AUTOREGRESSIVE DYNAMICS MODELS FOR OFFLINE POLICY EVALUATION AND OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Standard dynamics models for continuous control make use of feedforward computation and assume that different dimensions of the next state and reward are conditionally independent given the current state and action. This modeling choice may be driven by the fact that fully observable physics-based simulation environments entail deterministic transition dynamics. In this paper, we question this conditional independence assumption and propose a family of expressive *autoregressive dynamics models* that generate different dimensions of the next state and reward sequentially conditioned on previous dimensions. We demonstrate that autoregressive dynamics models indeed outperform standard feedforward models in log-likelihood on heldout trajectories. Furthermore, we compare different model-based and model-free off-policy evaluation (OPE) methods on RL Unplugged, a suite of offline MuJoCo datasets, and find that autoregressive dynamics models consistently outperform all baselines, achieving a new state-of-the-art. Finally, we show that autoregressive dynamics models are useful for offline policy optimization by serving as a way to enrich the replay buffer through data augmentation and improving performance using model-based planning.

1 INTRODUCTION

Model-based Reinforcement Learning (RL) aims to learn an approximate model of the environment's dynamics from existing logged interactions to facilitate efficient policy evaluation and optimization. Early work on Model-based RL uses simple tabular (Sutton, 1990; Moore and Atkeson, 1993; Peng and Williams, 1993) and locally linear (Atkeson et al., 1997) dynamics models, which often result in a large degree of model bias (Deisenroth and Rasmussen, 2011). Recent work adopts feedforward neural networks to model complex transition dynamics and improve generalization to unseen states and actions, achieving a high level of performance on standard RL benchmarks (Chua et al., 2018; Wang et al., 2019). However, standard feedforward dynamics models assume that different dimensions of the next state and reward are conditionally independent given the current state and action, which may lead to a poor estimation of uncertainty and unclear effects on RL applications.

In this work, we propose a new family of expressive *autoregressive dynamics models* and study their effectiveness for off-policy evaluation (OPE) and offline policy optimization on continuous control. Autoregressive dynamics models generate each dimension of the next state conditioned on previous dimensions of the next state, in addition to the current state and action (see Figure 1). This means that to sample the next state from an autoregressive dynamics model, one needs n sequential steps where n is the number of state dimensions, in addition to one more step to generate the reward. By contrast, standard feedforward dynamics models take current state and action as input and generate the next state and reward in one feedforward computation (*e.g.,* Chua et al. (2018)). Autoregressive generative models have seen success in generating natural images (Parmar et al., 2018), text (Brown et al., 2020), and speech (Oord et al., 2016), but they have not seen use in Model-based RL for continuous control.

We find that autoregressive dynamics models offer a better inductive bias for modeling transition dynamics compared to their feedforward counterparts, achieving higher log-likelihood on heldout validation trajectories of all DM continuous control tasks (Tassa et al., 2018) from the RL Unplugged dataset (Gulcehre et al., 2020). To determine the impact of improved transition dynamics models, we primarily focus on OPE because it allows us to isolate contributions of the dynamics model in value estimation *vs.* the many other factors of variation in policy optimization and data collection. We find that autoregressive dynamics models consistently outperform existing Model-based and Model-free OPE baselines on continuous control in both ranking and value estimation metrics. We expect that our

advances in model-based OPE will improve offline policy selection for offline RL (Paine et al., 2020). To test this, we show that our autoregressive dynamics models can also be used to improve offline policy optimization by model predictive control, achieving a new state-of-the-art on Cheetah-run and Fish-swim from RL Unplugged (Gulcehre et al., 2020).

Key contributions of this paper include:

- We propose autoregressive dynamics models to capture dependencies between state dimensions in forward prediction. We show that autoregressive models improve log-likelihood over non-autogressive models for continuous control tasks from the DM Control Suite (Tassa et al., 2018).
- We apply autoregressive dynamics models to Off-Policy Evaluation (OPE), surpassing the performance of state-of-the art baselines in median absolute error, rank correlation, and normalized top-5 regret across 9 control tasks.
- We show that autoregressive dynamics models are more useful than feedforward models for offline policy optimization, serving as a way to enrich experience replay by data augmentation and improving performance via model-based planning.

2 PRELIMINARIES

Here we introduce relevant notation and discuss off-policy (offline) policy evaluation (OPE). We refer the reader to Lange et al. (2012) and Levine et al. (2020) for background on offline RL, which is also known as batch RL in the literature.

A finite-horizon Markov Decision Process (MDP) is defined by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, d_0, r, \gamma)$, where $\mathcal{S}$ is a set of states $s \in \mathcal{S}$, $\mathcal{A}$ is a set of actions $a \in \mathcal{A}$, $\mathcal{T}$ defines transition probability distributions $p(s_{t+1}|s_t, a_t)$, d_0 defines the initial state distribution $d_0 \equiv p(s_0)$, r defines a reward function $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, and γ is a scalar discount factor. A policy $\pi(a \mid s)$ defines a conditional distribution over actions conditioned on states. A trajectory consists of a sequence of states and actions $\tau = (s_0, a_0, s_1, a_1, \ldots, s_H, a_H)$ of horizon length H. We use $s_{t,i}$ to denote the i-th dimension of the state at time step t (and similarly for actions). In reinforcement learning, the objective is to maximize the expected sum of discounted rewards over the trajectory distribution induced by the policy:

$$V_\gamma(\pi) = \mathbb{E}_{\tau \sim p_\pi(\tau)} \left[\sum_{t=0}^{H} \gamma^t r(s_t, a_t) \right]. \tag{1}$$

The trajectory distribution is characterized by the initial state distribution, policy, and transition probability distribution:

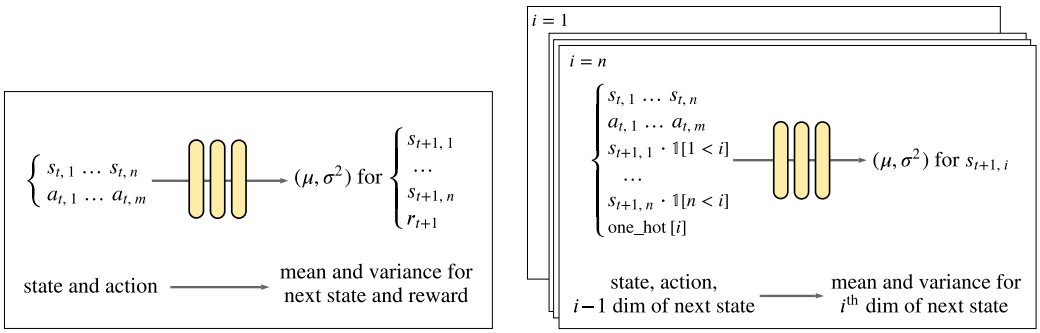
$$p_\pi(\tau) = d_0(s_0) \prod_{t=0}^{H} \pi(a_t|s_t) p(s_{t+1}|s_t, a_t). \tag{2}$$

In offline RL, we are given access to a dataset of transitions $\mathcal{D} = \{(s_t^i, a_t^i, r_{t+1}^i, s_{t+1}^i)\}_{i=1}^N$ and a set of initial states $\mathcal{S}_0$. Offline RL is inherently a data-driven approach since the agent needs to optimize the same objective as in Eq. (1) but is not allowed additional interactions with the environment. Even though offline RL offers the promise of leveraging existing logged datasets, current offline RL algorithms (Fujimoto et al., 2019; Agarwal et al., 2020; Kumar et al., 2019) are typically evaluated using online interaction, which limits their applicability in the real world.

The problem of off-policy (offline) policy evaluation (OPE) entails estimating $V_\gamma(\pi)$, the value of a target policy π, based on a fixed dataset of transitions denoted $\mathcal{D}$, without access to the environment's dynamics. Some OPE methods assume that $\mathcal{D}$ is generated from a known behavior (logging) policy μ and assume access to μ in addition to $\mathcal{D}$. In practice, the logged dataset $\mathcal{D}$ may be the result of following some existing system that does not have a probabilistic form. Hence, we will assume no access to the original behavior policy μ for OPE. That said, for methods that require access to μ, we train a behavior cloning policy on $\mathcal{D}$.

3 PROBABILISTIC DYNAMICS MODELS

Feedforward dynamics model. In the context of our paper, we use the term "model" to jointly refer to the forward dynamics model $p_s(s_{t+1}|s_t, a_t)$ and reward model $p_r(r_{t+1}|s_t, a_t)$. We use neural nets to parameterize both distributions since they are powerful function approximators that have been effective for model-based RL (Chua et al., 2018; Nagabandi et al., 2018; Janner et al., 2019).



Standard Feedforward Dynamics Models

Proposed Autoregressive Dynamics Model

Figure 1: Standard probabilistic dynamics models (*e.g.,* Chua et al. (2018)) use a neural network to predict the mean and standard deviation of different dimensions of the next state and reward simultaneously. By contrast, we use the same neural network architectures with several additional inputs and predict the mean and standard deviation of each dimension of the next state conditional on previous dimensions of the next state. As empirical results indicate, this small change makes a big difference in the expressive power of dynamics models. Note that reward prediction is not shown on the right to reduce clutter, but it can be thought of as $(n+1)$th state dimension.

Let θ denote the parameters of a fully connected network used to model $p_\theta(s_{t+1}, r_{t+1} \mid s_t, a_t)$. We expect joint modeling of the next state and reward to benefit from sharing intermediate network features. Similar to prior work (Janner et al., 2019), our baseline feedforward model outputs the mean and log variance of all state dimensions and reward simultaneously, as follows:

$$p_\theta(s_{t+1}, r_{t+1} \mid s_t, a_t) \;=\; \mathcal{N}\big(\mu(s_t, a_t), \mathrm{Diag}(\exp\{l(s_t, a_t)\})\big) \;, \tag{3}$$

where $\mu(s_t, a_t) \in \mathbb{R}^{n+1}$ denotes the mean for the concatenation of the next state and reward, $l(s_t, a_t) \in \mathbb{R}^{n+1}$ denotes the log variance, and $\mathrm{Diag}(v)$ is an operator that creates a diagonal matrix with the main diagonal specified by the vector v. During training, we seek to minimize the negative log likelihood of the parameters given observed transitions in the dataset $\mathcal{D}$:

$$\ell(\theta \mid \mathcal{D}) \;=\; -\sum\nolimits_{(s,a,r',s') \in \mathcal{D}} \log p_\theta(s', r' \mid s, a) \;. \tag{4}$$

While it is possible to place different weights on the loss for next state and reward prediction, we did not apply any special weighting and treated the reward as an additional state dimension in all of our experiments. This is straightforward to implement and does not require tuning an additional hyperparameter, which is challenging for OPE. Note that the input has $|s| + |a|$ dimensions.

Autoregressive dynamics model. We now describe our autoregressive model. We seek to demonstrate the utility of predicting state dimensions in an autoregressive way. Therefore, rather than using a complex neural network architecture, where improvements in log-likelihood and policy evaluation are confounded by architectural differences, we opt to make simple modifications to the feedforward model described above. This allows us to isolate the source of performance improvements.

The autoregressive model we use is a fully connected model that predicts the mean and log variance of a single state dimension. We augment the input space of the baseline with the previous predicted state dimensions and a one-hot encoding to indicate which dimension to predict. This is illustrated in Figure 1. The autoregressive model therefore has $3|s| + |a|$ input dimensions. Hence, the autoregressive model has a small number of additional weights in the first fully connected layer, but as will be shown in our experiments, these extra parameters are not the reason for a performance gain.

At training time, the autoregressive model has a similar computational cost to the fully connected model as we can mask ground truth states and use data parallelism to compute all state dimensions simultaneously. At inference, the autoregressive model requires additional forward passes, on the order of the number of state dimensions in a given environment. We use the default ordering for the state dimensions in a given environment, though it is interesting to explore different orderings in future works. The negative log-likelihood for an autoregressive model takes the form of:

$$\ell(\theta \mid \mathcal{D}) = -\sum\nolimits_{(s,a,r',s') \in \mathcal{D}} \Big[\log p_\theta(r' \mid s, a, s') + \sum\nolimits_{i=1}^{n} \log p_\theta(s'_i \mid s, a, s'_1, \dots, s'_{i-1})\Big] \;, \tag{5}$$

where we use chain rule to factorize the joint probability of $p(s', r' \mid s, a)$.

The main advantage of the autoregressive model is that it makes no conditional independence assumption between next state dimensions. This class of models can therefore capture non-unimodal dependencies, *e.g.,* between different joint angles of a robot. Paduraru (2007) demonstrates this increased expressivity in the tabular setting, constructing an example on which a model assuming conditional independence fails. While the expressive power of autoregressive models have been shown in various generative models (Parmar et al., 2018; Oord et al., 2016), autoregressive dynamics models have not seen much use in Model-based RL for continuous control before this work.

Model-based OPE. Once a dynamics model is trained from offline data, OPE can be performed in a direct and primitive way. We let the policy and model interact—the policy generates the next action, the model plays the role of the environment and generates the next state and reward. Due to the stochasticity in the model and the policy, we estimate the return for a policy with Monte-Carlo sampling and monitor standard error. See Algorithm 1 for a pseudocode.

Algorithm 1 Model-based OPE

Require: Number of rollouts n, discount factor γ, horizon length H, policy π, dynamics model p, set of initial states S_0
  for $i = 1, 2, \ldots n$ **do**
    $R_i \leftarrow 0$
    sample initial state $s_0 \sim \mathcal{S}_0$
    for $t = 0, 1, 2, \ldots, H-1$ **do**
      sample from policy: $a_t \sim \pi(a_t \mid s_t)$
      sample from the dynamics model:
        $s_{t+1}, r_{t+1} \sim p(\cdot, \cdot \mid s_t, a_t)$
      $R_i \leftarrow R_i + \gamma^t r_{t+1}$
    end for
  end for
  return $\frac{1}{n}\sum_{i=1}^{n} R_i$

4 Related Work

Our work follows a long line of research OPE, which is especially relevant to many practical domains such as medicine (Murphy et al., 2001), recommendation systems (Li et al., 2011), and education (Mandel et al., 2014) in order to avoid the costs and risks associated with online evaluation. There exists a large body of work on OPE, including methods based on importance weighting (Precup, 2000; Li et al., 2014) and Lagrangian duality (Nachum et al., 2019; Yang et al., 2020; Uehara and Jiang, 2019). The model-based approach that we focus on in this paper lies within the class of algorithms referred to as the *direct method* (Kostrikov and Nachum, 2020; Dudík et al., 2011; Voloshin et al., 2019), which approximate the value of a new policy by either explicitly or implicitly estimating the transition and reward functions of the environment. While model-based policy evaluation has been considered by previous works (Paduraru, 2007; Thomas and Brunskill, 2016a; Hanna et al., 2017), it has largely been confined to simple domains with finite state and action spaces where function approximation is not necessary. By contrast, our work provides an extensive demonstration of model-based OPE in challenging continuous control benchmark domains. Previous instances of the use of function approximation for model-based OPE (Hallak et al., 2015) impose strong assumptions on the probabilistic dynamics models, such as factorability of the MDP. Our results indicate that even seemingly benign assumptions about the independence of different state dimensions can have detrimental consequences for the effectiveness of a model-based OPE estimate.

While the use of model-based principles in OPE has been relatively rare, it has been more commonly used for policy optimization. The field of model-based RL has matured in recent years to yield impressive results for both online (Nagabandi et al., 2018; Chua et al., 2018; Kurutach et al., 2018; Janner et al., 2019) and offline (Matsushima et al., 2020; Kidambi et al., 2020; Yu et al., 2020; Argenson and Dulac-Arnold, 2020) policy optimization. Several of the techniques we employ, such as the normalization of the observation space, are borrowed from this previous literature (Nagabandi et al., 2018; Chua et al., 2018). Conversely, we present strong empirical evidence that the benefits of our introduced autoregressive generative models of state observations do carry over to model-based policy optimization, at least in the offline setting, and this is an interesting avenue for future work.

5 Results

We conduct our experiments on the DeepMind control suite (Tassa et al., 2018), a set of control tasks implemented in MuJoCo (Todorov et al., 2012). We use the offline datasets from RL Unplugged (Paine et al., 2020), the details of which are provided in Table 1. These environments capture a wide range of complexity, from 40K transitions in a 5-dimensional cartpole environment to 1.5 million transitions on complex manipulation tasks. We follow the evaluation protocol in the Deep OPE (Anonymous, 2021) benchmark and use policies generated by four different algorithms: behavioral cloning, D4PG (Barth-Maron et al., 2018), Critic Regularized Regression (Wang et al., 2020), and ABM (Siegel et al., 2019). With varied hyperparameters, these form a diverse set of policies of varying quality.

Table 1: Details about the offline datasets. Dataset size indicates the number of (s, a, r, s') tuples.

	cartpole swingup	cheetah run	finger turn hard	fish swim	humanoid run	walker stand	walker walk	manipulator insert ball	manipulator insert peg
State dim.	5	17	12	24	67	24	24	44	44
Action dim.	1	6	2	5	21	6	6	5	5
Dataset size	40K	300K	500K	200K	3M	200K	200K	1.5M	1.5M

Table 2: Negative log-likelihood on heldout validation sets for different RL Unplugged tasks (lower is better). For both family of dynamics models, we train 48 models with different hyperparameters. We report the Top-1 NLL on the top and average of Top-5 models on the bottom. On all of the tasks autoregressive dynamics models significantly outperform feedforward models in terms of NLL for both Top-1 and Top-5.

	Dynamics model architecture	cartpole swingup	cheetah run	finger turn hard	fish swim	humanoid run	walker stand	walker walk	manipulator insert ball	manipulator insert peg
Top-1	Feedforward	-6.81	-4.90	-5.58	-4.91	-3.42	-4.52	-3.84	-4.74	-4.34
	Autoregressive	-7.21	-6.36	-6.14	-5.21	-4.18	-4.73	-4.17	-5.62	-5.73
Top-5	Feedforward	-6.75	-4.85	-5.50	-4.90	-3.40	-4.49	-3.81	-4.64	-4.31
	Autoregressive	-7.14	-6.32	-5.94	-5.18	-4.15	-4.71	-4.15	-5.58	-5.29

We perform a thorough hyperparameter sweep in the experiments and use standard practice from generative modeling to improve the quality of the models. We allocate 80% of the data for training and 20% of the data for model selection. We vary the depth and width of the neural networks (number of layers $\in \{3, 4\}$, layer size $\in \{512, 1024\}$), add different amounts of to input states and actions, and consider two levels of weight decay for regularization (input noise $\in \{0, 1\text{e}{-6}, 1\text{e}{-7}\}$, weight decay $\in \{0, 1\text{e}{-6}\}$). For the choice of optimizer, we consider both Adam (Kingma and Ba, 2014) and SGD with momentum and find Adam to be more effective at maximizing log-likelihood across all tasks in preliminary experiments. We thus use Adam in all of our experiments with two learning rates $\in \{1\text{e}{-3}, 3\text{e}{-4}\}$. We decay the optimizer's learning rate linearly to zero throughout training, finding this choice to outperform a constant learning rate. Lastly, we find that longer training often improves log-likelihood results. We use 500 epochs for training final models.

For each task we consider in total 48 hyperparameter combinations (listed above) for both models and pick the best model in each model family based on validation log-likelihood. This model is then used for model-based OPE and policy optimization. Note that, in our experiments, 20% of the transitions are used only for validation, but we believe one can re-train the models with the best hyperparameter configuration on the full transition datasets to improve the results even further.

5.1 Autoregressive Dynamics Models Outperform Feedforward Models in NLL

To evaluate the effectiveness of autoregressive dynamics models compared to feedforward counterparts, Table 2 reports negative log-likelihood (NLL) on the *heldout validation* set for the best performing models from our hyperparameter sweep. For each environment, we report the NLL for the best-performing model (Top-1) and the average NLL across the Top-5 models. The autoregressive model has lower NLL on all environments, indicating that it generalizes better to unseen data.

To study the impact of model size on NLL, Figure 2 shows validation NLL as a function of parameter count. We find that on small datasets large models hurt, but more importantly autoregressive models outperform feedforward models regardless of the parameter count regime, *i.e.,* even small autoregressive models attain a lower validation NLL compared to big feedforward models. This indicates that autoregressive models have a better inductive bias in modeling the transition dynamics than feedforward models that make a conditional independence assumption.

5.2 Are Dynamics Models with Lower NLL better for Model-based OPE?

We ultimately care not about the log-likelihood numbers, but whether or not the dynamics models are faithful in policy evaluation and optimization. To study the relationship of NLL and OPE performance for model-based methods, we compute OPE estimates via Algorithm 1 and compute the Pearson correlation between the OPE estimates and the true discounted returns. This serves as a measure of

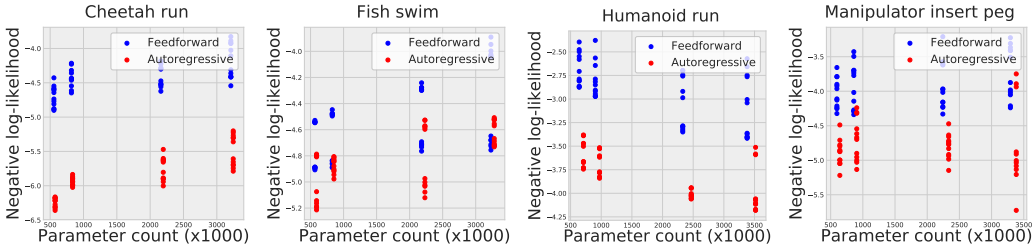


Figure 2: Validation negative log-likelihood *vs.* parameter count for autogressive and feedforward models. Autoregressive models often have a lower validation NLL irrespective of parameter count.

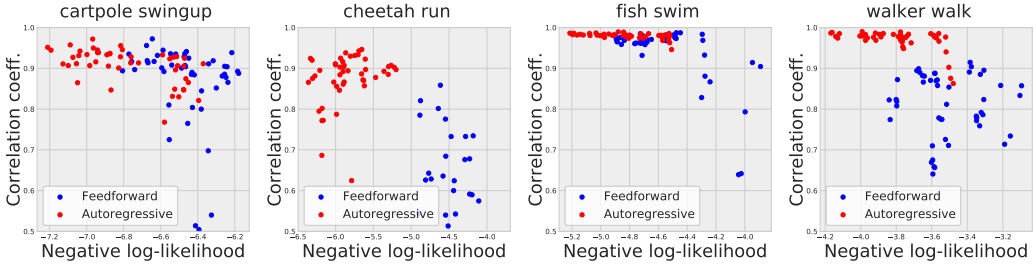


Figure 3: OPE correlation coefficients *vs.* validation NLL. On 4 RL Unplugged tasks, we conduct an extensive experiment in which 48 Autoregressive and 48 Feedforward Dynamics models are used to conduct model-based OPE. For each Dynamics model we calculate the correlation coefficient between model-based value estimates and ground truth values at a discount factor of 0.995. We find that low validation NLL numbers generally correspond to accurate policy evaluation, while higher NLL numbers are less meaningful.

the effectiveness of the model for OPE. We repeat this for all 96 dynamics models we trained on a given environment and plot the correlation coefficients against validation NLL in Figure 3.

Models with low NLL are generally more accurate in OPE. Lambert et al. (2020) have previously demonstrated that in Model-based RL, "training cost does not hold a strong correlation to maximization of episode reward." We use validation NLL instead, and our results on policy evaluation decouple the model from policy optimization suggesting a more nuanced picture: low validation NLL numbers generally correspond to accurate policy evaluation, while higher NLL numbers are generally less meaningful. In other words, if the dynamics model does not capture the transition dynamics accurately enough, then it is very hard to predict its performance on OPE. However, once the model starts to capture the dynamics faithfully, we conjecture that NLL starts to become a reasonable metric for model selection. For instance, validation NLL does not seem to be a great metric for ranking feedforward models, whereas it is more reasonable for autoregressive models.

5.3 Comparison with Other OPE Methods

We adopt a recently proposed continuous control benchmark for OPE (Anonymous, 2021) and compare our model-based approaches with state-of-the-art OPE baselines therein. Figures 4 and B.4 compare OPE estimates from two FQE baselines, our feedforward models, and the autoregressive approach. Each plot reports the Pearson correlation between the OPE estimates and the true returns. The autoregressive model consistently outperforms the feedforward model and FQE methods on most environments. We report ensembling results in the appendix, but compare single models for fairness.

We compute summary statistics for OPE methods in Table 3, Table A.1, and Table A.2. These tables report the Spearman's rank correlation, regret, and absolute error, respectively. These metrics capture different desirable properties of OPE methods (Anonymous, 2021); more details about how they are computed are in the appendix. In all three metrics, the autoregressive model achieves the best median performance across nine environments, whereas the baseline model is not as good as FQE. The only environment in which the autoregressive model has negative rank correlation is manipulator insert ball. In addition, a major advantage of our model-based approach over FQE is that the model only needs to be trained once per environment—we do not need to perform additional policy-specific optimization, whereas FQE needs to optimize a separate Q-function approximator per policy.

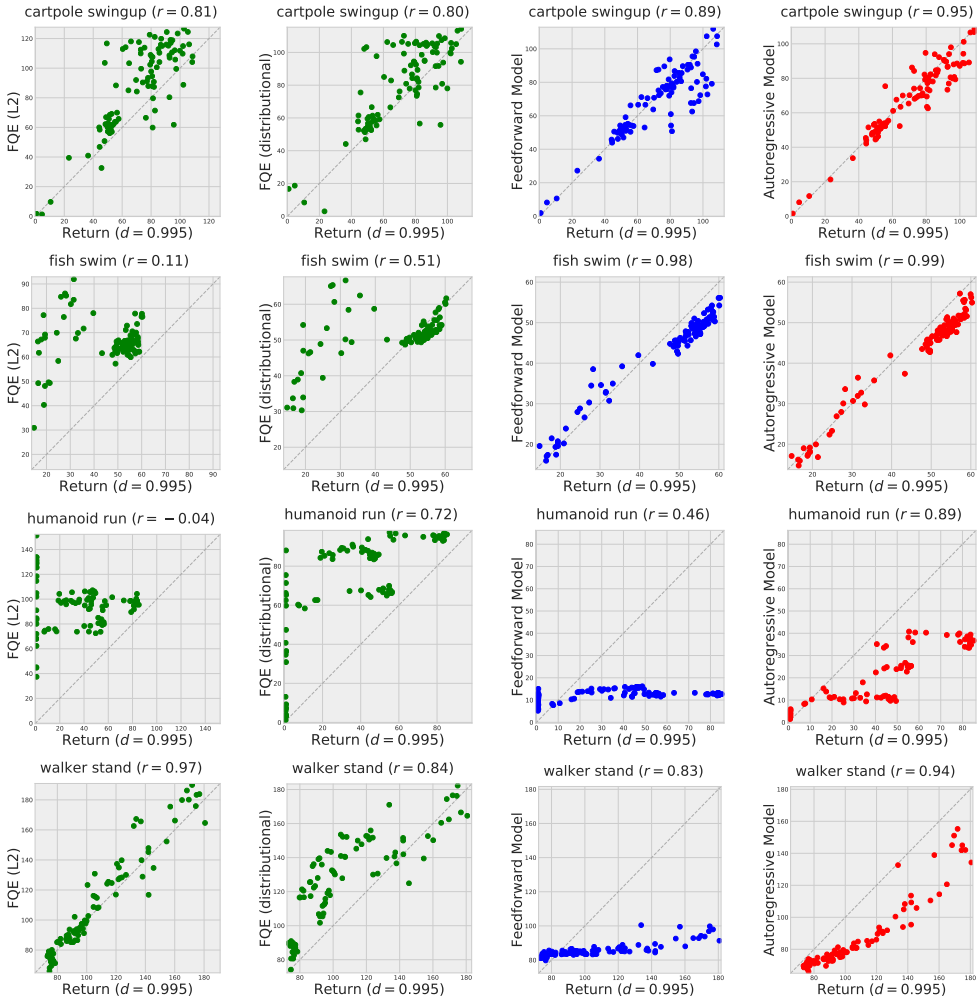


Figure 4: Comparison of model-based OPE using autoregressive and feedforward dynamics models with state-of-the-art FQE methods based on L2 and distributional Bellman error. We plot OPE estimates on the y-axis against ground truth returns with a discount of .995 on the x-axis. While feedforward models fall behind FQE on most tasks, autoregressive dynamics models are often superior. Also see Figure B.4 for additional scatter plots.

5.4 Autoregressive Dynamics Models for Offline Policy Optimization

Policy evaluation is an integral part of reinforcement learning. Improvement in policy evaluation can therefore be adapted for policy optimization. In this section, we explore two possibilities of using models to improve offline reinforcement learning. In all experiments, we use Critic Regularized Regression (CRR) as a base offline reinforcement learning algorithm (Wang et al., 2020).

First, we utilize the model during test time for planning by using a modified version of Model Predictive Path Integral (MPPI) (Williams et al., 2015). Unlike MPPI, we truncate the planning process after 10 steps of rollout and use the CRR critic to evaluate future discounted returns. We provide additional details in the appendix. Secondly, we use the model to augment the transition dataset to learn a better critic for CRR. More precisely, given $s_t^i \sim \mathcal{D}$, and the current policy π, we can generate additional data using the following process: $\hat{a}_t^i \sim \pi(\cdot|s_t^i), \quad \hat{s}_{t+1}^i, \hat{r}_{t+1}^i \sim p(\cdot, \cdot|s_t^i, \hat{a}_t)$.

These two options are orthogonal and can be applied jointly. We implemented both techniques on top of the CRR *exp* variant (Wang et al., 2020) and show their combined effect in Figure 5. The figure shows that autoregressive dynamics models also outperform feedforward ones in the policy optimization context. Notably, in the case of cheetah run and fish swim, using autoregressive models for planning as well as data augmentation enables us to outperform the previous state-of-the-art on these offline datasets. Additionally, when using autoregressive dynamics models, both techniques improve performance. In the appendix, we show this result as well as more ablations.

		Cartpole swingup	Cheetah run	Finger turn hard	Fish swim	Humanoid run
Rank Correlation btw. OPE and ground truth	Importance Sampling	-0.23 ± 0.11	-0.01 ± 0.12	-0.45 ± 0.08	-0.17 ± 0.11	$\mathbf{0.91 \pm 0.02}$
	Best DICE	-0.16 ± 0.11	0.07 ± 0.11	-0.22 ± 0.11	0.44 ± 0.09	-0.10 ± 0.10
	Variational power method	0.01 ± 0.11	0.01 ± 0.12	-0.25 ± 0.11	0.56 ± 0.08	0.36 ± 0.09
	Doubly Robust (IS, FQE)	0.55 ± 0.09	0.56 ± 0.08	0.67 ± 0.05	0.11 ± 0.12	-0.03 ± 0.12
	Feedforward Model	0.83 ± 0.05	$\mathbf{0.64 \pm 0.08}$	0.08 ± 0.11	$\mathbf{0.95 \pm 0.02}$	0.35 ± 0.10
	FQE (distributional)	0.69 ± 0.07	$\mathbf{0.67 \pm 0.06}$	$\mathbf{0.94 \pm 0.01}$	0.59 ± 0.10	0.74 ± 0.06
	FQE (L2)	0.70 ± 0.07	0.56 ± 0.08	0.83 ± 0.04	0.10 ± 0.12	-0.02 ± 0.12
	Autoregressive Model	$\mathbf{0.91 \pm 0.02}$	$\mathbf{0.74 \pm 0.07}$	0.57 ± 0.09	$\mathbf{0.96 \pm 0.01}$	$\mathbf{0.90 \pm 0.02}$

		Walker stand	Walker walk	Manipulator insert ball	Manipulator insert peg	Median ↑
Rank Correlation btw. OPE and ground truth	Importance Sampling	0.59 ± 0.08	0.38 ± 0.10	-0.72 ± 0.05	-0.25 ± 0.08	-0.17
	Best DICE	-0.11 ± 0.12	-0.58 ± 0.08	0.19 ± 0.11	-0.35 ± 0.10	-0.11
	Variational power method	-0.35 ± 0.10	-0.10 ± 0.11	$\mathbf{0.61 \pm 0.08}$	$\mathbf{0.41 \pm 0.09}$	0.01
	Doubly Robust (IS, FQE)	0.88 ± 0.03	0.85 ± 0.04	0.42 ± 0.10	-0.47 ± 0.09	0.55
	Feedforward Model	0.82 ± 0.04	0.80 ± 0.05	0.06 ± 0.10	-0.56 ± 0.08	0.64
	FQE (distributional)	0.87 ± 0.02	0.89 ± 0.03	$\mathbf{0.63 \pm 0.08}$	-0.23 ± 0.10	0.69
	FQE (L2)	$\mathbf{0.96 \pm 0.01}$	0.94 ± 0.02	$\mathbf{0.70 \pm 0.07}$	-0.48 ± 0.08	0.70
	Autoregressive Model	$\mathbf{0.96 \pm 0.01}$	$\mathbf{0.98 \pm 0.00}$	-0.33 ± 0.09	$\mathbf{0.47 \pm 0.09}$	0.90

Table 3: Spearman's rank correlation (ρ) coefficient (bootstrap mean $\pm$ standard deviation) between different OPE metrics and ground truth values at a discount factor of 0.995. In each column, rank correlation coefficients that are not significantly different from the best ($p > 0.05$) are bold faced. Methods are ordered by median. Also see Table A.1 and Table A.2 for Normalized Regret@5 and Average Absolute Error results.

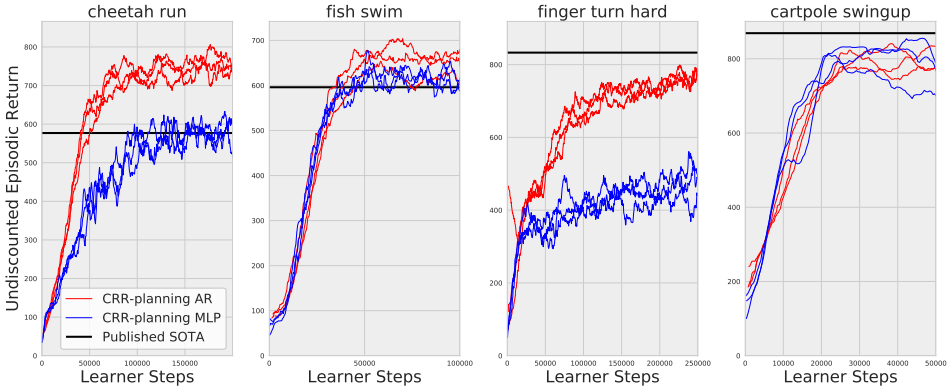


Figure 5: Policy optimization results. With planning and data augmentation, we manage to improve the performance over CRR exp (our baseline algorithm). When using autoregressive dynamics models (CRR-planning AR), we outperform state-of-the-art on Cheetah run and Fish swim. Previous SOTA results (Gulcehre et al., 2020; Wang et al., 2020) are obtained using different offline RL algorithms: Cheetah run - *CRR exp*, Fish swim - *CRR binary max*, Finger turn hard - *CRR binary max*, Cartpole swingup - *BRAC* (Wu et al., 2019).

6 Conclusion

This paper shows the promise of autoregressive models in learning transition dynamics for continuous control. We show strong empirical results from the use of autoregressive models for off-policy policy evaluation and offline policy optimization. Our contributions to offline model-based policy optimization are orthogonal to prior work that uses ensembles to lower the values when ensemble components disagree (Kidambi et al., 2020). Incorporating conservative value estimation into our method is an interesting avenue for future research. Further, better understanding of the source of improvements in our autoregressive dynamics models, even when transition dynamics are deterministic, is left for future investigation. We use relatively primitive autoregressive neural architectures in this paper to enable a fair comparison with existing feedforward dynamics models. That said, it will be exciting to apply more sophisticated autoregressive neural network architectures with cross attention (Bahdanau et al., 2014) and self-attention (Vaswani et al., 2017) to Model-based RL.

REFERENCES

Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. *International Conference on Machine Learning*, 2020.

Anonymous. Benchmarks for deep off-policy evaluation. In *Submitted to International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=kWSeGEeHvF8`. under review.

Arthur Argenson and Gabriel Dulac-Arnold. Model-based offline planning. *arXiv:2008.05556*, 2020.

Christopher G Atkeson, Andrew W Moore, and Stefan Schaal. Locally weighted learning. In *Lazy learning*, pages 11–73. Springer, 1997.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Gabriel Barth-Maron, Matthew W Hoffman, David Budden, Will Dabney, Dan Horgan, Dhruva Tb, Alistair Muldal, Nicolas Heess, and Timothy Lillicrap. Distributed distributional deterministic policy gradients. *arXiv:1804.08617*, 2018.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *arXiv:2005.14165*, 2020.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems*, 2018.

Marc Deisenroth and Carl E Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pages 465–472, 2011.

Miroslav Dudík, John Langford, and Lihong Li. Doubly robust policy evaluation and learning. *arXiv:1103.4601*, 2011.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pages 2052–2062, 2019.

Caglar Gulcehre, Ziyu Wang, Alexander Novikov, Tom Le Paine, Sergio Gómez Colmenarejo, Konrad Zolna, Rishabh Agarwal, Josh Merel, Daniel Mankowitz, Cosmin Paduraru, et al. Rl unplugged: Benchmarks for offline reinforcement learning. *arXiv:2006.13888*, 2020.

Assaf Hallak, François Schnitzler, Timothy Mann, and Shie Mannor. Off-policy model-based learning under unknown factored dynamics. In *International Conference on Machine Learning*, pages 711–719, 2015.

Josiah Hanna, Scott Niekum, and Peter Stone. Importance sampling policy evaluation with an estimated behavior policy. In *International Conference on Machine Learning*, pages 2605–2613. PMLR, 2019.

Josiah P Hanna, Peter Stone, and Scott Niekum. Bootstrapping with models: Confidence intervals for off-policy evaluation. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems*, 2019.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. MOReL : Model-based offline reinforcement learning. *arXiv:2005.05951*, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014.

Ilya Kostrikov and Ofir Nachum. Statistical bootstrapping for uncertainty estimation in off-policy evaluation, 2020.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, pages 11784–11794, 2019.

Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-Ensemble Trust-Region Policy Optimization. In *International Conference on Learning Representations*, 2018.

Nathan Lambert, Brandon Amos, Omry Yadan, and Roberto Calandra. Objective mismatch in model-based reinforcement learning. *arXiv:2002.04523*, 2020.

Sascha Lange, Thomas Gabel, and Martin Riedmiller. Batch reinforcement learning. In *Reinforcement learning*, pages 45–73. Springer, 2012.

Hoang M Le, Cameron Voloshin, and Yisong Yue. Batch policy learning under constraints. *arXiv preprint arXiv:1903.08738*, 2019.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv:2005.01643*, 2020.

Lihong Li, Wei Chu, John Langford, and Xuanhui Wang. Unbiased offline evaluation of contextual-bandit-based news article recommendation algorithms. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 297–306. ACM, 2011.

Lihong Li, Remi Munos, and Csaba Szepesvári. On minimax optimal offline policy evaluation. *arXiv:1409.3653*, 2014.

Qiang Liu, Lihong Li, Ziyang Tang, and Dengyong Zhou. Breaking the curse of horizon: Infinite-horizon off-policy estimation. In *Advances in Neural Information Processing Systems*, pages 5356–5366, 2018.

Travis Mandel, Yun-En Liu, Sergey Levine, Emma Brunskill, and Zoran Popovic. Offline policy evaluation across representations with applications to educational games. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*, pages 1077–1084. International Foundation for Autonomous Agents and Multiagent Systems, 2014.

Tatsuya Matsushima, Hiroki Furuta, Yutaka Matsuo, Ofir Nachum, and Shixiang Gu. Deployment-efficient reinforcement learning via model-based offline optimization. *arXiv:2006.03647*, 2020.

Andrew W Moore and Christopher G Atkeson. Memory-based reinforcement learning: Efficient computation with prioritized sweeping. In *Advances in neural information processing systems*, pages 263–270, 1993.

Susan A Murphy, Mark J van der Laan, James M Robins, and Conduct Problems Prevention Research Group. Marginal mean models for dynamic regimes. *Journal of the American Statistical Association*, 96(456):1410–1423, 2001.

Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. Dualdice: Behavior-agnostic estimation of discounted stationary distribution corrections. In *Advances in Neural Information Processing Systems*, pages 2318–2328, 2019.

Anusha Nagabandi, Gregory Kahn, Ronald S Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7559–7566. IEEE, 2018.

Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv:1609.03499*, 2016.

Cosmin Paduraru. Planning with approximate and learned models of markov decision processes. 2007.

Tom Le Paine, Cosmin Paduraru, Andrea Michi, Caglar Gulcehre, Konrad Zolna, Alexander Novikov, Ziyu Wang, and Nando de Freitas. Hyperparameter selection for offline reinforcement learning. *arXiv:2007.09055*, 2020.

Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Łukasz Kaiser, Noam Shazeer, Alexander Ku, and Dustin Tran. Image transformer. *arXiv:1802.05751*, 2018.

Jing Peng and Ronald J Williams. Efficient learning and planning within the dyna framework. *Adaptive behavior*, 1(4):437–454, 1993.

Doina Precup. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, page 80, 2000.

Noah Siegel, Jost Tobias Springenberg, Felix Berkenkamp, Abbas Abdolmaleki, Michael Neunert, Thomas Lampe, Roland Hafner, Nicolas Heess, and Martin Riedmiller. Keep doing what worked: Behavior modelling priors for offline reinforcement learning. In *International Conference on Learning Representations*, 2019.

Richard S Sutton. Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Machine learning proceedings 1990*, pages 216–224. Elsevier, 1990.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv:1801.00690*, 2018.

P. Thomas and E. Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning*, pages 2139–2148, 2016a.

Philip Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 2139–2148, 2016b.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

Masatoshi Uehara and Nan Jiang. Minimax weight and q-function learning for off-policy evaluation. *arXiv:1910.12809*, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Cameron Voloshin, Hoang M Le, Nan Jiang, and Yisong Yue. Empirical study of off-policy policy evaluation for reinforcement learning. *arXiv:1911.06854*, 2019.

Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. Benchmarking model-based reinforcement learning. *arXiv:1907.02057*, 2019.

Ziyu Wang, Alexander Novikov, Konrad Żołna, Jost Tobias Springenberg, Scott Reed, Bobak Shahriari, Noah Siegel, Josh Merel, Caglar Gulcehre, Nicolas Heess, et al. Critic regularized regression. *arXiv:2006.15134*, 2020.

Junfeng Wen, Bo Dai, Lihong Li, and Dale Schuurmans. Batch stationary distribution estimation. *arXiv preprint arXiv:2003.00722*, 2020.

Grady Williams, Andrew Aldrich, and Evangelos Theodorou. Model predictive path integral control using covariance variable importance sampling. *arXiv:1509.01149*, 2015.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv:1911.11361*, 2019.

Mengjiao Yang, Ofir Nachum, Bo Dai, Lihong Li, and Dale Schuurmans. Off-policy evaluation via the regularized lagrangian, 2020.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: Model-based offline policy optimization. *arXiv:2005.13239*, 2020.

A OFFLINE POLICY EVALUATION

We use the baseline results in Anonymous (2021). For convenience, we replicate their description of the OPE baselines and metrics.

A.1 OPE METRICS

To evaluate the OPE algorithms, we compute three different metrics between the estimated returns and the ground truth returns:

1. **Rank correlation** This metric assesses how well estimated values rank policies. It is equal to the correlation between the ranking (sorted order) by the OPE estimates and the ranking by the ground truth values.
2. **Absolute Error**: This metric measures the deviations of the estimates from the ground truth and does not directly access the usefulness for ranking.
3. **Regret@k** This metric measures how much worse the best policies identified by the estimates are than the best policy in the entire set. Regret@k is the difference between the actual expected return of the best policy in the entire set, and the actual value of the best policy in the top-k set.

A.2 OPE BASELINES

Fitted Q-Evaluation (FQE) As in Le et al. (2019), we train a neural network to estimate the value of the evaluation policy π_e by bootstrapping from $Q(s', \pi_e(s'))$. We tried two different implementations, one from Kostrikov and Nachum (2020) and another from Paine et al. (2020).

Importance Sampling (IS) We perform importance sampling with a learned behavior policy. We use the implementation from Kostrikov and Nachum (2020), which uses self-normalized (also known as weighted) step-wise importance sampling (Liu et al., 2018; Nachum et al., 2019). Since the behavior policy is not known explicitly, we learn an estimate of it via a max-likelihood objective over the dataset $\mathcal{D}$, as advocated by Hanna et al. (2019). In order to be able to compute log-probabilities when the target policy is deterministic, we add artificial Gaussian noise with standard deviation 0.01 for all deterministic target policies.

Doubly-Robust (DR) We perform weighted doubly-robust policy evaluation based on Thomas and Brunskill (2016b) and using the implementation of Kostrikov and Nachum (2020). Specifically, this method combines the IS technique above with a value estimator for variance reduction. The value estimator is learned according to Kostrikov and Nachum (2020), using deep FQE with an L2 loss function.

DICE This method uses a saddle-point objective to estimate marginalized importance weights $d^\pi(s,a)/d^{\pi_B}(s,a)$; these weights are then used to compute a weighted average of reward over the offline dataset, and this serves as an estimate of the policy's value in the MDP. We use the implementation from Yang et al. (2020) corresponding to the algorithm *BestDICE*.

Variational Power Method (VPM) This method runs a variational power iteration algorithm to estimate the importance weights $d^\pi(s,a)/d^{\pi_B}(s,a)$ without the knowledge of the behavior policy. It then estimates the target policy value using weighted average of rewards similar to the DICE method. Our implementation is based on the same network and hyperparameters for OPE setting as in Wen et al. (2020). We further tune the hyperparameters including the regularization parameter λ, learning rates α_θ and α_v, and number of iterations on the Cartpole swingup task using ground-truth policy value, and then fix them for all other tasks.

A.3 ENSEMBLING

As in Chua et al. (2018); Janner et al. (2019), we can form an ensemble using our best-performing models. We generate rollouts using the procedure detailed in Janner et al. (2019), forming an ensemble with 4 models. We see some improvement in policy evaluation results, as shown in Figure A.1. Ensembling could likely be further improved by forcing unique hyperparameter settings and seeds.

		Cartpole swingup	Cheetah run	Finger turn hard	Fish swim	Humanoid run
Regret@5 for OPE *vs.* ground truth	Importance Sampling	$0.73_{\pm 0.16}$	$0.40_{\pm 0.21}$	$0.64_{\pm 0.05}$	$0.12_{\pm 0.05}$	$\mathbf{0.31_{\pm 0.09}}$
	Best DICE	$0.68_{\pm 0.41}$	$0.27_{\pm 0.05}$	$0.44_{\pm 0.04}$	$\mathbf{0.35_{\pm 0.24}}$	$0.84_{\pm 0.22}$
	Variational power method	$0.50_{\pm 0.13}$	$0.37_{\pm 0.04}$	$0.45_{\pm 0.13}$	$\mathbf{0.02_{\pm 0.02}}$	$0.56_{\pm 0.08}$
	Doubly Robust (IS, FQE)	$0.28_{\pm 0.05}$	$\mathbf{0.09_{\pm 0.05}}$	$0.56_{\pm 0.12}$	$0.61_{\pm 0.12}$	$0.99_{\pm 0.00}$
	FQE (L2)	$0.06_{\pm 0.04}$	$0.17_{\pm 0.05}$	$\mathbf{0.30_{\pm 0.11}}$	$0.50_{\pm 0.03}$	$0.99_{\pm 0.00}$
	Feedforward Model	$\mathbf{0.02_{\pm 0.02}}$	$0.24_{\pm 0.12}$	$0.43_{\pm 0.04}$	$\mathbf{0.00_{\pm 0.00}}$	$0.44_{\pm 0.02}$
	FQE (distributional)	$\mathbf{0.03_{\pm 0.09}}$	$\mathbf{0.11_{\pm 0.09}}$	$\mathbf{0.10_{\pm 0.12}}$	$0.49_{\pm 0.06}$	$\mathbf{0.24_{\pm 0.15}}$
	Autoregressive Model	$\mathbf{0.00_{\pm 0.02}}$	$\mathbf{0.01_{\pm 0.02}}$	$0.63_{\pm 0.11}$	$\mathbf{0.03_{\pm 0.02}}$	$\mathbf{0.32_{\pm 0.06}}$

		Walker stand	Walker walk	Manipulator insert ball	Manipulator insert peg	Median ↓
Regret@5 for OPE *vs.* ground truth	Importance Sampling	$0.54_{\pm 0.11}$	$0.54_{\pm 0.23}$	$0.83_{\pm 0.05}$	$\mathbf{0.22_{\pm 0.03}}$	0.54
	Best DICE	$0.24_{\pm 0.07}$	$0.55_{\pm 0.06}$	$\mathbf{0.44_{\pm 0.07}}$	$0.75_{\pm 0.04}$	0.44
	Variational power method	$0.41_{\pm 0.02}$	$0.39_{\pm 0.02}$	$\mathbf{0.52_{\pm 0.20}}$	$0.32_{\pm 0.02}$	0.41
	Doubly Robust (IS, FQE)	$\mathbf{0.02_{\pm 0.01}}$	$\mathbf{0.05_{\pm 0.07}}$	$\mathbf{0.30_{\pm 0.10}}$	$0.73_{\pm 0.01}$	0.30
	FQE (L2)	$\mathbf{0.04_{\pm 0.02}}$	$\mathbf{0.00_{\pm 0.02}}$	$\mathbf{0.37_{\pm 0.07}}$	$0.74_{\pm 0.01}$	0.30
	Feedforward Model	$\mathbf{0.18_{\pm 0.10}}$	$\mathbf{0.03_{\pm 0.05}}$	$0.83_{\pm 0.06}$	$0.74_{\pm 0.01}$	0.24
	FQE (distributional)	$\mathbf{0.03_{\pm 0.03}}$	$\mathbf{0.01_{\pm 0.02}}$	$\mathbf{0.50_{\pm 0.30}}$	$0.73_{\pm 0.01}$	0.11
	Autoregressive Model	$\mathbf{0.04_{\pm 0.02}}$	$\mathbf{0.04_{\pm 0.02}}$	$0.85_{\pm 0.02}$	$\mathbf{0.30_{\pm 0.04}}$	0.04

Table A.1: Normalized Regret@5 (bootstrap mean $\pm$ standard deviation) for OPE methods *vs.* ground truth values at a discount factor of 0.995. In each column, normalized regret values that are not significantly different from the best ($p > 0.05$) are bold faced. Methods are ordered by median.

		Cartpole swingup	Cheetah run	Finger turn hard	Fish swim	Humanoid run
Absolute Error btw. OPE and ground truth	Variational power method	$37.53_{\pm 3.50}$	$61.89_{\pm 4.25}$	$46.22_{\pm 3.93}$	$31.27_{\pm 0.99}$	$35.29_{\pm 3.03}$
	Importance Sampling	$68.75_{\pm 2.39}$	$44.29_{\pm 1.91}$	$90.10_{\pm 4.68}$	$34.82_{\pm 1.93}$	$27.89_{\pm 1.98}$
	Best DICE	$22.73_{\pm 1.65}$	$23.35_{\pm 1.32}$	$33.52_{\pm 3.48}$	$59.48_{\pm 2.47}$	$31.42_{\pm 2.04}$
	Feedforward Model	$\mathbf{6.80_{\pm 0.85}}$	$13.64_{\pm 0.59}$	$35.99_{\pm 3.00}$	$\mathbf{4.75_{\pm 0.23}}$	$30.12_{\pm 2.40}$
	FQE (L2)	$19.02_{\pm 1.34}$	$48.26_{\pm 1.78}$	$27.91_{\pm 1.18}$	$19.82_{\pm 1.57}$	$56.28_{\pm 3.52}$
	Doubly Robust (IS, FQE)	$24.38_{\pm 2.51}$	$40.27_{\pm 2.05}$	$25.26_{\pm 2.48}$	$20.28_{\pm 1.90}$	$53.64_{\pm 3.68}$
	FQE (distributional)	$12.63_{\pm 1.21}$	$36.50_{\pm 1.62}$	$\mathbf{10.23_{\pm 0.93}}$	$7.76_{\pm 0.95}$	$32.36_{\pm 2.27}$
	Autoregressive Model	$\mathbf{5.32_{\pm 0.54}}$	$\mathbf{4.64_{\pm 0.46}}$	$22.93_{\pm 1.72}$	$\mathbf{4.31_{\pm 0.22}}$	$\mathbf{20.95_{\pm 1.61}}$

		Walker stand	Walker walk	Manipulator insert ball	Manipulator insert peg	Median ↓
Absolute Error btw. OPE and ground truth	Variational power method	$96.76_{\pm 3.59}$	$87.24_{\pm 4.25}$	$79.25_{\pm 6.19}$	$21.95_{\pm 1.17}$	46.22
	Importance Sampling	$66.50_{\pm 1.90}$	$67.24_{\pm 2.70}$	$29.93_{\pm 1.10}$	$12.78_{\pm 0.66}$	44.29
	Best DICE	$27.58_{\pm 3.01}$	$47.28_{\pm 3.13}$	$103.45_{\pm 5.21}$	$22.75_{\pm 3.00}$	31.42
	Feedforward Model	$23.34_{\pm 2.41}$	$52.23_{\pm 2.34}$	$34.30_{\pm 2.55}$	$121.12_{\pm 1.58}$	30.12
	FQE (L2)	$\mathbf{6.51_{\pm 0.71}}$	$18.34_{\pm 0.95}$	$36.32_{\pm 1.07}$	$31.12_{\pm 2.37}$	27.91
	Doubly Robust (IS, FQE)	$26.82_{\pm 2.66}$	$24.63_{\pm 1.69}$	$13.33_{\pm 1.16}$	$22.28_{\pm 2.34}$	24.63
	FQE (distributional)	$21.49_{\pm 1.41}$	$27.57_{\pm 1.54}$	$\mathbf{9.75_{\pm 1.10}}$	$12.66_{\pm 1.39}$	12.66
	Autoregressive Model	$19.12_{\pm 1.23}$	$\mathbf{5.14_{\pm 0.49}}$	$17.13_{\pm 1.34}$	$\mathbf{9.71_{\pm 0.70}}$	9.71

Table A.2: Average absolute error between OPE metrics and ground truth values at a discount factor of0.995 In each column, absolute error values that are not significantly different from the best ($p > 0.05$) are bold faced. Methods are ordered by median.

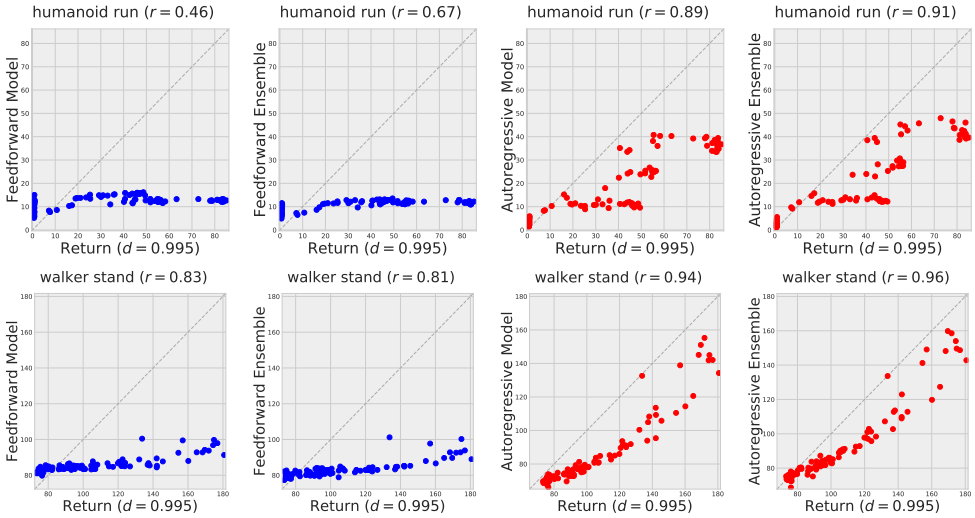


Figure A.1: Estimates of returns using the top model versus estimates of returns using the top-4 models.

Algorithm 2 Model Predictive Path Integral Planning

Require: state s, policy π, dynamics model p, critic Q, temperature β, and noise variance σ^2.
  for $m = 1, ..., M$ **do**
    for $n = 1, ..., N$ **do**
      $s^0 \leftarrow s$
      $R_n \leftarrow 0$
      for $\tau = 0, ..., H-1$ **do**
        $a_n^\tau \sim \pi(\cdot|s_n^\tau)$
        $s_n^{\tau+1}, r_n^{\tau+1} \sim \pi(\cdot, \cdot|s_n^\tau, a_n^\tau)$
        $R_n \leftarrow R_n + \gamma^\tau r_n^{\tau+1}$
      end for
      $a^H \sim \pi(\cdot|s^H)$
      $R_n \leftarrow R_n + \gamma^H Q(s_n^H, a_n^H)$
    end for
    Re-define π such that $\pi(\cdot|\hat{s}^\tau) = \sum_n \frac{\exp(R_n/\beta)}{\sum_m \exp(R_m/\beta)} \mathcal{N}(\cdot|a_n^\tau, \sigma^2 I)$. // π depends on τ and not $\hat{s}$.
  end for
  sample final action $a \sim \sum_n \frac{\exp(R_n/\beta)}{\sum_m \exp(R_m/\beta)} \delta(a_n^0)$
  return a

B Additional Details Regarding Policy Optimization

To test dynamic models for policy optimization, we implement the two methods discussed in Section 5.4 on top of CRR *exp*, one of the CRR variants (Wang et al., 2020). We use the RL Unplugged datasets Gulcehre et al. (2020) for all environments studied in this section When using data augmentation, we adopt a 1-to-1 ratio between the original dataset and the augmented dataset.

To take advantage of the dynamics models at test time, we use a variant of Model Predictive Path Integral (MPPI) for planning. To reduce the planning horizon, we truncate the model rollout using CRR critics. The details of the planning procedure is summarized in Algorithm 2. All hyperparameter tuning for the planning process is conducted on the "cartpole swingup" task. The hyperparameter used in the planning process are $M = 3, N = 16, H = 10, \beta = 0.1$, and $\sigma^2 = 0.01$. To match the temperature used in the planning component, we choose $\beta = 0.1$ for the CWP component of CRR. This change, however, does not change the baseline CRR agent performance very much. With the exception of β and the planning component, all hyperparameters are kept the same as CRR *exp*.

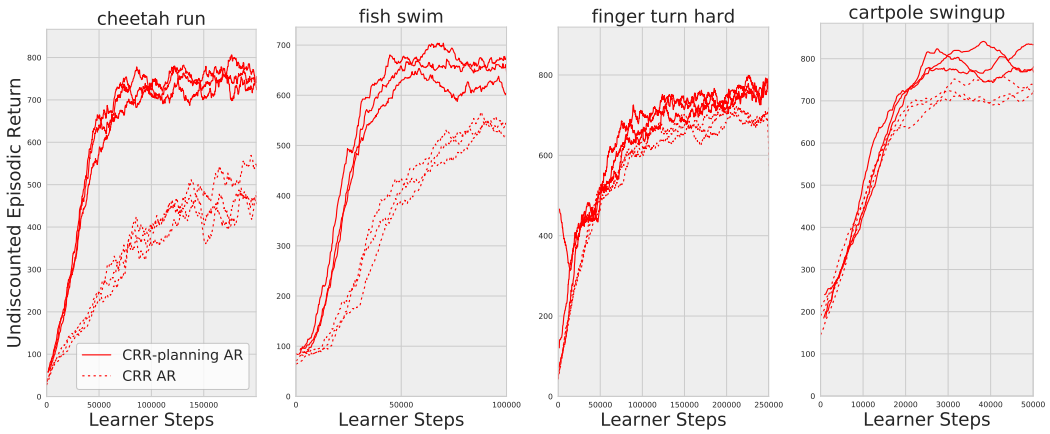


Figure B.2: Effects of the planning procedure. Here we compare using planning (CRR-planning AR) vs not ((CRR AR)) while using augmented data generated by the autoregressive model. Planning with autoregressive models helps in all environments tested.

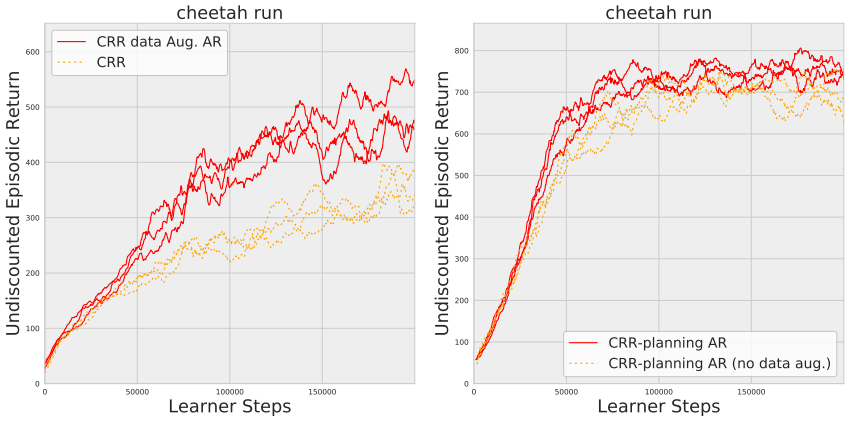


Figure B.3: Effects of the data augmentation on cheetah run. [LEFT] In the absence of planning, data augmentation significantly increase the performance of CRR agent. [RIGHT] With the planning procedure, data augmentation is still effective albeit to a lesser extent.

We compare the agents' performance with and without the planning procedure to test its effects. As shown in Figure B.2, planning using an autoregressive model significantly increases performance.

Data augmentation does not change the agents' performance on cartpole swingup, fish swim, or finger turn hard. It, however, boosts performance considerably on cheetah run. In Figure B.3, we show the effects of data augmentation on cheetah run.

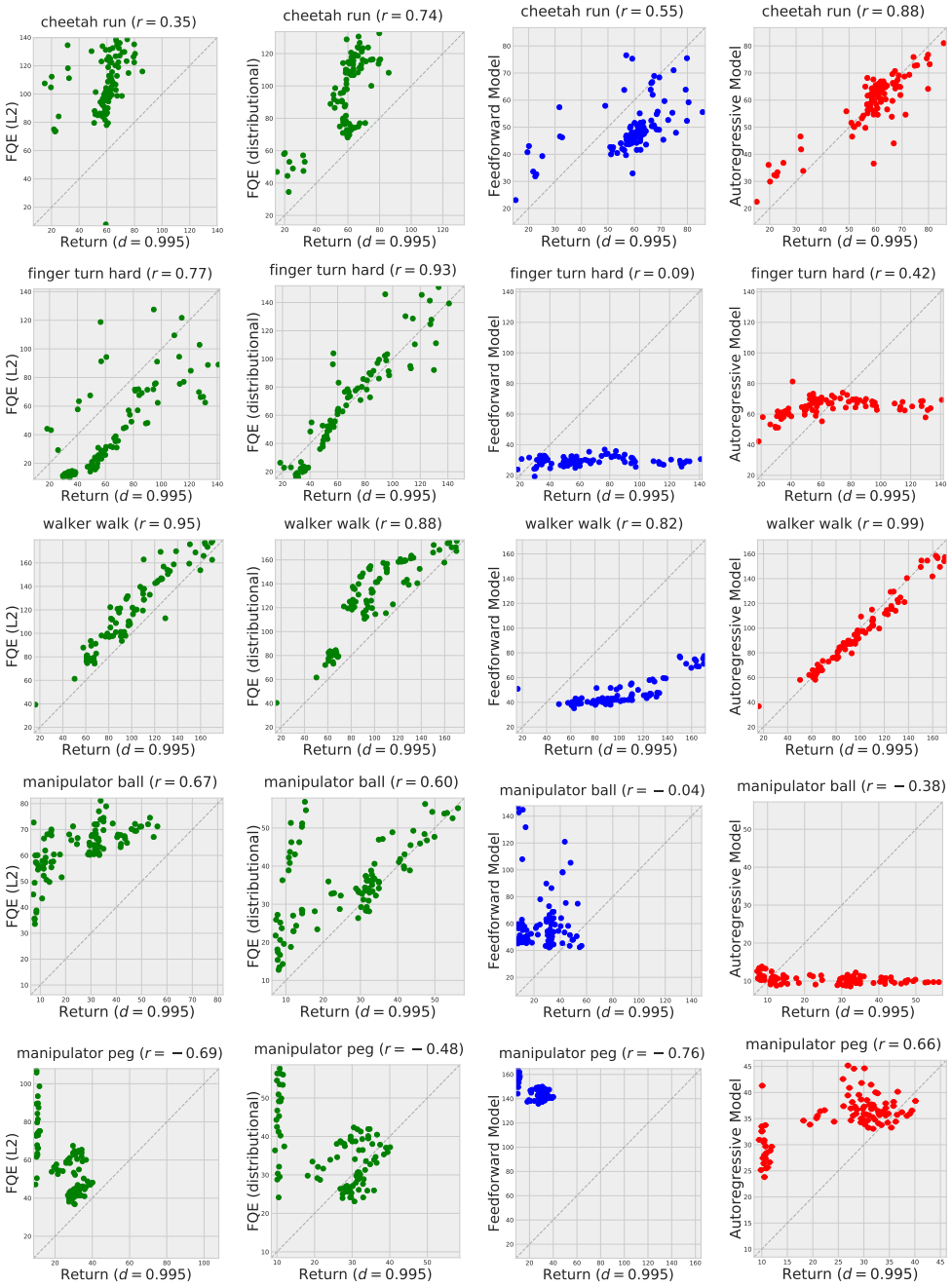


Figure B.4: Comparison of model-based OPE using autoregressive and feedforward dynamics models with state-of-the-art FQE methods based on L2 and distributional Bellman error. We plot ground truth returns on the x-axis against estimates of returns from various OPE methods on the y-axis. While feedforward models fall behind FQE on most tasks, autoregressive dynamics models are often superior. The remaining environments are plotted in Figure 4.