Fast and flexible: Human program induction in abstract reasoning tasks

Aysja Johnson[1] **(aysja.johnson@nyu.edu), Wai Keen Vong**[2] **(waikeen.vong@nyu.edu),**
Brenden M. Lake[1,2] **(brenden@nyu.edu), & Todd M. Gureckis**[1] **(todd.gureckis@nyu.edu)**
[1]Department of Psychology, New York University; [2]Center for Data Science, New York University

Abstract

The Abstraction and Reasoning Corpus (ARC) is a collection program induction tasks that was recently proposed by Chollet (2019) as a measure of machine intelligence. Here, we report a preliminary set of results from a behavioral study of humans solving a subset of tasks from ARC (40 out of 1000). We found that humans were able to infer the underlying program and generate the correct test output for a novel test input example, with an average of 84% of tasks solved per participant, and with 65% of tasks being solved by more than 80% of participants. Additionally, we find interesting patterns of behavioral consistency and variability across the action sequences to generate their responses, the natural language descriptions used to describe the rule for each task, and the errors people make. Our findings suggest that people can quickly and reliably determine the relevant features and properties of a task to compose a correct solution, despite limited experience in this domain. This dataset offers useful insights for designing AI systems that can solve abstract reasoning tasks such as ARC with the fluidity of human intelligence.

Keywords: concept learning, abstract reasoning, compositionality, program induction

Introduction

Despite recent advances in AI, contemporary systems still lack the ingenuity and flexibility of human intelligence. One factor limiting progress in the field of AI is the focus on narrow benchmark challenge problems (e.g., ImageNet (Deng et al., 2009), Arcade Learning Environment (Bellemare, Naddaf, Veness, & Bowling, 2013)). Although astounding progress has been made on such challenge problems, they often don't require the type of flexible, creative, and abstract reasoning that is the hallmark of human cognition (Hernandez-Orallo, Martinez-Plumed, Schmid, Siebers, & Dowe, 2016). As impressive as performance on ImageNet is, correctly identifying objects in photographs is unlikely to be the central assessment in a meaningful test of human intelligence.

To address this and other concerns, Chollet (2019) proposed a novel type of machine learning challenge problem named the Abstraction and Reasoning Corpus (or ARC). ARC emphasizes a set of perceptually simple, yet conceptually challenging, program induction tasks. Each task involves 2-6 input grids that undergo some kind of transformation to a corresponding output grid. The agent is tasked with using this small sample of input-output relationships to infer the underlying program or procedure that modifies the input to create the output. Performance is tested by presenting the agent with a novel test input grid, and tasking it to generate the expected output grid from scratch (see Figure 1).

Several aspects of ARC distinguish it from standard machine learning benchmarks. First, each of the 1000 tasks in the corpus is unique and rely on many different aspects of conceptual knowledge, including: objects, relations, geometry, number, symmetry, and quantification. Second, agents

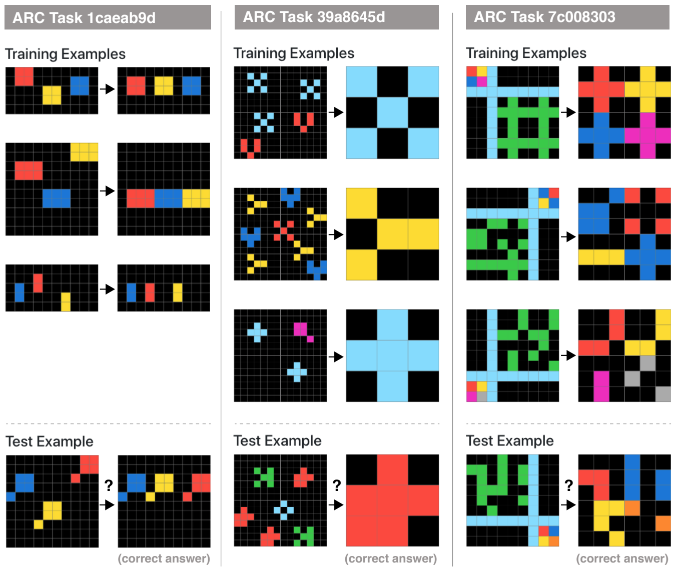

Figure 1: **Three example ARC tasks**. Each column contains a set of "training examples" and a "test example" for an ARC problem. The training examples contains a number of input patterns that point (→) to the corresponding output. Here there are always three input→output pairs but there can be fewer or more. The rules for transforming the input examples to the output examples are unique for each ARC task. The left task (which we refer to later as the **box alignment task**) requires aligning the red and yellow objects to the blue object along the vertical axis. The middle task requires counting each object and returning the most common object. The right task requires mapping the four colors in the 2x2 quadrant onto each quadrant of the green object. Understanding of the pattern is assessed with the text example. Here a single test input is provided and the agent has to create the expected output. The correct answer is displayed here for each problem but the agent does not have access to this.

need to manually generate their own response, including specifying the size of the output grid (grids range in size from 1x1 to 30x30), and the colors for each cell (there are 10 possible colors), making the task substantially more open-ended than traditional machine learning challenge problems. Finally, successful solutions to many ARC problems rely on a type of abductive reasoning (Peirce, 1935), requiring the agent to come up with plausible hypotheses that often leverage compositional rules and relationships between identified objects and parts. As a result, ARC presents a compelling task environment for studying aspects of higher-level cognition and intelligence.

The winner from a recent Kaggle challenge for ARC[1], using a program synthesis approach, was only able to solve 21% of the tasks from the test set, providing a sense of the

[1]The solution is hosted on their GitHub repository: https://github.com/top-quarks/ARC-solution

difficulty for generalization to new tasks with such a limited number of examples. The Kaggle algorithm was built using a domain-specific language (DSL) including functions such as 'Move', 'getSize', and 'count', all specified by hand. While we focus on this algorithm as a baseline for program synthesis approaches and for solutions to ARC more generally, we do not consider the Kaggle algorithm to be representative of a cognitively plausible approach nor do we believe it represents the upper bound of accuracy for approaches of this type.

Chollet (2019) reported that each ARC task was solvable among a subset of three individual (humans) tested, however this provides us with little information for which tasks are easier or harder for people on average, nor what kinds of strategies people would use to generate their solutions. In this paper, we report the first behavioral dataset collected on human performance in ARC examining human performance on a subset of 40 tasks from the ARC training set, with multiple participants per task. Our goals were two-fold. First, we were interested in a more precise assessment of how well people would perform on this benchmark given its reputation as a challenging task for existing AI systems. Second, we were interested in examining the different aspects of human cognition that underlie the human ability to solve ARC tasks in this challenging few-shot learning setting.

To foreshadow, our results show that humans perform well on ARC—each task we studied was solvable by at least one participant and the average accuracy across all of the tasks was 83.8%. In addition, we examined a wide variety of other behaviors generated by people in solving these tasks, ranging from the sequence of actions participants performed to generate the correct output, to natural language descriptions provided about the concepts underlying each task, to the kinds of errors people made. These behavioral phenomena provided us with additional windows to understanding the kinds of inductive biases that contribute to people's success on ARC.

Program Induction, Abduction, and Probabilistic Language of Thought

Before describing our experiment, in this section, we consider the ARC challenge in the context of prior work on program induction in cognitive science.

Within cognitive science, much of the work that is most reminiscent of ARC has been in the development of probabilistic language-of-thought (pLOT) models (Piantadosi, 2011; Goodman, Tenenbaum, & Gerstenberg, 2015; Piantadosi & Jacobs, 2016). These models assume that hypotheses can be represented as programs specified in a representation language such as first-order logic or lambda calculus. The goal of these kinds of models is to infer the underlying program using Bayesian inference, trading off between the complexity of the program and how well it captures the data. They have been used for modeling a variety of concept learning tasks, ranging from Boolean concepts (Goodman, Tenenbaum, Feldman, & Griffiths, 2008) to complex probabilistic programs (Piantadosi, Tenenbaum, & Goodman, 2016; Bramley, Rothe, Tenenbaum, Xu, & Gureckis, 2018).

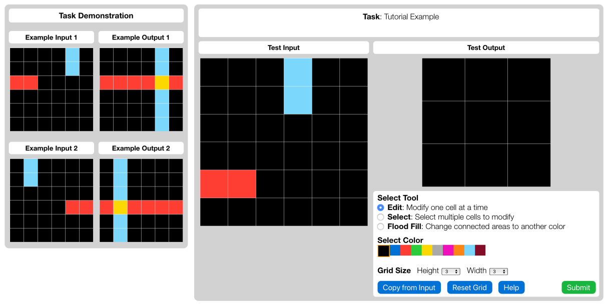

Figure 2: **ARC User Interface.** On each task, participants were presented with a limited number of example input-output pairs on the left, and their goal was to generate the correct output for a new test input presented on the right. Here we demonstrate the tutorial task presented to participants, where the rule is to extend the red line horizontally, the cyan line vertically, and to color the intersection yellow. The user interface contained a number of different tools for participants to flexibly generate solutions to ARC tasks.

In many experiments within this framework, the set of primitives and the underlying grammar is itself used to generate the concepts studied in experiments. Therefore the set of concepts studied are limited by programs that are easily expressed with a specific grammar. In contrast, the tasks in ARC were hand-designed without reference to a pre-specified grammar. Thus, the types of concepts in ARC are more *abductive* in nature, rather than *inductive*, requiring people to flexibly generate new rules or concepts on the fly, by determining the relevant features and variables to solving a given task. This style of abstract reasoning is one that is less well understood in cognitive science from a computational standpoint (Mitchell, 2021), although it is similar to other open-ended puzzles such as Bongard problems where the set of potential rules is also unconstrained (Hofstadter et al., 1979).

Another aspect of ARC, as mentioned above, is that agents are required to generate the correct response from scratch. Few prior studies have looked at people's ability to generate outputs from novel inputs based on a limited set of training examples (exceptions include Hofstadter and Mitchell (1988)'s work on CopyCat and recent work on program induction, Lake, Linzen, & Baroni, 2019; Rule, Schulz, Piantadosi, & Tenenbaum, 2018). This generative aspect of ARC adds another layer of complexity to the problem, but may provide greater insight into people's mental representations compared to simpler forced-choice judgments.

Experiment

Methods

Participants. We recruited 95 participants (57.7% male, 39.9% female, 2.4% other) from Amazon Mechanical Turk using the psiTurk platform (Gureckis et al., 2016). Participants varied in age from 21–70 years (*M* = 39.3, *SD* = 10.3). They were compensated $7.50 plus a potential one dollar bonus if they succeeded at a randomly selected task with an adequately descriptive written solution (see below).

Design. Forty tasks were randomly selected from a restricted portion of the training set of the Abstraction and Reasoning Corpus (Chollet, 2019), capturing a variety of the kinds of abstract reasoning required to solve these tasks. We restricted ourselves to sampling from tasks which only had a single test item, and where the output grid was no larger than 15x15. Due to time constraints, each participant was randomly assigned to complete 10 out of the 40 selected tasks, resulting in 23.5 participants solving each task on average.

Procedure. Participants were first provided with instructions about the experiment and the user interface, followed by a tutorial task as shown in Figure 2. To continue, participants had to generate the correct test output for the tutorial task. They were then required to answer three comprehension questions correctly (e.g., "how many attempts per task will you get?") in order to advance to the experiment.

The main experiment consisted of 10 ARC tasks, randomly selected from the set of 40 tasks described above. For each task, participants were presented with between 2 to 6 input-output pairs for training (the exact number depended on the specific task), and a single test input. The goal was to generate the correct test output for each task from scratch, starting from a blank 3x3 grid, using a variety of tools with the built-in editor.

The overall procedure was designed to reflect the original procedure described in Chollet (2019) as closely as possible, in order to allow for human-machine comparisons on ARC. Participants could edit single grid cells one at a time based on the currently selected color, or edit multiple cells using a selection tool. They could also apply a flood fill operation which colored all neighboring cells of the same color to a different color. Additionally, participants could resize the height and width of the output grid to the desired size, as well as copy the test input to the test output grid.

Participants were allowed three attempts to generate the correct test output, and they were given binary feedback about whether their response was correct or incorrect after each submission attempt. Upon a correct submission or three incorrect submissions, participants moved directly onto the next task.

Additionally, participants were asked to write a description of the solution for transforming input grids to output grids for each task. They were asked to do this once before they received any feedback after their first submission (i.e., after they submitted their response but before they knew whether they were right or wrong). If their first attempt was correct they only submitted one written solution. If it was incorrect, they were asked to submit another written description after they submitted a subsequent correct attempt or their third incorrect attempt. Although the tasks were not timed, participants completed the entire experiment in 41 minutes on average (range 17.8 - 72.4).

Results

Since ARC is new and quite open-ended, our analyses were primarily exploratory in nature. Our approach was to analyze the rich dataset we collected in a bottom-up fashion, i.e., by using what we saw in the data to guide the analyses we conducted. Although we had some prior expectations about the relationship between language and ARC problem difficulty, many of our results were observational. The following sections focus on four main areas: performance, action sequences, natural language descriptions, and errors. Visualizations of all participant responses are available at `https://arc-visualizations.github.io`.[2]

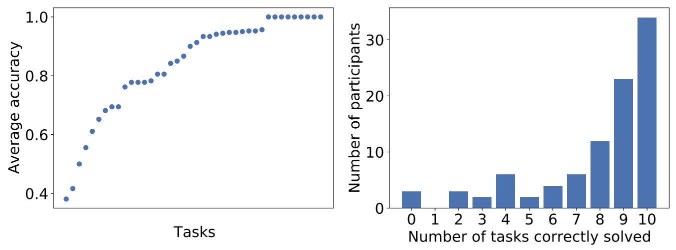

Figure 3: **Performance on ARC by task and participant**. The left figure shows the average accuracy across all 40 tasks, sorted by accuracy. The figure on the right shows how many tasks each participant solved correctly. Although there is considerable variability at both the task and participant level, performance was generally quite high.

We first analyzed overall performance, using accuracy defined as producing the correct test output for a task within three attempts. Overall, our results show that people performed well on the tasks we studied (*M* = 83.8% per task, *SD* = 16.7%). Figure 3 shows the distribution of accuracy across tasks, indicating considerable individual-level variation in difficulty. While most tasks were easily solvable by the majority of participants (65% of tasks have 80% or higher accuracy), only 38.1% of participants were able to generate the correct output in three attempts for the most challenging task. Qualitatively, tasks that relied on logic, rotations, or flips tended to be the more difficult, while tasks that involved simple color manipulations (such as color inversions or filling objects with colors) were easier. We also observed considerable variation in accuracy across participants as shown in Figure 3 (*M* = 8.38 of 10 tasks per subject, *SD* = *2.7*), however the modal frequency for tasks solved was 10 out of 10.

The average time to complete each task was 3 minutes and 6 seconds (*SD* = 2 minutes and 37 seconds), and the average time between first seeing the task and taking the first action was 36 seconds (*SD* = 1 minute and 7 seconds), suggesting that people were inferring rules in a short time frame. The average description length (across incorrect and correct submissions) provided by participants at the end of each task was 20 words (*SD* = 5 words). Participants took 1.59 attempts on average (*SD* = 0.46).

Although the Kaggle algorithms are not intended as candidates for human cognition, we nevertheless find it informative to note the differences between these algorithms and human performance in order to further elucidate the gap between ma-

[2]The website also contains state space graphs, participant errors, and language descriptions.

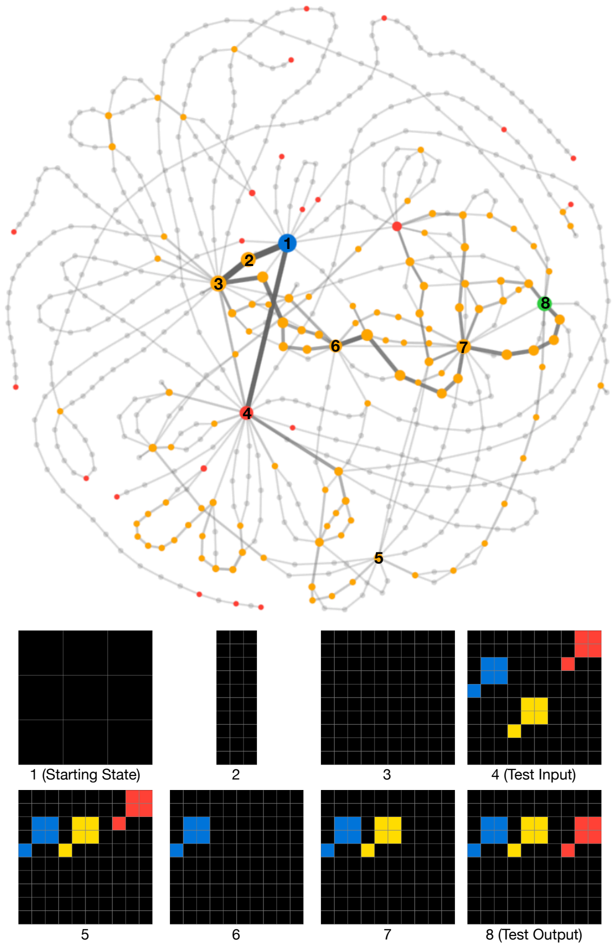

Figure 4: **State space graph for the box alignment task**. Each node represents an output state and each edge represents an action connecting two states. The colors of the various nodes depict the starting state (blue), the correct output state (green), incorrect submissions (red), and highly visited states (yellow). Larger nodes and thicker edges represent more frequent visits. Additionally, certain bottleneck states shown below the graph highlight that the sub-goals participants used to generate their responses were strongly object-based for this task, by either generating the correct output from scratch one object at a time, or copying the input first and moving the yellow and red objects to their correct location.

chine learning systems and human intelligence. The Kaggle algorithm achieved 57.5% accuracy on the 40 tasks we chose from the ARC training set (relative to 83.8% for humans). The Spearman rank correlation between human accuracy and the Kaggle algorithm accuracy was 0.35 ($p < 0.05$), although this was driven heavily by the algorithm failing to capture the hardest problems in the set while sometimes capturing the easiest ones.

In order to quantify which factors were linked to solving a given task, we fit a logistic mixed-effects model predicting success on each subject and task pairing using average description length per task as a fixed effect, with random intercepts for participants and tasks. We find a significant, negative effect of average description length on task accuracy (b = -0.17, 95% CI: [-0.315,-0.020], $p = 0.03$). This is in line

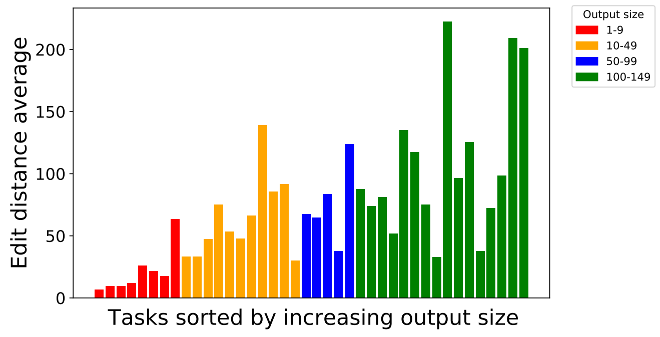

Figure 5: **Variability in action sequences across tasks**. This figure shows the variability in edit distances for all combinations of action sequences within a task, across all 40 tasks. The tasks are sorted according to the their true output grid size (length*width). Edit distance tended to be higher for tasks requiring more actions to solve, so we should expect to see overall variance increase with output size. However, within different output grid sizes there is still a range of variability, suggesting that divergence in action sequences is driven by more than output size.

with recent work (e.g. Lupyan, 2012; Lupyan & Zettersten, 2020) suggesting that longer description lengths are predictive of problem difficulty as longer descriptions suggests that the rule involves more complex transformations. Furthermore, we find a negative correlation (r(38) = -0.50, $p < 0.05$) between average description length and average accuracy per task.

Action Sequences As participants were required to generate their solutions to ARC tasks, we were able to collect a rich behavioral dataset of the sequences of actions participants performed to generate their responses. Figure 4 displays a state space graph constructed from the action sequences of all of the participants for the box alignment task (referenced in Fig 1) depicting the set of visited output states and transitions to other states based on their actions while generating responses. We found that the majority of participants' first actions are to either manually resize the output grid to the width and height of the intended solution (Figure 4, node 3), or to copy the test input grid (Figure 4, node 4). For this particular task, we can see that although there is substantial variability in the action sequences participants pursue within this task, participants converged on a few particular paths that pass through some common intermediate bottleneck states corresponding to drawing or moving objects. We qualitatively found that the bottleneck states are typically representative of task-relevant objects across other tasks too. Although further research is needed, we think this object-centric action planning reveals an important difference between human and machine solutions to ARC to date which will be further explored in the errors section.

We also examined the similarity of action sequences between participants across tasks, to determine the variability of how participants generated solutions to a particular task. To do so, we quantified the distance between two action sequences as the Levenshtein edit distance. Every state which

category	top unigrams
color	blue (396), color (353), red (244), colors (158)
size	size (58), 2x2 (33), 3x3 (33), 4x4 (21)
location	right (122), left (98), bottom (82), where (80)
relation	same (200), match (36), part (22), between (22)
object	squares (388), square (221), blocks (124)
geometric	line (136), lines (77), corner (52), diagonal (51)
number	one (139), number (114), 3 (59), two (54)
abstract	tetris (5), paint (4), vessel (2), flower (2)
transform	make (105), fill (87), extend (51), copy (49)

Table 1: **Top unigrams for each category**. Based on the natural language descriptions, we manually grouped words into 9 distinct content classes. The table above shows the top unigrams from each class along with their frequencies in parentheses.

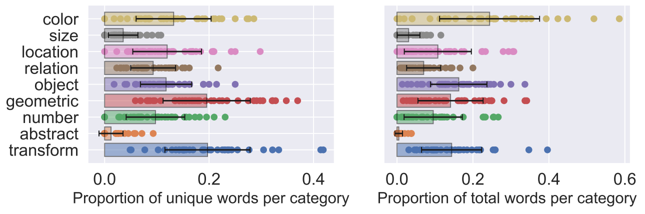

Figure 6: **Proportion of unique and total words per category**. The left figure shows the normalized proportion of unique words belonging to each category across all descriptions for each task. Geometric and transform words form the largest categories here, implying that people utilize a wider range of these concepts. The right figure shows the normalized proportion of total word counts for each category across all descriptions for all tasks. Here, we find that color words are utilized the most, indicating that people rely on these concepts more overall. Both figures exemplify the variability in concept use across tasks, both in the wide range of word use across tasks (overlaid data points) and the high variance (bars indicate standard deviation).

was visited by at least one participant was assigned a unique number; these numbers were then assembled into a sequence based on the intermediate states for each participant. We then computed the pairwise edit distance for every combination of participants within each task and computed the average. Overall, the average edit distance across all tasks (filtered by correct attempts) was 74 edits (*SD* = 53). Figure 5 shows the overall pattern of action sequence similarity across tasks which demonstrates considerable variability in how much participants actions overlapped per task.

Natural Language Descriptions In addition to the action sequences, we also analyzed the written natural language descriptions from the end of each task. The descriptions we collected were free form, although participants were encouraged to provide descriptions with enough detail so that they would be useful to future participants to reconstruct the correct test outputs.

We filtered the dataset to only include the final natural language descriptions participants provided, and to only include participants who correctly solved the task, as well as removing stop words.[3] We then categorized all of the words from the remaining set of descriptions into 9 distinct classes that captured the different kinds of concepts people used to describe the tasks (see Table 1 for a complete list of the categories, as well as the most frequent words used in each category). Figure 6 shows the proportion of each class across all correct descriptions. Color words were used the most overall as they are relevant in describing almost every task, with object and geometric words second most. Although abstract words are used the least, they provide some of the most interesting abstractions that people relied on to solve ARC tasks. For instance, in the **box alignment** task from Fig 1 a couple of participants referred to the pixel to the left of the boxes as a "tail", a creative mapping of an existing concept to this particular domain. In Figure 6, we show the proportion of unique words and total words used per category, demonstrating that category use was varied across tasks. Additionally, the large number of content words found across participants suggests that humans can deploy their existing rich conceptual knowledge and apply it to a new domain such as ARC with relatively little training, suggesting that language may act as a scaffold for task transfer.

[3]Participants who didn't solve a task generally used non-informative descriptions like "I have no idea", which was not meaningful for our analyses.

We were also interested in examining how consistent participant descriptions within a task were to one another. In order to measure consistency across descriptions, we use a recently developed measure by Lupyan and Zettersten (2020) known as naming divergence, which is calculated by (# of unique words / # of total words). Naming divergence scores lie between 0 and 1, where a lower naming divergence implies that the set of words used is consistent across participants, whereas a higher naming divergence implies that different participants describe the task differently. We computed naming divergence within tasks and compare this to a shuffled distribution. We generated this by sampling the average naming divergence of shuffled descriptions across tasks (i.e., the average naming divergence of 40 "tasks", with randomly sampled descriptions) 1000 times. The shuffled distribution has $M = 0.68$ and $SD = 0.003$, and the true average naming divergence of 0.41 (across tasks) was lower than any of these permutations ($p < 0.001$). This indicates that there was greater consistency in the language used to describe the same task.

Errors Because ARC required participants to generate their responses, the errors people make are informative about the kinds of representations people used to solve these tasks. One of the main aspects we were interested in was whether or not human errors were relevantly close to the correct answer in one or more dimensions. Although it was difficult to formulate measures to quantify this, it was qualitatively apparent that participants do in fact get most of the relevant features of problems right. For instance, Figure 7 shows how the errors participants made for the box alignment task were overall cor-

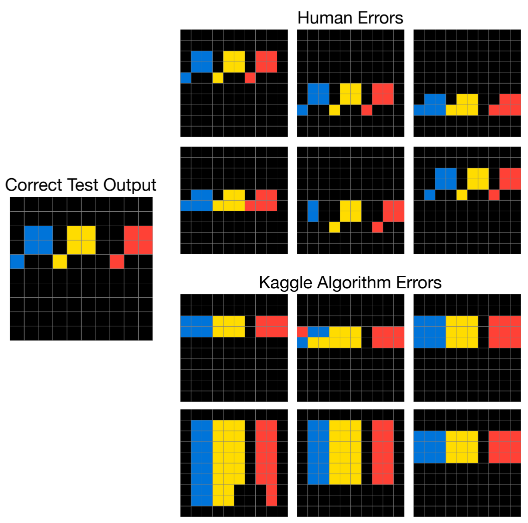

Figure 7: **Human and Kaggle algorithm errors for the box alignment task**. Here we display the set of most frequent human errors and the full set of Kaggle errors. For the Kaggle competition, each submission was allowed up to three guesses for each test input. Here, two separate submissions are displayed – each grid is a separate run of the algorithm, i.e., a new search over programs. The human errors obey object priors to a greater extent than the Kaggle errors do.

rect in inferring the right shapes, colors, and one dimension of alignment (along the y-axis). In contrast, although the colors are right for the Kaggle solutions, many of the shapes violate object-like priors from the input grid (e.g., the shapes are egregiously elongated, and one of the shapes appears to wrap around the grid). More examples of participant errors are available at `https://arc-visualizations.github.io`.

Discussion

We examined human performance on a subset of the Abstraction and Reasoning Corpus, a recently released machine learning benchmark that combines multiple challenging elements for existing systems such as flexible hypothesis generation, few-shot learning, compositionality and program induction. While the current best competitive machine learning systems perform poorly at the task, our results show that humans are very competent in the ARC domain. Given just a few input-output examples of a novel concept, the majority of participants were able to infer and apply the correct underlying transformation to create a corresponding test output in three or fewer attempts. Moreover, participants did not need expansive training within ARC, but were able to do so from the basis of a very limited amount of experience. The tasks in ARC vary widely in the types of prior knowledge they draw from (e.g. color, relations, objects, transformations, etc.), and yet participants easily recognized and applied the right kinds of knowledge in each task. Additionally, our dataset also contained the action sequences, natural language descriptions, and the errors made on each task. Analyses into each of these aspects revealed that there was considerable overlap in how people approached these tasks, and provided additional context and insight into the inductive biases and strategies used in solving ARC tasks.

Although language-of-thought (LOT) models seem well-matched to certain aspects of ARC, our findings may suggest that people's abilities are beyond current LOT models in some important ways. Standard LOT models use a hypothesis space with a fixed set of primitives defined in advance. Recent work has improved the ways in which these models can generate hypotheses, for example by grounding hypothesis generation based on observations (Bramley et al., 2018), or allowing for the LOT to be modified during the learning process (Rule et al., 2018; Ellis et al., 2020). However, a cognitive model of ARC may require a hypothesis space that is vast and semantically richer. Our analysis revealed a large set of geometric and transform words that participants chose from, as well as varied concepts like “square”, “wing”, and “tetris”, suggesting people can draw upon extensive background knowledge in addition to simpler primitives (Murphy & Medin, 1985). If so, this poses a challenge for existing LOT-based approaches, as it is not clear how the semantics of a large body of existing conceptual knowledge could be easily integrated into a LOT, nor how one could adapt that knowledge to specific tasks such as ARC.

One alternative account that is more parsimonious with our data—especially our findings regarding the relationship between description length and difficulty—is that hypothesis generation uses natural language as a scaffold for generating candidate hypotheses, either instead of, or in addition to a symbolic language of thought (Carruthers, 2002; Andreas, Klein, & Levine, 2017; Lupyan & Zettersten, 2020). This account is especially interesting in relation to abstract reasoning tasks like ARC. In standard LOT models, more complex and abstract concepts can be expressed through the composition of multiple primitives. However, generating complex, compositional hypotheses become increasingly less likely under the prior since the resulting programs are longer. However, natural language might facilitate this process by compressing compositional programs into a single function name or word for efficient communication (e.g., while you might explain a novel shape in many words by the number of sides, convexity, etc., one only needs to say “triangle” to convey an already established concept) (Regier, Kemp, & Kay, 2015). While non-linguistic primitives in a LOT model could be used to construct some of these concepts, it is unlikely to capture all of them. Thus, utilizing natural language to guide hypothesis generation in ARC, or as a means to augment LOT-based approaches, is a promising way to capture more flexible and human-like representational schemes.

Another difficulty for existing LOT-based models is through the lens of object perception. Standard LOT theories parse out the stimuli into symbolic representations that can be easily manipulated. Yet, the notion of what constitutes an object in ARC is flexible due to factors such as occlusion or whether to treat a set of grid cells as a single object or two, matching some of the challenges involved with real-world object perception. For instance, in the rightmost task in (Figure 1) it is not clear without context how one ought to parse out the objects – are the green lines in the input examples all part of one figure? Should the four colored pixels be considered a single unit or separately? It may be difficult to construct an object representation in advance that captures all of the possibilities. Humans, however, can navigate this ambiguity well and can flexibly parse the objects from ARC's input and output grids based on the task at hand.

Overall, this work is a first step at translating ARC (a machine learning challenge) into a compelling benchmark for program induction in humans as well. Future experimental work on ARC should investigate a wider range of tasks to validate the preliminary results reported here. In addition, once the basic variables are better understood other experimental designs could be overlaid on the task to manipulate aspects of human performance. One interesting observation from the behavioral data was that participants often take up to a minute between starting a new task and performing their first action to generate the output grid, suggesting that a lot of time might be spent thinking about and formulating hypotheses. Another extension of this work would be to develop a computational account for solving ARC tasks that incorporates some of the insights gleaned here about how humans solve these tasks. Our results further suggest that in addition to the set of core knowledge priors (e.g., objects, relations, and counting) described by Chollet (2019), part of the speed and flexibility may also come from incorporating existing conceptual knowledge into the program induction process. Finally, although ARC was designed to push the boundaries of machine intelligence, many of the critiques are also relevant to cognitive science: rich and challenging benchmarks are needed to fully understand and test the limits of compositional generalization and abstract reasoning capability in both humans and machines.

Acknowledgements

Thanks to Francois Chollet for creating ARC, and for his comments on this work. Additional thanks to Laura Ruis and Yanli Zhou for helpful feedback on the draft. Aysja Johnson and Todd M. Gureckis were supported by the John S. McDonnell Foundation Scholar Award.

References

Andreas, J., Klein, D., & Levine, S. (2017). Learning with latent language. *arXiv preprint arXiv:1711.00482*.

Bellemare, M. G., Naddaf, Y., Veness, J., & Bowling, M. (2013). The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, *47*, 253–279.

Bramley, N., Rothe, A., Tenenbaum, J., Xu, F., & Gureckis, T. (2018). Grounding compositional hypothesis generation in specific instances. In *Proceedings of the 40th annual conference of the cognitive science society.*

Carruthers, P. (2002). The cognitive functions of language. *Behavioral and Brain Sciences*.

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

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *2009 ieee conference on computer vision and pattern recognition* (pp. 248–255).

Ellis, K., Wong, C., Nye, M., Sable-Meyer, M., Cary, L., Morales, L., … Tenenbaum, J. B. (2020). Dreamcoder: Growing generalizable, interpretable knowledge with wake-sleep bayesian program learning. *arXiv preprint arXiv:2006.08381*.

Goodman, N. D., Tenenbaum, J. B., Feldman, J., & Griffiths, T. L. (2008). A rational analysis of rule-based concept learning. *Cognitive science*, *32*(1), 108–154.

Goodman, N. D., Tenenbaum, J. B., & Gerstenberg, T. (2015). Concepts in a probabilistic language of thought. In E. Margolis & S. Laurence (Eds.), *The conceptual mind: New directions in the study of concepts* (pp. 623–653). Cambridge, MA: MIT Press.

Gureckis, T. M., Martin, J., McDonnell, J., Rich, A. S., Markant, D., Coenen, A., … Chan, P. (2016). psiturk: An open-source framework for conducting replicable behavioral experiments online. *Behavior research methods*, *48*(3), 829–842.

Hernandez-Orallo, J., Martinez-Plumed, F., Schmid, U., Siebers, M., & Dowe, D. L. (2016). Computer models solving intelligence test problems: Progress and implications. *Artificial Intelligence*, *230*, 74-107.

Hofstadter, D. R., & Mitchell, M. (1988). Conceptual slippage and analogy-making: A report on the copy-cat project. In *Proceedings of the tenth annual conference of the cognitive science society.*

Hofstadter, D. R., et al. (1979). *Gödel, escher, bach: an eternal golden braid* (Vol. 13). Basic books New York.

Lake, B. M., Linzen, T., & Baroni, M. (2019). Human few-shot learning of compositional instructions. *Proceedings of the 41st Annual Conference of the Cognitive Science Society*.

Lupyan, G. (2012). Linguistically modulated perception and cognition: the label-feedback hypothesis. *Frontiers in Psychology*, *3*, 54.

Lupyan, G., & Zettersten, M. (2020). Does vocabulary help structure the mind?

Mitchell, M. (2021). Abstraction and analogy-making in artificial intelligence. *arXiv preprint arXiv:2102.10717*.

Murphy, G. L., & Medin, D. L. (1985). The role of theories in conceptual coherence. *Psychological review*, *92*(3), 289.

Peirce, C. S. (1935). Collected papers. scientific metaphysics (ed. by c. hartshorne and p. weiss.), vol. vi.

Piantadosi, S. T. (2011). *Learning and the language of thought* (Unpublished doctoral dissertation). Massachusetts Institute of Technology.

Piantadosi, S. T., & Jacobs, R. A. (2016). Four problems solved by the probabilistic language of thought. *Current Directions in Psychological Science*, *25*(1), 54–59.

Piantadosi, S. T., Tenenbaum, J. B., & Goodman, N. D. (2016). The logical primitives of thought: Empirical foundations for compositional cognitive models. *Psychological review*, *123*(4), 392.

Regier, T., Kemp, C., & Kay, P. (2015). Word meanings across languages support efficient communication. *The handbook of language emergence*, 237–263.

Rule, J., Schulz, E., Piantadosi, S. T., & Tenenbaum, J. B. (2018). Learning list concepts through program induction. *BioRxiv*, 321505.