Task-Agnostic Safety for Reinforcement Learning (Supplementary Material)

Submitted to the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023). **To be used for reviewing only**.

A LEARNING TASK-AGNOSTIC SAFETY POLICY

Algorithm 1 Training TAS-RL safety policy

Parameter: θ_0, number of training episode N, adversary-non adversary rollout ratio β, expert-random rollout ratio η
Output: θ

function SAFETY ROLLOUT(env, s_0)

preset env variables to s_0

$s = s_0$ and $done = False$

while not $done$ **do**

$a_{safe} \sim \pi_\theta^{safe}(s)$

$s, s', C(s), done = env.step(a_{safe})$

$\mathcal{D}_{safe} \leftarrow \mathcal{D}_{safe} \cup \{s, s', C(s), done\}$

$s = s'$

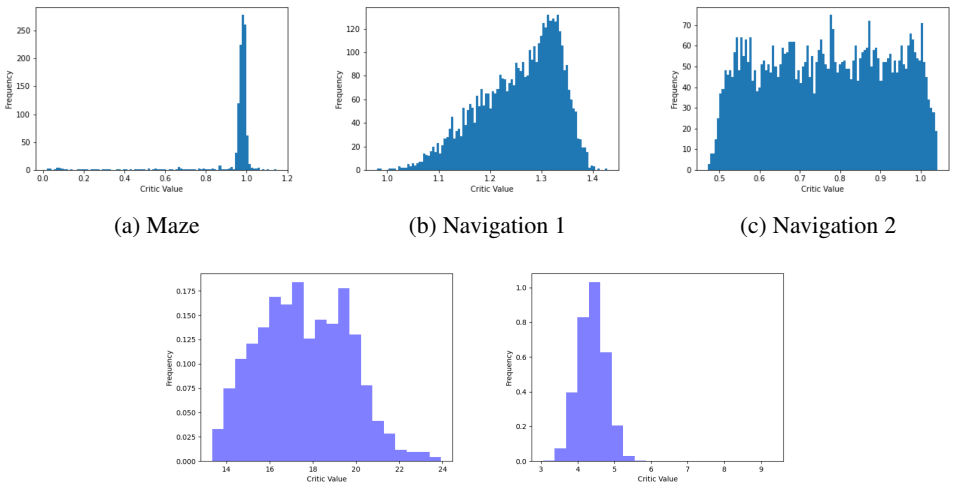
end while

end function

Initialize: $\pi^{task}, Q^{adv}, \pi^{adv}, \theta_0, \mathcal{D}_{safe}$
for $i \in (1, ..., N)$ **do**
  s = $env.reset()$ and $done = False$
  while not $done$ **do**
    $env_{copy} = copy(env)$
    SAFETY ROLLOUT(env_{copy}, s)
    if $random() < \beta$ **then**
      $a \sim \pi_\omega^{task}(s)$
    else
      if $random() < \eta$ **then**
        $a \sim \pi_\phi^{adv}(s)$
      else
        $a \sim env.action\_space.sample()$
      end if
    end if
    $s_n, r, done = env.step(a)$
    $s = s_n$
  end while
  Updates: Use sample from $\mathcal{D}_{safe}$ to update followings:
  (i) Update VAE: Update corresponding parameters of VAE's encoder and decoder using (Eqn. 12).
  (ii) Update parameter ψ of $Q^{est_{adv}}$: Perform gradient descent on (Eqn. 11) to learn ψ.
  (iii) Update parameter θ of π^{safe}: Perform gradient descent on (Eqn. 9) to learn θ.
end for

B SAFETY THRESHOLD

To find the safety threshold $\mathbb{T}safety$ value of the shield for each corresponding environment, we conducted a histogram analysis of $Q^{adv}\varphi(s, a)$ values. We collected the last 5 adversary critic $Q^{adv}\varphi(s, a)$ values before the agent enters into any unsafe states for 1000 episodes on each environment and drew their histogram, which is shown in Fig. 2. Based on the histogram, we define a particular shield's safety threshold value $\mathbb{T}safe$ for each environment, as presented in the Table.1.



(a) Maze (b) Navigation 1 (c) Navigation 2

(d) CarGoal1 (e) CarButton1

Figure 1: Histogram of adversary's critic value

Table 1: Safety threshold of the environments

Environment	Safety Threshold
Maze	0.90
Navigation 1	0.80
Navigation 2	0.50
CarGoal1	18.9
CarButton1	4.2

C ONLINE EXECUTION WITH SHIELD

Algorithm 2 Execution with shield and TAS-RL

Initialize: $\pi_{\omega}^{task}, Q_{\varphi}^{adv}, \pi_{\theta}^{safety}, \mathbb{T}_{shield}$
$s = env.reset()$ and $done = False$
while not $done$ **do**
    $a_{task} \sim \pi_{\omega}^{task}(s)$
    if $Q_{\varphi}^{adv}(s, a_{task}) > \mathbb{T}_{shield}$ **then**
        $a \sim \pi_{\theta}^{safety}(s)$
    else
        $a \sim a_{task}$
    end if
    $s_n, r_{adv}, done = env.step(a)$
    $s = s_n$
end while

D ENVIRONMENTS

We conducted performance evaluation of TAS-RL and the baseline on the following environments.

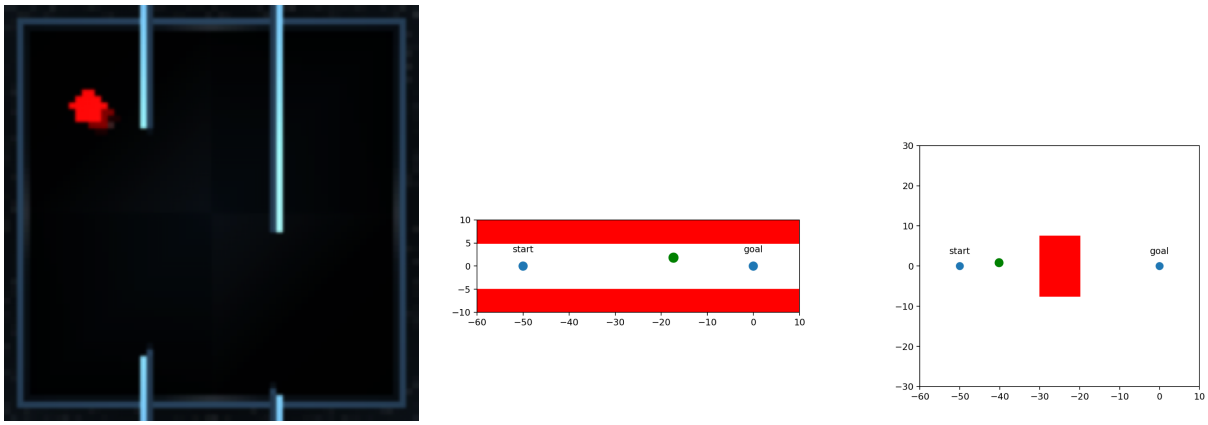
D.1 MUJOCO CMDP ENVIRONMENTS

We used the following CMDP environments from Thananjeyan et al. [2021] in our experiments. All the environments are built on MuJoCo Todorov et al. [2012] with continuous state-action space. Although the state space bound of these environments is $[-\infty, \infty]$, we have defined a particular fixed maximum distance to evaluate the task objective, which is to get as close to a specific goal. In addition, in the following three MuJoCo environments, any safety violation leads to episode termination. In all Navigation 1 and Navigation 2 scenarios, the agent starts from a fixed starting point.

Maze: This environment is built by Thananjeyan et al. [2021] from Nair et al. [2020] on MuJoCo. We also adopt their implementation of the environment in our experiment. In this environment the negative Euclidean from goal is used as the reward function. See (Table 2) for more information. Maze environment is shown in Fig.(2a)

Navigation 1: The dynamics of this environment follow a linear Gaussian model, with a Gaussian noise added to the system that is sampled from $N(0, \sigma^2 I)$. This environment also uses the negative Euclidean distance from the goal as the reward function. See Table 2 for more information. The Navigation 1 environment is shown in Figure 2b.

Navigation 2: Similar to the previous environment, the Navigation 2 environment has linear Gaussian dynamics with additive Gaussian noise sampled from $N(0, \sigma^2 I)$, and uses the negative Euclidean distance from the goal as the reward function. See Table 2 for more information. The Navigation 2 environment is depicted in Figure 2c.



(a) Maze (b) Navigation 1 (c) Navigation 2

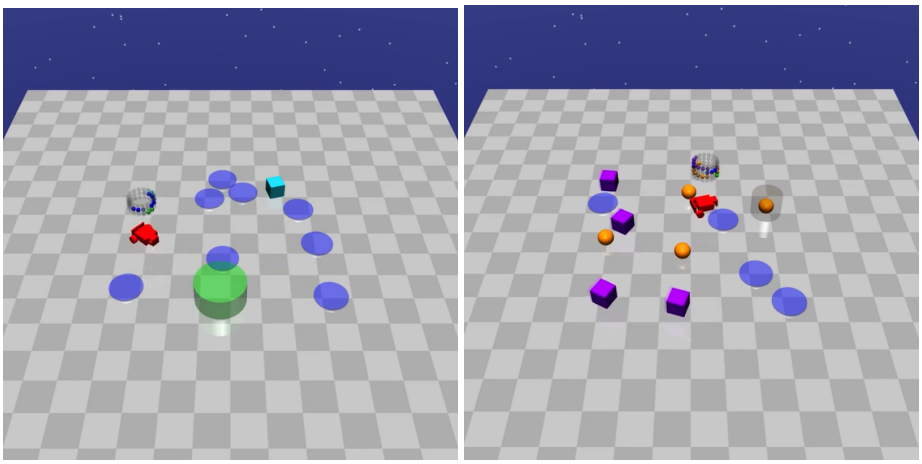
Figure 2: MuJoCo Experiment environments.

Table 2: Environment Specification

Environment	Attribute	Value/Specification
Maze	$Min_{distance}$	0.03
	$Max_{distance}$	5
	Maximum episode length, T_{max}	100
Navigation 1	$Min_{distance}$	4
	$Max_{distance}$	100
	Noise scale, σ	0.05
	Drag coefficient / air resistance	0.02
	Maximum episode length, T_{max}	100
Navigation 2	$Min_{distance}$	4
	$Max_{distance}$	100
	Noise scale, σ	0.05
	Drag coefficient/ air resistance	0.02
	Maximum episode length, T_{max}	100

D.2 SAFETYGYM ENVIOENMENTS

We also evaluated the performance of TAS-RL on the SafetyGym environment Ray et al. [2019]. Unlike the previous MuJoCo environments, episodes in the SafetyGym environment do not terminate when safety constraints are violated. As safety violations do not lead to episode termination in the SafetyGym environment, we did not use the cumulative constraint violation and ratio of success/violation as performance metrics. Instead, we evaluated the total default reward and cost value of the CMDP over an episode. We conducted experiments on the SafetyGym environment for the following two tasks:
(i) Goal: The task objective of the agent in the CarGoal environment (Fig. 3a) is to reach a specific goal while avoiding static obstacles or wrong goal buttons. The agent receives rewards based on its proximity to the goal, and upon reaching the goal, it receives a bonus reward. On the other hand, the agent incurs a penalty cost if it hits any obstacles or reaches a wrong goal. We employed the "Car" agent to solve this task in our experiments.
(ii) Button: This task environment (Fig. 3b) involves reaching a goal button while avoiding both static and dynamic obstacles. Because of the dynamic nature of the obstacles, this task is harder than the earlier goal task. The cost and rewards are given similarly to the goal task. We used the "Car" agent during experimenting on this task environment.



(a) CarGoal (b) CarButton

Figure 3: SafetyGym environments.

E BASELINES

We have compared TAS-RL against the following 7 baseline methods on the 3 MuJoCo environments:

i **SAC**: The standard SAC Haarnoja et al. [2018] algorithm is used to train the task policy that maximizes the task reward without any safety optimization.

ii **Lagrangian Relaxation (LR)**: In LR Thananjeyan et al. [2020], the task policy and safety are jointly learned through performing dual gradient descent on the following objective:

$$\max_{\lambda \geq 0} \min_{\pi}; \mathcal{L}\text{policy}(s;\pi) + \lambda \mathbb{E} a \sim \pi(\cdot|s)[Q_{\text{risk}}(s,a) - \epsilon_{\text{risk}}], \tag{1}$$

where Q_{risk} is a safety estimator and ϵ_{risk} is the safety threshold.

iii **Safety Q-Functions for RL (SQRL)**: SQRL Srinivasan et al. [2020] first learns a safety estimator Q^{π}_{safe}(s,a) through exploration and then uses it to employ a rejection sampling to learn safety policy by optimizing the following objective: $\mathcal{J}_{safe}(\psi) = \mathbb{E}_{s,a,s',a' \sim \rho_\pi}[Q^{\pi}_{safe}(s,a) - (I(s) + (1 - I(s))\gamma Q^{\pi}_{safe}(s',a'))]$, where $I(.)$ indicates if a state s is safe or not.

iv **Risk Sensitive Policy Optimization (RSPO)**: RSPO Mihatsch and Neuneier [2002] jointly learns the task and safety by minimizing the objective function:

$$\mathcal{L}_{policy}(s;\pi) + \lambda_t \mathbb{E}_{a \sim \pi(.|s)}[Q_{risk}(s,a) - \epsilon_{risk}] \tag{2}$$

where λ_t reduces to zero.

v **Critic Penalty Reward Constrained Policy Optimization (RCPO)**: RCPO Tessler et al. [2018] uses a risk estimator $Q^{\pi}_{\text{risk}}(s, a)$ to jointly learn task and safety by maximizing the policy objective:

$$\mathop{\mathbb{E}}_{\pi} \sum_{t=0} \gamma^t R(s_t, a_t) - \lambda Q^{\pi}_{risk}(s_t, a_t) \tag{3}$$

Its multiplier update is similar to LR.

vi **Reward Penalty (RP)**: RP Thananjeyan et al. [2021] applies a weighted penalty to the MDP reward function $\mathcal{R}(\cdot, \cdot)$ if a safety constraint is violated, indicated by function $C(\cdot)$, e.g., $\mathcal{R}(s_t, a_t) - \lambda C(s_t)$, where λ is the penalty weight.

vii **Recovery RL (Model Free) (RRL-MF)**: RRL-MF Thananjeyan et al. [2021] retrieves a critic safety estimator $Q^{\pi}_{\text{risk}}(s, a)$ using DDPG Silver et al. [2014] and later uses it to employ a shield and to learn the safety policy jointly with the task reward by solving the objective function:

$$\pi_{\text{safe}} = \underset{\pi \in \Pi}{\text{argmax}} [\mathcal{R}^{\pi} : Q^{\pi}_{\text{risk}}(s, a) \leq \epsilon_{\text{risk}}], \tag{4}$$

where ϵ_{risk} is the risk threshold.

In addition, for comparative performance analysis on the SafetyGym environments we used the following two baselines:

i **Trust Region Policy Optimization (TRPO)**: We use TRPO Schulman et al. [2015] as an unconstrained baseline.

ii **Constrained Policy Optimization (CPO)**: The on-policy CPO Achiam et al. [2017] is used as an Safe-RL baseline.

E.1 IMPLEMENTATION DETAILS

E.1.1 TAS-RL Implementation:

We use the DDPG algorithm Silver et al. [2014] framework to implement the training of TAS-RL's stochastic safety policy π^{safety} using Algorithm 1. We integrated the stochastic policy definition similar to the implementation of SAC in Tandon [2020] in our implementation. Furthermore, we replaced the DDPG policy update with TAS-RL safety policy π^{safety} objective (Eq. 8) and also incorporated changes relevant to the VAE as well as rollout strategy to improve exploration. We list the corresponding model architecture and hyperparameters that we used across the environments in Table 3, which were retrieved through cross-validation. The safety training curve for the maze, navigation 1, and navigation 2 environment have been given in Fig.4

Table 3: TAS-RL Hyperparameters

Baseline	**Network Parameters**						**TAS-RL Exploration Parameters**	
	Layer Size	**Number of Hidden Layers**	**Hidden Layer Activation**	**Optimizer**	**Learning Rate**	**Output Activation**	**Beta**	**Eta**
Maze	256	2	ReLU	Adam	3×10^{-4}	Tanh	0.7	0.5
Navigation 1	512	2	ReLU	Adam	3×10^{-6}	Tanh	0.7	0.5
Navigation 2	512	2	ReLU	Adam	3×10^{-4}	Tanh	0.7	0.5
CarGoal	256	3	ReLU	Adam	1×10^{-4}	Tanh	0.65	0.5
CarButton	256	3	ReLU	Adam	1×10^{-4}	Tanh	0.80	0.5

E.1.2 Baseline Implementation

We use the implementation of all the baselines along with their hyper-parameter settings from Thananjeyan et al. [2021]. The hyper-parameter settings for each algorithms is given in Table 4

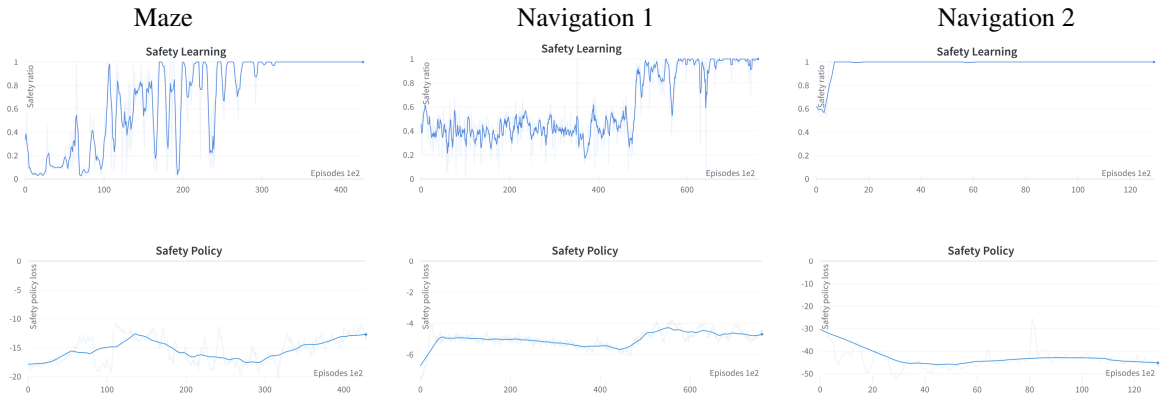


Figure 4: Training curve of TAS-RL on MuJoCo environments.

Table 4: Hyper-parameters of the baselines

Baseline	$Q_{risk}(s,a)$				(γ_{risk})	(ϵ_{risk})	(λ)
	Hidden size	Number of hidden layers	Output layer's activation	Environment			
LR	256	2	Sigmoid	Maze	0.5	0.15	100
				Navigation 1	0.8	0.3	5000
				Navigation 2	0.65	0.2	1000
SQRL	256	2	Sigmoid	Maze	0.5	0.15	100
				Navigation 1	0.8	0.3	5000
				Navigation 2	0.65	0.2	1000
RSPO	256	2	Sigmoid	Maze	0.5	0.15	200
				Navigation 1	0.8	0.3	1000
				Navigation 2	0.65	0.2	2000
RCPO	256	2	Sigmoid	Maze	0.5	0.15	50
				Navigation 1	0.8	0.3	1000
				Navigation 2	0.65	0.2	5000
RP	256	2	Sigmoid	Maze	-	-	50
				Navigation 1	-	-	1000
				Navigation 2	-	-	3000
RRL-MF	256	2	Sigmoid	Maze	0.5	0.15	-
				Navigation 1	0.8	0.3	-
				Navigation 2	0.65	0.2	-

F ROBUSTNESS EVALUATION

We argue to evaluate safety performance in uncertain conditions. To this end, during the experiment, we explicitly introduce two types of uncertainty i.e. (i) external action space perturbation and (ii) change of environment dynamics. First, we test TAS-RL against all the baselines under two white box action space perturbations from Tessler et al. [2019] while keeping the testing environment's dynamics same as the training environment. Details about the two-white box action perturbation is given below:

F.1 EXTERNAL ACTION SPACE PERTURBATION

(a) Random action perturbation: If the task policy of the RL agent selects an action a where $a \in \mathcal{A}$, random action perturbation replaces the selected action with any random action sampled from the action space.

(b) Alternative adversarial action: We take the form of external action perturbation from Tessler et al. [2019]. In this regard, we use the previously trained adversary's policy π^{adv} to select an alternative adversarial action given state s_t such that $a_{adv} \sim \pi^{adv}(s_t)$. Finally, we explicitly alternate the agent's action a selected by its task policy, i.e. $a \sim \pi^{task}$, with the action selected by the adversary policy a_{adv}.

F.2 CHANGE OF ENVIRONMENT DYNAMICS

In the second category of uncertainty injection, we change the dynamics of the testing environment from the training environment by varying their dynamics attributes such as air resistance and noise coefficient. This affects the state-action transition $\mathcal{P}(s_{t+1}|a_t, s_t)$ of the MDP.

Considering the inability to alter the dynamic specifications, we exclude the Maze environment and all the SafetyGym environments from this experiment. The changes in dynamics used in our experiments are shown in Table 5.

Table 5: Environment dynamics variation

			(variation scale)				
			2x	4x	6x	8x	10x
Environment	Dynamic Attribute	Default	Attribute values				
Navigation 1	Noise Scale	0.05	0.1	0.2	0.3	0.4	0.5
	Air Resistance	0.02	0.04	0.08	0.12	0.16	0.2
Navigation 2	Noise Scale	0.05	0.1	0.2	0.3	0.4	0.5
	Air Resistance	0.02	0.04	0.08	0.12	0.16	0.2

G ROBUSTNESS ANALYSIS RESULT

G.1 ROBUSTNESS ANALYSIS WITH EXTERNAL PERTURBATION (WITHOUT ANY CHANGE IN ENVIRONMENT DYNAMICS)

The cumulative constraint violation of TAS-RL and the baseline in presence of external action perturbation is shown in Table 6, Table 7, and Table 8 for maze, navigation 1 and navigation 2 environment respectively.

Table 6: Cumulative constraint violations with different rates of perturbation in Maze environment

Algorithms	**Mean Cumulative constraint violation over 100 episode**									
	Random Perturbation					**AAA Perturbation**				
	Perturbation Rate					**Perturbation Rate**				
	0%	**25%**	**50%**	**75%**	**100%**	**0%**	**25%**	**50%**	**75%**	**100%**
TAS-RL	0	0	0	0	0	0	0	0	0	0
SAC	0	3	27	38	48	0	25	65	93	96
RP	0	22	39	55	55	0	86	91	95	96
RSPO	0	23	42	54	62	0	85	93	95	96
LR	0	20	46	52	61	0	85	93	95	96
RCPO	0	24	41	54	60	0	83	92	95	96
RRL_MF	0	3	8	8	13	0	0	0	0	0
SQRL	27	32	38	54	61	27	76	91	95	96

Table 7: Cumulative constraint violations with different rates of perturbation in Navigation 1 environment

Algorithms	**Mean Cumulative constraint violation over 100 episode**									
	Random Perturbation					**AAA Perturbation**				
	Perturbation Rate					**Perturbation Rate**				
	0%	**25%**	**50%**	**75%**	**100%**	**0%**	**25%**	**50%**	**75%**	**100%**
TAS-RL	0	0	0	0	0	0	0	0	0	0
SAC	0	0	0	1	17	0	83	92	94	95
RP	0	0	0	0	1	0	6	69	91	95
RSPO	0	0	0	0	0	0	2	66	90	95
LR	0	0	0	0	0	0	7	66	91	94
RCPO	0	0	0	0	0	0	4	68	90	94
RRL_MF	0	0	0	0	0	0	0	0	0	0
SQRL	0	0	0	2	4	0	67	90	93	94

Table 8: Cumulative constraint violations with different rates of perturbation in Navigation 2 environment

Algorithms	**Mean Cumulative constraint violation over 100 episode**									
	Random Perturbation					**AAA Perturbation**				
	Perturbation Rate					**Perturbation Rate**				
	0%	**25%**	**50%**	**75%**	**100%**	**0%**	**25%**	**50%**	**75%**	**100%**
TAS-RL	0	0	0	0	0	0	0	0	0	0
SAC	0	0	4	11	21	0	46	76	80	79
RP	1	2	0	2	5	1	75	79	79	80
RSPO	1	1	1	5	5	1	75	79	79	79
LR	0	2	2	2	8	0	76	79	79	79
RCPO	1	2	0	5	5	1	75	79	80	79
RRL_MF	0	0	0	0	0	0	1	0	0	0
SQRL	15	16	12	11	5	15	42	69	79	80

G.2 ROBUSTNESS ANALYSIS WITH CHANGE IN ENVIRONMENT DYNAMICS

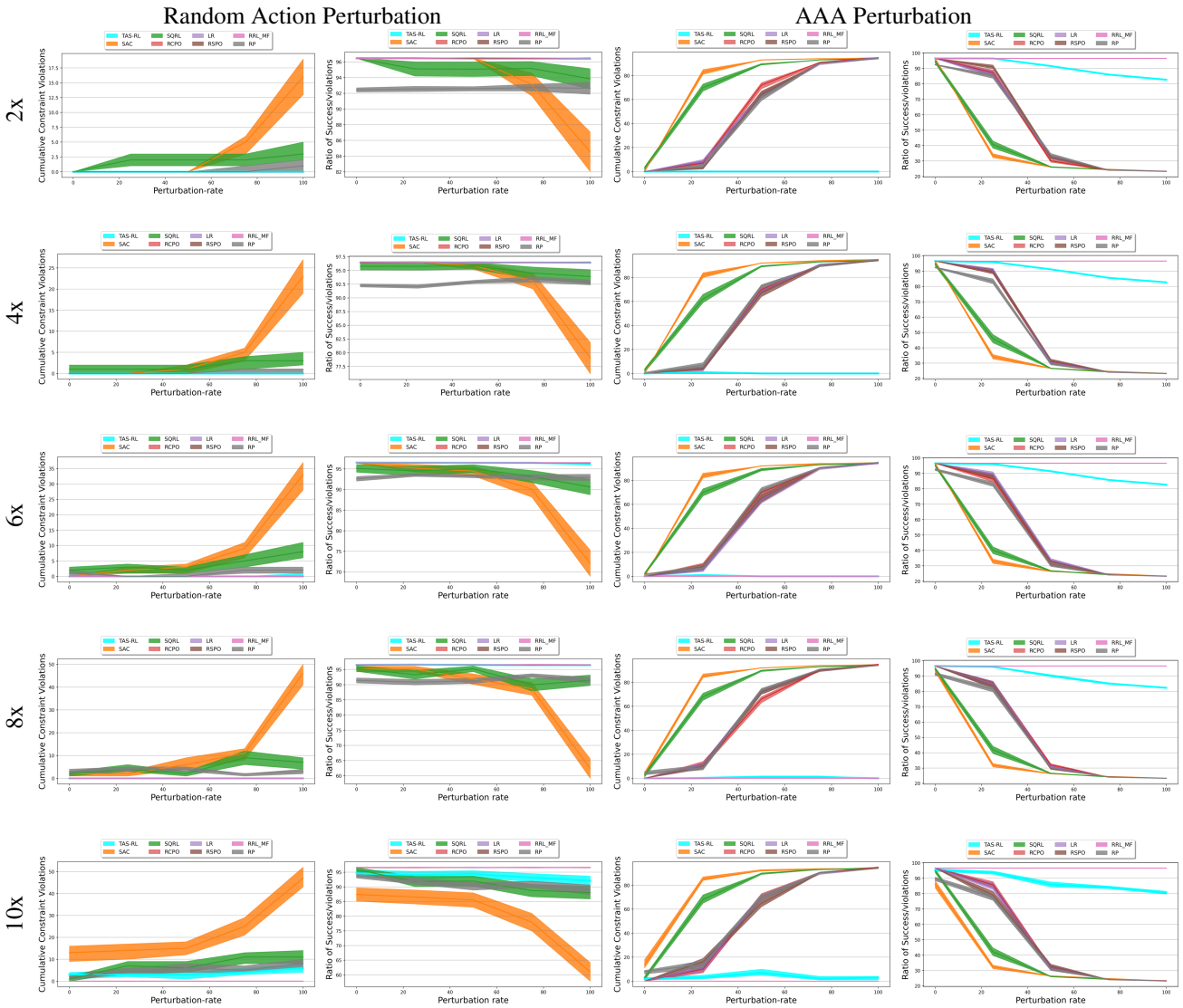


Figure 5: Comparative performance on Navigation 1 test environment with varying dynamics. From the top the five rows respectively represent 2 times, 4 times, 6 times, 8 times and 10 times dynamics variation. Performance under random action perturbations is presented in the leftmost two columns whereas the rightmost two columns show the performance under adversarial alternative action perturbations. The results are averaged over 100 episodic runs for each algorithm.

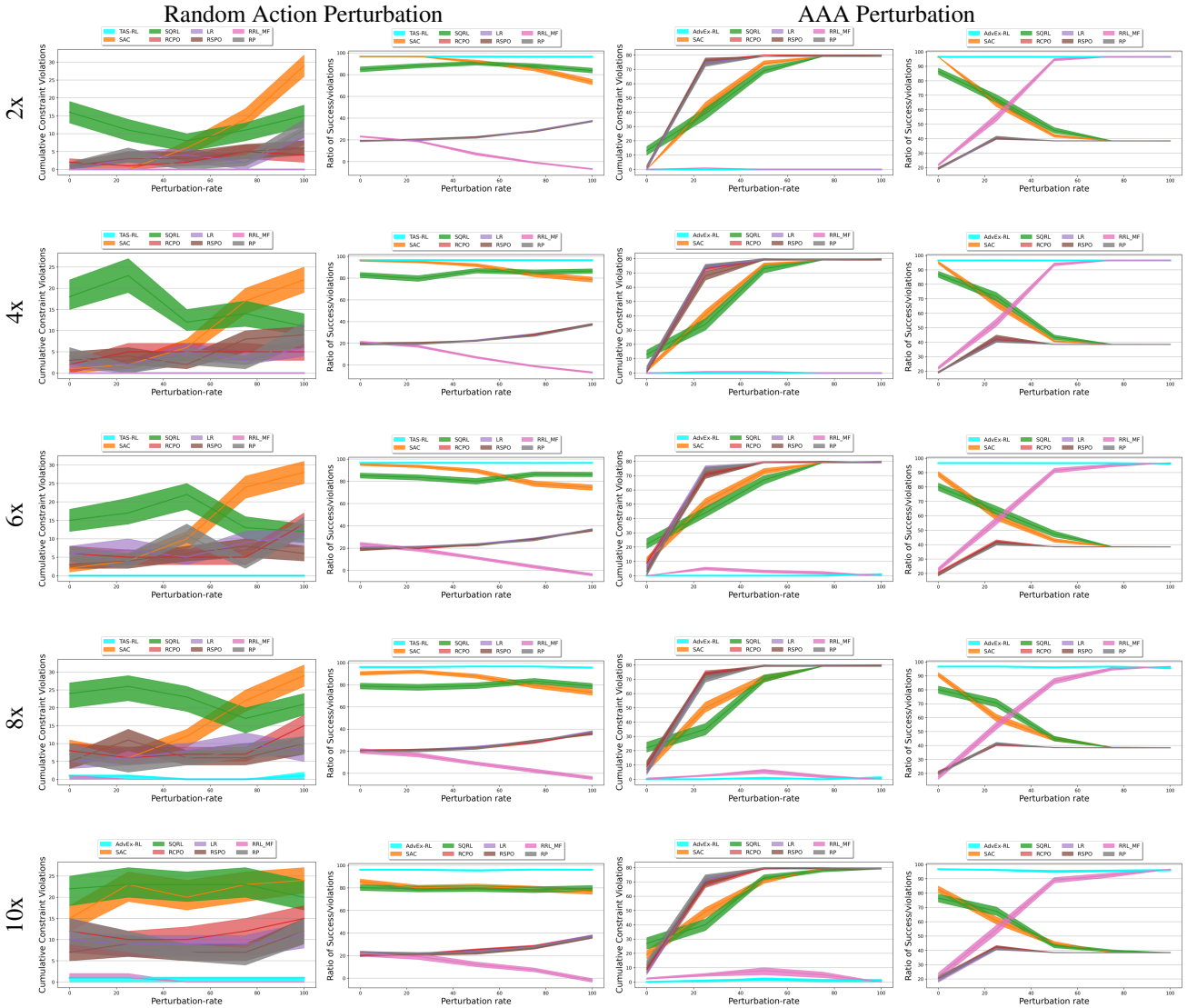


Figure 6: Comparative performance on Navigation 2 test environment with varying dynamics. From the top the five rows respectively represent 2 times, 4 times, 6 times, 8 times and 10 times dynamics variation. Performance under random action perturbations is presented in the leftmost two columns whereas the rightmost two columns show the performance under adversarial alternative action perturbations. The results are averaged over 100 episodic runs for each algorithm.

Performance of TAS-RL and the baseline in the presence of external perturbation with varying dynamic change is shown in Fig. 5 for the Navigation 1 environment and Fig. 6 for the Navigation 2 environment. As the environment dynamics varied by a factor of 10, it became evident that all safety techniques, including TAS-RL, were negatively impacted by the change in dynamics. Nevertheless, the influence of dynamic change on TAS-RL and RRL-MF was less pronounced than on the other baselines.

References

Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International conference on machine learning*, pages 22–31. PMLR, 2017.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep

reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

Oliver Mihatsch and Ralph Neuneier. Risk-sensitive reinforcement learning. *Machine learning*, 49(2):267–290, 2002.

Suraj Nair, Silvio Savarese, and Chelsea Finn. Goal-aware prediction: Learning to model what matters. In *International Conference on Machine Learning*, pages 7207–7219. PMLR, 2020.

Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking Safe Exploration in Deep Reinforcement Learning. 2019.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. PMLR, 2014.

Krishnan Srinivasan, Benjamin Eysenbach, Sehoon Ha, Jie Tan, and Chelsea Finn. Learning to be safe: Deep rl with a safety critic. *arXiv preprint arXiv:2010.14603*, 2020.

P. Tandon. Pytorch implementation of soft actor critic. `https://github.com/pranz24/pytorch-soft-actor-critic`, 2020.

Chen Tessler, Daniel J Mankowitz, and Shie Mannor. Reward constrained policy optimization. *arXiv preprint arXiv:1805.11074*, 2018.

Chen Tessler, Yonathan Efroni, and Shie Mannor. Action robust reinforcement learning and applications in continuous control. In *International Conference on Machine Learning*, pages 6215–6224. PMLR, 2019.

Brijen Thananjeyan, Ashwin Balakrishna, Ugo Rosolia, Felix Li, Rowan McAllister, Joseph E Gonzalez, Sergey Levine, Francesco Borrelli, and Ken Goldberg. Safety augmented value estimation from demonstrations (saved): Safe deep model-based rl for sparse cost robotic tasks. *IEEE Robotics and Automation Letters*, 5(2):3612–3619, 2020.

Brijen Thananjeyan, Ashwin Balakrishna, Suraj Nair, Michael Luo, Krishnan Srinivasan, Minho Hwang, Joseph E Gonzalez, Julian Ibarz, Chelsea Finn, and Ken Goldberg. Recovery rl: Safe reinforcement learning with learned recovery zones. *IEEE Robotics and Automation Letters*, 6(3):4915–4922, 2021.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012. doi: 10.1109/IROS.2012.6386109.