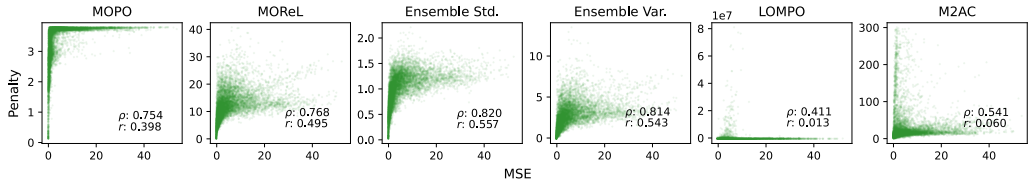
# A Calibration

## A.1 Choice of calibration metrics

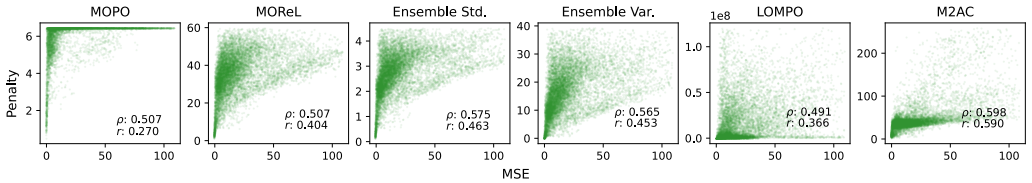
We consider both the Spearman rank correlation and Pearson bivariate correlation. We believe that the former better represents the actual statistical power of the metric compared to the true distributional shift value, as it is robust to outliers and isn't impacted by distributional shape (i.e., skewness, kurtosis). After all, we do not know if some 'true' $|G^{\pi}_{\hat{M}}(s,a)|$ is even linearly correlated with the MSE values that we report, so naïvely comparing based on bivariate correlation may result in incorrect assessment of penalty efficacy. However, we do also include the Pearson bivariate correlation to help us gain insight into how the penalty distribution shape changes with design choices. For instance, consider two metrics that have identical Spearman coefficients, but vastly different Pearson coefficients–this implies they have significantly different distributions.

## A.2 Offline Dataset Transfer Calibration

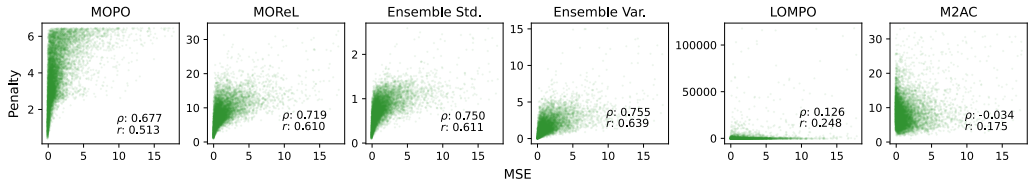
### A.2.1 HalfCheetah



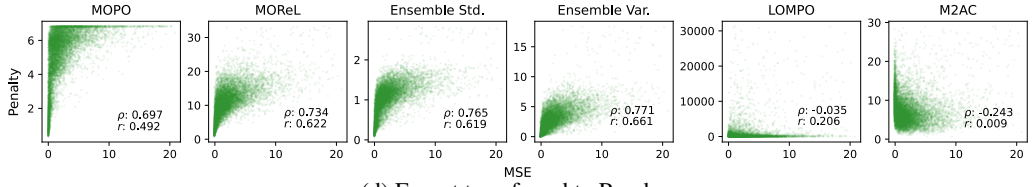

(a) Random transferred to Expert




(b) Medium transferred to Expert



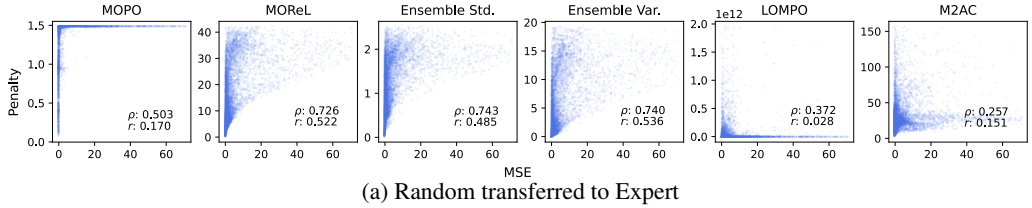

(c) Medium transferred to Random



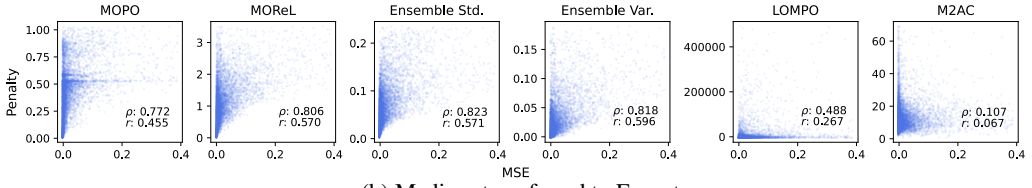

(d) Expert transferred to Random

Figure 6: Scatter Plots showing HalfCheetah D4RL transfer tasks.

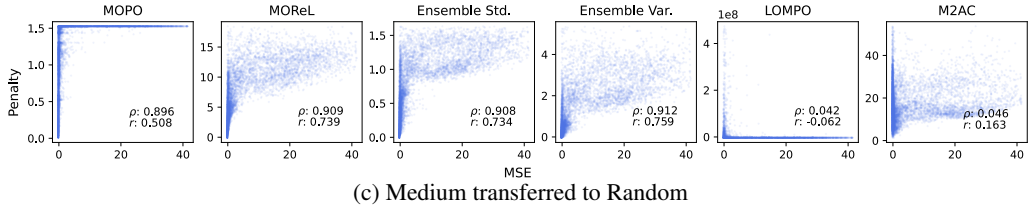
### A.2.2 Hopper



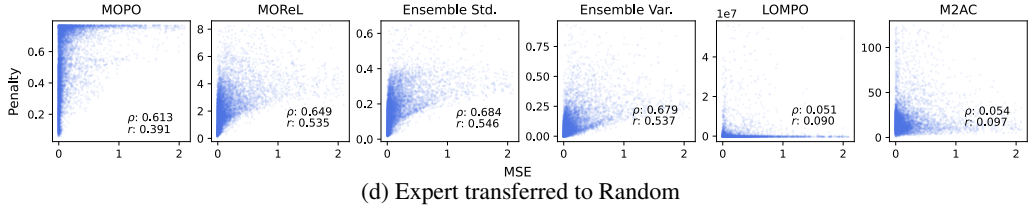

(a) Random transferred to Expert




(b) Medium transferred to Expert




(c) Medium transferred to Random

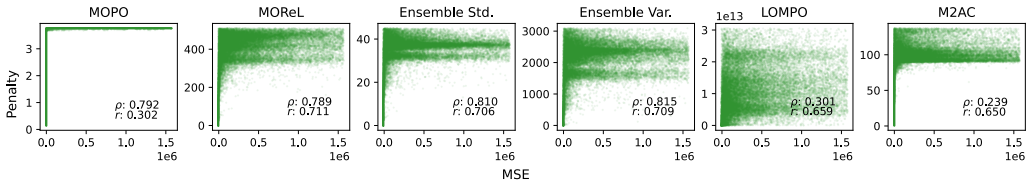



(d) Expert transferred to Random

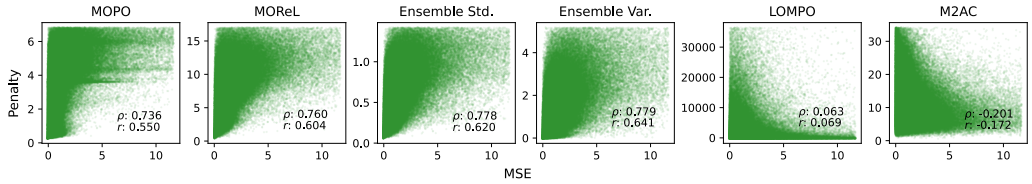
Figure 7: Scatter Plots showing Hopper D4RL transfer tasks.

## A.3 Ground Truth Calibration

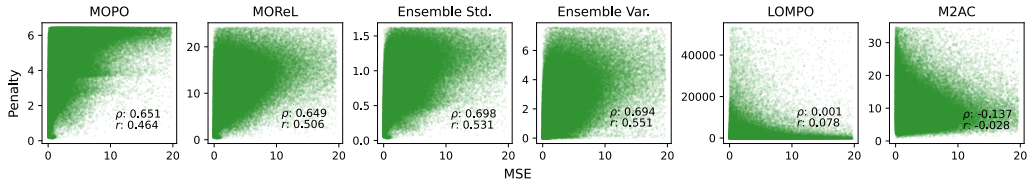
### A.3.1 HalfCheetah



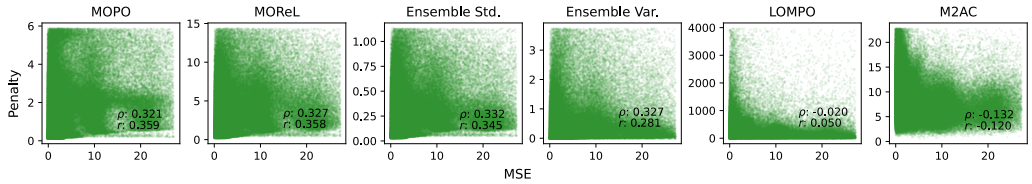

(a) Random



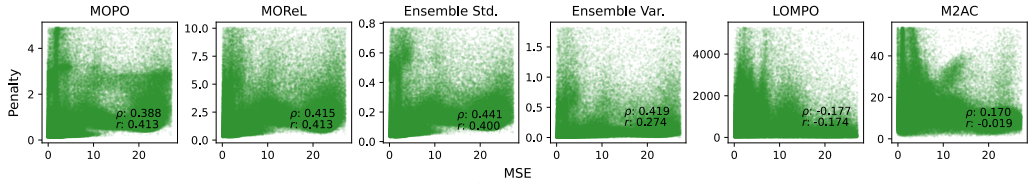

(b) Mixed




(c) Medium



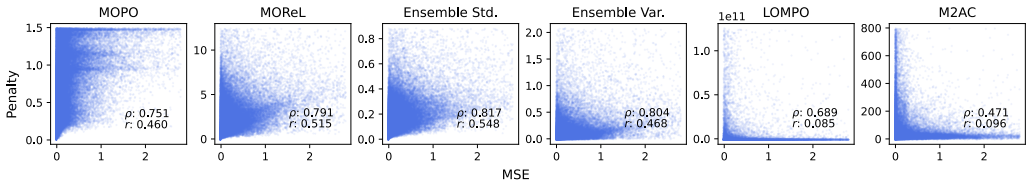

(d) Medium Expert



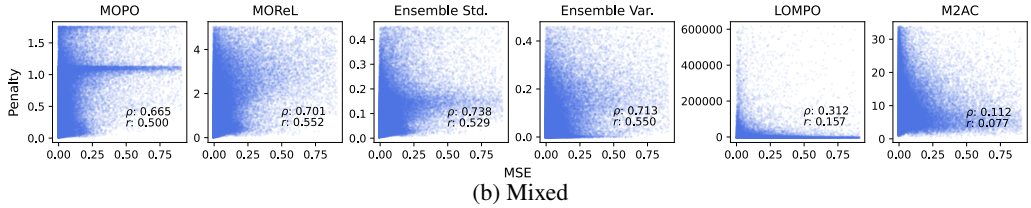

(e) Expert

Figure 8: Scatter Plots showing HalfCheetah D4RL ground truth calibration.

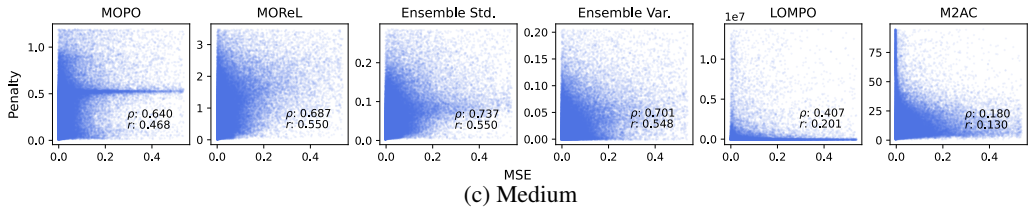
### A.3.2 Hopper



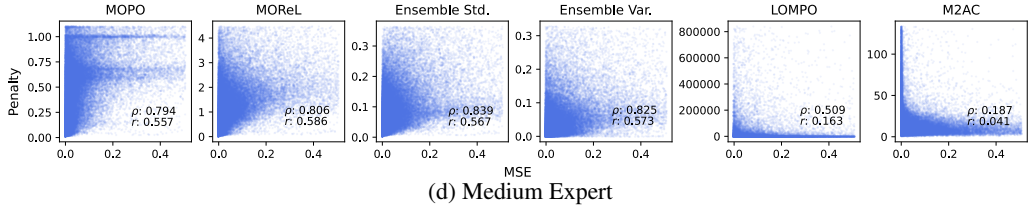

(a) Random



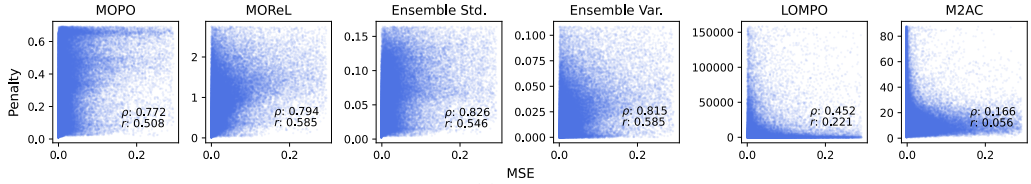

(b) Mixed




(c) Medium




(d) Medium Expert



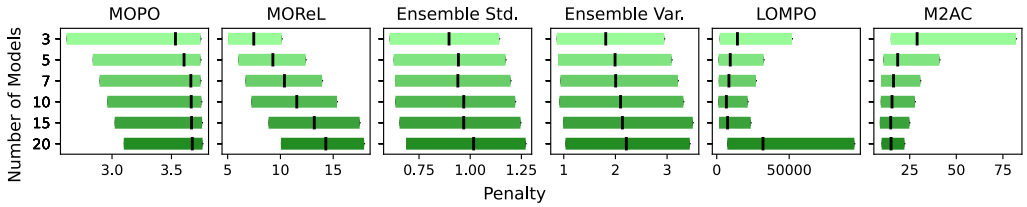

(e) Expert

Figure 9: Scatter Plots showing Hopper D4RL ground truth calibration.

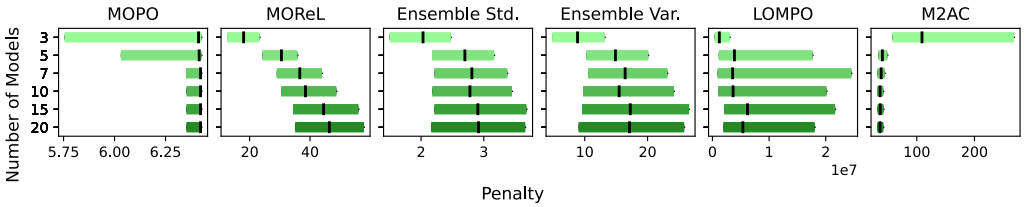
# B Full Results Increasing Models

## B.1 Penalty Distribution

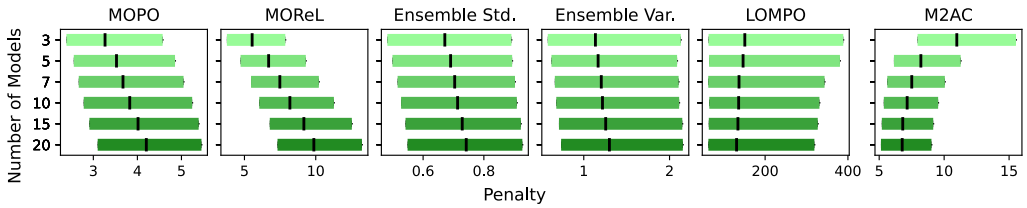
### B.1.1 Offline Dataset Transfer Distribution



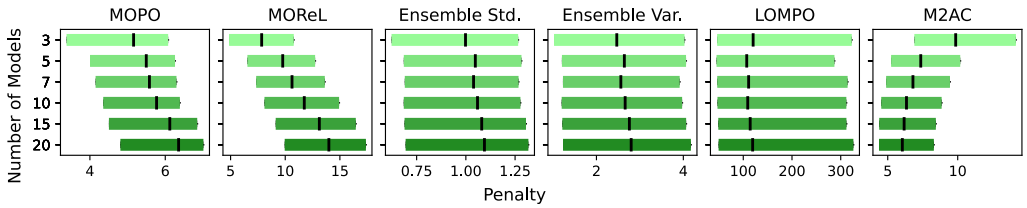

(a) Random transferred to Expert




(b) Medium transferred to Expert

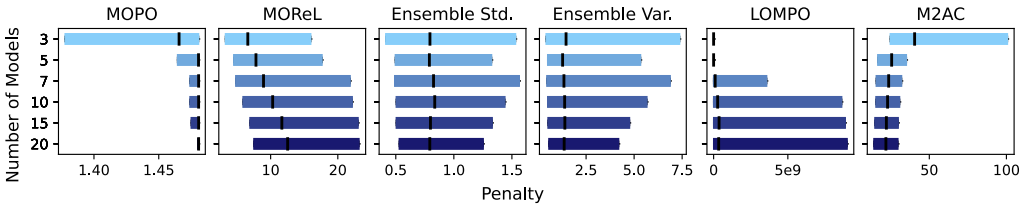



(c) Medium transferred to Random

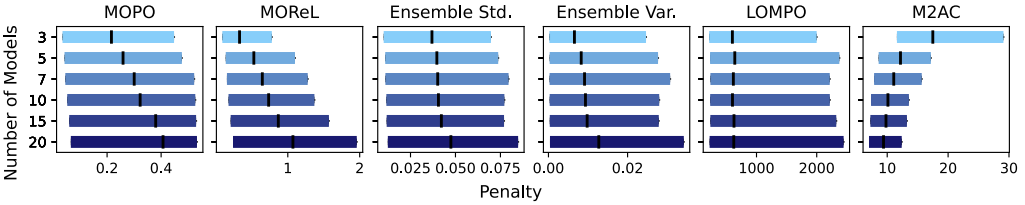



(d) Expert transferred to Random

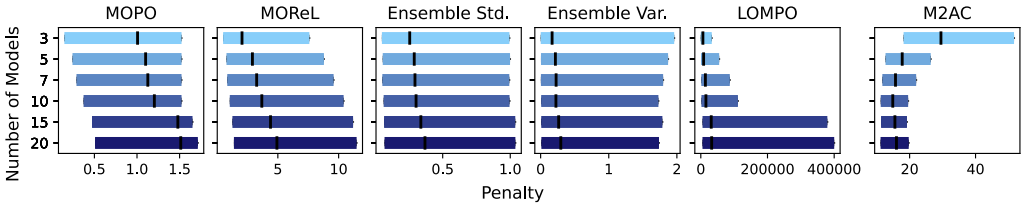
Figure 10: Box Plots showing HalfCheetah D4RL transfer tasks.



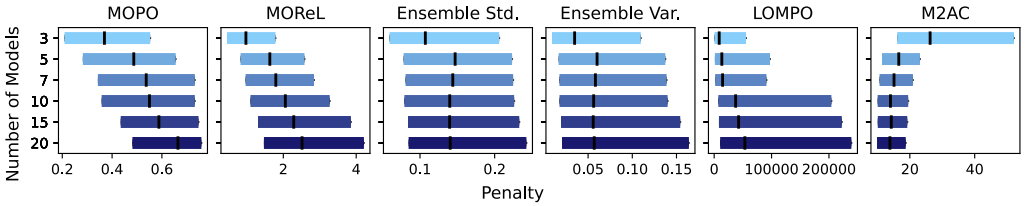

(a) Random transferred to Expert




(b) Medium transferred to Expert



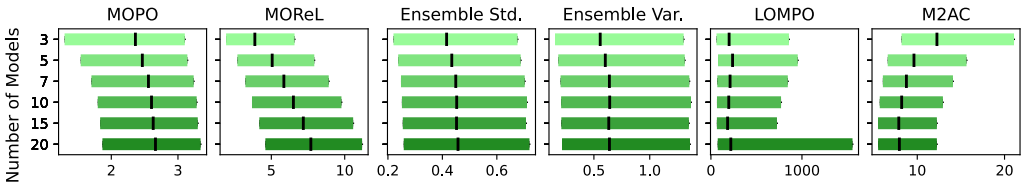

(c) Medium transferred to Random



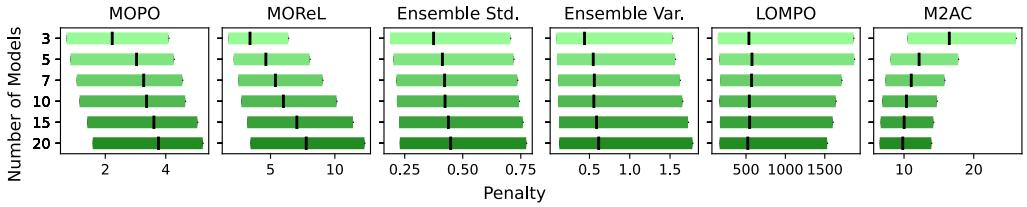

(d) Expert transferred to Random

Figure 11: Box Plots showing Hopper D4RL transfer tasks.

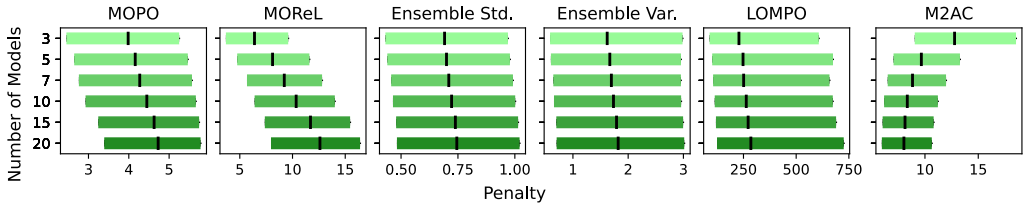
### B.1.2 Ground Truth Distribution



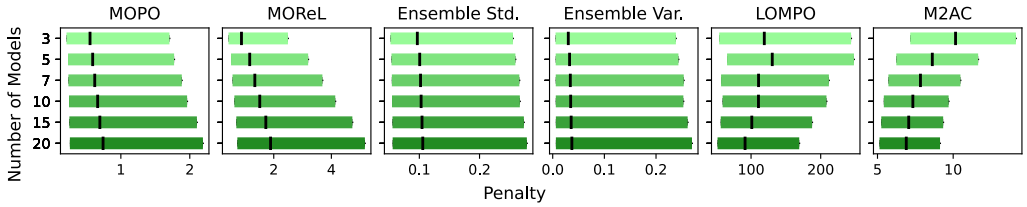

(a) Random



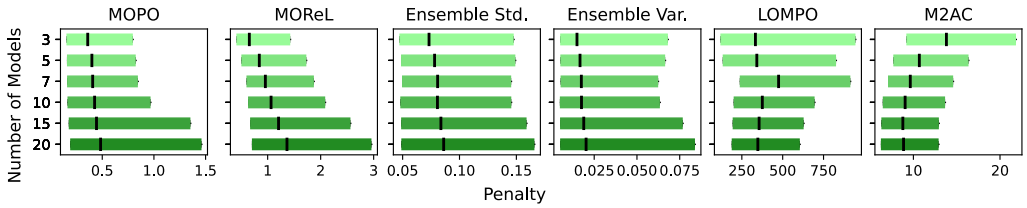

(b) Mixed




(c) Medium

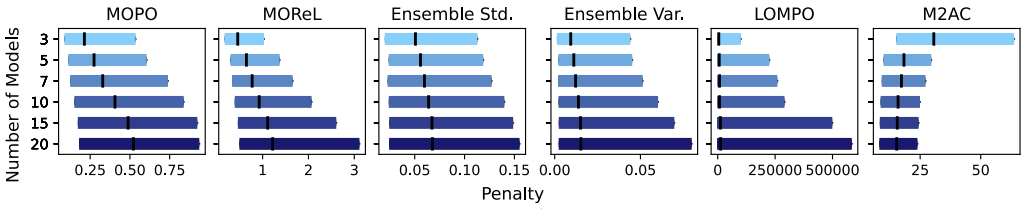



(d) Medium Expert

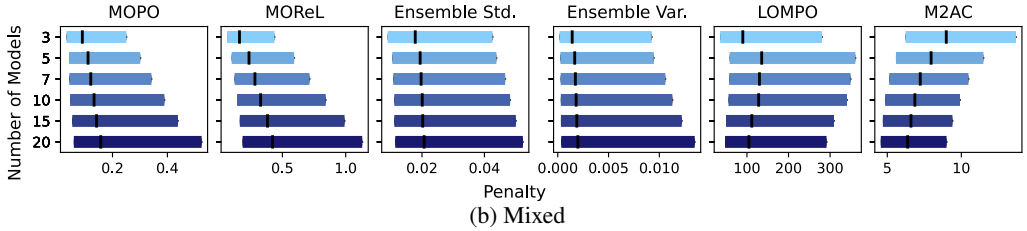



(e) Expert

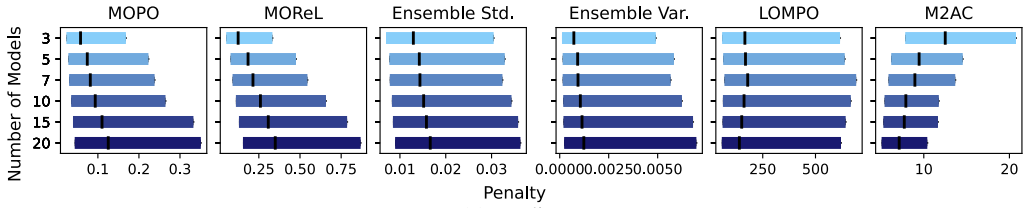
Figure 12: Boxplots showing HalfCheetah D4RL ground truth penalty distributions.



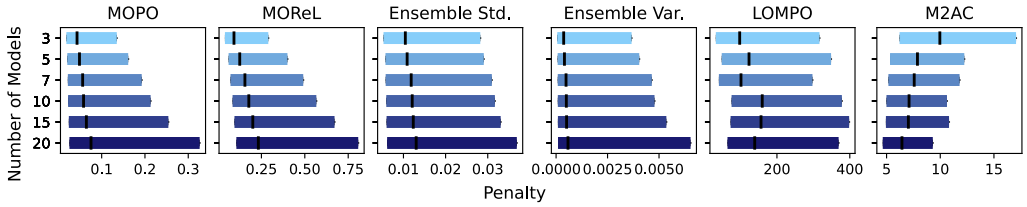

(a) Random



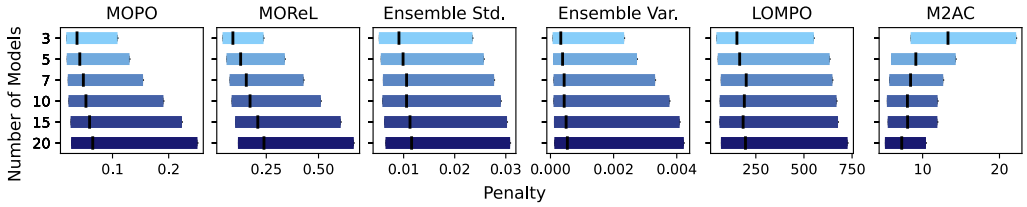

(b) Mixed




(c) Medium




(d) Medium Expert




(e) Expert

Figure 13: Boxplots showing Hopper D4RL ground truth penalty distributions.

## B.2 Penalty Performance

### B.2.1 HalfCheetah D4RL: Transfer

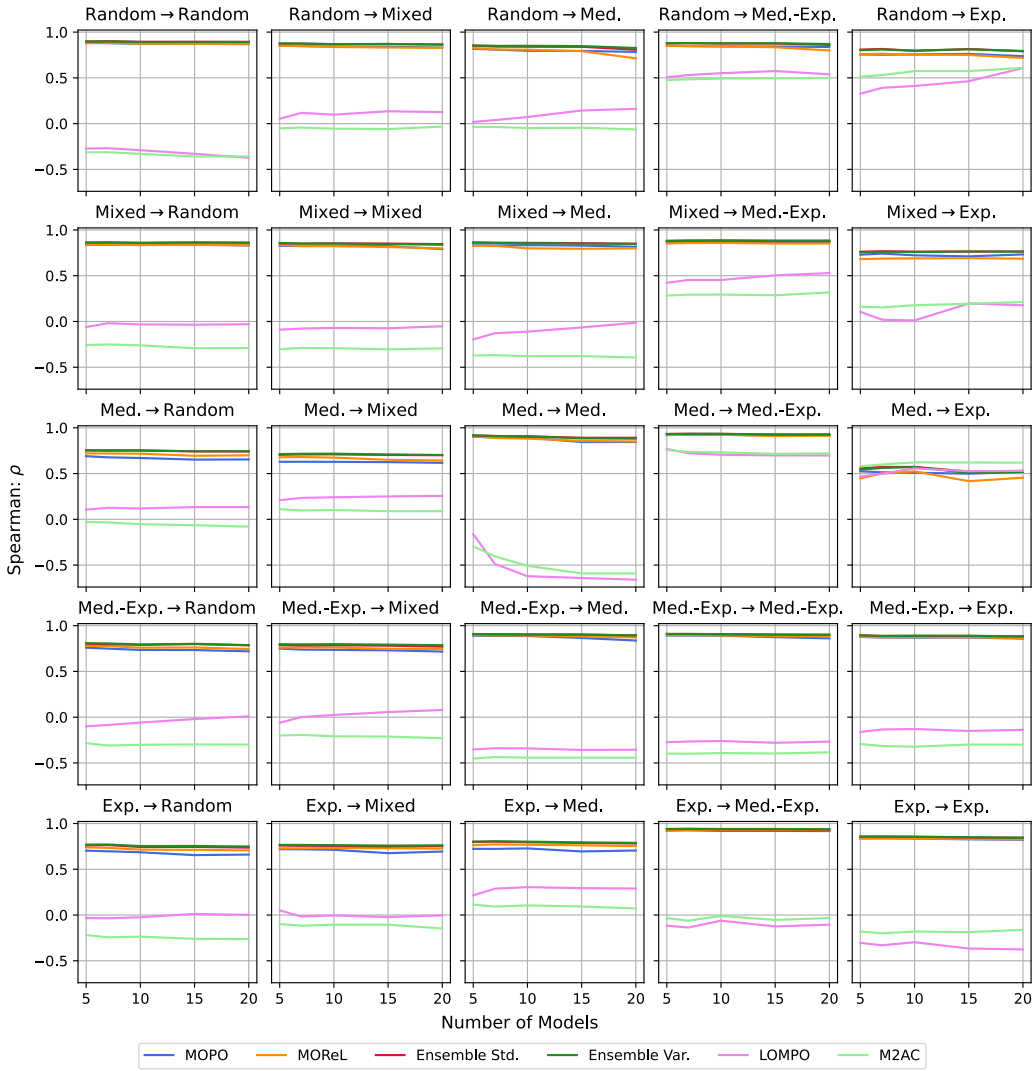



Figure 14: HalfCheetah Spearman Statistics

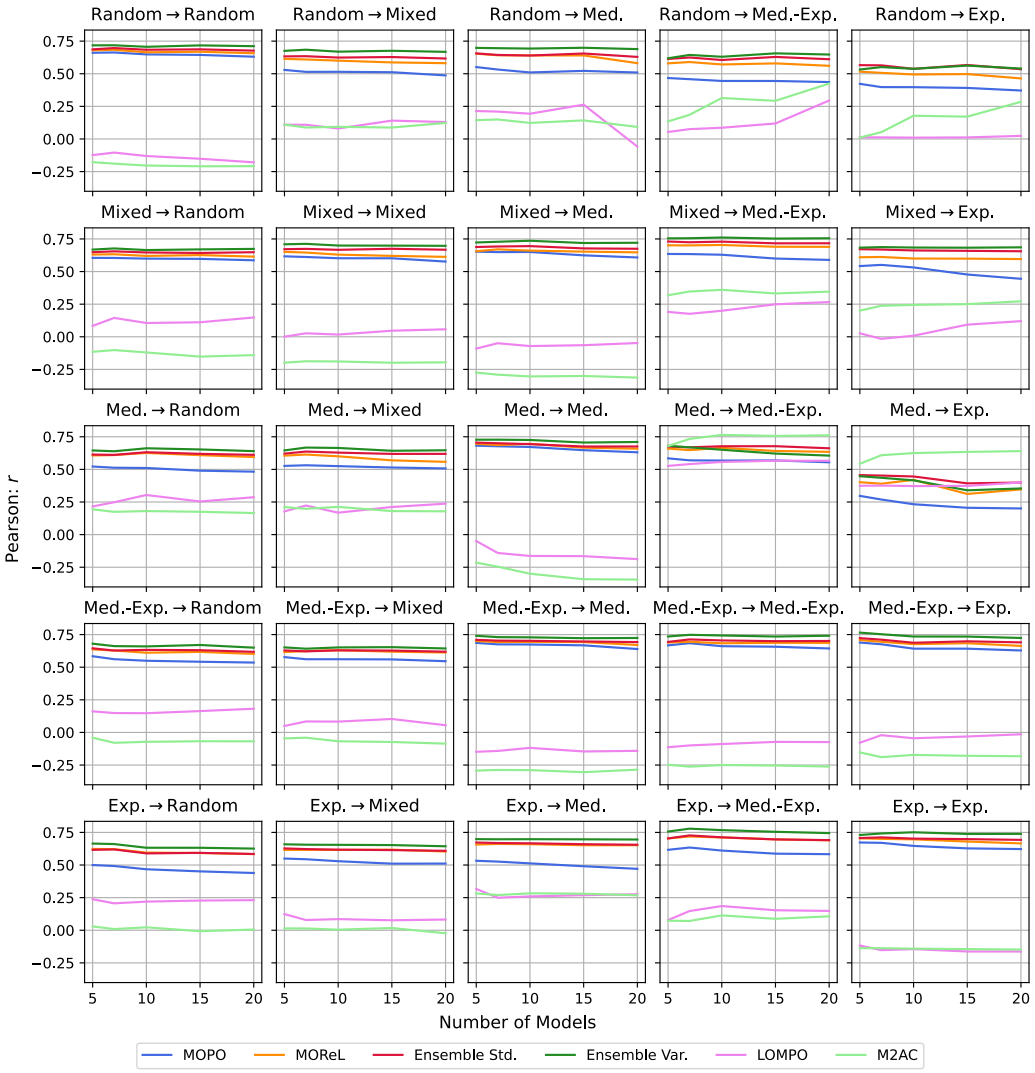



Figure 15: HalfCheetah Pearson Statistics

### B.2.2 Hopper D4RL: Transfer

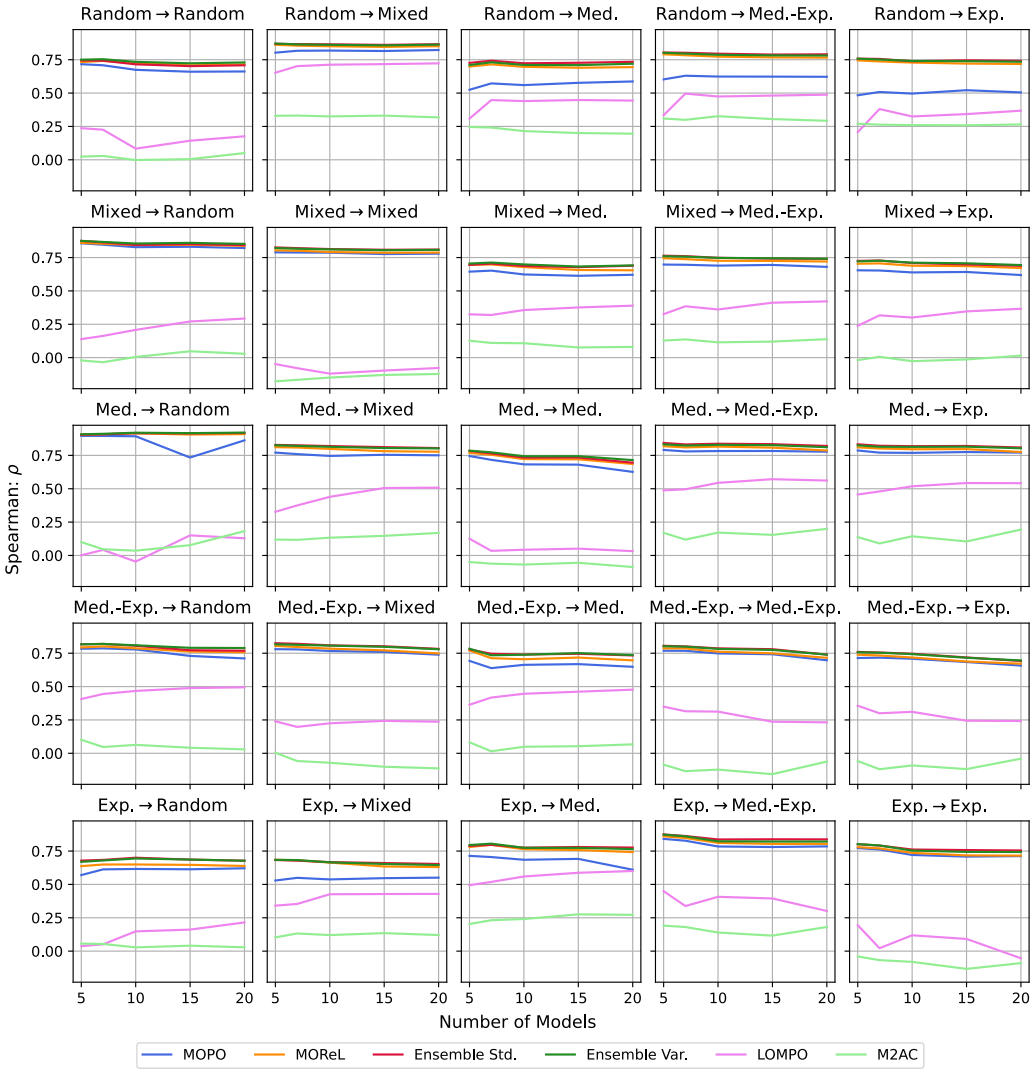



Figure 16: Hopper Spearman Statistics

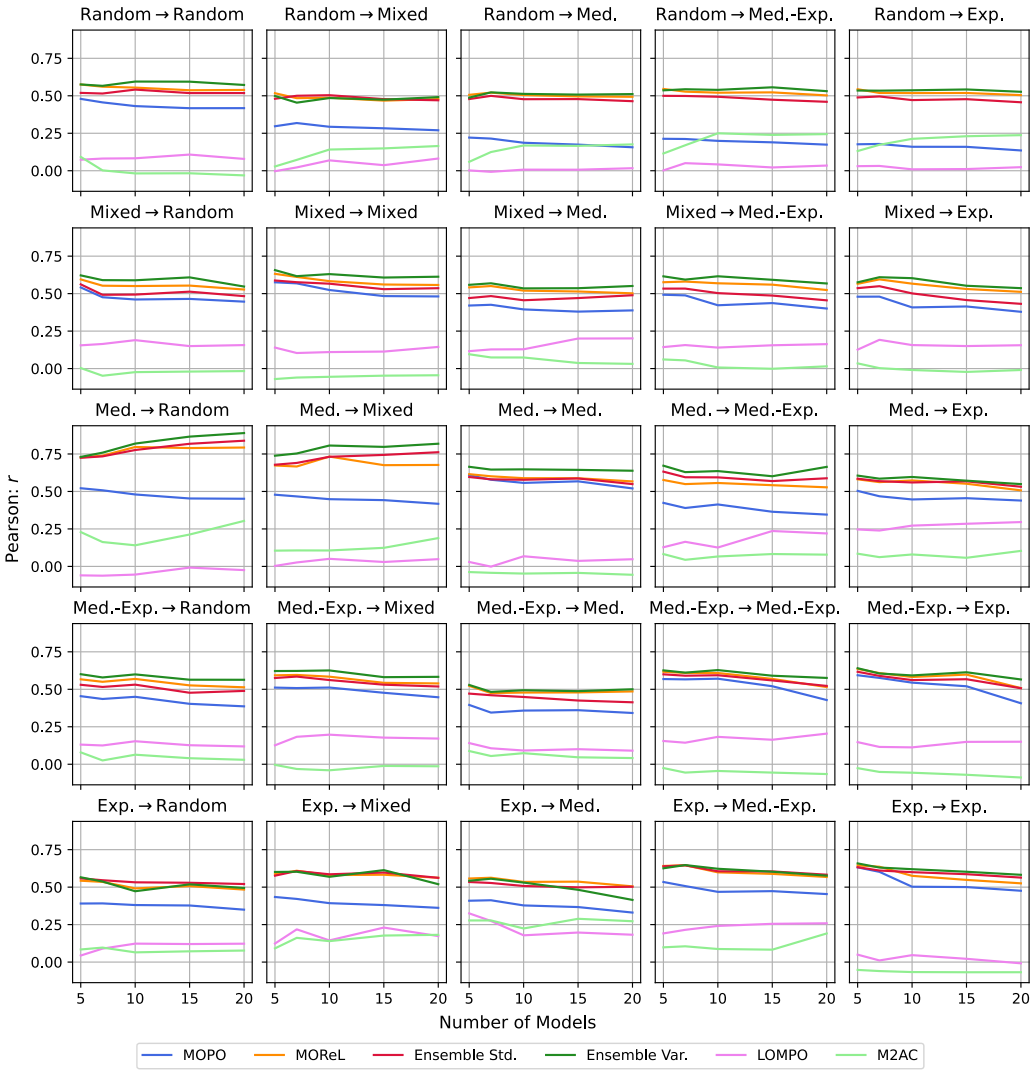



Figure 17: Hopper Pearson Statistics

### B.2.3 HalfCheetah D4RL: Ground Truth Dynamics

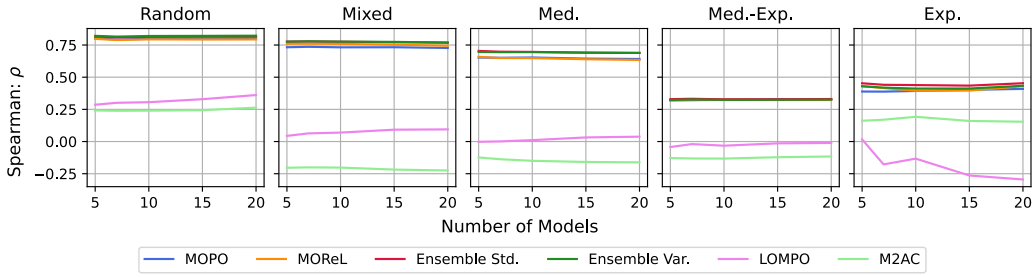



Figure 18: HalfCheetah Spearman Statistics

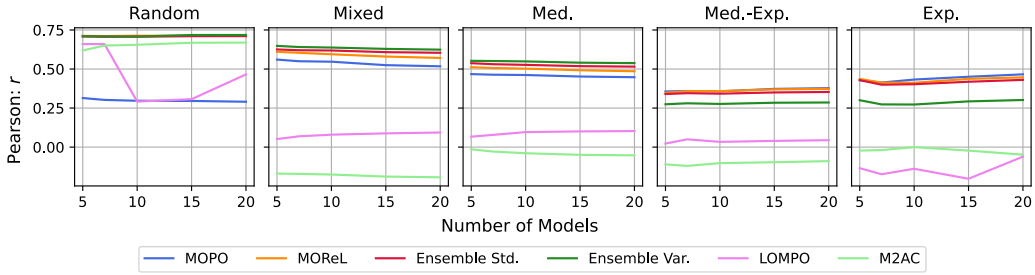



Figure 19: HalfCheetah Pearson Statistics

### B.2.4 Hopper D4RL: Ground Truth Dynamics

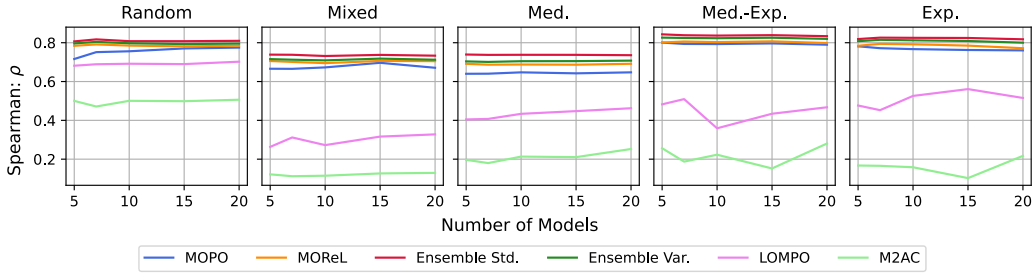



Figure 20: Hopper Spearman Statistics

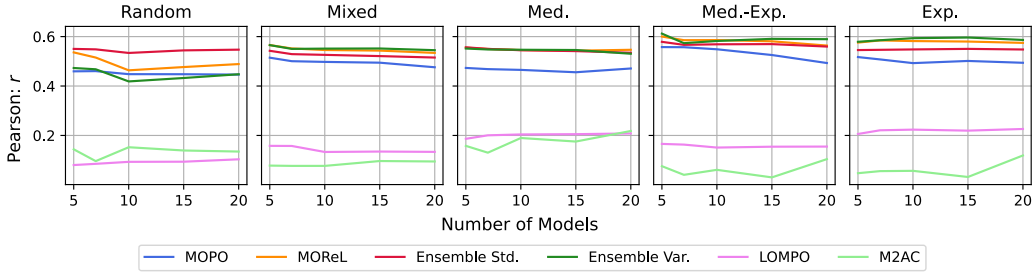



Figure 21: Hopper Pearson Statistics

# C Skewness and Kurtosis Comparisons

## C.1 Skewness and Kurtosis Overall

Table 5: Skew ($\gamma_1$) and Kurtosis ($\gamma_2$) statistics of all experiments averaged over different test settings using the MOPO Default of 7 models.

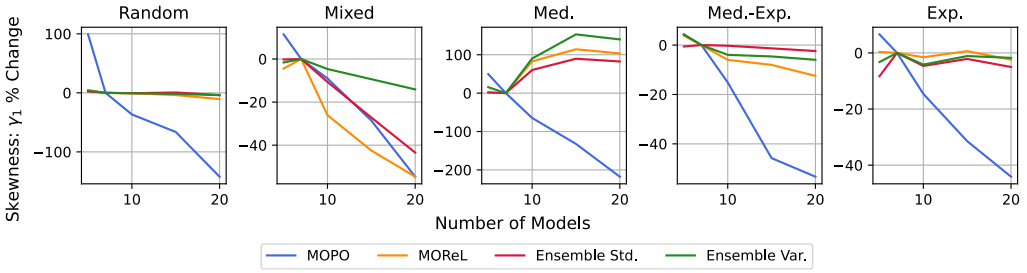
| | Transfer | | | | Ground Truth | | | |
|---|---|---|---|---|---|---|---|---|
| | HalfCheetah | | Hopper | | HalfCheetah | | Hopper | |
| **Penalty** | $\gamma_1$ | $\gamma_2$ | $\gamma_1$ | $\gamma_2$ | $\gamma_1$ | $\gamma_2$ | $\gamma_1$ | $\gamma_2$ |
| MOPO | -0.010 | 0.580 | 0.689 | 1.377 | 0.671 | 0.920 | 1.873 | 2.864 |
| MOReL | 0.919 | 0.957 | 1.967 | 4.578 | 1.661 | 3.081 | 2.571 | 7.465 |
| Ensemble Std. | 0.794 | 0.806 | 2.136 | 6.560 | 1.656 | 3.178 | 2.739 | 9.061 |
| Ensemble Var. | 1.823 | 4.830 | 3.436 | 15.983 | 2.612 | 8.800 | 4.517 | 25.380 |
| LOMPO | 6.893 | 114.843 | 10.920 | 180.716 | 5.100 | 37.865 | 14.415 | 251.705 |
| M2AC | 1.778 | 5.729 | 3.729 | 29.606 | 1.840 | 4.600 | 4.008 | 28.089 |

## C.2 Skew and Kurtosis Scaling with Model Count

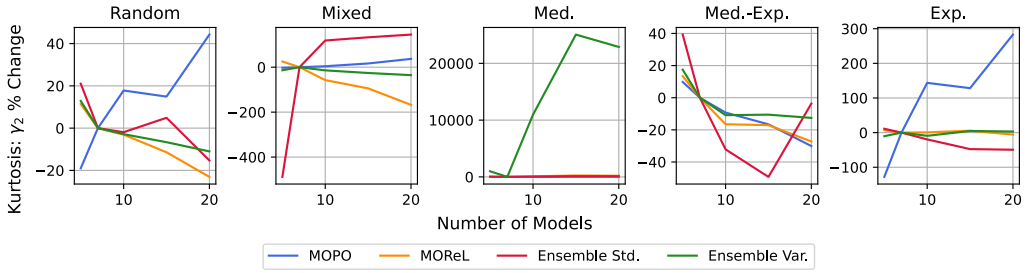
We omit LOMPO and M2AC due to the fact that their changes were so significant as to obfuscate the changes of the more performant penalties.

We choose 7 models to act as our 'baseline' (following the default MOPO setting), and we measure the change in the skew and kurtosis relative to this, hence 7 models always has a $0\%$ change in our graphs. For brevity, in the transfer experiments, we average over all 'transferred to' environments, e.g., Random, Medium, etc.; the graph title refers to the data that the model was trained on.

Again, we observe the environment *and* setting dependency of these metrics, sometimes having increasing skewness and kurtosis with model count, and other times decreasing. This further justifies using a ranking metric to compare penalties, as the overall penalty shape can vary hugely and unpredictably w.r.t. co-dependent hyperparameters. We do observe however in the ground truth experiments that ensemble standard deviation appears to be most robust to scaling with models. We also observe that the MOPO penalty can change shape significantly w.r.t. model count, and all penalties are not fully immune to this. This further advocates the use of shape meta-parameters to control for changing distribution properties when adjusting the number of models as a hyperparameter, as well as selecting penalties that are relatively invariant to model count to make tuning easier.

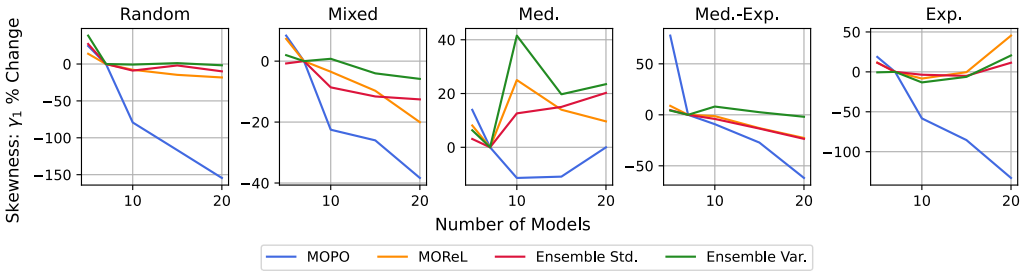



(a) Skewness Scaling

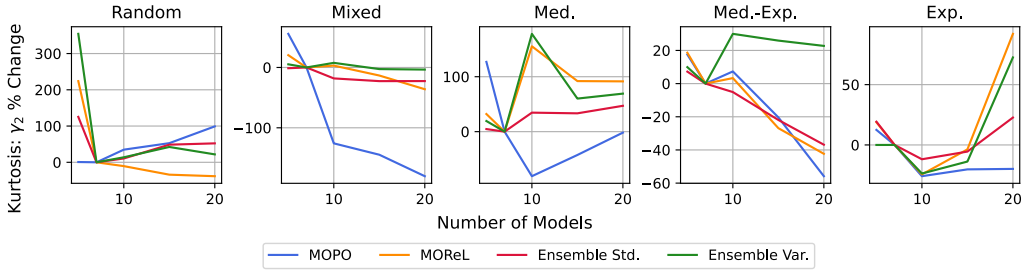



(b) Kurtosis Scaling

Figure 22: HalfCheetah Transfer.

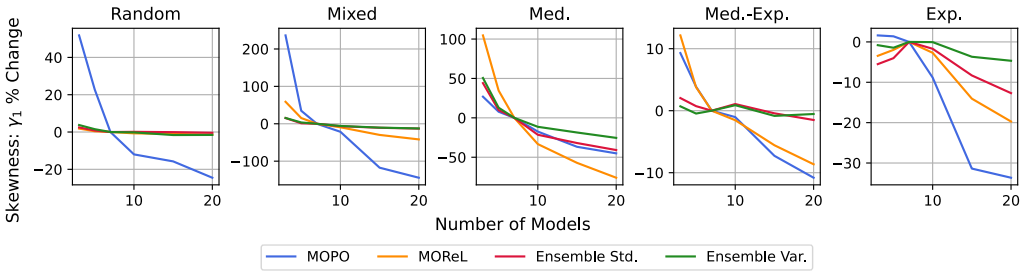



(a) Skewness Scaling

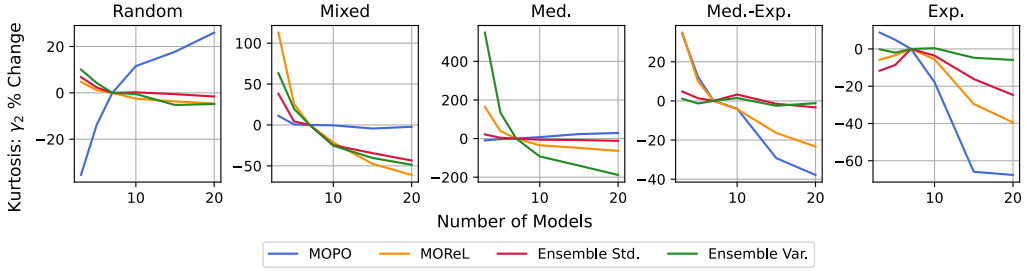



(b) Kurtosis Scaling

Figure 23: Hopper Transfer.

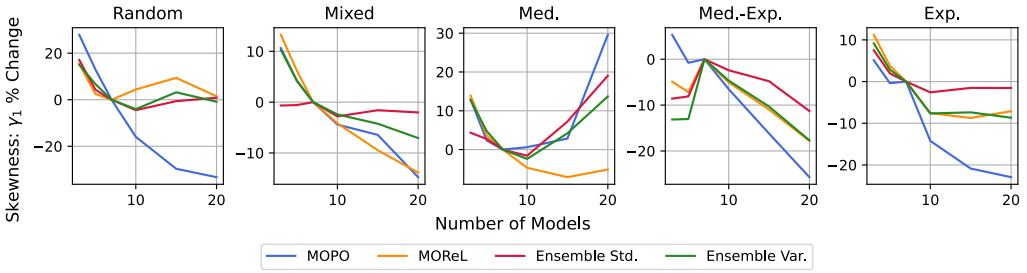



(a) Skewness Scaling

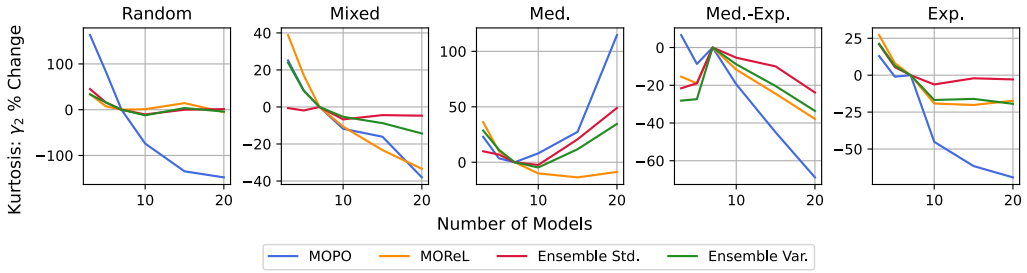



(b) Kurtosis Scaling

Figure 24: HalfCheetah Ground Truth.




(a) Skewness Scaling




(b) Kurtosis Scaling

Figure 25: Hopper Ground Truth.

# D Further details on Ground Truth experiments

## D.1 Methodological Details

We leverage the MuJoCo [45] simulator to provide us with ground truth dynamics that we can use to compare against our model predictions and penalties. This is done by providing the state and action inputs given to the model also into the simulator through the `set_state` method in the simulator API. It must be noted that this method also requires an addition 'displacement' value which is not modelled by the world models (nor is it provided in the D4RL data), however we found in practice this did not affect the dynamics predicted by the simulator, and simply setting this to 0 was sufficient to generate ground truth predictions.

This makes it possible to provide the simulator the hallucinated model states, and provide a true proxy to the *dynamics* discrepancy. We note that since the states are 'hallucinated' by the model, it might be the case that they may not be admissible under the true environment, but in reality the simulator was able to almost any combination of state and action, barring settings that featured anomalously large magnitudes; to handle such cases we found it necessary to clip the model states to the range $[-10, 10]$.

In order to assess the permissibility of states, as well as measure the accuracy of the penalties as OOD input detectors, we provide an alternative distance measure based on the distance away from the training set. We use this measure for our analysis in Section 5.3, and is calculated as the distance from the offline training dataset, which we define to be the 2-norm between a given state-action tuple and its nearest point in the offline data. We describe this quantity henceforth as 'Distribution Error'.

## D.2 On the nature of OOD data along hallucinated trajectories

Here we discuss the nature of OOD data along a single hallucinated trajectory (in the model) in offline MBRL, analyzing the inductive bias that some 'error' increases with increasing rollout length in the model. We find that there is merit to this assumption, and show this in Fig. 26 for all HalfCheetah and Hopper environments in D4RL. Here we plot the median error at each time-step across $30,000$ aggregated trajectories, and normalize them for comparison.

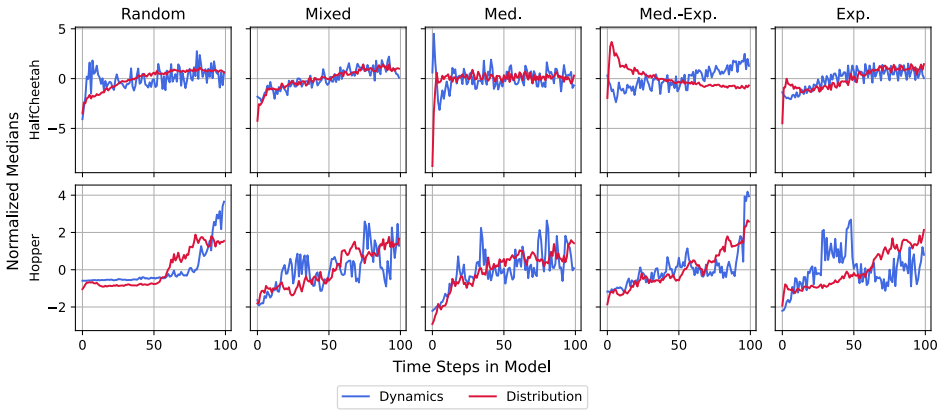



Figure 26: Median Errors across Rollouts of Ground Truths

We observe indeed that both median dynamics and distribution errors increase with increasing time step in the model. The only real exception is HalfCheetah Medium-Expert, which we believe to be due to our trained policy not being able to successfully exploit this environment.

The above analysis captures overall trends in the error over a large number of trajectories. However, the way errors manifest during an *individual rollout* is not so straightforward. To illustrate this, observe Fig. 27, where we plot a random subset of 5 individual rollouts from the Hopper Medium-Expert data we generated.

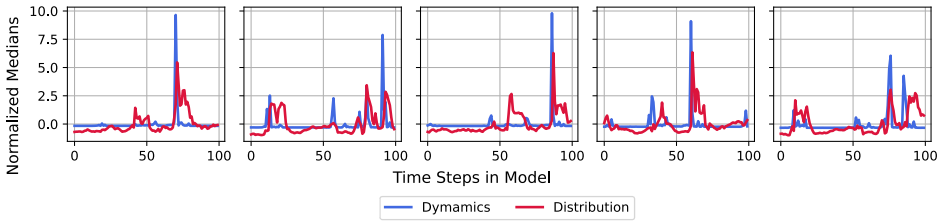



Figure 27: Several Individual Ground Truth Rollouts in Hopper Medium-Expert

We observe that errors along any single trajectory tend to manifest as 'spikes', and that it is entirely possible to recover from these, returning to either admissible dynamics, or parts of the state-action space that have been seen in the data. This speaks to the nature of how we ought to penalize policies for accessing regions of inaccuracy/uncertainty, and may justify a hybrid MOPO/MOReL approach, whereby we penalize individual transitions along a trajectory, but do not stop rollouts early. Indeed this is similar to the approach taken in M2AC (non-stop), albeit they choose to 'mask' uncertain transitions, not penalize them. We leave the design of such an algorithm to future work.

Finally, we address the issue of comparing OOD dynamics and inputs. As already observed in Fig. 27, these two errors are not necessarily always the same, and oftentimes it is possible that one quantity is large, whilst the other is small. We revisit Fig. 1 to explore this, now also plotting the Distribution Error in Fig. 28.

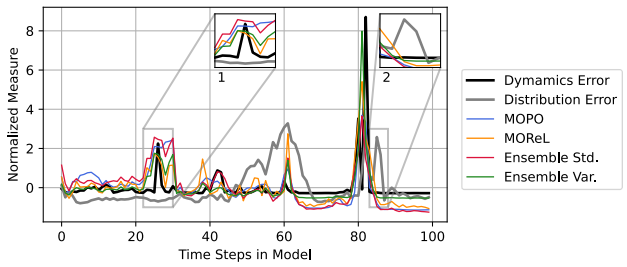



Figure 28: Comparing OOD dynamics and inputs on a Hopper Medium-Expert trajectory

We first speak to the inset annotated '1'. Here we observe that the transitions generated in fact closely resemble the data that our model was trained on, however the predicted dynamics are incorrect, and cause an aforementioned 'spike'. This is the opposite of what is observed in the inset annotated '2'; where we actually predict accurate dynamics, however the resultant state-action tuples do not closely resemble the data that our model was trained on. We generally observe that regions of high Distribution Error tend to be preceeded by 'spikes' pertaining to high Dynamics Error, and this present an exciting avenue for future work understanding how these quantities are related.

# E Using metrics as OOD event detectors

## E.1 Measuring Statistics

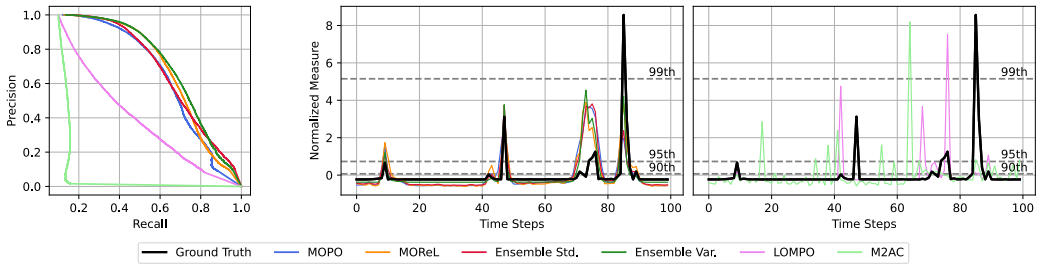
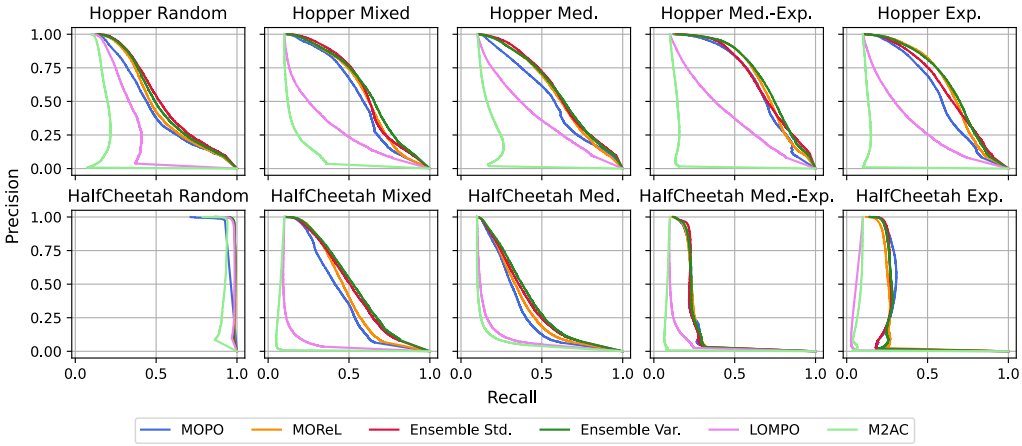



Figure 29: Hopper Medium-Expert Ground Truth Experiments; **Left:** Precision v.s. Recall against Ground Truth; **Middle:** Higher Performing Penalties v.s. Ground Truth MSE in Imagined Rollout; **Right:** Lower Performing Penalties v.s. Ground Truth MSE in Imagined Rollout

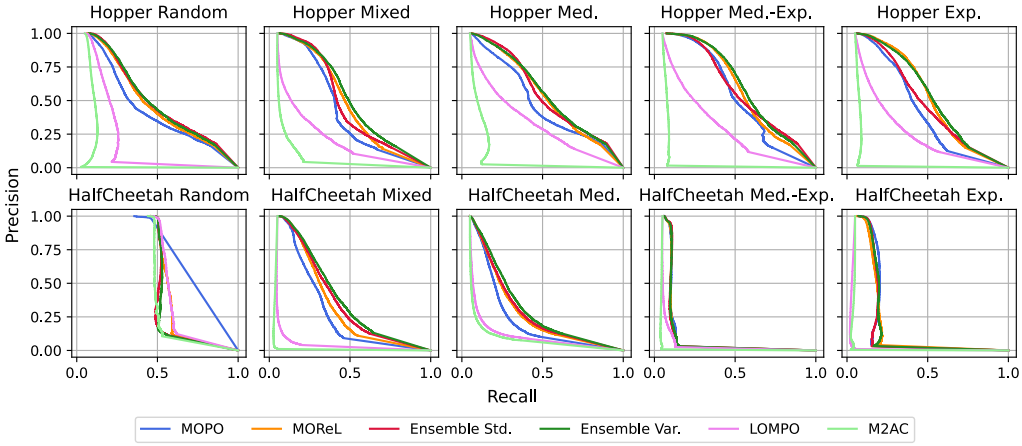
As noted previously, different penalties have varying scales and distribution profiles, so we need a way of standardizing the method of assessment. Using our observation that errors manifest as 'spikes' during a rollout, we propose treating each penalty as a classifier. Concretely, our test set consists of the ground truth data labeled by whether or not they exceed a certain percentile at a particular time step. Each penalty may be then be treated as a 'classifier' by normalizing its range to lie in $[0, 1]$. We can then use standard classification quality measures, such as AUC, to determine the effectiveness of these penalties at capturing these spikes, whilst sidestepping the issue of the different distributional profiles identified previously.

Fig. 29 shows how our proposed method may be used to compare the effectiveness of each metric at capturing OOD events. In the figure, we plot a single rollout in the model, and the resultant ground truth MSE between the predicted next state and the true next state in black. We then superimpose the 90th, 95th and 99th percentile MSEs across the entire imagined trajectories onto the figure in gray dashed lines. To construct our OOD labels, we label any point below the percentile line as being 'False', and any point above that line as being 'True'. Finally, we normalize the uncertainty metrics as previously described into values in the range $[0, 1]$, allowing us to construct precision-recall graphs and calculate classifier statistics.

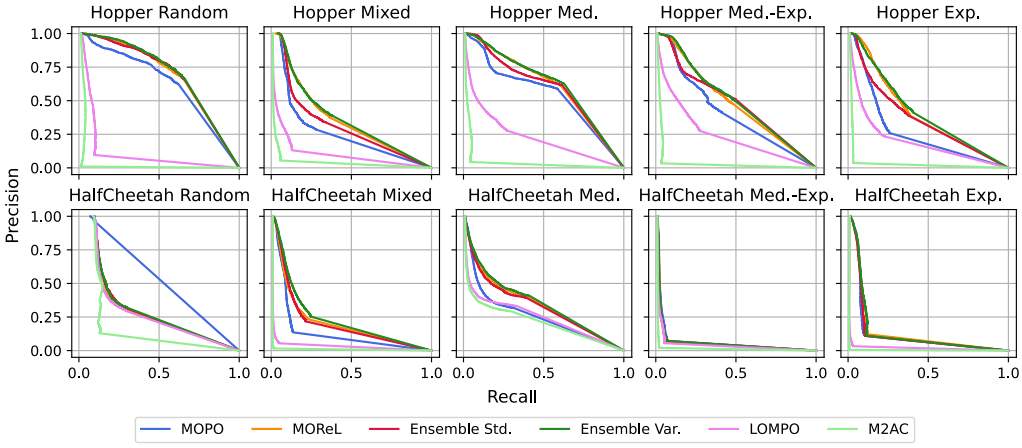
### E.2 Precision Recall Curves




(a) 90th Percentile




(b) 95th Percentile




(c) 99th Percentile

Figure 30: Precision Recall curves on ground truth data.

## F Key Differences between Code and Paper

Here we summarize key differences between the paper and code for the MOPO and MOReL algorithms which we compare against that are crucial to achieve the same reported performance.

In MOPO,

- Each layer in the model neural network has a different level of weight-decay
- The authors' code uses different objectives for training (log-likelihood) and validation (MSE).
- The authors use elites, but only for next state prediction (as discussed previously).

In MOReL,

- There is a difference in the authors' code about how the penalty threshold is calculated and tuned, and isn't provided as a hyperparameter in the appendix.
- The absorbing HALT state does not appear in the authors' code.
- The negative halt penalty appears significantly different between code and paper.
- There is a minimum trajectory steps parameter (hardcoded to 4) not mentioned in the paper.

# G Hyperparameters and Experiment Details

The D4RL [14] codebase and datasets used for the empirical evaluation is available under the CC BY 4.0 Licence.

The remaining hyperparameters for the MOPO algorithm that we do not vary by Bayesian Optimisation were taken from the original MOPO paper [50].

The hyperparameters used for the BO algorithm, CASMOPOLITAN, are listed in Table 6. We use the batch-mode of CASMOPOLITAN, where multiple hyperparameters settings are proposed and evaluated concurrently.

Table 6: CASMOPOLITAN Hyperparameters

| **Parameter** | **Value** |
|---|---|
| no. parallel trials | 4 |
| no. random initializing points | 20 |
| ARD | False |
| acquisition function | Thompson sampling |
| global BO | True |
| kernel | CoCaBo kernel [42] |

Each BO run on a D4RL environment took ~200 hours on a single NVIDIA GeForce GTX 1080 Ti GPU taken up predominantly by MOPO training.

Unless specified otherwise, plots and reported statistics are completed with 7 models in the ensemble, as this is the number chosen in the original MOPO paper used with the MOPO penalty.