Learning to Represent State with Perceptual Schemata

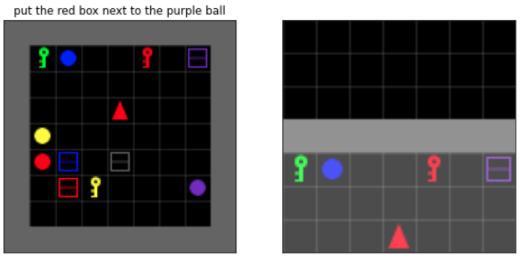
Anonymous Authors[1]

Abstract

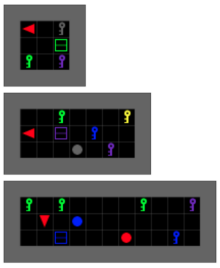
The real world is large and complex. It is filled with many objects besides those defined by a task and objects can move with their own interesting dynamics. How should an agent learn to represent state to support efficient learning and generalization in such an environment? In this work, we present a novel memory architecture, *Perceptual Schemata*, for learning and zero-shot generalization in environments that have many, potentially moving objects. Perceptual Schemata represents state using a combination of schema modules that each learn to attend to and maintain stateful representations of different subspaces of a spatio-temporal tensor describing the agent's observations. We present empirical results that Perceptual Schemata enables a state representation that can maintain multiple objects observed in sequence with independent dynamics while an LSTM cannot. We additionally show that Perceptual Schemata can generalize more gracefully to larger environments with more distractor objects, while an LSTM quickly overfits to the training tasks.

1. Introduction

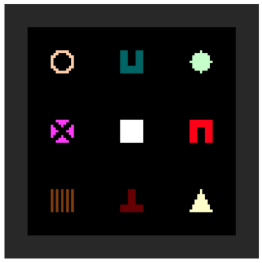
How should an agent learn to represent state to enable proficiency in large, complex environments? We focus on environments that have many, potentially moving objects where task completion requires decision-making over potentially hundreds of time-steps. For example, consider the "Keybox" task in figure 1(b) used in our experiments in §4. The agent begins at the left-most side of the hallway where a colored box (blue in the bottom hallway) indicates the color of the goal key the agent must retrieve from the other end of the hallway. The hallway is filled with distractor objects which may obstruct the agent's path. We are interested in studying how an agent's state representation affects (a) its capacity for efficient learning in such environments and (b) its capacity to generalize to larger, more complex environments. If the hallway length and number of distractors are doubled, can the agent generalize its goal-directed behavior in a zero-shot manner?



(a) Place X next to Y.



(b) Keybox task.



(c) Ballet task.

Figure 1. Tasks used for experiments.

It is important that we develop agents that can handle large environments with many, potentially moving objects because the real-world is such an environment. Houses are filled with objects that have to be moved or navigated around in order to accomplish tasks. Streets and roads are filled with other agents moving with their own dynamics. Artificially intelligent agents will need to learn representations for state that can handle these settings.

While the combination of deep learning and reinforcement learning (deep RL) has shown strong performance in single-task settings with millions of transitions (Mnih et al., 2015; Lillicrap et al., 2015; Silver et al., 2016; Schulman et al., 2017), it hasn't had the same success with generalization that deep learning has had in computer vision (He et al., 2020; Kawaguchi et al., 2017; Yosinski et al., 2014) and natural language processing (Ramesh et al., 2021; Devlin et al., 2018). For example, deep RL agents are known to have a hard time generalizing to small task-irrelevant changes to the observation space (Cobbe et al., 2019; Kansky et al., 2017; Witty et al., 2018; Farebrother et al., 2018). If simply adding task-irrelevant visual information is challenging,

it follows that having larger environments with more task-irrelevant objects to navigate around or move will further challenge a deep RL agent's capacity to generalize.

We hypothesize that one source of this challenge is the canonical choice for representing state: a long-short term memory (LSTM) (Hochreiter & Schmidhuber, 1997). LSTMs are known to have a recency bias (Ravfogel et al., 2019), so it may be hard to maintain sequentially observed object dynamics in state. Additionally, they have shown poor generalization to larger sequences in sequence-to-sequence tasks (Graves et al., 2014; Lake & Baroni, 2018) indicating they may not generalize representations of state to larger, more complex environments than observed during training.

In this work, we present a novel memory architecture, *Perceptual Schemata*, for learning and zero-shot generalization in environments that have many, potentially moving objects. Perceptual Schemata draws inspiration from cognitive science and represents state with a combination (or "assemblage" (Arbib, 1992)) of perceptual schemata modules. A schema is a "cognitive structure presenting [an agent's] knowledge about some entity or situation" (APA, 2020). A perceptual schema defines two sets of parameters–one for determining filtering schema-relevant information from the agent's observation (which we model with attention parameters) and another for maintaining a representation of the relationship of this information to the agent (which we model with recurrent parameters) (Minsky, 1979; Arbib, 1992). In order to learn perceptual schemata that can specialize on arbitrary environment fragments such as stationary objects, moving objects, or sets of objects, each schema learns to attend to subspaces along the feature dimension of a spatio-temporal tensor describing the agent's observation. Different schemata learn to specialize on different aspects of the environment and can be used in tandem to combine an estimate of the environmental state with a seperate representation of the agent's inferred goal.

We study Perceptual Schemata in **simple, diagnostic** grid-world experiments that test a memory architecture's capacity to learn and generalize when environments are either large, messy and filled with objects, or when they have multiple objects moving with independent dynamics in sequence. We present empirical results showing that Perceptual Schemata can disentangle and maintain in state multiple objects with independent dynamics while an LSTM cannot. We additionally show that Perceptual Schemata can generalize more gracefully to larger environments with more distractor objects, while an LSTM quickly overfits.

2. Background

We are concerned with partially observable environments where an agent experiences visual observations $x_t \in \mathcal{X} = \mathbb{R}^{H_{\texttt{image}} \times W_{\texttt{image}} \times C}$ at time-step t. The agent selects an action $a_t \in \mathcal{A}$ using a policy $\pi(a_t|s_t)$, where $s_t \in \mathcal{S}$ is the agent state representation that describes its summary of its current situation in the environment or *aleatoric state* (Lu et al., 2021). We focus on how to best learn this state representation.

2.1. Prior methods for representing state.

Representing state with a window of observations. Researchers have used a variety of methods to learn state-representations. One simple but scalable method is to use a window of the past N observations (Mnih et al., 2015; Lillicrap et al., 2015; Parisotto et al., 2020) with a function that processes them in a feedforward manner. Some researchers choose to leverage external memory so that an agent can incorporate arbitrary observations in the past into its state (Pritzel et al., 2017; Blundell et al., 2016).

Representing state with a memory architecture. In this work, we focus on the setting where an agent learns an iterative function of state based on a distinct initial state representation $s_0 \in \mathcal{S}$ and the consequently experienced observations and actions (Lu et al., 2021). This setting commonly manifests with three main functions: one, f, for learning features $z_t \in \mathcal{Z}$ of observations x_t; another, g, for using the previous state representation s_{t-1}, previous action a_{t-1} and current observation features to compute a representation of state $s_t \in \mathcal{S}$; and a third, the policy π from above:

$$f : \mathcal{X} \to \mathcal{Z} \tag{1}$$

$$g : \mathcal{S} \times \mathcal{A} \times \mathcal{Z} \to \mathcal{S} \tag{2}$$

$$\pi : \mathcal{S} \to \mathcal{A} \tag{3}$$

This flavor of deep RL is potentially the most popular and has been quite successful in mastering environments (Sorokin et al., 2015; Mirowski et al., 2017; Hessel et al., 2018; Jaderberg et al., 2017; Wang et al., 2017; Sallab et al., 2017; Heess et al., 2015; Schulman et al., 2017; Vinyals et al., 2019). Coupled with the strong success of neural networks for learning and transfer in computer vision settings (Krizhevsky et al., 2012; Yosinski et al., 2014; Kawaguchi et al., 2017; He et al., 2020), we posit that this is an indication that neural networks combined with reinforcement learning can already learn a suitable function f with good observation features.

A large body of work has found that deep RL agents often fail to *generalize* their policy π to minor variations of the environment (Cobbe et al., 2019; Kansky et al., 2017; Witty et al., 2018; Farebrother et al., 2018). We posit that

one source of this challenge is that we still have progress to make in learning a g that maps observation features to a high-level state representation. In particular we argue that part of the problem is the canonical architecture for learning g—a large, unstructured Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997; Shi et al., 2015)— which is known exhibit poor *systematic* generalization (Lake & Baroni, 2018).

Most similar to Perceptual Schemata are "Recurrent Independent Mechanisms" (RIMS) (Goyal et al., 2020b) and "Schema / object-file factorization" (SCOFF) (Goyal et al., 2020a). All of these memory architectures learn multiple smaller LSTMs which specialize on aspects of the environment. However, RIMS and SCOFF utilize transformer-style attention (Vaswani et al., 2017) to capture an object as a vector in the spatial grid of a Convolutional Neural Network's output tensor. We opt for a more flexible attention mechanism which can capture environment fragments *across* the spatial grid. SCOFF extends RIMs by decoupling "object-recognition" attention parameters from object-dynamics recurrence parameters—we leave such an extension of our architecture as future work. Additionally, they focus their experiments on showing zero-shot generalization for supervised learning tasks. We focus on zero-shot generalization in a reinforcement learning context where an agent must generalize its policy to larger environments that require action-prediction sequences two, three, and four times longer than observed in training.

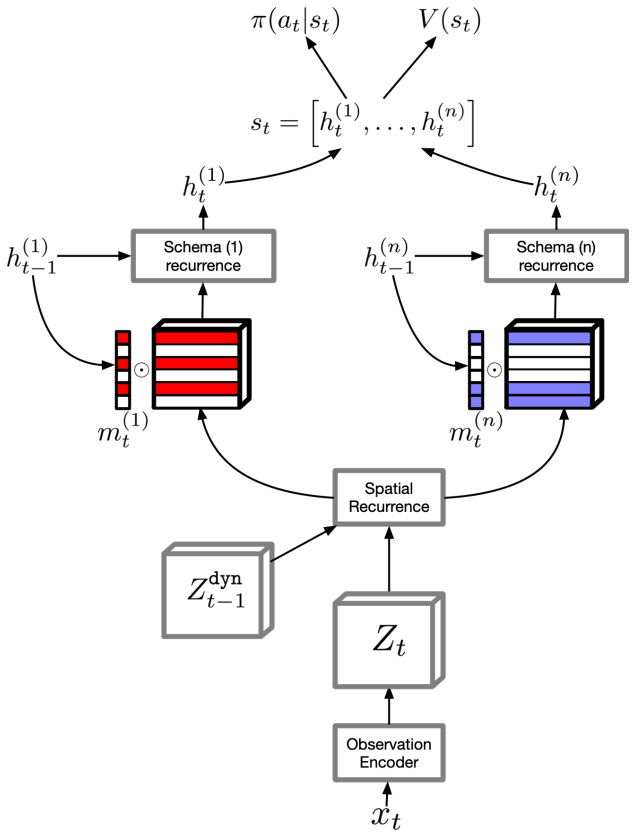


Figure 2. Architecture. To avoid clutter, we do not show message passing between schemata.

3. Perceptual Schemata

How can an agent discover perceptual schemata? We opine that the power and beauty of deep RL is its ability to find solutions unimagined by the thoughtful and clever AI researcher. With that in mind, we develop the inductive biases for discovering perceptual schemata such that they offer a neural network maximum flexibility to decompose its representation of the environment as is most useful for the given task.

Choice of f and π. As is common in vision settings, we use a convolutional neural network (CNN) as our function f for obtaining visual features $Z_t = \mathrm{CNN}(x_t)$ where $Z_t \in \mathbb{R}^{H_z \times W_z \times D_z}$. For π, we choose another standard choice—a multilayer perceptron (MLP) that outputs a categorical distribution over actions. We now focus our attention on a suitable choice of g.

3.1. Choice of g for representing state

Learning spatio-temporal features. We first observe that Z_t is impoverished in that it doesn't capture features that describe local spatial dynamics. We can obtain a representation that captures this by employing a convolutional LSTM (Shi et al., 2015): $Z_t^{\texttt{dyn}} = \mathrm{ConvLSTM}(Z_t, Z_{t-1}^{\texttt{dyn}})$. With this, we have a tensor $Z_t^{\texttt{dyn}} \in \mathbb{R}^{H_z \times W_z \times D_z}$ describing the expression of features $Z_{t,h,w}^{\texttt{dyn}}$ and their temporal dynamics over a spatial grid of size $H_z \times W_z$.

Prior work has show that in simple environments, a neural network can learn to associate high-level objects with individual features $Z_{t,h,w,j}^{\texttt{dyn}}$ (Kipf et al., 2019). However, in more complex environments, it may be suboptimal to use a single feature to describe a single object. An architecture may instead learn a *distributed* representation where different objects share features (Greff et al., 2020). If $Z_t^{\texttt{dyn}}$ captures information about task-relevant fragments of the environment — objects, sets of objects, etc. — but uses distributed representations, then what is represented by a particular vector $Z_{t,h,w}^{\texttt{dyn}}$ is **ambiguous** (Greff et al., 2020). In this work, we argue that we can improve zero-shot generalization if schemata can learn to **disambiguate** their environment fragment's contribution to the feature dimensions the fragment is represented over. We hypothesize that this will enable them to be recognized and integrated into "state" in novel ecological situations.

Learning to attend to environment fragments using fea-

ture subspaces. In this work we choose to learn n schemata with a collection of LSTMs, $\{\text{LSTM}_{\theta_i}\}_{i=1}^{n}$. Our key insight is that we can enable schemata to leverage disentangled representations of environment fragments by having them learn to attend to feature subspaces. We achieve this by having schema i obtain its input $v_t^{(i)}$ using a mask $m_t^{(i)}$ over features $Z_t^{\texttt{dyn}}$. Practically, we found it useful to project the features before and after masking using shared parameters as in (Hu et al., 2018; Andreas et al., 2016). This forces each schema to use a shared representation for representing their attention input (Greff et al., 2020). Schema i uses its state from the previous time-step $h_{t-1}^{(i)}$ to query for its input as follows:

$$m_t^{(i)} = \sigma(W_i^m h_{t-1}^{(i)}) \tag{4}$$

$$v_t^{(i)} = \text{conv}_2\left(\text{conv}_1(Z_t^{\texttt{dyn}}) \odot m_t^{(i)}\right) \tag{5}$$

where $m_t^{(i)} \in \mathbb{R}^{D_m}$, $v_t^{(i)} \in \mathbb{R}^{H_z \times W_z \times D_m}$, σ is a sigmoid function, and conv is a convolutional layer. By using a subspace of $Z_t^{\texttt{dyn}}$ expressed over the $H_z \times W_z$ grid, this allows schemata to capture either coarse or granular decompositions of the environment: i.e. objects, sets of objects, etc. Additionally, using feature subspaces allows schemata to overlap as needed across spatial dimensions, temporal dimensions, or feature dimensions. Importantly, this allows for the task-signal (i.e. reward signal) to drive how perception is decomposed.

Learning Perceptual Schemata. We now arrive at our update rule. Before each feature update, each schema gathers information from other schemata with a round of message-passing using transformer-style attention (Vaswani et al., 2017). This allows the schemata to coordinate and share information[1]. Using independent parameters, each schema treats its previous hidden-state and action as a query $q_t = \left[h_{t-1}^{(i)}, a_{t-1}\right] W_i^q \in \mathbb{R}^{1 \times D_m}$. We create matrix from all previous hidden-state $H_{t-1} = \left[h_{t-1}^{(i)}\right]_i$ to obtain keys and values $K_t = H_{t-1} W_i^k \in \mathbb{R}^{n \times D_m}$, $V_t = H_{t-1} W_i^v \in \mathbb{R}^{n \times D_m}$. The full update for schema i is

$$h_{t-1}^{\texttt{message},(i)} = \text{softmax}\left(\frac{q_t K_t^\top}{\sqrt{D_m}}\right) V_t \tag{6}$$

$$h_t^{(i)} = \text{LSTM}_{\theta_i}([v_t^{(i)}, h_{t-1}^{\texttt{message},(i)}, h_{t-1}^{(i)}, a_{t-1}]) \tag{7}$$

[1]We note that employing message-passing might have a similar utility to having a posterior likelihood model in a dynamics Bayesian network where each factor is conditioned on all factors in the previous time-step.

We summarize the parameters for schema i below:

$$W_i^m \text{ : subspace parameters}$$
$$\theta_i \text{ : update parameters}$$
$$(W_i^q, W_i^k, W_i^v) \text{ : messaging passing parameters}$$

As a simple first step, we represent state as the concatenation of all schemata-states:

$$s_t = [h_t^{(1)}, \ldots, h_t^{(n)}], \tag{8}$$

and use this to compute reinforcement learning quantities such as the value-function $V(s_t)$ and policy $\pi(a_t|s_t)$. We present a full schematic of our architecture in figure 2.

4. Experiments

In this section, we study the following questions:

1. Can Perceptual Schemata disentangle and maintain multiple objects with distinct dynamics in state?
2. Is Perceptual Schemata robust to an increasing number of distractor objects?
3. Can Perceptual Schemata generalize goal-oriented behaviors to longer horizons than trained on?
4. Does Perceptual Schemata maintain its generalization capacity as the number of distractor objects increases?

Disentangling multiple objects with distinct dynamics. We study this question with the "ballet" gridworld (Lampinen et al., 2021) in Figure 1(c). The agent is a white square in the middle of the grid. Each other object corresponds to a "ballet-dancer" which moves in a distinct pattern for 16 time-steps. Afterwards, there is a pause of 48 time-steps and another ballet-dancer "dances". Once all ballet-dancers have gone, the agent is given a language instruction—e.g., "go to the spinning ballet-dancer"—and it must go to the correct ballet-dancer. The agent gets a reward of 1 if chooses the correct dancer, and 0 otherwise. All shapes and colors are randomized making the dynamics the only feature that indicates the task-object. For recurrent memories, this task tests a memory's ability to maintain *separate, independent* dynamics in an agent's state-representation. A poorly performing agent will obtain chance performance, which corresponds to $1/m$, where m is the number of ballet-dancer's in the environment. We use this to test whether a memory architecture can disentangle multiple distinct dynamics and maintain them in a state representation.

We present results in figure 3. We compare Perceptual Schemata against an LSTM with the same number of parameters and a two-layer RNN, where the first layer is a

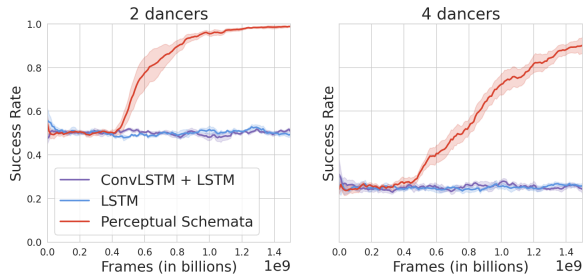


Figure 3. Performance on the Ballet task (5 runs).

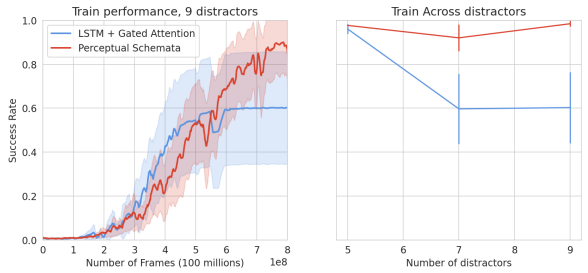


Figure 4. Performance Place X next to Y.

Convolutional LSTM and the second layer is an LSTM. We treat this experiment as an ablation because without our attention and structure within state, we are equivalent to a 2-layer RNN that uses a Convolutional LSTM and LSTM. Interestingly, we find that with only 2 ballet-dancers (figure 3, left), neither baseline goes above chance performance. While both baselines can certainly learn to model the dynamics, they seem less capable of *retaining* this information in memory, even when required so by the task.

This result should elicit pause for deep RL researchers and practitioners since LSTMs are a standard choice when learning a state-representation. This lack of "long-term" recall is a known problem in the community and has led to a rise in interest in alternative methods for representing state with better recall capacities, such as transformers (Parisotto et al., 2020). Fortunately, we show that at least this aspect of the recall challenge can be easily mitigated with our inductive bias for attending to subspaces of spatio-temporal features.

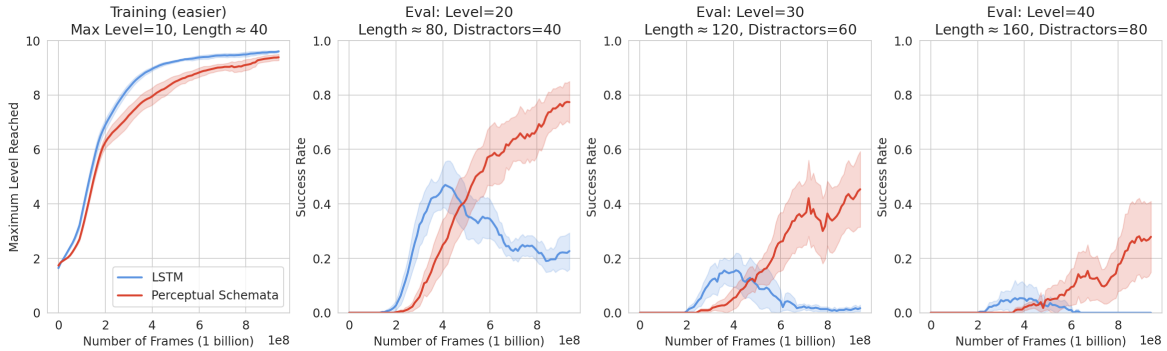
Robustness to distractors. We study this question with a simple task of "Place X next to Y" in the BabyAI gridworld (Chevalier-Boisvert et al., 2019) Figure 1(a). The agent is a red triangle. Other objects can be squares, boxes or circles and they can take on 7 colors. The agent receives a partial, egocentric observation of the environment (Figure 1(a), right) and is given a synthetic language instruction. The agent gets a reward of 1 if chooses the correct dancer, and 0 otherwise. This task should not be challenging. However, as the number of distractors increases, the likelihood a distractor is either (a) confounding with the task objects or (b) blocks/confuses the agent also increases. We study how agent performance on this task deteriorates for $\{5, 7, 9\}$ distractor objects. As the number of objects increases, the number of environment configurations of objects increases (at least for the level of sparsity we study) inducing different available dynamics for the agent.

We present results in figure 4. On the left we present results for the hardest setting: 9 distractors. On the right, we present the maximum success rate (and corresponding standard error) achieved by each method for each distractor setting. We compare our method to an LSTM paired with gated-attention (Chaplot et al., 2018) which has proven effective for navigating to objects described in language instructions. We note that both memories get about the same performance when there are 5 distractors. However, an LSTM with gated-attention degrades signifcantly in performance for 7 and 9 distractors. We conjecture that our Perceptual Schemata can better focus on task-objects and handle the various "situations" the agent faces as a result of different distractor configurations (e.g. being blocked by objects, or having to navigate around objects).

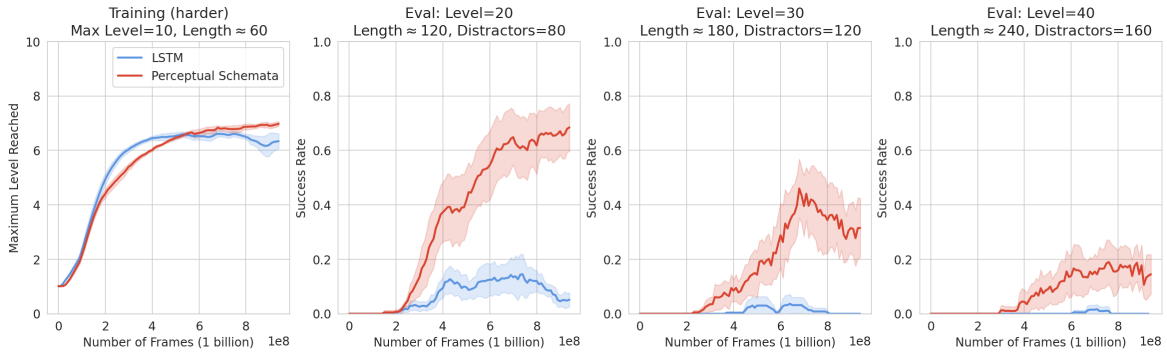
Generalizing to larger environments.

For this experiment, we create the multi-level "keybox" environment depicted in figure 1(b). The first level begins with an agent in a room that has a box, a **key of the same color**, and 2 distractor objects (either a ball of any color or keys of other colors). The agent's task is to bring the **key of the same color** to the box. Each time the agent succeeds, it is teleported into a larger hallway and gets a reward of $n/n_{\texttt{max}}$ where $n_{\texttt{max}}$ is the maximum level the agent can complete. We set $n_{\texttt{max}} = 10$. Hallways are divided into units of length $\approx l + 1$ where l is the height of the hallway. In level n, the agent is in a hallway of length $\approx n(l + 1)$ with $2n$ distractors. We call each successive room a "level". The agent has $50n$ time-steps to complete a level and may fail by timing out. Assuming the agent has completed up to level $n_{\texttt{done}}$, once it fails, the agent restarts to level $n \in [1, n_{\texttt{done}}]$. The box is always on the leftmost side of the hallway and the key is always on the rightmost side. The agent maximizes reward by achieving levels with increasingly long hallways. Thus, this task is a test of **memory** as it requires the agent learn to remember the color of the box for increasingly long time-horizons. The distractors pose an additional challenge of either obstructing or blocking paths. They have to be incorporated into the state-representation so they can be used to predict actions, but they can't overwrite the goal information obtained from the box. We note that the agent's memory isn't erased with each hallway unless it fails. We set the unit length $l = 3$, resulting in a maximum hallway length ≈ 40.

We present results in the top-panel of figure 5(a). On the



(a) Performance on easier version of Keybox task



(b) Performance on harder version of Keybox task where hallways are longer and there's a higher density of distractors.

Figure 5. Performance on Keybox task.

top-left, we present train results. An LSTM was able to get the highest level slightly more quickly than Perceptual Schemata. On the next 3 panels, we see evaluation of both architectures over the course of training for levels $\{20, 30, 40\}$. We see that an LSTM quickly overfits to shorter levels. Its performance never gets above 50% for level 20 or above 20% for level 30. We see that Perceptual Schemata is able to continue to improve on held-out evaluation levels, reach $\approx 80\%$ for level 20, $\approx 50\%$ for level 30, and $\approx 25\%$ for level 40.

Why might Perceptual Schemata achieve such a performance gap? Our structured state and attention allow it to separately store fragments of the environment like the box color. We saw in the Ballet-dancer task that Perceptual Schemata can retain task-relevant features in state even after observing other features. This may help it successfully incorporate distractor information into state to react appropriately while mitigating the loss of the other important task information such as the box color.

Maintaining distractor robustness when generalizing to larger environments. In this experiment we study the degree to which Perceptual Schemata maintains its generalization performance when we increase the unit-length of the hallway to $l = 5$ and increase the number of distractors per unit to $n = 4$. This leads to longer hallways with more distractors. We hypothesize this makes the task harder to learn. Some evidence of this is that we saw very slow learning unless we used a stricter learning curriculum where the agent **always** starts an episode on level 1 (as opposed to a randomly completed level). This leads the agent to obtain more training data with easier tasks.

We present results in the bottom panel of figure 5. While the maximum level is 10, neither agent achieves beyond level 7, which corresponds to a hallway of length ≈ 42 which is roughly the same as the maximum length of 40 in the easier setting. Despite reaching the same hallway length, we see that an LSTM has *considerably worse* generalization, achieving less than 20% success rate for any of these settings. We find that the distractor robustness observed for *training* in the Place X next to Y task was able to transfer towards *generalization robustness* for this task. Perceptual Schemata is able to achieve about 80% on level 20 and 45% on level 30.

5. Conclusion

We have presented Perceptual Schemata, a novel recurrent neural network architecture for learning to decompose an

agents perception as useful for the task at hand. We present empirical results showing that it can disentangle multiple objects with distinct dynamics and maintain them seperately in memory for later recollection. Perhaps our most striking result is that the *capacity for robustness to distractors* observed for Place X next to Y during training was able to transfer over as a **capacity for robustness to distractors during generalization to larger environments**. We find these to be compelling initial findings indicating that memory architectures for modelling fragments of the environment—that Perceptual Schemata—are interesting tools to do research on when trying to learn state representations for efficient learning and generalization.

References

Andreas, J., Rohrbach, M., Darrell, T., and Klein, D. Neural module networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 39–48, 2016.

APA. Schema, 2020. URL https://dictionary.apa.org/schema.

Arbib, M. A. Schema theory. *The encyclopedia of artificial intelligence*, 2:1427–1443, 1992.

Blundell, C., Uria, B., Pritzel, A., Li, Y., Ruderman, A., Leibo, J. Z., Rae, J. W., Wierstra, D., and Hassabis, D. Model-free episodic control. *ArXiv*, abs/1606.04460, 2016.

Chaplot, D. S., Sathyendra, K. M., Pasumarthi, R. K., Rajagopal, D., and Salakhutdinov, R. Gated-attention architectures for task-oriented language grounding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2018.

Chevalier-Boisvert, M., Bahdanau, D., Lahlou, S., Willems, L., Saharia, C., Nguyen, T. H., and Bengio, Y. BabyAI: First steps towards grounded language learning with a human in the loop. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=rJeXCo0cYX.

Cobbe, K., Klimov, O., Hesse, C., Kim, T., and Schulman, J. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*, pp. 1282–1289. PMLR, 2019.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Farebrother, J., Machado, M. C., and Bowling, M. Generalization and regularization in dqn. *arXiv preprint arXiv:1810.00123*, 2018.

Goyal, A., Lamb, A., Gampa, P., Beaudoin, P., Levine, S., Blundell, C., Bengio, Y., and Mozer, M. Object files and schemata: Factorizing declarative and procedural knowledge in dynamical systems. *arXiv preprint arXiv:2006.16225*, 2020a.

Goyal, A., Lamb, A., Hoffmann, J., Sodhani, S., Levine, S., Bengio, Y., and Schölkopf, B. Recurrent independent mechanisms. *ICLR*, 2020b.

Graves, A., Wayne, G., and Danihelka, I. Neural turing machines. *arXiv preprint arXiv:1410.5401*, 2014.

Greff, K., van Steenkiste, S., and Schmidhuber, J. On the binding problem in artificial neural networks. *arXiv preprint arXiv:2012.05208*, 2020.

He, K., Fan, H., Wu, Y., Xie, S., and Girshick, R. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738, 2020.

Heess, N., Hunt, J. J., Lillicrap, T. P., and Silver, D. Memory-based control with recurrent neural networks. *arXiv preprint arXiv:1512.04455*, 2015.

Hessel, M., Modayil, J., Hasselt, H. V., Schaul, T., Ostrovski, G., Dabney, W., Horgan, D., Piot, B., Azar, M. G., and Silver, D. Rainbow: Combining improvements in deep reinforcement learning. In *AAAI*, 2018.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Hu, J., Shen, L., and Sun, G. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018.

Jaderberg, M., Mnih, V., Czarnecki, W., Schaul, T., Leibo, J. Z., Silver, D., and Kavukcuoglu, K. Reinforcement learning with unsupervised auxiliary tasks. *ArXiv*, abs/1611.05397, 2017.

Kansky, K., Silver, T., Mély, D. A., Eldawy, M., Lázaro-Gredilla, M., Lou, X., Dorfman, N., Sidor, S., Phoenix, S., and George, D. Schema networks: Zero-shot transfer with a generative causal model of intuitive physics. In *International Conference on Machine Learning*, pp. 1809–1818. PMLR, 2017.

Kawaguchi, K., Kaelbling, L. P., and Bengio, Y. Generalization in deep learning. *arXiv preprint arXiv:1710.05468*, 2017.

Kipf, T., van der Pol, E., and Welling, M. Contrastive learning of structured world models. *arXiv preprint arXiv:1911.12247*, 2019.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25: 1097–1105, 2012.

Lake, B. and Baroni, M. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International Conference on Machine Learning*, pp. 2873–2882. PMLR, 2018.

Lampinen, A. K., Chan, S. C., Banino, A., and Hill, F. Towards mental time travel: a hierarchical memory for reinforcement learning agents. *arXiv preprint arXiv:2105.14039*, 2021.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Lu, X., Van Roy, B., Dwaracherla, V., Ibrahimi, M., Osband, I., and Wen, Z. Reinforcement learning, bit by bit. *arXiv preprint arXiv:2103.04047*, 2021.

Minsky, M. *A framework for representing knowledge*. De Gruyter, 1979.

Mirowski, P., Pascanu, R., Viola, F., Soyer, H., Ballard, A., Banino, A., Denil, M., Goroshin, R., Sifre, L., Kavukcuoglu, K., Kumaran, D., and Hadsell, R. Learning to navigate in complex environments. *ArXiv*, abs/1611.03673, 2017.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533, 2015.

Parisotto, E., Song, F., Rae, J., Pascanu, R., Gulcehre, C., Jayakumar, S., Jaderberg, M., Kaufman, R. L., Clark, A., Noury, S., et al. Stabilizing transformers for reinforcement learning. In *International Conference on Machine Learning*, pp. 7487–7498. PMLR, 2020.

Pritzel, A., Uria, B., Srinivasan, S., Badia, A. P., Vinyals, O., Hassabis, D., Wierstra, D., and Blundell, C. Neural episodic control. In *International Conference on Machine Learning*, pp. 2827–2836. PMLR, 2017.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. Zero-shot text-to-image generation. *arXiv preprint arXiv:2102.12092*, 2021.

Ravfogel, S., Goldberg, Y., and Linzen, T. Studying the inductive biases of rnns with synthetic variations of natural languages. *arXiv preprint arXiv:1903.06400*, 2019.

Sallab, A. E., Abdou, M., Perot, E., and Yogamani, S. Deep reinforcement learning framework for autonomous driving. *ArXiv*, abs/1704.02532, 2017.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Shi, X., Chen, Z., Wang, H., Yeung, D.-Y., Wong, W.-K., and Woo, W.-c. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28, 2015.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

Sorokin, I., Seleznev, A., Pavlov, M., Fedorov, A., and Ignateva, A. Deep attention recurrent q-network. *arXiv preprint arXiv:1512.01693*, 2015.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575 (7782):350–354, 2019.

Wang, J. X., Kurth-Nelson, Z., Soyer, H., Leibo, J. Z., Tirumala, D., Munos, R., Blundell, C., Kumaran, D., and Botvinick, M. Learning to reinforcement learn. *ArXiv*, abs/1611.05763, 2017.

Witty, S., Lee, J. K., Tosch, E., Atrey, A., Littman, M., and Jensen, D. Measuring and characterizing generalization in deep reinforcement learning. *arXiv preprint arXiv:1812.02868*, 2018.

Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. How transferable are features in deep neural networks? *arXiv preprint arXiv:1411.1792*, 2014.