Supplementary Material

A. Experimental Details

A.1. Environment

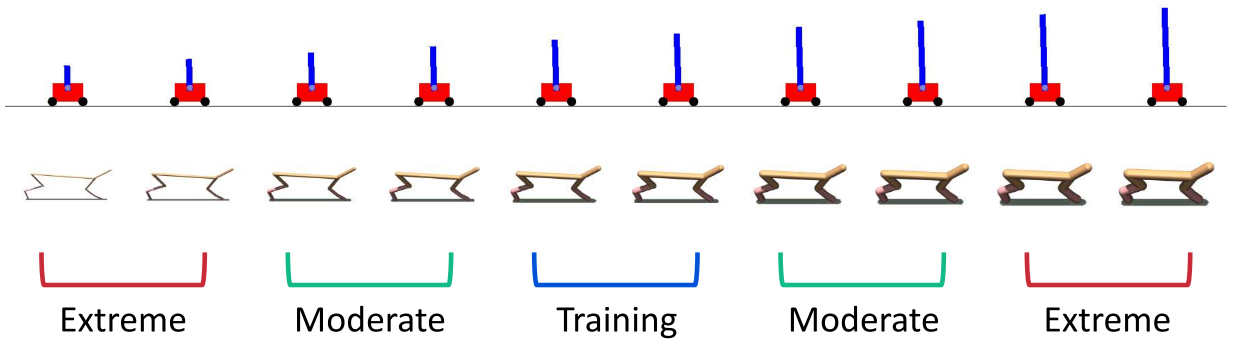


Figure 8. Illustration of training and test environments.

CartPole. CartPole consists of a pushable cart and a pole attached on top of it by an unactuated joint. The goal is to push the cart left and right, so that the pole does not fall over.

- Observation. $(x_t, \dot{x}_t, \theta_t, \dot{\theta}_t)$, where x is the cart position, and θ is the pole's angle from the upright position.
- Action. $\{0, 1\}$, where 0 and 1 correspond to pushing the cart left and right, respectively.
- Reward. $r_t = \mathbb{1}_{\{|x_{t+1}|<2.4 \wedge |\theta_{t+1}|<2\pi \cdot (\frac{14}{360})\}}$. That is, reward is 1 if the cart position is within ± 2.4 and pole angle is within $\pm 14^\circ$ during the next timestep, and 0 otherwise.
- Modifications. In our experiments, we modify push force f and pole length l (see Table 3).

Pendulum. Pendulum consists of a single rigid body, one end of which is attached to an actuated joint. The goal is to swing up the body and keep it in the upright position.

- Observation. $(\cos\theta, \sin\theta, \dot{\theta})$, where $\theta \in [-\pi, \pi]$ is zero in the upright position, and increases in the counter-clockwise direction.
- Action. $a \in [-2.0, 2.0]$ represents counter-clockwise torque applied to the pendulum body.
- Reward. $r_t = -(\theta_t^2 + 0.1\dot{\theta}_t^2 + 0.001 a_t^2)$, which penalises 1) deviation from the upright position, 2) non-zero angular velocity, and 3) joint actuation.
- Modifications. We modify pendulum mass m and pendulum length l (see Table 3).

Half-cheetah.[2]Half-cheetah agent is made up of 7 rigid links (1 for torso, 3 for forelimb, and 3 for hindlimb), connected by 6 joints. The goal is to move forward as fast as possible, while keeping the control cost minimal. We used additional pre-processing for observations as in (Chua et al., 2018).

- Observation. Observation is given by a 20-dimensional vector that includes 1) angular position and velocity of all 6 joints, 2) root joint's position (except for the x-coordinate) and velocity, and 3) center of mass of the torso.
- Action. $a \in [-1.0, 1.0]^6$ represents torques applied at six joints.
- Reward. $r_t = \dot{x}_{\text{torso},t} - 0.05\|a_t\|^2$, where $\dot{x}_{\text{torso},t}$ represents forward velocity of the torso.

	Train		Test (Moderate)		Test (Extreme)		Episode Length
CartPole	$f \in$	$\{5.0, 6.0, 7.0, 8.0, 9.0, 10.0, 11.0, 12.0, 13.0, 14.0, 15.0\}$	$f \in$	$\{3.0, 3.5, 16.5, 17.0\}$	$f \in$	$\{2.0, 2.5, 17.5, 18.0\}$	200
	$l \in$	$\{0.40, 0.45, 0.50, 0.55, 0.60\}$	$l \in$	$\{0.25, 0.30, 0.70, 0.75\}$	$l \in$	$\{0.15, 0.20, 0.80, 0.85\}$	
Pendulum	$m \in$	$\{0.75, 0.80, 0.85, 0.90, 0.95, 1.0, 1.05, 1.10, 1.15, 1.20, 1.25\}$	$m \in$	$\{0.50, 0.70, 1.30, 1.50\}$	$m \in$	$\{0.20, 0.40, 1.60, 1.80\}$	200
	$l \in$	$\{0.75, 0.80, 0.85, 0.90, 0.95, 1.0, 1.05, 1.10, 1.15, 1.20, 1.25\}$	$l \in$	$\{0.50, 0.70, 1.30, 1.50\}$	$l \in$	$\{0.20, 0.40, 1.60, 1.80\}$	
Half-cheetah	$m \in$	$\{0.75, 0.85, 1.0, 1.15, 1.25\}$	$m \in$	$\{0.40, 0.50, 1.50, 1.60\}$	$m \in$	$\{0.20, 0.30, 1.70, 1.80\}$	1000
	$d \in$	$\{0.75, 0.85, 1.0, 1.15, 1.25\}$	$d \in$	$\{0.40, 0.50, 1.50, 1.60\}$	$d \in$	$\{0.20, 0.30, 1.70, 1.80\}$	
Ant	$m \in$	$\{0.85, 0.90, 0.95, 1.0\}$	$m \in$	$\{0.20, 0.25, 0.30, 0.35, 0.40\}$	$m \in$	$\{0.40, 0.45, 0.50, 0.55, 0.60\}$	1000
SlimHumanoid	$m \in$	$\{0.80, 0.90, 1.0, 1.15, 1.25\}$	$m \in$	$\{0.60, 0.70, 1.50, 1.60\}$	$m \in$	$\{0.40, 0.50, 1.70, 1.80\}$	1000
	$d \in$	$\{0.80, 0.90, 1.0, 1.15, 1.25\}$	$d \in$	$\{0.60, 0.70, 1.50, 1.60\}$	$d \in$	$\{0.40, 0.50, 1.70, 1.80\}$	

Table 3. Environment parameters used for our experiments.

- Modification. As for dynamics modification, we 1) scale mass of every rigid link by a fixed scale factor m, and 2) scale damping of every joint by a fixed scale factor d (see Table 3).

Ant.[2] Ant agent consists of 13 rigid links connected by 8 joints. The goal is to move forward as fast as possible at a minimal control cost. We used additional pre-processing for observations as in (Chua et al., 2018).

- Observation. Observation is a 41-dimensional vector that includes 1) angular position and velocity of all 8 joints, 2) position and velocity of the root joint, 3) frame orientations (xmat) and 4) center of mass of the torso.
- Action. $a \in [-1, 1]^8$ represents torques applied at eight joints.
- Reward. $r_t = \dot{x}_{\text{torso},t} - 0.005\|a_t\|^2 + 0.05$, where $\dot{x}_{\text{torso},t}$ denotes forward velocity of the torso.
- Modification. As for dynamics modification, we modify the mass of every leg. Specifically, given a scale factor m, we modify two legs to have mass multiplied by m, and the other two legs to have mass multiplied by $\frac{1}{m}$ (see Table 3).

SlimHumanoid.[2] SlimHumanoid (Wang et al., 2019) consists of 13 rigid links with 17 actuators. The goal is to move forward as fast as possible, while keeping the control cost minimal. We used additional pre-processing for observations as in (Chua et al., 2018).

- Observation. Observation is a 45-dimensional vector that includes angular position and velocities.
- Action. $a \in [-0.4, 0.4]^{17}$.
- Reward. $r_t = 50/3 \times \dot{x}_{\text{torso},t} - 0.1\|a_t\|^2 + 5 \times \text{bool}(1.0 \leq z_{\text{torso},t} \leq 2.0)$, where $\dot{x}_{\text{torso},t}$ denotes forward velocity of the torso and z_t is the height of the torso.
- Modification. As for dynamics modification, we 1) scale mass of every rigid link by a fixed scale factor m, and 2) scale damping of every joint by a fixed scale factor d (see Table 3).

[2]We modified the publicly available code at `https://github.com/iclavera/learning_to_adapt` for implementation of these environments.

A.2. Training Details

Model-based RL. To train the dynamics model, we collect 10 trajectories with MPC controller from environments and train the model for 5 epochs at every iteration. We train the model for 20 iterations for every experiments. To report test performance, we evaluated trained models every iteration on environments with fixed random seeds. The Adam optimizer (Kingma & Ba, 2015) is used with the learning rate 0.001. For planning, we used the cross entropy method (CEM) with 200 candidate actions except for CartPole and Pendulum, where random shooting (RS) method with 1,000 candidate actions is used. The horizon of MPC is 30.

Model-free RL. We trained PPO agents for 5 million timesteps on Pendulum and MuJoCo environments (i.e. Half-cheetah, Ant, Crippled Half-cheetah, Walker) and 0.5 million timesteps on CartPole. We evaluated trained agents every 10,000 timesteps on environments with fixed random seeds. We use a discount factor $\gamma = 0.99$, a generalized advantage estimator (Schulman et al., 2016) parameter $\lambda = 0.95$ and an entropy bonus of 0.01 for exploration. We use 10 rollouts with 200 timesteps per each rollout, and then train agents for 8 epochs with 4 mini-batches. The Adam optimizer is used with the learning rate 0.0005.

A.3. Implementation of Context-aware Dynamics Model

For our method, the context encoder is modeled as multi-layer perceptrons (MLPs) with 3 hidden layers that produce a 10-dimensional vector. Then, CaDM receives the context vector as an additional input, i.e., the input is given as a concatenation of state, action, and context vector. We use $\beta \in \{0.25, 0.5, 1.0\}$ for the penalty parameter in (1), $K \in \{5, 10\}$ for the number of past observations and $M \in \{5, 10\}$ for the number of future observations.

A.4. Implementation of Model-based RL Baselines

Vanilla DM. We model the dynamics model as Gaussian, in which the mean is parameterized by MLPs with 4 hidden layers of 200 units each and Swish activations (Ramachandran et al., 2017) and the variance is fixed. Note that maximum likelihood estimation, i.e., $\log f(s'|s, a)$ corresponds to minimizing the mean squared error in this setup.

Stacked DM. We implemented Stacked DM to take additional 10 observations as an additional input.

GrBAL and ReBAL. We used a reference implementation provided by the authors.[3] For GrBAL, we model the dynamics model as MLPs with 3 layers of 512 units each and ReLU activations. The size of adaptive batch is 16. For ReBAL, we use 1-layer LSTM with 256 cells. We use planning horizon $h \in \{10, 30\}$ and learning rate 0.01.

PE-TS. We implemented PE-TS based on the code provided by the authors.[4] Following the setup in Chua et al. (2018), we used 5 bootstrap models for ensemble and 20 particles for trajectory sampling. We model the dynamics model as MLPs with 4 hidden layers of 200 units each and Swish activations.

A.5. Implementation of Model-free RL Baselines

Stacked PPO. We implemented Stacked PPO to take additional 10 observations as an additional input.

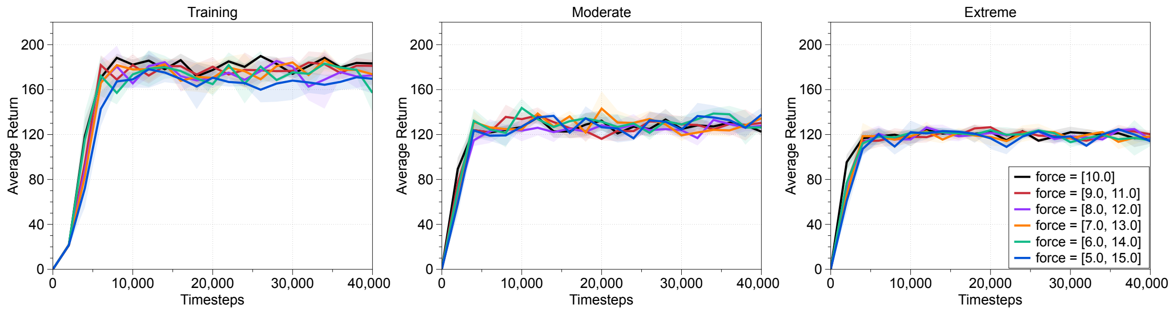
PPO + PC. This baseline is implemented based on Rakelly et al. (2019). We train a PPO agent with probabilistic context encoder that takes $K \in \{5, 10, 15\}$ observations as an additional input. We model the context encoder as the product of independent Gaussian factors, in which the mean and the variance are parameterized by MLPs with 3 hidden layers of (256, 128, 64) units that produce a 10-dimensional vector. Note that this probabilistic context encoder is permutation invariant as in the original paper. For β in KL divergence term from variational lower bound, we select β from $\{0.5, 1.0\}$.

PPO + EP. This baseline is implemented based on Zhou et al. (2019). We model the context encoder as the product of independent Gaussian factors, in which the mean and the variance are parameterized by MLPs with 3 hidden layers of (256, 128, 64) units that produce a 10-dimensional vector. We first train interaction policy using two dynamics models for $\{100\text{K}, 125\text{K}\}$ timesteps, which are the same as the number of samples used for training Vanilla + CaDM, and then train a context-conditional policy.

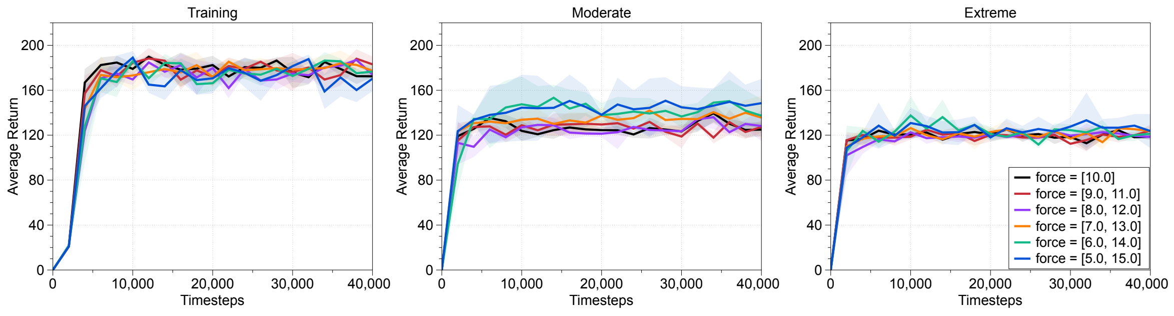
[3]https://github.com/iclavera/learning_to_adapt
[4]https://github.com/kchua/handful-of-trials

B. Effects of Training Environments

In this section, we analyze the effects of training environments on the generalization performance of trained dynamics models. First, we compare the performance of Vanilla DM with CaDM when the range of training environments gets wider. As shown in Figure 9, the generalization performance of Vanilla DM + CaDM on CartPole improves when the model is trained on more wider range of environments while the performance of Vanilla DM does not improve, which implies that our CaDM can indeed utilize contextual information. We also measure the generalization performance of trained Vanilla DM + CaDM on a wide range of simulation parameters of CartPole (see Figure 10). As expected, the test performance degrades as the simulation parameter deviates from training environments (pink shaded area). However, Vanilla DM + CaDM still consistently outperforms Vanilla DM in most simulation parameters.



(a) Vanilla DM



(b) Vanilla DM + CaDM

Figure 9. The performance (average returns) of (a) Vanilla DM and (b) Vanilla DM + CaDM on CartPole environment with varying training ranges. The transition dynamics of environments are changing in both training and test environments. The solid line and shaded regions represent the mean and standard deviation, respectively, across five runs.

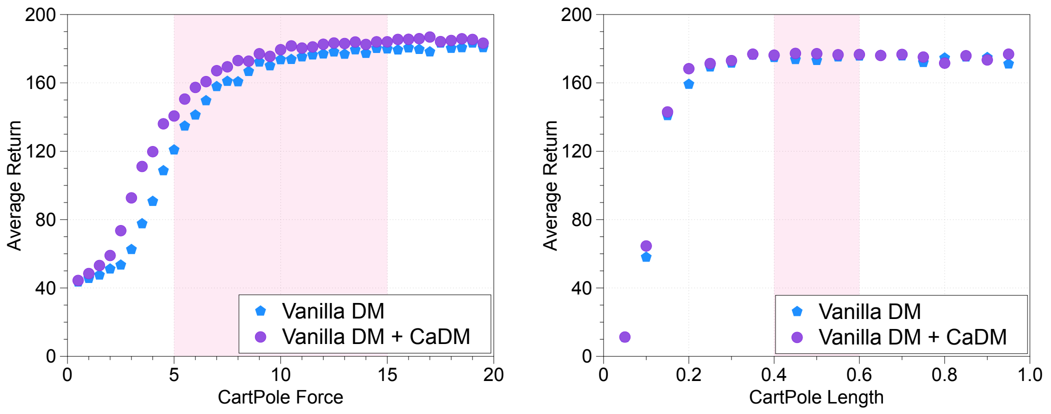
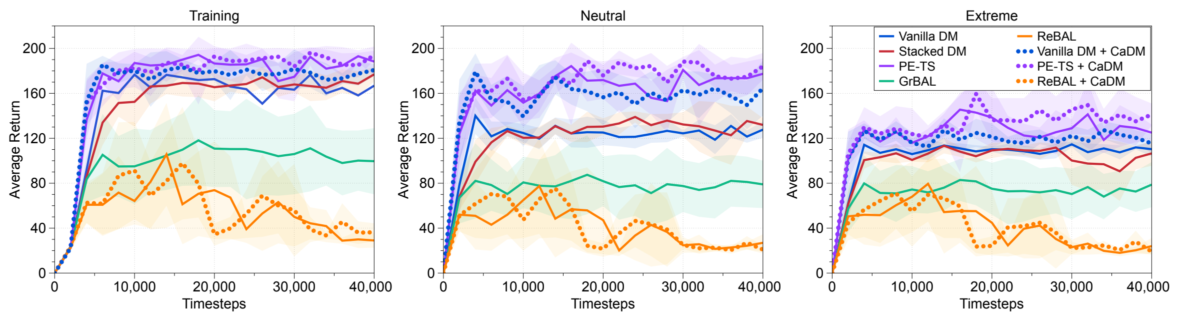
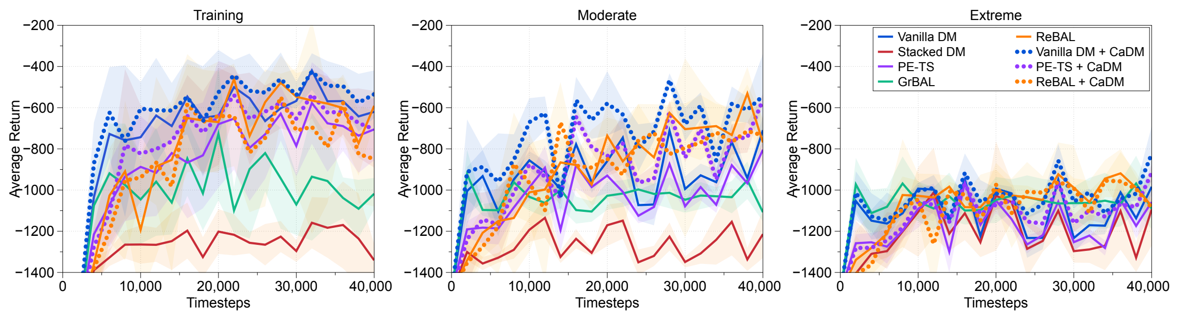


Figure 10. The performance of Vanilla DM and Vanilla DM + CaDM on unseen (moderate) CartPole environments with varying force and length values. The pink shaded area represents the training range. The results show the mean of returns averaged over three runs.

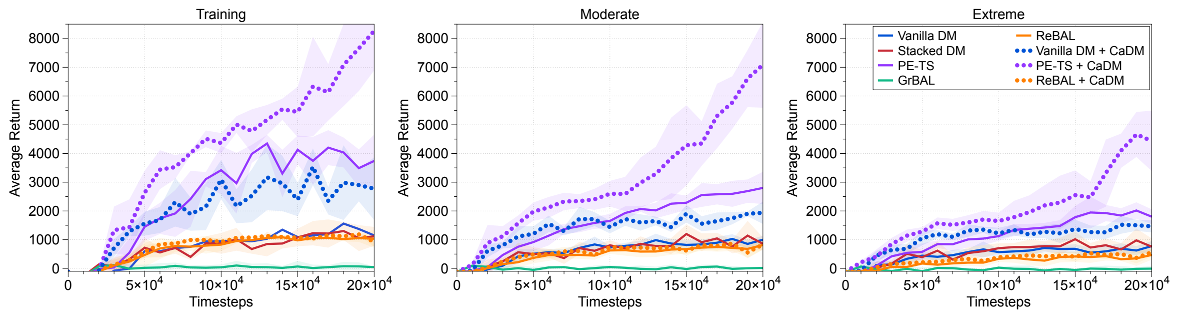
C. Learning Curves for Model-Based RL



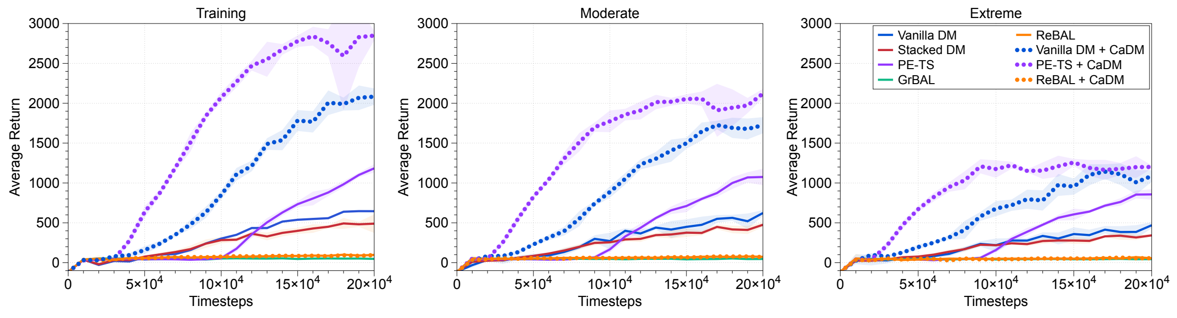
(a) CartPole



(b) Pendulum

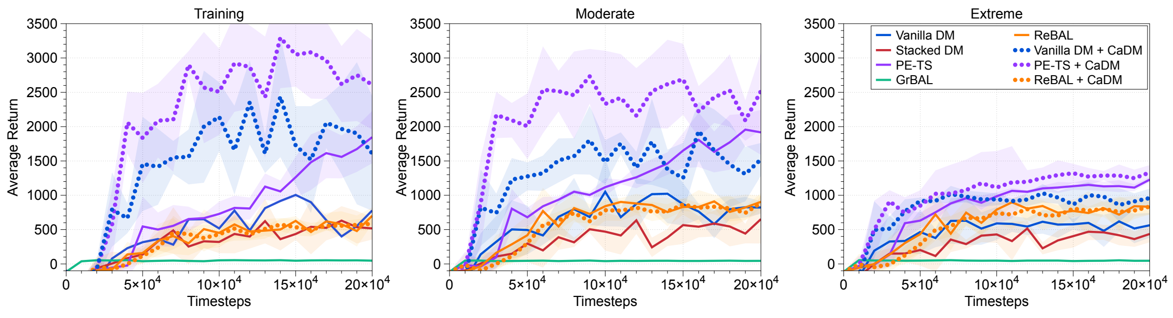


(c) HalfCheetah

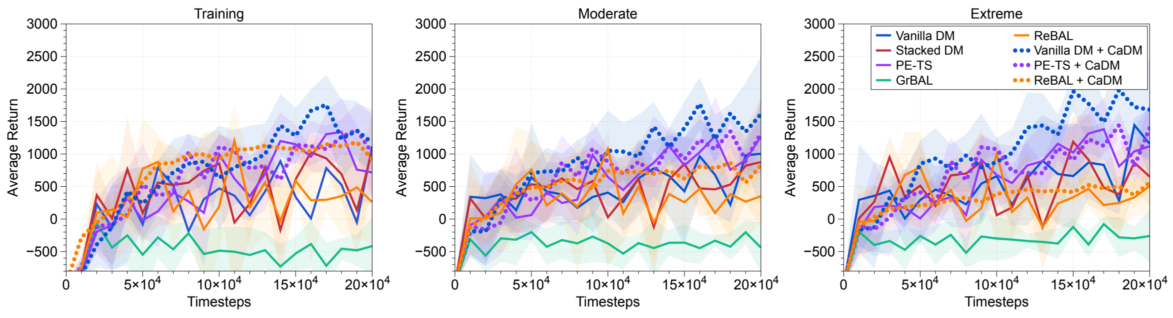


(d) Ant

Figure 11. The performance (average returns) of trained dynamics models on (a) CartPole, (b) Pendulum, (c) HalfCheetah, and (d) Ant. The transition dynamics of environments are changing in both training and test environments. We remark that test environments consist of moderate and extreme environments, where the former draws environment parameters from a closer (yet different) range to the training one, compared to the latter. The solid line and shaded regions represent the mean and standard deviation, respectively, across five runs.



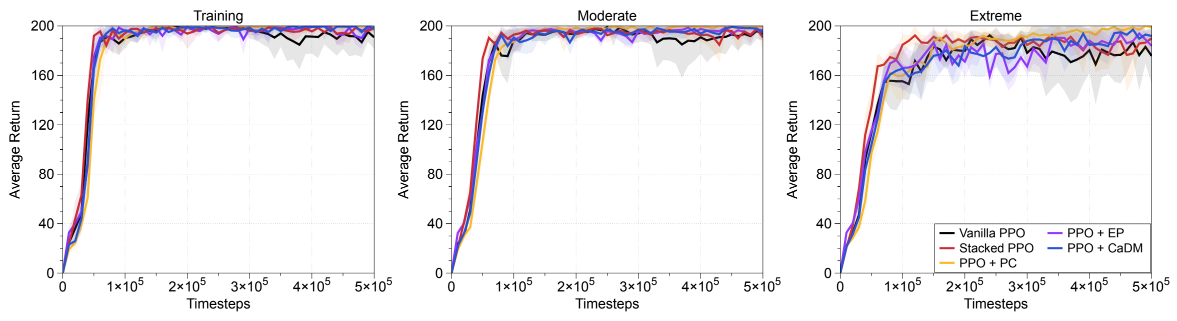
(a) CrippledHalfCheetah



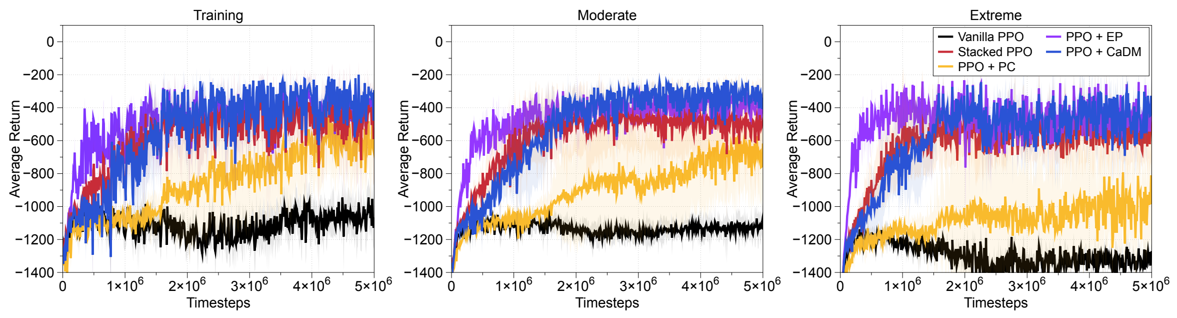
(b) SlimHumanoid

Figure 12. The performance (average returns) of trained dynamics models on (a) CrippledHalfCheetah, and (b) SlimHumanoid. The transition dynamics of environments are changing in both training and test environments. We remark that test environments consist of moderate and extreme environments, where the former draws environment parameters from a closer (yet different) range to the training one, compared to the latter. The solid line and shaded regions represent the mean and standard deviation, respectively, across five runs.

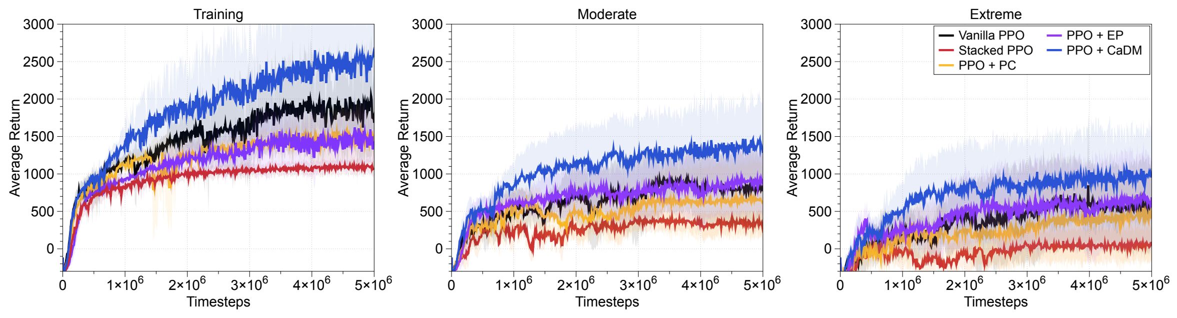
D. Learning Curves for Model-Free RL



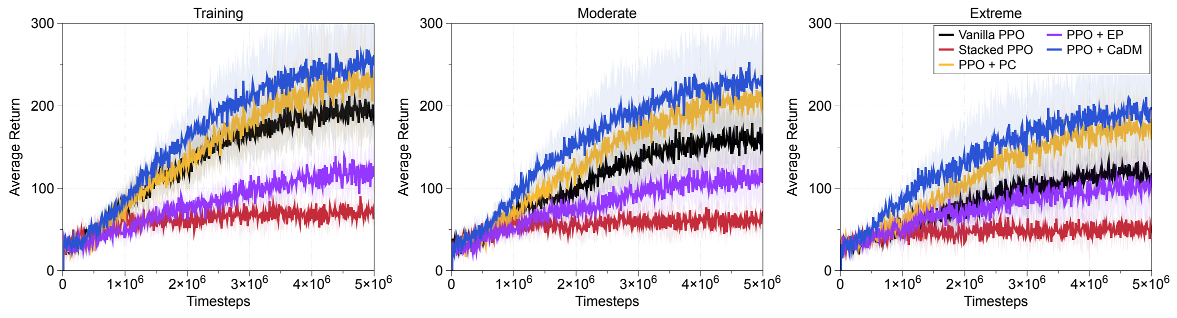
(a) CartPole



(b) Pendulum

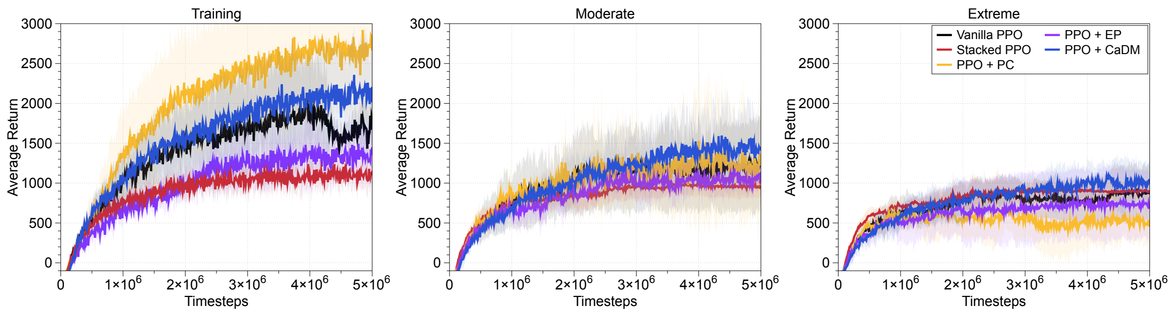


(c) Half-cheetah

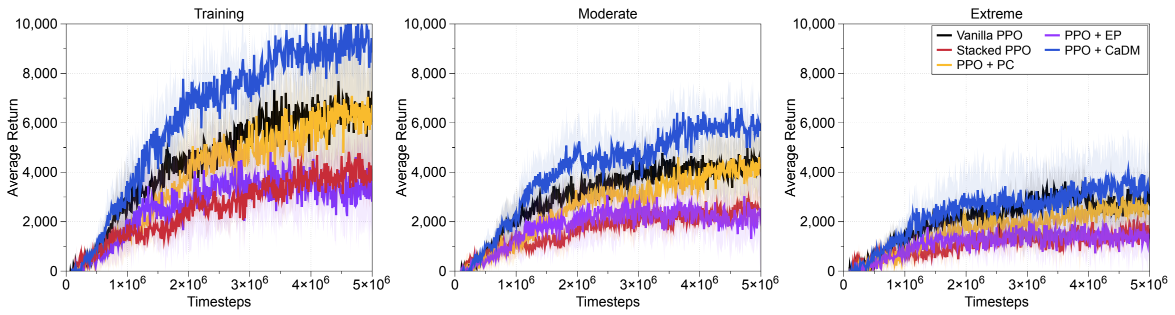


(d) Ant

Figure 13. The performance (average returns) of model-free methods on (a) CartPole, (b) Pendulum, (c) Half-cheetah and (d) Ant. The transition dynamics of environments are changing in both training and test environments. We remark that test environments consist of moderate and extreme environments, where the former draws environment parameters from a closer (yet different) range to the training one, compared to the latter. The solid line and shaded regions represent the mean and standard deviation, respectively, across five runs.



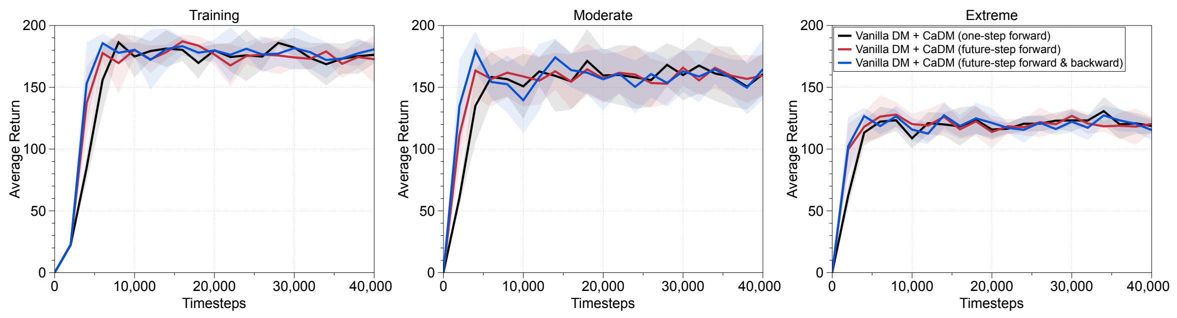
(a) CrippledHalfCheetah



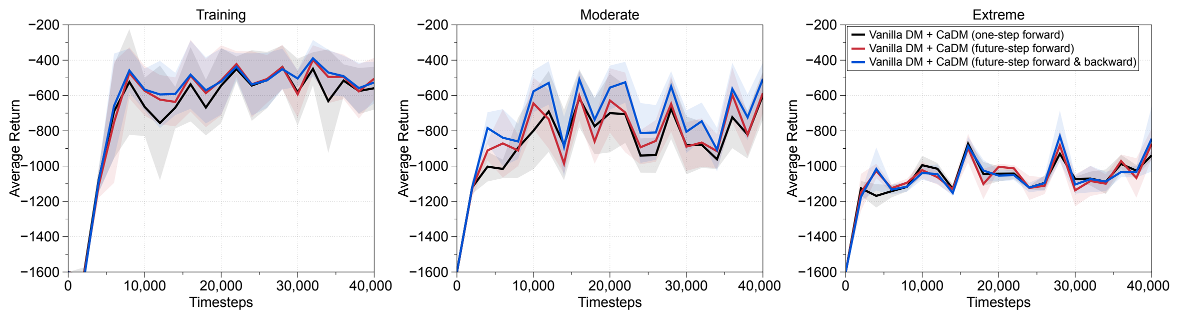
(b) SlimHumanoid

Figure 14. The performance (average returns) of model-free methods on (a) CrippledHalfCheetah, and (b) SlimHumanoid. The transition dynamics of environments are changing in both training and test environments. We remark that test environments consist of moderate and extreme environments, where the former draws environment parameters from a closer (yet different) range to the training one, compared to the latter. The solid line and shaded regions represent the mean and standard deviation, respectively, across five runs.

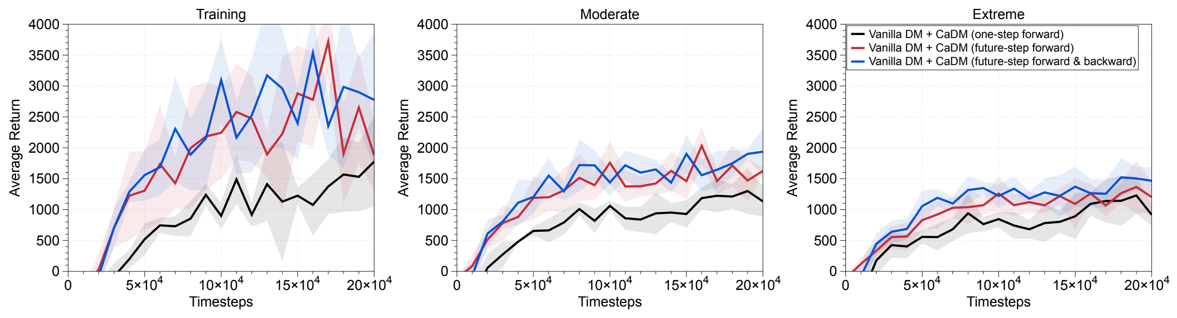
E. Effects of Prediction Loss



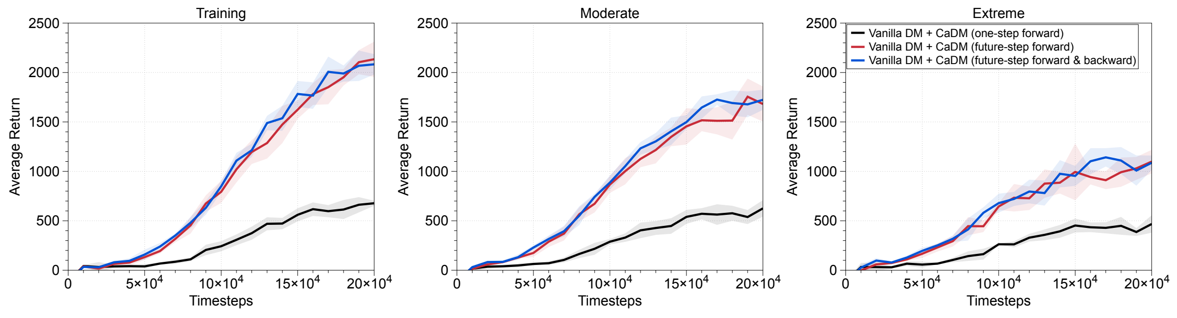
(a) CartPole



(b) Pendulum

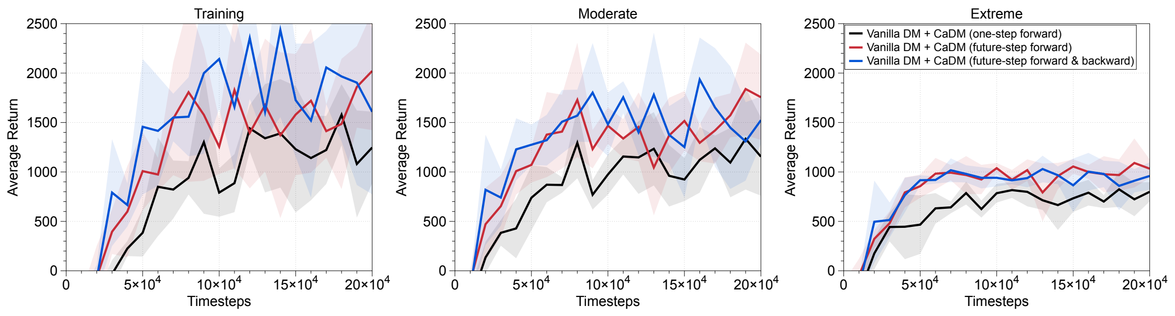


(c) Half-cheetah

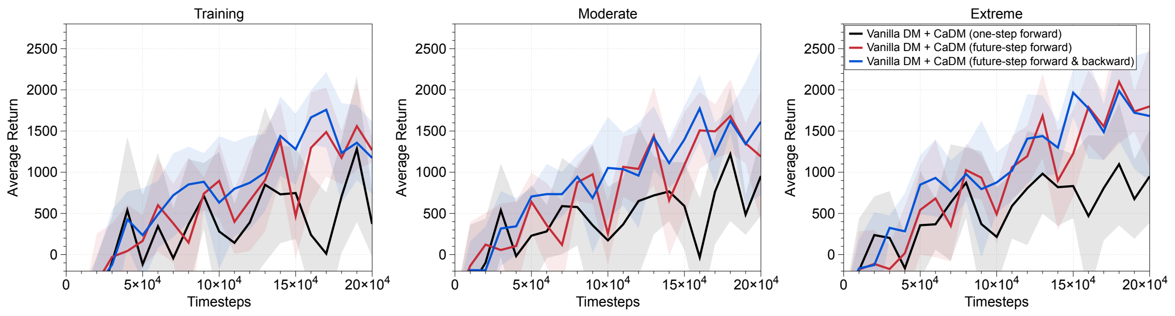


(d) Ant

Figure 15. The performance (average returns) of dynamics models optimized by variants of the proposed prediction objective in (1) on (a) CartPole, (b) Pendulum, (c) HalfCheetah, and (d) Ant. The transition dynamics of environments are changing in both training and test environments. We remark that test environments consist of moderate and extreme environments, where the former draws environment parameters from a closer (yet different) range to the training one, compared to the latter. The solid line and shaded regions represent the mean and standard deviation, respectively, across five runs.



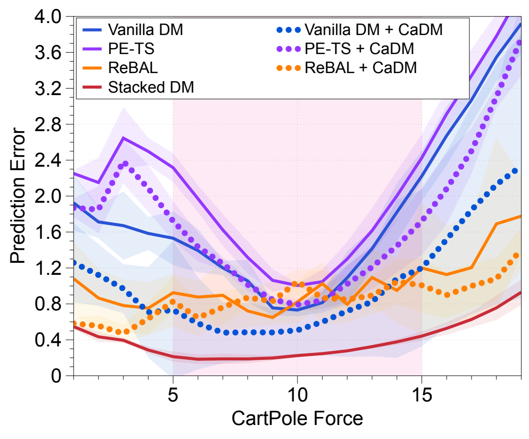
(a) CrippledHalfCheetah



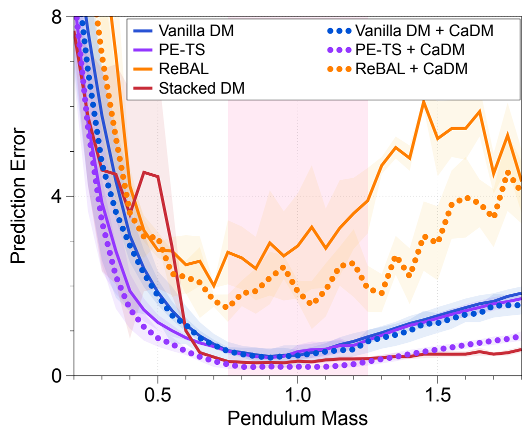
(b) SlimHumanoid

Figure 16. The performance (average returns) of dynamics models optimized by variants of the proposed prediction objective in (1) on (a) CrippledHalfCheetah, and (b) SlimHumanoid. The transition dynamics of environments are changing in both training and test environments. We remark that test environments consist of moderate and extreme environments, where the former draws environment parameters from a closer (yet different) range to the training one, compared to the latter. The solid line and shaded regions represent the mean and standard deviation, respectively, across five runs.

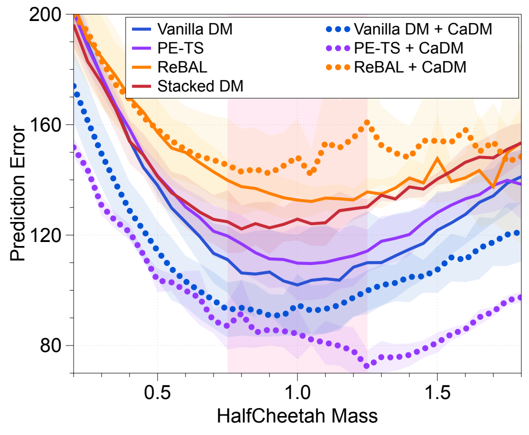
F. Prediction Error



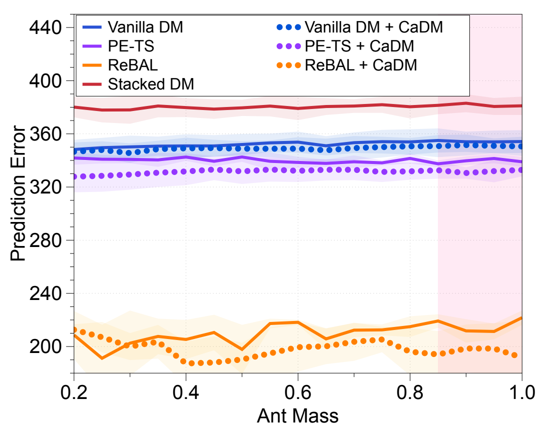
(a) CartPole



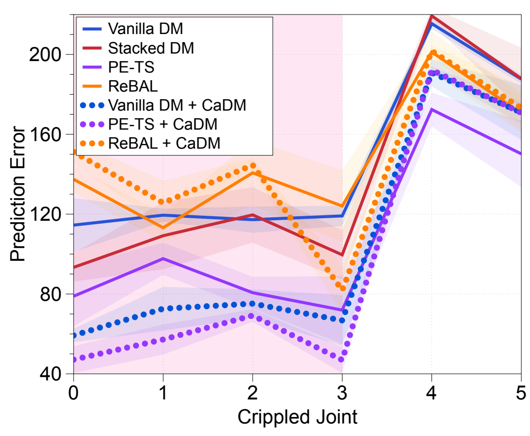
(b) Pendulum



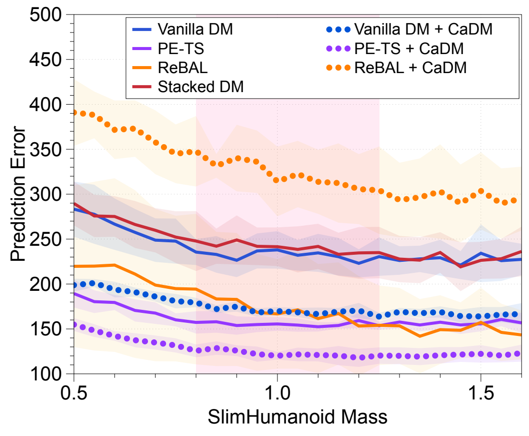
(c) Half-cheetah



(d) Ant



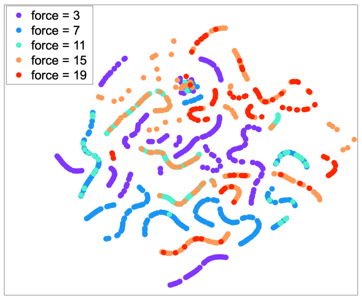
(e) CrippledHalfCheetah



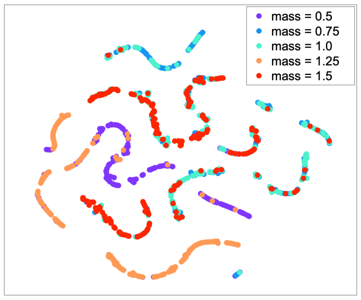
(f) SlimHumanoid

Figure 17. Prediction errors on (a) CartPole, (b) Pendulum, (c) HalfCheetah, (d) Ant, (e) CrippledHalfCheetah, and (f) SlimHumanoid with varying simulation parameters. The solid line and shaded regions represent the mean and standard deviation, respectively, across three runs.

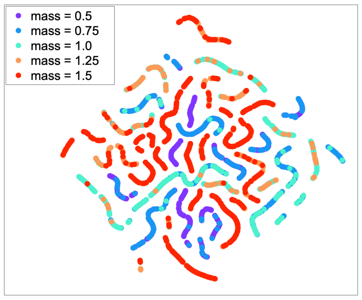
G. Embedding Analysis



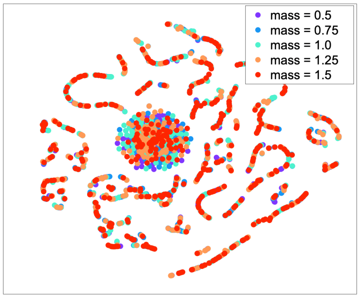
(a) CartPole



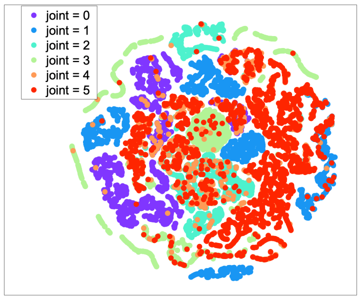
(b) Pendulum



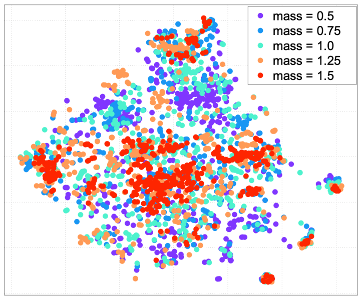
(c) HalfCheetah



(d) Ant

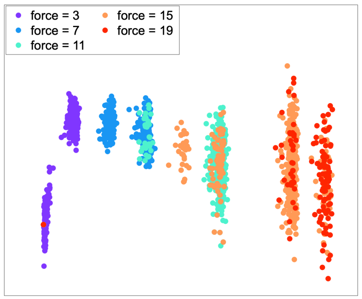


(e) CrippledHalfcheetah

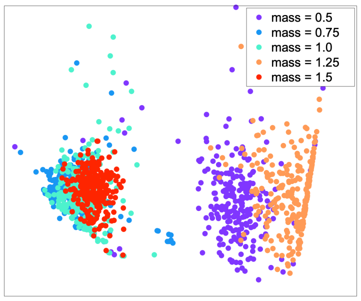


(f) SlimHumanoid

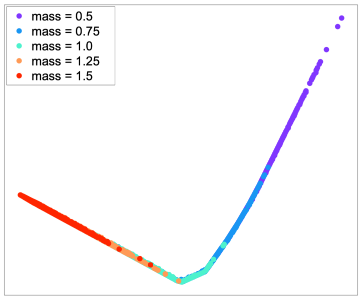
Figure 18. t-SNE (Maaten & Hinton, 2008) visualization of context latent vectors extracted from trajectories collected in various control tasks. Embedded points from environments with the same parameter have the same color.



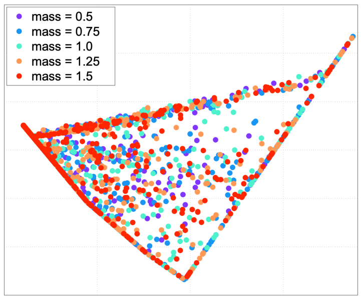
(a) CartPole



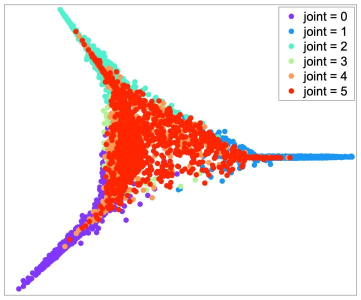
(b) Pendulum



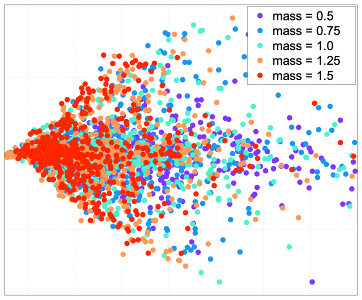
(c) HalfCheetah



(d) Ant

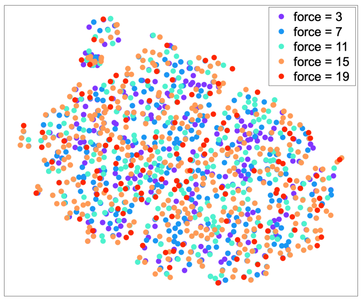


(e) CrippledHalfcheetah

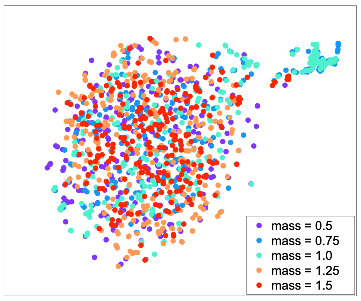


(f) SlimHumanoid

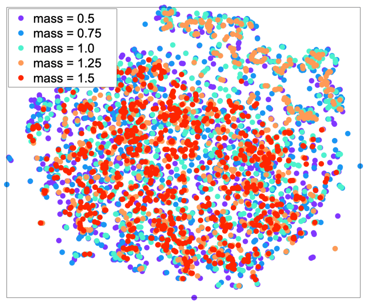
Figure 19. PCA visualization of context latent vectors extracted from trajectories collected in various control tasks. Embedded points from environments with the same parameter have the same color.



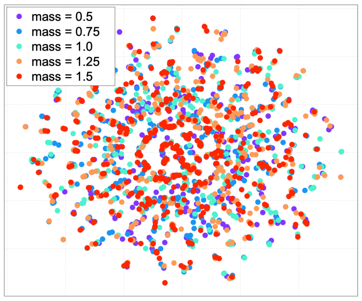
(a) CartPole



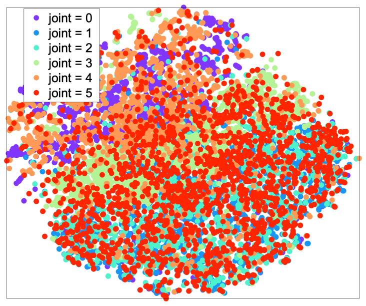
(b) Pendulum



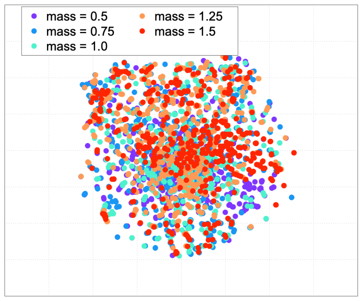
(c) HalfCheetah



(d) Ant

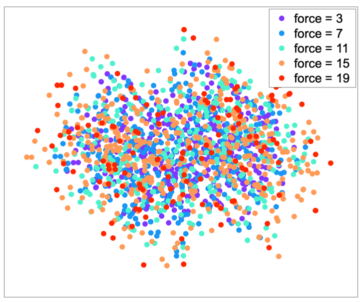


(e) CrippledHalfcheetah

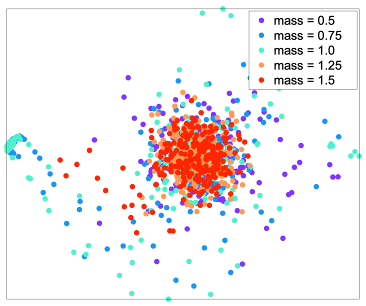


(f) SlimHumanoid

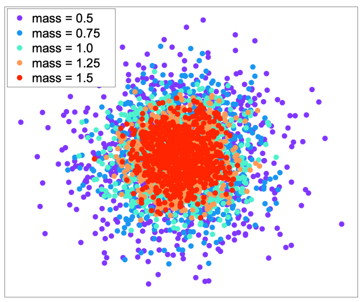
Figure 20. t-SNE (Maaten & Hinton, 2008) visualization of raw state-action vectors extracted from trajectories collected in various control tasks. Embedded points from environments with the same parameter have the same color.



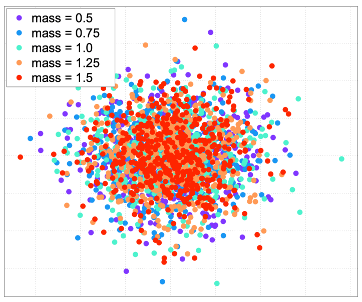
(a) CartPole



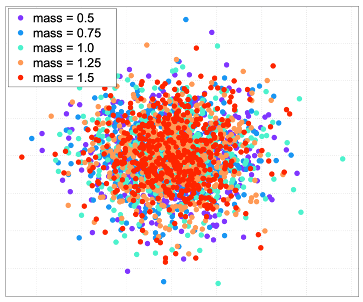
(b) Pendulum



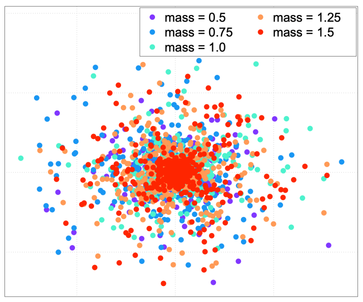
(c) HalfCheetah



(d) Ant



(e) CrippledHalfcheetah



(f) SlimHumanoid

Figure 21. PCA visualization of raw state-action vectors extracted from trajectories collected in various control tasks. Embedded points from environments with the same parameter have the same color.