Lab42

Essay Challenge

ARC Solution Concept

Outline of a Detailed Approach to Solving the Abstraction and Reasoning Corpus

The Hitchhiker's Guide to the ARC Challenge

Executive Summary

Simon Ouellette

27. May 2023

Solving the ARC challenge is fundamentally about algorithm learning, rather than "curve fitting". The literature presents at least 2 promising machine learning approaches that are capable of learning algorithms in a multi-task context: Universal Transformers[4, 6] and DreamCoder[5].

While the latter uses multi-task learning by default, the former can do it via "in-context learning". This refers to structuring the input sequence such that it contains a concatenated vector description of the input-output grids. The output sequence will start with a description of the input test grid. The model will then be expected to fill in the output grid by predicting the next tokens in the output sequence.

In the case of the ARC challenge, a data simulation framework will have to be built (there is not enough pre-existing training data). It's not just about the quantity of data examples, but also how they are presented to the learner.

Highly composite tasks, i.e. tasks that are composed of two or more non-linear tasks, can be difficult, even impossible to learn directly[2, 3, 8, 7]. Learning them is, in a sense, like searching for a needle in a haystack. However, if we decompose them into sub-tasks and learn these separately, it becomes possible to learn the full task. This general idea is known as "intermediate supervision", and can take many forms, such as the use of supervised hints or curriculum learning. The solution proposed here will emphasize the curriculum learning approach.

In summary, here are the requirements for a candidate solution:

1. It must be Turing-complete: it must be able to learn an algorithm (rather than only a mapping), that involves a dynamic, arbitrary number of loops.

2. It must be able to multitask, and operate in a meta-learning setting, such as via in-context learning.

3. It must use a form of intermediate supervision, such as curriculum learning.

4. Data must be generated/simulated, since there is not enough data in the ARC training set.

5. It must be able to scale to complex visual reasoning problems.

6. It will require a lot of computing resources, possibly at the same level as LLMs.

The curriculum learning approach used for training will follow the "combined strategy"[1]: every training sample is either drawn (randomly) from the "naive" strategy or the "mixed" strategy. The "naive" strategy refers to first training on a dataset of minimal complexity/scale/difficulty, until convergence. Then, the complexity, scale or difficulty is increased by a small amount, and model training is resumed on that new dataset. In the "mixed" strategy, instead of starting with only a small level of complexity, and increasing gradually, training samples are drawn from random levels of complexity. As a result, a training

batch may consist of small, intermediate examples as well as full-blown task examples.

While developing the data generator, each task must be associated with a difficulty level. This level will be used to determine the order in which the different tasks are presented to the learner.

The underlying assumption is that there are core "skills", like building blocks of intelligence, that have been perfected via evolution (as well as learned in infancy) to solve the reasoning problems encountered in human life. These core skills are finite in quantity, and composable in order to allow efficient adaptation to novel instances of problems. The training regime for the ARC solution should not only focus on solving complete ARC tasks. More basic exercises, corresponding to core skills and simpler sub-tasks, must also be included.

The proposed "core skills" to focus on when generating training tasks are:

1. **Cardinality**: the ability to count, to perform cardinality related comparisons such as equal, greater than, less than, maximum, minimum. The ability to perform arithmetic operations such as addition, subtraction, division, multiplication, modulo. Notions of odd and even (parity).

2. **Objectness**: the ability to "detect" objects, and separate them from the background (in a way that is roughly equivalent to how humans would do it). The notion of collision between objects. Distinguishing different objects.

3. **Geometry**: the notion of "Platonic" shapes. Pure straight lines, triangles, squares, rectangles, circles.

4. **Relations**: this is the most abstract group of core skills, and involves (visual) relational reasoning such as same vs different, bigger than and smaller than, "odd one out", is part of (is a component of), is a type of, is inside of, etc.

5. **Positioning**: the notion of relative positions - to the right of, to the left of, below, above, top-right, etc. Also involves more absolute positions such as the top-left quadrant of the grid, the left half, the right half, the bottom-right quadrant of the grid, etc.

6. **Transformations**: geometric transformations such as scaling, clipping, translation, rotation, horizontal and vertical flipping. Also includes color-based transformation, i.e. changing color and filling in shapes with a certain color.

7. **Symmetries**: this is the notion in the geometric sense, i.e. the idea that objects or grids can be composed of reducible components applied as a "mirror image".

8. **Patterns**: the ability to detect repeating, predictable groups of pixels or objects.

9. **Repetition**: the ability to iterate and repeat patterns while generating the output grid. This is one of the core knowledge groups that most directly expresses the need for a learner that is able to execute dynamic loops.

What follows is a roadmap for developing and validating the proposed solution:

Step 1: develop a reasonable number of tasks (100?) for a particular core skill group, such as cardinality.

Step 2: train one (or both, for comparison) of the algorithms that fit the aforementioned requirements on the task generator developed in Step 1. This training must use the aforementioned curriculum learning regime. Additionally, it must be done such that there is a held out subset of tasks, to be used only to evaluate the generalization ability of the trained models.

Step 3: evaluate performance on the held-out set. Here, two results might occur: either the performance is near perfect, or the performance has degraded relative to the training set. If it is the former, we've validated that the solution can learn algorithims in a way that generalizes out-of-distribution. If it is the latter, the data generator should be reviewed. Are there intermediate concepts that are required but are not seen in the training set? Are there not enough distinct training tasks to learn in a way that generalizes? Could the assessement of curriculum levels be faulty? Does it need more granularity?

Step 4: test on the ARC training set whether the trained model is able to solve a non-zero amount of tasks.

Step 5: implement a new group of skills and re-train the model on the new full dataset (including the new core group). The objectness core skill group is probably a good choice at this stage.

Step 6: re-test on the same ARC training set: did the number of successfully solved tasks go up? Estimate the number of new ARC tasks that are successfully solved per new distinct task implemented in the data generator. If this ratio is significantly greater than 1, the approach has been essentially validated.

Step 7: finish developing the data generator (that is, implementing the core skill groups that have been previously enumerated)

Step 8: increase the maximum grid size for the generated problems. Implement the necessary modifications to the Universal Transformer to support long sequences.

Step 9: using large-scale deep learning computing resources, train variants of the selected models with an increased hyperparameter scale. Note the test performance on the ARC validation set after each training phase. Repeat until convergence.

Step 10: if the performance plateau reached by scaling up the model is unsatisfactory, the first step is to troubleshoot which types of ARC tasks it fails on, and ask: is there a core skill that is missing in the training data?

In this abridged edition, the focus was on describing the proposed solution, rather than motivating it or anticipating criticisms. The skeptical reader is

encouraged to read the full version in order to understand the architectural choices behind the proposal.

References

[1] Wojciech Zaremba and Ilya Sutskever. "Learning to execute". In: *arXiv preprint arXiv:1410.4615* (2014).

[2] Çağlar Gülçehre and Yoshua Bengio. "Knowledge matters: Importance of prior information for optimization". In: *The Journal of Machine Learning Research* 17.1 (2016), pp. 226–257.

[3] Shai Shalev-Shwartz, Ohad Shamir, and Shaked Shammah. "Failures of gradient-based deep learning". In: *International Conference on Machine Learning*. PMLR. 2017, pp. 3067–3075.

[4] Mostafa Dehghani et al. "Universal Transformers". In: *Proceedings of International Conference on Learning Representations*. 2019. URL: https://openreview.net/forum?id=HyzdRiR9Y7.

[5] Kevin Ellis et al. "DreamCoder: Bootstrapping Inductive Program Synthesis with Wake-Sleep Library Learning". In: *Proceedings of the 42nd ACM SIGPLAN International Conference on Programming Language Design and Implementation*. PLDI 2021. Virtual, Canada: Association for Computing Machinery, 2021, pp. 835–850. ISBN: 9781450383912. DOI: 10.1145/3453483.3454080. URL: https://doi.org/10.1145/3453483.3454080.

[6] Samuel Cognolato and Alberto Testolin. "Transformers discover an elementary calculation system exploiting local attention and grid-like problem representation". In: *2022 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2022, pp. 1–8.

[7] Elisabetta Cornacchia and Elchanan Mossel. "A Mathematical Model for Curriculum Learning". In: *arXiv preprint arXiv:2301.13833* (2023).

[8] Noam Wies, Yoav Levine, and Amnon Shashua. "Sub-Task Decomposition Enables Learning in Sequence to Sequence Tasks". In: *Proceedings of 11th International Conference on Learning Representations (ICLR)*. 2023. URL: https://openreview.net/pdf?id=BrJATVZDWEH.