Offline Policy Comparison with Confidence: Benchmarks and Baselines

Anonymous authors
Paper under double-blind review

Abstract

Decision makers often wish to use offline historical data to compare sequential-action policies at various world states. Importantly, computational tools should produce confidence values for such offline policy comparison (OPC) to account for statistical variance and limited data coverage. Nevertheless, there is little work that directly evaluates the quality of confidence values for OPC. In this work, we address this issue by creating benchmarks for OPC with Confidence (OPCC), derived by adding sets of policy comparison queries to datasets from offline reinforcement learning. In addition, we present an empirical evaluation of the *risk versus coverage* trade-off for a class of model-based baselines. In particular, the baselines learn ensembles of dynamics models, which are used in various ways to produce simulations for answering queries with confidence values. While our results suggest advantages for certain baseline variations, there appears to be significant room for improvement in future work.

1 Introduction

Given historical data from a dynamic environment, how well can we make predictions about future trajectories while also quantifying the uncertainty of those predictions? Our main goal is to drive research toward a positive answer by encouraging work on a specific prediction problem, *offline policy comparison with confidence (OPCC)*. Toward this goal we contribute OPCC benchmarks with metrics that directly relate to uncertainty quantification along with a baseline pilot evaluation. The *benchmarks* [1] and *baselines* will be made publicly available.

OPCC involves using historical data to answer queries that each ask for: 1) a prediction of which of two policies is better for an initial state and horizon, where the policies, state, and horizon can be arbitrarily specified, and 2) a confidence value for the prediction. While here we use OPCC for benchmarking uncertainty quantification, it also has utility for both decision support and policy optimization. For decision support, a farm manager may want a prediction for which of two irrigation policies will best match season-level crop goals. A careful farm manager, however, would only take the prediction seriously if it comes with a meaningful measure of confidence. For policy optimization, we may want to search through policy variations to identify variations that confidently improve over others in light of historical data.

Offline reinforcement learning (ORL) (Levine et al., 2020), both for policy evaluation and optimization, offers a number of techniques relevant to decision support and OPCC in particular. One of the key ORL challenges is dealing with uncertainty due to statistical variance and limited coverage of historical data. This recognition has led to rapid progress in ORL, yielding different approaches for addressing uncertainty, e.g. pessimism in the face of uncertainty (Kumar et al., 2020; Buckman et al., 2020; Jin et al., 2021a; Shrestha et al., 2021) or regularizing policy learning toward the historical data (Fujimoto & Gu, 2021; Kostrikov et al., 2021; Kumar et al., 2019; Peng et al., 2019). However, there has been very little work on directly evaluating the uncertainty quantification capabilities embedded in these approaches. Rather, overall ORL performance is typically evaluated, which can be affected by many algorithmic choices that are not directly

[1] Benchmark python package and baselines are provided as supplementary material.

related to uncertainty quantification. A major motivation for our work is to better measure and understand the underlying uncertainty quantification embedded in popular ORL approaches.

Prior work has studied non-sequential prediction (e.g. image classification) with an abstention (or rejection) option (El-Yaniv et al., 2010; Geifman & El-Yaniv, 2017; 2019; Hendrickx et al., 2021; Xin et al., 2021; Condessa et al., 2017). Typically, these methods produce confidence values for predictions and abstain based on a confidence threshold. Ideally, if the confidence values strongly relate to prediction uncertainty, then abstentions will be biased toward the erroneous predictions. In order to directly evaluate the quality of uncertainty quantification, this line of work commonly reports measures of risk-coverage curves (RCCs) such as area under the curve (AUC) and reverse-pair proportion (RPP). To the best of our knowledge, analogous benchmarks and evaluations have not yet been established for sequential decision making. Our focus on establishing benchmarks and metrics for OPCC aims at partially filling this gap.

Contribution. The first contribution of this paper is to develop benchmarks for OPCC derived from existing ORL benchmarks and to suggest metrics for the quality of uncertainty quantification. Each benchmark includes: 1) a set of trajectory data D collected in an environment via different types of data collection policies, and 2) a set of queries Q, where each query asks which of two provided policies has a larger expected reward with respect to a specified horizon and initial states. Note that our OPCC benchmarks are related to recent benchmarks for offline policy evaluation (OPE) (Fu et al., 2021), which includes a policy ranking task similar to OPCC. That work, however, does not propose evaluation metrics and protocols for measuring uncertainty quantification over policy rankings. Further, our query sets Q span a much broader range of initial states than existing benchmarks, which is critical for understanding how uncertainty quantification varies across the wider state space as it relates to the trajectory data D. The benchmarks and baselines are publicly available with the intention of supporting community expansion over time.

Our second contribution is to present a pilot empirical evaluation of OPCC for a class of approaches that use ensembles as the mechanism to capture uncertainty, which is one of the prevalent approaches on ORL. This class uses learned ensembles of dynamics and reward models to produce Monte-Carlo simulations of each policy, which can then be compared in various ways to produce a prediction and confidence value. Our results for different variations of this class provide evidence that some variations may improve aspects of uncertainty quantification. However, overall, we did not observe sizeable and consistent improvements from most of the variations we considered. This suggests that there is significant room for future work aimed at consistent improvement for one or more of the uncertainty-quantification metrics.

2 Background

We formulate our work in the framework of Markov Decision Processes (MDPs), for which we assume basic familiarity (Puterman, 2014). An MDP is a tuple $M = \{S, A, P, R\}$, where S is the state space, A is the action space, and $P(s'|s, a)$ is the first-order Markovian transition function that gives the probability of transitioning to state s' given that action a is taken in state s. Finally, $R(s, a)$ is potentially a stochastic reward function, which returns the reward for taking action a in state s.

In this work, we focus on decision problems with a finite horizon h, where action selection can depend on the time step. A non-stationary policy $\pi(s, t)$ is a possibly stochastic function that returns an action for the specified state s and time step $t \in \{0, \ldots, h-1\}$. Given a horizon h and discount factor $\gamma \in [0, 1)$, the value of a policy π at state s, denoted $V^\pi(s, h)$, is the expected cumulative discounted future reward over the horizon:

$$V^\pi(s,h) = \mathbb{E}\left[\sum_{t=0}^{h-1} \gamma^t R\left(S_t, A_t\right) \middle| S_0 = s, A_t = \pi(S_t, t)\right]$$

where S_t and A_t are the state and action random variables at time t. It is important to note that we gain considerable flexibility by allowing for non-stationary policies. For example, π could be an open-loop policy or even a fixed sequence of actions, which are commonly used in the context of model-predictive control (Richards, 2005). Further, we can implicitly represent the action value function $Q^\pi(s, a, h)$ for a policy π by defining a new non-stationary policy π' that takes action a at $t = 0$ and then follows π thereafter, which yields $V^{\pi'}(s, h) = Q^\pi(s, a, h)$. For this reason, we will focus exclusively on comparisons in terms of state-value functions without loss of generality.

3 Offline Policy Comparison with Confidence

In this section, we first introduce the concept of policy comparison queries, which are then used to define the OPCC learning problem. Finally, we discuss the OPCC evaluation metrics used in our evaluations.

3.1 Policy Comparison Queries

We consider the fundamental decision problem of predicting the relative future performance of two policies, which we formalize via *policy comparison queries (PCQs)*. A PCQ is a tuple $q = (s, \pi, \hat{s}, \hat{\pi}, h)$, where s and $\hat{s}$ are arbitrary starting states, π and $\hat{\pi}$ are policies, and h is a horizon. The answer to a PCQ is the truth value of $V^{\pi}(s, h) < V^{\hat{\pi}}(\hat{s}, h)$. That is, a PCQ asks whether the h-horizon value of $\hat{\pi}$ is greater than π when started in $\hat{s}$ and s respectively.

As motivated in Section 1, PCQs are useful for both human-decision support and automated policy optimization. For example, if a farm manager wants information about which of two irrigation policies, π and $\hat{\pi}$, will result in the best future crop yield given the environment state s, then the corresponding PCQ would be $(s, \pi, s, \hat{\pi}, h)$. Alternatively, the manager may be interested in whether a policy π is better suited to an environmental state s or $\hat{s}$, which is captured by the PCQ $(s, \pi, \hat{s}, \pi, h)$. In addition, PCQs can be used as the basis for the classic policy improvement step of policy iteration (Puterman, 2014). In particular, we can improve over policy π at state s by identifying an action a' with higher action value than chosen by π. The corresponding PCQ for testing a' is (s, π, s, π', h), where π' is the non-stationary policy that first takes action a' and then follows π.

In practice, PCQs within an application domain need not be restricted to comparing policies via a single reward function. Rather there are often multiple quantities of interest to users. For example, a farm manager may be interested in understanding how two irrigation policies compare across multiple features of the future, such as cumulative water usage, plant stress, run off, etc. This can be facilitated by defining reward functions corresponding to each feature and issuing the appropriate PCQs.

3.2 Learning to Answer PCQs with Confidence

Given an accurate generative model of the environment MDP, a PCQ $(s, \pi, \hat{s}, \hat{\pi}, h)$ can be answered via Monte-Carlo trajectory sampling to estimate $V^{\pi}(s, h)$ and $V^{\hat{\pi}}(\hat{s}, h)$. Further, the confidence in the answer can be arbitrarily improved by increasing the number of sampled trajectories. In this work, we do not assume an environment model, but instead are provided with an offline data set of environment trajectories produced by one or more unknown behavior policies. We will denote this dataset by $\mathcal{D} = \{(s_i, a_i, s'_i, r_i)\}$ where each tuple corresponds to an observed transitions from state s_i to state s'_i after taking action a_i and receiving reward r_i.

Given a dataset $\mathcal{D}$ we would like to learn a model for predicting answers to PCQs from a query space $\mathcal{Q}$. Here, $\mathcal{Q}$ may assert application-specific restrictions on states and policies involved in PCQs. A fundamental challenge is that the coverage of $\mathcal{D}$ will not necessarily be representative of the dynamics and rewards relevant to answering all queries in $\mathcal{Q}$. Thus, if query answers are being used to inform important decisions, then it is critical for each answer to come with a meaningful measure of confidence that accounts for data coverage and statistical variance. Dealing with this uncertainty is also a core challenge for general offline RL (Levine et al., 2020), which has lead to a number of approaches for addressing it. However, there is little direct evaluation of the uncertainty-handling components.

The above motivates the *OPCC learning problem*, which provides a dataset $\mathcal{D}$ and desired constraints on the query space $\mathcal{Q}$. The learner should output a model $w = (f, c)$ composed of: 1) a query prediction function $f : \mathcal{Q} \to \{0, 1\}$, which returns a binary answer for any query in $\mathcal{Q}$, and 2) a confidence function $c : \mathcal{Q} \to [l, u]$ that maps queries in $\mathcal{Q}$ to a confidence value within a bounded interval. Given a query q, the intent is for larger values of $c(q)$ to indicate a higher confidence in the prediction $f(q)$. Note that we do not attach any predefined semantics to the values of $c(q)$ to allow for flexibility of potential solutions. Rather, we focus on defining metrics for directly evaluating the quality of uncertainty quantification provided by w. If desired, various methods can be used after learning to calibrate the confidence values of c to meaningful scales (e.g.

(Loh, 1987; Naeini et al., 2015). Section 5 discusses possible learning approach and the baselines evaluated in this paper.

3.3 Evaluation Metrics

Since OPCC involves confidence estimation for binary PCQ predictions, we can draw on evaluation metrics from prior work on selective classification (e.g. (El-Yaniv et al., 2010; Xin et al., 2021)). In selective classification, the aim is to reduce prediction errors by allowing a predictor to abstain from a prediction if the confidence is below a threshold. *The quality of confidence values is thus related to how well they result in abstaining when the prediction would have been incorrect.* This idea is formalized via *risk-coverage curves (RCCs)*, as outlined below.

Risk-Coverage Curve. Let $\mathcal{L}(q, \hat{y})$ be a loss function for predicting $\hat{y}$ for query q, e.g. 0/1 loss. Given a test set of queries $Q = \{q_1, \ldots, q_N\}$, a model $w = (f, c)$, and confidence threshold τ, the *coverage* is the fraction of test queries with confidence at least τ. The *selective risk* is the average loss of f over the covered queries. Formally, the *coverage* and *selective risk* are respectively define by

$$cov(w, Q, \tau) = \frac{1}{|Q|} \sum_{q \in Q} I[c(q) \geq \tau] \tag{1}$$

$$r(w, Q, \tau) = \frac{\sum_{q \in Q} I[c(q) \geq \tau] L(q, f(q))}{\sum_{q \in Q} I[c(q) \geq \tau]} \tag{2}$$

where I is the binary indicator function. Thus, each possible threshold corresponds to a risk-coverage operating point $\langle r(w, Q, \tau), cov(w, Q, \tau) \rangle$. An RCC is simply the risk versus coverage curve of these operating points when sweeping through possible thresholds. Practically, for a finite test set Q there can be at most $|Q|$ unique operating points since there are at most $|Q|$ distinct confidence values produced by c. Thus, when displaying empirical RCCs we linearly interpolate between those operating points.[2] Figure 1 shows an example of an RCC from our experiments. The curve starts at the point $(0, 0)$, since the risk is 0 at zero coverage, and ends at $(1, r_f)$, where r_f is the risk of f evaluated on all of Q.

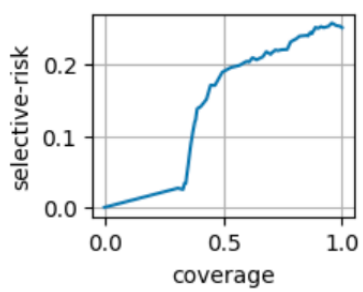


Figure 1: A sample of Risk-Coverage Curve (RCC)

In order to provide a single measure of the RCC quality, we aggregate across all thresholds to compute the *Area Under the RCC* (**AURCC**). Since lower risk is preferred, we consider a lower AURCC to indicate better confidence estimation. The minimum AURCC is 0, which occurs when the predictor f has zero risk on all of Q. Rather, for a ranomized confidence function that returns a uniformly random value in $[l, u]$, the expected AURCC is r_f[3], indicating no ability to quantify prediction uncertainty.

RPP. Our second selective-classification metric from (Xin et al., 2021) is *reverse pair proportion (RPP)*. The main idea is that the ordering of confidence values for a pair of queries should reflect the relative prediction

[2]This is justified by the fact that we can achieve (in expectation) any linearly interpolated operting point between two thresholds τ_1 and τ_2 by varying the probability $p \in [0, 1]$ of using τ_1 versus τ_2 to decide on abstention.

[3]This is because for any threshold τ and uniformly random confidence function, there is a uniform probability of covering any particular query. Thus, the expected risk for any threshold $\tau > l$ is the expected risk over a random draw from the query set, which is r_f.

loss for those queries. RPP measures how often the confidence value ordering conflicts with the relative losses across all pairs of queries. In particular, a conflict occurs when the loss of q_1 is less than q_2, but we are more confident about q_2 than q_1. The RPP is just the fraction of such conflicts.

$$RPP(w, Q) = \frac{1}{|Q|^2} \sum_{q_1, q_2 \in Q} I[l(q_1) < l(q_2), c(q_1) < c(q_2)]$$

where $l(q) = L(q, f(q))$ is the loss of f on q.

$\boldsymbol{CR_K}$**.** Finally, we introduce a new metric on just the confidence function c. A practical difference between different confidence functions is the resolution of values that they output in practice. For example, given a set of queries Q, one confidence function c_1 may result in only three distinct coverage values $cov(w, Q, \tau)$ across all thresholds, while another confidence function c_2 results in $|Q|$ distinct coverage values. All else being equal c_2 is the preferable function, since it provides a higher level of resolution with respect to abstention/coverage rates. We measure this via *coverage resolution at* K, denoted CR_K. To compute CR_K for $w = (c, f)$ and query set Q, the coverage interval $[0, 1]$ is partitioned into K equal bins and we return the fraction of bins which contain $cov(w, Q, \tau)$ for some threshold τ. By increasing K we get a finer grained distinction in coverage resolution.

4 OPCC Benchmark Construction

In this section, we describe our approach to constructing OPCC benchmarks. We first describe our choices of environments, which are based on existing offline RL benchmarks. In particular, the training datasets used in our benchmarks is based on data from those benchmarks. Next we describe our approach to constructing testing query sets for each of the benchmark environments. The outlined benchmark-construction schema is generic, which can be followed by others to extend the set of available OPCC benchmarks. A tabular summary of the benchmark can be found in Appendix A.

4.1 Environments

To support easier adoption of our benchmarks, we selected seven environments and corresponding datasets that are currently used in offline RL research. As a first set of OPCC benchmarks, we have chosen to focus on relatively low-dimensional environments with non-image-based observations. This helps focus initial studies on fundamental OPCC capabilities, rather than simultaneously addressing the additional complexities that enter with lower-level perceptual observations such as images.

Maze2d (4 environments). The Maze2d environments were introduced in D4RL (Fu et al., 2020) and comprise of 2d mazes of different complexities: *open, u-maze, medium,* and *large* as illustrated in Figure 3. Each environment has 4D observations giving the position and velocity of the ball being controlled and a 2D action space specifying the direction of movement. The goal in each environment is to control a rolling ball to reach a goal location. For our benchmarks, we used the sparse-reward version of the environments that provides unit reward for each time step in the goal region. There are no terminal states in these environments and episode ends after maximum number of allowed time-steps reported in Table 8. We use the datasets provided by D4RL, which we refer to as *"1M"* due to the datasets each having 1 million state transitions. The D4RL trajectory data sets were created by running a path-planning algorithm to navigate in the maze between different start and end points.

Gym-Mujoco (3 environments). The Gym-Mujoco environments are based on controlling the actuators of systems within the Mujoco physics simulators. We consider three locomotion-based environments from OpenAI Gym (Brockman et al., 2016): HalfCheetah, Walker2d, and Hopper. For each environment, we use the corresponding D4RL (Fu et al., 2020) datasets that include behavior trajectories of varying qualities. This includes *"random, medium, medium-replay, expert and expert-replay"*. These environments are qualitatively different from Maze2d in that they involve controlling periodic locomotion behavior based on continuous states and actions. Rather Maze2d is primarily about goal-based path planning (navigation) rather than controlling low-level locomotion.

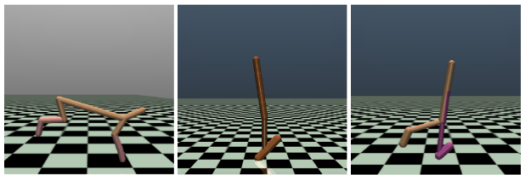


Figure 2: Gym-Mujoco tasks: half-cheetah, hopper, walker2d (left to right)

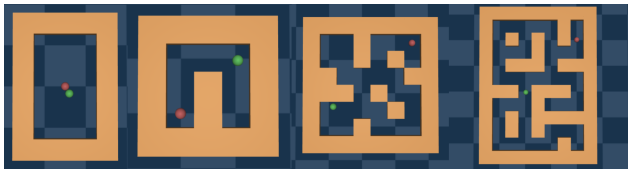


Figure 3: Maze2d tasks: open, umaze, medium, and large (left to right).

4.2 Query Set Construction

For each environment we must create a set of PCQs with ground truth answers for evaluating OPCC. A possible starting point is the off-policy evaluation (OPE) extension (Fu et al., 2021) to D4RL, which includes policies for a subset of the environments. In particular, one of the tasks considered is policy ranking, which is similar in spirit to OPCC. However, that extension of D4RL does not capture at least two important characteristics of OPCC. First, the evaluation protocols do not explicitly address measuring the quality of uncertainty quantification. Of course, this can be addressed by just extending the evaluation protocol and metrics.

Second, and more importantly, it is not clear how to define adequate query sets for evaluating OPCC. In particular, the OPE ranking task from D4RL is currently limited to just ranking policies based on their expected values over the initial state distribution of each environment. In contrast, OPCC evaluations should involve sets of PCQs that cover a wide range of states that are both in-distribution and out-of-distribution relative to the offline data set. Further, it is desirable to select the PCQs in a way that spans some notion of PCQ difficulty. In particular, the notion of difficulty we consider here for a PCQ $(s, \pi, \hat{s}, \hat{\pi}, h)$ is directly related to the performance gap between the policies, i.e $\left|V^{\pi}(s, h) - V^{\hat{\pi}}(\hat{s}, h)\right|$. It is expected that all else being equal, larger gaps will reason in easier discrimination between policies. Indeed one of the initial challenges in developing the benchmarks was to try to create query sets that were not all too easy or too hard.

Bases on above considerations we create the evaluation sets of PCQs for each environments via the following steps.

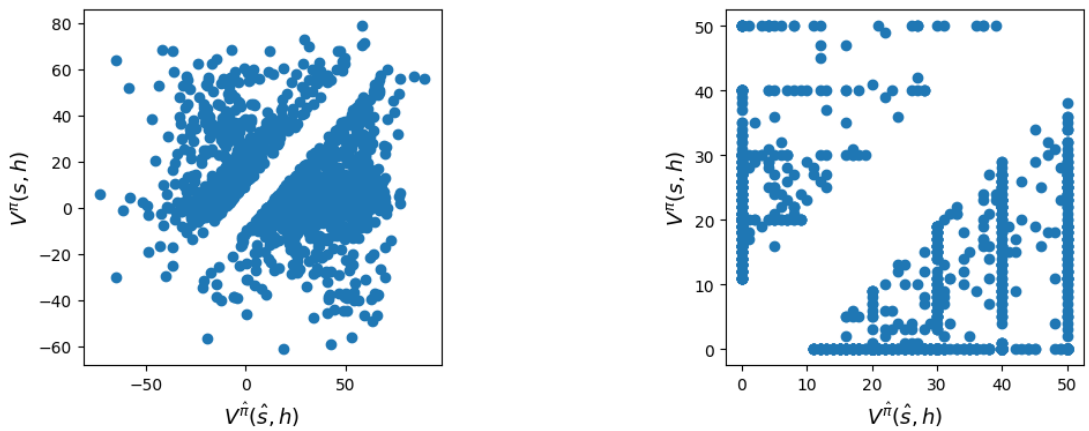
Step 1: Policy Generation. We first train multiple policies for an environment to serve as the policy used for PCQ construction. We chose to train new policies, rather than use policies from D4RL, to ensure they would be distinct from the behavior policies used to create the D4RL datasets. For each environment we used the corresponding simulator and multiple runs of the PPO algorithm (Schulman et al., 2017) to train a set of policies of varying quality. We then hand-picked 4 policies with an effort to ensure that they were sufficiently distinct in terms of quality and behavior to support non-trivial PCQs. Performance of these policies in shared in Table 9 (appendix).

Step 2: Initial State Generation. For each environment we generated a large set of potential initial states by running episodes of the random policy, the learned policies, and a mixture of random and learned policies. This produced a set of states that covered a wide range of the environment that extended well beyond the initial state distributions.

Step 3: Candidate PCQ Generation. For each horizon $h \in \{10, 20, 30, 40, 50\}$ we create a set of 2000 randomly constructed PCQs from the initial states and learned policies. This included explicitly creating random PCQs of the form $(s, \pi, s, \hat{\pi}, h)$ with s a random initial state and $\pi, \hat{\pi}$ a random pair of the learned policies. In addition, we create a set of PCQs of the form $(s, \pi, \hat{s}, \hat{\pi}, h)$ in the same way, except that two random initial states are used instead of one.

Step 4: PCQ Labeling and Selection. For each generated PCQ from step 3 we used Monte-Carlo simulation via the environment simulator to accurately estimate $V^{\pi}(s, h)$ and $V^{\hat{\pi}}(\hat{s}, h)$ and removed any PCQ having a difference of less than 10 between the value of each side of a query. The motivation is to filter out PCQs that are the most ambiguous and more likely to act as a source of noise in evaluations. Finally, for

each h, we randomly selected 1500 of the PCQs to include as the benchmark query set. In Figure 4, we show, a scatter plot of $\left(V^{\pi}(s,h), V^{\hat{\pi}}(\hat{s},h)\right)$ for the selected set of PCQs for Halfcheetah-v2 and maze2d-open-v0. Notice the lack of PCQs along the diagonal, which corresponds to the removal of ambiguous queries. Also note that the PCQs span a range of value gaps, which suggests that they span varying PCQs of varying difficult. If these plots showed a bias toward only large gap queries, then additional steps would be necessary to ensure that more variation was present in the selected query sets.



(a) Halfcheetah-v2

(b) Maze2d-open-v0

Figure 4: Scatter plot of the PCQ in a) HalfCheetah and b) Maze2d-open-v0. For each PCQ $(s, \pi, \hat{s}, \hat{\pi}, h)$, we plot $V^{\pi}(s,h)$ vs. $V^{\hat{\pi}}(\hat{s},h)$.

5 OPCC Baselines

In this section, we describe the class of baselines that will be made available with the benchmarks and included in our pilot experiments (Section 7). Recall that each baseline must provide a prediction function f and confidence function c that are derived from the dataset $\mathcal{D}$. Perhaps the most natural approach for f to answer a PCQ $(s, \pi, \hat{s}, \hat{\pi}, h)$ is to estimate and then compare the relevant values using OPE. The corresponding confidence function c might then be based on the uncertainty of the value estimates.

There are at least two types of OPE approaches to consider: model-free and model-based. Model-free approaches, such as fitted Q-evaluation (Ernst et al., 2005) typically learn a Q-function $Q^{\pi}(s,a)$ for a given policy π that can be evaluated for any state and action. Unfortunately, each function learned by such model-free methods is valid for the single policy π and the effective horizon used during training. Thus, answering PCQs involving other policies or horizons requires costly retraining. Since we are seeking an OPCC approach, which can be quickly applied to arbitrary policies, states, and horizons, we instead choose to use a model-based approach for our baselines.

Our baselines are variants of model-based ensemble approaches, which are one of the most common class of approaches used in model-based RL for dynamics modeling and capturing uncertainty (Argenson & Dulac-Arnold, 2020; Yu et al., 2020; Kidambi et al., 2020). In particular, our baselines all have the following structure:

1. Learn an ensemble of models $\{\hat{P}_i\}$ from $\mathcal{D}$ that each predict the dynamics and reward of the environment.

2. Use each model in the ensemble to generate estimates of the relevant PCQ values, $V^{\pi}(s,h)$ and $V^{\hat{\pi}}(\hat{s},h)$, via Monte-Carlo simulation of the policies.

3. Combine the ensemble estimates to provide a prediction and confidence value.

Compared to model-free approaches, this approach can instantly apply to arbitrary policies and horizons without costly retraining. That is, new policies and horizons can easily be swapped into the Monte-Carlo simulation of step 2 with no modifications to the model. We can obtain different baselines by varying the choices for learning the model ensemble (step 1) as well as varying the ensemble combination approach (step 3). Below we describe the variations used in our experiments.

5.1 Base Models

In our experiments, we consider two types of base models for forming ensembles. The first base model is the commonly use *Feed-Forward (FF)* Gaussian model, which, given the current state/observation and action as input, returns the mean and diagonal covariance matrix of a Gaussian distribution over the next state and reward. This model allows for stochastic Monte-Carlo simulations by drawing the next state from the model's Gaussian distribution at each time step. In this work, we use the same FF base-model architecture and training details as MBPO (Janner et al., 2019).

We also consider a recent base model (Zhang et al., 2021) (referred to as *Auto-regressive (AR)*), which was demonstrated in some cases to improve over the output architecture of FF. Instead of generating all n features of the predicted next state in a single pass, AG auto-regressively samples each feature one at a time using n forward passes. In particular, to sample state feature i of the next state, denoted s^i_{t+1}, the network receives the usual input s_t and a_t as well as the previously sampled state features $s^0_{t+1}, ...s^{i-1}_{t+1}$. AR then returns the mean and covariance for a Gaussian that is used to sample s^i_{t+1}. The intuition is that this approach may allow for representing non-Gaussian and multi-modal next-state distributions compared to the uni-modal Gaussian FF model.

5.2 Ensemble Learning

Model-based approaches to ORL have commonly used ensembles as an attempt to quantify uncertainty, e.g. via measures of ensemble-member disagreement (Janner et al., 2019). We consider two choices for generating ensembles. The first choice is the standard *bootstrapping ensemble* approach, which simply trains each ensemble member using a different random weight initialization and bootstrapped dataset $\hat{\mathcal{D}}$ by sampling from $\mathcal{D}$ with replacement $|\mathcal{D}|$ times. The intent is that the combination of classic statistical bootstrapping Efron & Tibshirani (1994) and random initialization will produce a diverse set of ensemble models.

Often, however, it is observed that the basic bootstrapping approach does not create enough diversity in an ensemble, which is counter to our motivation of representing uncertainty. For this reason, there are a number of proposals for increasing the ensemble diversity, of which, we consider just one in this work. In particular, work motivated by capturing uncertainty in ORL proposed the use of *randomized constant priors* to increase ensemble diversity (Osband et al., 2018). For each base model, a randomized constant prior is produced, which is simply a network with random initial weights. The base model is trained as an additive component on top of this prior and the final output is the sum of the two. The intuition is that the constant prior should cause ensemble members to disagree more often in unrepresented parts of the state-space, which will provide a better measure of disagreement-based uncertainty.

5.3 Prediction and Confidence Values

Given a PCQ $(s, \pi, \hat{s}, \hat{\pi}, h)$ query and ensemble of size M we generate a prediction and confidence by first using each ensemble member to generate, via Monte-Carlo simulation, a pair of value estimates of $V^{\pi}(s, h)$ and $V^{\hat{\pi}}(\hat{s}, h)$. This results in a set of M value estimate pairs, denoted by $\mathcal{V} = \{(V_1, \hat{V}_1), \ldots, (V_M, \hat{V}_M)\}$. This approach is based on the classic view, from bagging classifiers (Breiman, 1996), of the base learning algorithm being a stochastic function[4]. The set $\mathcal{V}$ can then be viewed as value-estimate pairs sampled from the distribution of learning algorithm runs. Bagging analysis tells us that if 50% of the learning algorithm runs result in estimates $(V_i, \hat{V}_i)$ that correctly rank the values, then a large enough ensemble will correctly

[4]Indeed, each run of the base algorithm has at least two sources of randomness in our implementation. First, each run uses a different bootstrap sample of the dataset. Second, each run uses randomized initial weights. Third, mini-batches in stochastic gradient descent depend on the random seed

predict the query.[5] Based on this view, below we describe the three approaches we consider for producing predictions and confidences from $\mathcal{V}$.

- *Ensemble Voting (EV).* Following (Dietterich, 2000), EV simply returns a prediction for a query based on the majority vote across the ensemble of $V_i < \hat{V}_i$. The confidence score is equal to the fraction of ensemble members that agree on the majority vote (in the range [0.5,1]), but re-scaled to fall in the range [0,1].
- *Paired Confidence Interval (PCI).* The PCI confidence value is computed by estimating the expected value of $V - \hat{V}$ for a random run of the learning algorithm. The mean estimate is given by $\sum_i V_i - \hat{V}_i$ and the prediction is based on the sign of this estimate. The confidence value is based on computing α percentile confidence intervals on the difference, denoted by $[l_\alpha, u_\alpha]$. In particular, it is equal to the largest value of α such that $0 \notin [l_\alpha, u_\alpha]$. Thus, a high confidence value reflects that there is strong evidence that the expected difference is either above or below zero (in agreement with the prediction). Confidence intervals are computed based on the t distribution.
- *UnPaired Confidence Interval (U-PCI).* This approach makes the prediction in the same way as PCI, but uses unpaired confidence intervals to compute the confidence, which should be expected to be more conservative. In particular, we compute α percentile confidence intervals for the means of the V_i and $\hat{V}_i$ denoted respectively by $[l_\alpha, u_\alpha]$ and $\left[\hat{l}_\alpha, \hat{u}_\alpha\right]$ and let the confidence be the maximum value of α for which the confidence intervals do not overlap.

6 Related Work

Dynamics Learning in RL. There has been much recent interest in learning deep models of dynamical systems to support model-based RL. Examples from online RL include Clavera et al. (2018); Kurutach et al. (2018), which learn one-step observation-based dynamics along with extensions to ensembles Deisenroth & Rasmussen (2011); Chua et al. (2018); Janner et al. (2019); Nagabandi et al. (2020). PILCO (Deisenroth & Rasmussen, 2011; Gal et al., 2016) is another model-based RL approach that learns dynamics via Gaussian Processes (Rasmussen, 2003), which are able to capture epistemic uncertainty. However, performance is primarily measured in terms of overall task performance and it is unclear how well uncertainty is actually quantified. Recent work on offline reinforcement, such as MBOP(Argenson & Dulac-Arnold, 2020), MOPO (Yu et al., 2020), and MoREL (Kidambi et al., 2020) has also considered learning dynamics models over observations from fixed, offline data sets. These approaches incorporate uncertainty estimates in different ways (e.g. pessimistic rewards or dynamics) and all use ensembles to estimate uncertainty. Thus, they are established exemplars of the baselines considered in our work. However, only the final task performance is tested and it is unclear how well uncertainty is actually captured by the models. COMBO (Yu et al., 2021), Muzero Unplugged (Schrittwieser et al., 2021), and LOMPO (Rafailov et al., 2021) investigate learning latent-space dynamics models for offline RL rather than learning in the observation space. Again, however, there is no explicit evaluation of the models ability to quantify uncertainty.

Policy Ranking. Similar to our work, Sonabend-W et al. (2020) also identifies the need to measure uncertainty for policies learned with limited data. In order to learn safe policies, their approach uses hypothesis testing for determining uncertainty in policy evaluation for a pair of candidate policies based on sampling from model posteriors. This helps in ranking them and selection of better performing policy over the behavior policy in a safe manner. The work, however, was limited to small flat state-spaces and did not explicitly evaluate uncertainty quantification. In contrast, our work produces a benchmark to primarily focus on uncertainty quantification of a system using offline data, rather than evaluating in terms of overall task performance.

DOPE (Fu et al., 2021) studies OPE and devises a protocol that measures policy evaluation, ranking, and selection. For this purpose, the approach introduces a set of candidate policies along with their expected value over a distribution of initial states. Rather, in our work, we question the ability of a system to rank

[5]This argument assumes independence of the ensemble members, which clearly is not true in practice due to at least correlation between their training data.

policies from any arbitrary state for a given horizon instead of limiting to initial state distribution only. This can help provide a more comprehensive view of uncertainty estimation across the state space.

In similar motivation, SOPR-T(Jin et al., 2021b) also considers policy ranking from offline data and additional policy-value supervision. This is done by learning an encoded representation of a policy using a transformer based architecture and a scoring function over the representation. In order to learn the representation, they require a set of pre-defined policies, each labeled by its ground truth value with respect to an initial state distribution. Our framework does not assume the availability of such policy-value supervision and also puts an emphasis on uncertainty quantification, which is not evaluated by this work.

Confidence Intervals. We make use of confidence intervals over policy value estimates for answering queries. Thomas et al. (2015) also studies confidence interval estimation over policy value estimates using trajectories generated by a different set of policies. Their approach uses importance sampling for unbiased value estimates, which suffers from high variance leading to loose confidence bounds. They also introduce the problem of *high confidence off-policy evaluation* and produce tighter bounds on estimates using improved concentration inequalities (Massart, 2007).

Another class of approaches is based on statistical bootstrapping (Efron, 1987). Hanna et al. (2017) bootstraps learned MDP transition models in order to estimate lower confidence bounds on policy evaluation estimates with limited data. Kostrikov & Nachum (2020) suggests that confidence intervals of these bootstrapped estimates are not guaranteed to be accurate. In practice, they are shown to be overly confident especially for insufficient sample sizes and under-coverage of the data distribution. They suggest, that, in practice, this issue may be mitigated by inducing noisy rewards and regularization to learn smoother empirical transition and reward functions. Evaluating that claim within our OPCC framework is a potential direction for future work.

CoinDICE (Dai et al., 2020), and similar methods (Nachum et al., 2019; Zhang et al., 2020; Strehl & Littman, 2008; Hao et al., 2021) progressively focus on confidence intervals for OPE based on the formulation of certain optimization problems. This iterative optimization approach for estimating policy value and confidence bounds induces a computational overload. This is undesirable in our framework which aims to rapidly answer queries of arbitrary horizons and policies making it computationally unsustainable. An interesting direction for future work is to consider generalizing this optimization-based approach to more flexibly handle arbitrary policies.

7 Experiments

Our pilot experiments explore the baseline methods on our benchmarks using the proposed metrics for OPCC. It is important to note that these experiments are not intended to identify a top performer. Rather our primary goal for these pilot experiments is to assess the adequacy of the benchmarks and metrics for future work and to establish a basic performance bar. Secondarily, we are interested to observe evidence or the lack of evidence for certain assumptions that might be drawn about the baselines from prior work. Based on those primary goals our experiments and analysis are designed to: 1) Assess whether the benchmarks appear to be too difficult or too easy for supporting future work; 2) Assess whether there is any evidence that our baselines are sensitive to the data-set type used for each benchmark environment. In particular, the performance of strong OPCC approach should vary with the coverage afforded by the data set. 3) Assess whether there is any evidence for performance differences among the baseline variations. In particular, certain features such as auto-regressive sampling, constant priors, and larger ensembles have been claimed to improve uncertainty handling in prior work.

In our experiments, unless otherwise specified the default model is an ensemble of 100 deterministic feed-forward models and uses EV for the confidence score. Two additional details are important to note for these experiments. First, as is customary in model-based RL (including ORL), we are using a pre-defined episode termination function rather than a learned one. We have found that this can significantly impact performance of model-based RL systems and also our OPCC evaluations. Second, we clipped predicted observations and rewards to keep them within the bounds of the available data sets, which is also a common practice in ORL that we found to be important.

Too hard or too easy? We first assess the degree of difficulty posed by our OPCC benchmark for our baselines. Figures 5 and 6 show RCCs of our default model for different data set types (averaged across the different PCQ horizons h) in gym-mujoco and maze2d environments. Tables 1 and 2 report their corresponding metrics i.e. AURCC, RPP, CR_k, and Loss (or risk) at complete coverage.

First, we consider risk at complete coverage and find that there is no significant difference in risk across dataset type, but varies significantly across gym-mujoco environments. This shows that some environments are more challenging than others due to their underlying complex dynamics and high dimensional observation and action sizes. Also, the risk at complete coverage for maze2d environments with a single dataset ('1m') is significantly lower than gym-mujoco. This is potentially due to data collection via a path-planning procedure leading to significant state-action space coverage. Further Medium and Umaze have very small risks without much room for risk improvement, while Large and Open appear to have room for improvement. Second, we consider how the risk varies across coverage values. In most cases, there are no thresholds that produce points within the coverage interval (0,0.5], which indicates a lack of sensitivity in that coverage range. There are typically multiple points between (0.5, and 1], though often just a few. Ideally we would hope for a more gradual degradation in risk spanning from no coverage to complete coverage. This suggests that there is significant room to improve the coverage sensitivity, especially in the range [0,0.5].

Overall, the current set of benchmarks, with the exception of 2 Maze2d environments, are not too easy and appears to offer significant room for improvement in terms of both overall risk and sensitivity of the RCCs across coverage values. Likewise, the observation that the risks achieved are significantly less than chance suggest that the benchmarks are not too hard.

Impact of dataset type. The different types of data sets provide different types of coverage of the system dynamics. Is there evidence that our baselines are able to distinguish among these types? Figure 5 shows that the RCCs for different datasets are quite similar for each of the gym-mujoco environment. The AURCC and RPP values in Table 1 are consistent with these observations. This could be due to the diverse coverage of queries across the state space that offer challenges for all datasets. The small variation in RCCs across dataset types could also be due to the models learned from different datasets providing similar types of generalization. It is also possible that differences between dataset types would become more prevalent for smaller versions of the datasets, which an interesting future extension to the benchmarks. Finally no significant patterns for CR in relation to data-set type are apparent, which is not surprising since CR is expected to be more heavily influenced by the type of baseline approach.

Table 1: Evaluation metrics for *dataset-type* comparison in *gym-mujoco* environments. This includes mean and confidence intervals estimates at 95% confidence level for metrics corresponding to 5(seed) dynamics trained over each dataset.

Env.	dataset-type	AURCC(↓)	RPP(↓)	CR_K(↑)	loss(↓)
Half Cheetah	expert	0.212 ± 0.002	0.05 ± 0.001	$0.5 \pm (< 0.001)$	0.361 ± 0.002
	medium	0.222 ± 0.001	0.048 ± 0.001	$0.5 \pm (< 0.001)$	0.374 ± 0.002
	medium-expert	0.24 ± 0.004	0.06 ± 0.002	$0.6 \pm (< 0.001)$	0.387 ± 0.003
	medium-replay	0.216 ± 0.001	0.04 ± 0.001	$0.4 \pm (< 0.001)$	0.368 ± 0.001
	random	0.206 ± 0.001	0.023 ± 0.001	$0.3 \pm (< 0.001)$	0.378 ± 0.001
Hopper	expert	0.152 ± 0.002	0.04 ± 0.001	$0.5 \pm (< 0.001)$	0.284 ± 0.002
	medium	0.133 ± 0.001	0.03 ± 0.001	$0.4 \pm (< 0.001)$	0.26 ± 0.002
	medium-expert	0.136 ± 0.001	$0.028 \pm (< 0.001)$	$0.4 \pm (< 0.001)$	0.265 ± 0.001
	medium-replay	0.128 ± 0.001	0.012 ± 0.001	$0.3 \pm (< 0.001)$	0.258 ± 0.001
	random	0.156 ± 0.008	0.045 ± 0.004	0.54 ± 0.043	$0.273 \pm (< 0.001)$
Walker 2d	expert	0.064 ± 0.001	$0.011 \pm (< 0.001)$	$0.3 \pm (< 0.001)$	$0.161 \pm (< 0.001)$
	medium	0.069 ± 0.001	$0.007 \pm (< 0.001)$	0.22 ± 0.035	0.156 ± 0.001
	medium-expert	$0.068 \pm (< 0.001)$	$0.007 \pm (< 0.001)$	0.24 ± 0.043	0.153 ± 0.001
	medium-replay	0.07 ± 0.001	$0.005 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.161 ± 0.001
	random	0.067 ± 0.001	0.024 ± 0.001	0.54 ± 0.043	0.165 ± 0.001

Table 2: Evaluation metrics for *dataset-types* comparison in *maze* environments. This includes mean and confidence intervals estimates at 95% confidence level for metrics corresponding to 5(seed) dynamics trained over each dataset.

ENV.	DATASET-TYPE	AURCC(↓)	RPP(↓)	CR_K(↑)	LOSS(↓)
LARGE	1M	0.14 ± 0.015	0.062 ± 0.004	0.82 ± 0.035	0.251 ± 0.029
MEDIUM	1M	$0.001 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.022 ± 0.001
OPEN	1M	0.029 ± 0.001	0.012 ± 0.001	$0.5 \pm (< 0.001)$	0.107 ± 0.005
UMAZE	1M	0.008 ± 0.001	$0.002 \pm (< 0.001)$	$0.3 \pm (< 0.001)$	0.075 ± 0.003

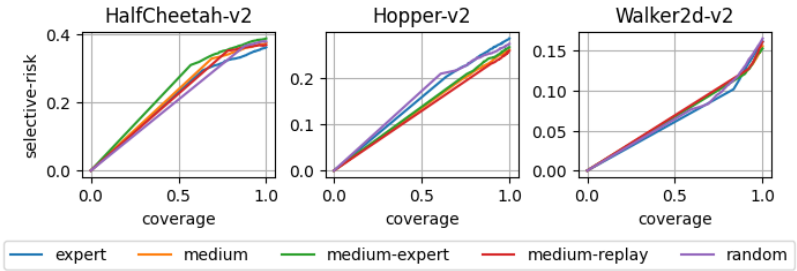


Figure 5: Selective-risk coverage curves for different *gym-mujoco* environements and *dataset types* (depicted by different colors). The x-axis spans from no(0) coverage to complete(1) coverage of queries and the y-axis is the risk for the corresponding query coverage. Each risk-coverage point is determined by varying the confidence threshold.

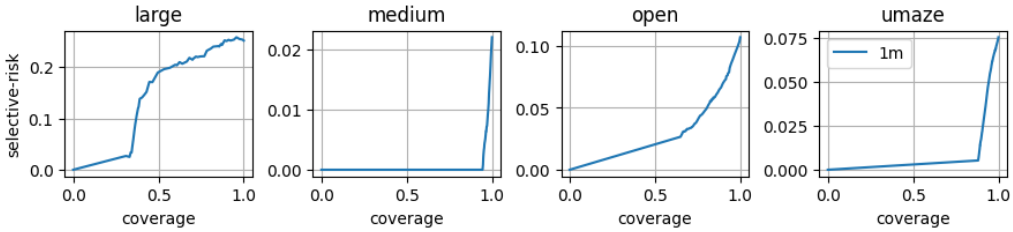


Figure 6: Selective-risk coverage curve for '1m' dataset in *maze* environments. This is the complete navigation dataset of *1 million* transactions.

Impact of Query Horizon. Learned dynamics are well known to suffer from error accumulation in multi-step rollouts. This leads to the hypothesis that OPCC performance might degrade with increasing query horizons. In Table 3 and Table 17 (Appendix) we provides metrics for various horizons h averaged across data-set types. As expected, we observe higher AURCCs for longer horizons, which provides positive evidence for the hypothesis. Interestingly, we observe that in most of the environments we have very low risk for short horizons. In general, we observe AURCCs for $h = 10$ or $h = 20$ are at least an order of magnitude smaller than for larger horizons across the benchmark. This suggests a possible threshold effect for OPCC with respect to increasing horizon due to error accumulation. It also suggests our current baselines are better suited for applications like reliable policy improvement with smaller horizons.

Influence of different confidence functions. Table 4 and Table 16(Appendix) gives metrics for our three different uncertainty functions (EV, PCI, and U-PCI) averaged over data-set types and horizons. The results for AURCC and RPP both indicate evidence that the confidence interval approaches (PCI and U-PCI) have an advantage over EV. This is encouraging as it suggests considering other more sophisticated statistical testing approaches may lead to further improvement. However, the results for CR indicate that the confidence interval approaches have significantly less resolution than EV. This may lead to poorer

Table 3: Evaluation metrics for *horizon* comparison in *gym-mujoco* environments. These mean and confidnce interval(95%) estimates are over 50 samples corresponding to 5(seed) dynamics trained over 5 different datasets.

Env.	horizon	AURCC(↓)	RPP(↓)	CR_K(↑)	loss(↓)
Half Cheetah	10.0	0.077 ± 0.004	0.008 ± 0.001	0.288 ± 0.017	0.191 ± 0.006
	20.0	0.217 ± 0.006	0.041 ± 0.005	0.416 ± 0.042	0.374 ± 0.005
	30.0	0.215 ± 0.004	0.038 ± 0.005	0.404 ± 0.038	0.377 ± 0.005
	40.0	0.223 ± 0.006	0.049 ± 0.006	0.464 ± 0.041	0.368 ± 0.005
	50.0	0.277 ± 0.008	0.063 ± 0.007	0.516 ± 0.054	0.428 ± 0.003
Hopper	10.0	0.017 ± 0.002	$0.001 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.048 ± 0.002
	20.0	0.078 ± 0.002	0.016 ± 0.003	0.336 ± 0.038	0.17 ± 0.003
	30.0	0.146 ± 0.004	0.029 ± 0.004	0.42 ± 0.033	0.284 ± 0.005
	40.0	0.169 ± 0.007	0.038 ± 0.006	0.432 ± 0.036	0.293 ± 0.004
	50.0	0.196 ± 0.009	0.047 ± 0.007	0.516 ± 0.049	0.334 ± 0.006
Walker 2d	10.0	$0.011 \pm (< 0.001)$	$0.001 \pm (< 0.001)$	0.22 ± 0.016	0.033 ± 0.003
	20.0	0.025 ± 0.002	0.003 ± 0.001	0.252 ± 0.042	0.077 ± 0.004
	30.0	0.059 ± 0.001	0.01 ± 0.003	0.3 ± 0.061	0.132 ± 0.004
	40.0	0.093 ± 0.002	0.017 ± 0.004	0.384 ± 0.049	0.209 ± 0.004
	50.0	0.131 ± 0.002	0.023 ± 0.005	0.392 ± 0.06	0.259 ± 0.002

Table 4: Evaluation metrics for *uncertainty-type* comparison in *gym-mujoco* environments. These mean and confidence interval(95%) estimates are over 50 samples corresponding to 5(seed) dynamics trained over 5 different datasets.

Env.	uncertainty-type	AURCC(↓)	RPP(↓)	CR_K(↑)	loss(↓)
Half Cheetah	ev	0.219 ± 0.005	0.044 ± 0.005	0.46 ± 0.04	0.374 ± 0.004
	pci	0.191 ± 0.002	0.006 ± 0.001	$0.2 \pm (< 0.001)$	0.373 ± 0.004
	u-pci	0.196 ± 0.003	0.014 ± 0.002	0.228 ± 0.018	0.373 ± 0.004
Hopper	ev	0.141 ± 0.005	0.031 ± 0.005	0.428 ± 0.034	0.268 ± 0.004
	pci	0.135 ± 0.002	0.004 ± 0.001	$0.2 \pm (< 0.001)$	0.269 ± 0.004
	u-pci	0.135 ± 0.003	0.009 ± 0.002	0.216 ± 0.014	0.269 ± 0.004
Walker 2d	ev	0.068 ± 0.001	0.011 ± 0.003	0.3 ± 0.051	0.159 ± 0.002
	pci	0.078 ± 0.001	0.002 ± 0.001	$0.2 \pm (< 0.001)$	0.16 ± 0.003
	u-pci	0.076 ± 0.001	0.004 ± 0.001	0.22 ± 0.016	0.16 ± 0.003

performance for probability calibration approaches applied to PCI or U-PCI confidence scores. Further work is required to understand this decrease in resolution.

Impact of Ensemble Size. We now consider the impact of ensemble size for our baselines. Table 5 and Table 18 (Appendix) show the results for ensemble sizes ranging from 10 to 100. Our prior expectation was that performance would increase with significant increase in ensemble-size. In general, we do not see statistically significant differences between ensemble sized for AURCC based on our current experimental budget (i.e. confidence intervals intersect). However, based on trends in the means, there is weak evidence of improved AURCC. The exception is HalfCheetah, where for AURCC, the trends is opposite of the expectation. However, the differences in means tends to be small, suggesting that ensemble size is not having a large impact even if more computational budget were devoted to support statistical significance.

For RPP and Coverage Resolution (CR_k) there is typically a statically significant improvement from ensemble size 10 to 100. The exceptions are umaze and medium-maze where losses are very small for all ensemble sizes. Overall, however, differences are relatively small in magnitude. This may be due to the ensembles not being diverse enough, or the base models used to construct the ensembles are not accurate enough. These results demonstrate the value of the OPCC benchmarks in being able to explicitly test hypotheses about uncertainty quantification, rather than relying on downstream results that may be impacted by many possible factors.

Table 5: Evaluation metrics for *ensemble-count* comparison in *gym-mujoco* environments. We train 5(seed) ensemble dynamics of size 100 for each dataset and start with ensemble of 10 models for OPCC metrics estimation. Thereafter, we incremently increase their exposure for metrics mean and confidence intervals (95%)

Env.	ensemble-count	AURCC(↓)	RPP(↓)	CR_K(↑)	loss(↓)
Half Cheetah	10	0.209 ± 0.004	0.028 ± 0.004	0.32 ± 0.029	0.378 ± 0.004
	20	0.213 ± 0.004	0.034 ± 0.004	0.36 ± 0.04	0.376 ± 0.004
	40	0.215 ± 0.004	0.039 ± 0.005	0.4 ± 0.035	0.374 ± 0.004
	80	0.219 ± 0.005	0.043 ± 0.005	0.452 ± 0.04	0.374 ± 0.004
	100	0.219 ± 0.005	0.044 ± 0.005	0.46 ± 0.04	0.374 ± 0.004
Hopper	10	0.141 ± 0.005	0.02 ± 0.004	0.316 ± 0.029	0.273 ± 0.005
	20	0.141 ± 0.005	0.024 ± 0.004	0.344 ± 0.037	0.272 ± 0.004
	40	0.141 ± 0.005	0.027 ± 0.004	0.388 ± 0.04	0.27 ± 0.004
	80	0.141 ± 0.005	0.03 ± 0.004	0.416 ± 0.038	0.269 ± 0.004
	100	0.141 ± 0.005	0.031 ± 0.005	0.428 ± 0.034	0.268 ± 0.004
Walker 2d	10	0.072 ± 0.001	0.007 ± 0.002	0.256 ± 0.03	0.16 ± 0.002
	20	0.071 ± 0.001	0.008 ± 0.002	0.268 ± 0.038	0.16 ± 0.002
	40	0.07 ± 0.001	0.01 ± 0.003	0.28 ± 0.046	0.16 ± 0.002
	80	0.068 ± 0.001	0.01 ± 0.003	0.284 ± 0.049	0.159 ± 0.002
	100	0.068 ± 0.001	0.011 ± 0.003	0.3 ± 0.051	0.159 ± 0.002

Table 6: Evaluation metrics for *prior-scale* comparison in *gym-mujoco* environments comprising of mean and confidence interval(95%) over 50 samples belonging to 5(seed) dynamics models for each of the 5 datasets. Prior scale of 0 means no randomized constant prior is added.

Env.	prior-scale	AURCC(↓)	RPP(↓)	CR_K(↑)	loss(↓)
Half Cheetah	0	0.219 ± 0.005	0.044 ± 0.005	0.46 ± 0.04	0.374 ± 0.004
	5	0.236 ± 0.005	0.067 ± 0.004	0.768 ± 0.081	0.373 ± 0.006
Hopper	0	0.141 ± 0.005	0.031 ± 0.005	0.428 ± 0.034	0.268 ± 0.004
	5	0.145 ± 0.003	0.045 ± 0.004	0.616 ± 0.05	0.269 ± 0.004
Walker 2d	0	0.068 ± 0.001	0.011 ± 0.003	0.3 ± 0.051	0.159 ± 0.002
	5	0.057 ± 0.001	0.017 ± 0.002	0.44 ± 0.04	0.159 ± 0.002

Randomized Constant Priors. In order to encourage diversity, we introduce *randomized constant priors* in our ensemble models. These are suggested to encourage extrapolation diversity, especially on out-of-distribution state-action pairs, which could improve disagreement-based uncertainty estimates. However, when we included the constant priors in our model, we didn't find significant improvements in our evaluation metrics as shown in Table 10(Appendix) and Table 6. We use the same architecture as the dynamics model for *prior* with random weights and scale them with *"prior-scale"* before adding them to ensemble models and a prior-scale of 0 indicates no usage of them. In the case of maze2d, we generally observe a slight (but statistically insignificant) reduction in *AURCC*, whereas *RPP* and CR_K tends to remains same. On the contrary, in the case of gym-mujoco (Table 6), we generally observe a slight (but statistically insignificant) increase in *AURCC* , *RPP* and CR_K.

Prior work by Osband et al. (2018) demonstrated improvement in end-task RL performance by having an ensemble of DQN (Mnih et al., 2013) models with randomized constant priors. However, explicit analysis of the uncertainty quantification was not provided. Our observations suggests that randomized constant priors do not appear to improve uncertainty quantification at least as measured through our OPCC benchmarks. Further investigation is necessary to better understand the performance differences observed in Osband et al. (2018). An interesting direction of future work is to consider other previously proposed mechanisms for improving ensemble diversity within the OPCC framework.

Table 7: Evaluation metrics for *dynamics-type* comparison in *gym-mujoco* environments comprising of mean and confidence interval(95%) estimates over 50 samples belonging to 5(seeds) dynamics models for each of the 5 datasets.

Env.	dynamics-type	AURCC(↓)	RPP(↓)	CR_K(↑)	loss(↓)
Half Cheetah	autoregressive	0.249 ± 0.008	0.068 ± 0.006	0.736 ± 0.087	0.379 ± 0.003
	feed-forward	0.219 ± 0.005	0.044 ± 0.005	0.46 ± 0.04	0.374 ± 0.004
Hopper	autoregressive	0.139 ± 0.007	0.034 ± 0.005	0.48 ± 0.042	0.268 ± 0.007
	feed-forward	0.141 ± 0.005	0.031 ± 0.005	0.428 ± 0.034	0.268 ± 0.004
Walker 2d	autoregressive	0.065 ± 0.002	0.013 ± 0.002	0.356 ± 0.046	0.158 ± 0.002
	feed-forward	0.068 ± 0.001	0.011 ± 0.003	0.3 ± 0.051	0.159 ± 0.002

Dynamics Model Types. Finally we compare the impact of the dynamics model type, in our case, either feed-forward(FF) or auto-regressive(AR). We use the same architecture as defined by Zhang et al. (2021). In Table 7 and Table 13 (Appendix), we do not observe significant evidence in favor of the AR model with respect to OPCC performance. There is a marginal, but no statistical significant reduction in $AURCC$ in some cases. We do see an increase in coverage resolution (CR_k) for the gym-mujoco environments when using the AR model, while it remains the same for the maze2d environments. This may be due to the additional uncertainty propagation that can occur during auto-regressive inference of each dimension, especially in the higher-dimensional gym-mujoco environments. Currently our results do not suggest that the extra computational cost of the AR model compared to FF is worthwhile with respect to uncertainty quantification as measured via OPCC. This may be due to the environments not needed to represent multi-modal output distribution, which is where the AR model could have a distinct advantage.

Determinism. Our baseline model is a deterministic version of the stochastic model defined in Chua et al. (2018), trained via regression loss. A classic improvement is to induce stochasticity into the model by learning a normal distribution over the next observation rather than a point estimate. We experimented with this modification and provide results in the Appendix (Tables 11 and 12). Though, limitations of deterministic models are well-understood for stochastic environments, it turns out we don't gain significantly with stochastic models in our pilot run. This is possibly due to deterministic nature of maze environments and low stochasticity in gym-mujoco case.

8 Summary

Properly quantifying uncertainty of complex models is a major open problem of practical significance in machine learning. Despite this fact, only a small fraction of the work in machine learning attempts to address this problem. Further, in areas such as offline RL, where methods for addressing uncertainty are developed, there is very little direct evaluation of uncertainty quantification. In recent years, there has been impressive progress on out-of-distribution detection for image classification, where quantifying uncertainty is a core problem. This has been largely driven by the availability of benchmarks that lower the overhead for conducting research and comparing methods. Currently, there is a lack of such benchmarks for sequential decision-making. The OPCC problem is a relatively simple problem to state, yet is rich enough to capture the essence of uncertainty quantification for sequential decision making. We hope that the OPCC benchmarks will inspire other researchers to develop new ideas for uncertainty quantification. Indeed, our pilot experiments show there is significant room to improve and that our understanding of current mechanisms is incomplete. Finally, we hope that this initial benchmark and baseline contribution is only the initial seed for the community at large to contribute to as progress is made.

References

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

Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Jacob Buckman, Carles Gelada, and Marc G Bellemare. The importance of pessimism in fixed-dataset policy optimization. *arXiv preprint arXiv:2009.06799*, 2020.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *arXiv preprint arXiv:1805.12114*, 2018.

Ignasi Clavera, Jonas Rothfuss, John Schulman, Yasuhiro Fujita, Tamim Asfour, and Pieter Abbeel. Model-based reinforcement learning via meta-policy optimization. In *Conference on Robot Learning*, pp. 617–629. PMLR, 2018.

Filipe Condessa, José Bioucas-Dias, and Jelena Kovačević. Performance measures for classification systems with rejection. *Pattern Recognition*, 63:437–450, 2017.

Bo Dai, Ofir Nachum, Yinlam Chow, Lihong Li, Csaba Szepesvári, and Dale Schuurmans. Coindice: Off-policy confidence interval estimation. *arXiv preprint arXiv:2010.11652*, 2020.

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)*, pp. 465–472. Citeseer, 2011.

Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pp. 1–15. Springer, 2000.

Bradley Efron. Better bootstrap confidence intervals. *Journal of the American statistical Association*, 82 (397):171–185, 1987.

Bradley Efron and Robert J Tibshirani. *An introduction to the bootstrap*. CRC press, 1994.

Ran El-Yaniv et al. On the foundations of noise-free selective classification. *Journal of Machine Learning Research*, 11(5), 2010.

Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, 2005.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Justin Fu, Mohammad Norouzi, Ofir Nachum, George Tucker, Ziyu Wang, Alexander Novikov, Mengjiao Yang, Michael R Zhang, Yutian Chen, Aviral Kumar, et al. Benchmarks for deep off-policy evaluation. *arXiv preprint arXiv:2103.16596*, 2021.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *arXiv preprint arXiv:2106.06860*, 2021.

Yarin Gal, Rowan McAllister, and Carl Edward Rasmussen. Improving pilco with bayesian neural network dynamics models. In *Data-Efficient Machine Learning workshop, ICML*, volume 4, pp. 25, 2016.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pp. 4885–4894, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN 9781510860964.

Yonatan Geifman and Ran El-Yaniv. Selectivenet: A deep neural network with an integrated reject option. In *International Conference on Machine Learning*, pp. 2151–2159. 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.

Botao Hao, Xiang Ji, Yaqi Duan, Hao Lu, Csaba Szepesvári, and Mengdi Wang. Bootstrapping statistical inference for off-policy evaluation. *arXiv preprint arXiv:2102.03607*, 2021.

Kilian Hendrickx, Lorenzo Perini, Dries Van der Plas, Wannes Meert, and Jesse Davis. Machine learning with a reject option: A survey. *arXiv preprint arXiv:2107.11277*, 2021.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *arXiv preprint arXiv:1906.08253*, 2019.

Ying Jin, Zhuoran Yang, and Zhaoran Wang. Is pessimism provably efficient for offline rl? In *International Conference on Machine Learning*, pp. 5084–5096. PMLR, 2021a.

Yue Jin, Yue Zhang, Tao Qin, Xudong Zhang, Jian Yuan, Houqiang Li, and Tie-Yan Liu. Supervised off-policy ranking. *arXiv preprint arXiv:2107.01360*, 2021b.

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

Ilya Kostrikov and Ofir Nachum. Statistical bootstrapping for uncertainty estimation in off-policy evaluation. *arXiv preprint arXiv:2007.13609*, 2020.

Ilya Kostrikov, Rob Fergus, Jonathan Tompson, and Ofir Nachum. Offline reinforcement learning with fisher divergence critic regularization. In *International Conference on Machine Learning*, pp. 5774–5783. PMLR, 2021.

Aviral Kumar, Justin Fu, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *arXiv preprint arXiv:1906.00949*, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *arXiv preprint arXiv:2006.04779*, 2020.

Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-ensemble trust-region policy optimization. *arXiv preprint arXiv:1802.10592*, 2018.

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

Wei-Yin Loh. Calibrating confidence coefficients. *Journal of the American Statistical Association*, 82(397): 155–162, 1987.

Pascal Massart. *Concentration inequalities and model selection: Ecole d'Eté de Probabilités de Saint-Flour XXXIII-2003*. Springer, 2007.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Ofir Nachum, Yinlam Chow, Bo Dai, and Lihong Li. Dualdice: Behavior-agnostic estimation of discounted stationary distribution corrections. *arXiv preprint arXiv:1906.04733*, 2019.

Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

Anusha Nagabandi, Kurt Konolige, Sergey Levine, and Vikash Kumar. Deep dynamics models for learning dexterous manipulation. In *Conference on Robot Learning*, pp. 1101–1112. PMLR, 2020.

Ian Osband, John Aslanides, and Albin Cassirer. Randomized prior functions for deep reinforcement learning. *arXiv preprint arXiv:1806.03335*, 2018.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

Rafael Rafailov, Tianhe Yu, Aravind Rajeswaran, and Chelsea Finn. Offline reinforcement learning from images with latent space models. In *Learning for Dynamics and Control*, pp. 1154–1168. PMLR, 2021.

Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pp. 63–71. Springer, 2003.

Arthur George Richards. *Robust constrained model predictive control*. PhD thesis, Massachusetts Institute of Technology, 2005.

Julian Schrittwieser, Thomas Hubert, Amol Mandhane, Mohammadamin Barekatain, Ioannis Antonoglou, and David Silver. Online and offline reinforcement learning by planning with a learned model. *arXiv preprint arXiv:2104.06294*, 2021.

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

Aayam Shrestha, Stefan Lee, Prasad Tadepalli, and Alan Fern. Deepaveragers: Offline reinforcement learning by solving derived non-parametric mdps. In *International Conference on Learning Representations*, 2021.

Aaron Sonabend-W, Junwei Lu, Leo A Celi, Tianxi Cai, and Peter Szolovits. Expert-supervised reinforcement learning for offline policy learning and evaluation. *arXiv preprint arXiv:2006.13189*, 2020.

Alexander L Strehl and Michael L Littman. An analysis of model-based interval estimation for markov decision processes. *Journal of Computer and System Sciences*, 74(8):1309–1331, 2008.

Philip Thomas, Georgios Theocharous, and Mohammad Ghavamzadeh. High-confidence off-policy evaluation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

Ji Xin, Raphael Tang, Yaoliang Yu, and Jimmy Lin. The art of abstention: Selective prediction and error regularization for natural language processing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 1040–1051, 2021.

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

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization. *arXiv preprint arXiv:2102.08363*, 2021.

Michael R Zhang, Tom Le Paine, Ofir Nachum, Cosmin Paduraru, George Tucker, Ziyu Wang, and Mohammad Norouzi. Autoregressive dynamics models for offline policy evaluation and optimization. *arXiv preprint arXiv:2104.13877*, 2021.

Shangtong Zhang, Bo Liu, and Shimon Whiteson. Gradientdice: Rethinking generalized offline estimation of stationary values. In *International Conference on Machine Learning*, pp. 11194–11203. PMLR, 2020.

A Appendix - OPCC Benchmark Summary

In the following, we share a snapshot of OPCC benchmark components.

Table 8: Information about OPCC Benchmark comprising of environment details , datasets and queries.

Env.	Observation-size	Action-Dimensions	Max. Env. Steps	Dataset-name	Query-count
maze2d-open-v0	4	2	150	1m	1500
maze2d-medium-v1	4	2	600	1m	1500
maze2d-umaze-v1	4	2	300	1m	1500
maze2d-umaze-v1	4	2	800	1m	121
HalfCheetah-v2	17	6	1000	random, expert, medium, medium-replay, medium-expert	1500
Hopper-v2	11	3	1000	random, expert, medium, medium-replay, medium-expert	1500
Walker2d-v2	17	6	1000	random, expert, medium, medium-replay, medium-expert	1500

Table 9: Performance of policies used in PCQs. These policies are trained using PPO (Schulman et al., 2017) over the original environment task and hand-picked at different performance levels. We report mean and standard deviation of policy performance over 20 episodes.

Env.	policy-1	policy-2	policy-3	policy-4
maze2d-open-v0	122 ± 10	104 ± 22	18 ± 14	4 ± 8
maze2d-umaze-v1	245 ±272	203 ±252	256 ±260	258 ±262
maze2d-medium-v1	235 ±35	197 ±58	23 ±73	3 ±9
maze2d-large-v1	231 ±268	160 ±201	50 ±76	9 ±9
HalfCheetah-v2	1168 ±80	1044 ±112	785 ±303	94 ±40
Hopper-v2	1195 ±794	1466 ±487	1832 ±560	236 ±1
Walker2d-v2	2506 ±698	811 ±321	387 ±42	162 ±102

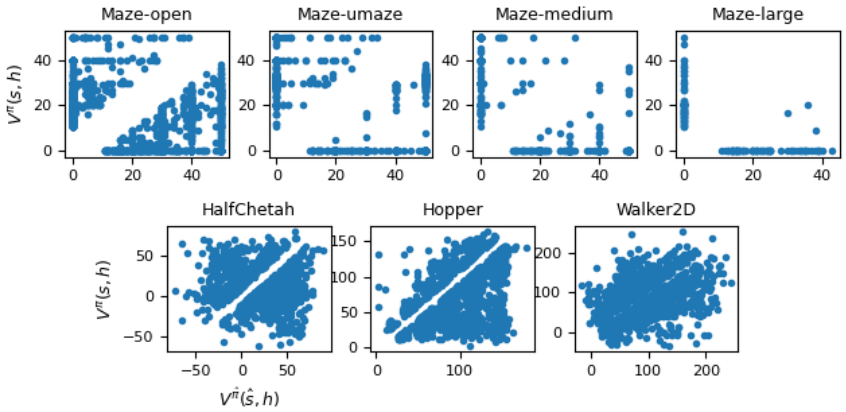


Figure 7: Scatter plot of the PCQ for each benchmark environment. For each PCQ $(s, \pi, \hat{s}, \hat{\pi}, h)$, we plot $V^{\pi}(s,h)$ vs. $V^{\hat{\pi}}(\hat{s},h)$.

B Appendix - Evaluation Metrics & Selective-Risk Coverage Curves for Ablations

In the following sub-sections, we share OPCC metrics for various ablations over our baseline. In each table-cell, we show mean and confidence interval at *95% confidence level* for corresponding metrics, estimated by evaluating 5 dynamics runs over each dataset of the corresponding environment.

B.1 Randomized Constant Priors

Figures 8,9 and Tables 6,10 shows impact of adding randomized constant priors to our baseline ensemble models. Output of prior models is scaled by *Prior-Scale* before adding to dynamic model. *Prior-Scale* of 0 implies randomized prior was not added. We observe significant performance gain only in Large-Maze environment.

Table 10: Evaluation metrics for *prior-scale* comparison in *maze* environments.

Env.	prior-scale	AURCC(↓)	RPP(↓)	CR_K(↑)	loss(↓)
large	0	0.14 ± 0.015	0.062 ± 0.004	0.82 ± 0.035	0.251 ± 0.029
	5	0.104 ± 0.017	0.051 ± 0.012	0.82 ± 0.035	0.197 ± 0.017
medium	0	$0.001 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.022 ± 0.001
	5	$0.0 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.02 ± 0.003
open	0	0.029 ± 0.001	0.012 ± 0.001	$0.5 \pm (< 0.001)$	0.107 ± 0.005
	5	0.032 ± 0.001	$0.012 \pm (< 0.001)$	$0.5 \pm (< 0.001)$	0.115 ± 0.008
umaze	0	0.008 ± 0.001	$0.002 \pm (< 0.001)$	$0.3 \pm (< 0.001)$	0.075 ± 0.003
	5	0.006 ± 0.001	$0.002 \pm (< 0.001)$	$0.3 \pm (< 0.001)$	0.071 ± 0.002

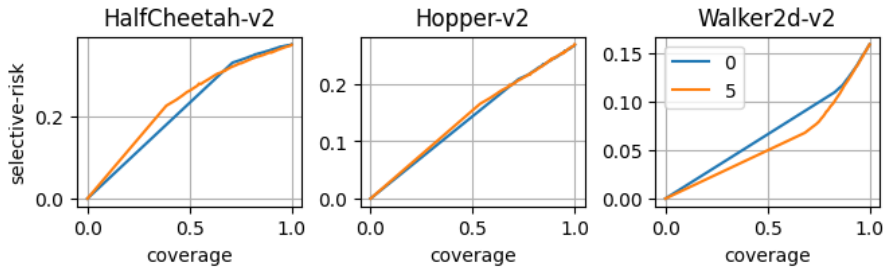


Figure 8: Selective-risk coverage curves for *prior-scale* in *gym-mujoco* environments.

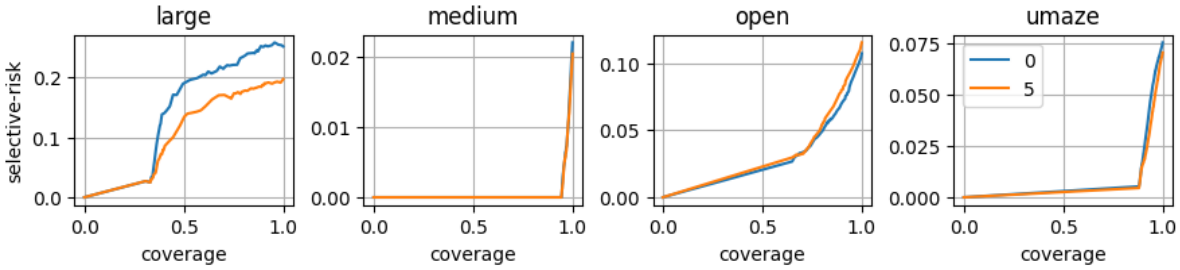


Figure 9: Selective-risk coverage curves for *prior-scale* in *maze* environments

B.2 Deterministic Model

In Tables 11, 12 and Figures 10,11, we share the choice of having a *deterministic(True)* versus *stochastic(False)* dynamics model. Though, the mean of stochastic model is lower, it's not significant as it's within confidence interval of the deterministic model.

Table 11: Evaluation metrics for *deterministic* model comparison in *gym-mujoco* environments.

Env.	deterministic	AURCC(↓)	RPP(↓)	CR_K(↑)	loss(↓)
Half Cheetah	False	0.229 ± 0.007	0.054 ± 0.006	0.568 ± 0.057	0.377 ± 0.005
	True	0.219 ± 0.005	0.044 ± 0.005	0.46 ± 0.04	0.374 ± 0.004
Hopper	False	0.138 ± 0.002	0.039 ± 0.002	0.524 ± 0.039	0.26 ± 0.003
	True	0.141 ± 0.005	0.031 ± 0.005	0.428 ± 0.034	0.268 ± 0.004
Walker 2d	False	$0.064 \pm (< 0.001)$	0.012 ± 0.001	0.328 ± 0.024	0.16 ± 0.002
	True	0.068 ± 0.001	0.011 ± 0.003	0.3 ± 0.051	0.159 ± 0.002

Table 12: Evaluation metrics for *deterministic* model comparison in *maze* environments.

Env.	deterministic	AURCC(↓)	RPP(↓)	CR_K(↑)	loss(↓)
large	False	0.152 ± 0.01	0.058 ± 0.007	0.54 ± 0.043	0.167 ± 0.003
	True	0.14 ± 0.015	0.062 ± 0.004	0.82 ± 0.035	0.251 ± 0.029
medium	False	$0.0 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	$0.003 \pm (< 0.001)$
	True	$0.001 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.022 ± 0.001
open	False	0.037 ± 0.001	$0.01 \pm (< 0.001)$	$0.4 \pm (< 0.001)$	0.143 ± 0.005
	True	0.029 ± 0.001	0.012 ± 0.001	$0.5 \pm (< 0.001)$	0.107 ± 0.005
umaze	False	$0.004 \pm (< 0.001)$	$0.001 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.059 ± 0.004
	True	0.008 ± 0.001	$0.002 \pm (< 0.001)$	$0.3 \pm (< 0.001)$	0.075 ± 0.003

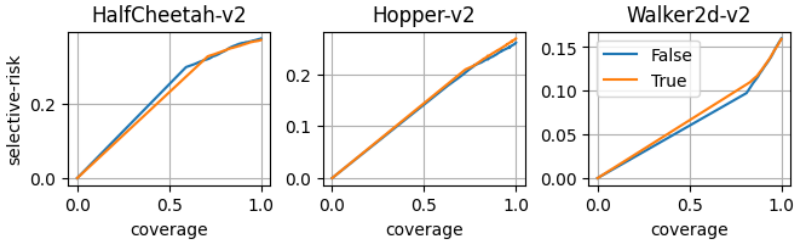


Figure 10: Selective-risk coverage curves for *deterministic* in *gym-mujoco* environments

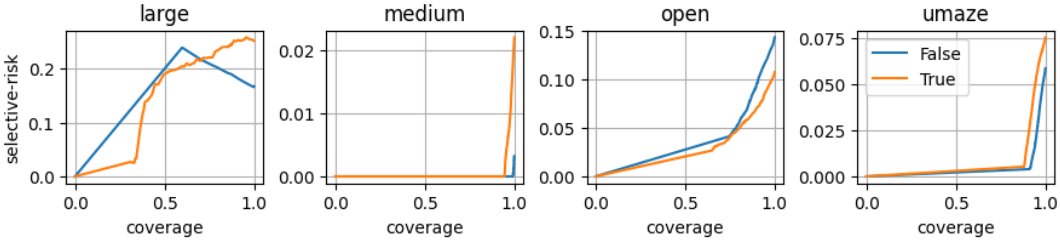


Figure 11: Selective-risk coverage curves for *deterministic* in *maze* environments

B.3 Dynamics Type

Table 13: Evaluation metrics for *dynamics-type* comparison in *maze* environments

ENV.	DYNAMICS-TYPE	AURCC(↓)	RPP(↓)	CR_K(↑)	LOSS(↓)
LARGE	AUTOREGRESSIVE	0.131 ± 0.017	0.06 ± 0.003	$0.8 \pm (< 0.001)$	0.233 ± 0.044
	FEED-FORWARD	0.14 ± 0.015	0.062 ± 0.004	0.82 ± 0.035	0.251 ± 0.029
MEDIUM	AUTOREGRESSIVE	$0.001 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.031 ± 0.001
	FEED-FORWARD	$0.001 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.022 ± 0.001
OPEN	AUTOREGRESSIVE	0.034 ± 0.003	0.014 ± 0.002	$0.5 \pm (< 0.001)$	0.123 ± 0.006
	FEED-FORWARD	0.029 ± 0.001	0.012 ± 0.001	$0.5 \pm (< 0.001)$	0.107 ± 0.005
UMAZE	AUTOREGRESSIVE	0.007 ± 0.001	$0.002 \pm (< 0.001)$	$0.3 \pm (< 0.001)$	0.07 ± 0.002
	FEED-FORWARD	0.008 ± 0.001	$0.002 \pm (< 0.001)$	$0.3 \pm (< 0.001)$	0.075 ± 0.003

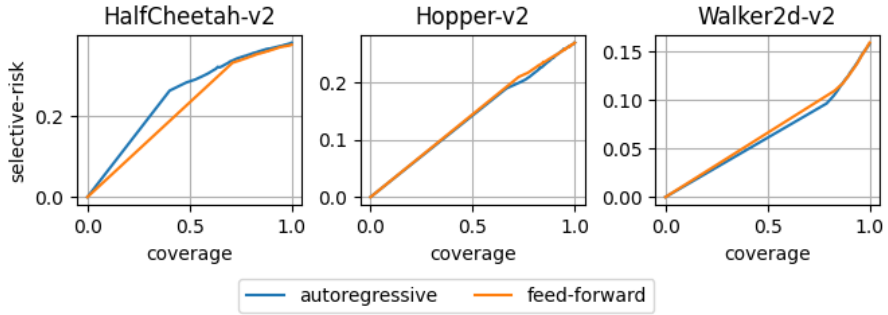


Figure 12: Selective-risk coverage curves for *dynamics-type* in *gym-mujoco* environments

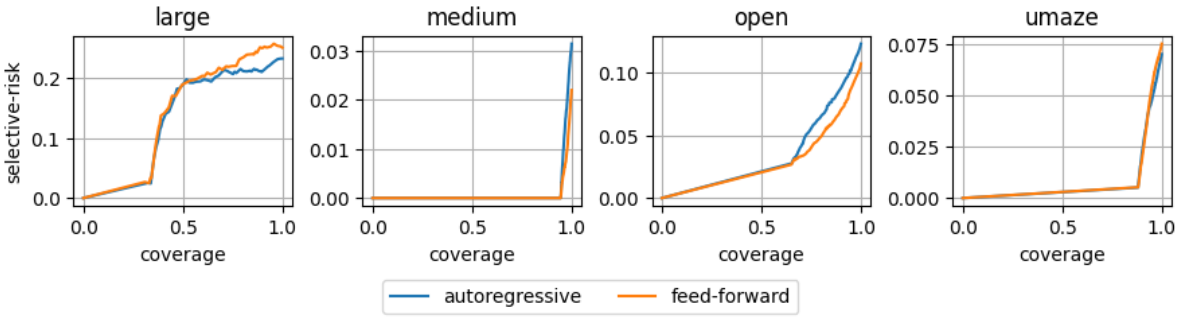


Figure 13: Selective-risk coverage curves for *dynamics-type* in *maze* environments

B.4 Normalization of input state-space

In Tables 14,15 and Figures 14,15, we investigate the imapct of learning dynamics with normalized state-space. Here, *'True'* implies the dynamics was learned with normalized state-space and *'False'* implies otherwise. There is marginal performance difference between either choice for MuJoco environments. Maze2D environments show a mix of results with normalization benefiting Umaze and hurting Large-Maze. Performance of Medium-Maze and Open-Maze is not impacted significantly.

Table 14: Evaluation metrics for *normalize* comparison in *gym-mujoco* environments

Env.	normalize	AURCC(↓)	RPP(↓)	CR_K(↑)	loss(↓)
Half Cheetah	False	0.228 ± 0.006	0.053 ± 0.007	0.548 ± 0.07	0.376 ± 0.003
	True	0.219 ± 0.005	0.044 ± 0.005	0.46 ± 0.04	0.374 ± 0.004
Hopper	False	0.128 ± 0.002	0.017 ± 0.002	0.3 ± 0.025	0.255 ± 0.003
	True	0.141 ± 0.005	0.031 ± 0.005	0.428 ± 0.034	0.268 ± 0.004
Walker 2d	False	0.078 ± 0.007	0.013 ± 0.005	0.316 ± 0.073	0.17 ± 0.009
	True	0.068 ± 0.001	0.011 ± 0.003	0.3 ± 0.051	0.159 ± 0.002

Table 15: Evaluation metrics for *normalize* comparison in *maze* environments

Env.	normalize	AURCC(↓)	RPP(↓)	CR_K(↑)	loss(↓)
large	False	0.215 ± 0.027	0.067 ± 0.007	0.82 ± 0.035	0.402 ± 0.037
	True	0.14 ± 0.015	0.062 ± 0.004	0.82 ± 0.035	0.251 ± 0.029
medium	False	$0.0 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.017 ± 0.002
	True	$0.001 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.022 ± 0.001
open	False	0.033 ± 0.001	$0.014 \pm (< 0.001)$	$0.5 \pm (< 0.001)$	0.108 ± 0.004
	True	0.029 ± 0.001	0.012 ± 0.001	$0.5 \pm (< 0.001)$	0.107 ± 0.005
umaze	False	$0.003 \pm (< 0.001)$	$0.002 \pm (< 0.001)$	$0.3 \pm (< 0.001)$	0.047 ± 0.004
	True	0.008 ± 0.001	$0.002 \pm (< 0.001)$	$0.3 \pm (< 0.001)$	0.075 ± 0.003

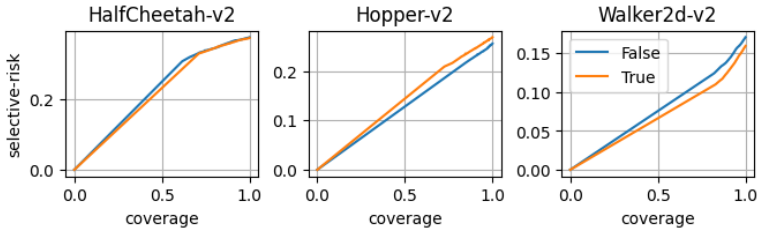


Figure 14: Selective-risk coverage curves for *normalize* in *gym-mujoco* environments.

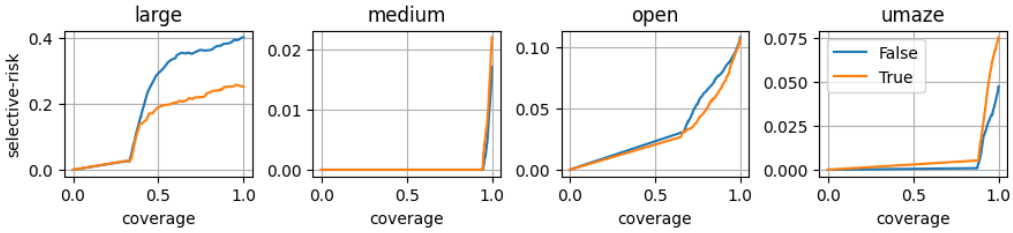


Figure 15: Selective-risk coverage curves for *normalize* in *maze* environments.

B.5 Uncertainty Types

In the following, *'EV, PCI, U-PCI'* refer to Ensemble-Voting, Paired Confidence interval and Unpaired Confidence Interval, respectively.

Table 16: Evaluation metrics for *uncertainty-type* comparison in *maze* environments

Env.	uncertainty-type	AURCC(↓)	RPP(↓)	CR_K(↑)	loss(↓)
MEDIUM	EV	$0.001 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.022 ± 0.001
	PCI	$0.001 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.009 ± 0.001
	U-PCI	$0.001 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.009 ± 0.001
OPEN	EV	0.029 ± 0.001	0.012 ± 0.001	$0.5 \pm (< 0.001)$	0.107 ± 0.005
	PCI	0.057 ± 0.004	0.012 ± 0.001	0.38 ± 0.035	0.168 ± 0.008
	U-PCI	0.05 ± 0.002	0.012 ± 0.001	$0.4 \pm (< 0.001)$	0.168 ± 0.008
UMAZE	EV	0.008 ± 0.001	$0.002 \pm (< 0.001)$	$0.3 \pm (< 0.001)$	0.075 ± 0.003
	PCI	0.035 ± 0.002	$0.001 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.084 ± 0.003
	U-PCI	0.017 ± 0.002	$0.001 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.084 ± 0.003

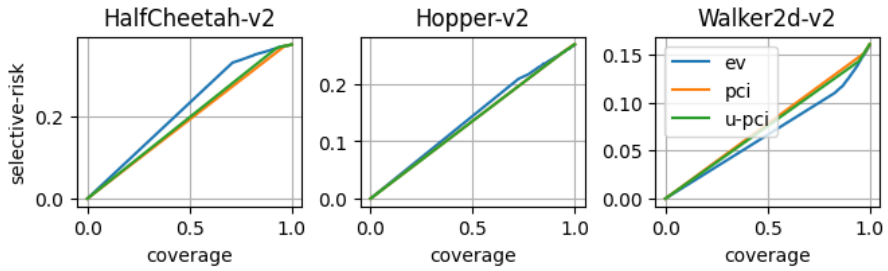


Figure 16: Selective-risk coverage curves for *uncertainty-type* in *gym-mujoco* environments.

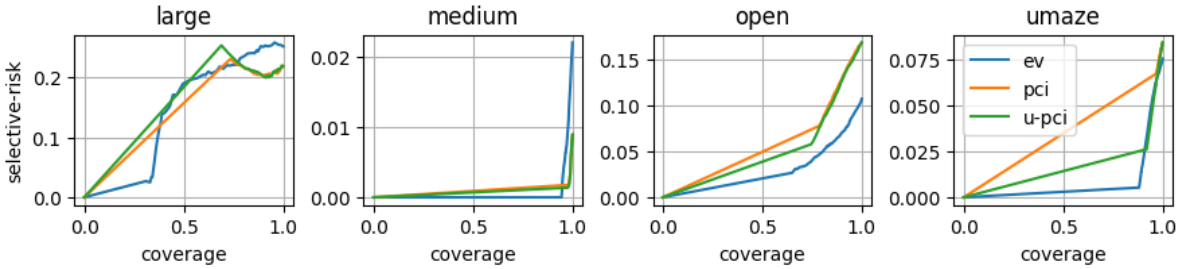


Figure 17: Selective-risk coverage curves for *uncertainty-type* in *maze* environments.

B.6 Horizon

Table 17: Evaluation metrics for *horizon* comparison in *maze* environments

Env.	horizon	AURCC(↓)	RPP(↓)	CR_K(↑)	loss(↓)
Env.	horizon	AURCC(↓)	RPP(↓)	CR_K(↑)	loss(↓)
large	20.0	$0.028 \pm (< 0.001)$	$0.021 \pm (< 0.001)$	0.62 ± 0.035	$0.059 \pm (< 0.001)$
	30.0	0.118 ± 0.013	0.047 ± 0.004	$0.8 \pm (< 0.001)$	0.295 ± 0.037
	40.0	0.218 ± 0.018	0.087 ± 0.004	0.82 ± 0.035	0.321 ± 0.027
	50.0	0.16 ± 0.018	0.07 ± 0.005	0.92 ± 0.035	0.247 ± 0.038
medium	20.0	$0.0 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.006 ± 0.001
	30.0	$0.0 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.019 ± 0.001
	40.0	$0.001 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.032 ± 0.003
	50.0	$0.001 \pm (< 0.001)$	$0.001 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	$0.031 \pm (< 0.001)$
open	20.0	$0.005 \pm (< 0.001)$	$0.001 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.029 ± 0.001
	30.0	0.015 ± 0.002	0.007 ± 0.001	$0.5 \pm (< 0.001)$	0.104 ± 0.006
	40.0	0.036 ± 0.002	0.016 ± 0.001	$0.6 \pm (< 0.001)$	0.119 ± 0.007
	50.0	0.062 ± 0.002	0.025 ± 0.001	$0.6 \pm (< 0.001)$	0.148 ± 0.006
umaze	20.0	$0.0 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.007 ± 0.001
	30.0	$0.0 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.012 ± 0.002
	40.0	0.006 ± 0.001	$0.002 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.048 ± 0.003
	50.0	0.035 ± 0.002	0.008 ± 0.001	$0.4 \pm (< 0.001)$	0.195 ± 0.007

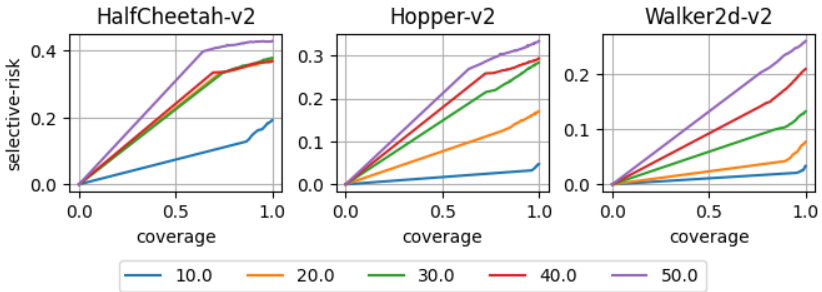


Figure 18: Selective-risk coverage curves for *horizon* in *gym-mujoco* environments.

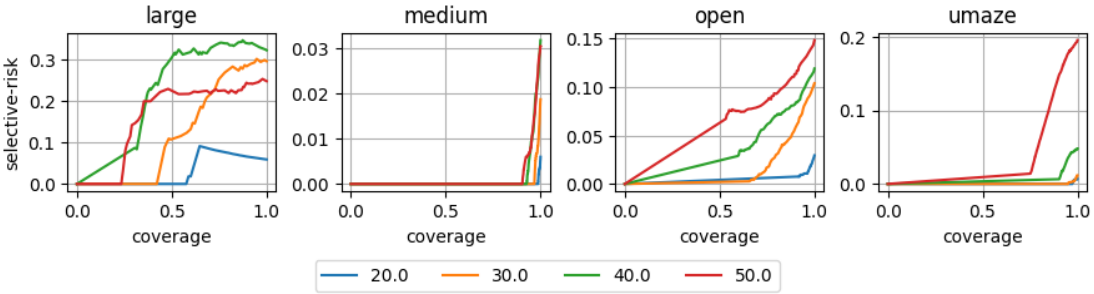


Figure 19: Selective-risk coverage curves for *horizon* in *maze* environments

B.7 Ensemble-Count

Table 18: Evaluation metrics for *ensemble-count* comparison in *maze* environments.

Env.	ensemble-count	AURCC(↓)	RPP(↓)	CR_K(↑)	loss(↓)
large	10	0.168 ± 0.044	0.051 ± 0.011	$0.6 \pm (< 0.001)$	0.307 ± 0.061
	20	0.149 ± 0.039	0.057 ± 0.015	0.78 ± 0.066	0.269 ± 0.065
	40	0.138 ± 0.031	0.056 ± 0.011	0.82 ± 0.035	0.264 ± 0.044
	80	0.146 ± 0.019	0.061 ± 0.007	0.82 ± 0.035	0.269 ± 0.027
	100	0.14 ± 0.015	0.062 ± 0.004	0.82 ± 0.035	0.251 ± 0.029
medium	10	$0.001 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.023 ± 0.007
	20	$0.001 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.024 ± 0.006
	40	$0.001 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.021 ± 0.003
	80	$0.001 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.022 ± 0.001
	100	$0.001 \pm (< 0.001)$	$0.0 \pm (< 0.001)$	$0.2 \pm (< 0.001)$	0.022 ± 0.001
open	10	0.033 ± 0.004	$0.009 \pm (< 0.001)$	0.48 ± 0.035	0.127 ± 0.015
	20	0.031 ± 0.003	$0.01 \pm (< 0.001)$	$0.5 \pm (< 0.001)$	0.117 ± 0.019
	40	0.03 ± 0.003	0.011 ± 0.001	$0.5 \pm (< 0.001)$	0.11 ± 0.014
	80	0.029 ± 0.002	0.012 ± 0.001	$0.5 \pm (< 0.001)$	0.108 ± 0.007
	100	0.029 ± 0.001	0.012 ± 0.001	$0.5 \pm (< 0.001)$	0.107 ± 0.005
umaze	10	0.011 ± 0.002	$0.002 \pm (< 0.001)$	$0.3 \pm (< 0.001)$	0.073 ± 0.006
	20	0.009 ± 0.001	$0.002 \pm (< 0.001)$	$0.3 \pm (< 0.001)$	0.074 ± 0.004
	40	0.008 ± 0.001	$0.002 \pm (< 0.001)$	$0.3 \pm (< 0.001)$	0.077 ± 0.004
	80	0.008 ± 0.001	$0.002 \pm (< 0.001)$	$0.3 \pm (< 0.001)$	0.075 ± 0.004
	100	0.008 ± 0.001	$0.002 \pm (< 0.001)$	$0.3 \pm (< 0.001)$	0.075 ± 0.003

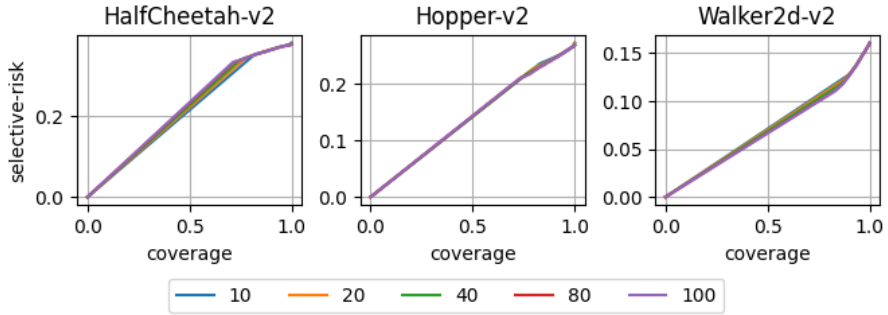


Figure 20: Selective-risk coverage curves for *ensemble-count* in *gym-mujoco* environments

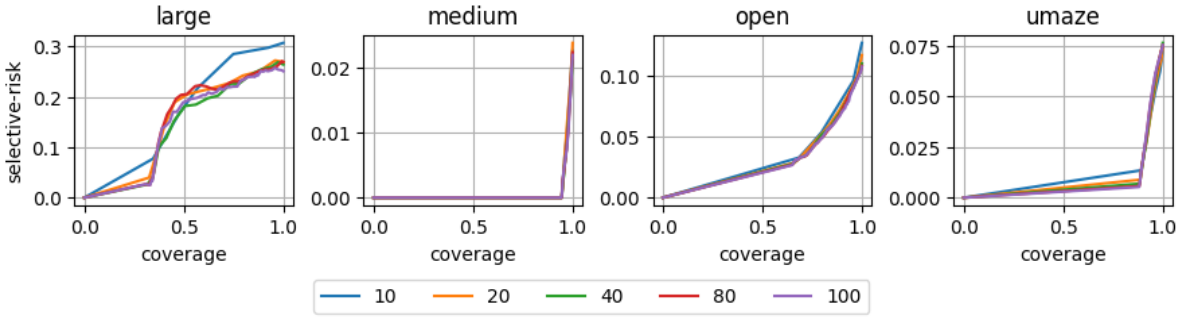


Figure 21: Selective-risk coverage curves for *ensemble-count* in *maze* environments