ACCELERATING ONLINE REINFORCEMENT LEARNING VIA MODEL-BASED META-LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Current reinforcement learning algorithms struggle to quickly adapt to new situations without large amounts of experience and usually without large amounts of optimization over that experience. In this work we seek to leverage meta-learning methods from MAML with model-based RL methods based on MuZero to design agents which can quickly adapt online. We propose a new model-based meta-RL algorithm that can adapt online to new experience and can be meta-trained without explicit task labels. Compared to prior model-based meta-learning methods, our work can scale to more visually complex image based environments with dynamics that change significantly over time and can handle the continual RL setting which has no episodic boundaries.

1 INTRODUCTION

In the real world, humans and animals, continually adapt to non-stationary environments that change significantly over time due to other agents (Lowe et al., 2017; Foerster et al., 2017), environment complexity (Sutton et al., 2007), or changes in dynamics or goals (Thrun, 1998). For example, an animal that migrates between continents must deal with altering terrarian, day / night cycles, varying weather patterns, and the presence of other interacting animals. This is depicted in Figure 1 where an agent must navigate a sequence of vary different environments over its lifetime which can be viewed as a sequence of tasks. In contrast, the traditional reinforcement learning setting is evaluated in stationary, single-task environments where an agent repeatedly attempts the same task, resetting each time to a new episode. If we want our agents to handle the non-stationary continual learning case, then we will need algorithms that can rapidly adapt to the next task as new experience comes online.

Meta-learning or *learning to learn* has become a promising method for (meta) training agents that can efficiently adapt to new tasks, environments, or dynamics with only a few samples. However, current meta-reinforcement learning methods do not support the life-long learning setting, where the agent must simultaneously adapt to new tasks *and* meta-train on all tasks seen so far, all from streaming non-episodic experience and without any control over which task to attempt next. In this setting, the streaming experience is not segmented into clear task boundaries, resets are unavailable, and the task distribution is continually changing as the agent progresses through the environment. In such settings, we need a meta-learning method that can both master each new challenge, and meta-learn so as to master each challenge *after that* more quickly and efficiently. To address this problem setting, we propose a MAML-based (Finn et al., 2017) meta-RL method for continual meta-learning.

In a continual learning environment, the agent does not have the ability to explicitly collect data for each task. An agent that can meta-train using off-policy data collected on previous tasks can be more efficient by training over all its experience. However, the MAML formulation of meta-RL training commonly uses policy gradient methods which require on-policy data. To support off-policy training, we develop a model-based RL algorithm that is trained using MAML where each task will be a slice in time of the current agent's experience.

MAML for model-based RL has been successful on low-dimensional tasks (Nagabandi et al., 2019b;a) with limited horizons but has not scaled to more complex image based domains where

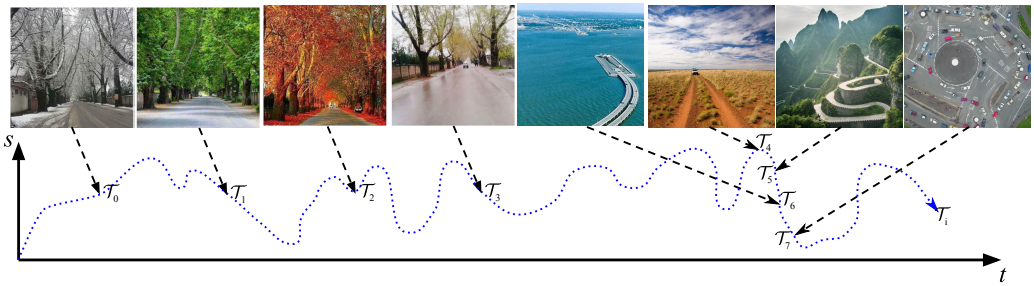

Figure 1: Here we show that as an agent interacts with the world over a lifetime (blue line) it gains diverse experience across tasks $\mathcal{T}_i$ in the environment. To adapt online to environmental changes and achieve high average reward on $\mathcal{T}_i$, the agent needs to learn how to generalize across the many conditions using off-policy experience.

a learned dynamics model would need to reconstruct image observations. Our model will be based off recent work, GCG and MuZero (Kahn et al., 2018; Schrittwieser et al., 2020) which learns a latent dynamics model that predicts a future sequence of states in latent space and a future sequence of rewards from these latents. This avoids the computational cost of image reconstruction while still enabling the model to quickly adapt to new situations.

Our method is evaluated over a collection of environments that change as the agent advances through the world. As a proof of concept, our first environment is a procedural maze which increases in difficutly over time. These environments do not include clear task boundaries or control over the task distribution and will be non-episodic. The agent must continually learn over a single infinitely long episode. We evaluate our method in terms of overall learning speed: instead of measuring the final reward when adapting to new test tasks, we measure how long the algorithm takes to master an entire game when starting from scratch. This evaluation illustrates the importance of a lifelong meta-RL setup: in the lifelong setting, meta-RL can effectively enable an agent to solve the entire task (e.g., beat the game) faster than a non-meta-learning-based method, since it continually accelerates the agent's ability to adapt to each new concept or feature.

2 Related Work

Lifelong learning is a well-known field with a long-standing background (Bottou, 1998; Thrun, 1998; Jafari et al., 2001; Parisi et al., 2019; Khetarpal et al., 2020). One class of lifelong learning method focuses on the issue of catastrophic forgetting (Kirkpatrick et al., 2017; Rebuffi et al., 2017; Zenke et al., 2017; Lopez-Paz & Ranzato, 2017; Nguyen et al., 2017). In our work, we are instead focusing on the problem of transferring knowledge forward to make learning new tasks more efficient (Rusu et al., 2015; Aljundi et al., 2019; Wang et al., 2017). Methods range from probabilistic filtering (Murphy & Russell, 2002; Hoffman et al., 2010; Broderick et al., 2013) to online gradient updates (Duchi et al., 2011). While gradient-based SGD can be used for online learning (Bottou, 1998), its performance is limited with deep neural networks (Sahoo et al., 2018), which work best when trained over large batches of data. In this work, we avoid this limitation by designing an online meta-RL method that learns to adapt to new tasks given only a small window of new data on the current task. This enables the agent to perform best on the most recent data while using previous experience to enable overall generalization skills.

Our methods build off of meta-reinforcement learning methods that train over a fixed set of well-defined tasks that are sampled i.i.d. from user-provided meta-training distributions (Santoro et al., 2016; Ravi & Larochelle, 2017; Munkhdalai & Yu, 2017; Wang et al., 2016; Duan et al., 2016). However, Our work neither assumes a fixed distribution of tasks or known task boundaries. Recent methods do support meta-learning without well-defined task boundaries, but they have only been shown to work on supervised meta-training problems (Harrison et al., 2020; Nagabandi et al., 2019b). Other recent work uses meta-learning to solve tasks sequentially and in an online fashion, but is not designed to support reinforcement learning tasks (Finn et al., 2019). Meta-reinforement learning methods have also considered non-stationarity within a short task (Al-Shedivat et al., 2018) and episodes involving multiple tasks at meta-test time (Ritter et al., 2018), but they do not consider continual online adaptation with unknown task boundaries. The most related work to our own per-

forms online adaptation using model-based meta-reinforcement learning on robotic tasks from true pose information and trains this model in an offline fashion by sampling task i.i.d (Nagabandi et al., 2019b). Instead, our method acts and trains in an online fashion from images without control over the task distribution.

3 Problem Setup and Preliminaries

In this section we introduce the problem setup and preliminaries.

Online Continual Reinforcement Learning In episodic RL, we learn a policy that maximizes the expected discounted return over a finite horizon H: $J_{\text{episodic}}(\pi) = V_\pi(s) = \mathbb{E}_\pi \left[\sum_{k=0}^{H-1} \gamma^k R_{t+k} | s_t = s\right]$. We can reset the agent to some initial state distribution $s_0 \sim \rho_0$ after an episode terminates and draw samples from this objective to evaluate the performance of the policy. This is in contrast to continual RL, where at each point in time we would like to maximize: $J_{\text{lifelong}}(\pi) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k R_{t+k} | s_t = s\right]$. In this setting, we cannot reset the agent to some start state distribution and so we cannot draw many samples to estimate policy performance. We will instead use the average reward up to that point in time T of the agent's lifetime: $J_{\text{average}}(\pi, T) = \frac{\sum_t^T R_t}{T}$ to measure performance. Importantly, as the agent progresses in a lifelong environment, the dynamics or reward function change, creating a nonstationary MDP. These changes can be due to natural shifts in the environment such as altering terrain or weather. For the agent to receive high average reward it must be able to quickly learn from its most recent experience and adjust to recent changes in the MDP.

MAML Meta-learning is used to automatically learn learning algorithms that are more efficient than learning from scratch. These methods use data from previous tasks to acquire a learning procedure that can quickly adapt to new tasks. It is assumed that the meta-training and meta-test tasks come from the same task distribution $\rho(\mathcal{T})$ and share common structure that can be exploited for fast learning. In the supervised learning setting, we learn a function f_θ with parameters θ that minimizes a loss function $\mathcal{L}_\mathcal{T}$. The goal of meta-learning is to find a learning procedure which produces, $\theta^* = U(\mathcal{T}^{tr}, \theta')$ that can generalize quickly to a new task $\mathcal{T}^{ts}$ given a set of training tasks $\mathcal{T}^{tr}$. In MAML, an inner learning procedure performs gradient descent on the loss function, $U(\mathcal{T}^{tr}, \theta) = \theta - \alpha \nabla_\theta \mathcal{L}(\mathcal{T}^{tr}, \theta)$. The full MAML objective is then $\min_\theta \mathbb{E}_\mathcal{T} \mathcal{L}(\mathcal{T}^{ts}, U(\mathcal{T}^{tr}, \theta))$. During test time adaptation, we then take the meta-learned parameters θ' and adapt it to new data.

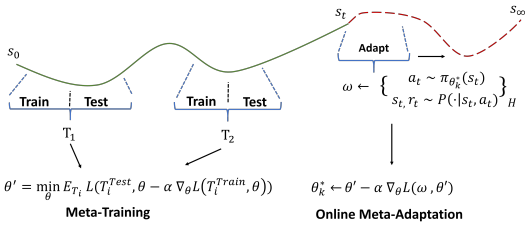

Figure 2: Method Overview. The agent experiences a single stream of experience starting from s_0 and is currently at s_t. During meta-training, it's current lifetime experience (in green) is split into training tasks where each task corresponds to a different window of time. The window is split in half to form training and test data to meta-learn an initial θ such that optimizing the loss on the first half of the window will result in low loss on the 2nd half. During online meta-adaptation, θ is adapted with the most recent window ω of experience to learn $\theta*$ which should perform well on any new experience.

4 LifeLong Learning via Model-Based Meta-Reinforcement Learning

Our agent is parametrized by the policy $\pi_\theta(s_t)$ will experience a single stream of experience from a nonstationary MDP. We denote this experience up the current point in time T with $\tau_{0:T} = (s_0, a_0, r_0, ..., s_T, a_T, r_T)$. Let ω be the most recent window of online experience up to the last H timesteps. Given $\tau_{0:T}$, the goal is to meta-learn parameters θ' such that online adaptation of θ' over ω with the procedure $\theta^* = U(w, \theta')$ will create a policy θ^* with high average return $J_{\text{average}}(\pi, T)$. In the continual setting, we alternate between meta-training and adapting to the newest experience ω which then gets added to the dataset $\mathcal{D}$ for meta-training. This is summarized in Algorithm 1.

In order to create tasks for our meta-learning method we will use slices or windows of the agent's experience $\tau_{0:T}$ as separate tasks. So task $\mathcal{T}_i$ will be a slice $\tau_{i-M:i+M}$ of length $2M$. The first M points of $\tau_{i-M:i+M}$ are used for meta-training training, $\mathcal{T}_i^{tr} = \tau_{i-M:i}$ while the second M points are used for meta-training testing, $\mathcal{T}_i^{ts} = \tau_{i:i+M}$. The intuition is that we want our agent to adapt with it's most recent experience to handle the current context. So for each of these windows, the first half will correspond to the data the agent is adapting to, and the second half is the future experience we want the adapted agent to generalize to. See Figure 2 for a visualization.

We use a model based RL method for our policy which allows us to use off-policy data from the agent's lifetime of experience. The model is a latent dynamics model that predicts future latent state and future reward given current observation and future actions. More precisely the model is composed of a dynamics component $p(z_{t:t+H}|s_t, a_{t:t+H})$ and immediate reward prediction module $p(r_t|z_t)$. The model is trained to minimize the negative log likelihood of sequences over the dataset given our model: $\mathcal{L}(\mathcal{D}, \theta) = \mathbb{E}_{\tau_{t:t+H}\sim\mathcal{D}} - \log p_\theta(r_{t:t+H}|s_t, a_{t:t+H})$. We obtain a policy from this model by using MPC which optimizes for the sequence of actions that obtains highest predicted cumulative return over the horizon H.

Algorithm 1 Lifelong Meta-Reinforcement Learning

Initialize θ_0, $D = \{\}, k = 0$
Get first state $s_0 \leftarrow$ `env.reset()`
while $t = 0, \ldots, \infty$ **do**
  $a_t \leftarrow \pi(s_t|\theta_t)$
  $s_{t+1}, r_t \sim P(\cdot|s_t, a_t)$
  $D \leftarrow D \bigcup \{s_t, a_t, r_t\}$
  Meta train θ using $\mathcal{T}_i \sim D$
  $\min_\theta \mathbb{E}_{\mathcal{T}_i} \mathcal{L}(\mathcal{T}_i^{ts}, \theta - \alpha \nabla_\theta \mathcal{L}(\mathcal{T}_i^{tr}, \theta))$
  Adapt online with $\omega_t \leftarrow D_{t-H:t}$
  $\theta_{t+1} \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}(\theta, \omega_t)$
end while

Given the loss function of the model, the meta-training objective is then:

$$\min_\theta \mathbb{E}_{\mathcal{T}_i \sim D} \left[\mathcal{L}(\mathcal{T}_i^{ts}, U(\mathcal{T}^{tr}, \theta)) \right] \tag{1}$$

Adaptation occurs online by taking gradient steps over the agent's most recent experience ω_t starting with the meta-learned parameters θ: $\theta_k \leftarrow \theta - \alpha \nabla_\theta \mathcal{L}(\theta, \omega_t)$.

5 EXPERIMENTS

We evaluate the proposed method (Meta-GCG) on its ability to adapt online and achieve higher rewards compared to a version of the method that does not use meta-learning (GCG). The architecture for the model is composed of an observation encoder $z_t = f(o_{t-K:t})$ which encodes the last K observations (64x64 grayscale images) with a CNN. A dynamics model through a Conv GRU, predict future latent states given an action and encoded observation: $z_{t+1} = g(z_t, a_t)$. A reward model predicts the immediate reward from the predicted latent states $r_{t+1} = h(z_{t+1})$. Architecture and training details are described further in Appendix A.

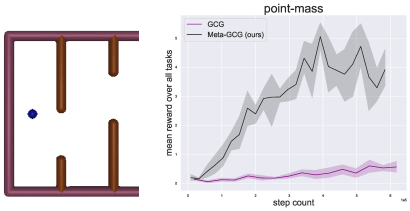

Figure 3: **Image-Based-Maze** shown on the left. The evaluation on the right shows Meta-GCG acheives higher rewards on average while learning on the procedural maze environment.

Environment. The evaluation is performed on an **Image-Based-Maze** where the objective is to navigate to the right as fast as possible while the distribution of obstacles and environment dynamics change over the course of the maze. Our current evaluation is still only episodic where the task distribution is explicitly defined. In this case, a task corresponds to a random rotation of the agent's actions by $0, 90, 180$, and 270 degrees. The results of the evaluation are shown in Figure 3 where Meta-GCG shows a marked improvement in average reward over GCG. We leave a non-episdoic evaluation for further work.

The method in this paper has indicated that we can use meta-learning to assist online adaptation. This adaptation has worked well in an environment with large variation. In the future we would like to apply the method to more open worlds with additional variation such as MineCraft and MarioKart.

REFERENCES

Maruan Al-Shedivat, Trapit Bansal, Yura Burda, Ilya Sutskever, Igor Mordatch, and Pieter Abbeel. Continuous adaptation via meta-learning in nonstationary and competitive environments. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=Sk2u1g-0-.

Rahaf Aljundi, Klaas Kelchtermans, and Tinne Tuytelaars. Task-free continual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 11254–11263, 2019.

Léon Bottou. Online learning and stochastic approximations. *On-line learning in neural networks*, 17(9):142, 1998.

Tamara Broderick, Nicholas Boyd, Andre Wibisono, Ashia C Wilson, and Michael I Jordan. Streaming variational bayes. In *Advances in neural information processing systems*, pp. 1727–1735, 2013.

Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 1126–1135. JMLR. org, 2017.

Chelsea Finn, Aravind Rajeswaran, Sham Kakade, and Sergey Levine. Online meta-learning. In *International Conference on Machine Learning*, pp. 1920–1930, 2019.

Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. *arXiv preprint arXiv:1705.08926*, 2017.

J. Harrison, Apoorva Sharma, Chelsea Finn, and M. Pavone. Continuous meta-learning without tasks. *ArXiv*, abs/1912.08866, 2020.

Matthew Hoffman, Francis R Bach, and David M Blei. Online learning for latent dirichlet allocation. In *advances in neural information processing systems*, pp. 856–864, 2010.

Amir Jafari, Amy Greenwald, David Gondek, and Gunes Ercal. On no-regret learning, fictitious play, and nash equilibrium. In *ICML*, volume 1, pp. 226–233, 2001.

Gregory Kahn, Adam Villaflor, Bosen Ding, Pieter Abbeel, and Sergey Levine. Self-supervised deep reinforcement learning with generalized computation graphs for robot navigation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1–8. IEEE, 2018.

Khimya Khetarpal, Matthew Riemer, Irina Rish, and Doina Precup. Towards continual reinforcement learning: A review and perspectives. *arXiv preprint arXiv:2012.13490*, 2020.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in neural information processing systems*, pp. 6467–6476, 2017.

Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in neural information processing systems*, pp. 6379–6390, 2017.

Tsendsuren Munkhdalai and Hong Yu. Meta networks. *Proceedings of machine learning research*, 70:2554, 2017.

Kevin Patrick Murphy and Stuart Russell. Dynamic bayesian networks: representation, inference and learning. 2002.

Anusha Nagabandi, I. Clavera, Simin Liu, Ronald S. Fearing, P. Abbeel, S. Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv: Learning*, 2019a.

Anusha Nagabandi, Chelsea Finn, and Sergey Levine. Deep online learning via meta-learning: Continual adaptation for model-based RL. *7th International Conference on Learning Representations, ICLR 2019*, pp. 1–15, 2019b.

Cuong V Nguyen, Yingzhen Li, Thang D Bui, and Richard E Turner. Variational continual learning. *arXiv preprint arXiv:1710.10628*, 2017.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. 2017.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 2001–2010, 2017.

Samuel Ritter, Jane Wang, Zeb Kurth-Nelson, Siddhant Jayakumar, Charles Blundell, Razvan Pascanu, and Matthew Botvinick. Been there, done that: Meta-learning with episodic recall. In *International Conference on Machine Learning*, pp. 4354–4363, 2018.

Andrei A Rusu, Sergio Gomez Colmenarejo, Caglar Gulcehre, Guillaume Desjardins, James Kirkpatrick, Razvan Pascanu, Volodymyr Mnih, Koray Kavukcuoglu, and Raia Hadsell. Policy distillation. *arXiv preprint arXiv:1511.06295*, 2015.

Doyen Sahoo, Quang Pham, Jing Lu, and Steven CH Hoi. Online deep learning: learning deep neural networks on the fly. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pp. 2660–2666, 2018.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International conference on machine learning*, pp. 1842–1850, 2016.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

Richard S Sutton, Anna Koop, and David Silver. On the role of tracking in stationary environments. In *Proceedings of the 24th international conference on Machine learning*, pp. 871–878, 2007.

Sebastian Thrun. Lifelong learning algorithms. In *Learning to learn*, pp. 181–209. Springer, 1998.

Jane X. Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z. Leibo, Rémi Munos, Charles Blundell, Dharshan Kumaran, and Matthew Botvinick. Learning to reinforcement learn. *CoRR*, abs/1611.05763, 2016. URL `http://arxiv.org/abs/1611.05RL 763`.

Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. Growing a brain: Fine-tuning by increasing model capacity. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2471–2480, 2017.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. *Proceedings of machine learning research*, 70:3987, 2017.

A EXPERIMENT DETAILS

For our experiments we use the following hyperparameters:

Outer optimizer	Adam
Outer learning rate	1e-4
Inner Optimizer	SGD
Inner learning rate	1e-2
Inner update steps	3
Meta-batch size	8
Batch size	32
MPC Horizon	5
Train Adaptation Window Size	400
Test Adaptation Window Size	10

During training the window size is 400 and is split in half into train / test splits. During test time adaptation, we alternate between collecting 10 environment steps and adapting to the last 10 steps.

Architecture: The architecture for the model is composed of an observation encoder $z_t = f(o_{t-K:t})$ which encodes the last K observations which are 64x64 grayscale images. A dynamics model predict future latent states given an action $z_{t+1} = g(z_t, a_t)$. A reward model predicts the immediate reward from the predicted latent states $r_{t+1} = h(z_{t+1})$. The observation encoder and reward prediction model are CNNs and the dynamics model is a Conv GRU. Network architecture details are listed below.

Model	Conv layers	Kernel Size	Hidden Sizes	Input Size	Output Size
Observation Encoder	32, 64, 64, 128	3, 3, 3, 2		64x64xK	6x6x128
Dynamics Model	128	3	128	6x6x128, $\lvert A \rvert$	7x7x64
Reward Predictor	64	2	128,128	7x7x64	41

Hidden size for the dynamics model means number of channels of the hidden state. The initial hidden state of the dynamics model is initialized with the encoded observation. The model is trained over sequences of length 5. Target reward values are binned into one of 41 bins and the model is trained with cross entropy loss. We use ELU activations after every non-output layer.

MPC with Learned Model: To obtain a policy from the model, we use MPC with CEM. Each CEM round, we sample 500 sequences of random actions, score each sequence with the sum of predicted rewards from the model, and use the top 10% of actions to create the next multinomial distribution to sample actions from for the next round of CEM. We use 4 rounds of CEM.