Investigating Flexible Role Binding in AI Agents

Brian Pennisi (bp2221@nyu.edu)
Center for Data Science, New York University
Bloomberg

Rheza Budiono (rheza@nyu.edu)
Department of Psychology, New York University

Todd M. Gureckis (todd.gureckis@nyu.edu)
Department of Psychology, New York University

Mark K. Ho (mkh260@nyu.edu)
Center for Data Science, New York University

Abstract

Humans can flexibly bind familiar functional roles to novel entities in their environment. For example, children who have the concept of "goal posts" can bind this abstract role to two hats placed on the street. In doing so, they can port over existing expectations of "goal posts" for the duration of the game. In this paper, we seek to explore artificial agents' ability to perform flexible role binding and rebinding. To this end, we designed a Gridworld navigation game and tested a popular CNN-based agent which has had success in other tasks involving visual and spatial state spaces (e.g. Atari or Minigrid). To our surprise, we found that while this architecture was capable of overfitting to the training set, it was not able to learn flexible role binding without intervention. We ultimately show that with carefully engineered data augmentation techniques, our artificial agent is able to learn the task. This suggests that the diversity of the training dataset was a limiting factor.

Keywords: role rebinding; zero shot generalization in reinforcement learning; reinforcement learning; vision reinforcement learning; overfitting; supervised learning; data augmentation;

Introduction

Humans have the ability to flexibly bind familiar functional roles to new entities in their environment. Consider the example of children who have the concept of "goal posts." They can bind the abstract role of "goal posts" to two hats placed on the street. With this, they port over their existing expectations of "goal posts" while playing the game.

Such abilities have often been described in terms of analogical reasoning within cognitive science literature, such as Gentner, Holyoak, and Kokinov (2001).

In this paper, we look to understand whether artificial agents can learn to perform flexible role binding and rebinding. To accomplish this, we designed a Gridworld-like navigation game (Chevalier-Boisvert et al., 2023). The game acts as a test of whether artificial agents are capable of flexible role binding. Our game takes heavy inspiration from the popular video game "Baba is You."[1] In our game, the player controls an agent and must navigate to a goal. Crucially, the agent and goal can take the form of different objects in each trial. For example, the agent can be a sheep and the goal can be a flag (or vice versa). The agent can use an arrangement of certain "word blocks" in the game to identify the correct role binding – for example, the user can read the correct role binding "flag is win" from the game screen (Figure 1). Despite this task requiring flexible role binding, it is relatively easy and intuitive for humans to complete.

We tested a CNN-based agent on this task similar to Mnih et al. (2015). These CNN-based architectures have proven effective at a range of visual and spatial tasks such as Atari games, the Mario-like side-scroller CoinRun, and a Minigrid ("DoorKey") benchmark that measures generalization (Mnih et al., 2015; Hilton, Cammarata, Carter, Goh, & Olah, 2020; Sonar, Pacelli, & Majumdar, 2021). Despite the overall simplicity of the task, we found that this architecture failed to learn. Subsequent analysis showed that the model tended to overfit to the training set but was not able to learn flexible role binding.

Finally, we sought to understand why the CNN-based agent failed to learn flexible role binding. We found that our model struggled to use the rule objects, which can be interpreted as configural cues, to complete the task (e.g., Figure 1), but had greater success when the correct role binding was signaled by a single object (e.g., the presence of a red ball). With this knowledge in hand, we hypothesized that our model overfit to the training set because it was "easier" to learn to memorize training examples rather than use these configural cues. Informed by the generalization angle of the diversity hypothesis in Hilton et al. (2020), we constructed an augmented training set which included more training data with diversity at a relevant level of abstraction. This technique improved performance suggesting that the diversity of the training dataset was a limiting factor in the original deep learning agent's ability to learn to use configural role cues.

Experiment Design

In these experiments, we created a simulation study with a minimally constructed video game ("task") to isolate and test flexible role binding. We chose a CNN-based architecture for

[1] https://www.hempuli.com/baba/

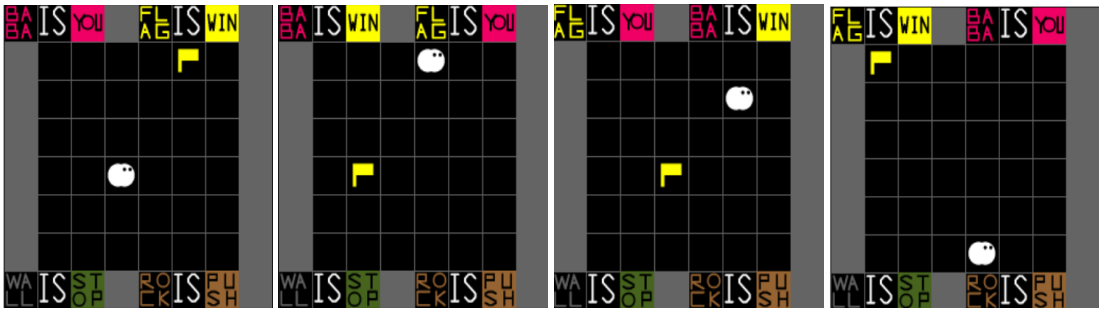

Figure 1: Role rebinding example states. On the far left and far right side, the agent needs to navigate the white figure (YOU object) to the flag (WIN object). On the middle left and middle right side, the roles are reversed. The agent needs to navigate the flag to the white figure. The roles are determined based on the arrangement of "word blocks" in the top row, we refer to these as "configural role cues." For example, in the far left state, the objects arrange to create "BABA IS YOU" and "FLAG IS WIN." The former rule maps the white object to the agent and the latter maps the flag object to the location which the YOU object must occupy to complete the task. The bottom row and the walls along the edge never change. The task was implemented as a Minigrid environment by members of Gureckis' lab.

the task given its success in other tasks involving visual and spatial state spaces, such as Atari or Minigrid. In the next section, we describe the so called "rebinding" task and how we attempted to measure the agent's ability to learn flexible role binding.

Spatial Role Rebinding Task

We consider a minimal *spatial role rebinding task* inspired by the popular game "Baba is You," depicted in Figure 1. The game is presented as a 2D Gridworld where an agent has to navigate to a goal location. Other elements in the world include obstacles (ie., walls, rocks) and "word blocks" that describe the rules of the game. Critically the binding of objects to roles was altered on different levels such that, in some cases, the task was to navigate the avatar (Baba) to the flag (goal) and in other cases the roles were reversed, such that the goal was to navigate the flag to the Baba avatar. Which role binding was required for a level was depicted using "word blocks" at the top of the Gridworld (e.g., "Baba is YOU, Flag is WIN" or "Flag is YOU, Baba is WIN"). To successfully complete a level of the game, the agent must navigate the object which is bound to the "YOU" role through a 6x6 space to occupy the same location as the object bound to the "WIN" role. In this way, the task involves both spatial reasoning and reasoning about object roles. Once the agent identifies the role of the objects, it needs to use the relative location of those objects to navigate to the goal (thus the optimal policy is highly conditional on the rule objects displayed at the top of the world).

We call this role rebinding, or the "rebinding" task, because the roles of the objects are changed between different runs of the game. For the control, we have two tasks. First is the "no rebinding" task. Here the agent need only learn the relative spatial locations. We use one of the configurations in Figure 1 (the far left one) for this task. We also experimented with an "object substitution" variant of the task, where each role could be occupied by different objects but the objects could only occupy a single role (ie., Baba is YOU, Flag is WIN; Rock is YOU, Skull is WIN). It is less challenging than "role rebinding" because the same object cannot take different roles on different trials. For example, the agent can learn that the YOU object is either "Baba" or "Rock." On the other hand, in the "role rebinding" task, "Baba" may be either the YOU object or the WIN object – requiring the agent to adapt its policy as a function of the role cues at the top of the screen.

Evaluating Spatially-Invariant Role-Rebinding

To fairly compare across different architectures and training environments, we focus on behavioral metrics of task performance. Specifically, we measured the success rate of the agent in a fixed number of 20 steps, similar to the metric used by the Multiroom experiments (Igl et al., 2019). To measure overfitting, we compared the success rate score on the training set to that of the test set. A gap between the training and test performance suggests the model is overfitting to the training data. Our evaluation approach is designed to assess generalization of role rebinding across spatial locations.

Problem Generation

We used Procedural Generation to create levels (Cobbe, Hesse, Hilton, & Schulman, 2020). This consisted of configuring a distribution of rules displayed on the top of the game. We also configured where the YOU and WIN objects can appear on the board, holding out 40% of Gridworld cells for the test set. The difference between the training and test set is that in the training set, the WIN object can only occupy 60% of the cells in the grid at random. In the test set, the remaining 40% of cells are used. We chose to hold out these cells to ensure the model had never seen these states before, preventing memorization. Given the translation invariant nature of CNNs, we expected a network that has not overfit to generalize to the test set. We chose to randomly select these cells to prevent the model from memorizing a pattern.

Model Architecture

For our Reinforcement Learning ("RL") training, we use a popular vision-based encoder inspired by Mnih et al. (2015) and implemented using the CleanRL PPO Atari framework from Huang et al. (2022). It uses a convolutional front end with a fully connected hidden layer. Our RL task is set up similar to the Multiroom setup in Igl et al. (2019). We also run experiments using supervised training, since this can act as our upper bound for RL when testing the efficacy of our model's ability to complete the task. Later in the paper we use supervised training. The model architecture is the same, only making adjustments necessary for supervised learning. We fit solely the policy network, training the model to imitate the pre-computed optimal policy with a cross entropy objective. More details about the model and training setup can be found in the Dropbox link.[2]

Results

We seek to understand if artificial agents, specifically CNN-based architectures, can perform flexible role binding. To establish this, we judge models on their ability to spatially generalize their learned policy to a held-out test set. We should expect the models to generalize in the "no rebinding" task because it is a purely spatial task, similar to past successes of these deep-RL agents.

On the other end of complexity, the "rebinding" task also requires reasoning about the roles different objects play in the game. We hypothesize that this should be the hardest for the model to spatially generalize. We expect the "object substitution" task to be somewhere in between since there are different objects playing each role in the game, but they are not (re-)bound to multiple roles.[3]

Task performance

Table 1 shows that the model is able to spatially generalize on "no rebinding" and "object substitution" tasks. However, the model is not able to generalize on the "rebinding" task. The best test score achieved on "rebinding" was 15% despite reaching a training score as high at 95% (indicative of overfitting). This was the best score achieved across many different training configurations that involved standard vision based and RL regularization techniques, including batch norm, dropout, and weight decay (all methods which are used to reduce overfitting) (Cobbe, Klimov, Hesse, Kim, & Schulman, 2019). While the details of these experiments are omitted for brevity, they can be found in the Dropbox link.

The performance on the control tasks is somewhat expected given vision-based encoders' track record solving navigation tasks (Mnih et al., 2015; Hilton et al., 2020). On the other hand, these results demonstrate that "role rebinding" is uniquely challenging despite its simplicity. In this next section, we interpret our model's representations to better understand why it overfits on "role rebinding."

[2] https://bit.ly/3JQsHM5

[3] Experiment code available upon request to the authors

Table 1: Model generalization on the "no rebinding," "object substitution," and "role rebinding" tasks. "Steps": number of steps used in RL training. "Return": moving average of the episodic return at the end of training. "Train"/"Test": percentage of tasks completed in 20 steps, or one trial, on Train/Test set. Both the no rebinding and object substitution models show strong test scores whereas the rebinding does not. Per the last row, the non regularized model overfits.

Task	Steps	Return	Train (%)	Test (%)
Random Policy	N/A	N/A	N/A	22
No rebinding	0.5m	0.50	100	96
Object substitution	1m	0.87	98	94
Role rebinding	1m	0.26	38	15
Role rebinding (no-reg)	1m	0.86	95	4

Interpreting model representations

Hilton et al. (2020) use model interpretability techniques to understand what visual inputs impact the behavior of deep RL agents. Specifically, they determine how different objects in the task influence the model's policy and value function. We do the same. However, since our input space is less complex, we use more traditional interpretability techniques from Molnar (2022). We also seek to understand whether the visual encoder is learning useful features by using those features to predict the rules and relative location of the goal compared to the agent. From this analysis, we find no evidence that the models in the rebinding task produce representations that can be used to determine the abstract rules of the game.

Attribution First, we use interpretability techniques, specifically attribution, using the Python library Captum to understand which Gridworld cells most influence the model policy (Kokhlikyan et al., 2020). As shown in Figure 2, both models have the strongest attribution on the YOU and WIN objects. Notably the rebinding model does not attend to the rules, despite the optimal policy being a function of the rules. Consistent with this, the learned policy has higher entropy compared with the no rebinding policy, which doesn't need to condition its behavior on the rules.

Feature Classification Next, we assessed whether the output of the value encoder, which captures visual features, contains information about the rules or the location of the goal relative to the agent. Specifically, we use the output of the last layer of the encoder, pre-trained on each task, to fit a linear classifier to predict the binding rule (i.e., is Baba "YOU" or is the Flag "YOU"?) or spatial information for states on that task. We found no evidence that the rebinding model represents the binding rules per Table 2, where it achieved a test F1 score of 49, which is below chance.

In Table 3, we assess whether models trained on the no rebinding and role rebinding tasks are able to identify the rel-

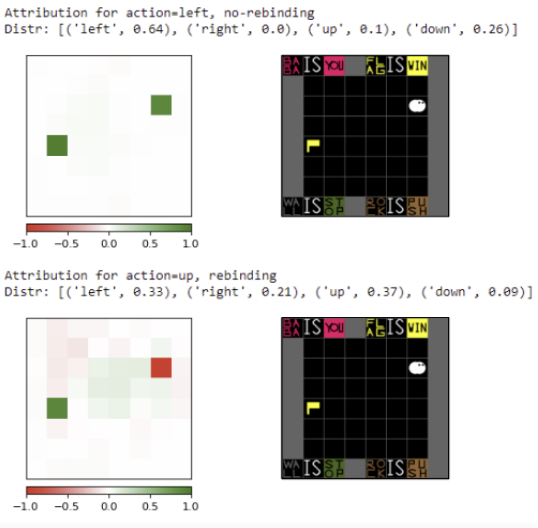

Figure 2: Baba Rebinding Attribution. The boxes in the left column show grid-level attribution by the model for a given action. The right side illustrates the state of the game. The top row is for the no rebinding model and the bottom is for the rebinding model. As we see in the grid level attribution charts, both models are attending to the YOU and WIN object but not the rules. For the rebinding model, where its behavior is conditioned on the rule, the entropy ("Distr") is higher.

ative location of the goal compared to the agent. We observe the test score of the model pretrained on the no rebinding task of 35. This suggests that pretraining on the purely navigation-based task can be used to improve the model's ability to determine the relative location of the goal compared to the agent. This makes sense since CNNs can complete tasks which require spatial reasoning, such as navigating 2D Gridworlds and some Atari games, like Ms. Pacman. Also the no rebinding task is not misled by the need to simultaneously learn the rules like with rebinding. With the rebinding task, the test performance of the pretrained model is 31. We have shown the model is aware of the location of each object but not the roles so it is conceivable that it is guessing between opposing directions, thus achieving a score close to 25. This may suggest the complexities introduced by configural role cues cause memorization, resulting in the model discarding information about the task.

Using supervised learning to understand factors limiting learning

Our RL agent could not learn the task. We first wondered if the failure was due to our RL training procedure or general difficulties in training RL agents. We thus used supervised learning to train our agent[4], and found that it also failed to learn the task in the supervised learning setting. Table 4 shows this result. This suggests that there is an inherent difficulty in the task, and that the limiting factor is either the training data or neural network architecture rather than the training procedure. For the remainder of the paper, we present results based on model training in the supervised setting.

[4]We trained the role rebinding model in a supervised setting (removing the additional complexity of approximating the optimal policy) since we could algorithmically compute the optimal policy for any state.

Table 2: The "rebinding" model does not produce meaningful representations of the binding rule (i.e., "Baba is YOU" or "Flag is WIN") as evidenced by its test score of 49, which is below the random chance score of 50. The object substitution model shows some evidence of being aware of the rules with a test score of 68. Reported metric is F1 score. "Random weights" is a randomly initialized network with no pre-training on the task. Plus/minus is the standard deviation.

	Random Weights		Pretrained	
Task	Train	Test	Train	Test
Object Substitution	61 ± 9	60 ± 9	68	59
Role Rebinding	64 ± 1	43 ± 0	48	49

Table 3: The "no rebinding" model produces representations which can be used to determine the relative location of the goal compared to the agent based on its test score of 35. The rebinding model achieves a score of 31, which may be within the margin of error for knowing spatial information but not the roles. Reported metric is F1 score. 8 class classification for all possible directions, random guess score is 12.5. "Random weights" is a randomly initialized network with no pre-training on the task. Plus/minus is standard deviation.

	Random Weights		Pretrained	
Task	Train	Test	Train	Test
No rebinding	35 ± 3	24 ± 2	46	35
Role Rebinding	40 ± 4	35 ± 2	29	31

Configural role cues are a limiting factor

Next, we sought to understand what factors of the task made it so challenging to learn. We hypothesized that the configural role cues represented a barrier to learning to act in a rule-dependent way. To test this hypothesis, we ran two experiments where we removed the barrier of configural role cues by way of providing a simpler, non-configural indicator of the role. In both experiments, our agent demonstrated significant improvement in its ability to perform flexible role binding in these simplified tasks. This demonstrates that the configural presentation of the role cues was a limiting factor in the previous experiments.

Role Injection We provided a signal of the rules directly to the feature layer of our network. This way, we bypass the CNN encoder which could be dropping binding rule information. At each time-step, the binding rule was converted to a one-hot vector and concatenated with the feature vector. This allows us to assess whether the linear head is capable

Table 4: Percentage of tasks completed in 20 steps (performance) of models trained using RL vs. supervised (SV) on role rebinding task as a function of number of gradient updates. "Train" ("Tr") and "test" ("Te") are reported as percentages.

RL			SV		
Updates	Tr.	Te.	Updates	Tr.	Te.
86	13	12	1	14	10
172	22	8	10	25	4
343	38	15	200	50	4
			2000	77	3

Table 5: Impact of binding "role injection." We see that with binding "role injection", the model is able to successfully generalize on the rebinding task. Trained on 200 updates.

	No Inj. (%)		Injection (%)	
Task	Train	Test	Train	Test
No rebinding	99	80	99	79
Rebinding	50	4	99	67

of solving the task when the role features are unambiguously available.

Per Table 5, we see the model is able to successfully generalize for the rebinding task. This can be seen by the fact that the test score for the "Injection" column is 67% for rebinding, which is well above the 4% observed without it.

Visual Role Hint Our role injection experiments demonstrated that our architecture was capable of learning a role-dependent navigation policy. Next, we sought to investigate if it could learn a role-dependent policy if a non-configural role hint was presented in the input. We wanted to see if the CNN was capable of propagating this information to the policy head.

The non-configural role hint took the form of an object placed in a fixed location on the grid. For example, a green dot would indicate the role "Baba is YOU" (left of Figure 3) while an orange square would indicate the role "Flag is YOU." As was also the case with the role injection experiments, we did not consider the order that rules were presented since it doesn't matter and removes another layer of complexity in interpreting the rules. This means binding rules of "Baba is YOU, Flag is WIN" and "Flag is WIN, Baba is YOU" are equivalent.

Per Table 6, we see that augmenting the inputs with visual role hints supported successful learning of the role rebinding task. This is evidenced by a test score reaching 78% on the task containing role hints, compared with 4% without them.

Together, the performance boosts afforded by both "role injections" and "visual role hints" tell us that our architecture is capable of flexible role rebinding when the rules are presented is a straightforward way. Put differently, the model struggles when the rules are presented in a distributed configural format, as in the standard rebinding task.

Table 6: Impact of "visual role hint." We see that with "visual role hint", the model is able to successfully generalize on the rebinding task. Role hints has half as many states.

	No Role Hints (%)		Role hints (%)	
Updates	Train	Test	Train	Test
20	50	4	29	6
200	50	4	96	78

Table 7: Performance on rebinding task with and without "task dependent" noise. Dataset sizes are 10k and 2.9k which is $<1\%$ and 98% of the dataset for the noise and no noise rebinding task respectively.

	Task Dep. Noise (%)		No noise (%)	
Updates	Train	Test	Train	Test
200	49	4	50	4
2000	99	54	77	3
10000	100	67	98	9

Data diversity supports learning to use configural role cues

We next wanted to understand if the inability to use configural role cues was due to a limitation of the training data distribution. In particular, we hypothesized that the training dataset might have been insufficiently large and/or diverse to support learning to use configural role cues. Indeed, previous work has shown that generalization increases with the number and diversity of procedurally generated levels in the training dataset (Hilton et al., 2020).

To construct a larger and more diverse dataset, we generated new levels by adding "task dependent" noise in which we add several benign "distractor" objects to the grid as in Figure 3. Per Table 7, the noise helps the model to generalize better in the "task dependent" noise setting.[5]

We also found that increasing diversity at the relevant level of abstraction was important. While generating new levels by adding distractor objects improved spatial generalization with configural role cues, generating new levels by adding Gaussian noise to the input did not (Table 8). Our intuitive explanation for the success of adding distractor objects is that adding diversity at the relevant level of abstraction makes naive memorization of training examples more difficult, thus regularizing the model towards learning the true abstract function.

[5]Although this strategy improves generalization with configural role cues, it still performed worse (67%) than role hints (78%). Moreover, we present the results of one successful training run but it was unstable. For more information on the instability see the Dropbox link.

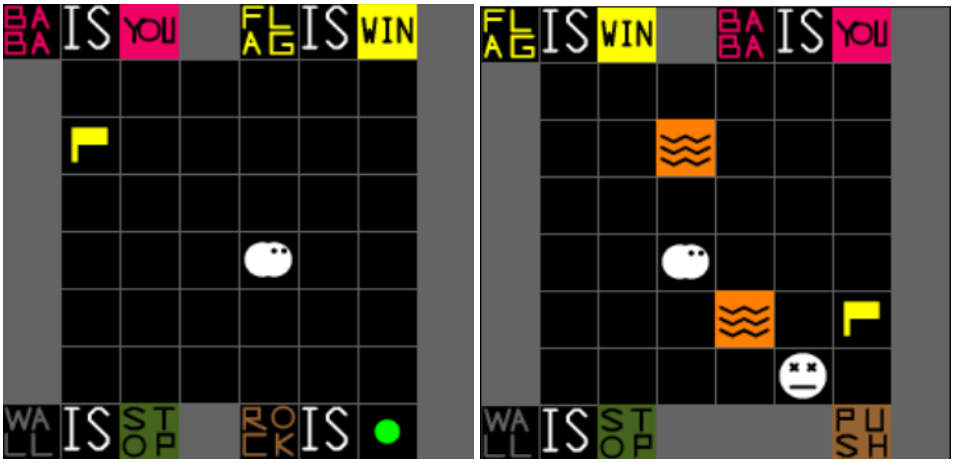

Figure 3: Left: Baba Rebinding with Role Hints. The green dot corresponds to the configuration of "Baba is YOU, Flag is WIN." Right: Baba Rebinding with Task Dependent Noise. Two different benign objects are added for each trial: lava (orange) and skull (white) to the task. The agent cannot interact with them.

Table 8: Performance on rebinding task with handcrafted task-dependent vs. Gaussian noise. Num updates=2k, dataset size=10k. To add Gaussian noise, we added a tensor with $\mu = 0, \sigma = 0.5$ to the original input. We also tried $\sigma = 1.0$.

Gaussian		Task Dependent	
Train (%)	Test (%)	Train (%)	Test (%)
36	6	99	54

Table 9: Summary of model spatial generalization results on rebinding task under various training settings.

Setting	Generalizes?
Default	No
No rebinding	Yes
Augmentation, Visual Hint	Yes
Augmentation, Feature Injection	Yes
Data Augmentation, Gaussian	No
Data Augmentation, Task Dep.	Yes

In summary, we found that the diversity of the training dataset was a limiting factor in the original deep learning agent's ability to learn to use configural role cues. From a psychological perspective, adding diversity to the training dataset corresponds to adding "desirable difficulties" to the learning curriculum, which has been shown to be beneficial for human learners (Bjork & Bjork, 2011).

Summary

We sought to study if artificial agents, specifically those CNN-based architectures that have found success in a range of game-like tasks, can learn to perform flexible role binding and rebinding. We did this by testing it on a Gridworld navigation game with a role rebinding element. Initial experiments showed that these agents fail to learn rebinding (measured via spatial generalization to the test set) when roles are presented as configural role cues (see Setting "Default" in Table 9). This suggests a gap between how these agents and humans learn. Where a human might look for an abstract pattern, the agent seems to naturally memorize. We then identified an augmented training regime which allowed the model to learn the abstraction and generalize on the task. It not only required a sufficient amount of training data, but also diversity at the relevant level of abstraction (see Setting "Data Augmentation, Task Dependent" in Table 9).

From a deep learning perspective, we conjecture that adding diversity at the relevant level of abstraction makes naive memorization of training examples more difficult for the agent, thus regularizing the model towards learning the true abstract function. Future work can be done to explore this further. Overall, our results highlight the importance of careful analysis of both the training environment and model architecture in determining the learnability of specific patterns.

References

Bjork, E. L., & Bjork, R. A. (2011). Making things hard on yourself, but in a good way: Creating desirable difficulties to enhance learning. *Psychology and the real world: Essays illustrating fundamental contributions to society*, *2*(59-68).

Chevalier-Boisvert, M., Dai, B., Towers, M., de Lazcano, R., Willems, L., Lahlou, S., ... Terry, J. (2023). Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks. *CoRR*, *abs/2306.13831*.

Cobbe, K., Hesse, C., Hilton, J., & Schulman, J. (2020, 13–18 Jul). Leveraging procedural generation to benchmark reinforcement learning. In H. D. III & A. Singh (Eds.), *Proceedings of the 37th international conference*

on machine learning (Vol. 119). PMLR. Retrieved from https://proceedings.mlr.press/v119/cobbe20a.html

Cobbe, K., Klimov, O., Hesse, C., Kim, T., & Schulman, J. (2019, 09–15 Jun). Quantifying generalization in reinforcement learning. In K. Chaudhuri & R. Salakhutdinov (Eds.), *Proceedings of the 36th international conference on machine learning* (Vol. 97). PMLR. Retrieved from https://proceedings.mlr.press/v97/cobbe19a.html

Gentner, D., Holyoak, K. J., & Kokinov, B. N. (2001). *The Analogical Mind: Perspectives from Cognitive Science*. The MIT Press. Retrieved from https://doi.org/10.7551/mitpress/1251.001.0001 doi: 10.7551/mitpress/1251.001.0001

Hilton, J., Cammarata, N., Carter, S., Goh, G., & Olah, C. (2020). Understanding rl vision. *Distill*. (https://distill.pub/2020/understanding-rl-vision) doi: 10.23915/distill.00029

Huang, S., Dossa, R. F. J., Ye, C., Braga, J., Chakraborty, D., Mehta, K., & Araújo, J. G. (2022). Cleanrl: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, *23*(274). Retrieved from http://jmlr.org/papers/v23/21-1342.html

Igl, M., Ciosek, K., Li, Y., Tschiatschek, S., Zhang, C., Devlin, S., & Hofmann, K. (2019). Generalization in reinforcement learning with selective noise injection and information bottleneck. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems* (Vol. 32). Curran Associates, Inc.

Kokhlikyan, N., Miglani, V., Martin, M., Wang, E., Alsallakh, B., Reynolds, J., … Reblitz-Richardson, O. (2020). *Captum: A unified and generic model interpretability library for pytorch.*

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., … Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*. Retrieved from https://doi.org/10.1038/nature14236

Molnar, C. (2022). *Interpretable machine learning* (2nd ed.). Retrieved from https://christophm.github.io/interpretable-ml-book

Sonar, A., Pacelli, V., & Majumdar, A. (2021, 07 – 08 June). Invariant policy optimization: Towards stronger generalization in reinforcement learning. In A. Jadbabaie et al. (Eds.), *Proceedings of the 3rd conference on learning for dynamics and control* (Vol. 144). PMLR. Retrieved from https://proceedings.mlr.press/v144/sonar21a.html