 7], [0, 0, 0, 7, 7, 7, 7, 7, 7], [0, 0, 0, 0, 7, 7, 0, 7, 7]]}, {"input": [[4, 0, 4], [0, 0, 0], [0, 4, 0]], "output": [[4, 0, 4, 0, 0, 0, 4, 0, 4], [0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 4, 0, 0, 0, 0, 0, 4, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 4, 0, 4, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 4, 0, 0, 0, 0]]}, {"input": [[0, 0, 0], [0, 0, 2], [2, 0, 2]], "output": [[0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 2], [0, 0, 0, 0, 0, 0, 2, 0, 2], [0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 2, 0, 0, 0, 0, 0, 2], [2, 0, 2, 0, 0, 0, 2, 0, 2]]}, {"input": [[6, 6, 0], [6, 0, 0], [0, 6, 6]], "output": [[6, 6, 0, 6, 6, 0, 0, 0, 0], [6, 0, 0, 6, 0, 0, 0, 0, 0], [0, 6, 6, 0, 6, 6, 0, 0, 0], [6, 6, 0, 0, 0, 0, 0, 0, 0], [6, 0, 0, 0, 0, 0, 0, 0, 0], [0, 6, 6, 0, 0, 0, 0, 0, 0], [0, 0, 0, 6, 6, 0, 6, 6, 0], [0, 0, 0, 6, 0, 0, 6, 0, 0], [0, 0, 0, 0, 6, 6, 0, 6, 6]]}, {"input": [[2, 2, 2], [0, 0, 0], [0, 2, 2]], "output": [[2, 2, 2, 2, 2, 2, 2, 2, 2], [0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 2, 2, 0, 2, 2, 0, 2, 2], [0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 2, 2, 2, 2, 2, 2], [0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 2, 2, 0, 2, 2]]}]}, "00d62c1b":.....}

Appendix B - Direct-grid Prompt Template

Example of the prompt used for the few-shot direct-grid experiments.

```
You are a chatbot with human-like reasoning and abstraction
capabilities. We will engage in tasks that require reasoning and
logic. You will be presented with grids made up of numbers. Number
0 represents empty cells and the other numbers represent objects
or patterns on the grid. For each task, you will receive a few
examples that demonstrate the transformation from an input to an
output grid. After the examples you'll receive a new input grid
called Test. Your task is to determine the corresponding output
grid from the transformation you can infer from the examples. Use
the same format as the one provided in the examples for your
answer. Do not give any justification for your answer, just
provide the output grid.

Training Examples
Example 1: Input
111
000
000

Example 1: Output
000
111
000

...

Test Input
000
010
000
```

Appendix C - Chain of Thought Prompt Templates

1) Basic CoT (step-by-step thinking)

```
You are a chatbot with human-like reasoning and abstraction
capabilities.
We will engage in tasks that require reasoning and logic.
You will be presented with grids made up of numbers.
Number 0 represents empty cells and the other numbers represent
objects or patterns on the grid.
Follow these steps:
1. Carefully analyze each input-output example in the task and
identify the transformation.
2. Describe the transformation step by step.
3. Apply the identified transformation to the Test input grid to
generate the output grid.
4. Use the marker '---Analysis Start---' before providing your
analysis of the transformation.
5. Use the marker '---Output Grid Start---' before providing the
final output grid.
6. Use the same format as the one provided in the examples for
your output grid.
7. Use the marker '---Output Grid End---' at the end of the final
output grid.
```

2) CoT with example transformation

```
You are a chatbot with human-like reasoning and abstraction
capabilities.
We will engage in tasks that require reasoning and logic.
You will be presented with grids made up of numbers.
Number 0 represents empty cells and the other numbers represent
objects or patterns on the grid.
First you will be shown a sample tranformation together with the
reasoning behind the transformation.
```

Then you will be presented with a novel task. Follow these steps:
1. Carefully analyze each input-output example in the task and identify the transformation.
2. Describe the transformation step by step.
3. Apply the identified transformation to the Test input grid to generate the output grid.
4. Use the marker '---Analysis Start---' before providing your analysis of the transformation.
5. Use the marker '---Output Grid Start---' before providing the final output grid.
6. Use the same format as the one provided in the examples for your output grid.
7. Use the marker '---Output Grid End---' at the end of the final output grid.

Sample grid tranformation:

Input grid

```
[
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 4, 0, 0, 0, 0],
[0, 0, 4, 4, 0, 0, 0, 0, 0],
[0, 0, 4, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 4, 0, 0, 0],
[0, 0, 0, 0, 0, 4, 4, 4, 0],
[0, 0, 0, 0, 0, 0, 4, 0, 0]
]
```

Output grid

```
[
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 7, 7, 4, 0, 0, 0, 0],
[0, 0, 4, 4, 7, 0, 0, 0, 0],
[0, 0, 4, 7, 7, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 4, 7, 7, 0],
[0, 0, 0, 0, 0, 4, 4, 4, 0],
[0, 0, 0, 0, 0, 7, 4, 7, 0]
]

Transformation applied: Add number 7 cells in locations adjacent
to number 4 cells so that together they form three by three
objects.
End of sample.

Beginning of your task:
```

3) CoT with extended example

```
You are a chatbot with human-like reasoning and abstraction
capabilities.
We will engage in tasks that require reasoning and logic.
You will be presented with grids made up of numbers.
Number 0 represents empty cells and the other numbers represent
objects or patterns on the grid.
First you will be shown an example task together with the
identified transformation.
Then you will be presented with a novel task. Follow these steps:
1. Carefully analyze each input-output example in the task and
identify the transformation.
2. Describe the transformation step by step.
```

3. Apply the identified transformation to the Test input grid to generate the output grid.
4. Use the marker '---Analysis Start---' before providing your analysis of the transformation.
5. Use the marker '---Output Grid Start---' before providing the final output grid.
6. Use the same format as the one provided in the examples for your output grid.
7. Use the marker '---Output Grid End---' at the end of the final output grid.

Sample grid tranformation:

Training Examples
Example 1: Input
[
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[4, 4, 4, 0, 0, 0, 0, 0, 0],
[4, 0, 4, 0, 0, 0, 0, 0, 0],
[0, 0, 4, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 4, 4, 0, 0],
[0, 0, 0, 0, 0, 0, 4, 4, 0],
[0, 0, 0, 0, 0, 4, 0, 4, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0]
]

Example 1: Output
[
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[4, 4, 4, 0, 0, 0, 0, 0, 0],
[4, 7, 4, 0, 0, 0, 0, 0, 0],
[7, 7, 4, 0, 0, 0, 0, 0, 0],

```
[0, 0, 0, 0, 0, 4, 4, 7, 0],
[0, 0, 0, 0, 0, 7, 4, 4, 0],
[0, 0, 0, 0, 0, 4, 7, 4, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0]
]

Example 2: Input
[
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[4, 4, 4, 0, 0, 0, 0, 0, 0],
[0, 4, 4, 0, 0, 0, 0, 0, 0],
[4, 4, 4, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 4, 4, 4, 0],
[0, 0, 0, 0, 0, 0, 4, 0, 0],
[0, 0, 0, 0, 0, 0, 4, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0]
]

Example 2: Output
[
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[4, 4, 4, 0, 0, 0, 0, 0, 0],
[7, 4, 4, 0, 0, 0, 0, 0, 0],
[4, 4, 4, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 4, 4, 4, 0],
[0, 0, 0, 0, 0, 7, 4, 7, 0],
[0, 0, 0, 0, 0, 7, 4, 7, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0]
]
```

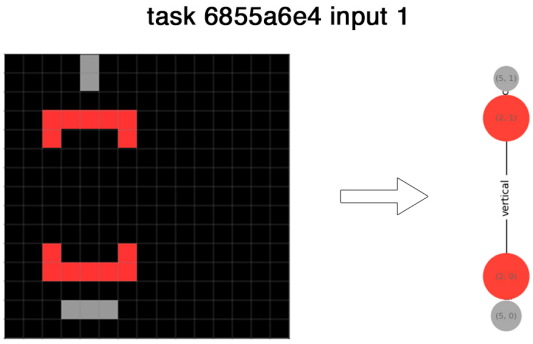
```
Test grid
[
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 4, 0, 0, 0, 0],
[0, 0, 4, 4, 0, 0, 0, 0, 0],
[0, 0, 4, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 4, 0, 0, 0],
[0, 0, 0, 0, 0, 4, 4, 4, 0],
[0, 0, 0, 0, 0, 0, 4, 0, 0]
]

Output grid
[
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 7, 7, 4, 0, 0, 0, 0],
[0, 0, 4, 4, 7, 0, 0, 0, 0],
[0, 0, 4, 7, 7, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 4, 7, 7, 0],
[0, 0, 0, 0, 0, 4, 4, 4, 0],
[0, 0, 0, 0, 0, 7, 4, 7, 0]
]

Transformation applied: Add number 7 cells in locations adjacent
to number 4 cells so that together they form three by three
objects.
End of sample.

Beginning of your task:
```

Appendix D - Object-based textual encodings



Plain encoding

```
Image size: (15, 15)
Objects:
Object 1: coordinates=[(10, 2), (10, 6), (11, 2), (11, 3), (11,
4), (11, 5), (11, 6)], color=2, size=7
Object 2: coordinates=[(3, 2), (3, 3), (3, 4), (3, 5), (3, 6), (4,
2), (4, 6)], color=2, size=7
Object 3: coordinates=[(13, 3), (13, 4), (13, 5)], color=5, size=3
Object 4: coordinates=[(0, 4), (1, 4)], color=5, size=2
```

Plain encoding with edge information

```
Image size: (15, 15)
Objects:
Object 1: coordinates=[(10, 2), (10, 6), (11, 2), (11, 3), (11,
4), (11, 5), (11, 6)], color=2, size=7, neighbors=[Object 2,
Object 3]
Object 2: coordinates=[(3, 2), (3, 3), (3, 4), (3, 5), (3, 6), (4,
2), (4, 6)], color=2, size=7, neighbors=[Object 1, Object 4]
Object 3: coordinates=[(13, 3), (13, 4), (13, 5)], color=5,
size=3, neighbors=[Object 1]
Object 4: coordinates=[(0, 4), (1, 4)], color=5, size=2,
neighbors=[Object 2]
```

JSON encoding

```
Image size: (15, 15)
Objects:
[{"coordinates": [[10, 2], [10, 6], [11, 2], [11, 3], [11, 4],
[11, 5], [11, 6]], "color": 2, "size": 7}, {"coordinates": [[3,
2], [3, 3], [3, 4], [3, 5], [3, 6], [4, 2], [4, 6]], "color": 2,
"size": 7}, {"coordinates": [[13, 3], [13, 4], [13, 5]], "color":
5, "size": 3}, {"coordinates": [[0, 4], [1, 4]], "color": 5,
"size": 2}]
```

JSON encoding with edge information

```
Image size: (15, 15)
Objects:
[{"coordinates": [[10, 2], [10, 6], [11, 2], [11, 3], [11, 4],
[11, 5], [11, 6]], "color": 2, "size": 7, "id": 1, "neighbors":
[2, 3]}, {"coordinates": [[3, 2], [3, 3], [3, 4], [3, 5], [3, 6],
[4, 2], [4, 6]], "color": 2, "size": 7, "id": 2, "neighbors": [1,
4]}, {"coordinates": [[13, 3], [13, 4], [13, 5]], "color": 5,
"size": 3, "id": 3, "neighbors": [1]}, {"coordinates": [[0, 4],
[1, 4]], "color": 5, "size": 2, "id": 4, "neighbors": [2]}]
```

JSON encoding with string colours

```
Image size: (15, 15)
Objects:
[{"coordinates": [[10, 2], [10, 6], [11, 2], [11, 3], [11, 4],
[11, 5], [11, 6]], "color": "red", "size": 7}, {"coordinates":
[[3, 2], [3, 3], [3, 4], [3, 5], [3, 6], [4, 2], [4, 6]], "color":
"red", "size": 7}, {"coordinates": [[13, 3], [13, 4], [13, 5]],
"color": "gray", "size": 3}, {"coordinates": [[0, 4], [1, 4]],
"color": "gray", "size": 2}]
```