PREVENTING MODE COLLAPSE WHEN IMITATING LATENT POLICIES FROM OBSERVATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Imitation from observations only (ILfO) is an extension of the classic imitation learning setting to cases where expert observations are easy to obtain but no expert actions are available. Most existing ILfO methods either require access to task-specific cost functions or large amounts of interactions with the target environment. Learning a forward dynamics model in combination with a latent policy has been shown to solve these issues. However, the limited supervision in the ILfO scenario can lead to a mode collapse in learning the generative forward model and the corresponding latent policy. In this paper, we analyse the mode collapse problem and show that it can occur whenever the expert is deterministic, and may also occur due to bad initialization of the models. Under the assumption of piecewise continuous system dynamics, we propose a method to prevent the mode collapse using clustering of expert transitions to pre-train the generative model and the latent policy. We show that the resulting method prevents mode collapse and improves performance in five different OpenAI Gym environments.

1 INTRODUCTION

Imitation learning (IL) as a paradigm to imitate the policy of another agent or expert has been introduced in the 90s (Michie et al., 1990; Pomerleau, 1991; Bain & Sammut, 1995). The recent advances in machine learning and the adoption of deep learning have allowed researchers to learn more complex models. As a result, IL has become a powerful tool to solve tasks like controlling autonomous vehicles (Pomerleau, 1991; Codevilla et al., 2018; Bojarski et al., 2016; Pan et al., 2017; Giusti et al., 2015; Abbeel & Ng, 2004), robotic manipulation (Fang et al., 2019; Finn et al., 2016) and imitation learning in simulated environments (Ho & Ermon, 2016). However, most state of the art IL methods rely on the availability of abundant expert state-action data or apply data-driven methods from reinforcement learning which are not always applicable in real-world problems.

The main issues that IL researchers currently address are (1) the data-efficiency of IL algorithms, (2) the reliance on knowledge about the task properties and (3), the availability of expert actions. To address those three problems, Edwards et al. (2019) proposed ILPO, a semi-supervised model-based ILfO method in which a latent policy is learned concurrently with a forward dynamics model conditioned on latent actions and expert state observations. Using this latent policy, a data-efficient mapping from latent actions to real action space can be learned with few environment interactions without the need for task-specific knowledge or expert actions. However, having the lack of supervision can make it difficult to recover from mode collapse, a state in which the latent dynamics model collapses and expresses the dynamics with only one latent action. Since forward dynamics and latent policy are learned in tandem, no useful latent policy can be obtained in this collapsed state. This problem makes the method sensitive to initialization of the model parameters and becomes especially an issue when expert demonstrations have been collected using a deterministic expert policy.

In this paper, we analyse the causes of mode collapse in model-based ILfO and discuss possible solutions. We consider in particular physical systems, which typically exhibit piecewise continuous system dynamics due to being governed by differential equations. Under the assumption of piecewise continuous system dynamics, similar expert transitions correspond to similar actions almost everywhere. Based on this assumption, we propose an unsupervised clustering approach to pre-train the latent policy and dynamics model, and prevent mode collapse during the latent policy learning.

We evaluate the performance of the learned latent policies and show that pre-training can prevent mode collapse and improve the performance of the baseline ILPO approach.

2 Imitation Learning From Observations Only

Imitation Learning from Observations Only (ILfO) constrains the IL setting further by not relying on expert action information (Torabi et al., 2019; Sun et al., 2019). The motivation is that in many real-world environments precise recording of expert actions is costly or even impossible, but often there is an abundance of video material or other expert state recordings to learn from. In their review on ILfO, (Torabi et al., 2019) identify two core challenges in this setting: (1) perception and (2) control. Perception refers to the fact that one needs to consider how features are extracted from the state-only demonstrations as well as possible embodiment and viewpoint differences to the agent. Control refers to the problem of learning a policy from the perceived observations or learned features. Both problems, feature extraction and policy learning, become difficult and intertwined when only expert states are available. In ILPO this is reflected by the approach of learning policy and dynamics simultaneously.

ILfO methods can be further categorised into model-free and model-based approaches. Model-free approaches usually address the lack of expert actions by using adversarial methods and/or reward engineering as substitute supervision signals. Adversarial methods (Torabi et al., 2019; Nair et al., 2017; Ho & Ermon, 2016; Schroecker & Isbell, 2017) usually require large amounts of interactions with the environment which is not always feasible or safe. Reward engineering can be used to address this, but requires either human knowledge of the task or other data-driven methods to extract supervision signals from the expert data (Sermanet et al., 2018; Liu et al., 2018; Gupta et al., 2017).

Compared to model-free methods, model-based methods promise benefits, especially with respect to data-efficiency and the fact that they can be used to learn good control policies. Behavioral Cloning from Observation (BCO) (Torabi et al., 2018) learns an inverse dynamics model $P(a|s_{t+1}, s_t)$ to infer actions from transitions while interacting with the environment and then uses the actions to learn the imitation policy. However, Sun et al. (2019) argue that there is no guarantee that obtaining an inverse dynamics model is possible. Consider that the inverse dynamics model can be rewritten as

$$P(a|s_{t+1}, s_t) = \frac{P(a, s_{t+1}, s_t)}{P(s_{t+1}, s_t)} = \frac{P(s_{t+1}|a, s_t)\pi^\star(a|s_t)}{P(s_{t+1}|s_t)} \tag{1}$$

Thus,

$$P(a|s_{t+1}, s_t) \propto P(s_{t+1}|a, s_t)\pi^\star(a|s_t) \tag{2}$$

so that the inverse model is ill-defined alone without considering the corresponding policy, in the case of stochastic dynamics. To learn a probabilistic inverse model for a particular expert, the data to learn the model needs to be gathered from the same expert.

Edwards et al. (2019) have proposed ILPO, a method that alleviates this problem by learning a latent policy $\pi_\omega(z|s_t)$ in parallel with a latent forward dynamics model $G(s_{t+1}|s_t, z)$, circumventing the need for the true expert policy. Later the latent policy is mapped to the real action space with a few environment interactions. Those two steps are referred to as (1) *latent policy learning* and (2) *action remapping*. The dependency on the expert policy is approximated by

$$\pi^\star(a|s_t) \sim P(a|z)\pi_\omega(z|s_t) \tag{3}$$

and the expert forward model by

$$P^\star(s_{t+1}|s_t, a) \sim G(s_{t+1}|s_t, z)\pi_\omega(z|s). \tag{4}$$

Note that ILPO assumes a discrete action space and therefore also a discrete latent action space. In summary, the reliance on the expert policy is split into an intermediate learning problem in which a low dimensional latent action space Z is learned in an unsupervised way from offline expert state observations.

In practice, the forward dynamics model G is trained by predicting $|Z|$ possible state transitions Δ_z using the forward dynamics model and picking the best prediction. The minimization loss

$$\mathcal{L}_{\text{min}} = \min_z ||\Delta_t - G(s_t, z)||^2 \tag{5}$$

assures gradient updates are performed on the instance of G which leads to the best prediction to learn to separate the different modes/latent actions. The latent policy is trained by using the forward model to predict the expected next state and minimizing the loss

$$\mathcal{L}_{\text{exp}} = ||s_{t+1} - \sum_z \pi_\omega(z|s_t) G(s_t, z)||^2. \tag{6}$$

Ideally one would want Z to be low dimensional such that the behavior of the expert can be sufficiently expressed with the available latent actions while the mapping to real actions $P(a|z)$ is as easy as possible to learn. Edwards et al. (2019) have investigated by hyperparameter search that a good initial guess for the number of latent actions is the true number of actions that the expert takes.

3 Mode Collapse when Imitating Latent Policies

While ILPO is extremely data efficient, it is not possible to guarantee that a good latent policy can be learned. A core component of ILPO, the forward dynamics model, which is a generative model, is susceptible to mode collapse.

3.1 Mode Collapse

Mode collapse has so far been discussed in generative models such as GANs and VAEs 5.2. In ILPO mode collapse occurs when the learned forward dynamics are able to model the expert observations with latent actions that cannot be remapped successfully. As a result, the number of effective latent actions is less than the number of true actions and worst case collapses to one mode or latent action. Thus, the mode collapse in ILPO can be observed by monitoring the diversity of predicted latent actions by the latent policy π_ω.

A deterministic expert policy $a = \pi^\star(s_t)$ can always cause mode collapse. This is due to the fact that a deterministic expert policy can be approximated as

$$a = \pi^\star(s_t) \sim P(a|z)\pi(z|s_t) = P(a|\pi(s_t)) \tag{7}$$

which is independent of z. Furthermore, the forward dynamics can be simplified to

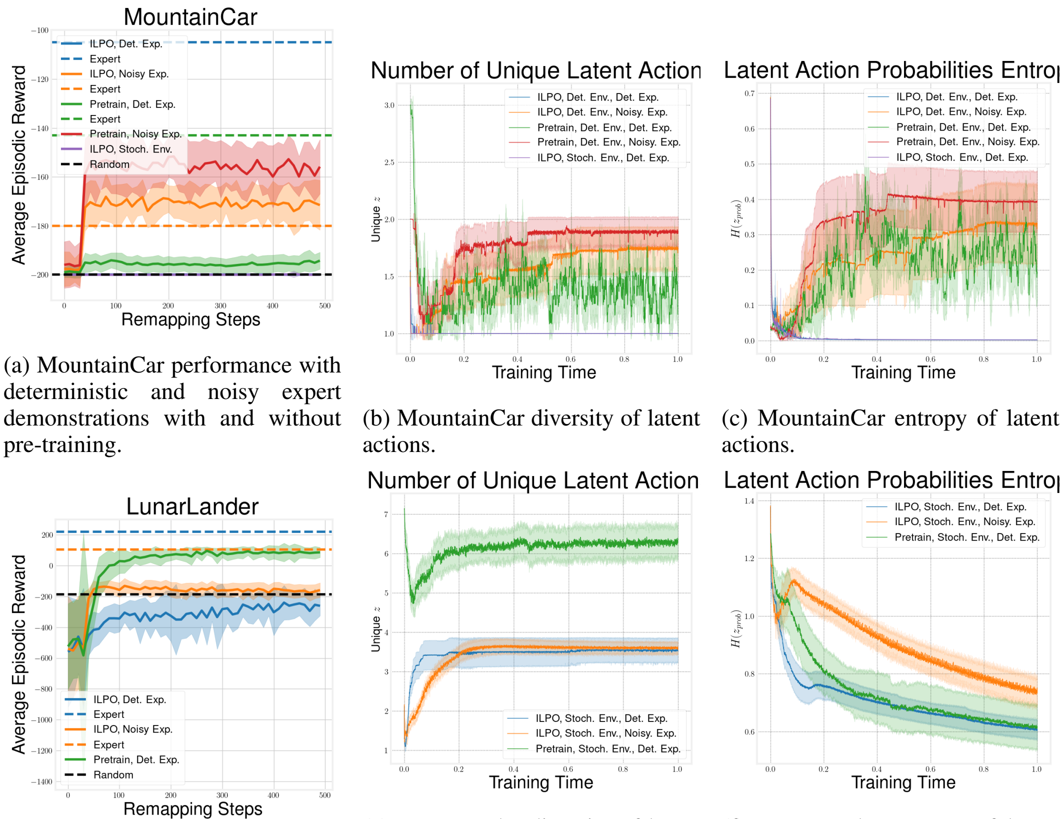
$$\begin{aligned} G(s_{t+1}|s_t, z) = &\frac{G(s_{t+1}, s_t, z)}{G(z, s_t)} = \\ &\frac{G(z)G(s_{t+1}, s_t)}{G(z)G(s_t)} = G(s_{t+1}|s_t) \end{aligned} \tag{8}$$

which does not depend on the latent action. As a result, the minimization loss term in Eq. 5 collapses to one mode, usually the one which due to initialization minimizes the loss in the beginning of the training. Therefore, in order for the approximation using a latent policy in Eq. 3 to work, the policy used to gather data π must be stochastic.

3.2 Experimental Evidence

Let us look at two cases of mode collapse, the Gym environments MountainCar and LunarLander[1]. Fig. 1a shows in blue that the task performance of the MountainCar task in the standard deterministic environment with deterministic expert actions is -200, the same as for a random policy, indicating that the task was not learned successfully. Mode collapse can be diagnosed by analysing the average number of latent actions in a batch and the entropy of the action probabilities predicted by π during latent policy learning. Fig. 1c illustrates that in the baseline MountainCar experiment plotted in blue, the latent policy and forward model collapse to one latent action almost immediately in the beginning of the latent policy learning. Figures 1d-1f show the same data in blue for the LunarLander experiment where the performance of the baseline case (stochastic environment and deterministic policy) is below a random policy. On average less than 4 latent actions are used with low entropy of latent actions which is not enough to achieve expert performance in the task which requires careful balancing of all 4 real actions.

[1]Implementation details and hyperparameters are presented in Appendix A.2.



(a) MountainCar performance with deterministic and noisy expert demonstrations with and without pre-training.

(b) MountainCar diversity of latent actions.

(c) MountainCar entropy of latent actions.

(d) LunarLander performance.

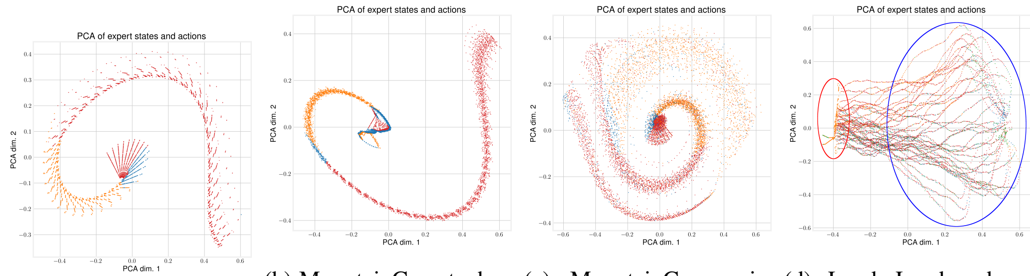
(e) LunarLander diversity of latent actions.

(f) LunarLander entropy of latent actions.

Figure 1: Task performance and action diversity measured as average number of latent actions and the average entropy of the latent action probabilities per batch during training. Each plot shows the mean and standard deviation of 250 experiment runs. More details on the experiments can be found in A.2.

Next, we demonstrate which properties of the expert demonstrations facilitate mode collapse in the latent policy learning step. Fig. 2a shows the state space of the MountainCar experiment colored by the ground truth expert actions. One can observe that the expert state demonstrations exhibit distinct action clusters in the state space. In similar states, the expert usually takes similar actions. As a result a generative model forward dynamics model $G(z, s_t)$ can express the dynamics in those regions independent of the latent action z as $G(s_t)$. Combining those local models yields a global dynamics model that is independent of z.

The model collapse in the LunarLander environment shown in Fig. 2d, is not as obvious. We hypothesize that the expert behavior in the demonstrations can be described by only a few latent actions. Although principal component analysis (PCA) has been applied to visualize the state-action space, one can see the individual paths each of the expert episodes took, starting on the right and landing on the left. The LunarLander expert uses all 4 actions, however qualitative analysis of the expert behavior shows that the task could be described at multiple levels of complexity which are highlighted in Fig. 2d with blue and red ellipses. Usually, the expert exhibits two main behaviors differing in their dynamics: (blue) controlling the descent by balancing the lander upright and centering it using the left and right thrusters, and (red) slowing down before touchdown in the landing zone using the main engine. While each of those behaviors consists of more complex behaviors comprised of patterns using multiple actions, the latent dynamics can be described by a simplified behavior which cannot be mapped to the real action space such that the task can be solved adequately. More experimental results in other environments are discussed in Appendix A.1.



(a) MountainCar, baseline. (b) MountainCar, stochastic environment. (c) MountainCar, noisy expert actions. (d) LanderLander, baseline.

Figure 2: State-action space for different expert policy and environment configurations with PCA reduced to 2 dimensions colored with expert actions.

Figure 3: State-action space with PCA to reduce to 2 dimensions colored by expert actions.

4 Addressing Mode Collapse

In the following, we will discuss how mode collapse in imitation learning has been addressed so far and propose a new method for pre-training latent policies to prevent mode collapse.

4.1 Stochastic Environments

Following from the above argumentation, it is evident that repeating the experiments in stochastic versions of the environments will not have much effect on the mode collapse problem. Stochastic environments merely add noise to the states. Fig. 2b shows the MountainCar expert state-action distribution with a deterministic expert policy in a stochastic version of the environment. Compared to the baseline in Fig. 2a there is more noise around the origin, but the general problem of clusters with the same expert action persists. Consequently, the performance and latent action diversity of the deterministic policy in the stochastic environment version of the MountainCar exhibit the same collapsed performance shown in purple in Fig. 1a-1c.

4.2 Noisy Expert Demonstrations in ILPO

While mode collapse has not been discussed by Edwards et Al., the authors briefly note that ILPO performs better when the expert actions are noisy. This is in line with our theoretical analysis in 3.1, a stochastic expert policy is required to learn a latent policy. Fig. 2c shows that the state-action space with noisy expert actions now displays locally diverse expert actions. The assumption in Edwards et al. (2019) is that the expert's action selection is noisy, for example when learning from a human. This however limits the range of applications since the noise must be introduced while recording the expert. It is not possible to introduce noise to deterministic expert data after the data collection since only states are recorded. We conduct a set of experiments with the baseline ILPO method where the expert takes random/noisy actions with a certain probability during the recording of the dataset.

In Fig. 1 we compare the performance of ILPO as a baseline with and without noisy expert data. We can see in orange that in the MountainCar environment (1a) ILPO performs better with noisy actions. In fact, ILPO was not able to solve the MountainCar task at all with deterministic expert actions. In the LunarLander task the performance with noisy expert actions is better but not significantly better than a random policy.

While noisy expert demonstrations improve performance, the results highlight another problem: tasks like the MountainCar require accurate control to solve the task which is not always possible when the expert is taking random actions. In the MountainCar experiment, the agent must build enough momentum which is significantly more difficult and often impossible when random actions counteract the momentum of the agent. In Fig. 1a we plot as an orange dashed line the median expert performance with a 20% chance of taking random actions during recording the MountainCar expert. The expert performance with this amount of noise is close to random performance (-200 for a random policy, -186 with noisy actions compared to -105 with deterministic actions).

Table 1: Pearson rank correlation computed on 10000 samples from the expert data. The p-value was 0 in all tests.

Experiment	**Pearson ρ (States/actions)**	**Pearson ρ (Transitions/actions)**
Acrobot	-0.07	-0.09
Acrobot (smaller Δt)	-0.12	-0.14
Cartpole	-0.16	-0.87
MountainCar	-0.22	-0.38
LunarLander	-0.13	0.69
Pong	-0.04	-0.48

Algorithm 1 Pre-training

Require: Expert state-only transitions D^e, latent policy parameters ω, latent dynamics parameters θ

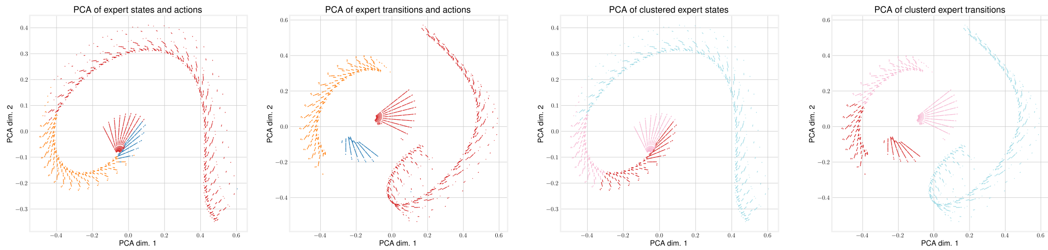
1: $z^l \leftarrow AgglomerativeClustering(D^e)$
2: $\omega \leftarrow Pre\text{-}train\pi_\omega(D^e, z^l)$
3: $\theta \leftarrow Pre\text{-}trainG_\theta(D^e, z^l)$
4: ILPO(D^e, ω, θ)

4.3 Pre-training ILPO

To circumvent the need for noisy expert actions in ILPO, we propose a pre-training method to prime the latent policy such that mode collapse does not occur, or bad initialisations are less likely. The core idea is that under the assumption of piecewise continuous dynamics, similar transitions are usually caused by similar actions of the expert. Therefore, the transitions in the expert data can provide a prior which we can use to pre-train the latent policy. We demonstrate this principle by obtaining distance matrices, containing the pair-wise distance of all states and transitions, and computing the Spearman rank correlation to an action distance matrix. Table 1 shows the Pearson rank correlation between the states and actions and transitions and actions. The transition distance matrices for 4 of the 5 environments have a high correlation with the actions while the state distance matrices do not.

Algorithm 1 shows how the novel pre-training step precedes the baseline ILPO training. First we we use agglomerative clustering on the expert transitions to generate latent action labels z^l, effectively assigning similar labels to similar transitions. The state only expert data and the resulting labels are then used to pre-train the latent policy π_ω and the forward dynamics model G_θ end-to-end. Figures 4e to 4h show a visualisation of the LunarLander expert states and transitions colored by expert actions and the clustered classes. We can see that the expert transition-action space exhibits clear clusters (Fig. 4f) while the state-action space does not (Fig. 4e). Fig. 4h shows that the agglomerative clustering can obtain sensible clusters from the expert transitions. Fig. 4g shows that the clustering is a good initial approximation of the expert actions. The same observations hold for the MountainCar experiments shown in the same figure and the CartPole and Pong environment discussed in A.1.

After performing the pre-training, the latent policy learning and remapping is performed like in the baseline ILPO method with the pre-training network weights ω and θ. The results are shown in Fig. 1 in green. In the LunarLander experiment, the pre-trained imitation policy achieves close to expert performance, clearly outperforming the baseline and noisy expert data case. The LunarLander experiment with pre-training also exhibits a higher variety of latent actions used and lower actions entropy indicating that a more complex latent policy has been learned and that the model is confident about the latent action selection. In the MountainCar task, the observed latent action diversity indicates that mode collapse has been prevented and the performance is better than without pre-training. However, the baseline with noisy expert policy in orange shows better performance. In red we plot the pre-trained MountainCar with noisy expert data, a combination of both presented to prevent mode collapse. This configuration shows the best results significantly outperforming the baseline noisy expert setup and even more interesting outperforming the expert itself from which

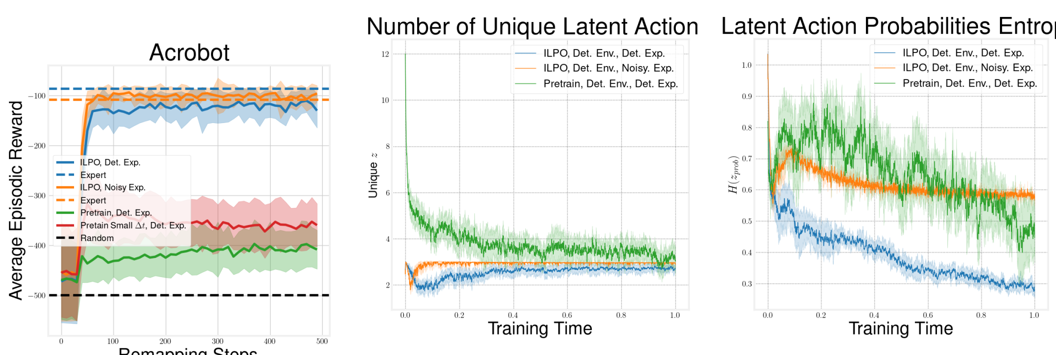


(a) Mountaincar state-action space. (b) Mountaincar transition-action space. (c) Mountaincar clustered states. (d) Mountaincar clustered transitions.

(e) Lander expert state-action space. (f) Lander expert transition-action space. (g) Lander clustered states. (h) Lander clustered transitions.

Figure 4: Visualizations of the states and transitions of the LunarLander expert data colored by true actions and clustered classes.

the data was collected by a large margin. This indicates that in some cases ILPO is able to learn a latent policy that is robust to random noise in the expert policy. We observed similar results in the Pong experiment. More details on the Pong and CartPole experiment can be found in Appendix A.1. In Appendix A.3 we analyse the performance when only pre-training and no policy learning is performed and show that in some environments the pre-trained latent policy is sufficient to achieve good performance.

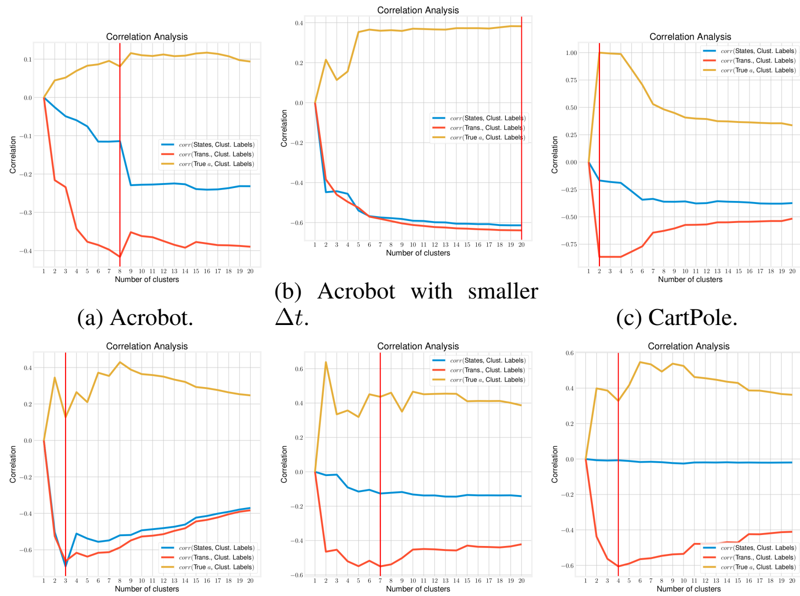


(a) Acrobot performance. (b) Acrobot diversity of latent actions. (c) Acrobot entropy of latent actions.

Figure 5: Task performance and action diversity measured as average number of latent actions and the average entropy of the latent action probabilities per batch during training.

4.4 ACROBOT ENVIRONMENT PROPERTIES

In the Acrobot environment, the performance of the pre-training method (green) does not reach expert performance in Fig. 5a. We have investigated the properties of the expert data visualised in Appendix A.1 in Fig. A.2 and found that neither transitions (A.2a) nor states A.2a exhibit any discernible clusters. From Table 1 we can see that the Acrobot expert data does not exhibit strong correlation between states or transitions and actions and thus the assumption of transition and action



(a) Acrobot. (b) Acrobot with smaller Δt. (c) CartPole.

(d) MountainCar. (e) LunarLander. (f) Pong.

Figure 6: Correlation between the latent action labels obtained from clustering. In the LunarLander experiment the data was from the stochastic environment with deterministic expert. In all other environments the environment was deterministic.

similarity does not hold in this environment. We investigated the implementation of the Acrobot environment and found that the physics integrator Δt is rather large with 0.2 seconds. The angular velocities of the second Acrobot joint in the expert data are very large and as a result, transitions become practically random as multiple rotations of the joints are possible in 0.2 seconds.

We conducted an additional experiment in which we changed the environment properties such that the maximum joint angular velocities are limited and the physics Δt is 0.075. A new expert was trained and new data was recorded. The absolute correlation in Table 1 has increased to -0.14 which indicates that the changes had the desired effect, albeit not strong. The state-action and transition-action distributions in Fig. A.2e and Fig. A.2f still do not look promising. Fig. 5a shows that now the pre-trained imitation policy performs better but still does not reach expert performance. These results however may not be very representative as the expert data is from another expert trained in an environment with different physical properties but has been evaluated in the same setting as the other Acrobot experiments.

4.5 Selecting the Number of Latent Actions

One important parameter in ILPO and for our pre-training extension is the number of clusters/latent actions. Edwards et al. (2019) have investigated the effect of the number of latent actions by repeating the experiments and plotting performance against the number of used latent actions. The results show that not all numbers of latent actions lead to good results. In our method, the selection of clusters is directly related to the number of latent actions. To this end, we propose a method to identify the number of latent actions/clusters based on how much the transition clusters follow the assumption we make on the transition-action similarity.

In Fig. 6 we plot the Spearman rank correlation between states, transitions and true actions, and the labels obtained from clustering for 1-20 clusters. We selected the number of latent actions/clusters for the experiments such that the absolute value of correlation between clustered labels and transitions (orange line) is maximized. Those values are shown by a vertical red line in the figures. The quality of this measure can be verified by comparing it to the correlation between cluster labels and expert actions in yellow. We can see that our method of picking latent actions usually yields a number of latent actions for which transitions and expert actions correlate.

5 Related Work

5.1 Imitation Learning

Imitation Learning (IL) describes methods in which an imitation policy is learned to mimic the behavior of an expert or a target agent. The IL setting usually assumes that expert demonstrations consisting of state and action pairs are available. Classic approaches in IL have approached the problem from two perspectives. First, behavioral cloning (BC), the predominant method proposed in the late 90s (Michie et al., 1990; Pomerleau, 1991; Bain & Sammut, 1995; Bagnell et al., 2006; Ross et al., 2011; Daftry et al., 2016), directly learns the imitation policy using the expert state action pairs. Drawbacks of BC are mainly related to covariate shift, which means in the IL setting that feedback loops and uncertainty in the imitated behavior may lead the imitator to new situations in which the learned imitated behavior might fail (Ross & Bagnell, 2010; Spencer et al., 2021).

The second approach to IL is based on inverse reinforcement learning (IRL) where expert demonstrations are used to infer the expert's reward function and then use reinforcement learning methods and access to the environment to learn an imitation policy. While those methods are considered more robust to covariate shift, most of the IRL-based methods require an extensive amount of environment interactions (Ng et al., 2000; Abbeel & Ng, 2004; Russell, 1998; Finn et al., 2016; Ho & Ermon, 2016). More recent IRL-based methods focus on using other aspects such as the temporal similarity of state action pairs (Schroecker & Isbell, 2017) or employ optimal transport and the Wasserstein distance as a measure between expert and imitator state-action distributions (Dadashi et al., 2020; Fickinger et al., 2021).

5.2 Mode Collapse in Generative Models

Mode collapse, usually discussed in the context of Generative Adversarial Neural Networks (GANs), describes a failure scenario in which a multimodal generative model collapses to one mode and the generator network generates data with low variety (Salimans et al., 2016; Che et al., 2016). Investigations of the mode collapse problem have shown that it is related to catastrophic forgetting and the optimization process in GANs which prevents the generator to break out of the model collapse (Che et al., 2016; Thanh-Tung & Tran, 2020). As a result, the generator fails to generate diverse data, which is well separated in observation space from the training data. This makes it very easy for the discriminator to detect them which in turn leads to the Discriminator not learning useful features.

Solutions to mode collapse in GANs include clustering the data based on knowledge about the classes in the training dataset, mode regularization, minibatch discrimination (Che et al., 2016), continual learning or using optimizers with momentum to propagate knowledge during training to prevent catastrophic forgetting (Thanh-Tung & Tran, 2020).

6 Conclusion and Further Work

In this work, we investigated the mode collapse problem when imitating latent policies from observations only. We showed that mode collapse in ILPO, a state of the art method by Edwards et al. (2019), can be caused by a combination of bad initialisation and unfavorable properties of the expert states such as lack of diversity in expert actions. To this end, we proposed a clustering-based method to pre-train the latent policy and latent dynamics model. We showed that with this modification ILPO can work in environments where it previously failed and where its performance improved in others. Furthermore, we analysed the properties of the environments and expert data and their influence on the ILfO process and found that discontinuity in the dynamics can reduce imitation learning performance. Lastly, we proposed a method to choose an important hyperparameter, the number of latent actions.

In the future, we see potential in further exploring design choices in both latent policy learning and action remapping. In the action remapping step, the exploration mechanism is a key component when obtaining high-quality data from the environments and greatly contributes to the data-efficiency of the method. In the latent policy learning, more advanced representation learning methods leveraging the temporal smoothness of the expert data or using sequences instead of single states as input could greatly improve the quality of the learned latent policy and dynamics model.

REPRODUCIBILITY STATEMENT

We have based our implementation on the code provided by Edwards et al. (2019) on GitHub. Modifications and hyperparameters we changed are described in A.2. In case of acceptance, we will make our code repository public.

REFERENCES

Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 1, 2004.

J Bagnell, Joel Chestnutt, David Bradley, and Nathan Ratliff. Boosting structured prediction for imitation learning. *Advances in Neural Information Processing Systems*, 19, 2006.

Michael Bain and Claude Sammut. A framework for behavioural cloning. In *Machine Intelligence 15*, pp. 103–129, 1995.

Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

Tong Che, Yanran Li, Athul Paul Jacob, Yoshua Bengio, and Wenjie Li. Mode regularized generative adversarial networks. *arXiv preprint arXiv:1612.02136*, 2016.

Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 4693–4700. IEEE, 2018.

Robert Dadashi, Léonard Hussenot, Matthieu Geist, and Olivier Pietquin. Primal wasserstein imitation learning. *arXiv preprint arXiv:2006.04678*, 2020.

Shreyansh Daftry, J Andrew Bagnell, and Martial Hebert. Learning transferable policies for monocular reactive mav control. In *International Symposium on Experimental Robotics*, pp. 3–11. Springer, 2016.

Ashley Edwards, Himanshu Sahni, Yannick Schroecker, and Charles Isbell. Imitating latent policies from observation. In *International conference on machine learning*, pp. 1755–1763. PMLR, 2019.

Bin Fang, Shidong Jia, Di Guo, Muhua Xu, Shuhuan Wen, and Fuchun Sun. Survey of imitation learning for robotic manipulation. *International Journal of Intelligent Robotics and Applications*, 3(4):362–369, 2019.

Arnaud Fickinger, Samuel Cohen, Stuart Russell, and Brandon Amos. Cross-domain imitation learning via optimal transport. *arXiv preprint arXiv:2110.03684*, 2021.

Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International conference on machine learning*, pp. 49–58. PMLR, 2016.

Alessandro Giusti, Jérôme Guzzi, Dan C Cireşan, Fang-Lin He, Juan P Rodríguez, Flavio Fontana, Matthias Faessler, Christian Forster, Jürgen Schmidhuber, Gianni Di Caro, et al. A machine learning approach to visual perception of forest trails for mobile robots. *IEEE Robotics and Automation Letters*, 1(2):661–667, 2015.

Abhishek Gupta, Coline Devin, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Learning invariant feature spaces to transfer skills with reinforcement learning. *arXiv preprint arXiv:1703.02949*, 2017.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.

YuXuan Liu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Imitation from observation: Learning to imitate behaviors from raw video via context translation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1118–1125. IEEE, 2018.

Donald Michie, Michael Bain, and J Hayes-Miches. Cognitive models from subcognitive skills. *IEE control engineering series*, 44:71–99, 1990.

Ashvin Nair, Dian Chen, Pulkit Agrawal, Phillip Isola, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Combining self-supervised learning and imitation for vision-based rope manipulation. In *2017 IEEE international conference on robotics and automation (ICRA)*, pp. 2146–2153. IEEE, 2017.

Andrew Y Ng, Stuart Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, pp. 2, 2000.

Yunpeng Pan, Ching-An Cheng, Kamil Saigol, Keuntaek Lee, Xinyan Yan, Evangelos Theodorou, and Byron Boots. Agile autonomous driving using end-to-end deep imitation learning. *arXiv preprint arXiv:1709.07174*, 2017.

Dean A Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3(1):88–97, 1991.

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021. URL `http://jmlr.org/papers/v22/20-1364.html`.

Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 661–668. JMLR Workshop and Conference Proceedings, 2010.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

Stuart Russell. Learning agents for uncertain environments. In *Proceedings of the eleventh annual conference on Computational learning theory*, pp. 101–103, 1998.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016.

Yannick Schroecker and Charles L Isbell. State aware imitation learning. *Advances in Neural Information Processing Systems*, 30, 2017.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Google Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 1134–1141. IEEE, 2018.

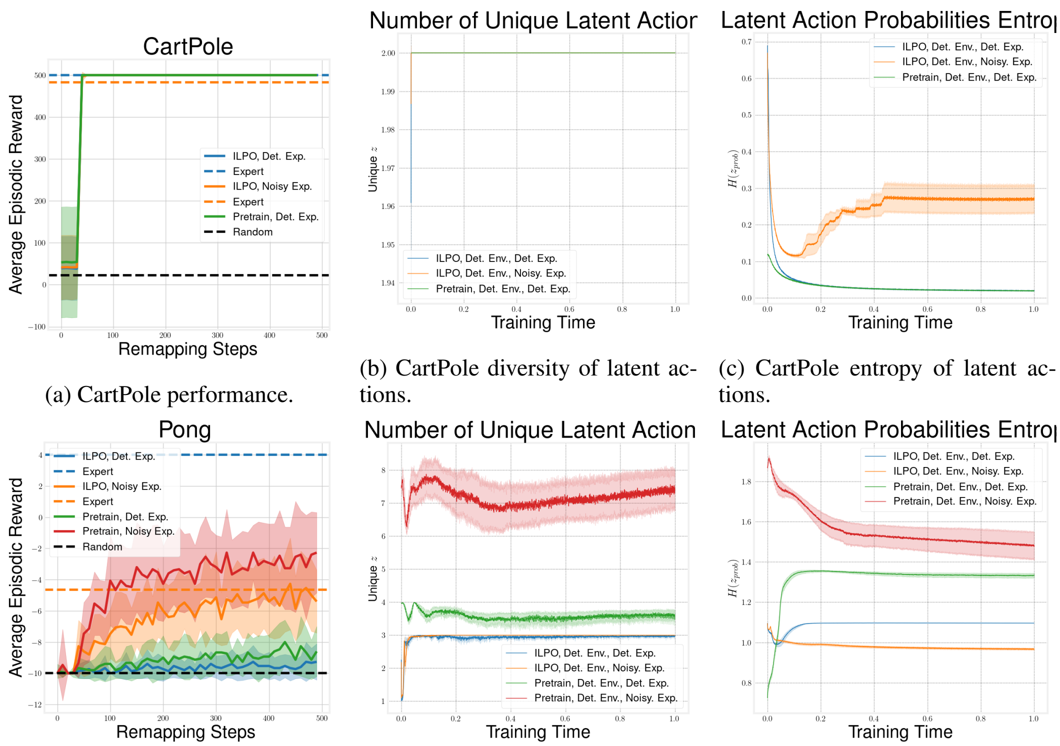
Jonathan Spencer, Sanjiban Choudhury, Arun Venkatraman, Brian Ziebart, and J Andrew Bagnell. Feedback in imitation learning: The three regimes of covariate shift. *arXiv preprint arXiv:2102.02872*, 2021.

Wen Sun, Anirudh Vemula, Byron Boots, and Drew Bagnell. Provably efficient imitation learning from observation alone. In *International conference on machine learning*, pp. 6036–6045. PMLR, 2019.

Hoang Thanh-Tung and Truyen Tran. Catastrophic forgetting and mode collapse in gans. In *2020 international joint conference on neural networks (ijcnn)*, pp. 1–10. IEEE, 2020.

Faraz Torabi, Garrett Warnell, and Peter Stone. Behavioral cloning from observation. *arXiv preprint arXiv:1805.01954*, 2018.

Faraz Torabi, Garrett Warnell, and Peter Stone. Recent advances in imitation learning from observation. *arXiv preprint arXiv:1905.13566*, 2019.



(a) CartPole performance.

(b) CartPole diversity of latent actions.

(c) CartPole entropy of latent actions.

(d) Pong performance.

(e) Pong diversity of latent actions.

(f) Pong entropy of latent actions.

Figure .1: Caption

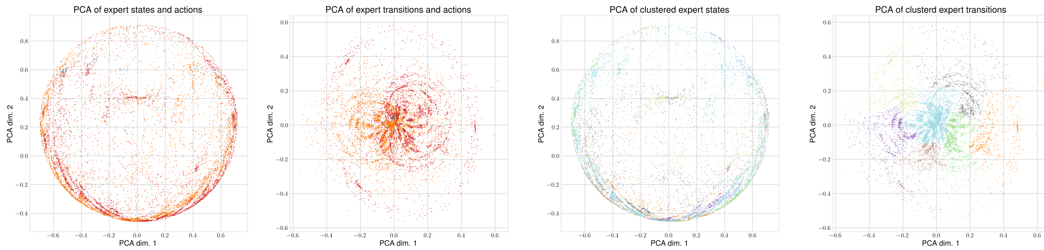
A Appendix

A.1 More Experiments

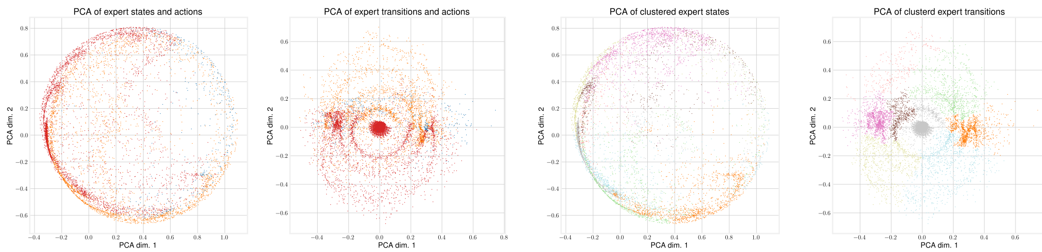
In this section, we discuss the results from the CartPole and Pong experiments.

The results of the CartPole experiment shown in Fig. .1 are very clear and don't leave much room for interpretation. Expert performance is achieved in all settings whiteout problems. The number of average unique latent actions per batch immediately reached 2, the true number of actions used by the expert. No mode collapse occurs in this experiment. The entropy of the latent actions is, as expected, higher for the noisy expert demonstrations, meaning the latent policy is able to capture the uncertainty about the action selection in the expert. This is a result of the task being easy to solve which becomes more evident looking at the state-action space distributions shown in Fig. A.2i to A.2l. In transition space two distinct regions correspond to the two different expert actions. The agglomerative clustering of the transitions is able to recover the expert actions.

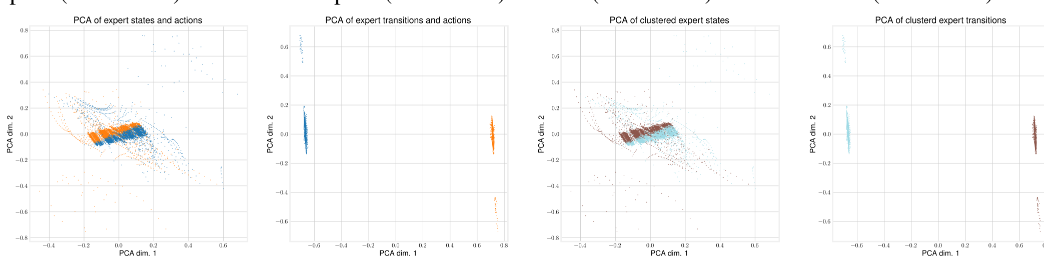
In the Pong experiment, the baseline ILPO method (blue) performed very poorly. Pre-training did not significantly improve the performance which is surprising given that the correlation analysis in Table 1 suggests a high correlation between transitions. A possible reason could be that the number of latent actions (4) suggested by our method obtained confusing clusters (A.2p). Fig. 6f shows that 4 latent actions coincide with a drop in correlation between the cluster labels and the expert actions and that 7 latent actions would have likely yielded better results. The cyan and gray transition clusters cover red and orange expert actions but split alongside another axis. This might be detrimental to the performance and selecting more latent actions could improve results. Interestingly, when combining the pre-training approach with noisy expert actions, the Pong performance is the best and even outperforms the expert from which the data was collected by a large margin.



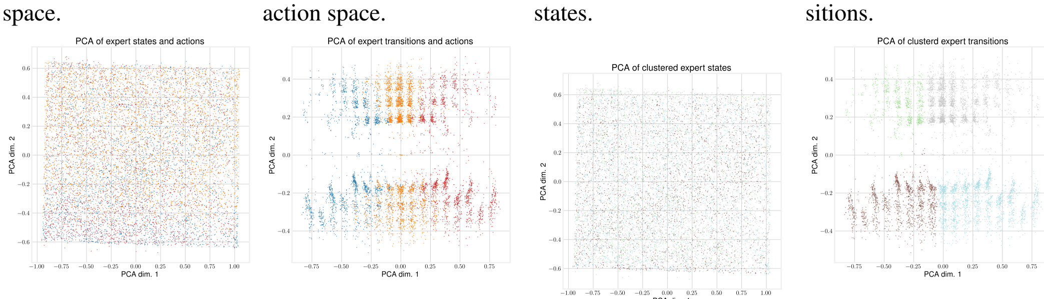
(a) Acrobot state-action space. (b) Acrobot transition-action space. (c) Acrobot clustered states. (d) Acrobot clustered transitions.



(e) Acrobot state-action space (smaller Δt). (f) Acrobot transition-action space (smaller Δt). (g) Acrobot clustered states (smaller Δt). (h) Acrobot clustered transitions (smaller Δt).



(i) Cartpole state-action space. (j) Cartpole transition-action space. (k) Cartpole clustered states. (l) Cartpole clustered transitions.



(m) Pong state-action space. (n) Pong transition-action space. (o) Pong clustered states. (p) Pong clustered transitions.

Figure A.2: A scatter plot of the stat-action spaces. PCA to reduce to 2 state dimensions and color represents the expert actions.

A.2 Experiment Details

In the experiments, we used the ILPO implementation by Edwards et al. (2019) and ported it from TensorFlow to PyTorch.

We used the same neural network architectures and mostly the same hyperparameters for latent policy learning and remapping as Edwards et al. (2019) with the only changes being:

- In all experiments we used 200 epochs for latent policy learning.
- We used 20 epochs in the pre-training step.

We used 10 seeds for the latent policy learning and recorded 25 remapping experiments for each latent policy, meaning that every performance plot in Fig. .1 and Fig. 1 is comprised of 250 experiment runs. We made sure to maintain the same initialisation of network weights (Xavier uniform for weights and zeros for biases) since we found that using different initialisation, such as the standard PyTorch weight initialisation leads to differences in performance when compared to the TensorFlow implementation. This further strengthens our finding that ILPO is very sensitive to model initialisation.

The Pong environment is a custom environment we implemented from scratch. The LunarLander environment in its baseline version is stochastic and the other environments are deterministic. For the experiments, we made stochastic versions of each environment, most notably the MountainCar environment which we discuss in this paper. The stochastic MountainCar environment applies uniform noise to the force applied to the cart.

The experts we used have been trained using the PPO implementation (Schulman et al., 2017) from stable baselines3 (Raffin et al., 2021). Deterministic expert data has been recorded by taking the mode of the PPO action distribution and noisy data by uniformly sampling from the action space with a 20% chance.

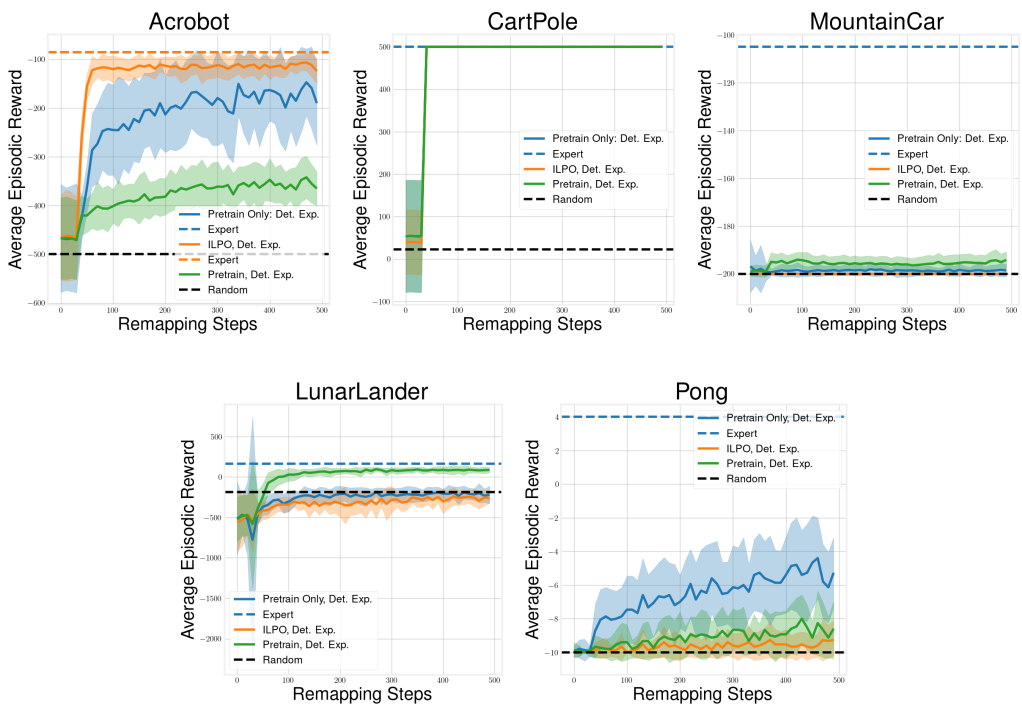


Figure A.3: Task performance in all environments when performing pre-training only instead of latent policy learning.

A.3 Pre-training only

In this section, we discuss the results when pre-training only is used to prime the latent policy before re-mapping and no policy learning is performed. The results in Fig. A.3 show that in all experiments the pre-training only latent policy leads to equal or better performance than baseline ILPO with deterministic expert demonstrations.

In the Acrobot experiment better performance is achieved than when pre-training and performing latent policy learning which is surprising given that our analysis in 4.4 suggests that the expert states do not exhibit a high correlation between transitions and actions. Overall, the baseline with no pre-training is able to achieve expert performance. One reason is the clustering shown in Fig. A.2d which does not capture the true actions in Fig. A.2b well and pre-training with this sub-optimal clustering leads to worse performance overall. In the CartPole experiment, expert performance was achieved in all cases which is expected because the obtained transition clusters shown in Fig. A.2l are able to recover the true action labels. In the MountainCar experiment, the performance is not significantly better than a random policy, however still better than the baseline ILPO which suffers strongly from mode collapse in this environment slightly worse than pre-training + latent policy learning. The performance in the LunarLander experiment is on par with baseline ILPO and much worse than ILPO + pre-training. This is also due to how well the clustering is able to recover the true action information. Lastly, the Pong experiment shows that pre-training performs better than the ILPO baseline but worse than the combination ILPO with noisy expert demonstrations + pre-training.

In conclusion, the pre-training alone is often capable to prime the latent policy to achieve some sensible imitation behavior. Especially the results in the MountainCar and LunarLander pre-training only experiments show that pre-training prepares the latent policy and latent dynamics to prevent latent policy collapse. The latent policy learning then further refines the latent policy.