Pareto-Efficient Multi-Objective Deep Reinforcement Learning For Critical Care Applications

Ali Shirali, Alexander Schubert

Abstract

Given the sequential nature of most medical treatment decisions within the Intensive Care Unit (ICU), there is substantial hope that reinforcement learning will be able to estimate precise data-driven treatment plans. However, a key challenge for most applications in this space is the sparse nature of the - primarily mortality-based - reward functions, leading to the decreased stability of off-policy estimates. In this work, we introduce a new Pareto-efficient multi-objective reinforcement learning approach to enable effective training in an environment where our true reward signal of interest (e.g., mortality) is delayed, rare, and costly, and where available reward proxies are frequent but inaccurate. Our method intends to simplify learning under sparse rewards, by first pruning the action space based on a Pareto-optimality criterion, and only afterwards training a final model on the newly reduced action space. Our empirical results indicate that pruning significantly reduces the size of the action space and enables to derive meaningful reinforcement learning policies from sparse reward data. The learned policies and Pareto-optimal action spaces could be used to detect high-risk patient trajectories and to derive personalized medical treatment plans.

1 Introduction

In recent years, there has been increased interest in exploring the prospects of reinforcement learning (RL) for applications in the critical care setting [27, 4, 25, 15, 29, 28, 7, 9, 17]. Given the sequential nature of most medical treatment decisions within the Intensive Care Unit (ICU) as well as the ability of RL methods to learn via trial and error without needing access to optimal ground truth decisions, there is hope that this technology will allow to uncover personalized data-driven treatment policies [15]. However, a key challenge in most of these applications is that the key outcome of interest for reward assignment is sparse and thus increases the variance of off-policy estimates while reducing the stability of trained algorithms. In addition, when intermediate reward proxies exist, it is often unclear how these multiple desirable factors should be combined with each other in order to facilitate effective learning.

For instance, focusing on the single task that is the management of vasopressor and intravenous fluids in sepsis patients, popular work of Komorowski et al. [15] solely rewards the agent based on 90-day survival, which only provides sparse signals at the final step of each trajectory. Alternatively, there exist intermediate rewards based on sepsis-related medical risk scores, such as the SOFA-score ([38]), and arterial lactate levels measured via lab tests ([29]), which, however, only provide a crude approximation of individualized sepsis mortality risk.

In order to overcome these challenges, we propose a new algorithm that leverages a pareto-efficient multi-objective reinforcement learning framework (MORL) to enable effective training in an environment where the true reward signal of interest (e.g., mortality) is delayed, rare, and costly, and available proxies are frequent but only provide rough indicators for ultimate patient outcomes (e.g., medical risk scores).

Different from common applications of MORL [41, 24, 1] our main focus does not lie on the inference of optimal preference weights, but instead on identifying state-action pairs that are dominated regardless of the choice of preference weight between different intermediate rewards. By removing these dominated actions, we simplify the learning task to subsequently derive a final policy based on the sparse mortality reward. Our algorithm implements this idea by first training an ensemble of deep Q-learning (DQN) [22] or conservative Q-learning (CQL) [16] models using only rewards that are available for multiple timesteps during the trajectory. Afterwards, we apply a pareto-criterion to remove actions that are dominated based

on the Q-values assigned by models trained in stage 1. Then, we train a double deep Q-network or a conservative Q-network based on the sparse rewards but restricted to the pruned action sets.

We first evaluate our framework in an off-policy learning setting on the OpenAI Gym LunarLander environment [3]. Here we show that our algorithms can effectively prune the action space and that this reduction in complexity significantly improves learning based on sparse reward signals compared to a deep Q-learning (DQN) baseline model. Next, we test our framework in an offline setting with real health records of septic patients in the intensive care unit. Our results indicate effective pruning of the action space, where our pruning approach enables us to significantly reduce the number of available state-action pairs without dropping the actions chosen by physicians in the majority of cases. Further, flagging dominated actions we observe a significantly increased mortality rate for trajectories that majorly consist of dominated actions. The same pattern continues when evaluating the Q-function learned based on the pruned-action space, where lower Q-values relate to a higher mortality risk and the Q-value distribution for patients who died is centered around a lower mean than for surviving patients. These results make us confident that our approach uncovers a meaningful Q-function in the offline learning setting despite only being trained based on a sparse reward.

Beyond the healthcare setting, our framework may be extended to most domains where frequent accurate reward signals are not available, or there exists ambiguity about the weights that should be assigned to different intermediate reward components. For instance, our experimental results indicate that our method is especially promising in off-policy learning environments, outperforming policies that are trained based on a sparse reward alone. The code for this project is publicly available[1].

2 Related Work

Multi-objective reinforcement learning: The main body of work on MORL can be divided into single-policy methods and multiple policy methods [18, 30, 35]. Single-policy methods assume known preferences and aim to estimate an optimal policy given these preferences among different objectives [34, 20]. The key focus of these methods lies in recovering an accurate and tractable mathematical representation of the preferences. These methods, however, cannot be used when the preferences themselves are unknown.

Instead, multi-policy methods aim to estimate the Pareto frontier based on a set of different policies. One way to achieve this is by training an ensemble of policies based on different preference specifications [23, 37, 24, 40]. Several of these methods employ vectorized versions of RL algorithms and use evolutionary or incrementally updated algorithms to push the ensemble towards the Pareto front [11, 30]. Similarly, work on MORL using value-based algorithms proposed extensions to the Bellman equation in order to preserve the convex hull of the Pareto frontier [2, 10, 12]. While most of these studies have focused on the online setting, initial work has also proposed approaches to extend MORL to the tabular offline RL setting [32, 39].

Overall, a key focus of this area of research lies around either inferring optimal preference weights or directly uncovering a set of Pareto-optimal policies. Instead, to the best of our knowledge, this study is the first to leverage MORL to prune the action space by identifying and eliminating dominated state-actions pairs in order to reduce the complexity of the learning space for subsequent learning tasks.

Reinforcement learning in health: In recent years, reinforcement learning has been the focus of several studies in healthcare [19], with a particular emphasis on applying it to infer personalized treatment plans for sepsis patients in the ICU [15, 9, 8, 29, 31, 26, 33, 28]. However, these studies either leverage sparse mortality information only to assign rewards or combine multiple intermediate sepsis risk proxies based on a deterministic weight. To the best of our knowledge, we are the first to explicitly address the challenges imposed by sparse rewards by using a Pareto-efficient multi-objective reinforcement learning approach to prune the action space before estimating a final policy.

Dead-end state identification: Given the focus of the pruning stage of our algorithm on eliminating dominated actions, our method is related to work on dead end's in reinforcement learning [6, 7, 13]. For instance, Fatemi et al. [7] develop an RL approach to identify and avoid "medical dead-end" states, defined as states from which no action can be made to achieve a positive terminal outcome (e.g. survival). While the aim of this method is conceptually close to the goal of our pruning stage, we propose a distinct technical approach to achieve this goal. Fatemi et al. [7]'s work focuses on classifying states that should be avoided;

[1] `https://github.com/alishiraliGit/multi-criteria-dqn`

in contrast, our method directly identifies and excludes dominated state-action pairs. Furthermore, our method does not stop at the identification of risky actions but instead leverages the pruned action space for the subsequent training of an optimal policy. The details of our approach are described in the next section.

3 Multi Q-learning

Our algorithms have two phases. In the first phase, we train an ensemble of Q-networks leveraging only (crude) intermediate reward signals. In the second phase, we will use the Q-values predicted by these networks to remove dominated actions and train a new Q-network based on the sparse rewards i.e. the terminal reward. In the following, we will explain the details of the MultiDQN algorithm for each phase.

3.1 Phase 1

In phase 1 of MultiDQN, we train deep Q-networks [22] using double Q-learning [36]. Given a replay buffer $\mathcal{B} = \{(s_t, a_t, s_{t+1}, \boldsymbol{r}_t)\}$ with vector rewards in $\mathbb{R}^D$, our algorithms follow the steps listed below:

1. Sample a transition $(s, a, s', \boldsymbol{r}) \in \mathcal{B}$.
2. Estimate target values at s:
   (a) Find values at s' (based on a target network Q').
   (b) Do Bellman-style backup.
3. Update the network Q by comparing its values at s to the estimated target values at s'.
4. Occasionally, update Q' by Q.

Given that the weighting of the intermediate rewards, $\boldsymbol{w}$, is drawn from a prior distribution $\mathcal{P}$, we train independent Q-networks on various realizations of $\boldsymbol{w}$. We then gather all the predictions of these networks for phase 2 and prune the action set.

Let us say we want to obtain n *intermediate* Q-networks. For each of them, we draw $\boldsymbol{w}_i \sim \mathcal{P}$ and train a classic double DQN with the scalar reward specified by $r = \boldsymbol{w}_i^{\mathbb{T}} \boldsymbol{r}$. Let $Q_{\boldsymbol{w}_i}$ be the resulting Q-network. Then we define a new vector Q-network by:

$$\boldsymbol{Q}(s, a) = [Q_{\boldsymbol{w}_1}(s, a), Q_{\boldsymbol{w}_2}(s, a), \cdots, Q_{\boldsymbol{w}_n}(s, a)]^{\mathbb{T}}. \tag{1}$$

3.2 Phase 2

Given the Q-networks from the first phase, we will now prune the action set by removing dominated actions.

Let $\boldsymbol{Q}(s, a) \in \mathbb{R}^n$, where n is the number of intermediate Q-networks. Here we use a notion of Pareto optimality to remove weak actions. The standard notion of Pareto optimality keeps actions under the condition defined below.

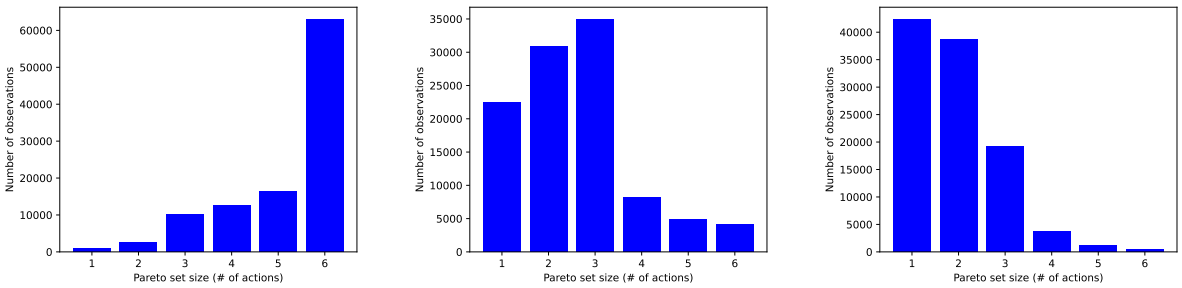
$$\Pi(s) = \Big\{ a \mid \nexists \bar{a} : \ \big(Q_d(s, \bar{a}) \geq Q_d(s, a), \forall d \in [D]\big) \text{ and } \big(Q_d(s, \bar{a}) > Q_d(s, a), \exists d \in [D]\big) \Big\}. \tag{2}$$

Our algorithm must be stable and powerful when n is large. However, the standard notion of Pareto optimality gets more and more unlikely to remove an action when n increases. Therefore, we define ϵ-strong Pareto optimal actions as follows, which works perfectly even for large values of n:

$$\Pi_\epsilon(s) = \Big\{ a \mid \forall \bar{a} \neq a : \ |\{d : Q_d(s, a) > Q_d(s, \bar{a})\}| > \epsilon \Big\}. \tag{3}$$

Note that Π_ϵ for $\epsilon = 0$ will be very similar to the standard notion of Pareto optimality. In the case $\Pi_\epsilon(s) =$, we will decrease ϵ in iterations until finding a non-empty set.

Now that we have obtained a pruned action set $\Pi(s)$ for each state, we train a standard double DQN agent restricted to actions within $\Pi(s)$. We will call this agent 'Pruned[Name of first phase method]'.



(a) $\epsilon = 0$ (b) $\epsilon = 0.2$ (c) $\epsilon = 0.3$

Figure 1: Distribution of size of pruned action sets for PrunedIndependentDQN.

Offline learning So far, we presented our algorithms in the context of off-policy learning. To move to offline learning, we use conservative Q-learning (CQL) [16] instead of DQN, defined as MultiCQL, in phase 1 and later train a CQL agent on pruned actions in phase 2. CQL aims to avoid the overestimation of Q-values for out-of-distribution states. This is achieved by estimating a function such that the expected value of a policy under this Q-function lower-bounds its true value. We implement this by augmenting the Q-function training with a regularizer that minimizes the soft-maximum of the Q-values $\log \sum_a \exp(Q(s,a))$ and maximizes the Q-value on the state-action pair seen in the dataset, Q(s, a). The overall CQL loss function is then given by the standard Q-learning objective augmented by the regularizer weighted by the hyperparameter β. The regularizer is defined as follows:

$$\beta\Big[\frac{1}{N}\sum_{i=1}^{N} \log\big(\sum_{a}\exp(Q(s_i,a))\big) - Q(s_i,a_i)\Big]. \tag{4}$$

4 Experiments

We begin by evaluating our algorithm within the OpenAI Gym LunarLander environment [3], which provides the advantage of allowing us to simulate the performance of the learned policy when rolled out. Subsequently, we test our framework in an offline setting using real health records of septic patients in the intensive care unit from the MIMIC (Medical Information Mart for Intensive Care)-III dataset (v1.4) [14].

4.1 LunarLander Experiments

The reward in the LunarLander environment is a sum of five (intermediate) rewards related to the shape and fuel efficiency of the lander and a final (sparse) reward based on the success of landing. Therefore, it provides a perfect simulation environment to test our algorithms.

Interestingly, a standard double DQN agent cannot learn to land solely based on the final reward, and the role of intermediate rewards seems to be vital. We would like to know if we have access to intermediate rewards, but we do not know how to mix them; can we prune the action set such that a standard double DQN now can effectively learn how to land?

Before answering this question, we first study the quality of pruning based on different algorithms and hyperparameters. Let $\boldsymbol{w}_*$ be the ground-truth weights to mix intermediate rewards. We first train a double DQN agent on $\boldsymbol{w}_*^{\mathbb{T}}\boldsymbol{r}+r_{final}$ and call it the optimal agent. Then, we obtain trajectories from the optimal agent interacting with the environment. At each state s of a trajectory, we obtain the set of pruned actions $\Pi(s)$. Note that these actions are obtained without the knowledge of $\boldsymbol{w}_*$ and solely based on intermediate rewards. Ideally, we would like $|\Pi(s)|$ to be relatively small compared to the total number of actions A while including the action a taken by the optimal agent.

Figure 1 shows the distribution of pruned action sets for PrunedMultiDQN with three different values of ϵ. Here, pruned action sets are ϵ-strong Pareto optimal actions obtained from the outputs of 30 intermediate double DQNs. As the results suggest, ϵ gives us great flexibility to control the aggressiveness of pruning.

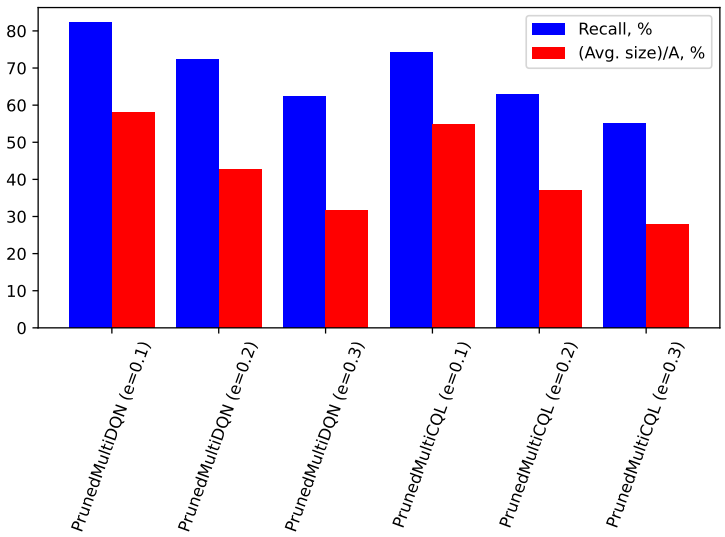
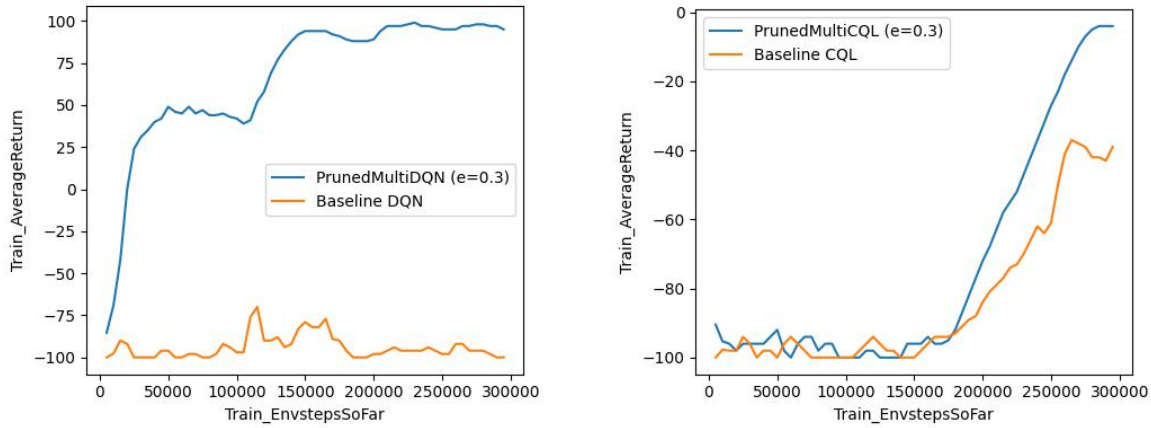


Figure 2: Chance level of having the optimal action is plotted along with the probability that the optimal action is included in the pruned action set. The chance level is calculated by the average size of the pruned set over the number of actions (6 in LunarLander).

Figure 2 shows the probability that the optimal action is within the pruned set $\Pi(s)$ (to be called recall). We have compared this value with the chance level that a set of randomly selected actions with the same size as the average size of $\Pi(s)$ consists of the optimal action. In other words, chance level is the average $|\Pi(s)|$ divided by A (which is 6 in LunarLander). Both, PrunedMultiDQN and PrunedMultiCQL maintain significantly more of the actions chosen by the optimal benchmark agent within their Pareto-set than would be by chance. Hence, the methods show great promise in reducing the size of the action set while maintaining a high recall.

Next, we get back to the question at the beginning of this section: Can a double DQN agent learn to land solely based on sparse final rewards while its available actions are restricted to the actions pruned based on intermediate rewards? Given the observations we had from the quality of pruning, we chose PrunedMultiDQN and PrunedMultiCQL with $\epsilon = 0.3$ for this experiment. Figure 3 shows the results for these two methods and a baseline double DQN or double CQL which is not assisted with pruned action sets. As the Figure suggests, unlike the baseline, PrunedMultiDQN and PrunedMultiCQL both have successfully learned an effective policy. PrunedMultiDQN also shows great stability across training steps. The effect for PrunedMultiCQL is less strong and policy improvement initiates later. This is not surprising since we are applying an offline algorithm in an off-policy setting. As CQL biases the policy towards choosing actions that have been observed in the replay buffer, and since the replay buffer initially consists of randomly chosen actions, it is intuitive that learning is slower.

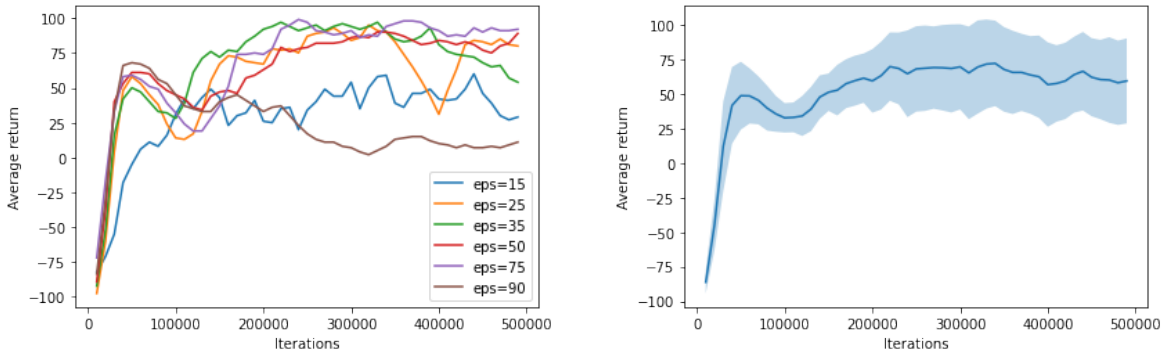
PrunedMultiDQN is stable and has a phase transition after 100k steps, which motivated us to conduct further analysis. We wanted to know how sensitive these observations are to the choice of ϵ and how the method would be affected by a less concentrated prior. Therefore, we ran similar experiments, with $\boldsymbol{w} \sim \text{uniform}^{\otimes D}(0, 1)$, 42 intermediate double DQNs, and various levels of ϵ. As the results show, ϵ should not be too low or too high and the same results hold for a wide range of $\epsilon \in [0.25, 0.75]$. These observations confirm the stability of PrunedMultiDQN to misspecified $\mathcal{P}$ and ϵ. Further, the phase transition is visible in almost all of the runs. We speculate that before this point, the policy is mostly relying on information from the intermediate rewards, which is implicitly embedded in the provided pruned actions. But after the transition, the policy mostly leverages the final sparse reward, which now has become possible.



(a) PrunedMultiDQN

(b) PrunedMultiCQL

Figure 3: PrunedMultiDQN and PrunedMultiCQL learning curves



(a) All runs

(b) Averaged run

Figure 4: PrunedMultiDQN learning curves for various choices of ϵ.

4.2 MIMIC experiments

We now move on to evaluating our framework in a real-life offline learning setup to estimate deep RL policies for the management of vasopressor and intravenous fluids in septic patients in the ICU. The main goal in this setting is to implement dosing decisions that minimize patient mortality, hence leading to 90-day mortality being usually used as the key indicator to assign rewards. However, 90-day mortality provides only a very sparse signal motivating us to investigate whether pruning the action space based on more noisy intermediate signals, such as the SOFA score or Lactate levels, may enable us to learn more effective policies.

4.2.1 Data

We will use data from a cohort of septic ICU patients in the MIMIC (Medical Information Mart for Intensive Care)-III dataset (v1.4) [14]. This dataset consists of deidentified electronic health records of over 40,000 patients that were admitted to the critical care units of the Beth Israel Deaconess Medical Center in Boston between 2001 and 2012. To construct our sample, we follow the pre-processing steps applied in Komorowski et al. [15]. We excluded patients younger than 18 years old at ICU admission or where mortality and intravenous fluid intake have not been documented. A trajectory includes data from up to 24 hours prior until 48 hours after Sepsis onset, recorded in 4-hour intervals. For each of these trajectories, we extracted a set of 48 variables, including demographics, Elixhauser comorbidity index [5], vital signs, laboratory test results, and medication dosing decisions.

Following previous work [15, 7], we discretize actions into 25 treatment choices (5 discrete levels for IV fluids and vasopressor dosage) and apply K-means clustering to obtain a 752-dimensional state space based on data from the 44 observational inputs (adding a terminal state for 90-day survival and death each). Since the resulting cluster indicators are numerically not meaningful, we additionally applied a variation of word2vec [21] to our trajectory sequences in order to obtain a 3-dimensional state representation. The resulting dataset consists of 20,912 ICU stay trajectories, of which 4,917 resulted in patient death within 90 days after the patient's critical care visit.

4.2.2 Rewards and model specification

Since we are dealing with an offline reinforcement learning setting, we apply PrunedMultiCQL. This algorithm extends PrunedMultiDQN by using a CQL loss function. All models are trained on 80% of the trajectory data, validated on 5% of the data, while the remaining 15% are held-out for final evaluation. In order to train PrunedIndependentCQL, we initially train 19 CQL models using different reward specifications. The rewards thereby are varying linear combinations of the 5 rewards presented below:

- **Binary SOFA score increase**: Assigns a reward of -1 if SOFA score increased (t to t+1)
- **Continuous SOFA score change**: Reward corresponds to the negative of the one-period change in SOFA score
- **Two-unit SOFA score change**: Assigns a reward of -1 if SOFA score increased by two units (t to t+1)
- **Binary Lactate level change**: Assigns a reward of -1 if Lactate level increased (t to t+1)
- **Continuous Lactate level change**: Reward corresponds to the negative of the one-period change in Lactate-levels

Both SOFA and Lactate levels are important health indicators for septic patients and have been considered in previous work of Raghu et al. [29]. SOFA [38] is a medical risk score that summarizes the extent of a patient's organ failure. Lactate levels measure cell-hypoxia, which is increased in septic patients since sepsis-induced low blood pressure reduces oxygen perfusion into tissue.

Each CQL model learns a Q-function as its critique that is specified as a 3-layer neural network model with a 64-dimensional hidden layer. Similarly, as in double DQN we use a separate target network during training. Using the Q-functions of these 19 CQL models, we then implement PrunedMultiCQL and estimate 5 different models varying the epsilon hyperparameter between 0 and 0.3. Throughout, we model using a CQL alpha of 0.2.

4.2.3 Observations

In order to evaluate PrunedMultiCQL, we investigate three key questions: 1) Does our pruning procedure allow us to significantly reduce the size of the action space? 2) Is the resulting action space meaningful? 3) Does the Q-function learned by our final model capture a relevant relationship to mortality?

In order to test whether we prune the action space in a meaningful way, Figure 5 presents the mean pruned action space size for different levels of the strictness of our strong Pareto criterion (indicated by ϵ). For the normal Pareto criterion (ϵ=0), the average action set size decreases by more than half from 25 to 13.2. For an ϵ of 30%, the mean action set size is only 2.0. We can therefore conclude that Pareto learning significantly reduces the size of the action space within which our PrunedIndependentCQL agent will be searching for an optimal policy.

Given this strong reduction in the size of the action space, a natural next question is whether the pruned space still contains the most important actions. Due to the lack of a ground truth best policy, we evaluate this by testing whether the pruned action sets contain the physician actions taken in each respective state in the hold-out set. Since physicians are trained professionals and observed more information than available in our dataset when deciding upon medication dosing, we assume that their decisions, on average, have been reasonable and should thus be accessible when training our PrunedMultiCQL agent. Similar to our LunarLander experiment, we call the share of Pareto sets that contain the recall of the physician action.

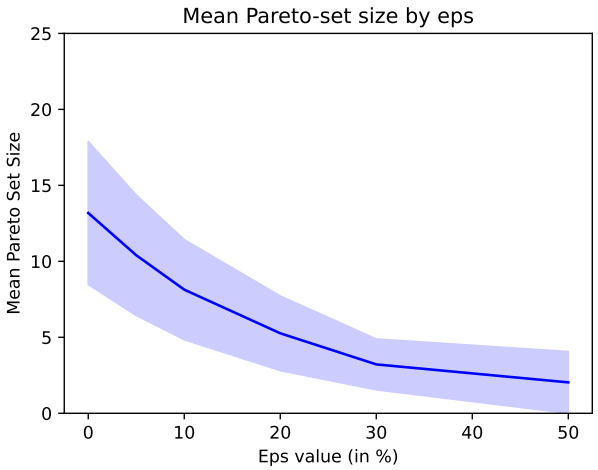
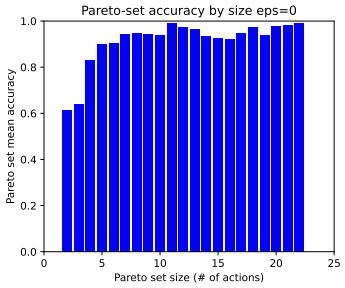
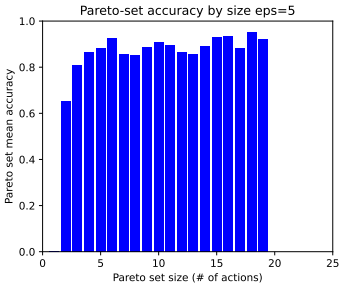


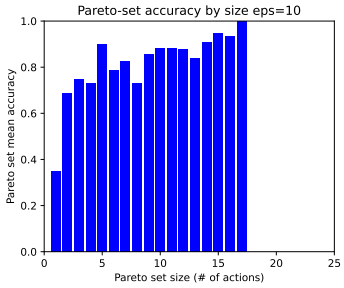
Figure 5: **Mean Pareto set size by pruning strength** This Figure displays the mean number of actions per Pareto set for different pruning strengths as indicated by the ϵ parameter. Shaded area represents a single standard deviation



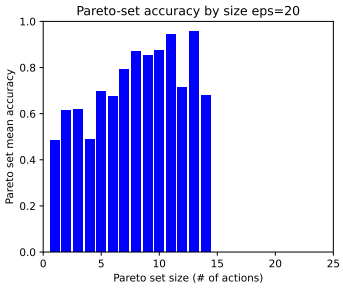
(a) ϵ=0



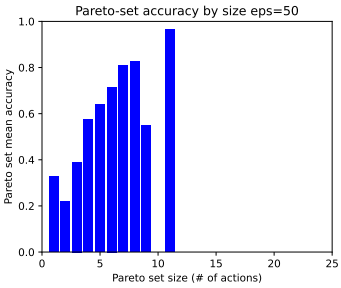
(b) ϵ=5%



(c) ϵ=10%

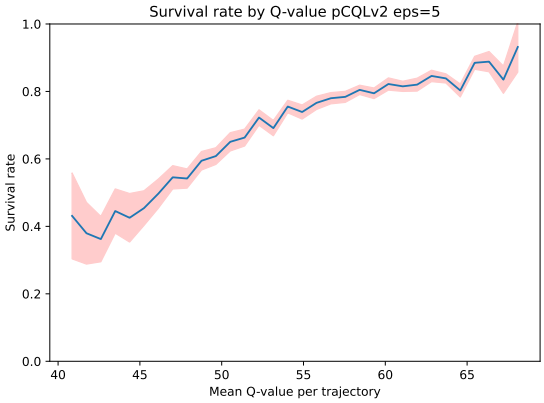


(d) ϵ=20%

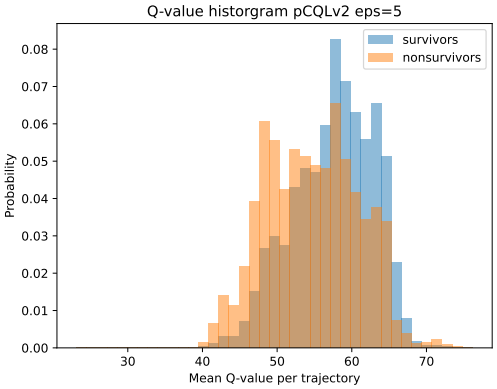


(e) ϵ=30%

Figure 6: **Recall per action set size** For each PrunedMultiCQL model, the Figure displays the Recall by action-set size. Recall is defined as the share of action sets that include the decision chosen by a physician in the respective state in our hold-out set. ϵ refers to the epsilon parameter that controls the pruning strength of our Pareto criterion.



(a) Survival rate by mean trajectory Q-value



(b) Distribution of survivor and non-survivor Q-values

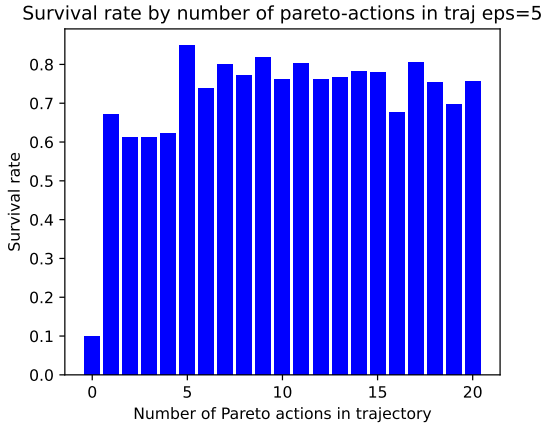
Figure 7: **Q-values and patient survival** Figure a) displays the average survival rate of patients based on the mean Q-value assigned by the learned PrunedMultiCQL model for the respective trajectory. The shaded area represents the 95% binomial proportion confidence interval. The figure displays the survival means for 50 bins over the range of the mean Q-values per patient trajectory. Figure b) show the distribution of Q-values of survivor compared to non-survivor trajectories.

Figure 6 displays the recall for different strictness levels and is split by action set size for each model. We make three key observations. First, recall is generally high, especially for ϵ strictness levels up to 10%, with mean recall for most pareto-set sizes ranging between 80-90%. Second, mean recall decreases with higher pruning strictness falling from 90.8% to 52.5% when increasing ϵ from 0 to 30%. Third, recall does not strongly correlate with action set size. Especially for ϵ levels up to 10%, accuracy appears to be unrelated to the action set size. This may indicate that our pruning criteria incorporate some notion of uncertainty when formulating action sets. States where the viable actions appear clear based on multiple criteria achieve small sets, while states where the correct action is more uncertain end up with a larger action set.

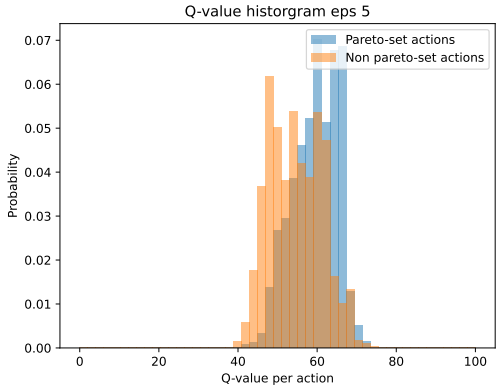
We now aim to evaluate whether our final model uncovers a relevant Q-function. Based on the pruning performance observations we choose the PrunedMultiCQL model with an $\epsilon = 5\%$ for the subsequent analysis. A key determinant of the relevance of our Q-function in this context is whether it successfully discriminates between high-risk and low-risk state action pairs. In order to evaluate this, Figure 7 plots the mean survival rates of patients by the mean Q-value of their trajectories. As desired, we observe a monotonically increasing curve, where the survival rate of patients on trajectories with the highest Q-values have around 50 percentage points higher survival rate than patients with the lowest Q-values. Further we see that the Q-value distribution of non-survivor trajectories is shifted to the left, further indicating a relationship between mortality and Q-values.

Given these findings, next we want to evaluate whether our pruning step itself could serve as an early-warning signal by indicating actions that should be avoided. To do so, in Figure 8a we plot the mean patient survival rate depending on the number of Pareto-actions within the trajectory. We observe 3 categories: First, trajectories with no Pareto-action have a survival rate of 10.0%, trajectories with 1-5 Pareto-actions have a survival rate around 60% and trajectories with more than 5 Pareto-actions obtain survival rates around 75%. This indicates that information about a state-action pair being within the Pareto-set may provide an approximate safety guide. It is fine if not all actions taken with a trajectory are part of the Pareto-set. However, consistently choosing actions that are outside of the Pareto-set estimated by PrunedMultiCQL may indicate high risks for patient health. Further, the left-shifted Q-value distribution of Non-Pareto actions in Figure 8b confirms this notion of increased risk being associated with these state-action pairs.

Overall, our evaluation of PrunedMultiCQL on the MIMIC dataset revealed promising results about the possibility of leveraging Pareto conditions to reduce the size of the action space while maintaining the most relevant actions. Further, we show that our final model uncovers a Q-function that meaningfully relates to mortality risk, and we observe that past trajectories that included actions within the algorithms Pareto-



(a) Survival rate by number of Pareto-actions in trajectory



(b) Distribution of Q-values of Pareto vs. Non-Pareto actions

Figure 8: **Pareto vs. Non-Pareto Actions** Figure a) displays the average survival rate of patients based on the number of Pareto-actions within their trajectory. Pareto-actions are defined as actions that are not eliminated during the pruning stage of PrunedMultiCQL. Figure b) show the distribution of Q-values of Pareto compared to Non-Pareto actions.

set have significantly increased survival risk. One may note that our policy evaluation for this algorithm has been limited by the fact that we are unable to evaluate policy performance beyond an analysis of the estimated Q-functions. For future work, it may be promising to conduct an additional evaluation of our offline learning method in a controlled semi-synthetic set-up, which would enable us to simulate rewards for the PrunedMultiCQL policy when rolled out.

5 Conclusion

We have investigated whether action space pruning provides a promising avenue to facilitate learning when the main reward signal is sparse, intermediate reward proxies are crude or there is ambiguity about the weights that should be given to optimal reward components. Our work introduces the algorithms PrunedMultiDQN and PrunedMultiCQL, which leverage an ensemble of Q-networks for Pareto-efficient action space pruning in order to facilitate more effective learning based on sparse main reward signals. In an off-policy setting tested on the LunarLander environment, both algorithms significantly outperform an agent trained on the sparse reward alone. Furthermore, robustness tests show PrunedMultiDQN is stable across a range of pruning-strictness levels. In addition, we validated our framework in an offline learning setting aiming to estimate vasopressor and intravenous fluid dosing policies for septic patients in an ICU setting. Our results indicate the effectiveness of the pruning approach to reduce the size of the action space while preserving relevant actions. Further, we demonstrate a strong relationship between mortality-risk and the Q-values assigned by our final model, and present how monitoring the overlap between physician actions and the Pareto-set estimated by our algorithm could be used to identify high-risk trajectories.

References

[1] Axel Abels et al. "Dynamic Weights in Multi-Objective Deep Reinforcement Learning". In: *International Conference on Machine Learning* (June 2019), pp. 11–20. URL: http://proceedings.mlr.press/v97/abels19a/abels19a.pdf.

[2] Leon Barrett and Srini Narayanan. "Learning all optimal policies with multiple criteria". In: *Proceedings of the 25th international conference on Machine learning - ICML '08* (2008). DOI: 10.1145/1390156.1390162. URL: http://dx.doi.org/10.1145/1390156.1390162.

[3] Greg Brockman et al. *OpenAI Gym*. 2016. eprint: arXiv:1606.01540.

[4] Li-Fang Cheng, Niranjani Prasad, and Barbara E. Engelhardt. "An Optimal Policy for Patient Laboratory Tests in Intensive Care Units". In: *Biocomputing 2019* (Nov. 2018). DOI: 10.1142/9789813279827\{_}0029. URL: http://dx.doi.org/10.1142/9789813279827_0029.

[5] Anne Elixhauser et al. "Comorbidity measures for use with administrative data". In: *Medical care* (1998), pp. 8–27.

[6] Mehdi Fatemi et al. "Dead-ends and Secure Exploration in Reinforcement Learning". In: *International Conference on Machine Learning* 97 (May 2019), pp. 1873–1881. URL: http://proceedings.mlr.press/v97/fatemi19a/fatemi19a.pdf.

[7] Mehdi Fatemi et al. "Medical Dead-ends and Learning to Identify High-risk States and Treatments". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 4856–4870.

[8] Joseph Futoma et al. "An Improved Multi-Output Gaussian Process RNN with Real-Time Validation for Early Sepsis Detection". In: *arXiv: Machine Learning* (Aug. 2017). URL: https://arxiv.org/pdf/1708.05894.pdf.

[9] Katharine E Henry et al. "A targeted real-time early warning score (TREWScore) for septic shock". In: *Science translational medicine* 7.299 (2015), 299ra122–299ra122.

[10] Kazuyuki Hiraoka, Manabu Yoshida, and Taketoshi Mishima. "Parallel reinforcement learning for weighted multi-criteria model with adaptive margin". In: *Cognitive Neurodynamics* 3.1 (Oct. 2008), pp. 17–24. DOI: 10.1007/s11571-008-9066-9. URL: http://dx.doi.org/10.1007/s11571-008-9066-9.

[11] Sandy Huang et al. "A Constrained Multi-Objective Reinforcement Learning Framework". In: *Conference on Robot Learning*. PMLR. 2022, pp. 883–893.

[12] Hitoshi Iima and Yasuaki Kuroe. "Multi-objective reinforcement learning for acquiring all Pareto optimal policies simultaneously - Method of determining scalarization weights". In: *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)* (Oct. 2014). DOI: 10.1109/smc.2014.6974022. URL: http://dx.doi.org/10.1109/smc.2014.6974022.

[13] Alex Irpan et al. "Off-Policy Evaluation via Off-Policy Classification". In: *Neural Information Processing Systems* 32 (Dec. 2018), pp. 5437–5448. URL: https://arxiv.org/pdf/1906.01624.pdf.

[14] Alistair EW Johnson et al. "MIMIC-III, a freely accessible critical care database". In: *Scientific data* 3.1 (2016), pp. 1–9.

[15] Matthieu Komorowski et al. "The artificial intelligence clinician learns optimal treatment strategies for sepsis in intensive care". In: *Nature medicine* 24.11 (2018), pp. 1716–1720.

[16] Aviral Kumar et al. "Conservative q-learning for offline reinforcement learning". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 1179–1191.

[17] Rongmei Lin et al. "A Deep Deterministic Policy Gradient Approach to Medication Dosing and Surveillance in the ICU". In: *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* (July 2018). DOI: 10.1109/embc.2018.8513203. URL: http://dx.doi.org/10.1109/embc.2018.8513203.

[18] Chunming Liu, Xin Xu, and Dewen Hu. "Multiobjective Reinforcement Learning: A Comprehensive Overview". In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 45.3 (Mar. 2015), pp. 385–398. DOI: 10.1109/tsmc.2014.2358639. URL: http://dx.doi.org/10.1109/tsmc.2014.2358639.

[19] Siqi Liu et al. "Reinforcement Learning for Clinical Decision Support in Critical Care: Comprehensive Review". In: *Journal of Medical Internet Research* 22.7 (July 2020), e18477. DOI: 10.2196/18477. URL: http://dx.doi.org/10.2196/18477.

[20] Shie Mannor and Nahum Shimkin. "The Steering Approach for Multi-Criteria Reinforcement Learning". In: *Neural Information Processing Systems* 14 (Jan. 2001), pp. 1563–1570. URL: https://papers.nips.cc/paper/1986-the-steering-approach-for-multi-criteria-reinforcement-learning.pdf.

[21] Tomas Mikolov et al. "Efficient estimation of word representations in vector space". In: *arXiv preprint arXiv:1301.3781* (2013).

[22] Volodymyr Mnih et al. "Human-level control through deep reinforcement learning". In: *Nature* 518.7540 (Feb. 2015), pp. 529–533. DOI: 10.1038/nature14236. URL: http://dx.doi.org/10.1038/nature14236.

[23] Hossam Mossalam et al. "Multi-Objective Deep Reinforcement Learning". In: *arXiv: Artificial Intelligence* (Oct. 2016). URL: https://arxiv.org/pdf/1610.02707.pdf.

[24] Sriraam Natarajan and Prasad Tadepalli. "Dynamic preferences in multi-criteria reinforcement learning". en. In: *Proceedings of the 22nd international conference on Machine learning - ICML '05*. Bonn, Germany: ACM Press, 2005, pp. 601–608. ISBN: 9781595931801. DOI: 10.1145/1102351.1102427. URL: http://portal.acm.org/citation.cfm?doid=1102351.1102427 (visited on 01/20/2023).

[25] Arne Peine et al. "Development and validation of a reinforcement learning algorithm to dynamically optimize mechanical ventilation in critical care". In: *npj Digital Medicine* 4.1 (Feb. 2021). DOI: 10.1038/s41746-021-00388-6. URL: http://dx.doi.org/10.1038/s41746-021-00388-6.

[26] Xuefeng Peng et al. "Improving Sepsis Treatment Strategies by Combining Deep and Kernel-Based Reinforcement Learning." In: *American Medical Informatics Association Annual Symposium* 2018 (Dec. 2018), pp. 887–896.

[27] Niranjani Prasad et al. "A Reinforcement Learning Approach to Weaning of Mechanical Ventilation in Intensive Care Units". In: *arXiv: Artificial Intelligence* (Apr. 2017). URL: https://arxiv.org/pdf/1704.06300.pdf.

[28] Aniruddh Raghu, Matthieu Komorowski, and Sumeetpal Singh. "Model-based reinforcement learning for sepsis treatment". In: *arXiv preprint arXiv:1811.09602* (2018).

[29] Aniruddh Raghu et al. "Continuous state-space models for optimal sepsis treatment: a deep reinforcement learning approach". In: *Machine Learning for Healthcare Conference*. PMLR. 2017, pp. 147–163.

[30] D. M. Roijers et al. "A Survey of Multi-Objective Sequential Decision-Making". In: *Journal of Artificial Intelligence Research* 48 (Oct. 2013), pp. 67–113. DOI: 10.1613/jair.3987. URL: http://dx.doi.org/10.1613/jair.3987.

[31] Suchi Saria. "Individualized sepsis treatment using reinforcement learning". In: *Nature Medicine* 24.11 (Nov. 2018), pp. 1641–1642. DOI: 10.1038/s41591-018-0253-x. URL: http://dx.doi.org/10.1038/s41591-018-0253-x.

[32] Harsh Satija et al. "Multi-Objective SPIBB: Seldonian Offline Policy Improvement with Safety Constraints in Finite MDPs". In: *arXiv: Learning* (May 2021). URL: http://export.arxiv.org/pdf/2106.00099.

[33] Shengpu Tang et al. "Clinician-in-the-Loop Decision Making: Reinforcement Learning with Near-Optimal Set-Valued Policies". In: *International Conference on Machine Learning* 1 (July 2020), pp. 9387–9396. URL: http://proceedings.mlr.press/v119/tang20c/tang20c.pdf.

[34] Gerald Tesauro et al. "Managing Power Consumption and Performance of Computing Systems Using Reinforcement Learning". In: *Neural Information Processing Systems* 20 (Dec. 2007), pp. 1497–1504. URL: http://lass.cs.umass.edu/~shenoy/courses/fall09/691gc/papers/Kephart-2007-1-NIPS2007_0906.pdf.

[35] Peter Vamplew et al. "Empirical evaluation methods for multiobjective reinforcement learning algorithms". In: *Machine Learning* 84.1-2 (Dec. 2010), pp. 51–80. DOI: 10.1007/s10994-010-5232-5. URL: http://dx.doi.org/10.1007/s10994-010-5232-5.

[36] Hado Van Hasselt, Arthur Guez, and David Silver. "Deep reinforcement learning with double q-learning". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 30. 1. 2016.

[37] Kristof Van Moffaert, Madalina M. Drugan, and Ann Nowe. "Scalarized multi-objective reinforcement learning: Novel design techniques". In: *2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)* (Apr. 2013). DOI: 10.1109/adprl.2013.6615007. URL: http://dx.doi.org/10.1109/adprl.2013.6615007.

[38] J. -L. Vincent et al. "The SOFA (Sepsis-related Organ Failure Assessment) score to describe organ dysfunction/failure". In: *Intensive Care Medicine* 22.7 (July 1996), pp. 707–710. ISSN: 1432-1238. DOI: 10.1007/BF01709751. URL: https://doi.org/10.1007/BF01709751.

[39] Runzhe Wu et al. "Offline Constrained Multi-Objective Reinforcement Learning via Pessimistic Dual Value Iteration". In: *Neural Information Processing Systems* 34 (Dec. 2021). URL: https://papers.nips.cc/paper/2021/hash/d5c8e1ab6fc0bfeb5f29aafa999cdb29-Abstract.html.

[40] Jie Xu et al. "Prediction-Guided Multi-Objective Reinforcement Learning for Continuous Robot Control". In: *International Conference on Machine Learning* 1 (July 2020), pp. 10607–10616. URL: http://proceedings.mlr.press/v119/xu20h/xu20h.pdf.

[41] Runzhe Yang, Xingyuan Sun, and Karthik Narasimhan. "A Generalized Algorithm for Multi-Objective Reinforcement Learning and Policy Adaptation". In: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*. Red Hook, NY, USA: Curran Associates Inc., 2019.

A Appendix

A.1 PrunedMultiCQL Learning Curves

We evaluate the performance of the estimated PrunedMultiCQL policies compared to a CQL model directly trained to optimize the sparse mortality reward. Unfortunately, our offline setup does not allow us to roll out the PrunedMultiCQL policy and observe the rewards it obtains. Thus, we approximate the performance of the policy by calculating the correlation between the Q-values estimates and the (mortality-based) reward-to-go achieved by the test-set trajectory. This should provide a crude assessment of the ability of the policies critic to distinguish between high and low mortality actions.

Figure 9 depicts the corresponding learning curves. The performance achieved by the PrunedMultiCQL policies is similar to the performance of the baseline CQL policy. Thus, at least by this measure, not indicating substantial RL policy improvement due to action space pruning.

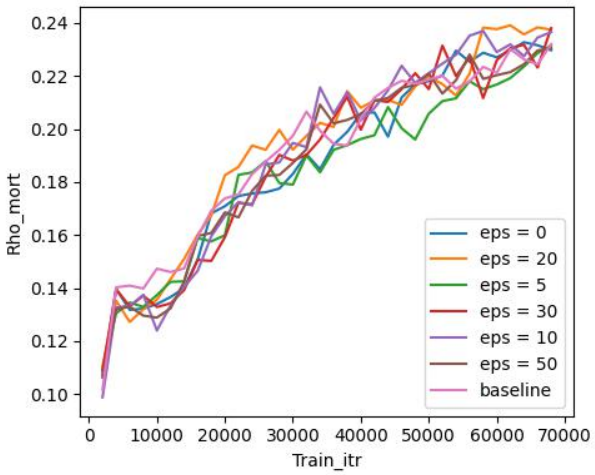


Figure 9: **Learning curves PrunedMultiCQL** Figure displays correlation between Q-values and mortality-based reward-to-go across training runs. PrunedMultiCQL models are indicated by their respective pruning level (ϵ). Baseline refers to a CQL model trained without prior action space pruning. Correlation is measured by applying Q functions to the hold-out set.