TRACES: TRAJECTORY BASED CREDIT ASSIGNMENT FROM SPARSE SAFETY FEEDBACK

Siow Meng Low & Akshat Kumar
School of Computing and Information Systems
Singapore Management University
Singapore
smlow.2020@phdcs.smu.edu.sg,akshatkumar@smu.edu.sg

ABSTRACT

In safe reinforcement learning (RL), auxiliary safety costs are used to align the agent to safe decision making. In practice, safety constraints, including cost functions and budgets, are unknown or hard to specify, as it requires anticipation of all possible unsafe behaviors. We therefore address a general setting where the true safety definition is unknown, and has to be learned from sparsely labeled data. Our key contributions are: *first*, we design a safety model that performs *credit assignment* to estimate each decision step's impact on the overall safety using a dataset of diverse trajectories and their corresponding *binary* safety labels (i.e., whether the corresponding trajectory is safe/unsafe). *Second*, we illustrate the architecture of our safety model to demonstrate its ability to learn a separate safety score for each timestep. *Third*, we reformulate the safe RL problem using the proposed safety model and derive an effective algorithm to optimize a safe yet rewarding policy. *Finally*, our empirical results corroborate our findings and show that this approach is effective in satisfying unknown safety definition, and scalable to various continuous control tasks.

1 INTRODUCTION

The Markov Decision Process (MDP) is a widely used framework to model and reason about the uncertainty of sequential decision making (Puterman, 1994). MDPs are the underlying model for reinforcement learning (RL) (Sutton et al., 1998), and have a wide range of applications in robotics (LaValle, 2006), path planning (Kiran et al., 2021), energy management (Zhang et al., 2019) among others. In complex real-world systems, the decision maker is often faced with optimizing multiple objectives, e.g., in traffic control, the goal is to optimize traffic signal timings to reduce congestion and travel times while ensuring safety and compliance with traffic regulations. Such problems are naturally modeled as constrained MDPs (Altman, 1998) where we optimize a primary objective subject to constraint. This enforces the agent to avoid undesirable behaviors.

Constraints require specification of cost values for each state-action pair. Typically, such cost functions are either assumed to be known in the planning context (Altman, 1998), or the environment simulator returns numerical costs during RL (Achiam et al., 2017; Tessler et al., 2018; Chow et al., 2018; Stooke et al., 2020; Ha et al., 2021). However, in several real world settings, information about precise cost function and budget may be lacking. This may be due to the incomplete MDP model of the environments (Saisubramanian et al., 2021), which is often the case for complex real world settings, or the inherent difficulty in explicitly specifying the constraints in a mathematical form (Malik et al., 2021) involving cost and budget. This is similar to the challenges encountered in reward specification design (Ng et al., 2000; Ray et al., 2019). In Chou et al. (2019; 2020), the authors illustrate the challenges in exhaustively programming all possible constraints. To address these challenges, our work targets a setting where both the cost function and tolerance threshold (i.e. budget) are *unknown*; only a trajectory dataset is provided, containing state-action trajectories with associated *binary* labels (constraint satisfied, not satisfied). Next, we review prior work in this area.

Related work In standard constrained RL, contemporary work focuses on modeling constraint costs as a function of state and action (Achiam et al., 2017; Tessler et al., 2018; Chow et al., 2018;

Stooke et al., 2020; Ha et al., 2021). As costs are unknown in our setting, such methods cannot be directly applied without learning the constraint. For *unknown* constraint setting, inverse constrained RL (ICRL) framework is proposed (Chou et al., 2019; Malik et al., 2021; Liu et al., 2023). In ICRL, demonstration dataset from an expert is required that contains both *safe* and *reward-maximizing* trajectories. Collecting such an expert dataset is challenging; in practical settings, demonstration dataset may contain both sub-optimal (reward-wise) and unsafe trajectories.

Other related works focus on learning the cost function from human feedback (Saisubramanian et al., 2022; 2021; Zhang et al., 2018; Chirra et al., 2024). Most of these approaches introduce the concept of *negative side effects* (NSEs), which are undesirable, unintended side effects that arise when agents operate based on incomplete model specifications. To avoid such NSEs in the future, a dataset from human (or expert) feedback is collected that specifies whether a given state-action pair incurs NSE or not. This dataset is then used for cost function learning. A key limitation of this approach is requirement of *finegrained* safety feedback at the state level. Such feedback may not be readily available for an unknown constraint which is difficulty to specify. The RLSF (Chirra et al., 2024) method attempts to address this by learning to assign safety credit to individual states based on trajectory-level safety feedback. This reduces the difficulty in dataset collection. However, RLSF still requires *full knowledge* of how many safety constraints need to be enforced, and the safety budget for each such constraint. Other works, involving human feedback, include inferring implicit human preferences from initial state configuration (Shah & Krasheninnikov, 2019), and nudging agent in preserving flexibility metrics such as reachability measure or attainable utility (Krakovna et al., 2018; Turner et al., 2020a;b).

Contributions We address key limitations of previous works in unknown constraint settings. We assume only binary safety labels are associated with trajectories, without any knowledge of cost functions, safety budget, or the number of constraints. Our safety model assigns safety credit score to state-action pairs at each time step despite sparse safety data. The dataset contains diverse and possibly sub-optimal trajectories and their associated safety labels.

More precisely, *first*, we design a constraint model and associated loss function to learn the unknown constraint from the (sparsely) labeled safety dataset. Our constraint model specifically performs *credit assignment* to assess contributions of each state-action pair on constraint satisfaction. This safety credit assignment is important to facilitate safe RL. *Second*, we approximate and reformulate the safe RL problem using the safety credit assignment module. *Third*, we devise a safe RL method which optimizes total reward while respecting the learned constraint. This method is an extension to PPO-Lagrangian (Schulman et al., 2017; Tessler et al., 2018), and we name it TraCeS – trajectory based credit assignment for safe RL. *Lastly*, our empirical results on a variety of continuous control domains demonstrate the applicability of our method to a wide range of RL problems with unknown constraint, with performance close to the oracle PPO-Lagrangian method which has *full* knowledge of the underlying true constraints.

2 Background Information

2.1 Constrained MDPs

Markov Decision Processes (MDPs) (Sutton et al., 1998) are widely used to model sequential decision making problem under uncertainty. Constrained reinforcement learning (CRL) methods utilize an extended MDP model, called Constrained Markov Decision Processes (CMDPs) (Altman, 1998). CMDPs allow specifying secondary objectives (e.g., safety, risk, resource budget) as constraints in the problem formulation; CRL algorithms seek policy which optimizes reward while respecting the specified constraints.

A CMDP is defined by tuple $(S, A, \mathcal{T}, R, C, b, \gamma)$. We consider a general setting with continuous state and action spaces ($S \subseteq \mathbb{R}^n$, $A \subseteq \mathbb{R}^m$). The environment state transition is characterized by the function $p(s_{t+1}|s_t, a_t) = \mathcal{T}(s_t, a_t, s_{t+1})$. The reward function $R : S \times A \to \mathbb{R}$ maps a state, action to a scalar reward value. Similarly, a constraint function $C : S \times A \to \mathbb{R}$ maps a state, action to a scalar cost value and b is the associated bound on this cost. Lastly, $0 \leq \gamma \leq 1$ is the discount rate; $\tau = \langle s_0, a_0, s_1, a_1, ..., s_{T-1}, a_{T-1} \rangle$ denotes the state-action sequence for a trajectory of arbitrary length T.

A CRL problem solves for a policy π (parameterized by θ) which maximizes the expected sum of rewards subject to the specified constraint:

$$\begin{aligned} &\max_{\theta} J_R(\pi_\theta) = \mathbb{E}_{\tau\sim\pi_\theta}[\sum_{t=0}^{\infty}\gamma^t R(s_t,a_t)] \\ &\text{s.t.} \quad J_c(\pi_\theta) = \mathbb{E}_{\tau\sim\pi_\theta}[\sum_{t=0}^{\infty}\gamma^t C(s_t,a_t)] \leq b \end{aligned} \tag{1}$$

In this paper, we assume both the cost function $C(\cdot,\cdot)$ and budget b are unknown and must be learned. This is different from RLSF (Chirra et al., 2024) where the budget b is assumed to be known and the learned cost is assumed to be binary. Our method TraCeS includes two constraint models to learn this information pertaining to safety constraints.

2.2 Safety Feedback Dataset

Many contemporary works propose to learn the cost function from human expert giving state-level safety feedback (i.e., whether each (s,a) pair is safe/unsafe). This type of data can be prohibitively expensive to collect, and therefore our safety feedback dataset is only sparsely labeled. A trajectory $\tau_{0:t}$ of length $t+1$ is provided with a single binary label, 0 or 1. We denote the true labeling function as $\Psi(\cdot)$. In our problem setting, the (unknown) true labeling function $\Psi(\tau)$ inspects the total true cost incurred in τ. If the total cost incurred is above the ground-truth budget, it outputs 0, i.e. $\Psi(\tau)=0$. Otherwise, label 1 is given, i.e. $\Psi(\tau)=1$. Note that this true labeling function is not known and we are only provided with the trajectory data sparsely labeled by the true labeling function. When multiple constraints are to be enforced, $\Psi(\tau)=1$ indicates satisfaction of all the constraints.

The safety labeling function is unknown in general and has to be learned from data. A dataset of labeled (possibly partial) trajectories $\{(\tau^i_{t_1:t_2},\psi^i)\}$ with safety label $\psi^i\in\{0,1\}$ is used to facilitate learning of such a safety constraint model. Each trajectory τ^i can be of variable length and the labeled trajectories need not be expert trajectories, and some can be unsafe or safe. This is different from some prior works where demonstration trajectories from an expert are provided.

In practical implementation, the observed trajectories during training are stored in a buffer, and the safety constraint models are continually retrained using the safety label data for some selected trajectories from this buffer. In our experiments, a script implementing the true labeling function is used to label the trajectories. We highlight again that this true labeling criterion is *unknown* to constrained models or TraCeS.

3 Safety Constraint Models

This paper focuses on CRL problem in equation 1 where the safety constraint is specified using (unknown) state-action costs $C(s,a)$ and total budget b. Given binary-labeled trajectories $\tau_{0:t}$ of variable length t in the safety feedback dataset, safety constraint models learn to approximate the safety constraint. The learned information will then be used to facilitate safe RL. We introduce two such models next. To aid our discussion, we adopt the following definition.

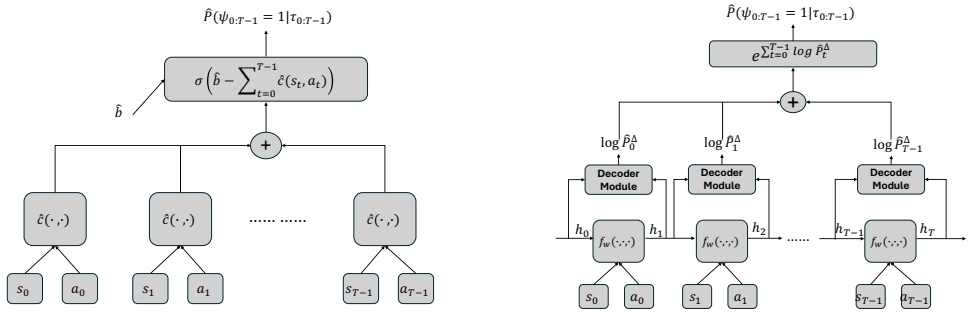
Definition 3.1. For a trajectory τ of length t, the probability estimate $\hat{P}(\psi_{0:t-1}=1|\tau_{0:t-1})$ from the constraint model, is defined as the estimated probability of trajectory $\tau_{0:t-1}$ being safe.

3.1 Cost & Budget Model

The first model is the cost & budget model (C&B Model), which is a trajectory classification model. Its model architecture and reformulated problem are described in appendix B. We use this as a baseline safety model before detailing our main credit assignment based safety model.

3.2 Safety Summary Vector Model

While C&B model provides a straightforward way to estimate costs and budget, it can be tricky to have an accurate estimate as there could be many possible combinations of $\hat{b}$ and $\hat{c}(s,a)$ which



(a) Cost & Budget Model

(b) Safety Summary Vector Model

Figure 1: Safety Model Architectures: (a) Cost & Budget model; (b) Safety Summary Vector Model

minimize the binary cross entropy loss (Bishop & Nasrabadi, 2006; Goodfellow et al., 2016). Say a constant k is added to $\hat{b}$, the estimated probability $\sigma(\hat{b} - \sum_{t=0}^{T_i-1} \hat{c}(s_t, a_t))$, will be unchanged if the same constant k is added to $\sum_{t=0}^{T_i-1} \hat{c}(s_t, a_t)$. Note that $\sigma(\cdot)$ is the sigmoid function used in C&B model.

For this reason, we propose a more general model named *Safety Summary Vector* (SSV) model, which can handle sparsely-labeled trajectory data more effectively. First, we observe that instead of predicting a single probability estimate for the entire trajectory, we can decompose it into a series of learnable probability scores, one per timestep.

3.2.1 Safety Credit Assignment

Recall from Definition 3.1 that for a trajectory of length T, $\hat{P}(\psi_{0:t-1} = 1|\tau_{0:t-1})$ refers to the estimated probability of entire trajectory being safe. From this definition, we can define T different probability estimate, one for each sub-segment within the trajectory $\tau_{0:t-1}$.

Definition 3.2. For a trajectory $\tau_{0:t-1}$ of length t, we define T probability estimates $\hat{P}(\psi_{0:t'-1} = 1|\tau_{0:t'-1}), 1 \leq t' \leq t$. Each of these refers to the probability estimate that a subsegment contained within $\tau_{0:t-1}$ is safe.

Safety Credit Assignment With the T different probability estimates in Definition 3.2, the objective of safety credit assignment is to assign *safety credit* to each timestep (we denote it as $\hat{P}_t^{\Delta}$, a learned per-timestep multiplicative safety score). Intuitively, this safety credit can act as pseudo-cost when we perform safe RL, and it is available for each time step, which allows us to apply standard safe RL methods. We next illustrate how such credit assignment safety variable can be learned.

Proposition 3.3. *A probability estimate $\hat{P}(\psi_{0:T-1} = 1|\tau_{0:T-1})$ can be broken down into a series of T different score estimates $\hat{P}_t^{\Delta}, 0 \leq t \leq T-1$ such that:*

$$\hat{P}(\psi_{0:T-1} = 1|\tau_{0:T-1}) \triangleq \prod_{t=0}^{T-1} \hat{P}_t^{\Delta} \tag{2}$$

Proof. We represent $\hat{P}(\psi_{0:T-1} = 1|\tau_{0:T-1})$ using the probability estimate of each subsegment:

$$\hat{P}(\psi_{0:T-1} = 1|\tau_{0:T-1}) = \frac{\hat{P}(\psi_{0:0} = 1|\tau_{0:0})}{1} \prod_{t=1}^{T-1} \frac{\hat{P}(\psi_{0:t} = 1|\tau_{0:t})}{\hat{P}(\psi_{0:t-1} = 1|\tau_{0:t-1})} \tag{3}$$

In the above ratio-based expression, only the last term $\hat{P}(\psi_{0:T-1} = 1|\tau_{0:T-1})$ remains; rest are canceled out. We then define our safety variable $\hat{P}_t^{\Delta}$.

$$\hat{P}_0^{\Delta} \triangleq \frac{\hat{P}(\psi_{0:0} = 1|\tau_{0:0})}{1} \quad \text{and} \quad \hat{P}_t^{\Delta} \triangleq \frac{\hat{P}(\psi_{0:t} = 1|\tau_{0:t})}{\hat{P}(\psi_{0:t-1} = 1|\tau_{0:t-1})}, 1 \leq t \leq T-1 \tag{4}$$

The overall probability of safety can now be represented using the safety variables:

$$\hat{P}(\psi_{0:T-1} = 1|\tau_{0:T-1}) \triangleq \prod_{t=0}^{T-1} \hat{P}_t^{\Delta} \tag{5}$$

□

The key importance of Proposition 3.3 lies in showing that instead of learning a single $\hat{P}(\psi_{0:T-1} = 1|\tau_{0:T-1})$ for the entire trajectory, we can learn a series of T estimates $\hat{P}_t^{\Delta}, 0 \leq t \leq T-1$ and each $\hat{P}_t^{\Delta}$ is interpreted as the multiplicative changes in the probability of being safe *after* observing state-action (s_t, a_t) at timestep t.

$\hat{P}_t^{\Delta}$ as safety credit score: We next analyze why $\hat{P}_t^{\Delta}$ can be treated as a safety credit score for (s_t, a_t). When $\hat{P}_t^{\Delta} = 1$ (its maximum value) for a time step t, it means the safety probability of segment $\tau_{0:t}$ remains the same as the last segment $\tau_{0:t-1}$. Even though both $\tau_{0:t}$ and $\tau_{0:t-1}$ can be unsafe, given $\hat{P}_t^{\Delta} = 1$ implies that the state-action (s_t, a_t) at time step t in $\tau_{0:t}$ did *not* incur any additional cost over the last time step $t-1$, and can be considered safe even though the entire trajectory $\tau_{0:t}$ might be unsafe. This reasoning also applies when $\hat{P}_t^{\Delta}$ is high (or close to 1); we can consider such (s_t, a_t) as safe in the sense that cost incurred $C(s_t, a_t)$ is low (compared to the remaining budget), thus such (s_t, a_t) have *high safety score* $\hat{P}_t^{\Delta}$. In contrast, when $\hat{P}_t^{\Delta}$ is low (or close to zero), it means $\tau_{0:t}$ is *significantly* more unsafe relative to the previous segment $\tau_{0:t-1}$. Thus, cost incurred in (s_t, a_t) must be high. Thus, we assign *low* safety score to such (s_t, a_t).

Learning $\hat{P}(\psi|\tau)$: To learn $\hat{P}(\psi|\tau)$, we can use standard binary cross entropy loss for classification (Bishop & Nasrabadi, 2006; Goodfellow et al., 2016):

$$\mathbb{E}_{\tau_i \sim \mathbb{D}}[\psi_i \log \hat{P}(\psi_i = 1|\tau_i) + (1 - \psi_i) \log(1 - \hat{P}(\psi_i = 1|\tau_i)] \tag{6}$$

where the probability of τ_i being safe, $\hat{P}(\psi_i = 1|\tau_i)$, is given as in proposition 3.3, and $\mathbb{D}$ refers to the feedback dataset containing $N+1$ safety-labeled trajectories $\{\tau_i\}_{i=0}^{N}$.

3.2.2 Problem Reformulation

A key benefit of the estimating scores $\hat{P}_t^{\Delta}$ is that it can be treated as proxy variable for safety at each timestep. An important difference between $\hat{P}_t^{\Delta}$ and $\hat{c}$ in C&B model is that $\hat{P}_t^{\Delta}$ is the multiplicative changes in the probability of trajectory being safe after observing state-action at timestep t where $\hat{c}$ is a scalar, representing the nominal cost incurred at timestep t. Consequently, we need to restructure the constraint threshold when using $\hat{P}_t^{\Delta}$ to estimate safety. Recall that the true safety labeling function $\Psi(\tau)$ outputs 1 for safe τ and 0 for unsafe τ. We can therefore impose the following safety constraint instead:

$$\mathbb{E}_{\tau \sim \pi_\theta}[\Psi(\tau)] \geq d, \quad 0 \leq d \leq 1 \tag{7}$$

This constraint is imposing that the proportion of safe trajectories among all trajectories τ generated by policy π_θ is at least d (empirically, we set d to a high value 0.9; setting $d = 1$ will make the policy highly conservative). With this constraint, we can then reformulate our CRL program using the probability estimates.

Lemma 3.4. *With the safety scores $\hat{P}_t^{\Delta}$ output by the model, the constraint equation 7 in the CRL program is approximated as: $\mathbb{E}_{\tau \sim \pi_\theta}[\prod_{t=0}^{T-1} \hat{P}_t^{\Delta}] \geq d$.*

Proof. Given the loss function in equation 6, it is a maximum likelihood estimator (Bishop & Nasrabadi, 2006; Goodfellow et al., 2016). As training sample size tends to infinity, the approximation $\mathbb{E}_{\tau \sim \pi_\theta}[\hat{P}(\psi = 1|\tau)] \approx \mathbb{E}_{\tau \sim \pi_\theta}[\Psi(\tau)]$ becomes more and more precise. With equation 5, we can approximate the constraint $\mathbb{E}_{\tau \sim \pi_\theta}[\Psi(\tau)] \approx \mathbb{E}_{\tau \sim \pi_\theta}[\prod_{t=0}^{T-1} \hat{P}_t^{\Delta}] \geq d$. □

Reformulated program The constrained RL program can be reformulated using the safety score $\hat{P}_t^{\Delta}$ output by the model:

$$\begin{aligned} &\max_\theta J_R(\pi_\theta) = \mathbb{E}_{\tau \sim \pi_\theta}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)] \\ &\text{s.t. } \mathbb{E}_{\tau \sim \pi_\theta}[\sum_{t=0}^{\infty} \gamma^t \log \hat{P}_t^{\Delta}] \geq \log d \end{aligned} \tag{8}$$

We now elaborate on the reformulation steps. From Lemma 3.4, the LHS of the constraint is $\mathbb{E}_{\tau_i\sim\pi_\theta}[\prod_{t=0}^{T_i-1}\hat{P}_t^\Delta] = \mathbb{E}_{\tau_i\sim\pi_\theta}[e^{\sum_{t=0}^{T_i-1}\log\hat{P}_t^\Delta}]$. The constraint is thus: $\log\mathbb{E}_{\tau_i\sim\pi_\theta}[e^{\sum_{t=0}^{T_i-1}\log\hat{P}_t^\Delta}] \geq \log d$. This term is hard to estimate directly, since the log is outside the expectation. Therefore, to address this in a principled manner, we derive a tractable lower bound using Jensen's inequality:

$$\begin{aligned}\log\mathbb{E}_{\tau_i\sim\pi_\theta}[e^{\sum_{t=0}^{T_i-1}\log\hat{P}_t^\Delta}] &\geq \mathbb{E}_{\tau_i\sim\pi_\theta}[\log e^{\sum_{t=0}^{T_i-1}\log\hat{P}_t^\Delta}]\\ \text{Let } \mathbb{E}_{\tau_i\sim\pi_\theta}[\log e^{\sum_{t=0}^{T_i-1}\log\hat{P}_t^\Delta}] &\geq \log d\\ \text{Then } \mathbb{E}_{\tau_i\sim\pi_\theta}[\sum_{t=0}^{T_i-1}\log\hat{P}_t^\Delta] &\geq \log d\end{aligned} \tag{9}$$

By constraining the lower bound expression to be greater or equal to $\log d$, the original constraint is also satisfied, i.e. greater or equal to $\log d$. Once trained, the model output $\hat{P}_t^\Delta$ can infer the safety score at arbitrary timestep t even for $t \to \infty$. To make the sum of $\log\hat{P}_t^\Delta$ finite, we apply the same discount rate to this sum:

$$\mathbb{E}_{\tau\sim\pi_\theta}[\sum_{t=0}^{\infty}\gamma^t\log\hat{P}_t^\Delta] \geq \log d \tag{10}$$

With the safety constraint reformulated via the sum of log safety scores in equation 10, the constrained RL program is reformulated to equation 8.

Corollary 3.5. *The program in equation 8 reformulated the constrained RL program in a standard CRL form: objective is to maximize the discounted sum of reward while constraint specifies the discounted sum of safety variables is above a specified threshold* $\log d$*. As a result, standard CRL techniques, such as PPO-Lagrangian, can be employed to solve this CRL program.*

Assumption 3.6. Revisiting the definition $\hat{P}(\psi_{0:T-1}=1|\tau_{0:T-1}) \triangleq \prod_{t=0}^{T-1}\hat{P}_t^\Delta$ in Proposition 3.3. It implies that the multiplication in the RHS, $\prod_{t=0}^{T-1}\hat{P}_t^\Delta$, must remain a probability and be in the range of $[0,1]$. As a result, $0 \leq \hat{P}_t^\Delta \leq 1, \forall t$ and $1 \geq \hat{P}(\psi_{0:0}=1|\tau_{0:0}) \geq \hat{P}(\psi_{0:1}=1|\tau_{0:1}) \geq \cdots \geq \hat{P}(\psi_{0:T-1}=1|\tau_{0:T-1}) \geq 0$.

Remark 3.7. The interpretation of Assumption 3.6 is that the probability of being safe can only stay the same or decrease as more and more timesteps are observed. We argue that this is not restrictive. In most safety-critical setting, an unsafe action in the past cannot be undone by future actions. In the context of standard CRL program which specifies constraint as budget and costs, Assumption 3.6 requires the cost to be non-negative so that future actions cannot produce a negative cost and bring an unsafe trajectory back to safety. We found this assumption not restrictive since the widely-used safe RL simulators (Ji et al., 2023; Gronauer, 2022) all model cost as non-negative.

3.2.3 Safety Model Architecture

Proposition 3.3 tells us that, in order to accurately estimate the safety variable $\hat{P}_t^\Delta$ at timestep t, the pertinent safety features, contained in both trajectory segments $\tau_{0:t-1}$ and $\tau_{0:t}$, have to be included as input. In our proposed safety summary vector (SSV) model, we train a compact summary vector h_{t+1} to encapsulate the safety features contained in trajectory segment $\tau_{0:t}$. Figure 1b illustrates our model architecture: a neural network function approximator ($f_w(\cdot,\cdot,\cdot)$, where w is its list of parameters, summarizes the history and outputs the updated safety summary vector h_{t+1}. The decoder module compares h_t (which contains information pertinent to safety in trajectory segment $\tau_{0:t-1}$) with h_{t+1} (which contains information pertinent to safety in trajectory segment $\tau_{0:t}$) for the purpose of estimating the safety variable $\log\hat{P}_t^\Delta$ at timestep t.

The main reason why we employ this architecture is due to its ability to summarize past information into a compact vector h_{t+1}. In this architecture, the safety proxy variable $\log\hat{P}_t^\Delta$ is a function of h_t and h_{t+1}. As depicted in Figure 1b, h_{t+1} is a function of (s_t, a_t, h_t). The safety variable $\log\hat{P}_t^\Delta$ is thus a function of (s_t, a_t, h_t). In model-free constrained RL method, such as PPO-Lagrangian, a safety critic (Ha et al., 2021) is often required and it needs to estimate the safety variable at timestep t from the observed input (s_t, a_t, h_t). To make our proposed safe RL method highly scalable, a compact summary vector h_t is required as it serves as an augmented state variable. We designed our SSV model architecture to support these.

Further elaborating on the point of augmented state variable, our SSV models $\log\hat{P}_t^\Delta$ as a function of (h_t, h_{t+1}). By extension, $\log\hat{P}_t^\Delta$ depends on (s_t, h_t, a_t). Modification are required before employing PPO-Lagrangian to solve this CRL program: the MDP state is expanded to accommodate

Table 1: Safe RL Performance Comparison (Evaluation Environment) in Safety Gymnasium. Safe policies in bold black color, and best among all safe in blue color; all others in grey. Numbers in () are standard deviations.

Task	Metric	PPO-Lagrangian	RLSF	C&B + CRL (Ours)	SSV + TraCeS (Ours)
PointCircle1	Reward	46.6 (1.4)	**41.9 (1.8)**	40.2 (1.1)	**43.2 (1.5)**
	Cost	21.5 (7.2)	**1.6 (2.2)**	25.3 (15.9)	**16.2 (7.2)**
	Labeled Trajectories	NA	**3244 (966)**	12000 (0)	**5830 (918)**
PointCircle2	Reward	41.7 (0.9)	**39.9 (0.8)**	**40.2 (0.8)**	41.1 (0.9)
	Cost	29.1 (6.00)	**2.6 (4.5)**	**22.7 (13.5)**	19.8 (9.5)
	Labeled Trajectories	NA	**3422 (589)**	**12000 (0)**	5465 (1328)
CarCircle1	Reward	18.3 (0.4)	14.3 (2.2)	18.0 (0.5)	**17.4 (0.9)**
	Cost	23.5 (5.9)	33.8 (41.7)	28.2 (11.7)	**11.3 (8.0)**
	Labeled Trajectories	NA	7479 (645)	12000 (0)	**4005 (7)**
CarCircle2	Reward	16.2 (0.2)	**13.7 (1.1)**	15.3 (0.9)	15.2 (0.9)
	Cost	28.3 (5.2)	**20.2 (17.0)**	44.0 (24.0)	11.9 (10.0)
	Labeled Trajectories	NA	**7601 (597)**	12000 (0)	4341 (243)
Ant	Reward	3313.1 (53.8)	**2220.7 (91.8)**	**3096.7 (63.7)**	**2885.4 (12.3)**
	Cost	14.4 (6.8)	**5.8 (2.6)**	**20.3 (4.1)**	**19.5 (5.3)**
	Labeled Trajectories	NA	**9036 (1289)**	**6000 (0)**	**4657 (551)**
HalfCheetah	Reward	3008.3 (52.9)	**2468.8 (177.0)**	2623.5 (264.6)	**2372.5 (272.6)**
	Cost	2.1 (1.2)	**0.5 (0.9)**	99.7 (74.5)	**22.4 (5.3)**
	Labeled Trajectories	NA	**4349 (793)**	6000 (0)	**2749 (557)**
Hopper	Reward	1046.1 (345.2)	**1577.7 (42.1)**	1620.1 (139.9)	1610.7 (48.9)
	Cost	29.1 (36.2)	**20.4 (44.2)**	284.2 (289.2)	17.1 (4.7)
	Labeled Trajectories	NA	**6690 (736)**	6000 (0)	2993 (484)
Walker2d	Reward	**2682.2 (333.2)**	2797.7 (122.0)	**2491.0** (478.3)	**2211.2 (182.0)**
	Cost	**10.3 (5.5)**	0.3 (0.3)	**16.7 (9.1)**	**24.1 (3.9)**
	Labeled Trajectories	**NA**	15227 (1920)	**6000 (0)**	**5735 (1041)**

h_t at timestep t. Consequently, the policy network $\pi_\theta(a|s,h)$ and the cost-critic network $V_c^\pi(s,h)$ both take s and h as input. We name our method, which learns the safety credit assignment model in figure 1b and safe RL using decomposed scores $\log \hat{P}_t^\Delta$, as TraCeS.

Continual Constraint Model Learning: Before commencing safe RL, the constraint models should reach some level of accuracy in order to provide accurate safety information to TraCeS. We therefore pre-train the constraint models using offline safe RL dataset in DSRL repository (Liu et al., 2024). We wrote a script to provide sparse safety labels to the trajectory segments found in DSRL dataset and pre-train our constraint models using this dataset to around 95% accuracy.

Another challenge in online RL is that as policy gets updated during training, the distribution of the observed trajectories also changes. To remain accurate, the constraint models need to be retrained with fresh set of trajectory data. We describe our continual learning strategy using coefficient of variation (CV) method in appendix C.

4 Empirical Results

Our experiments target to investigate the following: (1) Do the decomposed safety proxy variables from our constraint models provide sufficient information to guide RL agent in producing safe trajectory? What is the solution quality in terms of reward maximization? (2) How well do the decomposed safety proxy variables correspond to the unknown ground truth? Can it identify critical parts of the trajectory which have the highest impact on safety? (3) Is the CV-based method in Section C.2 effective in reducing the amount of human feedback required yet retain high solution quality?

Experiment Setup: We conducted experiment on 12 control tasks: four MuJoCo (Todorov et al., 2012) tasks, four circle tasks in safety gymnasium (Ji et al., 2023), and four run tasks in bullet safety gym (Gronauer, 2022). For all these tasks, we consider the true cost and budget to be unknown to the constraint models. The constraint model only learns the constraint from labeled trajectories. Following the setup in Ji et al. (2023; 2024); Gronauer (2022), the ground-truth cost signal is binary (0 or 1) and ground-truth budget is 25 in our experiments. To pretrain our constraint models, we use the offline safe RL dataset from Liu et al. (2024). Sparse safety labels are given to a variable-length trajectory segment: trajectory is only labeled safe if the true cost incurred is above the threshold. Note that the baseline RLSF method is also pretrained with the same dataset.

Baselines: Our constraint models are designed to enable constrained RL when both budget and cost are unknown and have to be learned from sparsely-labeled data. To the best of our knowledge,

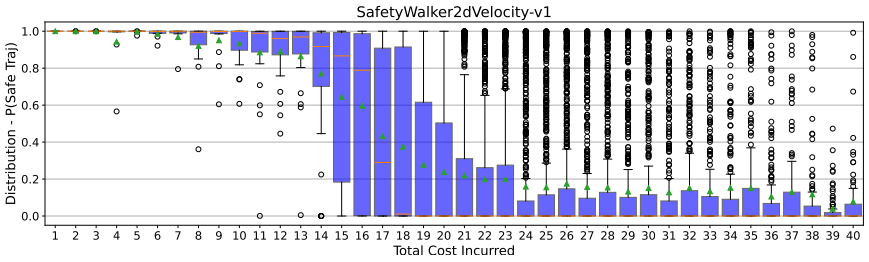


Figure 2: Distribution of Safety Score (Walker2d)

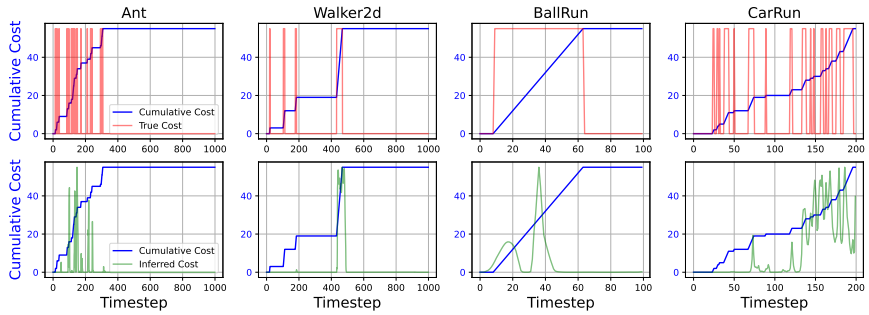


Figure 3: Sample Trajectories from Four Tasks

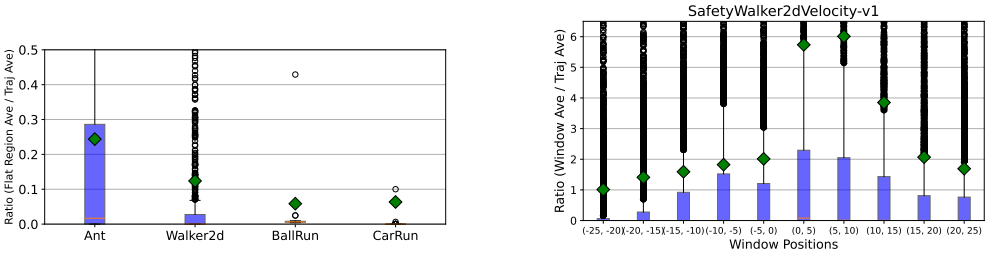
the existing notable work which can serve as baselines include: (1) PPO-Lagrangian, an *oracle* baseline where cost is directly observed and budget is known; (2) RLSF (Chirra et al., 2024), cost is learned from human feedback but budget is *known*. These two baselines are compared with our two proposed methods where both budget and cost are learned: (3) Cost Budget Model; and (4) Safety Summary Vector Model TraCeS. PPO-Lagrangian baseline results are generated using omnisafe framework (Ji et al., 2024). For our methods, the constraint models estimate the safety scores and PPO-Lagrangian is used to perform safe RL. Our implementation are based on the PPO-Lagrangian code in omnisafe framework.

Metrics: We assess the performance using two metrics: total return and total incurred true cost. We report the metrics by evaluating 100 separate test trajectories using the trained policy and constraint models at the end of safe RL training. These test trajectories are generated in another evaluation environment, separate from the training environment. The metrics are reported in the terms of average and standard deviation values across 8 training seeds.

Safe RL Performance: We performed safe RL trainng and evaluated our models against the baseline methods and the evaluation results are tabulated in Table 1, and 2 (in supplement). The total number of learning steps in all safety gymnasium tasks is 10 million steps. For bullet safety gym tasks, BallRun task was trained for 1 million steps while the other three tasks were trained for 2 million steps. The training curves for all these tasks can be found in Figure 5, 6, 7. For our proposed models, SSV method was trained with selective feedback (described in Section C.2) while C&B was trained with higher number of feedback trajectories than the SSV method as the selective feedback of section C.2 is not easily applicable in the C&B model.

From the final evaluation performance, we can observe that while C&B model achieves good quality solution in some tasks, it struggles to produce safe policy in a number of tasks, including two CarCircle tasks, HalfCheetah, Hopper, AntRun and CarRun tasks. In comparison, the policy trained using our SSV model is safe across all twelve tasks, with total return close to PPO-Lagrangian. We reiterate that PPO-Lagrangian requires cost and budget to be known and RLSF assumes known budget. SSV model does not have knowledge of both cost and budget, and its solution quality is close to PPO-Lagrangian and RLSF, requiring fewer labeled trajectories than RLSF in several domains.

Credit Assignment: The RL performance result shows that our SSV model aids safe RL by decomposing safety signal to the critical timestep. This subsection aims to investigate this safety credit assignment aspect further. The y-axes in the boxplots of Figure 2 (and figures 8, 9, 10 in sup-



(a) Ratio of Average Logscore within "Flat Region" to Overall Average Logscore

(b) Ratio of Window Average Score over Overall Average Score around Critical Region (Walker2d)

Figure 4: Credit Assignment within Flat Region and around Critical Region

plement) show the estimated probability of trajectory being safe while x-axis is the total cost in a trajectory. Trajectories with lower cost incurred (less than 15) generally have higher estimated probability of being safe. When the total incurred cost crosses beyond the ground-truth budget, i.e. 25, the estimated probability of safety drops to a low number. These figures demonstrate what the model learns from labeled trajectory data: trajectory is estimated to be unsafe once it crosses the ground-truth budget. This is expected as annotators label trajectories based on this threshold.

We next discuss how the inferred safety score is attributed to different timesteps within a trajectory. Figure 3 depicts four sample trajectories from different tasks. The red curve is the ground-truth 1-0 cost and blue curve is the accumulated cost at different timestep. In the second row of figure, we plot the normalized inferred cost from the SSV model, i.e. logscore at timestep t divided by minimum logscore in the trajectory. The main purpose of this green curve is to inspect its safety credit assignment behavior compared to the true cost. It can be observed from the figure that the inferred cost typically peaks when the accumulated cost reaches 25, the ground-truth budget, and there are other spikes closely following the true non-zero cost at other timesteps. In the "flat region" of the blue curve where there is no additional cost incurred for an extended period of time, the SSV model does not attribute cost to this region, which is accurate credit assignment as expected. To ascertain this feature, in Figure 4a we plotted the ratio of average logscore within these flat regions (of cumulative cost, blue curve) to the average logscore in the trajectory. It confirms our observation that the inferred cost in the "flat region" is much lower.

We saw from Figure 3 that most of the inferred cost concentrates around the critical timestep when the accumulated real cost exceeds 25. We plotted Figure 4b (and figures 11, 12, 13 in supplement) to inspect the statistics of inferred cost around this critical timestep. In each of these figures, x-axis refers to the window (with length of 5 timesteps) around the critical timestep and y-axis is the ratio of average logscore within the window to the overall average logscore within the trajectory. The inferred cost signal is typically highest in the next five timesteps immediately following the critical timestep and this inferred cost trails off as timestep increases. Due to the sparse safety feedback mechanism, we do expect some credit assignment to be slightly delayed but these figures confirm that most of the safety credit assignment happens right after the critical timestep.

Trajectory Selection: To assess the effectiveness of our proposed selective feedback strategy (described in Section C.2), we also trained the SSV method using all training trajectories encountered for labeling. The result is in Table 3. Compared to the selective feedback strategy, using all training trajectories for labeling does not seem to produce significantly different result. The total return and ground-truth cost are comparable across all twelve tasks. We therefore conclude that the selective feedback strategy does reduce feedback required while retaining good solution quality.

5 Conclusion

In this work, we study safe RL with unknown safety constraint. We propose two constraint models to model both the unknown cost and budget to enable constrained RL to be carried out. We discuss the unique design of our constraint model to perform credit assignment. The empirical result shows that, with our SSV safety model, safe policy is achieved while maintaining high solution quality with total return close to oracle methods.

REFERENCES

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

Eitan Altman. Constrained markov decision processes with total cost criteria: Lagrangian approach and dual linear program. *Mathematical methods of operations research*, 48(3):387–417, 1998.

Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

Shashank Reddy Chirra, Pradeep Varakantham, and Praveen Paruchuri. Safety through feedback in constrained RL. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=WSsht66fbC.

Glen Chou, Dmitry Berenson, and Necmiye Ozay. Learning constraints from demonstrations, 2019.

Glen Chou, Necmiye Ozay, and Dmitry Berenson. Learning parametric constraints in high dimensions from demonstrations. In *Proceedings of the Conference on Robot Learning*, pp. 1211–1230, 2020.

Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *Journal of Machine Learning Research*, 18(167): 1–51, 2018.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

Sven Gronauer. Bullet-safety-gym: A framework for constrained reinforcement learning. Technical report, mediaTUM, 2022.

Sehoon Ha, Peng Xu, Zhenyu Tan, Sergey Levine, and Jie Tan. Learning to walk in the real world with minimal human effort. In *Conference on Robot Learning*, pp. 1110–1120. PMLR, 2021.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. The elements of statistical learning: data mining, inference, and prediction, 2017.

Jiaming Ji, Borong Zhang, Jiayi Zhou, Xuehai Pan, Weidong Huang, Ruiyang Sun, Yiran Geng, Yifan Zhong, Josef Dai, and Yaodong Yang. Safety gymnasium: A unified safe reinforcement learning benchmark. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=WZmlxIuIGR.

Jiaming Ji, Jiayi Zhou, Borong Zhang, Juntao Dai, Xuehai Pan, Ruiyang Sun, Weidong Huang, Yiran Geng, Mickel Liu, and Yaodong Yang. Omnisafe: An infrastructure for accelerating safe reinforcement learning research. *Journal of Machine Learning Research*, 25(285):1–6, 2024. URL http://jmlr.org/papers/v25/23-0681.html.

B Ravi Kiran, Ibrahim Sobh, Victor Talpaert, Patrick Mannion, Ahmad A Al Sallab, Senthil Yogamani, and Patrick Pérez. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 23(6):4909–4926, 2021.

Victoria Krakovna, Laurent Orseau, Ramana Kumar, Miljan Martic, and Shane Legg. Penalizing side effects using stepwise relative reachability. *arXiv preprint arXiv:1806.01186*, 2018.

Steven M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006.

Guiliang Liu, Yudong Luo, Ashish Gaurav, Kasra Rezaee, and Pascal Poupart. Benchmarking constraint inference in inverse reinforcement learning. In *International Conference on Learning Representations*, 2023.

Zuxin Liu, Zijian Guo, Haohong Lin, Yihang Yao, Jiacheng Zhu, Zhepeng Cen, Hanjiang Hu, Wenhao Yu, Tingnan Zhang, Jie Tan, and Ding Zhao. Datasets and benchmarks for offline safe reinforcement learning. *Journal of Data-centric Machine Learning Research*, 2024.

Shehryar Malik, Usman Anwar, Alireza Aghasi, and Ali Ahmed. Inverse constrained reinforcement learning. In *International conference on machine learning*, pp. 7390–7399. PMLR, 2021.

Andrew Y Ng, Stuart Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, pp. 2, 2000.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., 1st edition, 1994.

Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. *arXiv preprint arXiv:1910.01708*, 7(1):2, 2019.

Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2): 123–146, 1995.

Sandhya Saisubramanian, Shlomo Zilberstein, and Ece Kamar. Avoiding negative side effects due to incomplete knowledge of AI systems. *AI Mag.*, 42(4):62–71, 2021.

Sandhya Saisubramanian, Ece Kamar, and Shlomo Zilberstein. Avoiding negative side effects of autonomous systems in the open world. *J. Artif. Intell. Res.*, 74:143–177, 2022.

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

R Shah and D Krasheninnikov. Preferences implicit in the state of the world. In *International Conference on Learning Representations (ICLR)*, 2019.

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

Richard S Sutton, Andrew G Barto, et al. Introduction to reinforcement learning, 1998.

Chen Tessler, Daniel J Mankowitz, and Shie Mannor. Reward constrained policy optimization. In *International Conference on Learning Representations*, 2018.

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*, pp. 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109.

Alex Turner, Neale Ratzlaff, and Prasad Tadepalli. Avoiding side effects in complex environments. *Advances in Neural Information Processing Systems*, 33:21406–21415, 2020a.

Alexander Matt Turner, Dylan Hadfield-Menell, and Prasad Tadepalli. Conservative agency via attainable utility preservation. In *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, pp. 385–391, 2020b.

Shun Zhang, Edmund H Durfee, and Satinder Singh. Minimax-regret querying on side effects for safe optimality in factored markov decision processes. In *IJCAI*, pp. 4867–4873, 2018.

Zidong Zhang, Dongxia Zhang, and Robert C Qiu. Deep reinforcement learning for power system applications: An overview. *CSEE Journal of Power and Energy Systems*, 6(1):213–225, 2019.

A Appendix

B Cost & Budget Model

B.1 Safety Model Architecture

This model aims to learn the cost function $\hat{c}$ and budget $\hat{b}$ using safety-labeled variable-length trajectories in the safety feedback dataset. The probability of a trajectory being safe is computed using the estimated remaining budget $\hat{b} - \sum_{t=0}^{T-1} \hat{c}(s_t, a_t)$ having observed a trajectory τ of length

T. The C&B model uses the sigmoid function to map the estimated remaining budget to estimated safety probability: with $\hat{b}$ being the estimated budget and $\hat{c}(s, a)$ being the estimated cost of executing action a at state s, the estimated probability of a trajectory $\tau_{0:T}$ being feasible is $\hat{P}(\psi_{0:T-1} = 1|\tau_{0:T-1}) = \sigma(\hat{b} - \sum_{t=0}^{T-1} \hat{c}(s_t, a_t))$ where $\sigma(\cdot)$ is the sigmoid function.

Figure 1a depicts the architecture of C&B Model. Given a trajectory $\tau_{0:T-1}$ of length T, the model outputs T estimated costs and a single estimated budget. The model parameters are trained by minimizing the binary cross entropy loss (Bishop & Nasrabadi, 2006; Goodfellow et al., 2016):

B.2 Safe RL Using C&B Model

With the estimates of costs $\hat{c}(\cdot, \cdot)$ and budget $\hat{b}$, we solve for the following reformulated problem:

$$\begin{aligned} &\max_{\theta} J_R(\pi_\theta) = \mathbb{E}_{\tau\sim\pi_\theta}[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)] \\ &\text{s.t.} \quad J_c(\pi_\theta) = \mathbb{E}_{\tau\sim\pi_\theta}[\sum_{t=0}^{\infty} \gamma^t \hat{c}(s_t, a_t)] \leq \hat{b} \end{aligned} \tag{11}$$

With the reformulated problem in Equation equation 11 being in the standard CRL form, we can employ PPO-Lagrangian (Tessler et al., 2018) to solve this program.

C Continual Learning for Safety Model

C.1 Continual Learning Strategy

Since our end goal is to use the constraint models to facilitate online safe RL, the constraint models need to be constantly updated with the in-distribution trajectory data encountered during safe RL training. Without continual learning, the constraint model might not remain accurate and it might provide erroneous guidance while trying to steer the policy toward feasible region.

To prevent catastrophic forgetting (Robins, 1995), the constraint models are retrained using sparsely labeled in-distribution data and rehearsed with old data used in pre-training. This retraining process is executed once sufficient number of training trajectories is collected during safe RL phase.

C.2 Selecting Trajectory for Feedback

To reduce the total number of feedback required for safety labeling, we included an enhancement to the safety summary vector model in 1b. Instead of giving a point estimate $\log \hat{P}_t^{\Delta}$, the decoder module outputs the parameters of a negative lognormal distribution $-\text{Lognormal}(\mu_t, \sigma_t)$ and the actual safety score $\log \hat{P}_t^{\Delta}$ is sampled from the distribution. The estimated log-probability of entire $\log \hat{P}(\psi_{0:T-1} = 1|\tau_{0:T-1})$ is thus the sum of these T random variables.

The rationale behind learning these parameters is to gauge the uncertainty behind the safety score estimate. For a continuous random variable, the coefficient of variation (CV) is typically used to measure its dispersion (Hastie et al., 2017). CV captures the relative uncertainty in the estimated safety score across a trajectory. Note that these T random variables are conditionally independent, conditioning on observing the trajectory $\tau_{0:T-1}$. The mean, variance and CV of the final sum are therefore:

$$\begin{aligned} &\mathbb{E}[\sum_{t=0}^{T-1} X] = \sum_{t=0}^{T-1} \mathbb{E}[X] \\ &\text{Var}[\sum_{t=0}^{T-1} X] = \sum_{t=0}^{T-1} \text{Var}[X] \\ &\text{CV}[\sum_{t=0}^{T-1} X] = \frac{\sqrt{\text{Var}[\sum_{t=0}^{T-1} X]}}{\mathbb{E}[\sum_{t=0}^{T-1} X]} = \frac{\sqrt{\sum_{t=0}^{T-1} \text{Var}[X]}}{\sum_{t=0}^{T-1} \mathbb{E}[X]} \end{aligned} \tag{12}$$

The CV of the final sum of safety scores (for a trajectory segment of length T) can be easily calculated using equation 12 since the decoder module outputs the distribution parameters of the negative lognormal distributions. During continual learning, we calculate the CV value for each training trajectory and only select the training trajectories with high CV for labeling and retraining.

D ADDITIONAL RESULTS

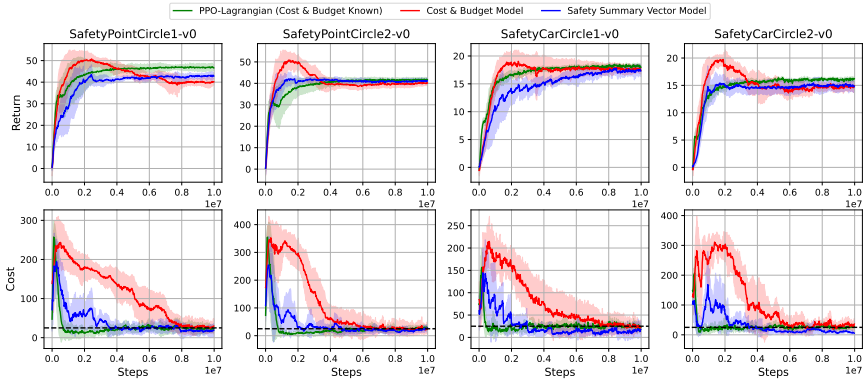


Figure 5: Training Curve for Circle Tasks

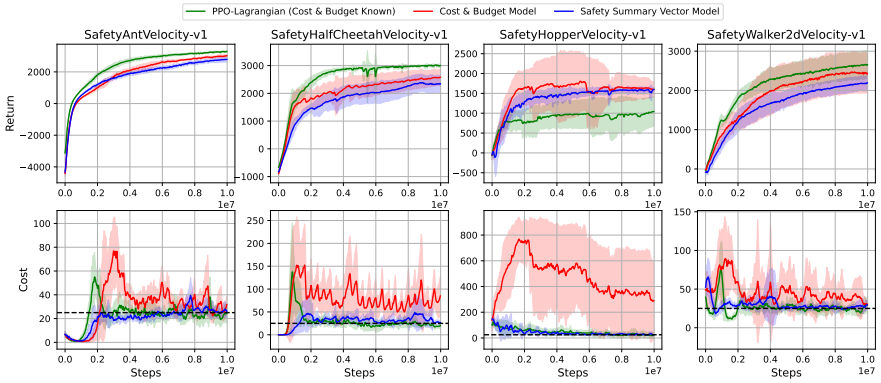


Figure 6: Training Curve for MuJoCo Velocity Tasks

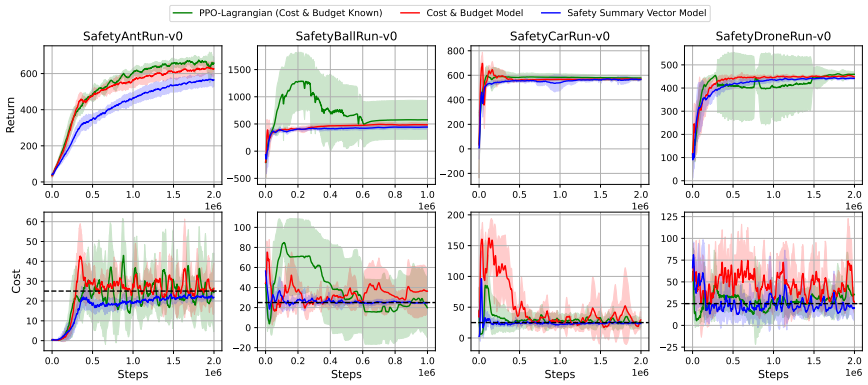


Figure 7: Training Curve for Bullet Run Tasks

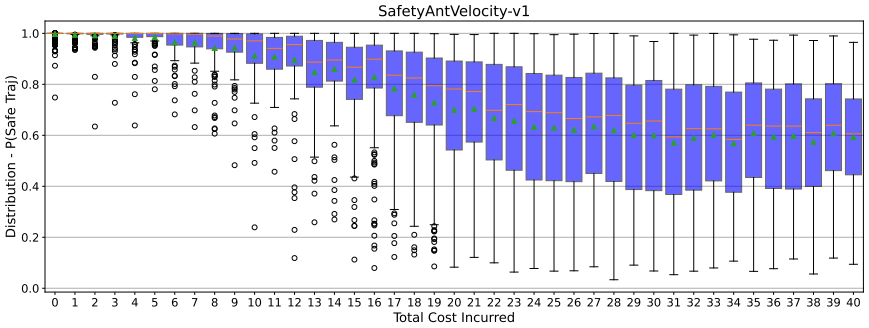


Figure 8: Distribution of Safety Score (Ant)

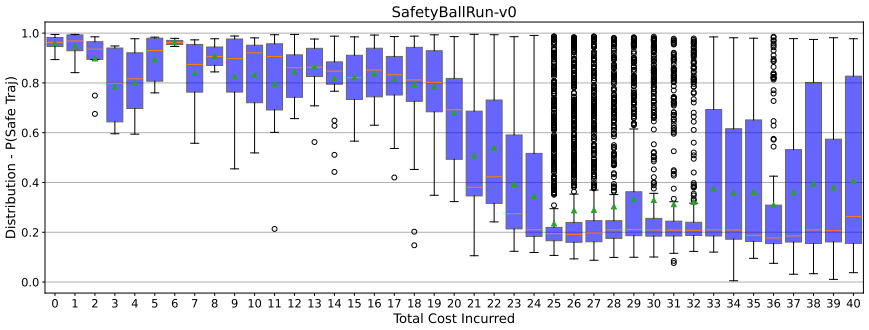


Figure 9: Distribution of Safety Score (BallRun)

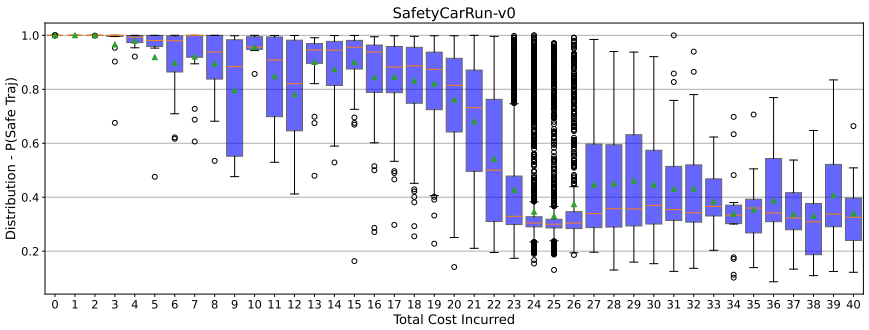


Figure 10: Distribution of Safety Score (CarRun)

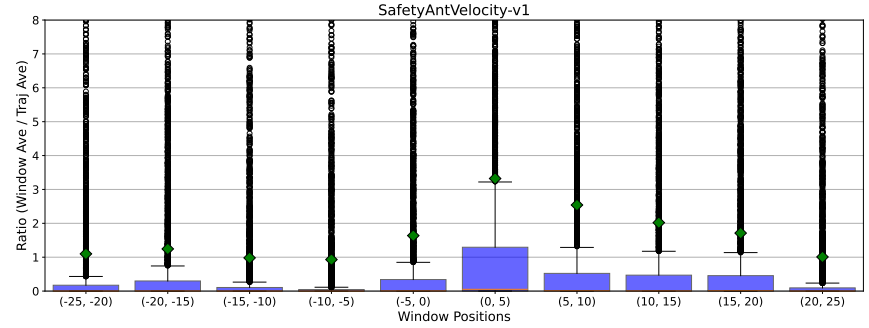


Figure 11: Ratio of Average Logscore within 5-Timestep Window to Overall Average Logscore around Critical Region (Ant)

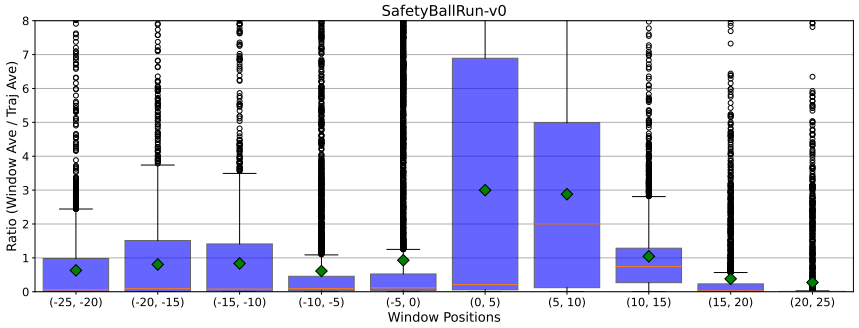


Figure 12: Ratio of Average Logscore within 5-Timestep Window to Overall Average Logscore around Critical Region (BallRun)

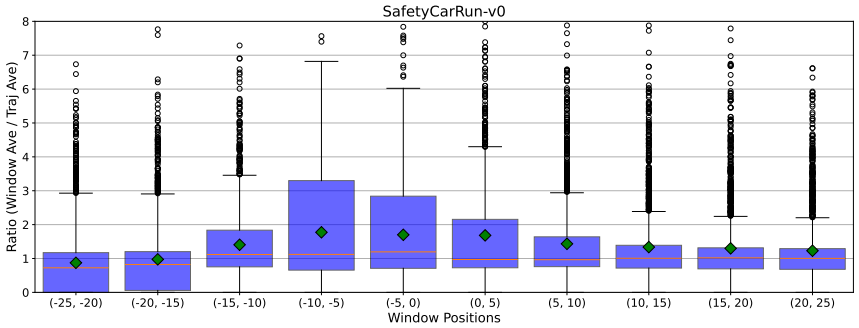


Figure 13: Ratio of Average Logscore within 5-Timestep Window to Overall Average Logscore around Critical Region (CarRun)

Table 2: Safe RL Performance Comparison (Evaluation Environment) in Bullet Safety Gym. Safe policies in bold black color, and best among all safe in blue color; all others in grey. Numbers in () are standard deviations. Note that RLSF implementation does not support Bullet Safety Gym.

Task	Metric	PPO-Lagrangian	C&B + CRL (Ours)	SSV + TraCeS (Ours)
AntRun	Reward	683.7 (24.7)	654.6 (39.8)	590.5 (32.3)
	Cost	25.4 (25.3)	33.3 (21.5)	21.4 (1.9)
	Labeled Trajectories	NA	6000 (0)	6521 (1008)
BallRun	Reward	579.5 (332.2)	485.8 (29.7)	**457.0 (30.1)**
	Cost	17.8 (28.9)	35.3 (22.3)	**24.7 (0.8)**
	Labeled Trajectories	NA	6000 (0)	**2627 (229)**
CarRun	Reward	577.7 (17.8)	**569.1 (8.2)**	**568.3 (9.4)**
	Cost	22.0 (8.9)	**13.3 (11.9)**	**23.3 (1.1)**
	Labeled Trajectories	NA	**6000 (0)**	**8832 (988)**
DroneRun	Reward	459.0 (10.4)	446.4 (5.1)	**440.0 (3.3)**
	Cost	33.2 (8.1)	18.1 (7.2)	**13.6 (10.5)**
	Labeled Trajectories	NA	6000 (0)	**3494 (541)**

Table 3: Safe RL (Evaluation Performance) Comparison for SSV Model with Selective Feedback. Safe policies in bold black color, and best among all safe in blue color; all others in grey. Numbers in () are standard deviations.

Task	Metric	SSV + TraCeS (Selective)	SSV + TraCeS (All)
PointCircle1	Reward	**43.2 (1.5)**	44.7 (2.5)
	Cost	**16.2 (7.2)**	17.0 (7.9)
PointCircle2	Reward	41.1 (0.9)	**40.1 (2.1)**
	Cost	19.8 (9.5)	**15.4 (10.3)**
CarCircle1	Reward	**17.4 (0.9)**	18.3 (0.3)
	Cost	**11.3 (8.0)**	14.9 (9.6)
CarCircle2	Reward	**15.2 (0.9)**	15.6 (0.6)
	Cost	**11.9 (10.0)**	10.0 (4.9)
Ant	Reward	2885.4 (12.3)	**2821.3 (242.5)**
	Cost	19.5 (5.3)	**17.5 (2.9)**
HalfCheetah	Reward	**2372.5 (272.6)**	2771.1 (139.4)
	Cost	**22.4 (5.3)**	19.5 (1.9)
Hopper	Reward	**1610.7 (48.9)**	1655.1 (41.0)
	Cost	**17.1 (4.7)**	20.6 (2.9)
Walker2d	Reward	**2211.2 (182.0)**	2329.4 (309.1)
	Cost	**24.1 (3.9)**	20.7 (2.0)
AntRun	Reward	590.5 (32.3)	**587.7 (27.4)**
	Cost	21.4 (1.9)	**22.8 (2.8)**
BallRun	Reward	**457.0 (30.1)**	499.2 (12.3)
	Cost	**24.7 (0.8)**	24.6 (0.8)
CarRun	Reward	568.3 (9.4)	**566.5 (14.9)**
	Cost	23.3 (1.1)	**23.7 (0.7)**
DroneRun	Reward	440.0 (3.3)	394.6 (134.5)
	Cost	13.6 (10.5)	32.0 (45.6)