MOPA: a Minimalist Off-Policy Approach to Safe-RL

Hao Sun * Cambridge
Ziping Xu University of Michigan
Zhenghao Peng UCLA
Meng Fang University of Liverpool
Bo Dai Shanghai AI Lab
Bolei Zhou UCLA

Abstract

Safety is one of the crucial concerns for the real-world application of reinforcement learning (RL). Previous works consider the safe exploration problem as Constrained Markov Decision Process (CMDP), where the policies are being optimized under constraints. However, when encountering any potential danger, human tends to stop immediately and rarely learns to behave safely in danger. Moreover, the off-policy learning nature of humans guarantees high learning efficiency in risky tasks. Motivated by human learning, we introduce a Minimalist Off-Policy Approach (MOPA) to address Safe-RL problem. We first define the Early Terminated MDP (ET-MDP) as a special type of MDPs that has the same optimal value function as its CMDP counterpart. An off-policy learning algorithm MOPA based on recurrent models is then proposed to solve the ET-MDP, which thereby solves the corresponding CMDP. Experiments on various Safe-RL tasks show a substantial improvement over previous methods that directly solve CMDP, in terms of higher asymptotic performance and better learning efficiency.

1 Introduction

While reinforcement learning (RL) achieves great successes in solving challenging decision making problems, several critical issues remain to be addressed before its real-world applications. Safety concern is one of those demanding issues [1]. As the learning paradigm of RL is composed of exploration and exploitation with experiences from trial-and-error [2], the agents need to attempt a wide range of states and actions to better estimate their values, some of which are not safe and may lead to major damage.

To tackle the Safe-RL problem, Altman [3] defines the Constrained Markov Decision Processes (CMDPs), where the policy optimization of standard RL algorithms should be executed in a constrained policy class. Many deep RL approaches for CMDPs are proposed after the rising of deep neural network as the function approximators: those works mainly focus on the optimization of the CMDP tasks, $i.e.$, how to effectively convert a CMDP task into a solvable form. Achiam et al. [4] extend the trust region methods [5] into the context of CMDPs and guarantees the monotonicity of policy improvement; the Lagrangian methods, barrier (interior point) methods used in normal constrained optimization tasks and Lyapunov methods are extended to solve the CMDPs based on their MDP counterparts in [6–13]; another approach is based on safety-critic, where an additional critic is learned beside the primal critic for rewards to predict the cost of possible behaviors [14–16]. Based on the CMDP formalism, almost all those previous works on are derived up on on-policy methods except in the work of Srinivasan et al. [16], an off-policy critic is explored to assist the reward learning.

In general, there are two important open questions we are well-motivated to solve:

*hs789@cam.ac.uk

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

1. Can Safe-RL tasks be formulated under a framework that is suitable for off-policy learning?
2. Can the sample efficiency of Safe-RL tasks be improved with off-policy learning?

Tackling the first question opens up new possibilities of applying various off-policy algorithms in solving Safe-RL problems, and tackling the second question leads to many potential benefits in many real-world applications such as healthcare and finance where safety is one of the major concerns.

In this work, we address those challenges and give positive answers to both questions. To answer the first question, our solution draws key insight from human learning that human tends to avoid danger or cost in the learning process by immediately stopping their risky behaviors. Formally, we begin with theoretical analysis on the problem formalism of Safe-RL, and show that those tasks—previously solved under framework of CMDPs—can be equivalently solved through their early terminated counterparts, which we term as ET-MDP in this work. The ET-MDP framework reveals the possibility of solving the Safe-RL tasks with a minimalist approach: in principle, those tasks can be solved by terminating episodes whenever the constraints are violated. To answer the second question, we face the challenge of **limited state visitation**: under the proposed ET-MDP framework, a roll-out episode will be terminated whenever an agent violates the constraints. Thereafter, the agent may be potentially limited to a small state space, leading to the difficulty of effective learning [17]. We show in our work that while conventional RL algorithms like TD3 [18] are not ideal choices in solving those ET-MDP tasks, our proposed off-policy learning algorithm based on recurrent models can properly circumvent the difficulty of limited state visitation during the learning.

We evaluate our method on a range of Safe-RL environments, including both the deterministic and the stochastic environments with different types of constraints. MOPA shows a remarkable performance in terms of both learning efficiency and asymptotic performance under constraints.

2 Preliminaries

Constrained Markov Decision Process. The standard formulation in solving Safe-RL tasks is based on CMDP. We consider the typical setting of deterministic CMDP with a fixed horizon $H \in \mathbb{N}^+$ denoted by a tuple $(\mathcal{S}, \mathcal{A}, H, r, c, C, \mathcal{T})$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action space; $r, c : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ denote the reward function and cost function; $C \in \mathbb{R}^+$ is the upper bound on the permitted expected cumulative cost; $\mathcal{T} : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$ denotes the transition function.

We use Π to denote the stationary policy class, where $\Pi = \{\pi : \mathcal{S} \times \mathcal{A} \to [0, 1], \sum_a \pi(a|s) = 1\}$. An algorithm for CMDP is to find $\pi^* \in \Pi$ as the result of the following optimization problem,

$$\max_{\pi \in \Pi} \mathbb{E}_{\tau \sim \pi, \mathcal{T}}[\sum_{t=1}^{H} r_t], \quad \text{s.t.} \quad \mathbb{E}_{\tau \sim \pi, \mathcal{T}}[\sum_{t=1}^{H} c_t] \leq C, \tag{1}$$

where the expectation is taken over the trajectory $\tau = (s_1, a_1, r_1, \ldots, s_H, a_H, r_H)$ generated by policy π under the environment $\mathcal{T}$.

Lagrangian Method. The Lagrangian method relaxes the problem Eqn.(1) to an unconstrained optimization problem with a penalty term

$$\pi^* = \max_{\pi \in \Pi} \min_{\lambda \geq 0} \mathbb{E}_{\tau \sim \pi, \mathcal{T}}[\sum_{t=1}^{H} r_t - \lambda c_t] + \lambda C, \tag{2}$$

where $\lambda \geq 0$ is known as the Lagrangian multiplier. Suppose the policy π is parameterized by θ, $i.e.$, $\pi = \pi_\theta$, the optimization over θ and λ can be conducted iteratively through policy gradient ascent and stochastic gradient descent respectively according to Eqn.(2). Chow et al. [11] points out that one of the possible defects of the Lagrangian methods is the violation of constraints during training, which is successfully solved by our proposed method.

Constrained Policy Optimization. Achiam et al. [4] proposes the Constrained Policy Optimization (CPO), an analytical way to solve CMDP through trust region optimization. Specifically, CPO develops an approximation of Eqn.(1) by replacing the objective and constraints with surrogate functions [4, 5] and provides theoretical analysis on the worst case performance and constraint violation. In CPO, the policy is updated as:

$$\pi_{k+1} = \arg\max_{\pi \in \Pi} \mathbb{E}[A^{\pi_k}_{r,1}(s, a)], \quad \text{s.t.} \widetilde{J}_c(\pi_k) \leq C, \quad \bar{D}_{KL}(\pi || \pi_k) \leq \delta, \tag{3}$$

Table 1: Comparison between MOPA and related work. MOPA is the only method satisfies all the following desiderata: (1) off-policy learning for sample efficiency (2) capable of tackling the non-Markovianity in Safe-RL (3) early termination to minimize training cost.

Method Class	Example	w/o Human Intervention	Off-Policy	Backbone	Marikovian Tasks	Non-Markovian Tasks	Model
Solve CMDP [4, 11]	CPO	✓	-	TRPO	✓	-	CMDP
Safety-Critic [15, 24]	WCSAC	✓	✓	SAC	✓	-	CMDP
Human-in-the-loop [25–27]	HIRL	-	✓	DQN/A3C	✓	✓	MDP
MOPA	(Ours)	✓	✓	Recurrent Model	✓	✓	ET-MDP

wherein $\widetilde{J}_c(\pi_k) = \mathbb{E}_{\tau\sim\pi_k,\mathcal{T}}[\sum_{t=1}^{H} c_t] + \mathbb{E}_{s,a}[A^{\pi_k}_{c,1}(s,a)],\ k = 0, 1, ..., K$. Here $A^{\pi_k}_{r,i}(s,a)$ and $A^{\pi_k}_{c,i}(s,a)$ denote the advantage functions of reward and cost at step i respectively. CPO is closely-connected to the θ-projection approach of Chow et al. [11]. The close relationship between CPO and the family of trust region algorithms makes it difficult to implement and extend to the existing RL algorithms. By contrast, our proposed approach is highly flexible and easy to implement on many algorithms in nature.

Generalization and Recurrent Models in RL. Among different approaches that address the generalization issues in RL [19–21], Meta-RL is one of the predominant approaches that aims to learn a good inductive bias of policy that can be quickly generalized to previously unseen tasks [22, 23]. In the meta-training phase, several tasks $\mathcal{D}_{train} = \{D_{(k)}\}_{k=1}^{K}$ are sampled from a task distribution $\mathcal{D}_{meta}$. In the meta-testing phase, $\mathcal{D}_{test} = \{D_{(k)}\}_{k=K+1}^{N}$ are sampled from the same task distribution. Although the meta-optimization approaches [28, 29] have been successfully applied to various image classification tasks, their performance is relatively limited in RL tasks [30]. The context approach [31] learns a latent representation of the task and construct a context model through recurrent networks [32, 33]. In this work, we follow Fakoor et al. [30] to use the simplest form of meta-training, *i.e.*, the multi-task objective: $\hat{\theta}_{meta} = \arg\max_\theta \frac{1}{n}\sum_{k=1}^{n}\mathbb{E}[\ell^{(k)}(\theta)]$, where $\ell^{(k)}(\theta)$ denotes the objective evaluated on the k-th task $D_{(k)}$. Besides the pursuance of generalization, recurrent models are widely used in solving partial-observable MDPs [34] or applied as memory mechanism [35, 36]. Differently, our work introduces recurrent models to solve safe RL tasks in an off-policy approach.

3 Related Work

Learning RL policy under safety constraints [37–39] becomes an important topic in RL community due to the safety concerns of RL in real-world applications. For example, Richter et al. [40] applies RL to the simulated surgical robot. Kendall et al. [41] implements RL in the autonomous driving scenario. In those applications, the safety of the learned policy is critical and the policy should be optimized under some safety constraints. The common practice for this problem is to involve human interventions [25, 26] or correction of the output action [27, 42] .

Most of previous works are based on on-policy RL: CPO proposed by Achiam et al. [4] provides an analytical solution to solve CMDP through trust region optimizations [5] yet this dependence makes it difficult to implement and extend to other existing RL algorithms. Our approach based on ET-MDP, on the contrary, is highly flexible and can be implemented on top of various algorithms. Another straightforward approach to the soft constraint problem is the Lagrangian method, which relaxes the hard-constrained optimization problem to an unconstrained one with an auxiliary penalty term. It was found in Ray et al. [39] that the approximation errors in CPO prevent it from fully satisfying the constraint, and Lagrangian method can find constraint-satisfying policies that attain limited returns. In Sun et al. [43], an early-termination technique is applied with on-policy methods for diverse policy generation.

There exist several works that apply off-policy learning to Safe-RL: Yang et al. [24] extended the Lagrangian method on top of SAC and further improved over it through worse-case analysis; Bharadhwaj et al. [15] equips SAC with a safety-aware critic; Urpí et al. [44] focus on the safety learning in offline settings. Yet all of those methods do not taken the non-Markovian nature of Safe-RL problems into consideration. Note that while previous works have discussed the effectiveness of applying early termination [45] and absorbing state [46] in constrained RL to further improve the

performance of their proposed methods [47], we show in our work such an early-termination can be formulated in a more principled way, and work in isolation as an minimalist alternative approach for Safe-RL. Table 1 summarizes the differences between MOPA and related works.

4 Method

In this work, we propose to handle constraints in safe RL tasks with a minimalist approach: *an early termination is triggered whenever the learning policy violates the constraints.* Such an early termination is previously used as a trick to improve the sample efficiency of solving regular MDPs [48]: terminating bad trajectories accelerates the learning process since the policy space to search is reduced and the time horizon is shortened. Moreover, we do not need to learn to proceed or recover after violations—an *ideal* policy should *never* break the constraints.

We first introduce two types of constraints in CMDPs in Section 4.1. We will show that the early-termination trick used in locomotion tasks is indeed an intuitive approach for solving CMDP with, what we call, loose constraints. In Section 4.2, we define ET-MDP as the foundation of our proposed method. ET-MDP enables the algorithms previously designed for MDP to solve CMDP tasks. We provide discussion on some practical issues and introduce our algorithm to solve ET-MDP efficiently in Section 4.3.

4.1 Constraint Types

To show the relationship between normal MDPs and CMDPs as well as better illustrate the inspiration that links CMDPs with early termination, we first unify CMDP and MDP formulation in the loose-constrained cases: MDPs can be regarded as loose-constrained CMDPs when the constraints do not change their optimal solution. Thus those CMDPs can be solved by the same policy trained from its early termination counterparts. We then extend similar idea to the other case where the constraints are tight. We start with the definition of the learning objective. If we denote a policy set satisfying the constraints C as

$$\Pi^c = \{\text{any policy } \pi : \sum_{t=1}^{H} c(s_t, \pi(s_t)) \leq C\}, \tag{4}$$

then the learning objective of Eqn.(1) becomes $\max_{\pi \in \Pi^c} \mathbb{E}_{\tau\sim\pi,\mathcal{T}}[\sum_{t=1}^{H} r_t]$. The two types of constraints differ in whether the optimal policy lies in the constrained policy class (Eqn.(4)) or not.

Loose Constraints In model-free RL, early termination is often used as a default environment setting to accelerate learning [49, 48]. In such problems, early termination is usually applied when the agent reaches some undesired state, e.g., the center of mass get lower than some certain threshold. We call this kind of constraints loose ones, because the solution of the CMDP is the same as the MDP without constraints: $\pi^* = \arg\max_{\pi\in\Pi} \mathbb{E}_{\tau\sim\pi,\mathcal{T}}[\sum_{t=1}^{H} r_t] \in \Pi^c$.

In such CMDPs, considering the constraints or not won't change the final policy as the optimal solution will learn to not break the constraints automatically. Those loose constraints are shown to be able to accelerate learning in [50].

Tight Constraints In other cases such as navigation in a space with barriers or lava, the barriers or lava can be regarded as constraints and will clearly change the optimal solution to navigate to the goal point compared with the environment of an empty space where there is no constraint applied: $\pi^* = \arg\max_{\pi\in\Pi} \mathbb{E}_{\tau\sim\pi,\mathcal{T}}[\sum_{t=1}^{H} r_t] \notin \Pi^c$.

In such CMDPs, learning to solve the MDP without the constraints can not lead to a satisfying policy for the CMDP as feasible behaviors of the agent must take the constraints into consideration.

Based on such insights, we investigate the approach to solve the CMDPs with their early-termination (ET) counterparts, namely the ET-MDPs. We show the major challenge may come from the limited state visitation problem, which will be further illustrated in detail in Section 4.3.

4.2 Early Terminated MDP (ET-MDP)

For any CMDP $(\mathcal{S}, \mathcal{A}, H, r, c, C, \mathcal{T})$, its ET-MDP is defined as a new unconstrained MDP $(\mathcal{S} \cup \{s_e\}, \mathcal{A}, H, r', \mathcal{T}')$, where s_e is the absorbing state after termination. Generally speaking, ET-MDP has a history-dependent transition dynamic. In order to have a regular MDP, one can introduce an extra dimension to state space recording the cumulative costs denoted by $b_t = \sum_{\tau=1}^{t} c_\tau$. Though b_t takes values from a large set, the transition dynamic that involves in b_t is known to the agent: $\mathcal{T}'(s, b, a) = \mathcal{T}(s, a)\mathbb{1}(b \leq C) + \mathbb{1}(s = s_e, b > C)$. The reward function becomes $r'(s, b, a) = r(s, a)\mathbb{1}(b \leq C) + r_e\mathbb{1}(b > C)$ for some $r_e \in \mathbb{R}$. Since we are searching for a policy for the original CMDP, we still consider policies that are stationary with respect to b, *i.e.* $\pi(s, b) \equiv \pi(s)$.

Proposition 4.1. *For sufficient small r_e, the optimal policy of ET-MDP coincidences with π^* of the original CMDP. (Proof is given by Appendix)*

Proposition 4.1 indicates that CMDP can be solved with their ET-MDP correspondence as long as the termination reward r_e is small enough, which can be easily implemented in practice. Intuitively, as an early-terminated episode is shorter than the original one, one should be able to save samples by solving ET-MDP. In the following part, we show the benefits of solving CMDP through its ET-MDP. Now that we consider a special case, where $c(s, a) = \mathbb{1}(\mathcal{T}(s, a) \in \mathcal{S}_c)$ for some $\mathcal{S}_c \subset \mathcal{S}$. Here the violation is caused by the entrance to some invalid states. We also assume that $\mathcal{S}_c$ is an absorbing class. This is an important case we consider in our experiments, as exploring invalid space is unnecessary and early termination can save samples.

To fairly show the benefits, we introduce a performance measure called regret, the difference between the total rewards of the optimal policy and the rewards received by the running algorithm $\mathcal{L}$:

$$R_T(\mathcal{L}) = \sum_{k=1}^{\lfloor T/H \rfloor} (V^c_{\pi^*} - V^c_{\pi_k}),$$

where V^c_π is the expected value function under policy π and π_k is the policy chosen for episode k. Regrets for deterministic MDPs can be lower and upper bounded.

Theorem 4.2 (Theorem 3 in [51])**.** *Any reinforcement learning algorithm $\mathcal{L}$ that takes a state space, an action space, a horizon as inputs there exists an MDP, such that the regret $\sup_T R_T(\mathcal{L}) \geq 2H|\mathcal{S}||\mathcal{A}|$. There exists an algorithm $\mathcal{L}$, such that for any MDP, the regret $\sup_T R_T(\mathcal{L}) \leq 2H|\mathcal{S}||\mathcal{A}|$.*

The above lower bound applies to the CMDP as one can construct an MDP with a extreme loose constraint such that all the policies are valid. The above upper bound applies to our ET-MDP since in this special case, $c(s, a)$ is either 0 or 1 and the termination happens whenever a cost of 1 is received, which makes it unnecessary to record the cumulative cost and our ET-MDP is a regular MDP with state space $(\mathcal{S} \setminus \mathcal{S}_c) \cup \{s_e\}$.

Corollary 4.3. *There exists a algorithm $\mathcal{L}_{ET}$ for ET-MDP such that for any algorithm $\mathcal{L}_c$ for the original CMDP, the ratio*

$$\frac{\sup_T R_T(\mathcal{L}_c)}{\sup_T R_T(\mathcal{L}_{ET})} \geq \frac{|\mathcal{S}|}{|\mathcal{S}| - |\mathcal{S}_c| + 1}.$$

Remark 4.4 (ET-MDPs reduce sample complexity)**.** *The above analysis ignored the fact that ET-MDP does not have to finish the whole H steps for each episode, which means that when an algorithm is actually running, the above dependence on H can be also decreased depending on the actual cutoffs.*

Corollary 4.3 shows that for tasks with a large invalid space, solving ET-MDP is more efficient. It provides the insight to apply similar methods to more complicated tasks like CMDPs for continuous control. Resembling the key insight where early termination trick is applied to the loose constrained tasks such as the MuJoCo locomotion suite [48], we don't need to collect samples from infeasible regions. Therefore, we are motivated to further investigate whether solving the ET-MDP can be a practically effective way to solve CMDPs.

4.3 Solving ET-MDP with Context Models

In the previous section, we have shown CMDPs can be solved with their ET-MDP counterparts. The next step is to build-up suitable solvers for those ET-MDP tasks. As the framework of ET-MDP is

covered by normal MDPs as there are no constraints that should be taken into consideration, any off-the-shelf algorithm can be applied as an ET-MDP solver, including on-policy methods [5, 52], off-policy methods [18, 53], and Evolution approaches [54].

However, there are also two desiderata a solver should have to be more capable of solving ET-MDP: (1) Non-Markovianity: some budget tasks are Non-Markovian by nature. (2) Generalization challenge: solving ET-MDP requires better generalization.

Markovian Binary Tasks In general, there are two different empirical settings in CMDP. The first is the case where the safety is considered to be extremely important that the constraints should never be broken in the deployment time of a learned policy. We call this kind of setting the binary CMDPs, where the binary indicates classifying a trajectory as safe or not safe. This is the relatively simple case and the constraints in Eqn.(1) can be simplified as $\sum_{t=0}^{\infty} c_t \leq 0$, where $c_t = c > 0$ if the constraints are broken and $c_t = 0$ otherwise. Besides, no more effort is needed to ensure the decision making process Markovian.

For example, navigation in a grid-world with lava belongs to such a setting. Another example is the MuJoCo locomotion tasks, where a hopper, walker or humanoid simulator is required to move forward as fast as possible, and through the whole time the agent should never fall down. In summary, this setting can be applied as long as the task can be accomplished without stepping into any state with a cost.

Non-Markovian Budget Tasks On the other hand, there are cases where there is a *budget* of behavior costs. Behaviors with some cost is not preferable but is permitted to some extent. Henceforth, to satisfy the constraints in Eqn.(1), the historical information of cumulative cost should be taken into consideration in making every-step decision. To achieve this, the primal state space $\mathcal{S}$ *should* be extended to $\mathcal{S}_{\text{ext}} = \mathcal{S} \oplus \mathcal{S}_{\text{budget}} \oplus \mathcal{S}_{\text{time}}$, where $\mathcal{S}_{\text{budget}}$ indicates the budget left in the episode, and $\mathcal{S}_{\text{time}}$ provides information on the number of time steps left in the episode [55].

Previous works fall in this setting include the Safety-Gym [39] and the PointGather [4], where the budget is a fixed positive integer that indicates how many times the agent is permitted to be at states with costs during an episode.

Generalization Challenge Since ET-MDP is a special kind of MDP where lots of termination states $\mathcal{S}_{\text{end}}$ are introduced, algorithms designed for normal MDP tasks are easy to get trapped in limited states due to early-terminations. Similar results has been shown in Agarwal et al. [17] that learning under limited state visitation will hinder the learning efficiency.

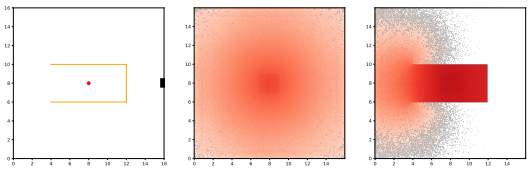

Figure 1: The difference in state visitation frequency of MDPs and ET-MDPs in a diagnostic 2-D navigation environment. **Left**: the environment, an agent starts from the central red point in each episode and yellow lines denote lava, i.e., danger zone; **Middle**: the state visitation frequency of a random agent in a MDP by ignoring the lava; **Right**: the state visitation frequency of a random agent with lava in a ET-MDP. The limited state visitation in ET-MDP is one of the major challenges for off-the-shelf MDP solvers.

Figure 1 illustrates the difference in state visitation frequency of normal MDP (middle) and ET-MDP (right) under random exploration in a 2-D navigation environment, where the central red point denotes the starting point and the constraints are shown as yellow boundaries in the left figure. As all of the constraint-violation states will lead to termination in ET-MDP, the generalization ability of the learned policy becomes extremely important. Intuitively, learning algorithms that can generalize better to previously unseen states will be more competent in such tasks.

To solve the non-Markovianity, a natural choice is to build policies based on recurrent models, whereas the solution to the generalization challenge also lies in recurrent models—namely the context models developed in Meta-RL literature [30]. Our key insight can be introduced through the Syllogism below:

Syllogism 4.5 (Context Models Improve Generalization)**.**
Major premise: *Given a task distribution $\mathcal{M}$, a context models trained with m training tasks*

$\mathcal{M}_{\text{train}} = \{M_1, ..., M_m\}$ *improves learning efficiency on held-out test tasks* $\mathcal{M}_{\text{test}} = \{M_{m+1}, ...\}$ *due to better generalization.*
Minor premise: *An MDP task* $M = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \rho_0, r)$ *can be decomposed as multiple sub-tasks according to initial state decomposition* $\rho_0 = \cup_{i=1}^{n} \rho_0^i$*, such that a training task sampled from the task distribution can be expressed as* $M = \mathcal{M}_{\text{train}} \cup \mathcal{M}_{\text{test}}$*, where* $\mathcal{M}_{\text{train}} \equiv \{(\mathcal{S}, \mathcal{A}, \mathcal{T}, \rho_0^i, r)\}_{i=1}^{m<n}$*, and* $\mathcal{M}_{\text{test}} \equiv \{(\mathcal{S}, \mathcal{A}, \mathcal{T}, \rho_0^i, r)\}_{i=m}^{n}$*.*
Conclusion: *Context model can improve the learning efficiency of MDP task* M *during learning, as it generalizes learned policies to rarely visited states, which corresponds to test tasks that initialize with such states.*

In a nutshell, context models in Meta-RL are shown to learn generalizable representations between a series of similar *tasks*. Thus we regard solving the ET-MDP task (e.g., avoid getting terminated and collect as many reward as possible) with different *states* as different tasks *in the same task distribution*, the context models should be able to learn transferable representations over different *states*, and generalize learned policies to previously unseen states to avoid being terminated.

We use the Gated Recurrent Units [33] to model context variables as generalizable representations to solve ET-MDPs and build the context model based on TD3 [18]. We adopt separated context networks for training stability, $i.e.$, we use $\mathcal{C}_{w_a}$ for actor and $\mathcal{C}_{w_c}$ for critic, such that both the actor π and critic Q take an additional context variable as input:

$$\pi = \pi(s, z_a), Q = Q(s, a, z_c), \tag{5}$$

where $z_a = \mathcal{C}_{w_a}(\mathcal{Z}'_L)$, $z_c = \mathcal{C}_{w_c}(\mathcal{Z}'_L)$ and $\mathcal{Z}'_L$ is the previous L step historical transitions: $\mathcal{Z}'_L = \{s_{t-L}, a_{t-L}, r_{t-L}, ..., s_{t-1}, a_{t-1}, r_{t-1}\}$. If $t - L \leq 0$, we use zero state $\mathbf{0}_s$, zero action $\mathbf{0}_a$ and zero reward $\mathbf{0}_r$ instead.

The context models $(\mathcal{C}_{w_a}, \mathcal{C}_{w_c})$ are optimized through the gradient chain rule in the optimization of actor and critic networks, with the gradient of

$$\begin{aligned} \nabla_{w_a} \mathcal{C}_{w_a} &= \nabla_a Q_{w_1}(s, a, z_c)|_{a=\pi_\theta(s,z_a)} \cdot \nabla_{z_a} \pi_\theta(s, z_a)|_{z_a=\mathcal{C}_{w_a}(\mathcal{Z}'_L)} \nabla_{w_a} \mathcal{C}_{w_a}(\mathcal{Z}'_L), \\ \nabla_{w_c} \mathcal{C}_{w_c} &= \nabla_{w_c} \mathbf{TD}(Q(s, a, \mathcal{C}_{w_c}(\mathcal{Z}'_L))) \end{aligned} \tag{6}$$

separately, where **TD** denotes the temporal difference error. Details of the proposed algorithm are provided in Algorithm 1 in Appendix B. We term our method as **MOPA**, acronym of "a **M**inimalist **O**ff-**P**olicy **A**pproach (to safe RL)". In the next section we will demonstrate the superiority of MOPA in solving a variety of ET-MDPs.

5 Experiments

We evaluate the proposed method on a diverse set of environments, including (1) loose constrained tasks of **Hopper-Not-Fall**, **Walker-Not-Fall**, **Humanoid-Not-Fall**; (2) constrained navigation tasks of **DangerZone** with different degrees of difficulty; (3) stochastic navigation tasks of **PointGoal1-v0**, **CarGoal1-v0**, and **PointGather**. The first two sets of environments are binary tasks while the last three tasks are budget tasks. Examples of environments are shown in Figure 5 in Appendix D.

We validate the following claims in our experiments:

1. CMDPs can be solved by solving their ET-MDP counterparts. MOPA improves learning efficiency and asymptotic performance on all tightly-constrained experiments (Section 5.1).
2. While directly applying the standard RL algorithms like TD3 faces the problem of generalization, the context model mitigates the problem and improves the sample efficiency (Section 5.3).
3. ET is crucial for efficient learning in loose-constrained tasks. MOPA is able to further improve the performance on those tasks (Section 5.2).

more empirical studies are provided in Appendix F including hyper-parameter stress-testing, network structure selection and ablation studies. [2]

[2]Code: https://github.com/holarissun/MOPA

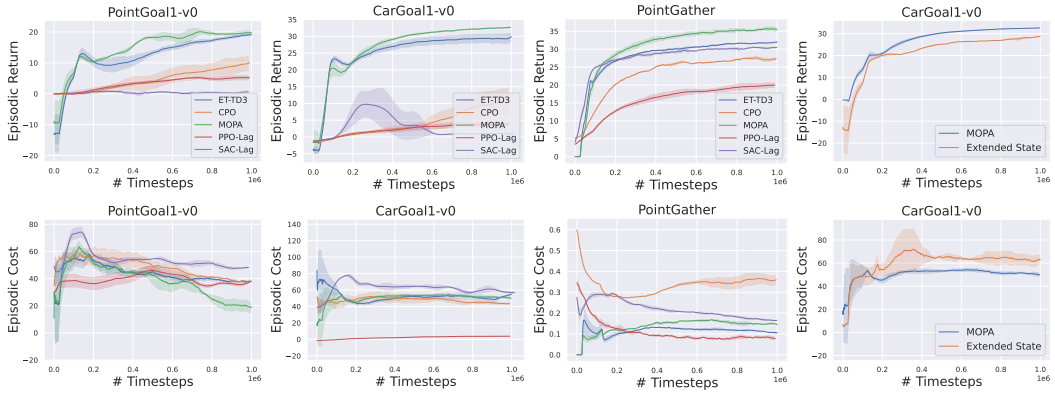

Figure 2: Results on the three budget tasks. The first three columns show the rewards and the costs of different methods on the three environments respectively, while the last column shows the performance comparison between learning with extended state space and tightened approximation. Results are over 10 repeated runs. Shaded areas in figures indicate the 25%-75% quantile values of the results.

5.1 Tight Constraints

In this section, we evaluate MOPA in a variety of environments where constraints change the optimal solution. As termed in the previous section, those are tight constrained problems.

Binary Tasks We first experiment on the DangerZone task where an agent needs to navigate in a maze for a goal point without stepping into the lava. The input of the agent is the coordinate of current state, and permitted action is limited to $[-1, 1]$. We generate four different level of tasks. In all experiments the size of the maze is set to be 16×16, and episodic length is set to be 32, which is two times of the side length. In each episode, the agent is initialized in the center of the maze. Stepping into the target position will result in a $+30$ reward, and stay in the position will continuously gain that reward. A tiny punishment of -0.1 is applied for every timestep, otherwise. More details of the environments can be found at Appendix D.

Table 2: Success rate of different methods on the DangerZone environment.

Success Rate	Easy	Medium	Hard
TD3	2/10	4/10	2/10
CPO	4/10	3/10	0/10
PPO-Lag	3/10	3/10	1/10
SAC-Lag	**10/10**	0/10	0/10
MOPA	**10/10**	**9/10**	**8/10**

In those tasks, the constraints are binary as the agent is not permitted to step into dangerous regions during navigation to the target point. Results shown in Figure 3 show the superior performance of MOPA in solving all level of DangerZone tasks. Table 2 provides the success rate of each method in reaching the target point in our repeated experiments. Detailed learning curves can be found at Appendix E.

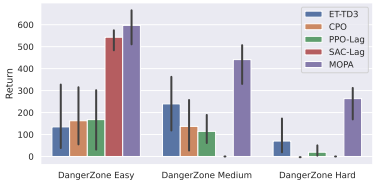

Figure 3: Experiment Results on the DangerZone environment. MOPA is the only method that is able to solve all levels of tasks.

Budget Tasks For the budget tasks, we experiment on PointGoal1-v0, CarGoal1-v0, and PointGather to show the performance of our proposed method. In PointGoal1-v0, a mass point navigates in a 2-D maze to collect reward while avoiding dangerous regions, which will lead to a $+1$ cost. The budget for the cost is $+25$ in the experiments [39]. In CarGoal1-v0, a car replaces the mass point in the previous environment to attain the same objective and the budget is increased to $+50$ as the task is more challenging [56, 15]. In PointGather, a mass point collects apples while avoiding bombs which will lead to a $+1$ cost, and the cost budget is set to 0.1 [4], i.e., the agent is permitted to run into a bomb every ten games on average.

As we have shown in Section 4.3, the previous information of cost should be taken as an additional input for policies to satisfy the Markov property in those environments. By contrast, MOPA equipped with recurrent models is able to address this issue without further modifications.

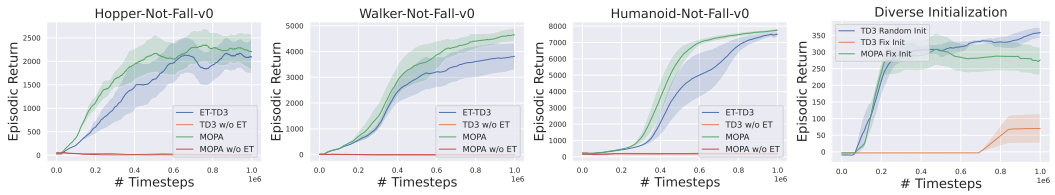

Figure 4: The first three figures show learning curves of TD3 and Context TD3 with/without early termination trick in three locomotion tasks; The last figure shows that context model can remarkably improve learning efficiency when the state visitation is limited. Results are over 10 repeated runs. Shaded areas in figures indicate standard deviation of the results.

Figure 2 shows our experiment results with the binary approximation. In all experiments, MOPA is able to reach the best asymptotic performance in terms of both high reward and low cost. We also experiment with the car navigation environment to compare the performance of the primal CMDP with extended state space $\mathcal{S}_{\text{ext}} = \mathcal{S} \oplus \mathcal{S}_{\text{budget}} \oplus \mathcal{S}_{\text{time}}$ and with the MOPA. The last column of Figure 2 shows the experimental results we get: MOPA outperforms the approximation by achieving better constraints-satisfaction, while being able to collect higher reward.

5.2 Loose Constraints

We evaluate our method on the MuJoCo locomotion benchmarks where early termination (ET) is previous applied as a default setting to benefit learning. This experiment shows that loose constraints like *center of mass higher than a certain threshold* are crucial for sample-efficient learning as they greatly reduce the state space.

In this set of experiments, we remove the *alive bonus* term in the reward during training, otherwise this term will be always the same (i.e., 1000 for Hopper, Walker and 5000 for Humanoid) and the rewards between environments with ET and without ET can not be compared fairly. In evaluation, the *alive bonus* term is kept as default settings so that the asymptotic performances can be compared to previous agents trained in the vanilla environments to get a basic sense of what our policies have learned. Results are shown in Figure 4: while both MOPA and TD3 are able to learn locomotion skills when ET is applied, neither of those methods get success without ET. As expected, MOPA outperforms TD3 in terms of learning efficiency in all three environments.

5.3 Diagnostic Study of MOPA on Generalization

We show the superiority of MOPA over vanilla TD3 in the DangerZone environment to demonstrate its ability in generalization (i.e., higher learning efficiency). In DangerZone, the tasks is no doubt an MDP, i.e., the decision of the agent should be made only based on its present state and has no relevance to the historical information. Henceforth, there is no need to maintain a memory mechanism to gain performance improvement. We hence attribute the improvement of MOPA to better generalization ability rather than the memory mechanism proposed in previous works that also leverage recurrent networks in RL.

In this set of experiments, two different environments are generated to compare MOPA and TD3. In the Random-Init environment, the initial position of the agent is uniformly distributed in the map while in the Fix-Init environment the initial position is fixed at the center of the map, which leads to a limited state visitation. The last column of Figure 4 shows our results: MOPA performs much better than vanilla TD3 in the Fix-Init environment, showing the experiences collected in limited region can be better generalized to unseen states when context models are introduced.

We further verify our key insight of Syllogism 4.5 with extensive empirical studies (16 tasks from the DeepMind-Control suite) in Appendix F.2.

6 Conclusion

In this work, we address the Safe-RL tasks in a minimalist approach: different from the previous CMDP formulations where the constraints require ad-hoc algorithm design, we propose an equivalent formulation, namely Early-Terminated MDP, of those Safe-RL tasks that both on-policy methods

and off-policy methods are applicable. We show in our work that solving ET-MDP lead to identical optimal value function as well as optimal policy to the previous CMDP formulation. To better exploit the potential benefit of solving Safe-RL tasks with ET-MDP, we introduce the MOPA based on recurrent models to mitigate the efficiency issues. Experiments conducted in a variety of experiments demonstrates the superiority of our proposed framework and the method.

Limitation and Future Work of this work is mainly on the heuristic choices of r_e. Although in general, it should be hard to have a single choice for all environments that have various reward scales, in our work we find using a uniform choice $r_e = -1$ for all environments works fairly well. Moreover, our empirical studies in Appendix F.1.3 show the performance of MOPA under different values of r_e are stable. In the future work, a more automated way of adjusting the value of such a punishment term can be developed.

References

[1] Gabriel Dulac-Arnold, Daniel Mankowitz, and Todd Hester. Challenges of real-world reinforcement learning. *arXiv preprint arXiv:1904.12901*, 2019.

[2] Richard S Sutton, Andrew G Barto, et al. *Introduction to reinforcement learning*, volume 2. MIT press Cambridge, 1998.

[3] Eitan Altman. *Constrained Markov decision processes*, volume 7. CRC Press, 1999.

[4] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 22–31. JMLR. org, 2017.

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

[6] Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research*, 18(1):6070–6120, 2017.

[7] Andrew Taylor, Andrew Singletary, Yisong Yue, and Aaron Ames. Learning for safety-critical control with control barrier functions. In *Learning for Dynamics and Control*, pages 708–717. PMLR, 2020.

[8] Yongshuai Liu, Jiaxin Ding, and Xin Liu. Ipo: Interior-point policy optimization under constraints. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4940–4947, 2020.

[9] Richard Cheng, Gábor Orosz, Richard M Murray, and Joel W Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3387–3395, 2019.

[10] Theodore J Perkins and Andrew G Barto. Lyapunov design for safe reinforcement learning. *Journal of Machine Learning Research*, 3(Dec):803–832, 2002.

[11] Yinlam Chow, Ofir Nachum, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning. In *Advances in neural information processing systems*, pages 8092–8101, 2018.

[12] Harshit Sikchi, Wenxuan Zhou, and David Held. Lyapunov barrier policy optimization. *NeurIPS Deep Reinforcement Learning Workshop 2020*, 2020.

[13] Yiming Zhang, Quan Vuong, and Keith W Ross. First order constrained optimization in policy space. *arXiv preprint arXiv:2002.06506*, 2020.

[14] Jesse Zhang, Brian Cheung, Chelsea Finn, Sergey Levine, and Dinesh Jayaraman. Cautious adaptation for reinforcement learning in safety-critical settings. In *International Conference on Machine Learning*, pages 11055–11065. PMLR, 2020.

[15] Homanga Bharadhwaj, Aviral Kumar, Nicholas Rhinehart, Sergey Levine, Florian Shkurti, and Animesh Garg. Conservative safety critics for exploration. *arXiv preprint arXiv:2010.14497*, 2020.

[16] 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.

[17] Alekh Agarwal, Sham M Kakade, Jason D Lee, and Gaurav Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *arXiv preprint arXiv:1908.00261*, 2019.

[18] Scott Fujimoto, Herke Van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. *arXiv preprint arXiv:1802.09477*, 2018.

[19] Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*, pages 1282–1289. PMLR, 2019.

[20] Maximilian Igl, Kamil Ciosek, Yingzhen Li, Sebastian Tschiatschek, Cheng Zhang, Sam Devlin, and Katja Hofmann. Generalization in reinforcement learning with selective noise injection and information bottleneck. *arXiv preprint arXiv:1910.12911*, 2019.

[21] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, pages 1094–1100. PMLR, 2020.

[22] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Learning to generalize: Meta-learning for domain generalization. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[23] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *arXiv preprint arXiv:2004.05439*, 2020.

[24] Qisong Yang, Thiago D Simão, Simon H Tindemans, and Matthijs TJ Spaan. Wcsac: Worst-case soft actor critic for safety-constrained reinforcement learning. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence. AAAI Press, online*, 2021.

[25] William Saunders, Girish Sastry, Andreas Stuhlmueller, and Owain Evans. Trial without error: Towards safe reinforcement learning via human intervention. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pages 2067–2069. International Foundation for Autonomous Agents and Multiagent Systems, 2018.

[26] Sehoon Ha, Peng Xu, Zhenyu Tan, Sergey Levine, and Jie Tan. Learning to walk in the real world with minimal human effort. *arXiv preprint arXiv:2002.08550*, 2020.

[27] Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa. Safe exploration in continuous action spaces. *arXiv preprint arXiv:1801.08757*, 2018.

[28] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.

[29] Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

[30] Rasool Fakoor, Pratik Chaudhari, Stefano Soatto, and Alexander J Smola. Meta-q-learning. *arXiv preprint arXiv:1910.00125*, 2019.

[31] Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, pages 5331–5340. PMLR, 2019.

[32] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. 1999.

[33] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[34] Tianwei Ni, Benjamin Eysenbach, and Ruslan Salakhutdinov. Recurrent model-free rl is a strong baseline for many pomdps. *arXiv preprint arXiv:2110.05038*, 2021.

[35] Bram Bakker et al. Reinforcement learning with long short-term memory. In *NIPS*, pages 1475–1482, 2001.

[36] Lingheng Meng, Rob Gorbet, and Dana Kulić. Memory-based deep reinforcement learning for pomdp. *arXiv preprint arXiv:2102.12344*, 2021.

[37] Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

[38] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

[39] Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. *openai*, 2019.

[40] Florian Richter, Ryan K Orosco, and Michael C Yip. Open-sourced reinforcement learning environments for surgical robotics. *arXiv preprint arXiv:1903.02090*, 2019.

[41] Alex Kendall, Jeffrey Hawke, David Janz, Przemyslaw Mazur, Daniele Reda, John-Mark Allen, Vinh-Dieu Lam, Alex Bewley, and Amar Shah. Learning to drive in a day. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8248–8254. IEEE, 2019.

[42] Benjamin Van Niekerk, Andreas Damianou, and Benjamin Rosman. Online constrained model-based reinforcement learning. *arXiv preprint arXiv:2004.03499*, 2020.

[43] Hao Sun, Zhenghao Peng, Bo Dai, Jian Guo, Dahua Lin, and Bolei Zhou. Novel policy seeking with constrained optimization. *arXiv preprint arXiv:2005.10696*, 2020.

[44] Núria Armengol Urpí, Sebastian Curi, and Andreas Krause. Risk-averse offline reinforcement learning. *arXiv preprint arXiv:2102.05371*, 2021.

[45] Perttu Hämäläinen, Juuso Toikka, Amin Babadi, and C Karen Liu. Visualizing movement control optimization landscapes. *arXiv preprint arXiv:1909.07869*, 2019.

[46] Peter Geibel and Fritz Wysotzki. Risk-sensitive reinforcement learning applied to control under constraints. *Journal of Artificial Intelligence Research*, 24:81–108, 2005.

[47] Akifumi Wachi and Yanan Sui. Safe reinforcement learning in constrained markov decision processes. In *International Conference on Machine Learning*, pages 9797–9806. PMLR, 2020.

[48] Tingwu Wang, Xuchan Bao, Ignasi Clavera, Jerrick Hoang, Yeming Wen, Eric Langlois, Shunshi Zhang, Guodong Zhang, Pieter Abbeel, and Jimmy Ba. Benchmarking model-based reinforcement learning. *arXiv preprint arXiv:1907.02057*, 2019.

[49] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, pages 1329–1338, 2016.

[50] Tu-Hoa Pham, Giovanni De Magistris, Don Joven Agravante, Subhajit Chaudhury, Asim Munawar, and Ryuki Tachibana. Constrained exploration and recovery from experience shaping. *arXiv preprint arXiv:1809.08925*, 2018.

[51] Zheng Wen and Benjamin Van Roy. Efficient exploration and value function generalization in deterministic systems. *Advances in Neural Information Processing Systems*, 26:3021–3029, 2013.

[52] Karl W Cobbe, Jacob Hilton, Oleg Klimov, and John Schulman. Phasic policy gradient. In *International Conference on Machine Learning*, pages 2020–2027. PMLR, 2021.

[53] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018.

[54] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

[55] Fabio Pardo, Arash Tavakoli, Vitaly Levdik, and Petar Kormushev. Time limits in reinforcement learning. In *International Conference on Machine Learning*, pages 4045–4054. PMLR, 2018.

[56] Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by pid lagrangian methods. In *International Conference on Machine Learning*, pages 9133–9143. PMLR, 2020.

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

[58] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

A Proof for Proposition1

Proof. The value function of CMDP is defined in the feasible region $\Pi^c = \{\pi \in \Pi^c : \sum_{t=1}^{H} c(s_t, a_t) \leq C, a_t = \pi(s_t), s_{t+1} = \mathcal{T}(s_t, a_t)\}$, where $\pi \in \Pi^c$ and

$$V_c^{\pi}(s) = \sum_{t=1}^{H} r(s_t, a_t), \text{ where } \quad a_t = \pi(s_t), s_{t+1} = \mathcal{T}(s_t, a_t) \tag{7}$$

The learning objective is to find $\pi \in \Pi^c$ such that

$$V_c^*(s) = \max_{\pi \in \Pi^c} V_c^{\pi}(s). \tag{8}$$

The value function for ET-MDP is defined similarly as normal MDP by

$$V_{ET}^{\pi}(s) = \sum_{t=1}^{H} r'(s_t, b_t, a_t), \text{ where } a_t = \pi(s_t), s_{t+1} = \mathcal{T}'(s_t, b_t, a_t), b_{t+1} = b_t + c_t. \tag{9}$$

For any $\pi \in \Pi^c$, the trajectories are the same in the ET-MDP and its counterpart. We have $r'(s_t, b_t, a_t) = r(s_t, a_t)$ for all $t \leq H$. Therefore, we have $V_c^{\pi} = V_{ET}^{\pi}$ for $\pi \in \Pi^c$.

The optimal value function of ET-MDP is defined over its optimal policy

$$V_{ET}^*(s) = \max\{\max_{\pi \in \Pi^c} V_c^{\pi}(s), \max_{\pi \notin \Pi^c} \sum_{t=1}^{h_\pi \leq H} r(s_t, a_t) + r_e\}, \tag{10}$$

where h_π is the step at which the constraint is violated. Therefore, $V_{ET}^*(s) = V_c^*(s)$ for sufficiently small r_e and the optimal state values are achieved for the same optimal policy $\pi^* \in \Pi^c$ □

B Detailed Pseudo-Code of the Proposed Method

Algorithm 1 MOPA for ET-MDP

Initialize critic networks Q_{w_1}, Q_{w_2}, actor network π_θ
Initialize context models $\mathcal{C}_{w_a}, \mathcal{C}_{w_c}$ for the actor and critic networks separately with recurrent networks.
Initialize target networks $w_1' \leftarrow w_1$, $w_2' \leftarrow w_2$, $\theta' \leftarrow \theta$
Initialize replay buffer $\mathcal{B} = \{\}$
Initialize a context queue $\mathcal{Z}_L$ with length L by $\mathcal{Z}_L = [\mathbf{0}_s, \mathbf{0}_a, \mathbf{0}_r] \times L$, maintain a copy $\mathcal{Z}_L' \leftarrow \mathcal{Z}_L$
for $t = 1, 2, ...$ **do**
    Interact with environment and get transition tuple (s, a, r, c, s'), $r \leftarrow r + r_e$ if $c > 0$.
    Update context queue with $\mathcal{Z}_L$, append (s, a, r), and store $(s, a, r, s', \mathcal{Z}_L', \mathcal{Z}_L)$ in $\mathcal{B}$, update $\mathcal{Z}_L' \leftarrow \mathcal{Z}_L$
    Sample a batch of transitions $\{(s, a, r, s', \mathcal{Z}_L', \mathcal{Z}_L)\}$ from $\mathcal{B}$
    Calculate context variable for actor and critic with $z_a = \mathcal{C}_{w_a}(\mathcal{Z}_L')$, $z_c = \mathcal{C}_{w_c}(\mathcal{Z}_L')$, and context variable for calculating the next action and next value $z_a' = \mathcal{C}_{w_a}(\mathcal{Z}_L)$, $z_c' = \mathcal{C}_{w_c}(\mathcal{Z}_L)$
    Calculate perturbed next action by $\tilde{a} \leftarrow \pi_{\theta'}(s', z_a') + \epsilon$, ϵ is sampled from a clipped Gaussian.
    Calculate target critic value y and update critic networks:
    $y \leftarrow r + \gamma \min_{i=1,2} Q_{w_i'}(s', \tilde{a}, z_c')$
    $w_i \leftarrow \arg\min_{w_i} \mathbf{MSE}(y, Q_{w_i}(s, a, z_c))$
    Update w_c, the context model for critic through
    $w_c \leftarrow \arg\min_{w_c} \mathbf{MSE}(y, Q_{w_i}(s, a, \mathcal{C}_{w_c}(\mathcal{Z}_L')))$
    Update θ by the deterministic policy gradient, with learning rate η:
    $\theta \leftarrow \theta - \eta \nabla_a Q_{w_1}(s, a, z_c)|_{a=\pi_\theta(s, z_a)} \nabla_\theta \pi_\theta(s, z_a)$
    Update w_a, the context model for actor according to Eqn.(6)
    Update target networks, with $\tau \in (0, 1)$:
    $w_i' \leftarrow \tau w_i + (1 - \tau) w_i'$; $\theta' \leftarrow \tau\theta + (1 - \tau)\theta'$
    Break this episode if constraint is broken.
end for

C Reproduction Checklist

Network Structure Our implementation of Context TD3 is mainly based on the code of [18]. The hyper-parameters of TD3 are the same as the authors recommend in the paper. In our Context TD3, we also use 3-layer MLPs for both actor and critic networks (with 256 hidden units).

We find in our experiments using separated context networks that trained through gradients of actor and critic will benefit learning. Details of ablation study on the network structure are provided in Appendix F.3

Value of r_e In our analysis, the value of r_e can be selected as any sufficiently small number. However selecting too small value may lead to over-conservative behavior. In our experiments reported in the main text, we find in experiments that $r_e = -1$ works fairly well. Ablation studies on the selection r_e are provided in Appendix F.1.3.

Batch Size In our experiments we follow Fujimoto et al. [18] to use a mini-batch size of 256. In PPO, CPO and PPO-Largrangian, we use a batch size of 1000 and mini-batch size of 256 for the short-horizon games (e.g., Maze, PointGather, both with $T \leq 32$), so that there are around 1000 episodes in training. For the long-horizon games where $T \sim 1000$, we collect 10 trajectories for each episode for better training stability [5, 57].

Hardware and Training Time We experiment on a server with 8 TITAN X GPUs and 32 Intel(R) E5-2640 CPUs. Experiments take 0.5 (the maze environment with 0.1M interactions) to 10 hours (the safety-gym with 1M interactions) to run. The training of Context TD3 will introduce higher computation expense as additional context models need to be trained.

D Environments Details

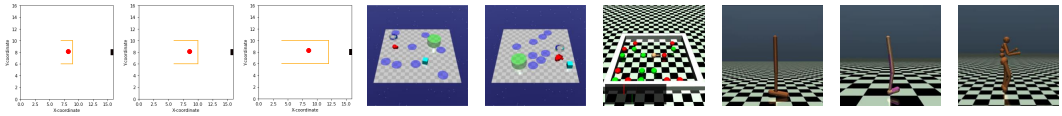

Figure 5: Examples of the tested environments: The first three figures show the DangerZone tasks with different level (different mazes); the following three figures show the budget tasks where agents control a point or a car to collect reward without hitting cost regions too many times; the last three figures show loose-constrained tasks where agents need to learn to move forward without falling.

D.1 DangerZone Environment

In the DangerZone environment, an agent needs to navigate in a maze for a goal point without stepping into the lava. The input of the agent is the coordinate of current state, and permitted action is limited to $[-1, 1]$. We generate four different level of tasks. In all experiments the size of the maze is set to be 16×16, and episodic length is set to be 32, which is two times of the side length. In each episode, the agent is initialized in the center of the maze. Stepping into the target position will result in a $+30$ reward, and stay in the position will continuously gain that reward. A tiny punishment of -0.1 is applied for every timestep, otherwise.

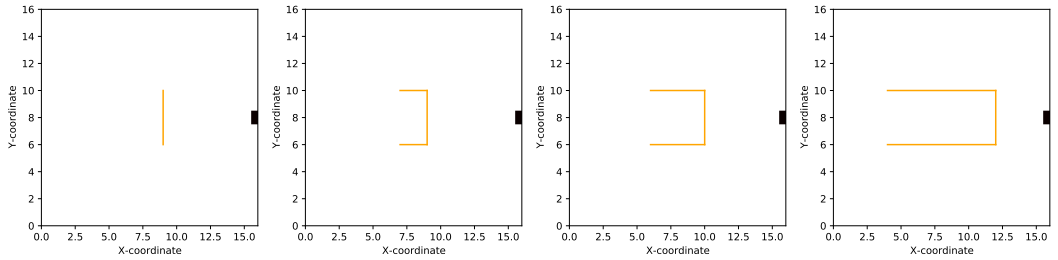

Figure 6: The four DangerZone environments with different levels in our experiments. From left to right: Maze-Level-1 (DangerZone Easy), Maze-Level-2 (DangerZone Medium), Maze-Level-3, Maze-level-4 (DangerZone Hard). The regions with orange color are dangerous region where the agent should not step into. For each game, the agent is initialized at center of the map, therefore the difficulty of finding a solution without violating the constraints becomes harder and harder from Level-1 to Level-4.

E Missing Learning Curves

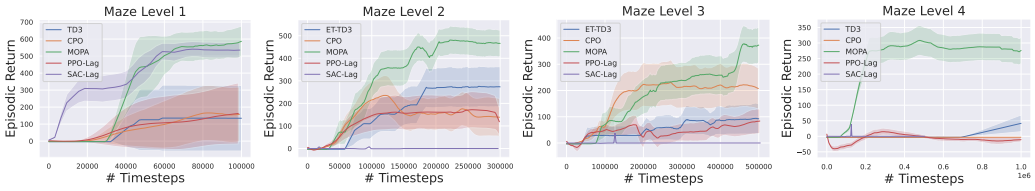

Figure 7: Learning curves of the Maze environments. As the constraints are binary, any reward gained when the constraints are violated is not taken into consideration. Thereafter, only episodic return curves are shown in the figures. i.e., All of those rewards are gained *without breaking any constraint*.

F Additional Empirical Studies

F.1 Sensitivity to Hyper-Parameter

F.1.1 Value of Historical Horizon

We experiment on the DangerZone environments to show how the proposed method work with different length of historical horizon in the context model. Results are shown in Figure 8. **Context** 1 means we only include the past state, action, reward in the computation of context variables, while **Context** 7 indicates the past 7 steps of transitions are leveraged in generating the context variables. We find the context model with historical horizon 3 achieve fairly well performance in all levels of environments.

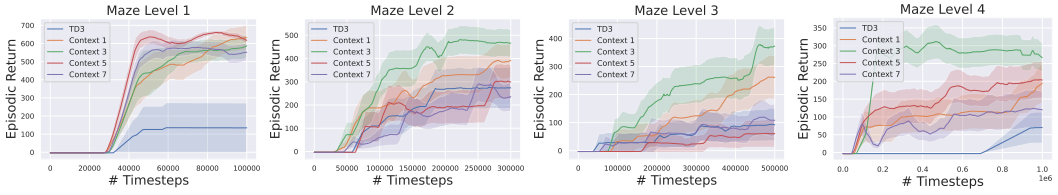

Figure 8: Ablation studies on the selection of different length of historical horizon. All corresponding costs are zero and omitted under our ET-MDP settings.

F.1.2 Number of Hidden Units in GRUs

We experiment on the selection of different number of hidden units used in GRUs. We compare the results with 30 hidden units (reported in the main text, denoted as **MOPA** in Figure 9) with the results with 120 hidden units (denoted as **MOPA-Large** in Figure 9)). We find using 30 hidden units is enough to achieve improved performance, and in the same time balance the computational cost. And using too much hidden units may lead to reduction on learning efficiency (in the Humanoid-Not-Fall-v0 environment).

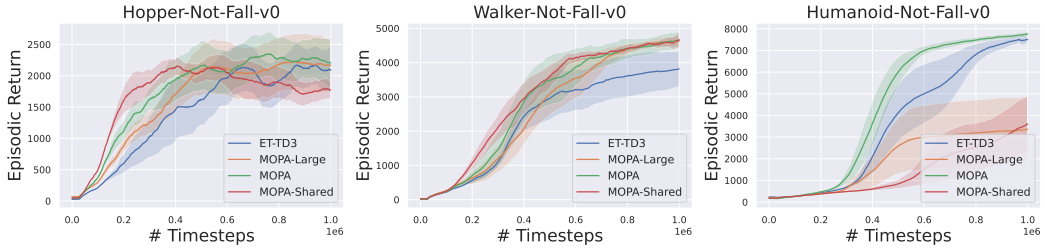

Figure 9: Ablation studies on the number of hidden units used in GRU, and comparison on different selection of network structure: shared v.s. separated context model.

F.1.3 Value of r_e

We show experimental results on the selection of different value of the ending reward r_e in this section. Figure 10 shows the results on the CarGoal, PointGoal and PointGather environments. In both TD3 and Context TD3 working in ETMDP, smaller r_e's result in more conservative policies that achieve lower cost and lower primal task reward.

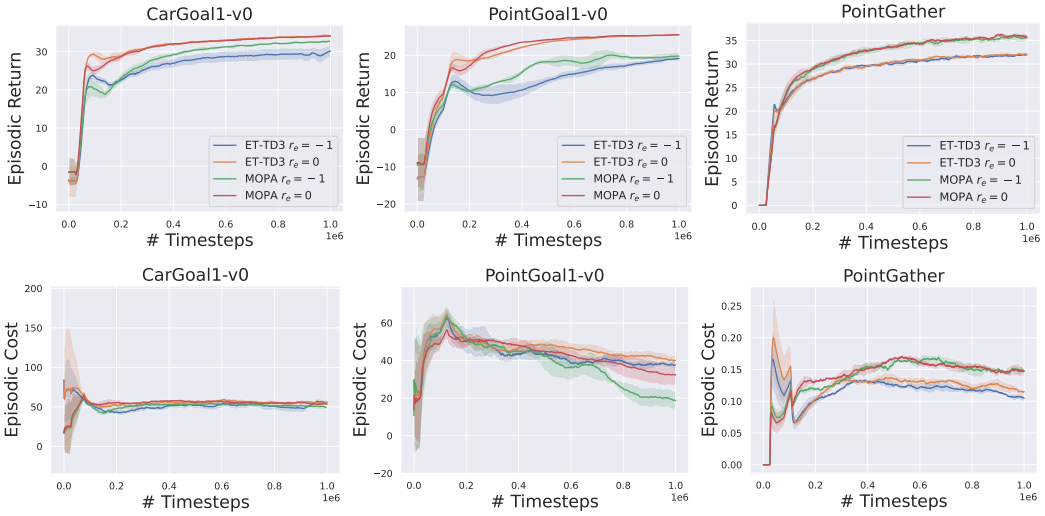

Figure 10: Ablation studies on the selection of r_e, the value of absorbing reward. The first line shows the episodic return curves of each methods in different environments while the second line shows the corresponding episodic costs. Using smaller r_e will lead to more conservative behavior, i.e., slightly lower return and lower cost.

F.2 More Experiments

F.2.1 Validation of Syllogism 4.5 on Other Benchmarks

In this section we show our experiments on various MuJoCo and DeepMind Control benchmarks to show the superiority of the Context TD3 over the vanilla TD3 in sample-efficient learning in MDP

tasks. Figure 11 shows the experiment results. In most environments, Context TD3 achieves better asymptotic performance while being able to converge faster. We use the same hyper-parameter of historical horizon $= 7$ in all experiments. Elaborated searching for hyper-parameters may result in even better performance.

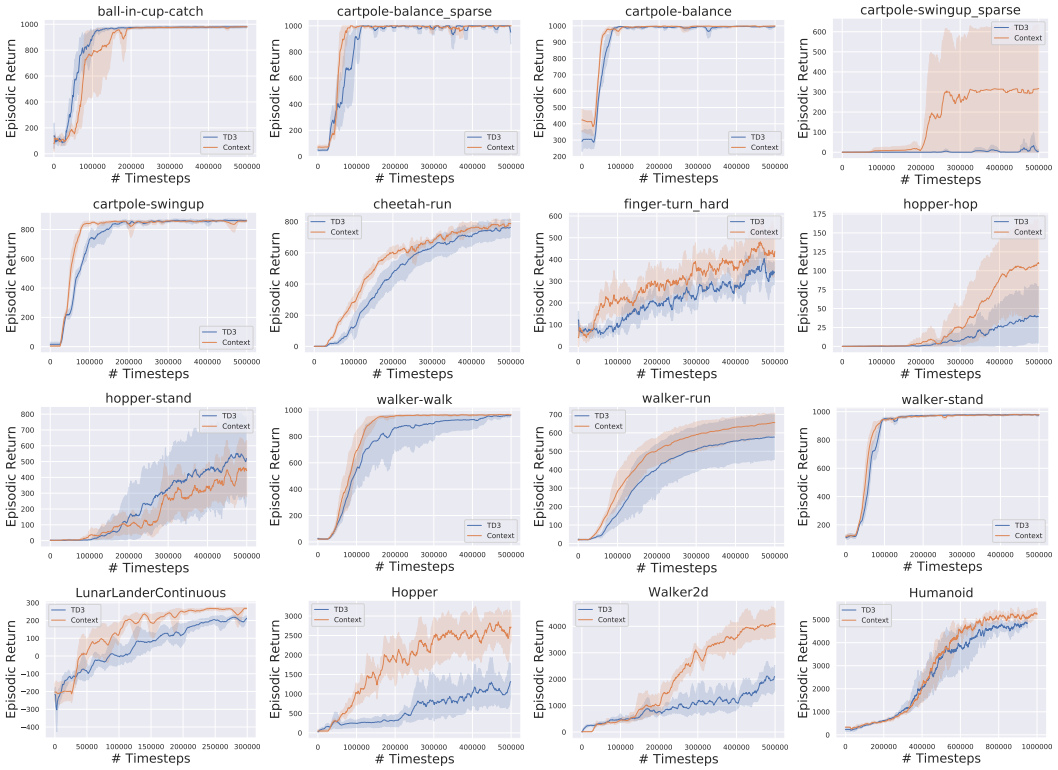

Figure 11: Experiment results on the DeepMind Control Suite. In all 16 benchmark environments we experimented on, context models outperforms normal TD3 in most environments (13 out of 16), showing the superiority of context models in improving learning efficiency.

F.3 Model Structure

F.3.1 GRU v.s. Transformer

In this section we provide ablation studies on the choice of context models: we compare the results of Context models based on GRUs and based on recent advances of self-attention based models [58]. The results are shown in Figure 12, where we find leveraging the transformer models can not result in better performance.

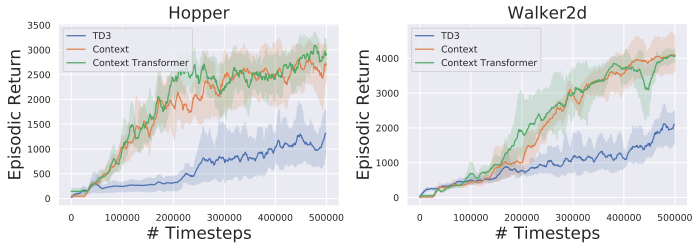

Figure 12: Ablation studies on the selection of context models.

F.3.2 Shared v.s. Separated Context Variables

In the work of [30], the context model is trained only through the learning of critic networks. Differently, in our experiments we find training context models separately for the actor and critic can result in better performance. **MOPA-Shared** in Figure 9 denotes the results when the context model is shared by actor and critic as recommended in the Meta-RL literature [30].

G Qualitative Results

We also include demo videos in the supplemental materials. Where the performance of agents trained with different algorithms in the PointGoal1-v0 and CarGoal1-v0 safe-navigation tasks are shown qualitatively.