Large Language Models Are Not Strong Abstract Reasoners

Gaël Gendron , Qiming Bao , Michael Witbrock , Gillian Dobbie
University of Auckland
gael.gendron@auckland.ac.nz

Abstract

Large Language Models have shown tremendous performance on a large variety of natural language processing tasks, ranging from text comprehension to common sense reasoning. However, the mechanisms responsible for this success remain opaque, and it is unclear whether LLMs can achieve human-like cognitive capabilities or whether these models are still fundamentally circumscribed. Abstract reasoning is a fundamental task for cognition, consisting of finding and applying a general pattern from few data. Evaluating deep neural architectures on this task could give insight into their potential limitations regarding reasoning and their broad generalisation abilities, yet this is currently an under-explored area. In this paper, we introduce a new benchmark for evaluating language models beyond memorisation on abstract reasoning tasks. We perform extensive evaluations of state-of-the-art LLMs, showing that they currently achieve very limited performance in contrast with other natural language tasks, even when applying techniques that have been shown to improve performance on other NLP tasks. We argue that guiding LLM generation to follow causal paths could help improve the generalisation and reasoning abilities of LLMs.

1 Introduction

Large Language Models (LLMs) have achieved impressive performance on a large variety of Natural Language Processing (NLP) tasks, including text comprehension [Devlin *et al.*, 2019; Radford *et al.*, 2019], commonsense reasoning [Talmor *et al.*, 2020], and code generation [Bubeck *et al.*, 2023], and have shown promising results for out-of-distribution generalisation [Brown *et al.*, 2020; Bubeck *et al.*, 2023]. The most recent and larger language models also perform well on mathematical problems, which had been out of reach for transformers for a long time [Chen *et al.*, 2022; Stolfo *et al.*, 2022]. While empirical testing of LLMs trained on large corpora of data yields signs of high comprehension of presented problems, there is little theoretical evidence regarding how this performance has been achieved and whether these models are simply memorising the training data, extrapolating it, or some combination [Tirumala *et al.*, 2022; Goyal and Bengio, 2020]. A notable limitation of these models is a lack of control mechanisms, or possible misalignment [Ouyang *et al.*, 2022], for which the absence of a world model or causal representation have been advanced as explanations [Bender *et al.*, 2021; Zecevic *et al.*, 2023]. Experiments on GPT-4 showed signs of limitations on reasoning tasks requiring planning and backtracking or with an uncommon distribution [Bubeck *et al.*, 2023; Wu *et al.*, 2023]. Despite these limitations, the question of whether or not LLMs can perform human-like reasoning remains open, as measuring the intelligence, or more broadly, the competence, of a system is a challenging task [Chollet, 2019].

Abstract reasoning is a potential task for effective measurement of the cognitive abilities of neural models [Santoro *et al.*, 2018; Chollet, 2019]. Abstract reasoning problems consist of identifying generic structures over a small set of examples and applying them to unseen cases. They aim to evaluate the ability of a system to integrate a new skill or process from limited data. The abstract nature of these problems helps avoid spurious correlations that could lie in the data and may create potential bias in the results. In particular, this task is well-suited for evaluating the broad or strong generalisation capacity of a system, i.e. its ability to handle a large category of tasks and environments without human intervention, including situations that may not have been foreseen when the system was created [Chollet, 2019]. This is a well-studied class of task in the field of program induction [Ellis *et al.*, 2020; Lake *et al.*, 2015]. However, the problem of abstract reasoning has long remained outside the scope of evaluation of language models, and there currently exist no extensive evaluations of the performance of LLMs in this domain.

In this paper, we seek to bridge this gap by investigating the abstract reasoning abilities of LLMs and by providing insight into the following question: Do LLMs contain sufficient building blocks for broad generalisation, or do they lack fundamental capabities? We evaluate state-of-the-art LLMs on abstract reasoning tasks, applying recent fine-tuning and prompt design techniques that have been shown to improve performance on other NLP tasks. To this end, we create a benchmark based on existing datasets and novel datasets transposed from vision tasks and adapted to text-based models. We then perform extensive experiments on

this benchmark. We also build and fine-tune LLMs for abstract reasoning and compare their performances with the general models. Our results indicate that Large Language Models do not yet have the ability to perform sound abstract reasoning. All of the tested models exhibit poor performance, and the tuning techniques that improved LLM reasoning abilities do not provide significant help for abstract reasoning. We investigate potential reasons for this setback. We release our code and data at: https://github.com/Strong-AI-Lab/Logical-and-abstract-reasoning. Our contributions can be summarised as follows:

- We perform an extensive evaluation of pre-trained and fine-tuned LLMs on abstract reasoning tasks.
- We show that existing training and tuning techniques do not help increase the performance of LLMs in abstract reasoning, and investigate the reasons and leads for improvement. In particular, we show that LLMs fail to grasp abstract patterns and learn causal mechanisms.
- We create a benchmark for the evaluation of language models for abstract reasoning.

2 Related Work

The abilities of Language Models have been thoroughly studied on a wide range of problems. Their reasoning capacities are the focus of a great deal of recent work. Some of this [Wei *et al.*, 2022; Li *et al.*, 2022; Chen *et al.*, 2022] has explored prompt techniques to improve mathematical reasoning in LLMs; [Stolfo *et al.*, 2022] propose a framework based on causality theory to evaluate language models on this kind of task. Recently, GPT-4 has been shown to perform well on mathematical problems although it still produces calculation mistakes [Bubeck *et al.*, 2023]. In the domain of logical reasoning, evaluations showed that the logical reasoning abilities of LLMs are tied to the semantics of the input and can hallucinate in uncommon situations [Tang *et al.*, 2023; Xu *et al.*, 2023]. Causal structure discovery and causal inference are other domains where LLMs have shown mixed results [Zecevic *et al.*, 2023; Kiciman *et al.*, 2023; Jin *et al.*, 2023]. These tasks are distinct from commonsense causal reasoning, where LLMs perform well [Kiciman *et al.*, 2023] . Experiments with GPT-4 [Bubeck *et al.*, 2023] showed that, despite presenting systematically better performance than its previous versions, it still has some innate limitations. The authors introduce several examples indicating that the autoregressive nature of LLMs may prevent them from planning and backtracking, two abilities necessary for complex reasoning [Bubeck *et al.*, 2023]. GPT-4 also does not always reason in a consistent manner. Although it produces consistent results more often than GPT-3, there are no guarantees that the process leading to the result is always correct. GPT-4's performance also drops on counterfactual tasks, i.e. common problems in unfamiliar settings (e.g. arithmetic in base nine) [Wu *et al.*, 2023]. These experiments highlight a lack of abstraction when solving a task but the reason for these shortcomings remain unknown. The scope of cognitive abilities of the system remain incompletely characterised, especially for precise reasoning [Bubeck *et al.*, 2023].

The evaluations described above do not, of course, provide a measure of the intelligence or global cognitive abilities of those models; measuring the level of intelligence of LLMs and other AI systems is challenging as there is no clear widely accepted definition [Booch *et al.*, 2021; Goyal and Bengio, 2020]. [Chollet, 2019] defines the intelligence of a system as "a measure of its skill-acquisition efficiency over a scope of tasks, with respect to priors, experience, and generalization difficulty". Following this definition, abstract reasoning is a well-suited domain over which to measure aspects of the learning and generalisation abilities of a system. To this end, the Abstract Reasoning Challenge (ARC) has been proposed as a benchmark for artificial systems [Chollet, 2019]. A handful of works have proposed to measure abstract reasoning abilities in neural networks, but they focus on visual tasks [Santoro *et al.*, 2018; Zhang *et al.*, 2019; Zhang *et al.*, 2021a]. To the best of our knowledge, this paper is the first to present an extensive evaluation of abstract reasoning for Large Language Models. Other domains of study focus on problems similar to abstract reasoning. Notably, in program induction, DreamCoder is a system that learns to solve problems described by a small set of input-output pairs by writing programs [Ellis *et al.*, 2020]. Abstract reasoning can also be related to causal representation learning, as finding abstract relations amounts to recovering the causal structure of a task and the Independent Causal Mechanisms (ICMs) linking the variables [Schölkopf *et al.*, 2021; Gendron *et al.*, 2023].

3 Evaluation Method

3.1 Evaluation Data

Large Language Models are challenging to evaluate due to the lack of information on their training set. It is often hard to distinguish memorisation from in-context reasoning. In particular, evaluation datasets can have leaked to the LLM training sets [Li and Flanigan, 2024]. Visual tests are less likely to be part of LLM training sets and their decomposition into text allows multiple groundings. Symbolic tasks have a specific grammar and cannot be solved using information from other domains (e.g. knowledge of syntactic rules). Therefore, we build a new framework that adapts symbolic and vision datasets for abstract reasoning. Each dataset contains a task consisting of recovering an abstract pattern from a small set of context examples. We consider the pattern *abstract* as they depend solely on very few axiomatic priors and not on memorised knowledge (e.g. innate human priors [Chollet, 2019]). We select the tasks based on their capacity to evaluate the ability of a system to find a general abstract rule from limited examples. The visual datasets are converted into text and symbolic versions to be used with language models. After formatting, the datasets can be divided into two categories: Open-Ended Question Answering (Open QA) and Multiple-Choice Question Answering (MCQA). Open QA datasets require the model to generate the correct answer, while MCQA requires it to choose the answer from a set of possible answers. We note that most of the evaluated models are built for general-purpose text generation. Therefore, even when choosing between several options, they must *generate* the

correct choice and may fail to do so (e.g. answering D when only options A, B, or C are available). For comparison, we also evaluate models built for question answering. We give more details in Section 3.2. The datasets obtained are summarised in Table 1.

Example Cases

[9, 4] → [9, 4]
[8, 2, 9, 4, 1, 7] → [9, 4, 8, 2, 1, 7]
[5, 7, 3, 4, 9, 2, 0] → [3, 4, 5, 7, 9, 2, 0]
[1, 5, 6, 4, 0, 3, 7] → [6, 4, 1, 5, 0, 3, 7]

Test Case

[4, 3, 2, 8, 9, 6, 7, 1] → *[2, 8, 4, 3, 9, 6, 7, 1]*

Figure 1: Example task in the BIG-Bench-F dataset. The system must return the input list with the first two elements switched with the following two if they exist. Pre-prompts are omitted from the input. In the test case, the target answer is indicated in *italics*.

Dataset	Type	Versions	
		Text	Symb
ARC^T	Open QA		✓
BIG-Bench-F			✓
Evals-S			✓
PVR			✓
ACRE^T	MCQA	✓	✓
Evals-P			✓
RAVEN^T		✓	✓

Table 1: Datasets considered. When not written, type is similar to the one above. Datasets can exist in text or symbolic versions. Text datasets built from an image dataset are indicated with ^{T}.

Models	
✎ LLaMA-7B	✎ GPT-2
✎ LLaMA2-7B	✎ Text-Davinci-3
✎ Alpaca	✎ GPT-3.5-Turbo
✎ Alpaca-LoRA	✎ GPT-4
✎ LLaMA-7B-AR-LoRA*	✎ Zephyr-7B-β
✎ LLaMA2-7B-AR-LoRA*	
⁞≡ RoBERTa-AR*	
⁞≡ MERIt-AR*	

Table 2: Models considered. When not written, type is similar to the one above. Models with * are introduced in this paper. "-AR" indicates that the model has been fine-tuned for abstract reasoning. The ✎ and ⁞≡ icons indicate text completion and QA engines.

Datasets We build a text-based version of the Abstract Causal Reasoning (ACRE) dataset [Zhang *et al.*, 2021a] that we name ACRE^T. ACRE is a Visual Question-Answering (VQA) dataset. Each sample in the data comprises six context images and four test cases. Each context image comprises a set of objects with various shapes, colours and textures causing the activation of a light. The goal of a system is to determine from the context examples if the light is on, off, or if its state cannot be determined in the test cases. To solve this task, the model has to determine for each sample what objects are causally responsible for the activation of the light. We generate two versions of the dataset: in ACRE^T-Text, each image is replaced by a high-level textual description, and in ACRE^T-Symbolic, each image is replaced with a numerical vector representation.

The second dataset is the Abstract Reasoning Challenge (ARC) dataset [Chollet, 2019]. The dataset is composed of tasks comprising three input and output grids. The goal of the system is to determine the algorithm that converts the input to the output and apply it to a test case. The grids have a variable size comprised between 8×8 and 30×30, and contain visual patterns (e.g. recognisable shapes, symmetries). We provide the raw grid to the model as a two-dimensional array of integers. We name this version ARC^T. The high dimensionality of the input makes it a challenging task for LLMs. The tasks themselves are also challenging as their transcription in natural language is often complex and supposedly impossible for 12% of them [Acquaviva *et al.*, 2021].

We select a subset of the BIG-Bench dataset [Rule, 2020; Srivastava *et al.*, 2022] that we name BIG-Bench-F for *Functions*. The subset comprises tasks corresponding to a function taking a list as input and returning a new transformed list. For each task, several input-output samples are given. In BIG-Bench-F, we give four samples per task by default. The functions include typical list processing like replacing the value of one element, selecting a subset, or counting elements. An example is given in Figure 1. The challenge in this task is to accurately recognise the function from a few samples.

We select a subset of the Evals dataset [OpenAI, 2023] representing logic puzzles. Evals-P is a set of tasks where a tuple containing a character and a list of characters is given as an input, and a single word from the set {"foo", "bar"} is generated from the input according to a logic hidden from the evaluated system. The task consists of finding the logic from a few samples and applying it to a test case. Evals-S is another set of tasks where a list of integers is given as an input, and an output list of words is generated with a hidden logic.

Pointer-Value Retrieval (PVR) tasks [Zhang *et al.*, 2021b] involve selecting one or several values in a list and applying a function on this subset. For each task, the system must recognise the retrieval and application functions and apply them to a test case. Samples are composed of a pointer-values pair and a label. The values are stored in an array, and the pointer is an integer pointing to an index in the array. The pointer indicates the subset of values to consider for the task. We generate a new PVR dataset following this methodology.

RAVEN [Zhang *et al.*, 2019] is a VQA dataset composed of sequences of images to complete. The images contain Raven matrices [Raven, 1938], i.e. geometric shapes (e.g. square, circle, pentagon) assembled together. RAVEN is a dataset similar to Procedurally Generated Matrices (PGM) [Santoro *et al.*, 2018] but also provides a tree structure describing the semantics of each image. We focus on a subset where a single shape appears in the image. The task is, given a sequence of eight images and eight possible choices, to pick the correct image that follows in the sequence. As RAVEN is a visual

dataset like ACRE, we use the given semantic tree structure to generate a text description of each image we will feed to the evaluated models. We create two sets: RAVEN^T-Text contains natural language descriptions, and RAVEN^T-Symbolic contains symbolic descriptions. We also build another version of the dataset where choices are hidden. We name the former RAVEN^T-mcqa and the latter RAVEN^T-opqa.

3.2 Models Evaluated

We perform evaluations on the most recent and popular architectures for NLP tasks. Table 2 provides the list of models used in the experiments. More details are provided in the appendix. We restrict our experiments to Large Language models . We conduct experiments on the popular family of GPT architectures. We include three generations of GPT models: GPT-2 [Radford *et al.*, 2019], a 1.5B parameter model; aligned GPT-3 models with Text-Davinci-3, optimised for text completion, and GPT-3.5-Turbo, optimised for chat, two 175B models [Brown *et al.*, 2020; Ouyang *et al.*, 2022]; and GPT-4, with unknown training and architectural details [OpenAI, 2023]. We also perform experiments on the popular open models LLaMA [Touvron *et al.*, 2023a] and LLaMA2 [Touvron *et al.*, 2023b]. Alpaca is a fine-tuned version of LLaMA to respond to instructions [Wang *et al.*, 2022; Taori *et al.*, 2023], and Alpaca-LoRA is a LLaMA model instruction-tuned using Low-Rank Adaptation [Hu *et al.*, 2022]. We also fine-tune our own LLaMA and LLaMA2 models for abstract reasoning. For all models, we evaluate the 7B parameters versions by default. Finally, we evaluate the more recent Zephyr-7B-β [Tunstall *et al.*, 2023a; Tunstall *et al.*, 2023b], a 7B parameters model fine-tuned from Mistral-7B [Jiang *et al.*, 2023]. We also compare these generic models on architecture fine-tuned for Multiple-Choice Question Answering. They discriminate the solution from a small set of options, unlike text completion engines that produce an output text. We fine-tune two models: RoBERTa-large [Liu *et al.*, 2019], a language model used for text comprehension, and MERIt [Jiao *et al.*, 2022], a model using contrastive pre-training on rules-based data to perform logical reasoning.

4 Experiments

4.1 Open-Ended Question Answering

In this section, we detail our experiments on open-ended abstract reasoning. Depending on the dataset, the answer can be in natural language or a symbolic format. The model is asked to provide the answer directly. The accuracy for each model on every dataset is summarised in Table 3a.

Our results indicate poor performance of language models on all the presented datasets, although the performance varies between datasets and models. In particular, Text-Davinci-3 and GPT-4 consistently achieve the best performance across the datasets. Zephyr-7B-β has almost systematically the best accuracy among open models. On the other hand, LLaMA-7B has the worst performance of all models. LLaMA2-7B gets a similar accuracy except on BIG-Bench. Alpaca and Alpaca-LoRA present slight improvements on BIG-Bench-F, PVR and RAVEN^T. This improvement is explained by the instruction-tuning used to build Alpaca and Alpaca-LoRA. We provide several examples in the appendix that illustrate this difference. LLaMA-7B often does not attempt to solve the problem but completes the text by giving more examples. These examples do no match the abstract rule for the task. Alpaca and Alpaca-LoRA follow the instructions more faithfully but also fail to grasp the abstract patterns. Instruction-tuning seems to help the model understand the format of the answer and what it is asked to do but provides little help on how to solve the tasks. Moreover, the performance difference between Text-Davinci-3 and GPT-3.5-Turbo indicates that the type of instruction-tuning matters as Text-Davinci-3 performs systematically better than GPT-3.5-Turbo despite being based on the same model. Overall, GPT-4 performs noticeably better than all the other models. As the details of its architecture and training set are unavailable, we cannot provide satisfactory explanations for this difference. However, the increase in performance is highest on the RAVEN^T dataset. Given that Raven matrices are a standard and long-existing test [Raven, 1938; Carpenter *et al.*, 1990], we can hypothesize that the training data of GPT-4 included some versions of the test. The same remark can be made for BIG-Bench-F as it includes traditional list processing algorithms. Text-Davinci-3 and GPT-4 also achieve good performance on the ARC^T dataset relative to other existing architectures challenged on the task, making them 11^{th} and 14^{th} on the Kaggle leaderboard[1]. However, they still fail to answer a vast majority of the tasks correctly. All LLMs generally fail to answer most of the tasks in each dataset. Despite a performance increase compared to previous versions, the most recent language models do not perform open-ended abstract reasoning well.

4.2 Multiple-Choice Question Answering

As seen in Section 4.1, open-ended abstract reasoning is a challenging problem for language models. We also perform a series of experiments on Multiple-Choice Question Answering tasks where the models are given a set of possible answers and must pick a single one from the set. This task is more accessible than Open-Ended QA, as the valid response is given as part of the input. Results are given in Table 4.

We first compare the results of RAVEN^T-mcqa and RAVEN^T-opqa from Table 3a. RAVEN^T-opqa contains the same questions as RAVEN^T-mcqa, but the answer choices have been removed. Following intuition, giving multiple choices to LLMs helps systematically improve their performance. Only the performance of LLaMA remains the same, and the performance of Alpaca and Zephyr-7B-β are slightly reduced. Given the low accuracy in both cases, it can be interpreted as noise. MCQA models achieve slightly above random performance (see details in appendix), performing better than most LLMs. However, they have an advantage compared to completion engines as they have to select one answer among a list of possible choices, whereas completion models must generate the correct answer. Therefore, the lat-

[1] https://www.kaggle.com/competitions/abstraction-and-reasoning-challenge/leaderboard

	ARC^T	BBF	Evals-S	PVR	RAVEN^T-opqa	
					Text	Symb
Text-Davinci-3	*0.105*	*0.404*	**0.314**	**0.228**	*0.343*	*0.234*
GPT-3.5-Turbo	0.033	0.153	0.186	0.124	0.226	0.161
GPT-4	**0.119**	**0.514**	*0.304*	0.177	**0.410**	**0.330**
LLaMA-7B	0.010	0.012	0.014	0.060	0.000	0.000
LLaMA2-7B	0.005	0.108	0.000	0.000	0.000	0.001
Alpaca	0.010	0.188	0.014	0.184	0.075	0.030
Alpaca-LoRA	0.012	0.144	0.000	0.152	0.000	0.067
Zephyr-7B-β	0.015	0.292	0.043	*0.209*	0.009	0.145

(a) Accuracy of Large Language Models on Open QA datasets.

	BBF	PVR	RAVEN^T-opqa	
			Text	Symb
GPT-3.5-Turbo-cot	*0.097*	**0.210**	*0.302*	*0.211*
GPT-4-cot	**0.476**	*0.174*	**0.385**	**0.354**
Alpaca-LoRA-cot	0.084	0.152	0.000	0.069

	ACRE^T		RAVEN^T-mcqa	
	Text	Symb	Text	Symb
GPT-3.5-Turbo-cot	**0.255**	*0.345*	*0.257*	*0.144*
GPT-4-cot	*0.214*	**0.394**	**0.596**	**0.517**
Alpaca-LoRA-cot	*0.059*	0.114	0.000	0.114

(b) Accuracy with *Chain-of-Thought* prompting.

Table 3: Main results on Open QA and MCQA datasets. Datasets are represented in columns, and models in rows. The best result for each dataset is indicated in **bold**, and the second best is indicated in *italics*. BBF stands for BIG-Bench-F.

	ACRE^T		Evals-P	RAVEN^T-mcqa	
	Text	Symb		Text	Symb
GPT-2	**0.371**	0.00	0.496	0.00	0.126
Text-Davinci-3	0.098	0.427	*0.560*	*0.461*	*0.452*
GPT-3.5-Turbo	0.184	0.445	0.481	0.276	0.315
GPT-4	*0.272*	*0.512*	**0.625**	**0.697**	**0.535**
LLaMA-7B	0.000	0.257	0.544	0.004	0.000
LLaMA2-7B	0.014	0.003	0.500	0.026	0.149
Alpaca	0.036	0.238	0.544	0.015	0.058
Alpaca-LoRA	0.015	0.123	0.552	0.082	0.124
Zephyr-7B-β	0.106	**0.516**	0.504	0.000	0.022
random	0.33	0.33	0.5	0.125	0.125

Table 4: Accuracy of Large Language Models for Multiple-Choice QA on the ACRE^T, Evals-P and RAVEN^T datasets. The last line indicates random performance. Completion models can perform worse than random if they do not reply with a valid answer. The best result for each dataset is indicated in **bold**, and the second best is indicated in *italics*.

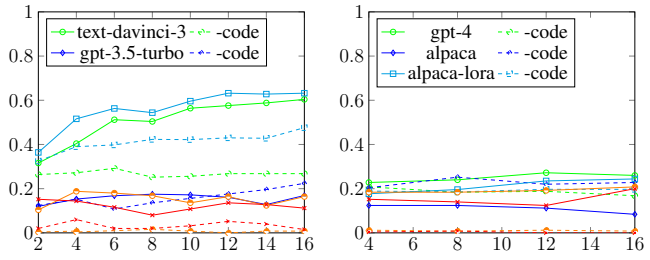
ter may not return any valuable output (e.g. a nonsensical or empty answer), explaining how they can achieve worse than random performance. By constrast, the smaller GPT-2 obtains random performance by returning a *plausible* answer keyword instead of attempting to comprehend the task. The main takeaway from these experiments is that the performance of LLMs remains low even in discriminative settings. When given a set of possible answers, the models cannot recognise the proper solution among the other choices. This finding indicates that using LLMs as evaluators (as done in self-refinement techniques [Madaan *et al.*, 2023]) is not suited for tasks requiring abstract reasoning. We confirm this with additional experiments in Section 4.7 using different refinement strategies. Additionally, when comparing the results between natural language and symbolic tasks on ACRE^T, we observe that the results are better across all models when the input is symbolic. Inputs that use symbolic data are smaller and may convey only relevant information, while natural language could contain distracting information or biases harmful to task performance. The same observation can be made concerning RAVEN^T-mcqa, except for GPT-4. In the open-ended version of RAVEN^T, models perform better with the natural language representation. Without the answer set available, inductive biases caused by language help performance.

4.3 Chain-of-Thought Prompting

We perform experiments on a subset of our framework using *Chain-of-Thought* prompting [Wei *et al.*, 2022]. The complete experiments are provided in the appendix (and include a side-by-side comparison for better readability). We perform experiments with GPT-3.5-turbo, GPT-4, and Alpaca-LoRA. Our experiments with *Chain-of-Thought* have the suffix *model*-cot. Our results are presented in Table 3b. Overall, the results obtained using *Chain-of-Thought* prompting are not higher than those obtained with the base models. On The BIG-Bench-F dataset, the *Chain-of-Thought* versions achieve systematically lower performance than their base counterparts, although no no significant performance drop is observed. On PVR and RAVEN^T-opqa, while the accuracy for GPT-4 and Alpaca-LoRA remain unchanged or slightly reduced, the performance of GPT-3.5-Turbo is increased. On RAVEN^T-mcqa, the performance of all the models decreases. These experiments show that the quality of the prompt has little impact on the answer quality. It hints that the models can understand the instructions but that their failures are due to their unability to provide faithful reasoning. This limitation is further illustrated with examples in the appendix.

4.4 Fine-tuning LLaMA2

We now study the performance of LLaMA2 models after fine-tuning on RAVEN^T-mcqa. Experiments on more datasets are provided in the appendix. The training and test sets may share distribution-specific patterns that the model may learn during the fine-tuning phase. It may overfit on these patterns instead of learning the correct abstract patterns. To alleviate this pitfall, we generate out-of-distribution (o.o.d) splits. The *-Four* split contains samples with four figures instead of one. The *-In-Center* splits contains samples with two figures instead of one, a big and a small located within the former. The shape and colours of the figures all are observed in the training set. The two splits can be considered as compositional splits. The results on RAVEN^T-mcqa are shown in Table 5. We observe a significant increase in the accuracy on the test set. LLaMA2 achieves close to perfect accuracy. The performance partially transfers to the alternative syntax task. We now observe the performance on the o.o.d splits. The performance of the fine-tuned LLaMA2 significantly drops on the new tasks, showing a lack of generalisation. We can deduce that fine-tuning



(a) BIG-Bench-F dataset. (b) PVR dataset.

Figure 2: Evolution of the model accuracy as a function of the number of context examples seen. The legend is shared by both figures. Models with straight lines are used with default prompting, while models with dashed lines are prompted to produce code.

yields representations that are highly invariant to the syntax but does not transfer other abstract reasoning abilities. The rules required to solve the *-Four* and *-In-Center* splits manipulate several figures, they are compositions of rules used for single figures. LLMs can compose with unseen quantities (e.g. new syntax) but have more difficulty composing new abstract rules.

4.5 Varying the Example Set Size

We perform further experiments on the BIG-Bench-F and PVR datasets. For these two datasets, we alter the number of examples given to the system before the test case. By default, we give four examples to the model before asking it to answer. The results are shown in Figures 2a and 2b. In this section, we focus on the results of the base models (without the "-code" suffix). We first observe that, for both datasets, there is no linear relationship linking performance and number of examples. For all but the Text-Davinci-3 and GPT-4 models, adding more examples has little or no effect on the accuracy. Text-Davinci-3 and GPT-4 perform similarly across all cases, and their performances consistently increase with the number of examples, achieving up to an accuracy of 0.6 when given 16 examples on the BIG-Bench-F dataset. However, on PVR, Text-Davinci-3 achieves only 0.26 when given 12 examples. GPT-4 follows a similar trend but performs slightly worse than its predecessor. In the absence of technical details for GPT-4, we can only speculate on the reasons. As this effect is observed only on BIG-Bench-F and not PVR, we can assume that the models perform better because their training sets contain the list processing algorithms used by BIG-Bench-F. We perform additional experiments in the appendix, where we provide solved instances into the prompt (input and solution program) to propel the model to reason correctly. No real improvements are observed.

4.6 Enabling Structure Discovery with Code

In the next experiments, we follow an idea similar to *Progam-of-Thought* prompting [Chen *et al.*, 2022] and ask the model to generate the code of the function responsible for generating the output from the input. Then, we execute the produced code on the test case and evaluate the result. This method differs from a base prompt as we do not ask the model to produce the answer directly. This part is delegated to a code interpreter in Python. This method aims to verify the ability of LLMs to extract the correct structure behind each abstract reasoning task under code format. We test this method on the BIG-Bench-F and PVR datasets. The results of these models (with the "-code" suffix) can be compared with their original counterparts in Figures 2a and 2b. In general, we observe that the models prompted to produce code perform worse than those tasked to produce the answer directly. The only exception is GPT-3.5-Turbo. On the BIG-Bench-F dataset, the performance of GPT-3.5-Turbo-code increases steadily while that of GPT-3.5-Turbo stagnates, and on PVR, GPT-3.5-Turbo-code outperforms GPT-3.5-Turbo by a significant margin. Producing code solving the abstract problem is a more complicated task for an LLM as it requires the model to produce a rigorous code explanation for its answer. It is consistent with the results for most models, but we also observe in the case of GPT-3.5-Turbo-code that it can help the model better understand the task. On BIG-Bench-F, the code versions of Text-Davinci-3 and GPT-4 perform better than both base and code versions of the other models. As this behaviour is not observed with PVR, we infer that this performance is due to the functions being part of the training sets of the models. The models can almost always generate code able to compile and produce an answer (details are in the appendix). We deduce that producing a program with a valid syntax is not a bottleneck for performance. The issue lies in the recovery of the correct reasoning process.

4.7 Refinement

In this section, we investigate prompt-tuning strategies based on refinement and filtering that have been successful in improving LLM reasoning abilities and see if they can be used to improve abstract reasoning performance. We study two types of strategies: code-*based* and *self-based*. *Code-filtering* is a code-based strategy that consists of generating multiple code responses and filtering out the programs that cannot solve the example cases. *Code-refinement* [Wang *et al.*, 2023; Qiu *et al.*, 2023] is an iterative process where the model generates a first program. The program is run on the context examples and, if not all answers are correct, the model is prompted to correct its answer based on the output of the interpreter. *Self-filtering* and *self-refinement* [Qiu *et al.*, 2023; Madaan *et al.*, 2023] are similar self-based techniques. They ask the LLM to assess whether the given answer is correct rather than relying on an interpreter. We conduct experiments on BIG-Bench-F and PVR using GPT-4-Turbo (gpt-4-1106-preview). Additional experiments are provided in the appendix.

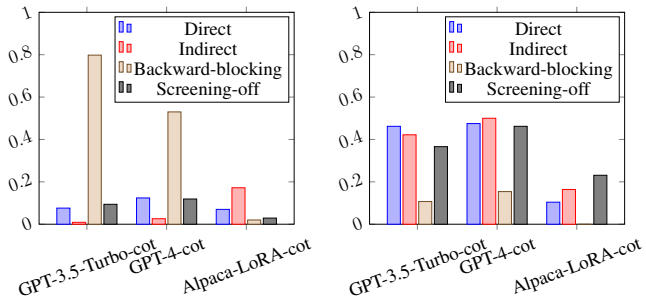
Table 6 shows the main results. Overall, the improvements brought by the refinement strategies are limited. The bottleneck in the reasoning is the recognition of the abstract rule linking the context examples. Therefore, the LLM cannot be a good evaluator. This is consistent with the MCQA results observed in Section 4.2 where the LLMs fail to discriminate the good answers. Unlike self-refinement, self-filtering generates multiple answers independently, not conditioned on the previous iterations. As the LLM performance as a discriminator is above chance, the filtering process can help improving the performance. Code-refinement provides slight improve-

Model	RAVEN^T-Eval		-Four		-In-Center	
	Text	Symb	Text	Symb	Text	Symb
LLaMA2-7B	0.135	0.114	0.073	0.121	0.000	0.001
LLaMA2-7B-AR-LoRA-Text*	**0.977**	0.694	**0.557**	**0.522**	0.536	**0.085**
LLaMA2-7B-AR-LoRA-Symb*	0.965	**0.938**	0.498	0.442	**0.767**	0.064

Table 5: Accuracy of base and fine-tuned LLaMA2 on the RAVEN^T-mcqa dataset i.i.d and o.o.d splits. The base LLaMA2 is compared against counterparts fine-tuned on RAVEN^T-mcqa i.i.d text and symbolic training sets. The best result for each dataset in indicated in **bold**.

	BIG-Bench-F	PVR
GPT-4-Turbo-code	0.280	**0.152**
GPT-4-Turbo-code-filtering	**0.400**	**0.152**
GPT-4-Turbo-code-refinement	0.296	0.144
GPT-4-Turbo	0.268	0.000
GPT-4-Turbo-self-filtering	0.284	0.004
GPT-4-Turbo-self-refinement	0.252	0.000

Table 6: Accuracy of refined Large Language Models on BIG-Bench-F and PVR datasets. The best result for each dataset is indicated in **bold**. Experiments are performed with the latest version of GPT-4 (*gpt-4-1106-preview*).



(a) Text ACRE^T. (b) Symbolic ACRE^T.

Figure 3: Accuracy of chain-of-thought models on ACRE^T divided by causal paths.

ments in the accuracy for BIG-Bench but decreases it for PVR. The LLMs struggle to accurately exploit the feedback from the interpreter. On BIG-Bench, code-filtering improves the performance the most. The reasons are similar to the self-filtering strategy although the code interpreter is a more rigorous discriminator.

4.8 A Perspective from Causal Induction

We perform further analysis on ACRE^T. The dataset can be divided into four causal paths: Direct, Indirect, Backward-blocking, Screening-off [Zhang *et al.*, 2021a]. Direct path queries can be established using direct evidence. Indirect paths require the combination of multiple pieces of evidence. Backward-blocking paths cannot be determined because the true mechanisms cannot be discriminated from other possibilities based solely on the data. Screening-off paths are causal paths affected by spurious correlations. Figure 3 shows the results for each type of query. We restrict our analysis to the *Chain-of-Thought* models (see the appendix for the full analysis). Although accuracy scores are similar, the distribution of the results among the causal paths differs between models and input types. GPT models overfit to backward-blocking cases on the text ACRE^T but not on the symbolic version. We can deduce that natural language contains distracting information or biases harmful to abstract reasoning performance. It is consistent with the higher score of the models on the symbolic tasks.

5 Conclusion

Understanding the potential reasoning capabilities of LLMs is crucial as they are starting to be widely adopted. Measuring the level of intelligence of a system is hard, but abstract reasoning provides a valuable framework for this task. In this paper, we present what is, to the best of our knowledge, the first extensive evaluation of Large Language Models for abstract reasoning. We show that LLMs do not perform well on all types of tasks, although not all models are equally poor. Prompting and refinement techniques that improve performance on NLP tasks do not work for abstract reasoning. From our experiments, we argue that the bottleneck in the performance lies in the recognition of new unseen abstract patterns and not in a lack of understanding of the task or the prompt. For instance, the experiments with Program-of-Thought show that almost all the generated programs compile and return an answer with the expected format, even if the answer is incorrect. The presented solutions are extremely grounded to the context examples, even when increasing the number of examples. These results hold in discriminative settings, where the models must find the correct answer within a small set of propositions. A qualitative study of selected failure cases in the appendix further reveals that models tend to reason inconsistently and in a shallow way. Models also tend to produce convoluted reasonings that can match the input but hardly generalise to new instances. We hypothesise that current self-supervised autoregressive LLMs lack fundamental properties for strong abstract reasoning tasks and human-like cognition. In particular, we posit that methods based on causal reasoning and program induction could help improve the reasoning abilities of LLMs.

References

[Acquaviva *et al.*, 2021] Samuel Acquaviva, Yewen Pu, Marta Kryven, Catherine Wong, Gabrielle E. Ecanow, Maxwell I. Nye, Theodoros Sechopoulos, Michael Henry Tessler, and Joshua B. Tenenbaum. Communicating natural programs to humans and machines. *CoRR*, abs/2106.07824, 2021.

[Bender *et al.*, 2021] Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. On the dangers of stochastic parrots: Can language models

be too big? In *FAccT '21: 2021 ACM Conference on Fairness, Accountability, and Transparency, Virtual Event / Toronto, Canada, March 3-10, 2021*, pages 610–623. ACM, 2021.

[Booch *et al.*, 2021] Grady Booch, Francesco Fabiano, Lior Horesh, Kiran Kate, Jonathan Lenchner, Nick Linck, Andreas Loreggia, Keerthiram Murgesan, Nicholas Mattei, Francesca Rossi, et al. Thinking fast and slow in AI. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*, 2021.

[Brown *et al.*, 2020] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. In *NeurIPS*, 2020.

[Bubeck *et al.*, 2023] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, et al. Sparks of artificial general intelligence: Early experiments with GPT-4. *CoRR*, abs/2303.12712, 2023.

[Carpenter *et al.*, 1990] Patricia A Carpenter, Marcel A Just, and Peter Shell. What one intelligence test measures: a theoretical account of the processing in the raven progressive matrices test. *Psychological review*, 97(3):404, 1990.

[Chen *et al.*, 2022] Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W. Cohen. Program of thoughts prompting: Disentangling computation from reasoning for numerical reasoning tasks. *CoRR*, abs/2211.12588, 2022.

[Chollet, 2019] François Chollet. On the measure of intelligence. *CoRR*, abs/1911.01547, 2019.

[Devlin *et al.*, 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186. Association for Computational Linguistics, 2019.

[Ellis *et al.*, 2020] Kevin Ellis, Catherine Wong, Maxwell I. Nye, Mathias Sablé-Meyer, Luc Cary, Lucas Morales, Luke B. Hewitt, Armando Solar-Lezama, and Joshua B. Tenenbaum. Dreamcoder: Growing generalizable, interpretable knowledge with wake-sleep bayesian program learning. *CoRR*, abs/2006.08381, 2020.

[Gendron *et al.*, 2023] Gaël Gendron, Michael Witbrock, and Gillian Dobbie. A survey of methods, challenges and perspectives in causality. *CoRR*, abs/2302.00293, 2023.

[Goyal and Bengio, 2020] Anirudh Goyal and Yoshua Bengio. Inductive biases for deep learning of higher-level cognition. *CoRR*, abs/2011.15091, 2020.

[Hu *et al.*, 2022] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022*. OpenReview.net, 2022.

[Jiang *et al.*, 2023] Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *CoRR*, abs/2310.06825, 2023.

[Jiao *et al.*, 2022] Fangkai Jiao, Yangyang Guo, Xuemeng Song, and Liqiang Nie. Merit: Meta-path guided contrastive learning for logical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 3496–3509. Association for Computational Linguistics, 2022.

[Jin *et al.*, 2023] Zhijing Jin, Jiarui Liu, Zhiheng Lyu, Spencer Poff, Mrinmaya Sachan, Rada Mihalcea, Mona T. Diab, and Bernhard Schölkopf. Can large language models infer causation from correlation? *CoRR*, abs/2306.05836, 2023.

[Kiciman *et al.*, 2023] Emre Kiciman, Robert Ness, Amit Sharma, and Chenhao Tan. Causal reasoning and large language models: Opening a new frontier for causality. *CoRR*, abs/2305.00050, 2023.

[Lake *et al.*, 2015] Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

[Li and Flanigan, 2024] Changmao Li and Jeffrey Flanigan. Task contamination: Language models may not be few-shot anymore. In Michael J. Wooldridge, Jennifer G. Dy, and Sriraam Natarajan, editors, *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024*, 2024.

[Li *et al.*, 2022] Yifei Li, Zeqi Lin, Shizhuo Zhang, Qiang Fu, Bei Chen, Jian-Guang Lou, and Weizhu Chen. On the advance of making language models better reasoners. *CoRR*, abs/2206.02336, 2022.

[Liu *et al.*, 2019] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019.

[Madaan *et al.*, 2023] Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, et al. Self-refine: Iterative refinement with self-feedback. *CoRR*, abs/2303.17651, 2023.

[OpenAI, 2023] OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023.

[Ouyang *et al.*, 2022] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. In *NeurIPS*, 2022.

[Qiu *et al.*, 2023] Linlu Qiu, Liwei Jiang, Ximing Lu, Melanie Sclar, Valentina Pyatkin, Chandra Bhagavatula, Bailin Wang, Yoon Kim, Yejin Choi, Nouha Dziri, et al. Phenomenal yet puzzling: Testing inductive reasoning capabilities of language models with hypothesis refinement. *CoRR*, abs/2310.08559, 2023.

[Radford *et al.*, 2019] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[Raven, 1938] John C Raven. Raven standard progressive matrices. *Journal of Cognition and Development*, 1938.

[Rule, 2020] Joshua Stewart Rule. *The child as hacker: building more human-like models of learning*. PhD thesis, Massachusetts Institute of Technology, 2020.

[Santoro *et al.*, 2018] Adam Santoro, Felix Hill, David G. T. Barrett, Ari S. Morcos, and Timothy P. Lillicrap. Measuring abstract reasoning in neural networks. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018*, 2018.

[Schölkopf *et al.*, 2021] Bernhard Schölkopf, Francesco Locatello, Stefan Bauer, Nan Rosemary Ke, Nal Kalchbrenner, Anirudh Goyal, and Yoshua Bengio. Toward causal representation learning. *Proc. IEEE*, 109(5):612–634, 2021.

[Srivastava *et al.*, 2022] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Md Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adrià Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *CoRR*, abs/2206.04615, 2022.

[Stolfo *et al.*, 2022] Alessandro Stolfo, Zhijing Jin, Kumar Shridhar, Bernhard Schölkopf, and Mrinmaya Sachan. A causal framework to quantify the robustness of mathematical reasoning with language models. *CoRR*, abs/2210.12023, 2022.

[Talmor *et al.*, 2020] Alon Talmor, Oyvind Tafjord, Peter Clark, Yoav Goldberg, and Jonathan Berant. Leap-of-thought: Teaching pre-trained models to systematically reason over implicit knowledge. In *NeurIPS*, 2020.

[Tang *et al.*, 2023] Xiaojuan Tang, Zilong Zheng, Jiaqi Li, Fanxu Meng, Song-Chun Zhu, Yitao Liang, and Muhan Zhang. Large language models are in-context semantic reasoners rather than symbolic reasoners. *CoRR*, abs/2305.14825, 2023.

[Taori *et al.*, 2023] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model, 2023.

[Tirumala *et al.*, 2022] Kushal Tirumala, Aram H. Markosyan, Luke Zettlemoyer, and Armen Aghajanyan. Memorization without overfitting: Analyzing the training dynamics of large language models. In *NeurIPS*, 2022.

[Touvron *et al.*, 2023a] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *CoRR*, abs/2302.13971, 2023.

[Touvron *et al.*, 2023b] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288, 2023.

[Tunstall *et al.*, 2023a] Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Kashif Rasul, Younes Belkada, Shengyi Huang, Leandro von Werra, Clémentine Fourrier, Nathan Habib, et al. Zephyr: Direct distillation of LM alignment. *CoRR*, abs/2310.16944, 2023.

[Tunstall *et al.*, 2023b] Lewis Tunstall, Edward Beeching, Nathan Lambert, Nazneen Rajani, Alexander M. Rush, and Thomas Wolf. The alignment handbook. https://github.com/huggingface/alignment-handbook, 2023.

[Wang *et al.*, 2022] Yizhong Wang, Yeganeh Kordi, Swaroop Mishra, Alisa Liu, Noah A. Smith, Daniel Khashabi, and Hannaneh Hajishirzi. Self-instruct: Aligning language model with self generated instructions. *CoRR*, abs/2212.10560, 2022.

[Wang *et al.*, 2023] Ruocheng Wang, Eric Zelikman, Gabriel Poesia, Yewen Pu, Nick Haber, and Noah D. Goodman. Hypothesis search: Inductive reasoning with language models. *CoRR*, abs/2309.05660, 2023.

[Wei *et al.*, 2022] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In *NeurIPS*, 2022.

[Wu *et al.*, 2023] Zhaofeng Wu, Linlu Qiu, Alexis Ross, Ekin Akyürek, Boyuan Chen, Bailin Wang, Najoung Kim, Jacob Andreas, and Yoon Kim. Reasoning or reciting? exploring the capabilities and limitations of language models through counterfactual tasks. *CoRR*, abs/2307.02477, 2023.

[Xu *et al.*, 2023] Fangzhi Xu, Qika Lin, Jiawei Han, Tianzhe Zhao, Jun Liu, and Erik Cambria. Are large language models really good logical reasoners? A comprehensive evaluation from deductive, inductive and abductive views. *CoRR*, abs/2306.09841, 2023.

[Zecevic *et al.*, 2023] Matej Zecevic, Moritz Willig, Devendra Singh Dhami, and Kristian Kersting. Causal parrots: Large language models may talk causality but are not causal. *CoRR*, abs/2308.13067, 2023.

[Zhang *et al.*, 2019] Chi Zhang, Feng Gao, Baoxiong Jia, Yixin Zhu, and Song-Chun Zhu. RAVEN: A dataset for relational and analogical visual reasoning. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019*. Computer Vision Foundation / IEEE, 2019.

[Zhang *et al.*, 2021a] Chi Zhang, Baoxiong Jia, Mark Edmonds, Song-Chun Zhu, and Yixin Zhu. ACRE: abstract causal reasoning beyond covariation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021*. Computer Vision Foundation / IEEE, 2021.

[Zhang *et al.*, 2021b] Chiyuan Zhang, Maithra Raghu, Jon M. Kleinberg, and Samy Bengio. Pointer value retrieval: A new benchmark for understanding the limits of neural network generalization. *CoRR*, abs/2107.12580, 2021.