The Impact of Task Underspecification in Evaluating Deep Reinforcement Learning - Appendix

1 Introduction

1.1 Task underspecification in common benchmark DRL tasks

In Table 1.1, we present some of the commonly used DRL benchmark tasks for evaluations and some ways they may be underspecified. We note that the ways in which a task is underspecified may depend on the application context, and thus the table is meant to be illustrative rather than exhaustive.

Task	Underspecification
Cartpole	• Cart mass, pole mass, pole length, and gravity.
Mountain car	• Heights of the mountains, the trigonometric curves defining the mountains, and gravity.
Lunar lander	• Landing polygon size, lander leg heights, widths, and gravity changes.
Acrobot	• Masses and lengths of the two links connected linearly to form a acrobot chain.
Pendulum	• Mass and length of the pendulum.
Swimmer	• Capsule sizes for the segments comprising the robot.
Walker2D	• Capsule sizes for the segments comprising the robot.
Breakout	• Paddle length and friction of the paddle surface.
Pong	• Paddle lengths.

2 Shortcomings of Point MDP-based Performance Evaluations

2.1 Example task configurations

Table 1 provides the MDP configurations that was used in Section **??** for elaborating on the potential shortcomings of point MDP-based evaluations. For each task, $\star$ denotes the generic MDP, and † represents the simplified MDP. The simplified MDP is usually designed to be what is supposed to be the easiest MDP or the one that simplifies the transition dynamics. The generic MDP is the parametric mean MDP of four other MDPs of the family. We verified that all the MDPs provided in Table 1 can be solved under the enforced actuator limits and that none of the settings are under-actuated.

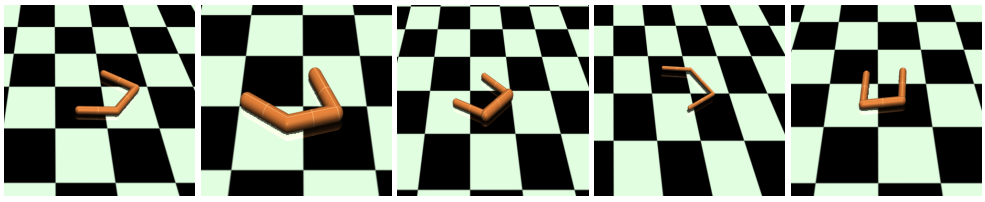
Table 1: Control task and the MDP families. For each task, $\star$ denotes the generic MDP and † represents the simplified MDP.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

Task	Task description	MDP family
Quad	• **Goal**: maneuver a 2D quadcopter through an obstacle course using its vertical acceleration as the control action. • **MDP family**: MDPs with varying the obstacle course lengths (upper obstacle length u and lower obstacle length l)	• (u=0.0, l=6.0)† • (u=0.5, l=3.0) • (u=0.625, l=4.0)* • (u=1.0, l=2.0) • (u=1.0, l=5.0)
Pendulum	• **Goal**: swing up the pendulum. • **MDP family**: MDPs with varying masses (m) and lengths (l) of the pendulum	• (m=1.0, l=1.0)† • (m=1.5, l=4.0) • (m=1.875, l=5.25)* • (m=2.0, l=6.0) • (m=3.0, l=10.0)
Swimmer	• MuJoCo 3-link swimming robot in a viscous fluid. • **Goal**: make the robot swim forward as fast as possible by actuating the two joints. • **MDP family**: MDPs with varying capsule sizes for the segments comprising the robot swimmer (a, b and c)	• MDP 1 =(a=0.10, b=0.10, c=0.10) • MDP 2 =(a=0.15, b=0.15, c=0.15) • MDP 3 =(a=0.10, b=0.15, c=0.10) • MDP 4 =(a=0.05, b=0.05, c=0.05)† • MDP 5 =(a=0.10, b=0.1125, c=0.10)*

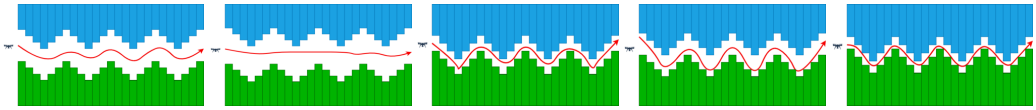
2.2 Visual illustrations of MDP families

In Figures 1, 2 and 3, we visually demonstrate the family of MDPs described in Table 1.



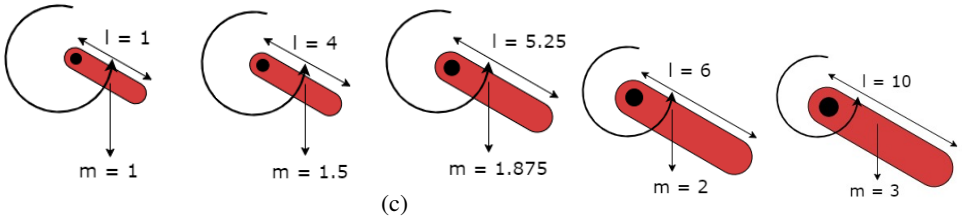
(a) MDP 1 (b) MDP 2 (c) MDP 3 (d) MDP 4 (e) MDP 5

Figure 1: Visual illustration of family of point MDPs used in the Swimmer task



(a) (u=1,l=5) (b) (u=0,l=6) (c) (u=0.5,l=3) (d) (u=0.625,l=4) (e) (u=1,l=2)

Figure 2: Visual illustration of family of point MDPs used in the Quad task

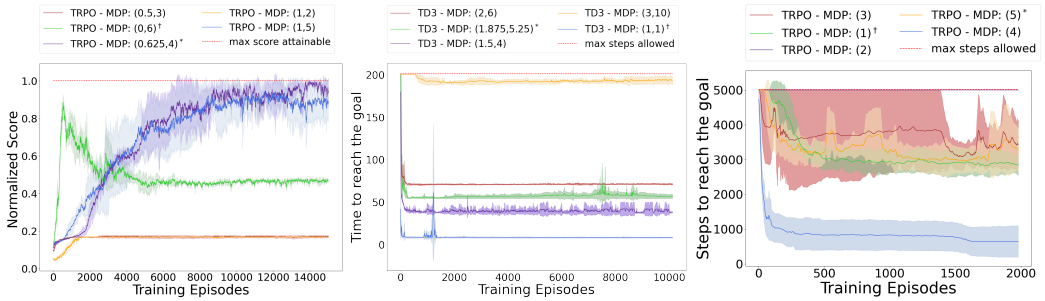


(a) (m=1,l=1) (b) (m=1.5,l=4) (c) (m=1.875,l=5.25) (d) (m=2,l=6) (e) (m=3,l=10)

Figure 3: Visual illustration of family of point MDPs used in the Pendulum task

2.3 Further examples on illustrating DRL training can be sensitive to the selected point MDP properties.

Figure 4 illustrates another example set of training curves for illustrating observation 2 of Section **??**. Here we use TRPO for training Quad and Swimmer, whereas TD3 was used for Pendulum. In summary, clearly, there are variations in the training depending on the choice of point MDP for all three control tasks as pointed out in Section **??**.



(a) Quad TRPO training (b) Pendulum TD3 training (c) Swimmer TRPO training

Figure 4: Training progress of each task for different point MDPs. For Quad, performance is a normalized score that indicates the distance the quadcopter traveled before crashing or reaching the goal with respect to the total distance required to travel (higher the better). In the pendulum, the training progress is measured as the time to swing the pendulum up (lower the better). Finally, for the swimmer, the training progress is measured as the time to reach the goal (lower the better). TRPO was used in training Quad and Swimmer, whereas TD3 was used for the Pendulum.

3 Case Study: Traffic Signal Control

3.1 RESCO traffic signal control benchmark

RESCO provides a standardized implementation of state-of-the-art DRL algorithms for traffic signal control that have become popular in recent years. It also provides non-RL baseline methods from the traffic engineering community.

We use six algorithms from RESCO in our case study, namely: (1) **IDQN**: a deep Q-learning approach, (2) **IPPO ?**: same as IDQN with a modified output layer, (3) **MPLight ?**: scalable FRAP model **?** approach using the pressure concept, (4) **MPLight***: MPLight implementation with the addition of sensing information, (5) **Fixed time**: a fine tuned non-RL pre-defined controller where phases are enabled for a fixed duration following a fixed cycle, (6) **Max pressure ?**: a non-RL controller where phase selection is based on the maximal joint pressure.

3.2 Intersection distribution

In this work, we use Salt Lake City intersection data for building the distribution due to its well-documented and advanced traffic network system. Our data for building the intersection distribution comes from a combination of open data sources. For most of the street network data which includes street geometry and layout, we use OpenStreetMaps (OSM) **?**. As for the traffic signal and demand data, we utilize data from the UDOT Open Data Portal **?** and the Automated Traffic Signal Performance Measures (ATSPM) **?**.

The first part of the data used consists of road networks obtained from OSM. We use the OSMnx Python package to manipulate the OSM data. We perform a data pre-processing that involves using mean substitution for NaN values (for example, the mean speed of 25 mph was used as a default) and the removal of motorways and motor links which are beyond the scope of this analysis. Next, we join the OSM data with UDOT Data Portal 2019 traffic demand data **?** by intersecting the two datasets based on the edge locations. This is a manual step done within ArcGIS Pro, which involves buffering the data to account for slight positional differences in the location of edges from the two datasets as well as calculating bearings to correct for any improperly intersected edges. The UDOT

Feature	Units	Mean	Standard Dev
Lane Count	-	3.8	1.37
Speed	mph	32.6	5.40
Length of Lanes	meters	260.8	193
Vehicle inflow	vehicles/hour	73.5	774
Left Turns Count	-	0.229	0.496
Right Turns Count	-	0.100	0.298

Table 2: Overview of the features that describe the intersection point MDP distribution

provides traffic demand data in terms of average annual daily traffic, a measure of the traffic volume of an entire year averaged over 365 days. We utilized this dataset to define the vehicle inflow rates (i.e., vehicles per hour) at the intersections. With that, we finally use six features to describe an intersection: number of lanes, maximum allowed speed, lane length, traffic inflow, number of left turning lanes, and number of right turning lanes. In Table 2, we summarize the mean and standard deviation of the selected six features.

Next, we filter this network to take a subset of intersections (and their adjacent streets) that correspond geospatially to signalized intersections in Salt Lake City from the Automated Traffic Signal Performance Measures. Finally, we build the full distribution of intersections based on this dataset. In our analysis, we use 345 intersections.

4 Shortcomings of Point MDP-based Evaluations in Common DRL Tasks

In this section, we provide an in-detail view of the shortcomings of point MDP-based evaluations by taking three popular control tasks as examples: cartpole (discrete actions), pendulum (continuous actions), and half cheetah (continuous actions). For each task, we devise a family of MDPs as described in Table 3 using CARL benchmark suite **?**. Our MDP families consist of 576, 180, and 120 point MDPs for cartpole, pendulum, and half cheetah, respectively.

4.1 MDP family configurations

Table 3: Popular DRL control tasks and the context features defining the MDP families

Task	Task description	Context features
Cartpole	• **Goal**: balance a pole attached to an unactuated joint of a cart by applying forces in the left and right direction on the cart.	• pole length (0.05, 0.5, 3, 5) • mass of the cart (0.1, 1, 6, 10) • mass of the pole (0.01, 0.1, 0.5, 1) • force magnifier (1, 50, 100) • gravity (0.1, 9.8, 19.6)
Pendulum	• **Goal**: apply torque on the free end of the pendulum to swing it into an upright position.	• mass of the pendulum (0.4, 1, 1.5, 2, 3, 4) • length of the pendulum (0.5, 1, 2, 4, 7, 10) • gravity (2, 5, 10, 12, 15)
Half-cheetah	• A 2-dimensional robot consisting of 9 links and 8 joints connecting them (including two paws). • **Goal**: apply a torque on the joints to make the cheetah run forward as fast as possible.	• gravity (-2, -5, -9.8, -12, -15) • torso mass (0.5, 2, 5, 9.457, 12, 15) • friction (0.1, 0.3, 0.6, 1.0)

In creating the MDP families, we generate point MDPs by generating all combinations of context features as specified in Table 3.

4.2 Analysis of performance scores

In Figures 5, 6 and 7, we illustrate performance score variations in cartpole, pendulum and half-cheetah tasks in addition to what we illustrated in Section **??**. The reported performance of each task is measured by training and evaluating DRL methods on commonly used single point-MDP given in the benchmark suites (OpenAI gym for cartpole and pendulum and Brax for half cheetah). For measuring overall performance, we use uniform point MDP distribution (each point MDP is equally important).

Figure 5 provided details insights into the performance of DRL methods in the cartpole task. As can be seen, all four DRL methods report a performance of 1.0, indicating optimally solving the cartpole task. However, when evaluated on the family, we see significant discrepancies. First, not all methods are equally well performing as reported. Specifically, methods like ARS and DQN underperform significantly compared to methods like PPO or TRPO. Second, all methods struggle to solve nearly 17% of point MDPs, while PPO and TRPO have the highest success in optimally solving point MDPs (61% and 56%, respectively). This calls for a significant method ranking change. Although reported results would rank all methods as equally well-performing, our analysis indicates that the ranking of methods is PPO, TRPO, DQN, and then ARS.

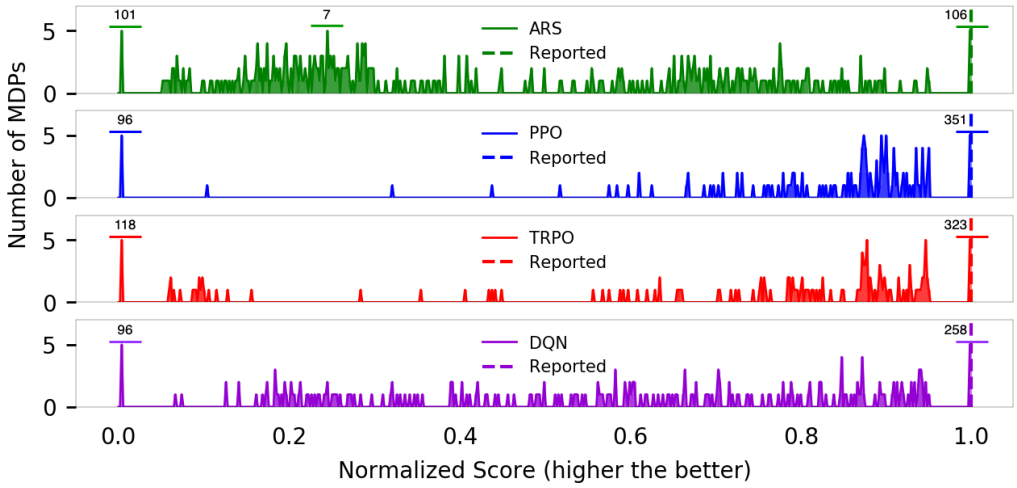


Figure 5: Performance vs. the number of point MDPs that demonstrate the performance using four popular DRL algorithms in Cartpole. 576 unique point MDPs were considered for each method. For better visibility, we limit the height of the y-axis to five point MDPs. If there are more than five point MDPs for a given performance score, we indicate the total number on the corresponding vertical plot line.

Similar insights can be seen in Figure 6 for the pendulum task. The performance of TRPO and SAC are clearly overestimated under reported performances. Also, there is a significant variation in performance under all four DRL methods. Although one would tend to pick TRPO as the best-performing method under reported performance, our analysis shows that SAC is, in fact, the highest-performing method. (Section **??** Figure **??**). This again calls for a ranking change.

Figure 7 shows similar results for half cheetah task. However, we see slightly less variation in performances across MDPs. We believe if more contextual features are used for describing point-MDPs, we would see more variations, just as seen in other tasks. In terms of ranking of DRL methods, although in this case, the first and second performing methods (SAC and TRPO) stay the same, third and fourth places are swapped. According to the reported performances, PPO outperforms TD3, but we show that TD3 is in fact better than PPO when overall performance is used for ranking. Additionally, although TRPO's rank stays the same, we see its performance has degraded

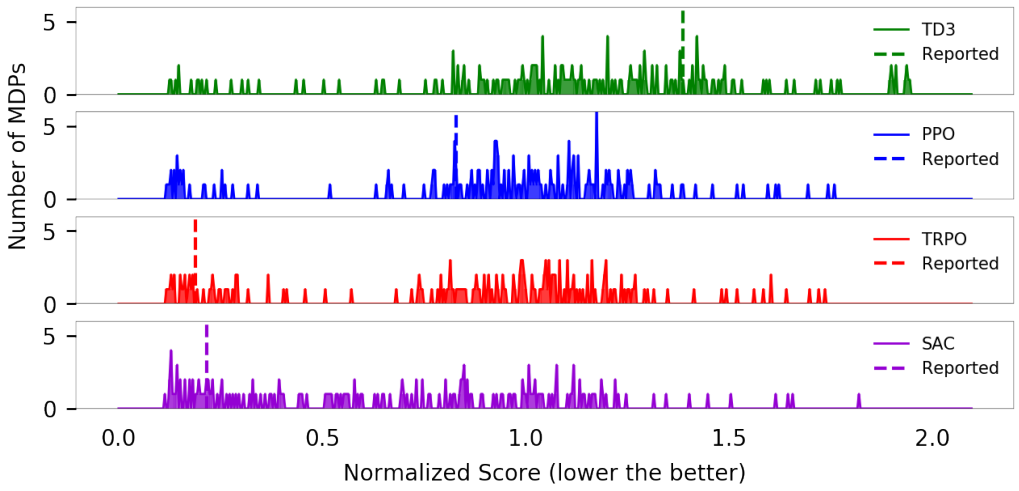


Figure 6: Performance vs. the number of point MDPs that demonstrate the performance using four popular DRL algorithms in the Pendulum. 180 unique point MDPs were considered for each method.

by a significant margin. These phenomena again highlight the shortcomings of point MDP-based evaluations when used in evaluating algorithmic generalization within a task.

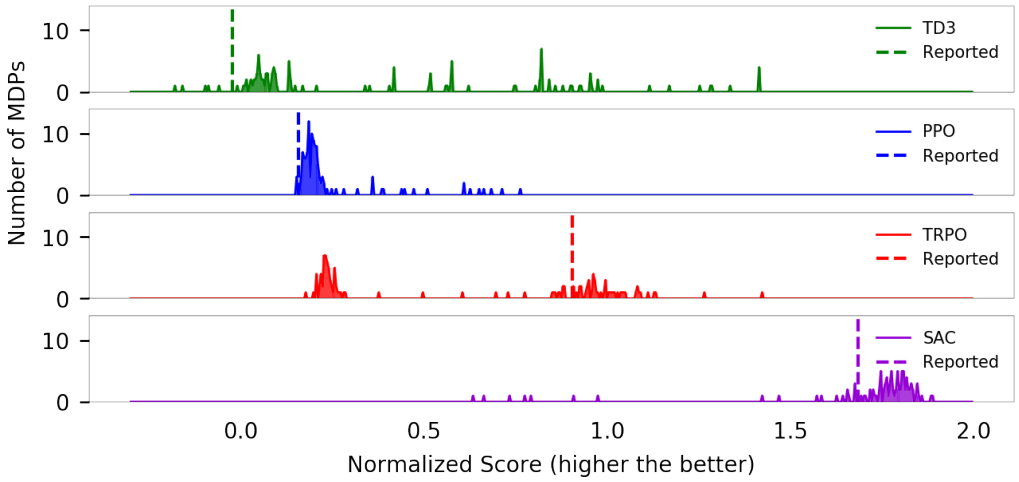


Figure 7: Performance vs. the number of point MDPs that demonstrate the performance using four popular DRL algorithms in Half Cheetah. 120 unique point MDPs were considered for each method.

5 Reliable Evaluations Within a Task

5.1 Evaluate DRL methods on a family of MDPs instead of point MDPs

Further details on the approximation algorithms used for performance approximations are presented below.

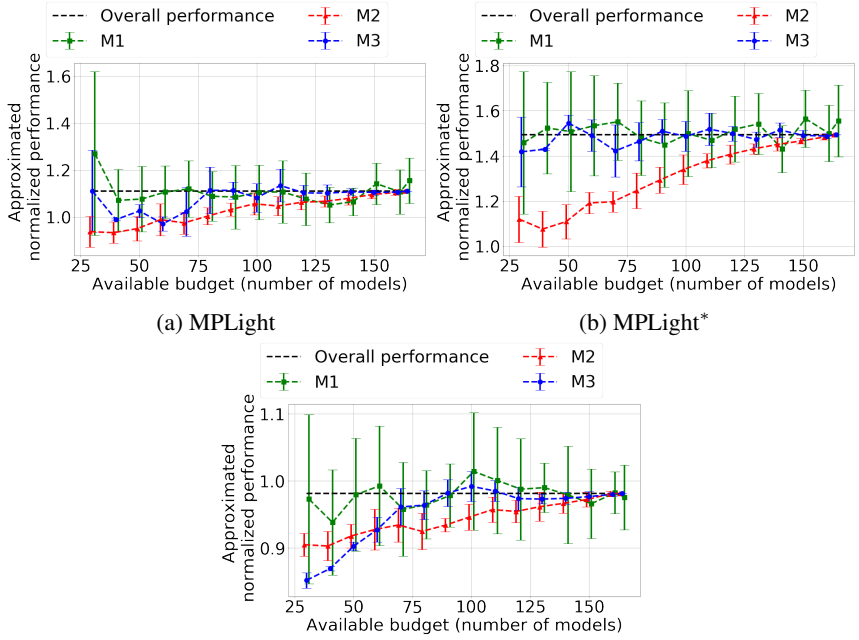
M1: random sampling with replacements: Under this method, we sample point MDPs from the distribution randomly with replacements. We sample exactly the same number of point MDPs as the maximum budget. One can decide to sample N/n point MDPs where N is the budget, and n is the number of parallel runs used for the evaluation of each point MDP. In this work we set $n = 1$.

After sampling and evaluations on each point MDP, the overall performance is defined as $\sum_{i \in X} s_{R,i}$ where X denotes the index set of point MDPs that were sampled.

M2: random sampling without replacements: This method is similar to M1 except that we do not perform replacements. The overall performance is defined as $\sum_{i \in X} q_{\hat{T},i} s_{R,i}$ where $q_{\hat{T},i} = \frac{q_{T,i}}{\sum_{i \in X} q_{T,i}}$ and $q_{T,i}$ is the probability of point MDP i according to the distribution. X denotes the index set of point MDPs that were sampled.

M3: clustering point MDPs using k-means: In this method, we cluster the point MDPs with k-means clustering giving each point MDP an equal weight. Next, we assign the sum of probabilities of all point MDPs that belong to the same cluster to its centroid. However, a point MDP that is represented by a centroid may not actually exist in the problem domain. Therefore, we match each centroid to the nearest actual point MDP from the family and conduct evaluations. The overall performance is then defined as $\sum_{i \in X} q^*_{T,i} s_{R,i}$ where $q^*_{T,i}$ is the total probability assigned to each point MDP selected to represent a cluster centroid. X denotes the index set of point MDPs that were selected based on centroids.

In Figure 8, we show the approximate evaluations of MPLight, MPLight* and Max Pressure methods in traffic signal control in comparison to the ground-truth performance while varying the budget size. Other related plots are given in Figure **??**.



(a) MPLight

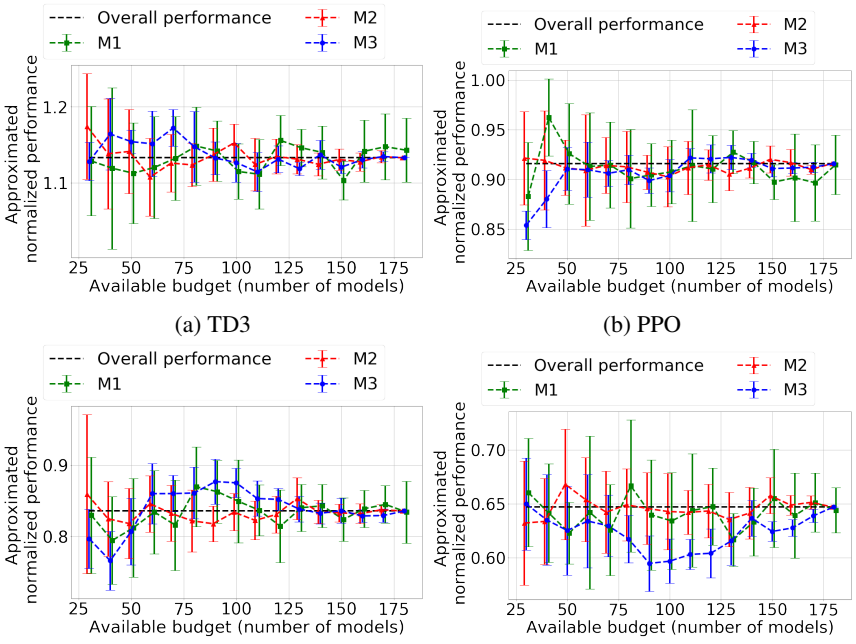
(b) MPLight*

(c) Max Pressure

Figure 8: Evaluations over a family of MDPs in traffic signal control with varying budget limits and sampling techniques. Closer to the overall performance the better.

Figure 9, Figure 10 and Figure 11 demonstrate the results of performance approximations in pendulum, cartpole, and half cheetah tasks, respectively. In all three control tasks, we see that the random sampling with replacements method has a high variance overall (M1). In comparison, the random sampling without replacements method (M2) and k-means clustering-based method (M3) produce more accurate performance approximations. Unlike in the traffic signal control case study, where the random sampling without replacements method had high variation and significant inaccuracies in approximated performance, here in the three control tasks, we see it demonstrates better performance approximations. K-means clustering-based method consistently produces good approximated performances. We note that even though the random sampling without replacements method produces on average good approximated performances, k-mean based clustering method has comparatively low variance and therefore is preferred for most of the cases (except for a few cases where it seems

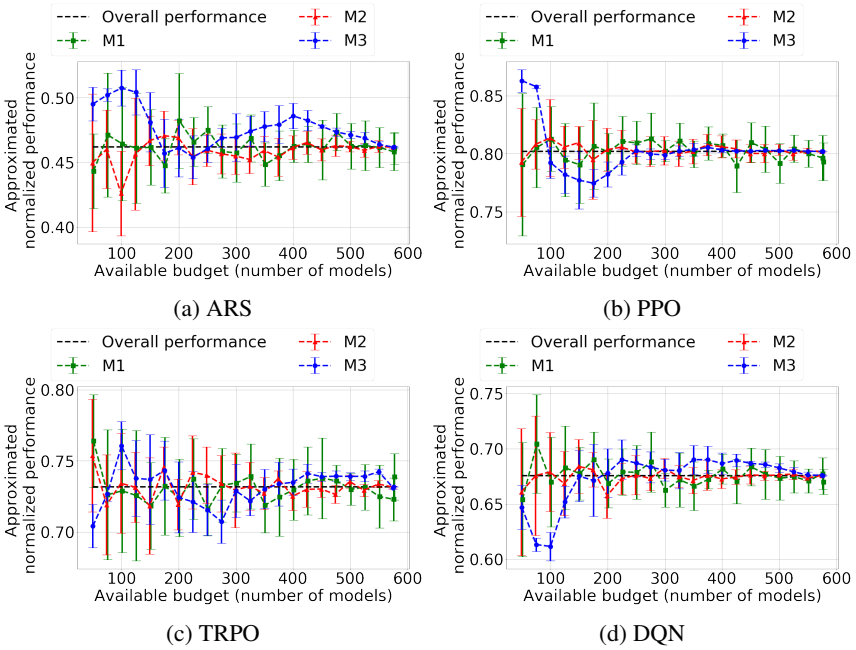
to have a slightly high error in approximations). Therefore, in general, we recommend using the k-means-based clustering method as it consistently gives us better performance approximations.



(a) TD3 (b) PPO

(c) TRPO (d) SAC

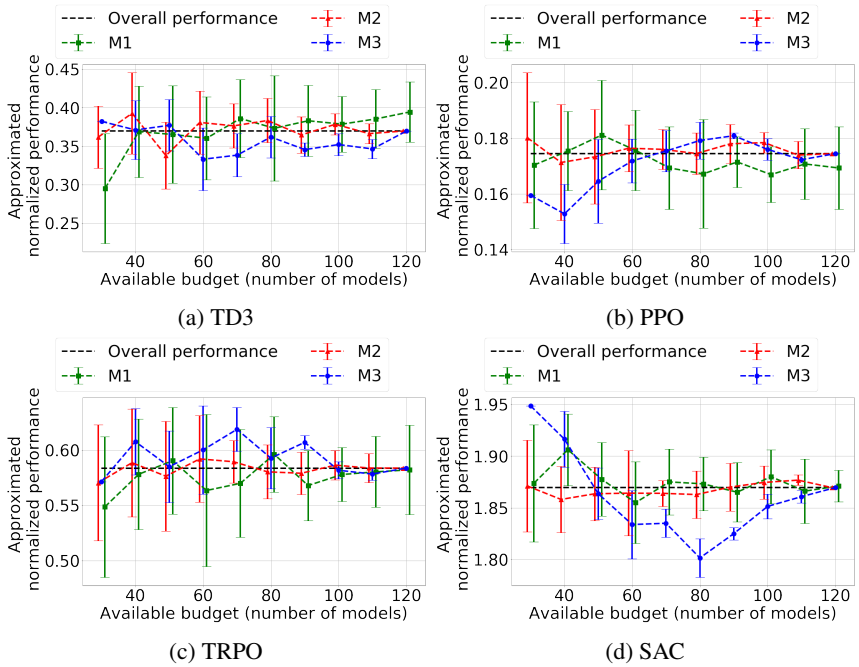
Figure 9: Evaluations over a family of MDPs in pendulum task with varying budget limits and sampling techniques. Closer to the overall performance the better.



(a) ARS (b) PPO

(c) TRPO (d) DQN

Figure 10: Evaluations over a family of MDPs in cartpole task with varying budget limits and sampling techniques. Closer to the overall performance the better.



(a) TD3

(b) PPO

(c) TRPO

(d) SAC

Figure 11: Evaluations over a family of MDPs in half cheetah task with varying budget limits and sampling techniques. Closer to the overall performance the better.

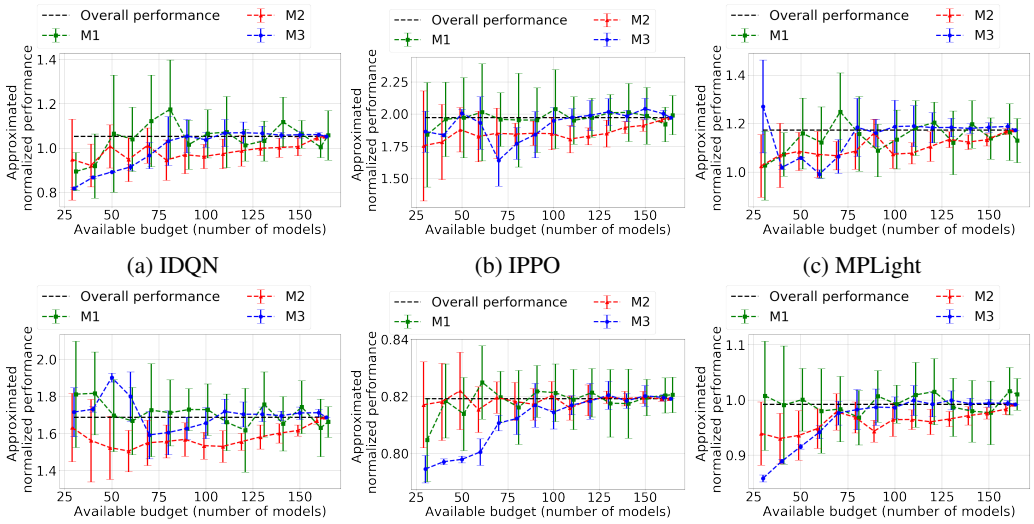
5.2 Further experiments on approximation algorithms

In this section, we perform a sensitivity analysis of the proposed approximation algorithms in Section **??**. Since the overall evaluations are subject to task failures, we investigate the sensitivity of the approximate evaluation methods to the prevalence of "failure" point MDPs. In particular, we look at the approximation accuracy as we change the underlying point MDP distribution for the same set of point MDPs used in the traffic signal control case study. We define a failure as any MDP with an average per-vehicle waiting time greater than the 20s. We conduct three sets of experiments. First, we look at the case where the failure MDPs are given low probabilities (compared to the rest of the point MDPs) in the point MDP distribution. Second, we look at the case where the failure MDPs are given high probabilities. Finally, to consider a fairly different distribution, we consider the case where the prevalence of each point MDP is uniformly random.

As a reminder, our notation for the three performance approximation techniques as introduced in Section **??** are (1) **M1**: random sampling with replacements from the point MDP distribution, (2) **M2**: random sampling without replacements, and (3) **M3**: clustering point MDPs using k-means and assigning probability mass of all point MDPs that belong to same cluster to its centroid. Here, k defines the number of point MDPs used for approximate evaluations. Further details of each method can be found in Appendix 5.1.

5.2.1 Case 1: Failure point MDPs get low probabilities.

Figure 12 denotes the performance approximations for each of the three methods with varying budget sizes when the failure point MDPs get low probabilities. It is clear from the figure that the k-means-based clustering approach performs the best in this case, specifically when the budget size is greater than half the size of total point MDPs. Random sampling with replacements has an acceptable mean performance but with a high variance. Finally, random sampling without replacements often overestimates the performance unless the budget size is closer to the total point MDPs count (note: lower the normalize score the better).



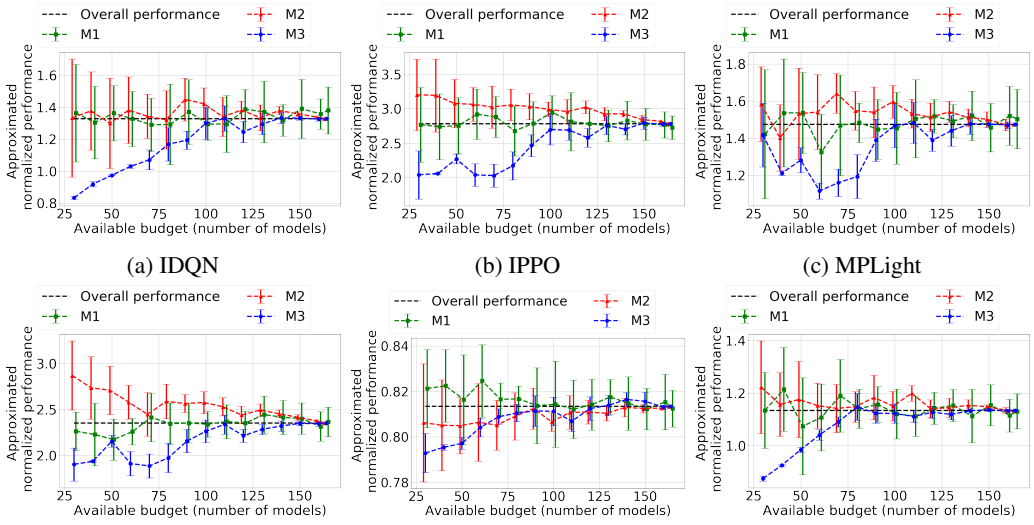
(a) IDQN (b) IPPO (c) MPLight

(d) MPLight* (e) Fixed Time (f) Max Pressure

Figure 12: Evaluations over a family of MDPs in traffic signal control with varying budget limits and sampling techniques when the failure point MDPs are assigned low probabilities than other point MDPs. Closer to the overall performance the better.

5.2.2 Case 2: Failure MDPs get high probabilities.

As the second case, we look at the case where failure MDPs get high probabilities than the rest of the point MDPs in Figure 13. While k-means-based clustering may underestimate the performance if the budget size is small, it still is the best method when the budget size is greater than half the total MDP count. Random sampling with the replacement method seems to work quite well in this case. Specifically, when the budget size is small, it performs better than random sampling without the replacement method, which always seems to underestimate the performance (note: lower the normalize score the better).



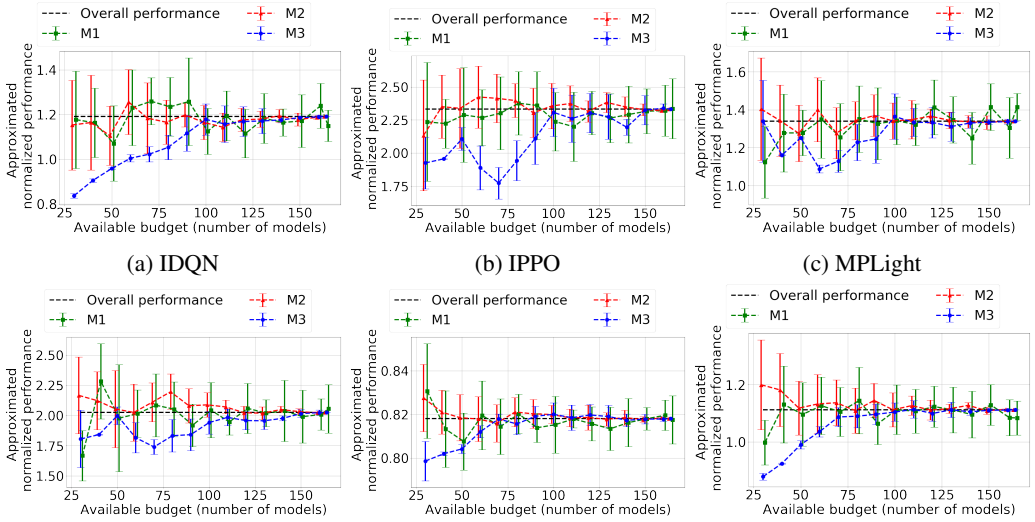
(a) IDQN (b) IPPO (c) MPLight

(d) MPLight* (e) Fixed Time (f) Max Pressure

Figure 13: Evaluations over a family of MDPs in traffic signal control with varying budget limits and sampling techniques when the failure point MDPs are assigned a high probabilities than other point MDPs. Closer to the overall performance the better.

5.2.3 Case 3: Point MDP prevalence is uniformly random.

In Figure 14, we denote the performance estimates when the prevalence of each point MDP is uniformly random. Here we observe an interesting result. The random sampling without replacements method performs quite well under this setting, yielding low variance. As in the other cases, the k-means-based clustering method performs well when the budget size is greater than half the total point MDP count. The random sampling with replacements method performs as usual but has a high variance.



(a) IDQN (b) IPPO (c) MPLight

(d) MPLight* (e) Fixed Time (f) Max Pressure

Figure 14: Evaluations over a family of MDPs in traffic signal control with varying budget limits and sampling techniques when the support of the point MDP distribution is a discrete normal distribution. Closer to the overall performance the better.

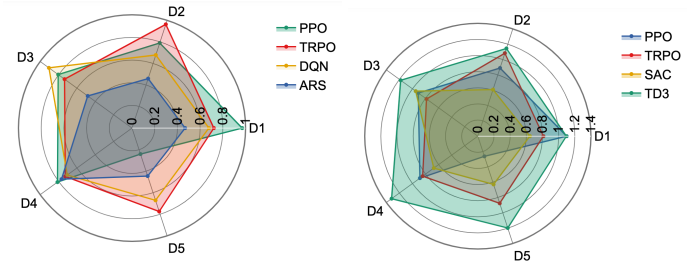
Overall, we see that the k-means-based clustering method (M3) performs consistently across three cases when the budget size is at least half the total MDP count. Random sampling with replacements (M1) method in expectation provides a reasonable performance estimate but may have a high variance. Random sampling without replacements method (M2) only performs well when all point MDPs are given a fair chance of getting any probability. It is clear that failure MDPs have an overall impact on which method suits best for a given use case. However, there is not enough information to know which point MDPs are failure MDPs as a prior because it poses a chicken-and-egg problem, as we pointed out in Section **??**. Therefore, in general, the k-means clustering-based method (M3) is recommended. However, k-means clustering can be inefficient when the point MDP distribution has a higher number of dimensions (since k-means is not robust to high dimensional data). If that is the case, we recommend using random sampling with replacements (M1). However, domain experts may have insights into which point MDPs are more susceptible to failure (e.g., in traffic signal control, intersections with short approaching lanes and high vehicle inflows can be challenging). If such domain knowledge can be incorporated into the process and thereby can be confident that point MDP distribution does not contain many failure point MDPs, the experiment designers may utilize the random sampling without replacements (M2) method since it performs better under such setting.

6 Impact of Point MDP Distribution on Evaluations

The choice of point MDP distribution can have a significant impact on the overall performance of an algorithm and, subsequently, on the ranking of a set of algorithms evaluated on the same task. Point MDP distribution essentially acts as a calibration distribution for evaluations according to some pre-defined criteria. For example, traffic engineers in New York may use a different point MDP distribution in comparison to what Salt Lake City engineers would use in traffic signal control benchmarking as they have different requirements. To demonstrate the impact of the point MDP distribution on overall evaluations, we perform a family of MDP-based performance evaluations

using multiple point MDP distributions on cartpole and pendulum tasks in CARL benchmark suite **?**. For cartpole, we create a family of MDPs that contain 576 point MDPs, and for the pendulum, we create a family of MDPs with 180 point MDPs. For illustration purposes, we randomly generate five point MDP distributions $D_1 \ldots D_5$ and plot the overall performance of each algorithm under each distribution in Figure 15.

From Figure 15, it is clear that under both tasks, depending on the choice of underlying point-MDP distribution, the ranking of methods can significantly change. For example, in cartpole, under $D1$ distribution, PPO performs the best among the other methods, but under $D5$ distribution, PPO performs the worst, and TRPO performs way better than PPO. Similar results can be seen in the Pendulum task as well. In conclusion, it is evident that the choice of the point MDP distribution plays a major role in point MDP family-based evaluations and that it should be carefully set to fit the pre-defined requirements.



(a) Cartpole (higher the scores the better)

(b) Pendulum (lower the scores the better)

Figure 15: Evaluations of cartpole and pendulum over a family of MDPs under different point-MDP distributions. $D_1 \ldots D_5$ denotes the five random distributions and scores on the plots are normalized performance scores.

7 Do Point MDP-based Evaluations Affect Standard DRL Evaluations Across Tasks?

In this work, our primary focus is identifying and pointing out shortcomings of point MDPs-based evaluations when DRL methods are evaluated for algorithmic generalization within a task. However, we hypothesize such shortcomings due to point MDPs are not just isolated to this case but also can happen when DRL methods are evaluated for algorithmic generalization across tasks. The standard practice of evaluating general DRL methods is to consider a suite of tasks (one point MDP from each task) and evaluate the methods on all of them to report the robust performance of the method. In this section, we conduct experiments to provide insights that suggest that, even in this case, the choice of point MDPs can have an overall impact on how one would rank DRL methods based on this evaluation protocol. Therefore, we call for future work to further analyze this phenomenon.

We used pendulum, half cheetah, and mountain car continuous control as a suite of tasks. We use the same point MDP families described in Table 3 for pendulum and half cheetah and generate the MDP family for mountain car by varying max speed, goal position, and power. We use four DRL methods: PPO, TRPO, SAC, and TD3. To mimic the standard evaluation procedure of evaluating DRL methods on one point MDP selected from each task, we generate all combinations of one point MDP from each task. Then we calculate the overall performance score for each DRL method evaluated on each point MDP combination. In Table 4, we report the percentage of the times each DRL algorithm obtained each rank.

In Table 4, we see some interesting results. Despite what would otherwise conclude based on default point MDPs currently used in benchmark suits for the pendulum, half cheetah, and mountain car, we observe that the choice of the point MDP from each task matters a lot. In fact, different choices can lead to completely different ranking orders. We show that depending on which point MDP is used from each task, PPO is ranked first for 19% times, TRPO for 26%, SAC for 43%, and TD3 for 12%. For comparison, algorithms are ranked SAC, TRPO, PPO, and TD3 when trained on the default point MDPs. These findings provide insightful evidence that considering point MDP families may even

Method	Rank 1	Rank 2	Rank 3	Rank 4
PPO	18.75%	29.54%	27.74%	23.97%
TRPO	26.23%	24.99%	30.45%	18.33%
SAC	42.56%	29.09%	18.50%	9.85%
TD3	12.46%	16.38%	23.31%	47.85%
Reported	SAC	TRPO	PPO	TD3

Table 4: Percentage of the times each DRL algorithm obtained each rank

benefit the evaluation practices for DRL across tasks because doing so may inform better selection of "default" point MDPs. We, therefore, call for future work in this direction to further analyze the phenomena in detail.