OGRE: An Object-based Generalization for Reasoning Environment

Kelsey R. Allen^{1,2,*}, Anton Bakhtin^{3,*}, Kevin Smith^{1,2}, Josh Tenenbaum^{1,2,4}, Laurens van der Maaten^{3}
* These authors contributed equally
^{1} MIT BCS, ^{2} Center for Brains, Minds and Machines, ^{3} Facebook AI Research, ^{4} MIT CSAIL

Abstract

If an agent understands how to reason about some objects, can it generalize this understanding to new objects that it has never seen before? We propose the Object-based Generalization for Reasoning Environment (OGRE) for testing object generalization in the context of *active physical reasoning*. OGRE emphasizes evaluating agents by how efficiently they solve novel physical reasoning tasks, not just how well they can predict the future. OGRE provides two levels of generalization: generalization over reasoning strategies with familiar objects, and generalization over new object types that still share similar material properties to those in training. We run three baseline agents on OGRE, showing that an image-based Deep Q-Network [6] can learn reasoning strategies that generalize in a limited way across familiar object types, but does not generalize at all to new object types. We hope OGRE will encourage advances in building object representations that more explicitly enable generalizable reasoning and planning compared to previous benchmarks. Code to use OGRE can be found at `https://github.com/facebookresearch/phyre/blob/master/agents/OGRE.md`.

1 Introduction

In recent years, AI systems have become increasingly adept at perceiving and predicting objects' dynamics, supported by the development of datasets [5, 8] that explicitly test these abilities. However, it is not simply passive prediction accuracy that allows humans and other animals to survive and adapt – it is how those mechanisms support active behavior that can be generalized to new situations. For this reason, physical problem-solving paradigms have been widely used as explicit tests of intelligence throughout the animal kingdom [7, 4, 9, 3].

Inspired by this observation, the PHYRE and Virtual Tools benchmarks were developed to provide similarly difficult physical problem-solving tests for artificial and biological agents. Virtual Tools [1] found that humans are adept and efficient in their physical problem-solving, and that this is supported by object-oriented, model-based planning. In contrast, PHYRE [2] showed that even well-trained deep reinforcement learning agents fail abysmally at these kinds of puzzles. Both PHYRE and Virtual Tools emphasize evaluating agents by the efficiency of their *actions* rather than their ability to make passive predictions about how the world will unfold.

OGRE extends beyond PHYRE and Virtual Tools to test **object generalization**: the ability to both generalize physical reasoning to new puzzles, as well as to novel objects. All of the tasks in PHYRE – including both training and testing – were designed using a small number of object types, and so the benchmark does not test whether agents that can solve its task have object representations that extend to other domains. Virtual Tools was designed as a test suite with training left to model designers, and so cannot assess how well objects generalize between training and testing. In contrast, OGRE explicitly separates training and testing domains so that while they have identical action spaces and dynamics, there are objects in the test suite that are not observed in training, but whose properties can be determined from their appearance in the same way as in training. Thus, to solve OGRE, agents

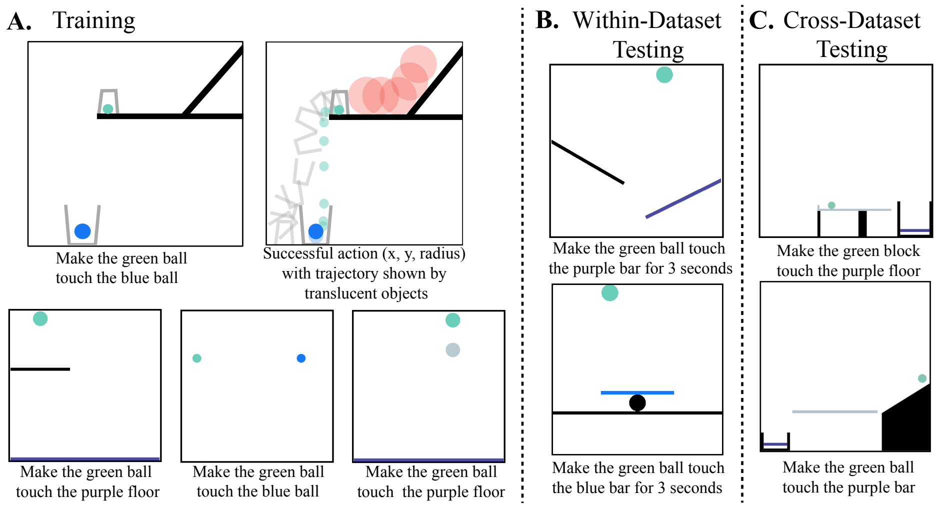
A. Training
Make the green ball touch the blue ball
Successful action (x, y, radius) with trajectory shown by translucent objects
Make the green ball touch the purple floor
Make the green ball touch the blue ball
Make the green ball touch the purple floor
B. Within-Dataset Testing
Make the green ball touch the purple bar for 3 seconds
Make the green ball touch the blue bar for 3 seconds
C. Cross-Dataset Testing
Make the green block touch the purple floor
Make the green ball touch the purple bar

Figure 1: **A** Top: an example of a level within the training set of OGRE. Black and purple objects are static; objects with any other color are dynamic and subject to gravity. Actions are single balls at a position (x, y) with radius r, depicted as a red ball which falls under gravity once placed. Agents can observe the outcomes of these actions for a large set of training levels. Bottom: other example levels that might be included in training. **B**: within-dataset testing includes levels that use the same object types, but require different kinds of strategies to succeed. **C**: cross-dataset testing includes a set of levels inspired by the Virtual Tools environment, which contains different object shapes.

must have the ability to extract abstract object properties from a scene, and use those representations to reason creatively about how to solve a novel task.

2 OGRE: Object-based Generalization for Reasoning

Example puzzles from OGRE are shown in Figure 1. Each puzzle has a goal state, which in OGRE is always to make a green object touch a blue or purple object, and an initial state in which the goal is not satisfied. A puzzle can be solved by placing one ball in the environment such that when the physical simulation is run the goal is satisfied. Thus, like PHYRE and Virtual Tools, OGRE requires agents have the ability to solve a diversity of tasks, and perform long-horizon causal reasoning to select actions likely to solve the puzzle. Unlike in PHYRE and Virtual Tools, OGRE additionally requires object generalization: the agent needs to re-use and adapt the knowledge it obtained by interacting with one object type in training to interacting with new object types during testing. In this section, we will explain how the environment works and the proposed training/test generalization suite of OGRE.

2.1 Game mechanisms

Tasks in OGRE, as in PHYRE and Virtual Tools, are represented as initial static scenes composed of objects of different types (see Figure 1). All objects are non-deformable and are either *static* or *dynamic*, distinguished by color. Static objects remain at a fixed position and do not move, while dynamic objects can move due to gravity or collisions with other objects. Objects can be of arbitrary shape, including convex polygons, containers, line segments, or other compound shapes (see Figure 1). The goal for the scene is always to ensure that a green object touches a blue or purple object for at least 3 seconds.

Unlike in many other environments, an agent in OGRE must achieve the goal by taking a *single action*. The action is parameterized as a tuple (x, y, r); the location in x and y for a dynamic ball of radius r. When the ball is placed, it must not extend beyond the world boundaries or intersect other objects in the scene. If an action violates these requirements, it is *invalid* and cannot be simulated.

Once a ball is placed, the simulator runs until the goal is achieved or until 1000 frames elapse. If the agent was unsuccessful, the world is reset to its initial state and the agent can try again. After acting, agents receive a binary *reward* indicating whether the goal was achieved, as well as a series of *observation images* rasterizing the world to a 256×256 grid. Each grid cell takes one of seven values specifying whether that location is a (1) dynamic goal object, (2) static goal subject, (3) dynamic goal subject, (4) static non-goal body, (5) dynamic non-goal body, (6) body placed by the agent, or (7) background. Figure 1 shows some sample OGRE tasks with goals written in natural language solely for the convenience of the reader.

2.2 Generalization Tiers

OGRE posits that object representations should be useful for *sample-efficient acting*, and therefore focuses on evaluations that measure an agent's ability to generalize object knowledge by those metrics. To test this, we provide two types of variation in OGRE – (a) variation in terms of the tasks that need to be solved between training and testing but keeping object types consistent (**within-dataset**), and (b) variation in the representation of objects between training and testing (**cross-dataset**).

The two datasets are taken from the PHYRE [2] and Virtual Tools game [1] levels respectively. Each dataset consists of task *template*s (25 for PHYRE, 12 for Virtual Tools) which define a set of 100 related tasks with a common goal but different initial states. Individual levels in a template are generated by varying task template parameters (such as positions of initial objects, sizes of initial objects, and shapes of initial objects).

In the **within-dataset** generalization condition, the templates **within** the PHYRE dataset are split into *training*, *evaluation* and *test* tasks. Evaluation split is expected to be used for all hyperparameter tuning and preliminary experiments, while test split is expected to be used to report only the final numbers. Crucially, there is no overlap between tasks in each condition, which forces agents to generalize their object-oriented reasoning capabilities between train and test.

In the **cross-dataset** generalization condition, training templates are taken exclusively from PHYRE, while test templates are taken exclusively from Virtual Tools. This generalization condition is much harder, but still within reach for current AI and machine learning approaches. Both datasets use different object types to construct their puzzles (e.g., there are no filled non-rectangular polygons in PHYRE but there are in Virtual Tools) but both datasets use the same dynamics engine, goal definitions, and object rendering process (e.g., black objects are always static, grey objects are always dynamic non-goal objects, etc.). Thus an agent can learn general world and object properties that transfer between training and testing, but will be unable to transfer learning about particular object types.

In the **training phase**, the agent has access to the training tasks and unlimited access to the simulator. The agent does not have access to task solutions, but can use the simulator to train models that can solve tasks. Such models may include forward-prediction or action-prediction models.

In the **testing phase**, the agent receives test tasks that it needs to solve in as few *attempts* (queries to the simulator) as possible. After each attempt, the agent receives a binary reward and observations of intermediate world states either as a video or in a symbolic form. The agent can use this information to refine its action for the next attempt.

Measuring Performance Agents are judged by the efficiency of their actions in the testing phase. We define efficiency by the number of actions tried before solving a given task – fewer attempts means greater efficiency. As in PHYRE and Tools, this can be formalized by recording the cumulative percentage of test tasks that were solved as a function of the number of attempts k taken by a model. From this curve, we compute the **AUCCESS**, or *area under the success-percentage curve*, that aggregates the success percentages s_k in the curve via a weighted average. To encourage more sample-efficient agents, we take a log-scaling of the attempts to calculate the **AUCCESS**. The weights are therefore given as $w_k = \log(k+1) - \log(k)$, yielding AUCCESS $= \sum_k w_k \cdot s_k / \sum_k w_k$.

3 Analysis

We conduct experiments to obtain baseline results for within-dataset and cross-dataset generalization on OGRE. Code reproducing the results of our experiments is available from https://github.

com/facebookresearch/phyre/blob/ogre/agents/OGRE.md. We test the following agents on the within-dataset and cross-dataset generalization settings:

Random agent (RAND). This agent simply samples valid actions uniformly at random from the 3D action space at test time.

Object-Oriented Random Agent (OORAND). This agent samples actions whose position in x overlaps any dynamic object in the scene, thus selecting from actions that are likely to have an impact on the scene. The radius is sampled uniformly at random.

Deep Q-network (DQN). The DQN agent is taken directly from [2]. It is trained to predict reward on a balanced set of successful and unsuccessful observation-action-reward triplets collected from unrolling the simulator. Note that this balanced training is a serious limitation as it solves the exploration problem for many levels. We did not tune parameters for the cross-dataset setting and use parameters for within-dataset setting from the original paper.

Results We show the results of running each of the above agents on both the **within-dataset** setting, and the **cross-dataset** setting in Table 1. For now, we only evaluate on PHYRE->PHYRE (within-dataset) and PHYRE->Virtual Tools (cross-dataset). For the within-dataset condition, the DQN model outperforms both the random and object-oriented random agent, suggesting that it is learning something more general than simply tools near dynamic objects in the scene. However, for the cross-dataset condition, the DQN no longer outperforms the simple object-oriented random agent. This suggests that it has **not** learned a robust representation of objects that generalizes across datasets. In both the within-dataset or cross-dataset conditions, no model is performing at human level. In the related Virtual Tools environment, humans have a success percentage of approximately 75% at $k = 10$ attempts. For more agent results, please see appendix A.

	Within-Dataset	**Cross-Dataset**
RAND	13.0$\pm$5.0	17.6$\pm$2.3
OORAND	25.3$\pm$9.0	27.5$\pm$4.3
DQN	36.8$\pm$9.7	26.5$\pm$6.9

(a) Area under the success-percentage curve (AUCCESS) of three agents. Higher is better.

	Within-Dataset	**Cross-Dataset**
RAND	6.8$\pm$5.0	9.7$\pm$3.1
OORAND	24.5$\pm$11.4	25.3$\pm$5.9
DQN	34.5$\pm$10.2	23.8$\pm$7.6

(b) Success percentage at $k = 10$ attempts of three agents. Higher is better.

Table 1: Comparison of the three agents on **within-dataset** and **cross-dataset** generalization. Mean and standard deviation on the 10 folds are reported.

4 Discussion and Future Work

OGRE aims to encourage the development of object representations that directly improve the reasoning and problem-solving capabilities of artificial agents, mirroring one of the most quintessentially human traits: tool use. We have shown that both **within-dataset** and **cross-dataset** generalization are challenging problems for current agents, and we expect significant improvements will require more object-oriented, model-based approaches.

OGRE benefits from being based on two existing, independently developed environments. Virtual Tools [1] showed that humans are capable of solving these 2-dimensional games in just a handful of attempts, and that only an object-oriented, model-based approach could achieve similar efficiency. PHYRE [2] tested a variety of different machine learning architectures, showing that there is substantial room for improvement. This underscores the appeal of OGRE as a benchmark for object-oriented representations and generalization – Allen et al. [1] provides strong evidence that humans require objects and dynamics models to play these games, while the coding environment of PHYRE [2] is fast, well-engineered, and robust for rapid development of agents.

Future work can extend the testing generalization. Alternative object-oriented generalization, like generalization over the action space (using two objects instead of one, or picking a shape from a set of options), and generalization over object appearance (changing colors or textures during testing) are easy to implement in the OGRE environment.

References

[1] K. Allen, K. Smith, and J. Tenenbaum. The tools challenge: Rapid trial-and-error learning in physical problem solving. In *arXiv 1907.09620*, 2019.

[2] A. Bakhtin, L. van der Maaten, J. Johnson, L. Gustafson, and R. Girshick. Phyre: A new benchmark for physical reasoning. In *Advances in Neural Information Processing Systems*, pages 5082–5093, 2019.

[3] S. R. Beck, I. A. Apperly, J. Chappell, C. Guthrie, and N. Cutting. Making tools isn't child's play. *Cognition*, 119(2):301–306, 2011.

[4] S. A. Jelbert, A. H. Taylor, L. G. Cheke, N. S. Clayton, and R. D. Gray. Using the aesop's fable paradigm to investigate causal understanding of water displacement by new caledonian crows. *PloS one*, 9(3):e92895, 2014.

[5] A. Lerer, S. Gross, and R. Fergus. Learning physical intuition of block towers by example. In *ICML*, 2016.

[6] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

[7] M. Osvath and H. Osvath. Chimpanzee (pan troglodytes) and orangutan (pongo abelii) forethought: self-control and pre-experience in the face of future tool use. *Animal cognition*, 11(4): 661–674, 2008.

[8] R. Riochet, M. Y. Castro, M. Bernard, A. Lerer, R. Fergus, V. Izard, and E. Dupoux. Intphys: A framework and benchmark for visual intuitive physics reasoning. In *arXiv 1803.07616*, 2018.

[9] R. W. Shumaker, K. R. Walkup, and B. B. Beck. *Animal tool behavior: the use and manufacture of tools by animals*. JHU Press, 2011.

A Additional analyses

We compare complexity of the tasks in the original PHYRE tier with the tasks in the Virtual Tools tier Fig. 2. The test tasks in Cross-Dataset settings are easier than ones in the Within-Dataset settings that indicates that the failure of DQN generalization is not caused by increased complexity of the tasks.

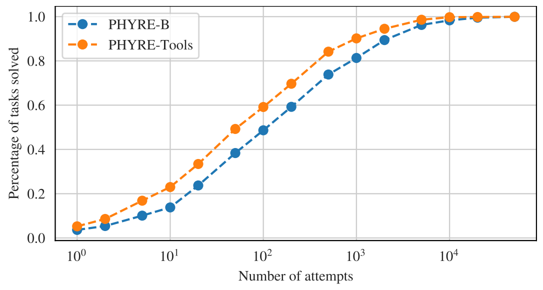

Figure 2: Percentage of tasks solved by a *random* agent (y-axis) as a function of the number of attempts (x-axis; log scale) for both original PHYRE-B and the new PHYRE-Tools tiers. Values are averaged over 10 runs over all tasks in the tier; error bars indicate one standard deviation. PHYRE-Tools are slightly easier for solve using a random search.

We report result for all baseline agents from the PHYRE dataset [2] at table 2. We use the same models and hyperparameters that were used from cross-template testing in the original paper. For convenience we reproduce descriptions of the agents below.

Random agent (RAND). This agent does not perform any training and instead samples actions uniformly at random from the 3D (depending on the tier) action space at test time.

Non-parametric agent (MEM). At training time, this agent generates a set of R random actions and uses the simulator to check if each of these actions can solve each of the training tasks. For each action a, the agent computes p_a: the fraction of training tasks that the action solves. The agent then sorts the R actions by p_a (highest to lowest), and tries them in this order at test time. This agent is non-parametric because it uses a list of "memorized" actions at test time.

In the *within-dataset* and *cross-dataset* settings, the test tasks come from previously unseen task templates and this simple agent cannot relate them to tasks seen during training. It therefore uses the same action ranking for all tasks and ignores the observation of the initial state.

Non-parametric agent with online learning (MEM-O). This agent has the same training phase as the non-parametric agent, but continues to learn online at test time. Specifically, after finishing each test task (either successfully or unsuccessfully), the agent updates p_a based on the reward received for each action a in the subset of the actions it attempted. The updated ranking is used when the next task is attempted. Such online updates are beneficial, in particular, in the within-dataset setting because they allow the agent to learn something about the tasks in the previously unseen templates.

Object-Oriented Random Agent (OORAND). This agent samples actions whose position in x overlaps any dynamic object in the scene, thus selecting from actions that are likely to have an impact on the scene. The radius is sampled uniformly at random.

Deep Q-network (DQN). As before, the DQN agent collects a set of observation-action-reward triplets by randomly sampling actions and running them through the simulator. The agent trains a deep network on the resulting data to predict the reward for an observation-action pair.

Deep Q-network with online learning (DQN-O). Akin to MEM-O, this agent uses rewards from test tasks to perform online updates. After finishing a test task, the agent performs a number of gradient descent updates using examples obtained from that task. The updated model is then used for the next test task.

Note that the DQN-online agent performs remarkably well in the cross-dataset condition. We hypothesize that this is because levels within a template require very similar actions to succeed within the Virtual Tools tier. DQN-Online is therefore less appropriate for evaluating cross-dataset transfer, as improvements can be mostly due to learning what general class of actions works well for a particular template.

	Within-Dataset	**Cross-Dataset**
RAND	13.0$\pm$5.0	17.6$\pm$2.3
MEM	18.5$\pm$5.1	24.5$\pm$4.9
MEM-O	22.8$\pm$5.0	29.0$\pm$4.5
OORAND	25.3$\pm$9.0	27.5$\pm$4.3
DQN	36.8$\pm$9.7	26.5$\pm$6.9
DQN-O	56.2$\pm$10.5	48.6$\pm$7.1

(a) Area under the success-percentage curve (AUCCESS) of three agents. Higher is better.

	Within-Dataset	**Cross-Dataset**
RAND	6.8$\pm$5.0	9.7$\pm$3.1
MEM	15.2$\pm$5.9	23.9$\pm$5.8
MEM-O	20.1$\pm$5.6	27.4$\pm$5.1
OORAND	24.5$\pm$11.4	25.3$\pm$5.9
DQN	34.5$\pm$10.2	23.8$\pm$7.6
DQN-O	58.2$\pm$10.9	51.1$\pm$7.8

(b) Success percentage at $k = 10$ attempts of three agents. Higher is better.

Table 2: Comparison of the three agents on **within-dataset** and **cross-dataset** generalization. Mean and standard deviation on the 10 folds are reported.