Risk Aversion and Guided Exploration in Safety-Constrained Reinforcement Learning

Yang, Q.

DOI
10.4233/uuid:ca5a81c2-f895-4638-bce5-1423a5943381

Publication date
2023

Document Version
Final published version

Citation (APA)
Yang, Q. (2023). *Risk Aversion and Guided Exploration in Safety-Constrained Reinforcement Learning*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:ca5a81c2-f895-4638-bce5-1423a5943381

Risk Aversion and Guided Exploration in Safety-Constrained Reinforcement Learning

Qisong YANG

RISK AVERSION AND GUIDED EXPLORATION IN SAFETY-CONSTRAINED REINFORCEMENT LEARNING

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus prof. dr. ir. T.H.J.J. van der Hagen,
Chair of the Board for Doctorates
to be defended publicly on
Friday 23 June 2023 at 12:30 o'clock

by

Qisong YANG

Master of Science in Engineering,
Xidian University, Xi'an, China
born in Xiangyang, Hubei, China

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus	Chairperson
Dr. M.T.J. Spaan	Technische Universiteit Delft, promoter
Dr. S.H. Tindemans	Technische Universiteit Delft, copromoter
Independent members:	
Prof. dr. A. Farinelli	Università degli Studi di Verona, Italy
Prof. dr. G.C.H.E. de Croon	Technische Universiteit Delft
Prof. dr. M.M. de Weerdt	Technische Universiteit Delft
Dr. N.H. Jansen	Radboud Universiteit
Dr. J.W. Böhmer	Technische Universiteit Delft

This research was partially supported by Xidian University.

Keywords: reinforcement learning, constrained optimization, quantile regression, task-agnostic exploration

ISBN 978-94-6384-458-1

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

献给一楠和子牧

To Yinan and Zimu :)

CONTENTS

SUMMARY

In traditional reinforcement learning (RL) problems, agents can explore environments to learn optimal policies through trials and errors that are sometimes unsafe. However, unsafe interactions with environments are unacceptable in many safety-critical problems, for instance in robot navigation tasks. Even though RL agents can be trained in simulators, there are many real-world problems without simulators of sufficient fidelity. Constructing safe exploration algorithms for dangerous environments is challenging because we have to optimize policies under the premise of safety. In general, safety is still an open problem that hinders the wider application of RL.

Constrained RL is taken as the main formalism of safe exploration, where the reward function and cost function (related to safety) are distinct. This framework tries to mitigate the problem of designing a single reward function that needs to carefully select a trade-off between safety and performance, which is problematic in most instances. However, in this formulation, it can be hazardous to set constraints on the expected safety signal without considering the tail of the distribution. For instance, in safety-critical domains, worst-case analysis is required to limit the frequency of very unsafe outcomes. We propose a method called Worst-Case Soft Actor Critic (WCSAC) for safe RL that approximates the distribution of accumulated safety costs to achieve risk control. More specifically, a certain level of conditional Value-at-Risk from the distribution is regarded as a safety constraint, which guides the change of adaptive safety weights to achieve a trade-off between reward and safety. As a result, we can compute policies whose worst-case performance satisfies the constraints. We investigate two ways to estimate the safety-cost distribution, namely a Gaussian approximation and a quantile regression algorithm. The Gaussian approximation is simple and easy to implement, but may underestimate the safety cost. Moreover, the quantile regression leads to a more conservative behavior. The empirical analysis shows that both versions of WCSAC attain better risk control compared to expectation-based methods, and the quantile regression version shows strong adaptability in complex safety-constrained environments.

Often, RL agents are trained in a controlled environment, such as a laboratory, before being deployed in the real world. However, the target reward might be unknown prior to deployment. Reward-free RL addresses this problem by training an agent without the reward to adapt quickly once the reward is revealed. We consider the constrained reward-free setting, where an agent learns to explore safely without the reward signal.

This agent is trained in a controlled environment, which allows unsafe interactions and still provides the safety signal. We propose a practical Constrained Entropy Maximization (CEM) algorithm to solve task-agnostic safe exploration problems. The CEM algorithm aims to learn a policy that maximizes state entropy under the premise of safety. To avoid needing to explicitly approximate the state density in complex domains, CEM leverages a model-free entropy estimator to evaluate the efficiency of exploration. In terms of safety, CEM minimizes the safety costs, and adaptively trades off safety to exploration based on the current safety performance. After the target task is revealed, unsafety during training is not allowed anymore. We present a safe guide (SaGui) framework for safe transfer learning. To ensure safety during training, the safe exploration policy is leveraged to compose a safe sampling policy. Drawing from transfer learning, we also regularize a target policy towards the safe exploration policy as long as the target policy is unreliable and gradually eliminate the influence from the guide as training progresses. The empirical analysis shows that CEM allows learning a safe exploration policy efficiently, and with SaGui, the learned policy can benefit downstream tasks in safety and sample efficiency.

According to the practical requirements of real-world RL applications, we design the algorithm from two perspectives, i.e., risk control in safety, and safety during training. The proposed algorithms effectively solve the problems of safety risk avoidance, training process safety, and rapid adaptation to new tasks. The proposed algorithms made important steps to further apply RL in the real world. In the future, we will further explore more efficient ways to approximate the distribution of the accumulated safety costs, and meta-learning methods to realize the safety during training.

SAMENVATTING

In traditionele reinforcement learning (RL) problemen kunnen agenten omgevingen verkennen om een optimale beleid te leren door middel van trial-and-error-methoden die soms onveilig zijn. Echter, onveilige interacties met de omgeving zijn onacceptabel in veel veiligheidskritieke problemen, zoals bijvoorbeeld bij robotnavigatie. Hoewel RL-agenten kunnen worden getraind in simulaties, zijn er veel real-world problemen zonder simulaties met voldoende nauwkeurigheid. Het construeren van veilige exploratie-algoritmen voor gevaarlijke omgevingen is uitdagend omdat we een beleid moeten optimaliseren met het oog op veiligheid. Over het algemeen is veiligheid nog steeds een open probleem dat de bredere toepassing van RL belemmert.

Constrained RL wordt beschouwd als de belangrijkste formalisering van veilige verkenning, waarbij de beloningsfunctie en kostenfunctie (gerelateerd aan veiligheid) gescheiden zijn. Deze formalisering tracht het probleem van het ontwerpen van een enkele beloningsfunctie die een zorgvuldige afweging moet maken tussen veiligheid en prestaties te omzeilen, omdat dit in de meeste gevallen problematisch is. Echter, in deze formulering schuilt er gevaar in het opleggen van beperkingen aan het verwachte veiligheidssignaal zonder rekening te houden met de staart van de verdeling. Bijvoorbeeld, in veiligheidskritieke domeinen is een worst-case analyse vereist om de frequentie van zeer onveilige resultaten te beperken. Wij stellen een methode voor met de naam Worst-Case Soft Actor Critic (WCSAC) voor veilige RL die de verdeling van opgetelde veiligheidskosten benadert om risicobeheersing te bereiken. Meer specifiek wordt een bepaald niveau van voorwaardelijke Value-at-Risk uit de verdeling beschouwd als een veiligheidsdrempel, die de verandering van aanpasbare veiligheidscoëfficiënten leidt om een afweging tussen beloning en veiligheid te bereiken. Als resultaat kunnen we een beleidsfunctie berekenen waarvan de prestaties in het slechtste geval aan de veiligheidsdrempel voldoen. We onderzoeken twee manieren om de veiligheidskostenverdeling te schatten, namelijk een Gaussische benadering en een kwantielregressie-algoritme. De Gaussische benadering is eenvoudig en gemakkelijk te implementeren, maar kan de veiligheidskosten onderschatten. Bovendien leidt de kwantielregressie tot meer conservatief gedrag. De empirische analyse toont aan dat beide versies van WCSAC betere risicobeheersing bereiken in vergelijking met methoden die zich baseren op verwachtingswaarden, en de versie met kwantielregressie toont sterke aanpasbaarheid in complexe veiligheidsbeperkte omgevingen.

Vaak worden RL-agents getraind in een gecontroleerde omgeving, zoals een laboratorium, voordat ze in de echte wereld worden ingezet. De daadwerkelijk te leren belongingsfunctie kan echter nog onbekend zijn voordat de agent wordt ingezet. Reward-free RL pakt dit probleem aan door een agent te trainen zonder beloningen, zodat deze zich snel kan aanpassen zodra de beloningsfunctie bekend wordt. We beschouwen de constrained reward-free setting, waarin een agent leert om veilig te verkennen zonder de beloningsignalen. Deze agent wordt getraind in een gecontroleerde omgeving, die onveilige interacties mogelijk maakt maar wel het veiligheidssignaal biedt. We stellen een praktisch Constrained Entropy Maximization (CEM) algoritme voor om taakagnostische, veilige verkeninnigsproblemen op te lossen. Het CEM-algoritme heeft als doel een beleid te leren dat de entropie van de toestand maximaliseert onder de voorwaarde van veiligheid. Om te voorkomen dat de dichtheid van de toestand expliciet moet worden benaderd in complexe domeinen, maakt CEM gebruik van een modelvrije entropieschatting om de efficiëntie van de verkenning te evalueren. Wat veiligheid betreft, minimaliseert CEM de veiligheidskosten en maakt het dynamisch een afweging tussen veiligheid en verkenning op basis van de huidige veiligheidsprestaties. Nadat de doeltaak bekend is, is onveiligheid tijdens de training niet meer toegestaan. We presenteren een veilig gids (SaGui) framework voor veilige transfer learning. Om veiligheid tijdens de training te waarborgen, wordt het veilige verkenningbeleid gebruikt om een veilig samplingsbeleid op te stellen. Gebaseerd op transfer learning, reguleren we ook een doelbeleidsfunctie dichter naar het veilige verkenningbeleid zolang de doelbeleidsfunctie onbetrouwbaar is, en elimineren we geleidelijk de invloed van de gids naarmate de training vordert. De empirische analyse toont aan dat CEM efficiënt een veilig verkenningbeleid kan leren, en met SaGui kan het geleerde beleid voordelen opleveren voor navolgende taken op het gebied van veiligheid en steekproefefficiëntie.

Aan de hand van de praktische vereisten voor RL-toepassingen in de echte wereld ontwerpen we het algoritme vanuit twee perspectieven, namelijk risicobeheersing op het gebied van veiligheid en veiligheid tijdens training. De voorgestelde algoritmen lossen effectief de moeilijkheid op van het vermijden van veiligheidsrisico's, veiligheid tijdens het trainingsproces en snelle aanpassing aan nieuwe taken. De voorgestelde algoritmen zijn belangrijke stappen om RL verder toe te passen in de echte wereld. In de toekomst zullen we verder onderzoeken naar efficiëntere manieren om de verdeling van de opgebouwde veiligheidskosten te benaderen, en meta-leren methoden om veiligheid tijdens training te realiseren.

总结

在传统的强化学习问题中，智能体通过探索环境以学习最优策略，而无需考虑安全问题。然而，在许多现实问题中，安全性是尤其关键的。例如在机器人导航任务中，与环境的不安全交互是不可接受的。尽管智能体可以先在模拟器中进行训练，但在没有足够逼真度的模拟器的情况下，仍然存在许多现实问题。构建用于现实环境的安全强化学习算法非常具有挑战性，因为智能体必须在安全的前提下进行学习。总的来说，安全仍然是阻碍强化学习广泛应用的一个重要问题。

我们通常将强化学习中安全处理成一种约束，并将安全函数分离出奖励函数。这种形式避免了设计单一奖励函数而需要仔细权衡安全和性能的问题。然而，我们通常会选择约束长期安全成本的期望，在不考虑分布尾部的情况下对期望设置约束可能是危险的。在对安全敏感的领域中，需要对最坏的情况进行分析，以避免灾难性的结果。我们提出了一种Worst-Case Soft Actor Critic (WCSAC) 算法框架，该方法通过近似累积安全成本的分布，以实现风险控制。更具体地说，来自分布的一定水平的条件风险值被视为安全约束，它引导自适应安全权重的变化，以实现奖励和安全之间的权衡。因此，我们可以计算最坏情况性能满足约束的策略。我们研究了两种估计安全成本分布的方法，即高斯近似和分位数回归算法。一方面，高斯近似简单且易于实现，但可能低估安全成本，另一方面，分位数回归导致更保守的行为。实验分析表明，两种估计方法都可以实现良好的风险控制，但分位数回归方法在复杂的安全约束环境中取得更加优异的结果。

智能体在被部署到现实世界之前，会在受控环境（如实验室）中进行训练。但是，目标奖励在部署之前可能未知。我们考虑有安全约束的无奖励环境，在其中训练一个安全探索策略。该智能体在受控环境中进行训练，允许与环境不安全的交互，但仍提供安全信号。我们提出了Constrained Entropy Maximization (CEM)算法，旨在安全前提下最大化状态熵，以学习可以实现对所有状态进行均匀访问的安全探索策略。该方法利用了无模型的熵估计器以避免对整个状态密度的近似。CEM 采用置信域算法提高安全性，并基于当前安全表现自适应地权衡安全与探索。目标任务出现后，与环境的不安全交互将不再被允许。因此，我们提出了Safe Guide (SaGui)安全迁移学习框架。训练好的安全探索策略将首先被用于制定安全的采样策略。同时，在目标策略还不成熟的情况下，我们还会将目标策略往安全探索策略正则，并随着训练的进展逐渐消除安全探索策略对目标策略的影响。实验分析表明，CEM 方法可以有效地学习安全探索策略，该策略在SaGui的框架下实现了安全迁移学习，帮助更快地学习到解决目标任务的最优策略。

本文依据强化学习应用的现实要求，从安全风险控制和训练安全两个角度进行了算法研究。提出的算法有效地解决了安全风险规避，训练过程安全及新任务快速适应的问题，为进一步将强化学习应用于现实世界提供了新的参考。在未来，我们会进一步深入研究对累积安全成本分布的估计，及通过元学习的方式实现训练过程的安全。

ACKNOWLEDGEMENTS

Over the course of the past four years, the completion of this dissertation has represented a precious journey that has resulted in profound scientific and personal growth. While numerous individuals have contributed directly or indirectly to this work, I may not be able to personally mention each and every one of you. Nevertheless, I want you all to know that I am deeply grateful for the unwavering support I have received. Without your kindness and assistance, this dissertation would not have come to fruition.

First and foremost, I extend my heartfelt gratitude to my esteemed supervisor, Matthijs Spaan, and my co-supervisor, Simon Tindemans. I am immensely thankful to both of you for engaging in countless captivating discussions, co-authoring papers, guiding me toward my research path, always being available, and providing overall support. Matthijs, I consider myself fortunate to have had you as my advisor, as you have granted me the freedom to explore novel ideas while offering the appropriate guidance and support during challenging times. Simon, I deeply appreciate your disciplined and critical thinking, which has served as a fundamental source of motivation for me. Your candid and constructive feedback has consistently enhanced both myself and my work.

I would like to extend special thanks to Thiago Dias Simão for our close collaboration. Working with you has always been inspiring and enjoyable. Your valuable insights have greatly influenced my research. Furthermore, your guidance on making my technical writing more engaging has left a profound impact on this dissertation. Wendelin, I highly value the insightful discussions we've had during coffee breaks. I would also like to express my gratitude to Nils Jansen, Danial Kamran, and Johannes Fischer for their contributions to co-authoring papers.

I am indebted to all my colleagues in the Algorithmics group for fostering a pleasant and supportive office environment throughout these years. The combination of exchanging scientific ideas and having moments of joy has been truly inspiring. I have learned something valuable after each lunch talk, particularly during my initial months, where your advice on life and work greatly benefited me, allowing me to swiftly adapt to the new environment. Our nostalgic team-building activities, such as canoeing and farmer golf, as well as the retreat in Julianadorp, where we got to know each other better, will forever hold a special place in my heart. I would like to express my gratitude to Kim, Greg, and Koos for providing me with immense support as I settled down in Delft. Warm greetings to Mathijs,

Neil, Frans, Cees, Emir, Anna, Sebastijan, Alexandru, Grigorii, Ksenija, Eghonghon, Moritz, Pascal, Noah, Joery, Oussama, Max, and Benjamin.

To Longjian, I am grateful for your support during the period of separation from my family. Although we didn't know each other very well prior to the onset of the pandemic, we found solace in supporting one another and embarked on a challenging yet extraordinary journey that only the two of us truly understood. Canmanie and Lei, your heartfelt gift of sixty-six handmade dumplings for Zimu brought warmth and joy to the lives of Yinan and me. Yang, I extend my thanks for our collaboration, your academic guidance, and your career advice. The parties at your house are forever etched in my memory. Junhan, I appreciate your assistance in picking up lunch or dinner meals, and the captivating stories we shared during each coffee break. You never made a mistake in giving bubble tea advice.

To Jinke, Zilong, Cheng, Bo Sun, Xueer, Wei, Jun, Li, and Ruoyun, I am profoundly grateful for always being there for me. The cherished moments we spent together in Delft will forever be etched in my mind. Whether it was cooking, traveling, or exercising together, all the pressures and troubles seemed to fade away. Your companionship has brought an end to the loneliness in my life. Warm greetings to Hao, Tianqi, Chi, Bo Li, Chenguang, and Na.

Qisong Yang
Delft, 1 December 2022

List of Abbreviations

AI	artificial intelligence
RL	reinforcement learning
MDP	Markov decision process
CMDP	constrained Markov decision process
CVaR	conditional Value-at-Risk
SAC	Soft Actor Critic
WCSAC	Worst-Case Soft Actor Critic
TD	temporal difference
CDF	cumulative distribution function
PDF	probability density function
SaGui	Safe Guide
DQN	Deep Q Network
CPO	Constrained Policy Optimization
TRPO	Trust Region Policy Optimization
PPO	Proximal Policy Optimization
QR	quantile regression
FQF	Fully parameterized Quantile Function
IQN	implicit quantile network
CEM	constrained entropy maximization
k-NN	k-nearest neighbors
KKT	Karush-Kuhn-Tucker
TASE	task-agnostic safe exploration

List of Parameters

d	safety threshold
h	minimum policy entropy
k	number of neighbors for entropy
J	loss function
T	time horizon
α	risk level
β	policy entropy weight
γ	discount factor
δ	trust-region threshold
θ	neural network parameters
ω	safety weight
ι	initial state distribution
λ	learning rate
ρ	state density
τ	quantile fraction

1

INTRODUCTION

Artificial intelligence (AI; Dick, 2019) is becoming more and more important in people's lives. AI-powered applications in online advertising, machine translation, smart cities, etc. are revolutionizing our future. A milestone of AI development was the AlphaGo computer program's defeat of world Go champions Ke Jie and Lee Sedol. The core technology of AlphaGo is reinforcement learning (RL; Sutton and Barto, 2018), which is inspired by behavioral psychology (Mills, 1998) and widely considered the most likely way to achieve universal AI. RL, a methodology within the field of machine learning (ML; Mitchell and Mitchell, 1997), is utilized to address and solve problems where agents aim to maximize returns or attain specific objectives by learning and adapting policies through interaction with the environment (Sutton and Barto, 2018).

Compared to other ML paradigms, RL also requires a large amount of data for training, but does not need any training data to be given in advance, and the types of data are different. Instead of diversified labeled or unlabeled data, RL collects data interactively using a reward signal. RL does not concern about the formalization of the input, but focuses more on what action should be taken under the current input to achieve the ultimate goal. RL has a distinct advantage over traditional supervised and unsupervised learning approaches when it comes to solving complex sequential decision making (SDM) problems (Roy, 2002; Sutton and Barto, 2018). In an RL setting, the environment is often dynamic, uncertain, and modeled as a Markov decision process (MDP) (MDP; Puterman, 2014). Typically, an exact mathematical model of the MDP is not known, and the

Parts of this chapter have been published in (Yang, Simão, Tindemans, et al., 2021; Yang, Simão, Jansen, et al., 2022; Yang, Simão, Tindemans, et al., 2023; Yang and Spaan, 2023).

focus is on large MDPs where traditional methods such as dynamic programming become impractical. In an RL problem (Figure 1.1), the agent interacts with the environment at discrete time steps. At each step, the agent receives the current state and associated reward, selects an action from the available options, and the environment transitions to a new state and the reward associated with the transition is determined. The ultimate goal of an RL agent is to find a policy that maximizes the expected cumulative reward.

RL has seen significant advancements in recent years, with key solutions emerging to tackle various challenges. One such solution is the Soft Actor Critic (SAC; Haarnoja, Zhou, Abbeel, et al., 2018) algorithm. SAC combines elements from previous RL algorithms, such as Deep Q-Networks (DQN; Mnih et al., 2015) and Deep Deterministic Policy Gradient (DDPG; Silver et al., 2014), to address the exploration-exploitation trade-off and improve sample efficiency. DQN utilizes a neural network to approximate the action-value function, while DDPG employs an actor-critic architecture with deterministic policies. SAC builds upon these foundations by introducing an entropy regularization term, which encourages exploration and improves robustness. By simultaneously maximizing the policy entropy and expected reward, SAC achieves a balance between exploration and exploitation, leading to improved performance in RL tasks.

In classical RL, the agents learn by trial and error, where arbitrary short-term loss is acceptable for long-term gain when exploring the environment. However, in some situations with safety concerns, we should not only pay attention to the long-term rewards, but also safety assurance. A few safety-critical domains of RL are listed as follows:

- RL has the potential to train autonomous robots, such as self-driving cars or service robots, to efficiently complete assigned tasks. However, it is essential to ensure that the safety of humans and property is prioritized over the completion of tasks. Specifically, the RL-trained robot should never take actions that could result in the breaking of expensive equipment or harm to humans, even if completing the task would otherwise be possible. This emphasis on safety must be integrated into the design and implementation of the RL system to guarantee the reliability and security of the robot in real-world scenarios.

- The integration of new features, such as increasing amounts of renewable energy, into power networks presents challenges for human operators. RL has the potential to address these challenges by replacing human operators and allowing for greater adaptability in power network operations. However, it is important to note that the use of RL in power network operations also carries risks, particularly the potential for random actions that could cause a blackout. Such outcomes are strictly unacceptable, highlighting the importance of considering the safety and reliability implications when implementing RL in power network operations, and implement-

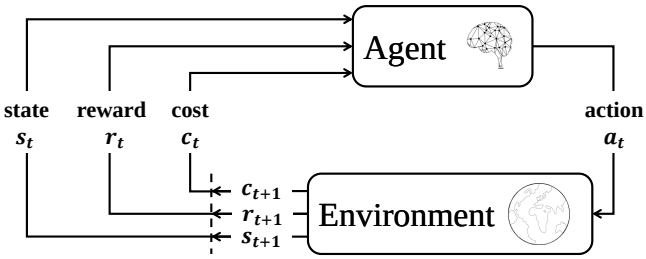


Figure 1.1: RL with a separate safety signal. In addition to the reward function in standard RL, we have a cost function for the environment.

ing rigorous safety mechanisms to prevent catastrophic failures (Marot et al., 2020; Subramanian et al., 2021).

- Recommender systems based on RL have the potential to be powerful tools for improving user experience and personalization. However, it is important to note that these systems can also pose risks for certain groups of users. Specifically, the random exploration of the system may reveal content that is psychologically harmful or extremist in nature. Therefore, it is crucial to consider the potential negative impacts of these systems when designing and implementing them, and to take steps to mitigate these risks (Di Noia et al., 2022).

Therefore, given the learning objectives, it is essential to explore safe ways to finish the task. This sub-field within RL is called safe reinforcement learning (safe RL; García and Fernández, 2015). Safe RL is defined as "the process of learning policies that maximize the expectation of the return in problems in which it is important to ensure reasonable system performance and/or respect safety constraints during the learning and/or deployment processes" (Mihatsch and Neuneier, 2002). RL agents often learn in a safety-insensitive controlled environment (García and Fernández, 2015), such as a laboratory or a simulator, before they are deployed in the real world. In this case, we only need a safe policy for deployment without safety concerns during the learning. However, when controlled environments are not available, safety for learning and deployment are both critical.

1.1. SAFETY DEFINITIONS AND ALGORITHMS

The core issue in safe RL is determining the definition of safety and how it can be incorporated into the learning process. Throughout the whole dissertation, we formulate safety as constraints that are not only addressed for the deployment processes, but also during the learning.

1.1.1. SAFETY-CONSTRAINED RL

Constrained RL is taken as a natural and universally-relevant formalism of safe RL (Altman, 1999). Safety constraints are usually related to resource consumption (Walraven and Spaan, 2018), or built on the most common definition of safety that is related to the given label (safe or unsafe) of the environment state. An RL agent is considered safe if it never exceeds the resource limit or visits unsafe states within a frequency limit (Hans et al., 2008).

In safety-constrained RL, we have a separate safety signal, which can be regarded as a second (negative) reward that is called cost. The cost function is an incentive mechanism that tells the agent what is safe or unsafe. As shown in Figure 1.1, at each time step, the agent takes into account the current cost, the state, and the reward received for the previous action taken. The safety is evaluated by comparing the long-term safety costs to a given safety threshold, which describes our goal in safety. For instance, when running an electric vehicle, the consumption of electricity after taking each action can be regarded as the cost, and the battery capacity is the safety threshold.

The safety constraints avoid designing a complex reward signal. In standard RL, we just maximize the long-term rewards. With additional safety concerns, we must design a single reward that carefully trades off performance for safety. However, before running an RL algorithm, we cannot check if we have the correct trade-off parameter for the desired safety and performance. Even though we may find a fixed trade-off parameter that results in our desired safety specification, it ignores the fact that we also need to meet the safety requirements during training (Ray, Achiam, and Amodei, 2019). These problems are unavoidable with a single reward and have been observed in practice (Achiam et al., 2017; Dalal et al., 2018; Pham, De Magistris, and Tachibana, 2018). Instead, after decoupling safety from reward by safety constraints, we mitigate the above problems.

Accordingly, constrained RL algorithms to address safety have been proposed. The Constrained Policy Optimization (CPO; Achiam et al., 2017) method is a general-purpose policy search algorithm for constrained RL that allows for specifying both reward functions and constraints, ensuring near-constraint satisfaction at each iteration and providing guarantees about policy behavior throughout training. The Interior-point Policy Optimization (IPO; Liu, Ding, and Liu, 2020) method utilizes logarithmic barrier functions inspired by the interior-point method to optimize policies in RL algorithms, enabling the maximization of long-term rewards while satisfying cumulative constraints in various multi-constraint settings. The Lagrangian version of a collection of traditional RL algorithms, e.g., Proximal Policy Optimization (PPO-Lag; Ray, Achiam, and Amodei, 2019) and Soft Actor Critic (SAC-Lag; Ha et al., 2020), leverages the Lagrangian relaxation technique to handle constrained RL problems, where an augmented Lagrangian objective is incorporated into the traditional RL algorithm, enabling the optimization of constrained policies

through the use of penalty terms. These constrained RL approaches offer valuable insights into how to handle various constraints and improve RL performance in challenging scenarios.

1.1.2. ALTERNATE APPROACHES

Over the past years, safe RL has achieved great progress in learning policies under the premise of safety. Even though constrained RL is widely considered as the primary approach for safe exploration, the definition of safety may not always be consistent or identical. While previous studies such as (García and Fernández, 2015; Ray, Achiam, and Amodei, 2019) have provided a detailed categorization of safe RL, in order to better contextualize our work within the field, we have chosen to categorize safe RL based on three different criteria: the specific methods and techniques used, the objectives or outcomes that the approach aims to achieve, and the stage of the RL process at which the approach is applied.

OPERATION METHODS

We can categorize safe RL based on the methods or techniques used to ensure safe behavior. Then, it becomes easier to compare our methods to different approaches and identify their strengths and weaknesses.

- **Reward shaping.** We can guide the RL agent towards the desired behavior by modifying the reward signal. Specifically, we can enhance safety by implementing reward-shaping functions, which add an additional bonus or penalty to the original reward based on the agent's actions (Ng, Russell, et al., 2000; Grbic and Risi, 2020; Kamran, Simão, et al., 2022).

- **Safeguard.** We can leverage the environment model (Prakash et al., 2019), expert intervention (Peng et al., 2022), or shielding mechanism (Alshiekh et al., 2018) to make more informed decisions or provide corrections to the agent's dangerous actions. The methods in this category are particularly crucial when the learning phase is required to be safe.

- **Constrained RL.** We use safety constraints to ensure that the agent does not violate any predefined rules or regulations, where we have to define a set of constraints and use a separate constraint-based algorithm (Chow et al., 2017; Yang, Rosca, et al., 2020). All the methods presented in this dissertation can be classified within this category.

SAFETY GOALS

We can categorize safe RL based on the specific objectives or outcomes, which have different implications for the safety of the agent and require different methods to achieve.

- **Preventing detrimental results.** When the environment is divided into safe and unsafe states, it is crucial to guide the agent towards safe actions while limiting its ability to take unsafe actions (Hans et al., 2008). If the RL problem is related to resource consumption, we may also need to limit the resource consumption (Walraven and Spaan, 2018). The safety goal in this dissertation can be taken as preventing detrimental results, which may be caused by exceeding the resource limit or visiting unsafe states over a frequency limit.

- **Policy improvement.** Safety in RL can be defined in terms of the monotonicity of the return during the training process. This means that we aim to prevent any significant decrease in performance during the learning process, as it can be detrimental to the agent's overall performance. To achieve this, our safety goal is to ensure that the agent's return improves at each gradient step and that the agent does not experience any dangerous performance degradation (Pirotta et al., 2013; Papini, Pirotta, and Restelli, 2019; Simão and Spaan, 2019).

- **Ergodicity.** The safety goal can be to ensure ergodicity in RL. If an agent is not ergodic, it may get stuck in an unsafe or suboptimal state and will not be able to limit its cost (Moldovan and Abbeel, 2012; Turchetta, Berkenkamp, and Krause, 2016; Eysenbach et al., 2018). In this case, safety is related to if actions are reversible. This goal considers an agent to be safe if it has the ability to transition between any state it encounters, without any restrictions.

LEARNING STAGES

We can categorize safe RL based on the stage of the RL process at which the approach is applied. In this dissertation, we not only focus on ensuring safety during the deployment phase, but also on addressing safety concerns during the entire training process.

- **Deployment phase.** When we have access to simulations that accurately replicate the deployment environment, we can use them to train an agent that is safe in the actual deployment phase (García and Fernández, 2015). In this case, we can focus on ensuring that the final policy is safe, without worrying about safety during the training process. This can be effective only when the simulations have high fidelity and are able to accurately replicate the dynamics of the deployment environment. However, it is important to keep in mind that this approach may not guarantee safety during the training phase and additional safety mechanisms may still be

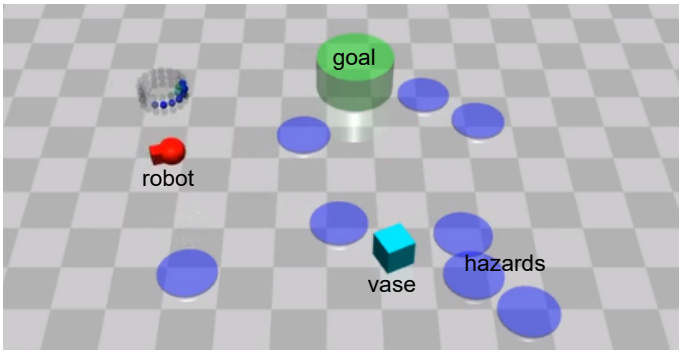


Figure 1.2: Example Safety Gym environment. In a 2D map, a point robot navigates to reach a goal area while trying to avoid a vase and several hazards.

required (Achiam et al., 2017; Ha et al., 2020; Yang, Rosca, et al., 2020; Qin, Chen, and Fan, 2021).

- **Learning phase.** In the absence of a simulation environment, addressing safety during the training phase becomes crucial. In such cases, incorporating prior knowledge or a predictive mechanism (safeguard in operation methods) can be an effective way to avoid learning from scratch and ensure the safety of the agent (Simão, Jansen, and Spaan, 2021; Peng et al., 2022; Yang, Simão, Jansen, et al., 2022). For instance, we need to know an initial set of safe states to ensure the early stages of learning are safe (Turchetta, Berkenkamp, and Krause, 2016). Then, the agent gradually increases the set of safe states and decreases the uncertainty in the safety function. In this case, we can ensure safety in the learning phase.

Note that the above categories for safe RL are not mutually exclusive and many safe RL approaches may fall into multiple categories. For instance, the work by Peng et al. (2022) can be a safeguard method to prevent detrimental results during the learning phase. While we have provided an overview of the most popular and mainstream directions in safe RL, it is important to keep in mind that new and innovative approaches may not fit neatly into any one category.

1.1.3. BENCHMARKING RL SAFETY

In this dissertation, we evaluate the effectiveness of our proposed methods by using various benchmark environments. These environments help to showcase the safety concerns that can arise in RL. Generally, traditional RL benchmark environments do not have specific safety concerns, e.g., the Arcade Learning Environment (Bellemare, Naddaf, et al., 2013), OpenAI Gym (Brockman et al., 2016), and the Deepmind Control

Suite (Tassa et al., 2018). However, some of the environments can be easily modified to have an immediate constraint cost at each time step, and a given cost limit, e.g., the adapted MountainCar and CartPole in Chapter 5.

To build the main benchmark environments in this dissertation, we leverage the Safety Gym (Ray, Achiam, and Amodei, 2019), a suite of complex continuous control environments for evaluating the progress towards RL agents that respect safety constraints during training. In the standard Safety Gym benchmark suite, each environment is formed as a combination of a robot, a task, and a level of difficulty. For instance, in Figure 1.2, a point robot (red) navigates in a 2D map to reach a goal area (green) while trying to avoid a vase (cyan) and several hazards (blue). The robot can get rewards when approaching the goal, but some costs will be incurred if it touches the obstacles. In addition to the standard suite, we can also create custom environments using the Safety Gym engine according to our needs, e.g., the environments we build in Chapter 4 to evaluate the exploration capabilities. In general, Safety Gym is a tool for building the safety-constrained environments we need.

1.2. NECESSITY OF RISK CONTROL

In this dissertation, we formulate safety concerns by constraints (Achiam et al., 2017; Ray, Achiam, and Amodei, 2019) to mitigate the problem of designing a single reward. Previous methods usually define safety on the expected long-term safety costs, which ensures the average performance is safe (Achiam et al., 2017; Liu, Ding, and Liu, 2020; Yang, Rosca, et al., 2020). However, with the stochastic policy and the dynamics of the environment, the expected value cannot capture the probable risks caused by the randomness in safety. Hence, the individual episodic costs generated by the learned policy might exceed the safety threshold with a high probability. Especially when the distribution of cost-return has long tails, we will have a high risk of detrimental events. For safety-critical problems, it can be hazardous to only ensure the average performance to be safe, where a safe policy has higher returns and higher variance in safety may be preferred over another safe policy with lower returns and lower variance in safety.

Although implementing expectation-based methods with a lower cost threshold could lead to better performance in safety, their worst-case performance is not guaranteed to be safe. After setting the safety threshold to be extremely small, we still cannot control the tail-end of the cost distribution, and the worst-case costs could still be large. On the other hand, for industrial and robotic settings (Jardine, Lin, and Banjevic, 2006; De Nijs, Spaan, and De Weerdt, 2015; Boutilier and Lu, 2016), the safety constraints are always built on the real cost limit. Hence, it is not simple to quantify how much the safety threshold should be shrunk to explicitly limits the violations in the worst case.

Based on the distributional Bellman operator (Sobel, 1982; Morimura et al., 2010; Tamar, Di Castro, and Mannor, 2016), we can model the distribution of cumulative safety costs as a Gaussian (Tang, Zhang, and Salakhutdinov, 2020). However, the Gaussian approximation can be coarse in many domains, especially when the distribution has long tails. With the progress in distributional RL (Bellemare, Dabney, and Munos, 2017; Dabney, Ostrovski, et al., 2018; Dabney, Rowland, et al., 2018; Yang, Zhao, et al., 2019), we can capture the uncertainty in safety more accurately. Even though the state-of-the-art distributional RL algorithms are originally designed for deep Q-networks (DQN; Mnih et al., 2015) with discrete action spaces, they are easy to be generalized to our setting with continuous action spaces. The approximated distribution of cumulative safety costs allows us to take the risks in safety into account, i.e., the various possibilities of the individual episodic costs, especially in the worst cases. Compared to only modeling expected values, better alternatives for safety-constrained RL are algorithms that compute policies based on varying risk requirements, specialized to risk-neutral or risk-averse behavior.

1.3. TRAINING SAFETY ASSURANCE

In traditional RL, agents learn by trial and error that is not allowed in some safety-critical applications. Therefore, we expect to learn policies that can be used in the real world through simulators (García and Fernández, 2015). In this way, we have no concerns about safety, and the simulation may provide a potentially infinite data source. The trial-and-error nature of RL usually refers to the training phase. Under the assumption that high-fidelity simulators exist, most of the current research in safe RL focuses more on how to learn a safe policy, while safety in the training phase is not guaranteed (Achiam et al., 2017; Liu, Ding, and Liu, 2020; Yang, Rosca, et al., 2020). However, it is not always possible to build a simulator with sufficient fidelity, such that the trained policy is acceptable for real-world deployment. Even if multiple research efforts are being directed towards improving the realism of simulations to better transfer knowledge gained in simulation to the real world (Zhao, Queralta, and Westerlund, 2020), and the agent may learn from a mixture of simulation data and real data (Cutler, Walsh, and How, 2014), the direct interactions with the real world during training are still necessary in most cases, so that we have to guarantee safety during training.

Usually, agents will emphasize exploration under an insufficient understanding of the environment. Thus, it is unavoidable for the agents to take some undesired actions that may be dangerous for the environment, especially at the early stage of training. In general, we cannot ensure safety during training if learning from scratch. We may draw ideas from human learning that never starts from scratch, especially in terms of safety

(Grbic and Risi, 2020). To improve the chances of survival, millions of years of evolution have enabled humans to possess many safety instincts, e.g., the evolved instinctual fear of infants to some dangerous animals like spiders and snakes (Hoehl et al., 2017). However, the instinct for safety cannot cope with all situations in the real world. More importantly, infants rarely learn alone, but under the intervention of adults (Saunders et al., 2018; Kelly et al., 2019). On the one hand, the infant can observe the right demonstrations given by the adults and learns rapidly by imitating their behavior and learning how to tackle dangerous situations. On the other hand, under the supervision of adults, the infant can explore the world freely before unconsciously taking dangerous actions (Peng et al., 2022). Therefore, the infant can learn from both the imitation of the adult and the free exploration, which ensures the safety and efficiency of learning.

Similar to human learning, the agent's learning for safety does not need to start from scratch. In this dissertation, we consider that some prior knowledge is required to train RL agents without violating the safety constraints. Instead of evolving safety instincts that never change through lifetime learning (Grbic and Risi, 2020), we investigate how to acquire task-agnostic knowledge to intervene in learning (Abel et al., 2017; Spencer et al., 2020). In this case, Mutti, Pratissoli, and Restelli (2021) indicate that we can embed the knowledge into a meta-reward function, an estimation of the environment dynamics, or an exploration policy, but the exploration policy is intuitively more transferable considering the possible environmental changes and additional policy optimization. Similar to the intervention of adults, prior knowledge will have different influences at different learning stages. With better performance in the target task, the learning will become increasingly independent. In general, we expect that the prior knowledge can give consideration to both safety and exploration, and benefit the target tasks in terms of training-safety assurance and learning acceleration.

1.4. RESEARCH QUESTIONS

This dissertation attempts to solve the problems that hinder safe RL to be further applied in the real world. It begins by defining safety with risk control and ends with task-agnostic safe exploration. Our research is generally built on two natural safety requirements that are critical for the real world:

1. Considering the randomness generated by the stochastic policy and the dynamics of the environment, safety should be defined based on external risk requirements that the user can specify.

2. For RL problems without simulators of sufficient fidelity, prior knowledge is necessary to ensure safety during training if interactions with safety-critical environments are inevitable.

The above safety requirements help formulate the main research question of this dissertation, namely:

How to control risks and ensure safety during training in cost-constrained RL?

We propose sub-questions to answer this main research question in four steps.

Q1 How to formulate the safe RL problems with risk control?

This sub-question aims to expound what is the definition of safety in this dissertation. With separate reward and safety signals, we formulate safety by constraints, which are potentially associated with risk requirements. Risk control means that we have different safety constraints under different risk levels. Instead of only requiring the average performance to be safe, we aim to define safety based on varying risk requirements, specialized to risk-neutral or risk-averse behavior.

Q2 How to optimize a policy under the premise of safety ?

As a continuation of Q1, this question focuses on how to design an RL algorithm for our safety definition. To achieve risk control, we focus on the safety distribution rather than the expected value. Therefore, we must first approximate the distribution of cost-return to set up a general view of safety. In this way, policies can be optimized given different levels of conditional Value-at-Risk (CVaR; Rockafellar and Uryasev, 2000), which determine the degree of risk aversion from a safety perspective.

Q3 How to ensure safety during training by transferring safe exploration policies?

This sub-question represents our additional focus on safety during training. In light of the fact that safety cannot be guaranteed if we learn from scratch, extra knowledge before training is essential to keep the learning process to be safe. Inspired by the intervention of adults in the learning of infants, we propose to leverage a safe exploration policy to guide the learning in downstream tasks, where constraint violations are not allowed during training.

Q4 How to train a safe exploration policy in a principled way?

Following Q3, this sub-question aims to further expound the algorithm to get the safe exploration policy. We expect a policy that can induce a uniform distribution over the state space in a safe way. Accordingly, we maximize the entropy of the state density under the safety constraint. The resulting policy is a general starting point to solve any (unknown) subsequent task. Except for safety, we also expect the exploration capabilities of the safe exploration policy to be beneficial to speed up the learning of the target task.

1.5. CONTRIBUTIONS TO SAFETY-CONSTRAINED RL

We carried out the research from two aspects, i.e., safety-constrained RL with risk control, and training and transferring safe exploration policies[1].

[1]The code in this dissertation is available at https://github.com/qisong-yang/.

Firstly, in safety-constrained RL, it can be hazardous to set constraints on the expected safety performance without considering the randomness in safety. In safety-critical domains, worst-case analysis is required to limit the frequency of very unsafe outcomes. To achieve risk control, we propose a new criterion, risk-averse constrained RL, for safety critical problems, such that we can choose to be risk-averse or risk-neutral in safety. Accordingly, we designed an off-policy algorithm framework Worst-Case Soft Actor Critic (WCSAC). A certain level of CVaR from the distribution is regarded as a safety constraint, which guides the change of adaptive safety weights to achieve a trade-off between reward and safety. As a result, we can compute policies whose worst-case performance satisfies the constraints. We investigate two ways to estimate the safety-cost distribution, namely a Gaussian approximation and a quantile regression algorithm. The Gaussian approximation is simple and easy to implement, but may underestimate the safety cost, and the quantile regression leads to a more conservative behavior.

Secondly, to ensure safety during training, we propose a safe transfer learning framework Safe Guide (SaGui) to leverage a safe exploration policy to enhance safety and improve sample efficiency in downstream tasks. The safe exploration policy is trained in a constrained reward-free setting, where an agent (the guide) learns to explore safely without the reward signal. This agent is assumed to be trained in a controlled environment, which allows unsafe interactions and still provides the safety signal. Accordingly, we propose a practical Constrained Entropy Maximization (CEM) algorithm to solve task-agnostic safe exploration (TASE) problems. The CEM algorithm aims to learn a policy that maximizes state entropy under the premise of safety. Correspondingly, CEM leverages a model-free entropy estimator to evaluate the efficiency of exploration, and adaptively trades off safety to exploration based on the current safety performance. After the target task is revealed, safety violations are not allowed anymore. Thus, the guide is leveraged to compose a safe sampling policy. Drawing from transfer learning, we also regularize a target policy (the student) towards the guide while the student is unreliable and gradually eliminate the influence of the guide as training progresses.

1.6. OUTLINE OF THIS THESIS

Figure 1.3 shows the structure of the dissertation and how each chapter contributes to the overall aim. The rest of this dissertation is organized as follows.

In Chapter 2, we describe mathematical definitions and notations used throughout the dissertation. We first present the basics of CMDPs that are used to model the safety-constrained RL problems. Accordingly, we describe how to learn safety-constrained policies by SAC-Lag. To get more robust policies in safety-critical RL problems, we also elaborate on the formulation of uncertainty in safety by quantile regression methods.

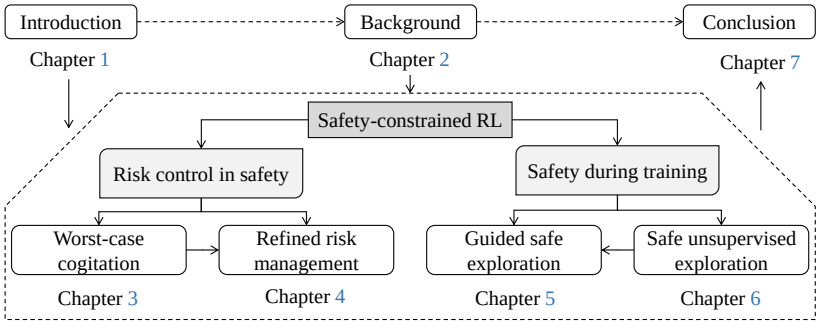


Figure 1.3: Overview of the dissertation structure. Outside the dashed box, we show the introduction and the conclusion of the dissertation. Inside the dashed box, we show the connection between all the technical chapters, and the contribution of each chapter to the overall aim.

In addition, we introduce the state density induced by the stochastic policy and the dynamics of the environment, and the approximation of state density in a model-free manner.

In Chapter 3, we first make a comprehensive analysis of the necessity to control the risk in safe RL. Because of the randomness in realized safety (long-term costs), the expectation-based safety constraint is not enough to capture the risks. Accordingly, we define safety in a more risk-averse way, and propose the WCSAC framework. WCSAC augments SAC with a separate distributional safety critic (parallel to the reward critic) to make the algorithm more adaptive when facing RL problems with higher safety requirements. We elaborate the Gaussian approximation for the cost-return, such that the worst-case performance in safety can be considered when updating the policy. The empirical analysis shows that our algorithm attains better risk control compared to methods with expectation-based safety.

In Chapter 4, we further improve the WCSAC framework, and investigate the pitfalls of the Gaussian safety critic. When the distribution of cost-return is not Gaussian, the approximation cannot describe the distribution accurately by its expectation and variance, and the tail of the distribution might be underestimated, especially when the distribution has long tails. Besides, the Gaussian approximation does not possess the general advantages of distributional RL techniques. So, we present a distributional safety critic modeled by an implicit quantile network (IQN; Dabney, Ostrovski, et al., 2018), which provides a more precise estimate of the upper tail part of the distribution. We also empirically show that, with this more accurate safety critic, WCSAC can achieve better risk control in more complex safety-constrained environments.

In Chapter 5, we emphasize the necessity to introduce prior knowledge to ensure safety during training, which is impossible if learning from scratch. To give consideration to both fast adaptability and safety, we present the safe guide (SaGui) framework to leverage

a safe exploration policy in the downstream tasks, where safety constraint violations are not allowed during training. We explicitly point out that the safe exploration policy (the guide) should have strong exploration capabilities under the premise of safety. Then, we elaborate on how to use the guide to compose a safe behavior policy during data collection, and how to adaptively regularize the target policy to the guide by policy distillation. We also empirically show that SaGui is a safe and sample-efficient way of training the agent on a target task.

In Chapter 6, to further improve the safe guide framework, we propose the CEM algorithm to get the safe exploration policy in a principled way. We analyse the infeasibility to solve the TASE problem based on the traditional optimization objective in RL, i.e., maximizing the long-term intrinsic rewards (exploration bonus). Considering the intricacy to approximate the full state density in complex domains, we elaborate on how to use the k-nearest neighbor state entropy estimator for exploration in safety-critical domains. We also show how to update the policy with an adaptive balance between exploration and safety, which are conflicting objectives. The empirical analysis shows that CEM allows learning a safe exploration policy in complex continuous-control domains, and the policy benefits the downstream tasks.

Finally, Chapter 7 concludes the dissertation by summarising our findings of the safe RL framework designs, listing the contributions and limits of this work, and suggesting future research directions on making RL more applicable in the real world.

2

BACKGROUND

In this dissertation, we solve safety-constrained RL problems, which can be modeled as Constrained Markov Decision Processes (CMDPs). The goal of the agent is to maximize the (discounted) sum of rewards, while adhering to the cost limits provided. Before detailing the contributions and algorithms in this dissertation, we first describe the notations, the models, and the previous methods to handle such safety-constrained RL problems. We first present the basics of CMDPs that are used throughout the whole dissertation in Section 2.1. Accordingly, in Section 2.2, we describe how we can learn a policy for the problem by a Lagrangian version of Soft Actor Critic (SAC-Lag), where the safety is built on the average case. To get more robust policies in safety-critical RL problems, we explain how to formulate the uncertainty in safety by quantile regression methods, presented in Section 2.3.

2.1. CONSTRAINED MARKOV DECISION PROCESSES

We formulate the safe RL problem as a Constrained Markov Decision Process (CMDP; Altman, 1999; Borkar, 2005), defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, c, d, T, \iota)$: where $\mathcal{S}$ is the state space and $\mathcal{A}$ is the action space. In constrained RL an agent interacts with a CMDP, without knowledge about the transition, reward, and cost functions ($\mathcal{P}: \mathcal{S} \times \mathcal{A} \mapsto Dist(\mathcal{S})$, $r: \mathcal{S} \times \mathcal{A} \mapsto [r_{min}, r_{max}]$, and $c: \mathcal{S} \times \mathcal{A} \mapsto [c_{min}, c_{max}]$). Each episode begins in a random state $s_0 \sim \iota(\cdot)$, where $\iota \in Dist(\mathcal{S})$. At each timestep t of an episode, the agent observes the current state $s_t \in \mathcal{S}$, and takes an action $a_t \in \mathcal{A}$. Then, it observes a reward $r(s_t, a_t)$, a cost $c(s_t, a_t)$, and the next state $s_{t+1} \sim \mathcal{P}(\cdot \mid s_t, a_t)$. This process is repeated until some terminal condition is met, such as reaching the time horizon T. The behavior of the agent

is defined by a policy $\pi : \mathcal{S} \mapsto Dist(\mathcal{A})$. This way, a policy π induces a distribution over full trajectories $\mathcal{T}_\pi = (s_0, a_0, s_1, \cdots)$ where $s_0 \sim \iota$, $a_t \sim \pi(\cdot \mid s_t)$, and $s_{t+1} \sim \mathcal{P}(\cdot \mid s_t, a_t)$. With assigned initial state-action pair $(s_0, a_0) = (s, a)$, the distribution over full trajectories is denoted as $\mathcal{T}'_\pi(s, a)$, where s, a are omitted if they are clear from the context.

In a CMDP there are two random variables of interest, the *return* $Z^r_\pi = \sum_{t=0}^{T} r(s_t, a_t)$ and the *cost-return* $Z^c_\pi = \sum_{t=0}^{T} c(s_t, a_t)$ that are, respectively, the sum of rewards and the sum of costs obtained in a trajectory following a fixed policy π.

Definition 2.1 (Safety based on Expected Value)**.** *A policy π is safe if its expected cost-return remains below a safety threshold d (Achiam et al., 2017; Yang, Simão, Tindemans, et al., 2023):*

$$\mathbb{E}\left[Z^c_\pi\right] \leq d$$

During training, the agent aims to learn a safe policy π that maximizes the expected return for each episode:

$$\max_\pi \mathbb{E}\left[Z^r_\pi\right] \quad \text{s.t.} \quad \mathbb{E}\left[Z^c_\pi\right] \leq d. \tag{2.1}$$

For a complex and long-horizon problem ($T \gg 1$), it is common to introduce a discount factor $\gamma \in (0.0, 1.0)$ to make the problem tractable, since it allows the agent to compute a single stationary value function, instead of indexing it by the time step. Henceforth, we consider the discounted *return* and discounted *cost-return*, accumulated discounted rewards and costs, respectively, from (s, a) as

$$\begin{aligned} Z^r_\pi(s, a) &= \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \mid s_0 = s, a_0 = a, \text{ and} \\ Z^c_\pi(s, a) &= \sum_{t=0}^{\infty} \gamma^t c(s_t, a_t) \mid s_0 = s, a_0 = a. \end{aligned} \tag{2.2}$$

We will refer to the cost-return $Z^c_\pi(s, a)$ as C whenever π, s and a are clear from the context. We have

$$\begin{aligned} Q^{r'}_\pi(s, a) &= \mathbb{E}_{(s_t, a_t) \sim \mathcal{T}'_\pi}\left[Z^r_\pi(s, a)\right], \text{ and} \\ Q^c_\pi(s, a) &= \mathbb{E}_{(s_t, a_t) \sim \mathcal{T}'_\pi}[Z^c_\pi(s, a)] = \mathbb{E}_{(s_t, a_t) \sim \mathcal{T}'_\pi}[C]. \end{aligned} \tag{2.3}$$

For safety-costs, we express the value function as $V^c_\pi(s) = \mathbb{E}_{(s_t, a_t) \sim \mathcal{T}'_\pi}[\sum_{t=0}^{\infty} \gamma^t c(s_t, a_t) | s_0 = s]$, and the advantage function for costs is $A^c_\pi(s, a) = Q^c_\pi(s, a) - V^c_\pi(s)$.

2.2. MAXIMUM ENTROPY RL WITH SAFETY CONSTRAINTS

When the agent knows nothing about the environment, the safety constraint cannot be strictly fulfilled during exploration. During the early steps of learning, we still hope to

encourage exploration to learn more about the environment. Maximum policy entropy can lead to diverse behaviors of the agent and better exploration. But the policy's entropy must be carefully balanced with the safety constraint, and the policy must be allowed to converge to a relatively deterministic policy, which reduces risks in terms of (safety-related) cost. Soft actor-critic (SAC; Haarnoja, Zhou, Abbeel, et al., 2018) is an off-policy method built on the actor-critic framework, which encourages agents to explore by including a policy's entropy as a part of the reward. SAC-based methods with entropy constraints and adaptive entropy weights (Haarnoja, Zhou, Hartikainen, et al., 2018) are candidates to meet the above conditions.

Building on policy entropy constraints and corresponding Lagrange multipliers in SAC (Haarnoja, Zhou, Hartikainen, et al., 2018; Haarnoja, Ha, et al., 2019), Ha et al. (2020) augment SAC with a safety constraint and introduce a second Lagrange multiplier to solve safety-constrained RL problems. We name this Lagrangian version of Soft Actor Critic SAC-Lag in the following parts of this dissertation. Although the SAC-Lag method was described as imposing a safety constraint at every time step, only a time-averaged constraint (with a single Lagrange multiplier) was implemented. This SAC-Lag method, which maximizes long-term rewards subject to policy entropy and safety constraints, is described below.

$$\max_{\pi} \mathop{\mathbb{E}}_{\substack{s_t\sim\mathcal{D}\\ a_t\sim\pi(\cdot|s_t)}} \left[Q_\pi^{r'}(s_t,a_t)\right]$$
$$\text{s.t.} \begin{cases} \mathop{\mathbb{E}}_{\substack{s_t\sim\mathcal{D}\\ a_t\sim\pi(\cdot|s_t)}} \left[Q_\pi^{c}(s_t,a_t)\right] \le \overline{d} \\ \mathop{\mathbb{E}}_{\substack{s_t\sim\mathcal{D}\\ a_t\sim\pi(\cdot|s_t)}} \left[-\log(\pi_t(a_t \mid s_t))\right] \ge h \quad \forall t \end{cases}, \tag{2.4}$$

where $\mathcal{D}$ is the replay buffer, h is the minimum entropy, and $\overline{d}$ is the discounted approximation of d. We define the discounted version of d as $\overline{d} = [(1-\gamma^{T_{\max}})d]/[(1-\gamma)T_{\max}]$, where we assume that equal cost $d/T_{\max}$ is accumulated at each step, and $T_{\max}$ is the maximum length of the episode. The assumption is not strictly correct, since we do not have equal costs at each step of a real episode, and often no costs are incurred early in the episode. However, since our algorithm optimizes the discounted infinite horizon from each state-action pair in the replay buffer, we should be approximately correct here. Notice that we have a global constraint on the cost-return (over trajectories) and a local constraint on the policy entropy (for each time step).

In general, SAC-Lag is a SAC-based method that has two critics, where we use the reward critic to estimate the expected return (possibly with entropy) to promote reward during learning, while the safety critic estimates the cost-return to encourage safety. In SAC-Lag, the constrained optimization problems are solved by Lagrangian methods (Bertsekas, 1982). To manage a trade-off among exploration, reward, and safety, adaptive

entropy and safety weights (Lagrange-multipliers) β and ω are introduced to the constrained optimization (2.1). With the policy entropy weight β, we have the soft Q-function (Haarnoja, Zhou, Abbeel, et al., 2018)

$$Q_\pi^r(s,a) = \mathop{\mathbb{E}}_{(s_t,a_t)\sim\mathcal{T}_\pi'}\left[Z_\pi^r(s,a) + \beta \sum_{t=1}^{\infty} \gamma^t \mathcal{H}(\pi(\cdot|s_t)) \right]. \tag{2.5}$$

Note that the soft Q-function Q_π^r is different from the Q-function $Q_\pi^{r'}$ in (2.3).

In this dissertation, we use J to denote loss functions, and θ to denote neural network parameters. Similar to the formulation used by Haarnoja, Zhou, Hartikainen, et al. (2018), we can get the actor loss:

$$J_\pi(\theta_\pi) = \mathop{\mathbb{E}}_{\substack{s_t\sim\mathcal{D}\\ a_t\sim\pi(\cdot|s_t)}} \left[\beta \log \pi(a_t \mid s_t) - Q_\pi^r(s_t,a_t) + \omega Q_\pi^c(s_t,a_t)\right], \tag{2.6}$$

where the entropy weight β (Lagrange multiplier) manages the stochasticity of the policy π and also determines the relative importance of the entropy term compared to rewards and costs. θ_π indicates the parameters of the policy π.

The safety and reward critics (including a bonus for the policy entropy) are, respectively, trained to minimize

$$J_C(\theta_C) = \mathop{\mathbb{E}}_{(s_t,a_t)\sim\mathcal{D}} \left[\frac{1}{2} \left(Q_{\theta_C}^c(s_t,a_t) - \left(c_t + \gamma Q_{\theta_C}^c(s_{t+1},a_{t+1}) \right) \right)^2 \right] \tag{2.7}$$

and

$$J_R(\theta_R) = \mathop{\mathbb{E}}_{(s_t,a_t)\sim\mathcal{D}} \left[\frac{1}{2} \left(Q_{\theta_R}^r(s_t,a_t) - \left(r_t + \gamma \left(Q_{\theta_R}^r(s_{t+1},a_{t+1}) - \beta \log(\pi(a_{t+1} \mid s_{t+1})) \right) \right) \right)^2 \right], \tag{2.8}$$

where $r_t = r(s_t,a_t)$, $c_t = c(s_t,a_t)$, and $a_{t+1} \sim \pi(\cdot \mid s_{t+1})$. Q^c and Q^r are parameterized by θ_C and θ_R, respectively.

Finally, let θ_ω and θ_β be the parameters learned for the safety and exploration weight such that $\omega = \text{softplus}(\theta_\omega)$ and $\beta = \text{softplus}(\theta_\beta)$, where

$$\text{softplus}(x) = \log\left(\exp(x) + 1\right).$$

We can learn ω and β by minimizing the loss functions

$$\begin{aligned} J_s(\theta_\omega) &= \mathop{\mathbb{E}}_{\substack{s_t\sim\mathcal{D}\\ a_t\sim\pi(\cdot|s_t)}} \left[\omega \left(\overline{d} - Q_\pi^c(s_t,a_t) \right) \right], \text{ and} \\ J_e(\theta_\beta) &= \mathop{\mathbb{E}}_{\substack{s_t\sim\mathcal{D}\\ a_t\sim\pi(\cdot|s_t)}} \left[-\beta \left(\log(\pi(a_t \mid s_t)) + h \right) \right], \end{aligned} \tag{2.9}$$

simultaneously with (2.6)-(2.8). This ensures the weights will rapidly grow if the constraints are violated, that is, if we estimate that the current policy is unsafe or if it does not have enough entropy.

2.3. DISTRIBUTIONAL RL BASED ON QUANTILE REGRESSION

So far, we considered only the expected value of the return and the cost return. In this section, we describe how we can estimate the full distribution of these random variables. Later, we will discuss how to use the tails of the cost-return to compute safer policies.

Distributional RL provides a means to estimate the return distribution instead of only modeling expected values (Bellemare, Dabney, and Munos, 2017; Dabney, Ostrovski, et al., 2018; Dabney, Rowland, et al., 2018; Yang, Zhao, et al., 2019). So it is natural to apply distributional RL in risk-averse domains. Even in traditional RL problems, distributional RL algorithms show better sample efficiency and ultimate performance compared to the standard expectation-based approach, but the state-of-the-art techniques have not been applied to safety-constrained RL with separate reward and safety signals.

Quantile regression, one of the main techniques in distributional RL, is widely used to estimate the return distribution, which has been combined with DQN (Mnih et al., 2015) to generate distributional variants such as QR-DQN (Dabney, Rowland, et al., 2018), IQN (Dabney, Ostrovski, et al., 2018), and FQF (Yang, Zhao, et al., 2019). In these methods, the difference between distributions is measured by 1-Wasserstein distance:

$$W_1(u, v) \doteq \int_0^1 |F_u^{-1}(x) - F_v^{-1}(x)| dx, \tag{2.10}$$

where u and v are random variables (e.g., the return or cost-return), and F is the cumulative distribution function (CDF). In these methods, we learn the inverse CDF of the return distribution, i.e., mapping quantile fraction $\tau \in [0, 1]$ to the corresponding quantile function value Z^τ [1], which can be expressed as $Z^\tau = F_Z^{-1}(\tau)$. QR-DQN, IQN, and FQF differ in how to generate the quantile fractions during training. Compared to fixing the quantile fractions (QR-DQN) and random sampling (IQN), we can theoretically better approximate the real distribution by using a proposal network (FQF) that generates appropriate quantile fractions for each state-action pair. However, IQN has been found to perform better in experiments and has fewer parameters to tune in complex environments (Ma et al., 2020). The quantile values of IQN are learned based on the Huber quantile regression

[1] In this section, Z stands for the return $Z_\pi^r(s, a)$, but this method can easily be adapted to estimate the cost-return distribution.

loss (Huber, 1964):

$$\mathcal{J}_\tau^\kappa(\varrho) = |\tau - \mathbb{I}\{\varrho < 0\}| \frac{\mathcal{L}_\kappa(\varrho)}{\kappa}, \text{ where } \mathcal{L}_\kappa(\varrho) = \begin{cases} \frac{1}{2}\varrho^2, & \text{if } |\varrho \le \kappa| \\ \kappa(|\varrho - \frac{1}{2}\kappa|), & \text{otherwise} \end{cases}, \tag{2.11}$$

where κ is the threshold to make the loss within the intervals $[-\kappa, 0]$ and $[0, \kappa]$ quadratic but a regular quantile loss if outside the interval. Based on the distributional Bellman operator (Sobel, 1982; Morimura et al., 2010; Tamar, Di Castro, and Mannor, 2016)

$$\mathcal{B}^\pi Z(s,a) \doteq r(s,a) + \gamma Z(s', a'), \tag{2.12}$$

we can get the TD error ϱ_{ij} between the quantile values at quantile fractions τ_i and τ'_j, i.e.,

$$\varrho_{ij} = r(s,a) + \gamma Z^{\tau'_j}(s', \pi(s')) - Z^{\tau_i}(s,a), \tag{2.13}$$

where (s, a, r, s') is sampled from the replay buffer $\mathcal{D}$, and $\pi(s) = \arg\max_{a \in \mathcal{A}} Q^{r'}(s,a)$ for a deterministic policy. Subsequently, with N and N' i.i.d. samples of $\tau, \tau' \sim U([0,1])$ respectively, we can get the loss function for IQN, i.e.,

$$\mathcal{J}(s,a,r,s') = \frac{1}{N'} \sum_{i=1}^{N} \sum_{j=1}^{N'} \mathcal{J}_{\tau_i}^\kappa(\varrho_t^{\tau_i, \tau'_j}). \tag{2.14}$$

To evaluate the policy, we can approximate $Q^{r'}(s,a)$ using K i.i.d. samples of $\tilde{\tau} \sim U([0,1])$:

$$Q^{r'}(s,a) \doteq \frac{1}{K} \sum_{k=1}^{K} Z^{\tilde{\tau}_k}(s,a). \tag{2.15}$$

It is important to note that τ, τ', and $\tilde{\tau}$ are sampled from continuous and independent distributions in IQN. τ' is for the TD target (average quantile values at several τ'), and τ is the given quantile we aim to estimate.

3

WORST-CASE CONSIDERATION

With separate reward and safety signals, it is natural to cast safety in RL as constraints, where expected cost-return of policies are constrained. However, it can be hazardous to set constraints on the expected safety signal without considering the tail of the distribution. For instance, in safety-critical domains, worst-case analysis is required to avoid disastrous results. We present a novel reinforcement learning algorithm called Worst-Case Soft Actor Critic, which extends the Soft Actor Critic algorithm with a safety critic to achieve risk control. More specifically, a certain level of conditional Value-at-Risk from the distribution is regarded as a safety measure to judge the constraint satisfaction, which guides the change of adaptive safety weights to achieve a trade-off between reward and safety. As a result, we can optimize policies under the premise that their worst-case performance satisfies the constraints. The empirical analysis shows that our algorithm attains better risk control compared to expectation-based methods.

3.1. INTRODUCTION

Ray, Achiam, and Amodei (2019) propose to make constrained RL the main formalism of safe exploration, where the reward function and cost function (related to safety) are distinct. This framework tries to mitigate the problem of designing a single reward function that needs to carefully select a trade-off between safety and performance, which is problematic in most instances. In addition, it is generally desirable to optimize the sample efficiency, i.e., to minimize the number of samples required to learn safe optimal

This chapter has been published in AAAI (2021) (Yang, Simão, Tindemans, et al., 2021).

policies. Off-policy methods can reuse past experience to be more sample efficient, and which safe exploration can benefit from (Mnih et al., 2015). Thus off-policy methods are preferred over on-policy methods, which need new experiences to evaluate a policy (Schulman, Levine, et al., 2015; Achiam et al., 2017; Schulman, Wolski, et al., 2017).

In this chapter, we focus on designing an off-policy algorithm for safety-constrained RL. Soft actor critic (SAC; Haarnoja, Zhou, Abbeel, et al., 2018; Haarnoja, Zhou, Hartikainen, et al., 2018) is an off-policy method built on the actor critic framework, which encourages agents to explore by including a policy's entropy as a part of the reward. SAC exhibits better sample efficiency and asymptotic performance compared to prior on-policy and off-policy methods. SAC-Lag (Ha et al., 2020) combines SAC with Lagrangian methods to address safety-constrained RL with local constraints, i.e., constraints are set for each timestep instead of each episode. The empirical analysis of SAC-Lag shows that the optimal policy with constraint-satisfying expected long-term costs can be learned with a low number of constraint violations. However, the cost of individual episodes might exceed the expected-cost bound with a high probability. For safety-critical problems, it can be hazardous to use the expected long-term costs as safety evaluation. Instead, better alternatives for safety-constrained RL are algorithms that compute policies based on varying risk requirements, specialized to risk-neutral or risk-averse behavior (Duan et al., 2020; Ma et al., 2020).

We propose the Worst-Case Soft Actor Critic (WCSAC) algorithm that uses a separate safety critic to estimate the distribution of accumulated cost to achieve risk control. We focus on the upper tail of the cost distribution, represented by the conditional Value-at-Risk (CVaR; Rockafellar and Uryasev, 2000). In this way, policies can be optimized given different levels of CVaR, which determine the degree of risk aversion from a safety perspective. In addition, we endow safety and entropy with weights that are automatically adapted according to the performance of current policies. Experimental analysis shows that by setting the level of risk control, the WCSAC algorithm attains stronger adaptability (compared to expectation-based baselines) when facing RL problems with higher safety requirements.

3.2. RISK-AVERSE CONSTRAINED RL

Traditional expectation-based safe RL methods maximize the return under the premise that the expected cost-return remains below the safety threshold d. In this way, RL agents are not aware of the potential risks because of the randomness in cost-return, which is generated by the stochastic policy and the dynamics of the environment. In expectation-based cases, if a safe policy has higher returns and higher variance in safety costs, it will be preferred over another safe policy with lower returns and lower variance in safety costs.

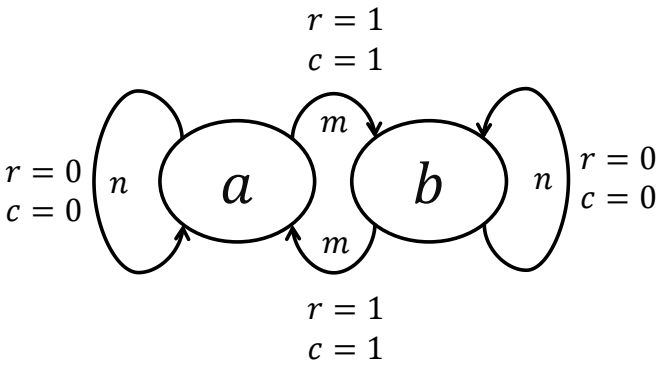


Figure 3.1: A CMDP example with $d = 1.8$ and $T = 2$. It has state space $\{a, b\}$ and action space $\{m, n\}$.

In safety-critical domains, the optimal policies are expected to be more robust, i.e., to have a lower risk of hazardous events even for stochastic or heavy-tailed cost-return.

An example of such a case is a simple CMDP shown in Figure 3.1. In each state, we can choose to move or not. If we choose to move (take action m), we will get a reward 1 and a cost 1, otherwise, both reward and cost will be 0. With an episode length of two timesteps and safety threshold $d = 1.8$ for each episode, the optimal policy will be $\pi(m|a) = 0.9, \pi(n|a) = 0.1, \pi(m|b) = 0.9, \pi(n|b) = 0.1$. But the real costs generated by the policy will be larger than the threshold with probability $p = 0.81$. Thus the optimal policy is hardly thought to be acceptable for safety-critical problems, even though it satisfies the constraint.

In Figure 3.2, the x-axis depicts the cost-return C (Equation (2.2)). The y-axis depicts the density of its probability distribution. The expectation-based algorithm focuses on the average performance in safety when optimizing policies. Thus, π, Q_π^c, and the shape of the cost-return distribution $p^\pi(C \mid s, a)$ will be changed during the training process until Q_π^c (blue line) is shifted to the left side of the boundary (red line). After that, there is still a strong likelihood that the constraint value d is exceeded. For a policy π, Q_π^c can only be used as the evaluation of average performance in safety, however, in safety-critical domains, the worst-case performance in safety is preferred over the average performance. Therefore, we replace the expected value with the Conditional Value-at-Risk (CVaR; Rockafellar and Uryasev, 2000), using the upper α of the distribution to assess the safety of a policy. In the right panel of Figure 3.2, we set the constraint on CVaR. Thus we optimize policies that will move the tail-end of $p^\pi(C \mid s, a)$ (blue line) to the left side of the boundary d (red line).

Definition 3.1 (Risk level.)**.** *A positive scalar $\alpha \in (0, 1]$ is used to define the risk level in*

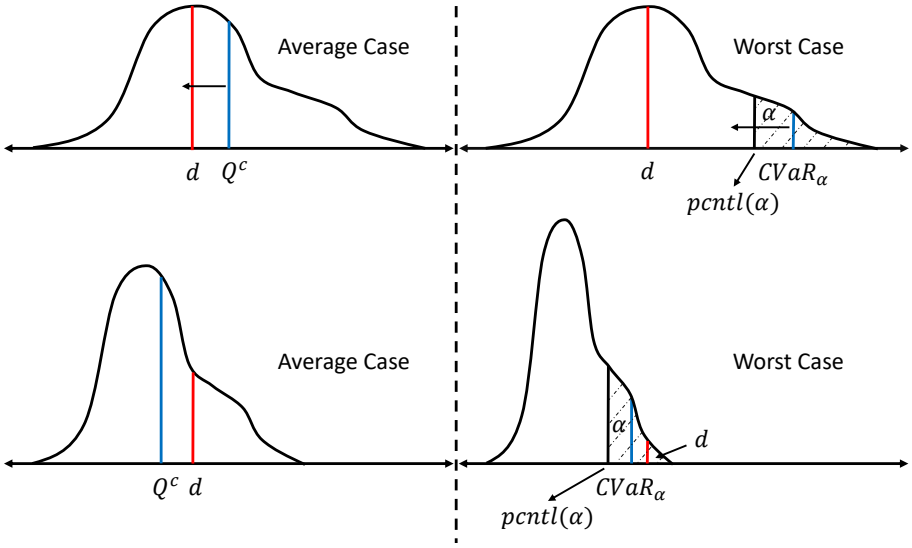


Figure 3.2: In the average case (left panel), the policies are optimized to ensure Q^c (blue line) is moved to the left side of the fixed boundary d (red line). In the worst case (right panel), we optimize policies to ensure the $CVaR_\alpha$ (blue line) measure on the left side of the fixed boundary d (red line).

WCSAC. A WCSAC with smaller α ($\alpha \to 0$) is expected to be more pessimistic and risk-averse. Conversely, a larger value of α leads to a less risk-averse behavior, with $\alpha = 1$ corresponding to the risk-neutral case.

Considering the probability distribution of cost-returns $p^\pi(C)$ induced by the aleatoric uncertainty of the environment and the policy π, we model the safety-constrained RL problem in a more risk-averse way than the traditional formulation (2.1). We focus on the α-percentile $F_C^{-1}(1-\alpha)$, where F_C is the CDF of $p^\pi(C \mid s,a)$, so we can get the CVaR:

$$\Gamma_\pi(s,a,\alpha) \doteq \text{CVaR}_\pi^\alpha(C) = \mathop{\mathbb{E}}_{p^\pi}[C \mid C \geq F_C^{-1}(1-\alpha)]. \tag{3.1}$$

The following definition gives us a new constraint to learn risk-averse policies, which differs from the traditional constraint (2.1).

Definition 3.2 (Safety based on CVaR)**.** *Given the risk level α, a policy π is safe if it satisfies $\Gamma_\pi(s_t,a_t,\alpha) \leq \overline{d} \quad \forall t$, where $(s_t,a_t) \sim \mathcal{T}_\pi$ and $s_0 \sim \iota$.*

Now we can generalize the framework from Section 2.2, using maximum entropy RL with the above risk-sensitive safety constraints. That is, the optimal policy in a constrained RL problem might be stochastic therefore it is reasonable to seek a policy with some

entropy (2.4). So, the policy is optimized to satisfy

$$\begin{aligned} \max_{\pi} \quad & \mathbb{E}\left[Z_\pi^r\right] \\ \text{s.t.} \quad & \text{CVaR}_\pi^\alpha(C) \leq \overline{d} \quad \text{and} \quad \mathop{\mathbb{E}}_{(s_t,a_t)\sim\mathcal{T}_\pi}\left[-\log(\pi_t(a_t \mid s_t))\right] \geq h \quad \forall\, t. \end{aligned} \tag{3.2}$$

With (3.2) it is possible to solve safe RL problems using the Soft Actor Critic (SAC; Haarnoja, Zhou, Abbeel, et al., 2018) framework, maintaining a minimum expected entropy (Haarnoja, Zhou, Hartikainen, et al., 2018).

3.3. WORST-CASE SOFT ACTOR CRITIC

To solve the *risk-averse constrained RL* problem (3.2), we design the Worst-Case Soft Actor Critic (WCSAC) algorithm. WCSAC generalizes SAC-Lag (Section 2.2), because SAC-Lag can be regarded as WCSAC with $\alpha = 1$, such that $\Gamma_\pi(s,a,1) = Q_\pi^c(s,a)$ (3.1). In this section, we start describing a safety critic that assumes the cost-return distribution is Gaussian, then we show how to optimize the actor with the new safety critics and present an overview of the full algorithm.

3.3.1. GAUSSIAN SAFETY CRITIC

In this section, we present how to obtain a Gaussian approximation of the safety critic. We will refer to the WCSAC with a Gaussian safety critic as WCSAC-GS in the following parts.

GAUSSIAN APPROXIMATION

WCSAC-GS uses a separate Gaussian safety critic (parallel to the reward critic for the return) to estimate the distribution of C instead of computing a point estimate of the expected cost-return, as the SAC-Lag algorithm. To obtain the cost-return distribution, $p^\pi(C \mid s,a)$ is approximated with a Gaussian, i.e.,

$$Z_\pi^c(s,a) \sim \mathcal{N}(Q_\pi^c(s,a), V_\pi^c(s,a)), \tag{3.3}$$

where $V_\pi^c(s,a) = \mathbb{E}_{p^\pi}[C^2 \mid s,a] - (Q_\pi^c(s,a))^2$ is the variance of the cost-return.

Given the Gaussian approximation (Khokhlov, 2016), the CVaR measure is easily computed. At each iteration, $Q_\pi^c(s,a)$ and $V_\pi^c(s,a)$ can be estimated (Tang, Zhang, and Salakhutdinov, 2020). Thus, the new safety measure for risk level α is computed by

$$\Gamma_\pi(s,a,\alpha) \doteq Q_\pi^c(s,a) + \alpha^{-1}\phi(\Phi^{-1}(\alpha))\sqrt{V_\pi^c(s,a)}, \tag{3.4}$$

where $\phi(\cdot)$ and $\Phi(\cdot)$ denote the probability distribution function (PDF) and the cumulative distribution function (CDF) of the standard normal distribution (Khokhlov, 2016).

WCSAC-GS learns the mean and variance of $p^{\pi}(C)$. To estimate Q_{π}^{c}, we can use the standard Bellman function:

$$Q_{\pi}^{c}(s,a) = c(s,a) + \gamma \sum_{s' \in S} p(s' \mid s,a) \sum_{a' \in A} \pi(a' \mid s') Q_{\pi}^{c}(s',a'). \tag{3.5}$$

The projection equation for estimating $V_{\pi}^{c}(s,a)$ is:

$$\begin{aligned} V_{\pi}^{c}(s,a) &= c(s,a)^2 - Q_{\pi}^{c}(s,a)^2 \\ &+ 2\gamma c(s,a) \sum_{s' \in S} p(s' \mid s,a) \sum_{a' \in A} \pi(a' \mid s') Q_{\pi}^{c}(s',a') \\ &+ \gamma^2 \sum_{s' \in S} p(s' \mid s,a) \sum_{a' \in A} \pi(a' \mid s') V_{\pi}^{c}(s',a') \\ &+ \gamma^2 \sum_{s' \in S} p(s' \mid s,a) \sum_{a' \in A} \pi(a' \mid s') Q_{\pi}^{c}(s',a')^2. \end{aligned} \tag{3.6}$$

We refer the reader to Tang, Zhang, and Salakhutdinov (2020) for the proof of (3.6).

GAUSSIAN SAFETY CRITIC LEARNING

WCSAC-GS uses two neural networks parameterized by θ_C^{μ} and θ_C^{σ}, respectively, to estimate the safety critic, i.e.,

$$Q_{\theta_C^{\mu}}^{c}(s,a) \to \hat{Q}_{\pi}^{c}(s,a) \text{ and } V_{\theta_C^{\sigma}}^{c}(s,a) \to \hat{V}_{\pi}^{c}(s,a).$$

In order to learn the safety critic, the distance between value distributions is measured by the 2-Wasserstein distance (Olkin and Pukelsheim, 1982; Bellemare, Dabney, and Munos, 2017): $W_2(u,v) \doteq \left(\int_0^1 |F_u^{-1}(x) - F_v^{-1}(x)|^2 dx \right)^{1/2}$, where $u \sim \mathcal{N}(Q_1, V_1)$, $v \sim \mathcal{N}(Q_2, V_2)$. WCSAC-GS uses the simplified 2-Wasserstein distance (Tang, Zhang, and Salakhutdinov, 2020) to estimate the safety critic loss:

$$W_2(u,v) = \|Q_1 - Q_2\|_2^2 + trace(V_1 + V_2 - 2(V_2^{1/2} V_1 V_2^{1/2})^{1/2}). \tag{3.7}$$

The 2-Wasserstein distance, in comparison to the 1-Wasserstein distance, incorporates the variance or spread of probability distributions, providing a more refined measure of dissimilarity that captures the geometric structure of the distributions, particularly beneficial for analyzing distributions with different variances or high-dimensional data. The 2-Wasserstein distance can be computed as the Temporal Difference (TD) error based on the projection equations (3.5) and (3.6) to update the safety critic, i.e., WCSAC-GS

minimizes the following values:

$$\begin{aligned} J_C^{\mu}(\theta_C^{\mu}) &= \mathop{\mathbb{E}}_{(s_t,a_t)\sim\mathcal{D}} \|\Delta Q(s_t,a_t,\theta_C^{\mu})\|_2^2, \text{ and} \\ J_C^{\sigma}(\theta_C^{\sigma}) &= \mathop{\mathbb{E}}_{(s_t,a_t)\sim\mathcal{D}} \text{trace}(\Delta V(s_t,a_t,\theta_C^{\sigma})), \end{aligned} \tag{3.8}$$

where $J_C^{\mu}(\theta_C^{\mu})$ is the loss function of $Q^c_{\theta_C^{\mu}}$, and $J_C^{\sigma}(\theta_C^{\sigma})$ is the loss function of $V^c_{\theta_C^{\sigma}}$. So,

$$\Delta Q(s_t,a_t,\theta_C^{\mu}) = \overline{Q}^c_{\theta_C^{\mu}}(s_t,a_t) - Q^c_{\theta_C^{\mu}}(s_t,a_t), \tag{3.9}$$

where $\overline{Q}^c_{\theta_C^{\mu}}(s_t,a_t)$ is the TD target from (3.5), and

$$\begin{aligned} \Delta V(s_t,a_t,\theta_C^{\sigma}) = \overline{V}^c_{\theta_C^{\sigma}}(s_t,a_t) + V^c_{\theta_C^{\sigma}}(s_t,a_t) \\ - 2(V^c_{\theta_C^{\sigma}}(s_t,a_t)^{1/2}\overline{V}^c_{\theta_C^{\sigma}}(s_t,a_t)V^c_{\theta_C^{\sigma}}(s_t,a_t)^{1/2})^{1/2}, \end{aligned} \tag{3.10}$$

where $\overline{V}^c_{\theta_C^{\sigma}}(s_t,a_t)$ is the TD target from (3.6).

3.3.2. WORST-CASE ACTOR

For a certain risk level α, we optimize the policy π until it satisfies the safety criterion $\Gamma_\pi(s_t,a_t,\alpha) \le \overline{d} \quad \forall t$ according to Definition 3.2. Based on the balance between safety and performance, the policy will be gradually adjusted according to the current policy evaluation $X^{\pi}_{\alpha,\omega}(s,a) = Q^r_\pi(s,a) - \omega\Gamma_\pi(s,a,\alpha)$ (Haarnoja, Zhou, Abbeel, et al., 2018). The role of safety changes over the training process. As the policy becomes safe, the influence of the safety term wanes, then the return optimization will play a greater role in our formulation.

In practice, based on the work by Haarnoja, Zhou, Abbeel, et al. (2018), we minimize the following KL divergence (Kullback and Leibler, 1951) to update the policy within a parametric space Π:

$$\begin{aligned} &\min_{\pi\in\Pi} D_{KL}\left(\pi(\cdot \mid s_t) \middle\| \frac{\exp\left(\frac{1}{\beta}(Q^r_\pi(s_t,\cdot) - \omega\Gamma_\pi(s_t,\cdot,\alpha))\right)}{\Lambda^{\pi}(s_t)}\right) \\ =&\min_{\pi\in\Pi} D_{KL}\left(\pi(\cdot \mid s_t) \| \exp\left(\frac{1}{\beta}X^{\pi}_{\alpha,\omega}(s_t,\cdot) - \log(\Lambda^{\pi}(s_t))\right)\right), \end{aligned} \tag{3.11}$$

where $\Lambda^{\pi}(s_t)$ is the partition function to normalize the distribution. β and ω are the adaptive entropy and safety weights, respectively. A loss function can be constructed by averaging the KL divergence over all states in the sample buffer and approximating the KL

divergence using a single sampled action, resulting in

$$\begin{aligned} &\mathop{\mathbb{E}}_{\substack{s_t\sim\mathcal{D}\\ a_t\sim\pi(\cdot|s_t)}}\left[\log\left(\frac{\pi(a_t\mid s_t)}{\exp\left(\frac{1}{\beta}X^{\pi}_{\alpha,\omega}(s_t,a_t)-\log(\Lambda^{\pi}(s_t))\right)}\right)\right] \\ =&\mathop{\mathbb{E}}_{\substack{s_t\sim\mathcal{D}\\ a_t\sim\pi(\cdot|s_t)}}\left[\log\pi(a_t\mid s_t)-\frac{1}{\beta}X^{\pi}_{\alpha,\omega}(s_t,a_t)+\log(\Lambda^{\pi}(s_t))\right]. \end{aligned} \tag{3.12}$$

3

$\Lambda^{\pi}(s_t)$ has no influence on updating θ, thus it can be omitted. The resulting actor loss is

$$J_{\pi}(\theta_{\pi})=\mathop{\mathbb{E}}_{\substack{s_t\sim\mathcal{D}\\ a_t\sim\pi(\cdot|s_t)}}\left[\beta\log\pi(a_t\mid s_t)-Q^{r}_{\pi}(s_t,a_t)+\omega\Gamma_{\pi}(s_t,a_t,\alpha)\right]. \tag{3.13}$$

We update the reward critic Q^r (2.8) and entropy weight β (2.9) in the same way as the SAC-Lag method in Section 2.2. Based on the new safety measure, the safety weight ω can be learned by minimizing the loss function:

$$J_s(\theta_{\omega})=\mathop{\mathbb{E}}_{\substack{s\sim\mathcal{D}\\ a\sim\pi(\cdot|s)}}\left[\omega(\overline{d}-\Gamma_{\pi}(s,a,\alpha))\right], \tag{3.14}$$

so ω will be decreased if $\overline{d}\geq\Gamma_{\pi}(s,a,\alpha)$, otherwise ω will be increased to emphasize safety more. The main difference to how SAC-Lag optimizes its policy and safety weight, is the use of the CVaR estimate, in opposite to the mean estimate in (2.6) and (2.9). We note that in (3.14), we sample from the replay buffer $\mathcal{D}$, whereas (2.2) suggests that the constraint applies to the initial state distribution. This replacement is certainly valid in the strongly discounted regime, or when episodes are very long. In this case, each visited state can be considered an initial state for the cost calculation. Although the replay buffer may initially be strongly off-policy, this deviation reduces over time. Moreover, this replacement also turns out to work well in practice when these conditions do not apply.

3.3.3. COMPLETE ALGORITHM

Algorithm 1 presents our method. At each environment step, the agent executes a new action sampled from the current policy and then proceeds to the next state. The experience will be stored in the replay buffer (lines 3-6). For the gradient descent steps, the method uses batches sampled from the replay buffer to update all function parameters (lines 7-17).

In standard maximum entropy RL, the entropy of the policy is expected to be as large as possible. However, relatively deterministic policies are preferred over stochastic policies in safe exploration, even though it is essential to encourage exploration during the early steps of learning. In SAC, the entropy of the policy is constrained to ensure that the

Algorithm 1 WCSAC-GS

Require: Hyperparameters α, d, h, η

1: **initialize** $\theta_\pi, \theta_R, \theta_C, \theta_\beta, \theta_\omega, \langle \overline{\theta}_R, \overline{\theta}_C \rangle \leftarrow \langle \theta_R, \theta_C \rangle, \mathcal{D} \leftarrow \emptyset$
2: **for** each iteration **do**
3: **for** each environment step **do**
4: $a_t \sim \pi(a_t \mid s_t)$, $s_{t+1} \sim \mathcal{P}(s_{t+1} \mid s_t, a_t)$
5: $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t, a_t, r(s_t, a_t), c(s_t, a_t), s_{t+1})\}$
6: **end for**
7: **for** each gradient step **do**
8: Sample experience from replay buffer $\mathcal{D}$
9: $\theta_R \leftarrow \theta_R - \lambda_R \hat{\nabla}_{\theta_R} J_R(\theta_R)$ ▷ Reward critic (2.8)
10: $\theta_C \leftarrow \theta_C - \lambda_C \hat{\nabla}_{\theta_C} J_C(\theta_C)$ ▷ Safety critic (3.8)
11: Compute Γ_{π_θ} ▷ CVaR estimate (3.4)
12: $\theta_\pi \leftarrow \theta_\pi - \lambda_\pi \hat{\nabla}_{\theta_\pi} J_\pi(\theta_\pi)$ ▷ Actor (3.13)
13: $\theta_\beta \leftarrow \theta_\beta - \lambda_\beta \hat{\nabla}_{\theta_\beta} J_e(\theta_\beta)$ ▷ Exploration weight (2.9)
14: $\theta_\omega \leftarrow \theta_\omega - \lambda_\omega \hat{\nabla}_{\theta_\omega} J_s(\theta_\omega)$ ▷ Safety weight (3.14)
15: $\overline{\theta}_R \leftarrow \eta \theta_R + (1-\eta)\overline{\theta}_R$
16: $\overline{\theta}_C \leftarrow \eta \theta_C + (1-\eta)\overline{\theta}_C$
17: **end for**
18: **end for**

Output: Optimized parameters $\theta_\pi, \theta_R, \theta_C, \theta_\beta, \theta_\omega$

final optimal policy is more robust (Haarnoja, Zhou, Hartikainen, et al., 2018). Therefore, for safety-critical domains, it is preferred to set a relatively low minimum requirement $\mathcal{H}_0$ for the entropy, or omit this constraint altogether.

For the reward critic, to avoid overestimation and reduce the positive bias during the policy improvement process, we also learn two soft Q-functions independently, which are parameterized by θ_{R1} and θ_{R2}. The minimum Q-function is used in each gradient step. For the safety critic, we use two separate neural networks to estimate the mean function and variance function respectively. The size of each network can be smaller than using one network to estimate the mean function and variance together, so it does not add more parameters to be trained. Besides, it is much easier to compare the distributional safety critic of WCSAC to the regular safety critic of SAC-Lag, which can be seen as ablation of WCSAC. We use four target networks to achieve stable updating, a common technique used in DQN (Mnih et al., 2015) and DDPG (Lillicrap et al., 2015). Specifically, the parameters of target networks (including safety critic and reward critic) are updated by moving averages (lines 15-16), where hyperparameter $\eta \in [0, 1]$ is used to reduce fluctuations.

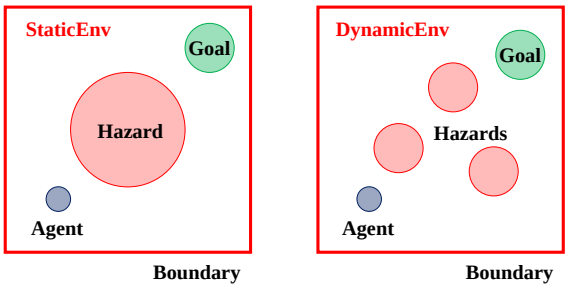
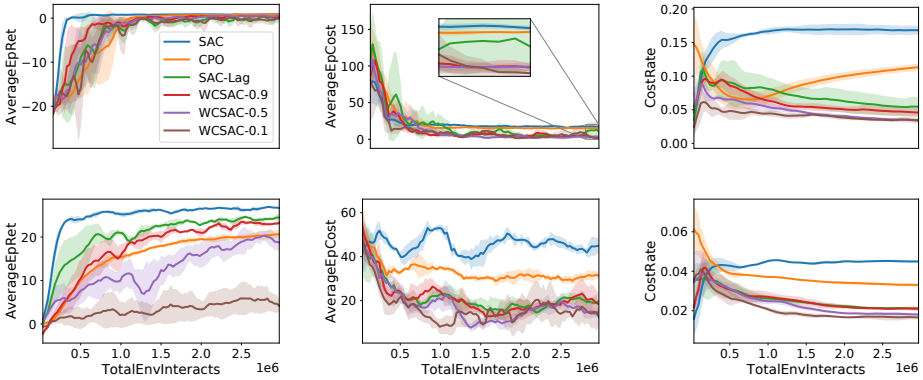


Figure 3.3: Point navigation domains StaticEnv and DynamicEnv. The environments differ in the number and size of hazards, and generation of goal and hazards' locations.

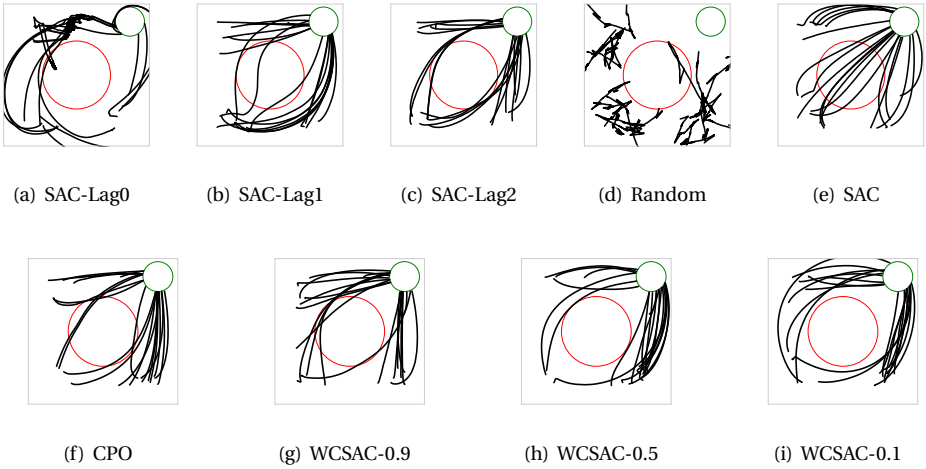


(a) Average episodic return (b) Average episodic cost (c) Cost rate

Figure 3.4: Comparison of SAC, CPO, SAC-Lag, and WCSAC during training in StaticEnv (top row) and DynamicEnv (bottom row). The lines are the average of all runs, and the shaded area is the standard deviation.

3.4. EMPIRICAL ANALYSIS

We evaluate our method on the Safety Gym benchmark (Ray, Achiam, and Amodei, 2019). In these environments (see Figure 3.3) a point agent (one actuator for turning and another one for moving forward/backward) navigates in a 2D map to reach the goal position while trying to avoid hazardous areas. In StaticEnv (Figure 3.3 left), the agent gets a reward $r - 0.2$ in each step, where r is the original reward signal of Safety Gym (distance towards goal plus a constant for being within range of goal), and the offset -0.2 incentivizes the agent to reach the target in the smallest number of time steps. DynamicEnv keeps the original reward signal. In both environments, the initial state of the agent is randomly initialized in each episode. The episodic locations of goal and hazards are also arbitrarily generated in DynamicEnv but fixed in StaticEnv. In each step, if the agent stays in the hazardous area, it incurs a cost $c = 1$, otherwise $c = 0$. We use a discount factor $\gamma = 0.99$.



(a) SAC-Lag0 (b) SAC-Lag1 (c) SAC-Lag2 (d) Random (e) SAC

(f) CPO (g) WCSAC-0.9 (h) WCSAC-0.5 (i) WCSAC-0.1

Figure 3.5: Trajectory analysis. (a)-(c) show the trajectories generated by policies from SAC-Lag at the beginning, middle stage and end of training respectively. (d)-(i) show the final trajectories from random policy, SAC, CPO, and WCSAC separately.

We set $d = 15$ for the expected (real-world) cost limit, which needs to be reverse-discounted in SAC-Lag and WCSAC. All the agents are trained for 100 epochs, where the length of each epoch is 30000 environment interaction steps and the maximal length of each episode is 1000 environment interaction steps. Furthermore, all experiments were run with three random seeds. Specifically, we will compare SAC, CPO (Achiam et al., 2017), SAC-Lag, and WCSAC with different risk levels in safety, i.e., WCSAC-0.1 ($\alpha = 0.1$: highly risk-averse), WCSAC-0.5 ($\alpha = 0.5$) and WCSAC-0.9 ($\alpha = 0.9$: almost risk-neutral). Four CPUs were used in parallel for training in all cases. In our following experimental results, the shown return and cumulative cost are undiscounted.

3.4.1. RESULTS

During training, we use the following metrics: average episodic return, average episodic cost, and cost rate (Ray, Achiam, and Amodei, 2019). The cost rate at each epoch is computed by dividing the cumulative costs by the number of environment interaction steps. Figure 3.4 shows the results. We observe that all algorithms find policies that can reach the goal at the end of the training (Figure 3.4(a)), but with different convergence rates. Compared to safe methods (SAC-Lag, CPO, and WCSAC), SAC has better and more stable performance in average episodic return obviously, however it does not satisfy the constraint. Regarding safety, Figures 3.4(b) and 3.4(c) show that all safe methods except for CPO converge to constraint-satisfying policies, and WCSAC-0.9 performance is close

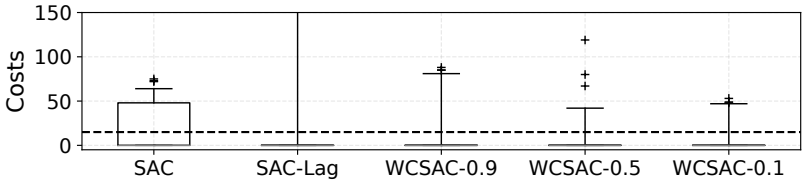
	EC	C0.9	C0.5	C0.1	ER
SAC	21.7	24.1	42.8	56.5	0.97
SAC-Lag	**14.3**	15.9	28.6	141.8	0.27
WCSAC-0.9	4.2	**4.6**	8.4	31.4	0.56
WCSAC-0.5	1.8	2.0	**3.6**	17.9	0.19
WCSAC-0.1	1.4	1.6	2.9	**14.3**	-0.43
CPO	**16.3**	18.1	32.5	58.3	0.84
Random Policy	49.1	54.5	98.1	431.6	-20.10

Table 3.1: Performance of the agents on the StaticEnv according to multiple metrics: expected cost (EC), cost-CVaR-0.9 (C0.9), cost-CVaR-0.5 (C0.5), cost-CVaR-0.1 (C0.1), and expected return (ER). In bold we indicate the constraint used by the agent.

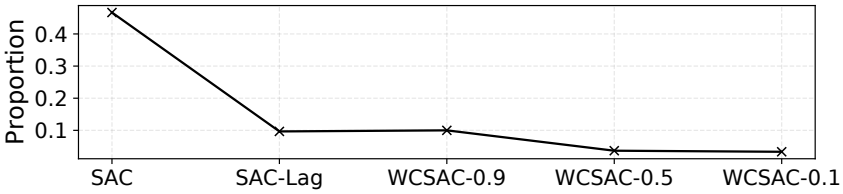
to SAC-Lag. Besides, WCSAC with a lower risk level generates fewer constraint violations during training.

We execute a trajectory analysis in StaticEnv, see Figure 3.5. We only show the training process of SAC-Lag at different stages because other methods show similar behavior during training. At the beginning of learning (Figure 3.5(a)), it is possible that the agent cannot get out of the hazard, and gets stuck before arriving at the goal area. In Figure 3.5(d), the trajectories of random policy are highly chaotic. The final policies from SAC-Lag, CPO, and WCSAC-0.9 perform better than before, but still prefer to take a risk within the budget to get a larger return (Figures 3.5(c), 3.5(f) and 3.5(g)). Conversely, the agent from WCSAC-0.5 becomes more risk-averse in Figure 3.5(h). The behavior of the SAC agent is presented in Figure 3.5(e), the agent chooses the shortest path to reach the target directly since the safety constraint is not considered. Finally, in Figure 3.5(i), we can see that the agent from WCSAC-0.1 prefers to stay away from the hazardous area given its risk level setting.

After training, we use 300 test runs (100 runs for each random seed) to evaluate the final policies of these algorithms. In Table 3.1, the results show that each WCSAC variation satisfies its corresponding CVaR bound (estimated by the average costs of the worst 300α trajectories), while only WCSAC-0.1 results in $CVaR_{0.1} < 15$. In Figure 3.6, we compare the SAC-based methods. The whiskers of the boxplot Figure 3.6(a) are set at the $[1, 99]$ percentiles of the data. To make the boundary d more clear, we set the y-axis view limits (so the data of SAC-Lag is out of the chart). The optimal policies learned by SAC have poor performance in safety without considering the constraint. The optimal policies from SAC-Lag can ensure that most of the trajectories are safe, but some dangerous events happen, which is undesirable for safety-critical problems. As to the proportion of budget exceedance, WCSAC-0.9 has a similar average performance to SAC-Lag, but the boxplot shows that extreme cost events are much less likely to happen. Compared to SAC-Lag and WCSAC-0.9, WCSAC with lower risk levels has a more stable performance in terms of



(a) Boxplot of safety evaluation after training



(b) Proportion of budget exceedance

Figure 3.6: Evaluation after training (StaticEnv). The boxplot (a) shows the statistical properties of long-term costs generated by the SAC-based algorithms, and the dashed line indicates the safety threshold. In (b), we use the proportion of budget exceedance to analyse the safety of the SAC-based algorithms.

safety. Although the policies from WCSAC-0.1 and WCSAC-0.5 still generate some unsafe trajectories, the likelihood is much lower.

3.5. CONCLUSION

In this chapter, we propose the WCSAC algorithm to solve safety-constrained RL problems. We augment SAC with a separate distributional safety critic (parallel to the reward critic) to make the algorithm more adaptive when facing RL problems with higher safety requirements. In the experiments, the agent has different performance in safety under different risk levels. When $\alpha \ll 1$, we can get more risk-averse policies for a safety-critical domain. Hence, our research is meaningful for the development of safe exploration. In this chapter, the distribution of long-term costs is approximated to be a Gaussian distribution. In the future, we can further explore modeling the uncertainty of the safety critic in different ways. Moreover, the focus of this chapter has been on safety, but a similar approach could be taken to consider the variability of long-term rewards. It is then also interesting to consider trade-offs between a distributional reward critic and a distributional safety critic.

4

Refined Risk Management

In this chapter, we propose an improved WCSAC that uses an implicit quantile network (WCSAC-IQN) to approximate the distribution of accumulated safety-costs. Chapter 3 described WCSAC with a Gaussian safety critic (WCSAC-GS), which can attain better risk control compared to expectation-based methods. However, the total cost-return distribution of different trajectories is still largely unexplored. The Gaussian approximation can be coarse in many domains, especially when the cost-return distribution has long tails. Using an accurate estimate of the cost-return distribution, in particular of the upper tail of the distribution, greatly improves the performance of risk-averse RL agents. The empirical analysis shows that WCSAC-IQN achieves good risk control in complex safety-constrained environments.

4.1. Introduction

Following the work in Chapter 3, we model the problem using a risk-averse constrained Markov decision process (CMDP; Altman, 1999), where the safety constraints are expressed by the conditional value at risk (CVaR; Rockafellar and Uryasev, 2000). However, in Chapter 3, we estimate the distribution of accumulated costs using a coarse Gaussian approximation. While this might be reasonable for some tasks, in some situations this can lead to an underestimation of the CVaR, leading the agent to an unsafe behavior. Considering such situations, one should compute a more refined distribution of the safety-costs to ensure the safety constraints are satisfied even in extreme scenarios.

This chapter has been published in Machine Learning (2023) (Yang, Simão, Tindemans, et al., 2023).

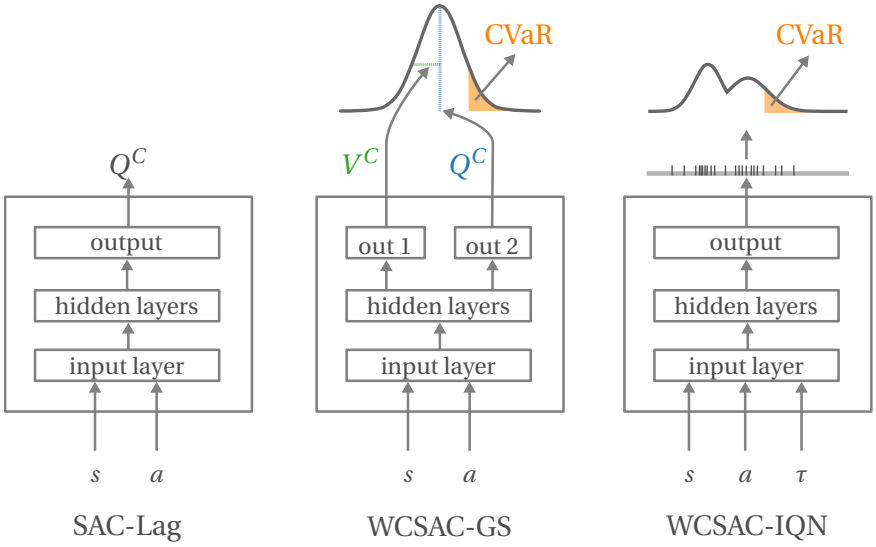


Figure 4.1: Overview of the safety critics. The traditional safety critic only estimates the average of the cost-return distribution Q^C, while the critics of the WCSAC algorithms keep track of the full distribution, and estimate the CVaR for the safety constraints. The first assumes the distribution is Gaussian and the second uses the IQN algorithm for quantile regression.

While distributional RL method (Bellemare, Dabney, and Munos, 2017; Dabney, Ostrovski, et al., 2018) have shown great improvements with respect to its non-distributional counterparts, for instance, controlling a balloon in the stratosphere (Bellemare, Candido, et al., 2020), the use of the distribution of the costs in the safe RL literature still remains largely under-explored. Some exceptions include risk-averse RL algorithms to ensure the agent maintains a reasonable performance (Chow et al., 2017) and the use of an implicit quantile network (IQN; Dabney, Ostrovski, et al., 2018) in an offline setting to estimate the distribution of returns instead of only the average, making the final policy more pessimistic (Urpí, Curi, and Krause, 2021). Inspired by these results, we use IQN to compute a distributional safety critic, which provides a more precise estimate of the upper tail part of the distribution, as illustrated on Figure 4.1 (Right).

The improved version of WCSAC extends IQN (WCSAC-IQN), originally designed for discrete action spaces, to safe RL with continuous action spaces, and addresses the key limitation of WCSAC-GS in Chapter 3 caused by the approximation errors, resulting in a general framework for safe RL. In this case, we show the versatility of WCSAC by using two methods to approximate the cost-return distribution. To validate the efficiency of WCSAC-IQN, we design two toy games: Spy-Unimodal (approximately Gaussian) and Spy-Bimodal (non-Gaussian), where WCSAC-IQN has significant advantages in the non-Gaussian case. When deploying the algorithms on large and complex standard environments, WCSAC-

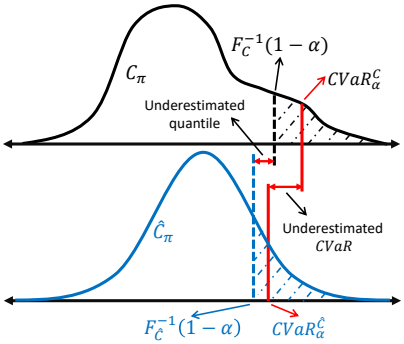


Figure 4.2: Unreliability of Gaussian approximation. The top curve depicts the true cost distribution, while the bottom curve depicts the estimated Gaussian distribution, based on the correct mean and standard deviation. In this case, the $(1-\alpha)$-quantile and corresponding CVaR are underestimated.

IQN attains outstanding performance compared to other safe RL baselines. After limited training, only the WCSAC-IQN agent attains constraint-satisfying policies.

4.2. SAFETY CRITIC WITH QUANTILE REGRESSION

Although the Gaussian approximation leverages distributional information to attain more risk-averse policies, only an additional variance is estimated compared to regular constrained RL methods. This means the information of the experiences collected are only used to a limited extent. Thus, the Gaussian approximation does not possess the general advantages of distributional RL algorithms.

Besides, it is not always appropriate to approximate the cost-return by a Gaussian distribution, as shown in Figure 4.2, since the contribution from the tail of the cost distribution might be underestimated. In this case, the agent might converge to an unsafe policy, according to Equation 3.2. In this section, we present a distributional safety critic modeled by IQN, as illustrated on Figure 4.1, which provides a more precise estimate of the upper tail part of the distribution. Henceforth, we refer to WCSAC with a safety critic modeled by IQN as WCSAC-IQN.

4.2.1. ESTIMATING SAFETY CRITIC WITH IQN

We propose to use an *implicit quantile network* to model the cost-return distribution (WCSAC-IQN), regarded as the safety critic of the SAC method. WCSAC-IQN maps the samples from a base distribution (usually $\tau \sim U([0,1])$) to the corresponding quantile values of the cost-return distribution. In theory, by adjusting the capacity of the neural network, WCSAC-IQN can fit the cost-return distribution with arbitrary precision, which

is essential for safety-critical problems.

We denote $F_C^{-1}(\tau)$ as the quantile function for the cost-return C and, for clarity of exposition, we define $C^\tau = F_C^{-1}(\tau)$. We use θ_C to parameterize the safety critic in WCSAC-IQN. The approximation is implemented as $\hat{C}^\tau(s,a) \leftarrow f_{IQN}(s,a,\tau \mid \theta_C)$, which also takes the quantile fraction τ as the input of the model, so that it uses the neural network to fit the entire continuous distribution. When training f_{IQN}, two quantile fraction samples $\tau, \tau' \sim U([0,1])$ at time step t are used to get the sampled TD error:

$$\varrho_t^{\tau,\tau'} = c_t + \gamma C^{\tau'}(s_{t+1}, a_{t+1}) - C^\tau(s_t, a_t). \tag{4.1}$$

4

Following Equation (2.11), we can get the loss function for the safety critic, i.e.,

$$J_C(\theta_C) = \mathop{\mathbb{E}}_{(s_t,a_t,c_t,s_{t+1})\sim\mathcal{D}} \mathcal{J}_C(s_t, a_t, c_t, s_{t+1} \mid \theta_C), \tag{4.2}$$

where

$$\begin{aligned}
\mathcal{J}_C(s_t, a_t, c_t, s_{t+1} \mid \theta_C) &\underset{(a)}{=} \sum_{i=1}^{N} \mathop{\mathbb{E}}_{\mathcal{B}^\pi C} [\mathcal{J}_{\tau_i}^\kappa (\mathcal{B}^\pi C(s_t, a_t) - C^{\tau_i}(s_t, a_t))] \\
&\underset{(b)}{=} \sum_{i=1}^{N} \mathop{\mathbb{E}}_{C} [\mathcal{J}_{\tau_i}^\kappa (c_t + \gamma C(s_{t+1}, a_{t+1}) - C^{\tau_i}(s_t, a_t))] \\
&\underset{(c)}{\doteq} \frac{1}{N'} \sum_{i=1}^{N} \sum_{j=1}^{N'} \mathcal{J}_{\tau_i}^\kappa (c_t + \gamma C^{\tau'_j}(s_{t+1}, a_{t+1}) - C^{\tau_i}(s_t, a_t)) \\
&\underset{(d)}{=} \frac{1}{N'} \sum_{i=1}^{N} \sum_{j=1}^{N'} \mathcal{J}_{\tau_i}^\kappa (\varrho_t^{\tau_i,\tau'_j}).
\end{aligned} \tag{4.3}$$

In Equation 4.3: $\mathcal{J}_\tau^\kappa$ is the quantile regression loss (2.11), (a) indicates that the total loss of all the target quantiles $\tau_i, i = 1, \cdots, N$ is computed at once, and applies the distributional Bellman operator $\mathcal{B}$ (Bellemare, Dabney, and Munos, 2017), (b) expands the Bellman operator $\mathcal{B}^\pi C(s_t, a_t)$ to get $c_t + \gamma C(s_{t+1}, a_{t+1})$, taking an action for the next state sampled from the current policy $a_{t+1} \sim \pi(\cdot \mid s_{t+1})$, (c) introduces τ_j to estimate the TD target, and (d) uses Equation 4.1. We point out that for the estimation of quantiles, the quantile loss is replaced by the Huber loss to ease training, as in the regular IQN method (Dabney, Ostrovski, et al., 2018). However, this may lead to a bias in the safety distribution (Rowland et al., 2019), especially for larger values of κ. The imputation approach proposed in the work by (Rowland et al., 2019) can be combined with the proposed method to reduce this bias. Investigation of the extent of the bias and the efficacy of the correction in risk-averse RL is a subject for future research.

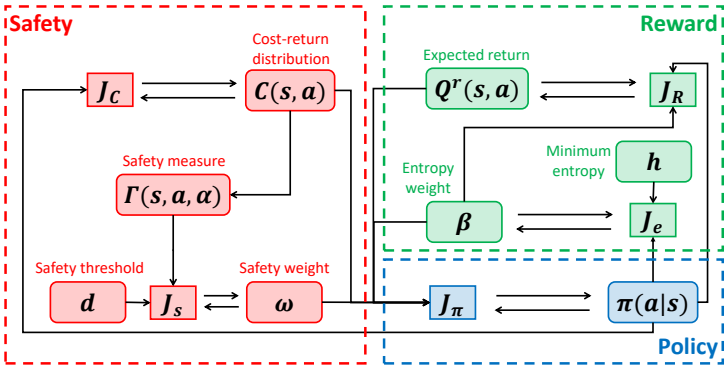


Figure 4.3: WCSAC's structure. The elements are grouped by function: Safety, Reward, and Policy.

4

4.2.2. CVaR SAFETY MEASURE BASED ON SAMPLE MEAN

Since we base our estimate of the distribution of cost-return on a quantile-parameterized approximation, we approximate the CVaR based on the expectation over the values of the quantile τ as $\Gamma_\pi(s,a,\alpha) \doteq \mathbb{E}_{\tau\sim U([1-\alpha,1])}\left[C^\tau_\pi(s,a)\right]$. This allows us to estimate $\Gamma_\pi(s,a,\alpha)$ at each update step using K i.i.d. samples of $\tilde{\tau} \sim U([1-\alpha,1])$:

$$\Gamma_\pi(s,a,\alpha) \doteq \frac{1}{K}\sum_{k=1}^{K} C^{\tilde{\tau}_k}_\pi(s,a). \tag{4.4}$$

While the Gaussian approximation leverages a closed-form approach to estimate the CVaR, which is inherently limited by the Gaussian assumption, our method efficiently estimates the CVaR using a sampling approach. This can attain higher accuracy due to the quantile regression framework. We also highlight that this method still estimates the full distribution, sampling τ, τ' from $U([0,1])$ to compute the safety critic loss. We use Equation 4.4 only when estimating the CVaR to compute the Lagrangian safety loss J_s.

Figure 4.3 shows a general overview of the proposed algorithm, indicating the relations between safety, reward, and policy components. The arrows depict the relations between all terms in the method, i.e., the element at the beginning of an arrow influences the element at its end. We may notice that the safety and reward terms only influence each other through the policy.

4.2.3. COMPLETE ALGORITHM

The complete algorithm WCSAC-IQN is presented in Algorithm 2, where we list the input of the algorithm and all initialization objects in line 1. Under a certain safety requirement α, we input $\langle d, h\rangle$ for the constraints. With the WCSAC-IQN, we also need

Algorithm 2 WCSAC-IQN

Require: Hyperparameters α, d, h, η, N, N', K

1: **initialize** $\theta_\pi, \theta_R, \theta_C, \theta_\beta, \theta_\omega, \langle\overline{\theta}_R, \overline{\theta}_C\rangle \leftarrow \langle\theta_R, \theta_C\rangle, \mathcal{D} \leftarrow \emptyset$
2: **for** each iteration **do**
3: **for** each environment step **do**
4: $a_t \sim \pi(a_t \mid s_t)$, $s_{t+1} \sim \mathcal{P}(s_{t+1} \mid s_t, a_t)$
5: $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t, a_t, r(s_t, a_t), c(s_t, a_t), s_{t+1})\}$
6: **end for**
7: **for** each gradient step **do**
8: Sample experience from replay buffer $\mathcal{D}$
9: Sample $\tau_i \sim U([0,1])$, $\tau'_j \sim U([0,1])$, and $\widetilde{\tau}_k \sim U([\alpha,1])$
10: $\theta_R \leftarrow \theta_R - \lambda_R \hat{\nabla}_{\theta_R} J_R(\theta_R)$ ▷ Reward critic (2.8)
11: $\theta_C \leftarrow \theta_C - \lambda_C \hat{\nabla}_{\theta_C} J_C(\theta_C)$ ▷ Safety critic (4.2)
12: Compute Γ_{π_θ} ▷ CVaR estimate (4.4)
13: $\theta_\pi \leftarrow \theta_\pi - \lambda_\pi \hat{\nabla}_{\theta_\pi} J_\pi(\theta_\pi)$ ▷ Actor (3.13)
14: $\theta_\beta \leftarrow \theta_\beta - \lambda_\beta \hat{\nabla}_{\theta_\beta} J_e(\theta_\beta)$ ▷ Exploration weight (2.9)
15: $\theta_\omega \leftarrow \theta_\omega - \lambda_\omega \hat{\nabla}_{\theta_\omega} J_s(\theta_\omega)$ ▷ Safety weight (3.14)
16: $\overline{\theta}_R \leftarrow \eta\theta_R + (1-\eta)\overline{\theta}_R$
17: $\overline{\theta}_C \leftarrow \eta\theta_C + (1-\eta)\overline{\theta}_C$
18: **end for**
19: **end for**

Output: Optimized parameters $\theta_\pi, \theta_R, \theta_C, \theta_\beta, \theta_\omega$

the hyperparameters $\langle N, N'\rangle$ for updating the safety critic, K for computing the new safety measure Γ_π. For the environment steps (lines 3-6), we sample actions from the policy to attain experience for the replay buffer $\mathcal{D}$, which allows us to get batches for updating all parameters at each gradient descent step (lines 7-18). After line 19, we return all the optimized parameters of the algorithm.

As to the neural network structure of safety critic, we use the same function as in IQN for return (Dabney, Ostrovski, et al., 2018), i.e., a DQN-like network with an additional embedding for the quantile fraction τ. When selecting the learning rate for the neural networks (λ_R, λ_C, λ_π, λ_β, and λ_ω) which are used to minimize the corresponding loss functions (J_R, J_C, J_π, J_β, and J_ω), we usually make λ_ω larger than the others to enhance the safety constraint. A relatively low learning rate for the safety weight does not converge fast enough to improve the safety of the actor's policy, but the practical learning rate should be set according to the environment. Typically, the disparity between λ_ω and the remaining learning rates (λ_R, λ_C, λ_π, and λ_β) will be more pronounced in more complex and safety-critical environments.

4.3. EMPIRICAL ANALYSIS

In this section, we evaluate our methods WCSAC-GS and WCSAC-IQN on the tasks with different difficulties, i.e., two SpyGame environments and a Safety Gym benchmark (Ray, Achiam, and Amodei, 2019). This section has three goals: *i*) test the hypothesis that WCSAC-IQN can achieve good risk control in an environment with a *Gaussian* cost-return distribution; *ii*) test the hypothesis that WCSAC-IQN can find a safe policy on environments with a *non-Gaussian* cost-return distribution when equipped with an appropriate estimate of the distribution; and *iii*) evaluate the performance of the proposed method in *highly-complex environments*.

4.3.1. SPYGAME ENVIRONMENTS

The SpyGame is a toy model, meant to give rise to unimodal (approximately Gaussian) and bimodal cost distributions. To test whether the two WCSAC algorithms can achieve safe behavior in environments with a Gaussian cost-return, and whether WCSAC-IQN indeed has better performance in environments with non-Gaussian cost-return distribution, we designed two SpyGame environments: Spy-Unimodal and Spy-Bimodal. For any policy, Spy-Unimodal leads to a unimodal cost-return distribution (approximately Gaussian), while Spy-Bimodal has a bimodal cost-return distribution (non-Gaussian). For this model, we consider an agency that trains spies to go on covert missions. On each mission, the spy gets a random amount of useful information (the reward) and leaves some traces (the cost). If too many clues to the spy's identity are left across missions, the spy is likely to get discovered. In order to control the risk of discovery, a safety constraint is implemented on cumulative cost. For each mission, the spy has a choice of low-risk, low-reward, and high-risk, high-reward approaches, parameterized by the action $a \in [0,1]$. For a choice of a, random rewards and costs are drawn from uniform distributions as follows (Figure 4.4):

$$r(a) \sim U(-0.25 + a, 0.75 + a + 0.5a^2) \tag{4.5}$$

$$c(a) \sim U(0.5a, 1.5a). \tag{4.6}$$

Two variants of the game are implemented in the `SpyEnv` environment, which are named *Spy-Unimodal* and *Spy-Bimodal* according to the shape of their cost distributions.

Spy-Unimodal: Each spy executes 100 missions until retirement. The aim is to maximize the expected reward subject to a cost constraint (in expectation or CVaR). The cumulative costs are a linear sum of a large number of independent random variables, so they are approximately normally distributed.

Spy-Bimodal: In this variant, spies face early retirement if they do not gain sufficiently useful information. After 5 missions, a stopping criterion is evaluated that terminates the

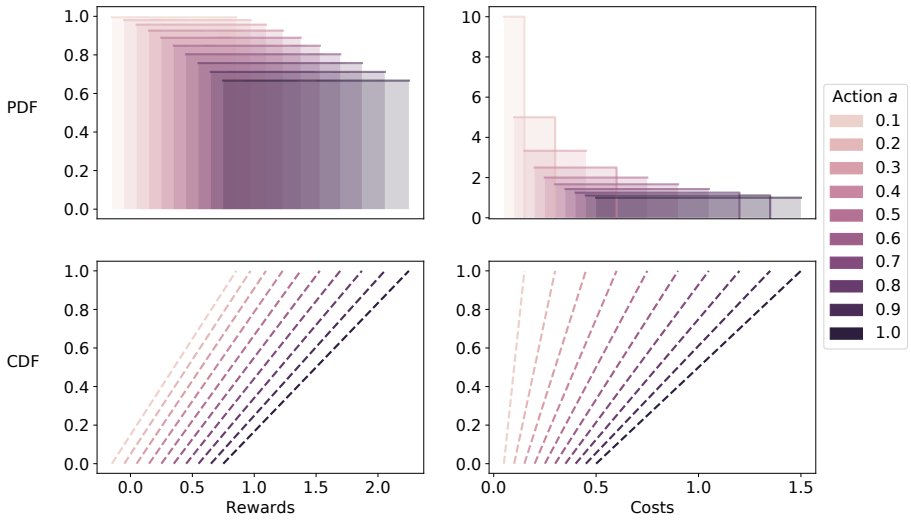


Figure 4.4: The PDF and CDF of the reward and cost functions under different actions.

game unless the *average* reward per mission exceeds 0.15. This results in a significant fraction of spies retiring early, which is reflected in a bimodal cost distribution.

We set the safety thresholds $d = 25$ for Spy-Unimodal, and $d = 15$ for Spy-Bimodal. We use WCSAC-GS and WCSAC-IQN with risk-neutral and risk-averse constraints (cost-CVaR-α) to solve both variations of the SpyGame. Each algorithm uses small neural networks (2 layers with 16 units) and trains for 30000 steps. After training, we run each of the final policies for 10000 episodes to evaluate the cost-returns of our algorithms.

Cost-Return Distribution Evaluation

In Figure 4.5, we compare the two algorithms with risk-neutral ($\alpha = 1$) and risk-averse ($\alpha = 0.1$) constraints on both versions of the SpyGame. We report the distribution of cost-returns. This gives a clear overview of the full cost-return distribution, allowing us to evaluate the frequency the safety constraint is violated. We also report the metric used as safety constraints to verify when each agent can reach the designated safety requirements.

At the top of Figure 4.5 (risk-neutral case), we can see that the two WCSAC algorithms approximately attain a constraint-satisfying expected cost-return in both environments, and the realized values are very close. So, in the average case, WCSAC-GS and WCSAC-IQN have similar performances independently of the underlying distribution.

At the bottom of Figure 4.5 (risk-averse case), first we notice that on the Spy-Unimodal environment (Gaussian) both WCSAC algorithms attain a cost-CVaR-0.1 below the threshold. We can also notice that WCSAC-IQN is closer to the bound showing a slightly better

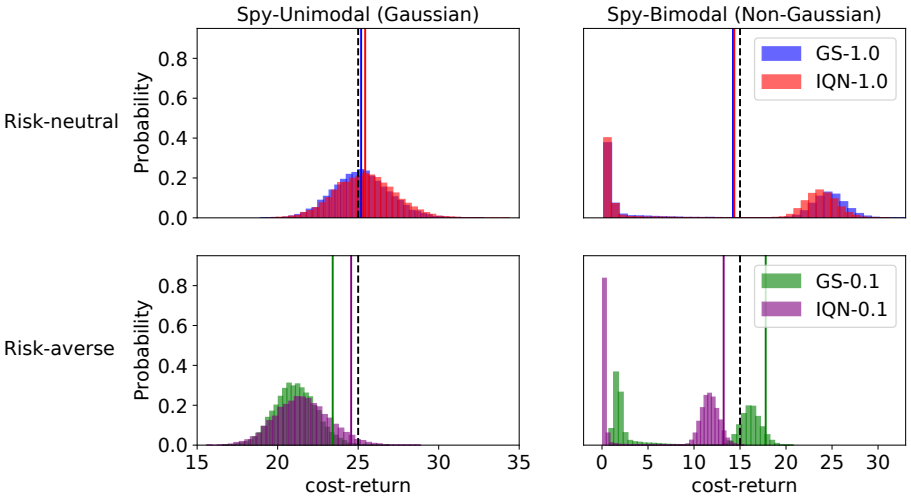


Figure 4.5: Cost-return distribution in Spy-Unimodal (left) and Spy-Bimodal (right) considering risk-neutral constraints with CVaR-1.0 (top) and risk-averse constraints with CVaR-0.1 (bottom). Dashed lines show the imposed safety threshold d and solid lines the realized values for each method.

control over the cost-CVaR-0.1. On Spy-Bimodal (non-Gaussian), WCSAC-GS is unable to satisfy the safety constraint, attaining a cost-CVaR-0.1 larger than the bound. This indicates that the Gaussian approximation cannot control the risk level in this domain.

Overall, comparing to the top and bottom plots in Figure 4.5, we can see that both WCSAC algorithms can attain a more risk-averse behavior by setting the risk level α to be a small value, reducing significantly the probability that a trajectory violates the safety constraints.

VARYING LEVEL OF SAFETY CONSTRAINT

To get a better overview of when the safety constraints are violated or not, we consider the same environment setting different risk level constraints. In Figure 4.6, the x-axis depicts the risk level α, under which the agents are trained. The y-axis depicts the corresponding cost-CVaR-α (Figure 4.6 top) and expected return (Figure 4.6 bottom) generated by the final policies and the standard deviation over 5 repetitions.

At the bottom of Figure 4.6, we can see that, in the more risk-averse settings (lower value of α), WCSAC-GS and WCSAC-IQN will both have lower expected return. In general, the changes of the cost-CVaR-α and expected return under different risk levels show the same trend, i.e., a larger cost-CVaR-α corresponds to a larger expected return at the risk level α.

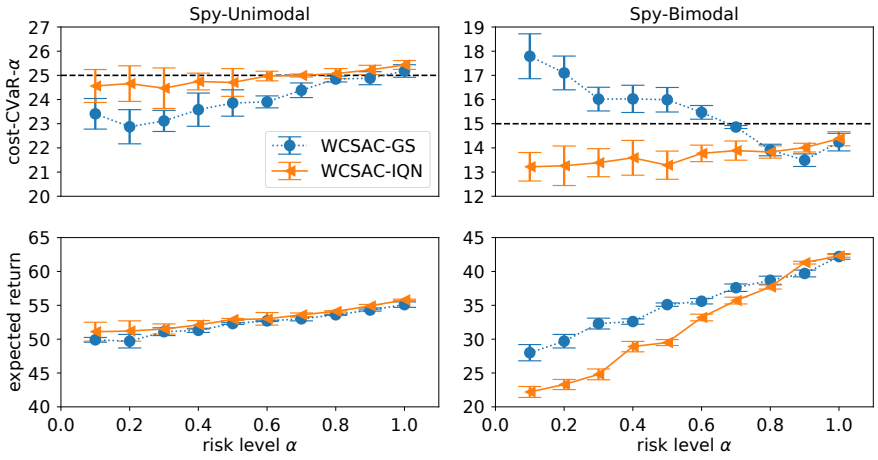


Figure 4.6: Performance of WCSAC-GS and WCSAC-IQN in terms of different risk levels. The error bars are the standard deviation over 5 runs. Dashed lines show the imposed safety threshold d.

When we have a unimodal cost-return distribution (left panel of Figure 4.6), we can see that the WCSAC algorithms attain safe performance with different risk levels α. But when we have higher safety requirements, both WCSAC algorithms generate a greater variance, and the distance between the corresponding cost-CVaR-α and the safety bound d becomes larger. Compared to WCSAC-IQN, WCSAC-GS is more over-conservative with lower α.

At the right panel of Figure 4.6, we show the results with a bimodal cost-return distribution, where a Gaussian approximation can underestimate the CVaR, as we have seen in the previous section. In this case, WCSAC-IQN is safe in all different α, and WCSAC-GS can also obtain safe performance for values closer to the risk-neutral constraint ($\alpha \in [0.7, 1.0]$). However, WCSAC-GS starts to become increasingly unsafe for lower values of α (more risk-averse constraints).

No matter in the unimodal or bimodal cases, both WCSAC algorithms approach the safety boundary better with higher α (more risk-neutral). But we have greater deviation with lower α, especially in the bimodal case. Even with WCSAC-IQN, our safety performance is becoming more pessimistic when we decrease α. Based on the experimental results in the work by (Théate et al., 2021), it appears to be the case that the quantile regression methods may result in more approximation errors for higher-order moments compared to the first-order moment.

Table 4.1: Performance of policies after training in terms of expected return (ER), expected cost-return (EC), and cost-CVaR-0.5 (C0.5). The metric of the safety constraint used by each agent is in bold.

Env	StaticEnv ($d = 8$)			PointGoal ($d = 25$)			CarButton ($d = 25$)		
	ER	EC	C0.5	ER	EC	C0.5	ER	EC	C0.5
SAC	0.87	18.53	19.11	28.85	62.77	71.59	21.36	201.06	247.13
CPO	-0.63	**9.25**	14.71	21.48	**42.99**	50.39	3.62	**80.16**	116.31
PPO-Lag	-0.76	**6.31**	8.93	17.81	**24.47**	30.11	0.45	**26.18**	35.09
GS-1.0	0.75	**8.04**	13.42	23.08	**40.39**	61.19	11.45	**65.03**	80.86
IQN-1.0	0.46	**7.83**	9.61	15.46	**23.41**	28.74	1.40	**24.02**	32.34
GS-0.5	0.66	6.16	**7.30**	19.50	37.23	**38.29**	7.06	58.51	**62.97**
IQN-0.5	0.40	5.10	**7.18**	9.23	20.80	**22.60**	0.64	15.95	**20.13**

4.3.2. SAFETYGYM ENVIRONMENTS

Next, we evaluate our method in three domains from the *Safety Gym* benchmark suite (Ray, Achiam, and Amodei, 2019), where a robot navigates in a 2D map to reach target positions while trying to avoid dangerous areas, with different complexity levels (Figure 4.7). The first one is StaticEnv with one fixed hazard and one fixed goal, but the initial position of the Point agent is randomly generated at the beginning of each episode. The second is PointGoal (`Safexp-PointGoal1-v0` in *Safety Gym*) with one Point agent, several hazards, and one vase. The third and more complex environment is CarButton (`Safexp-CarButton1-v0` in *Safety Gym*) where a Car robot (higher dimensional action space than Point) is navigating to press a goal button while trying to avoid hazards and moving gremlins, and not pressing any of the wrong buttons. These tasks are particularly complex due to the observation space, instead of observing its location, the agent has a lidar that indicates the distance to other objects. All experiments are performed with 10 random seeds. In all environments, $c = 1$ if an unsafe interaction happens, otherwise $c = 0$. We use the original reward signal in *Safety Gym*, i.e., the absolute distance towards the goal plus a constant for finishing the task, e.g., pressing the goal button.

We evaluate four versions of the WCSAC: GS-1.0 (WCSAC-GS with $\alpha = 1.0$), GS-0.5 (WCSAC-GS with $\alpha = 0.5$), IQN-1.0 (WCSAC-IQN with $\alpha = 1.0$), and IQN-0.5 (WCSAC-IQN with $\alpha = 0.5$). For comparison, we used SAC (Haarnoja, Zhou, Hartikainen, et al., 2018), CPO (Achiam et al., 2017), and PPO-Lagrangian (PPO-Lag; Schulman, Wolski, et al., 2017; Ray, Achiam, and Amodei, 2019) as baselines. In this experiment, we use the discount factor $\gamma = 0.99$ and $\kappa = 1$ for the Huber loss in WCSAC-IQN. The safety thresholds are set to be $d = 8$ for StaticEnv, and $d = 25$ for PointGoal and CarButton. We train each agent for 50 epochs in StaticEnv, and for 150 epochs in PointGoal and CarButton. The epoch length is 30000 environment steps, and the maximal episodic length is 1000 environment steps.

To evaluate the performance of the algorithms, we use the following metrics: CVaR-

(a) StaticEnv (b) PointGoal (c) CarButton

Figure 4.7: Robot navigation environments. The three environments differ in complexity, i.e., the action space, the type of obstacles, and the task. The Point or Car agent should avoid Hazards (dangerous areas), Vases (fragile objects), and Gremlins (moving objects) in the above environments, then move to the Goal position or press the Goal Button

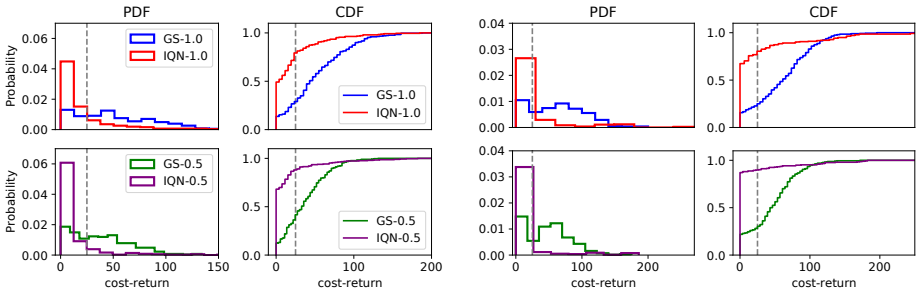
.

0.5 of the cost-return (cost-CVaR-0.5), expected cost (AverageEpCost), and expected reward (AverageEpRet). Table 4.1 shows the performance of the policies returned by the algorithms after training. We use 1000 episodes (100 runs for each random seed) to evaluate the final policy of each method; the expected cost and expected return are estimated by the average of all runs, while the cost-CVaR-0.5 is estimated by the average of the worst 500 runs. In Figure 4.8, we visualize the distribution by plotting the PDF and CDF histograms of sampled episodic costs in PointGoal and CarButton. Finally, Figure 4.9 shows the behavior of the algorithms during training. We provide a collection of videos of the execution of the final policies on the following webpage: `https://sites.google.com/view/wcsac`.

FINAL BEHAVIOR

We start our analysis by considering the behavior of the final policy. In Table 4.1, we can see that only IQN-1.0 and IQN-0.5 can be considered safe, because they satisfy the cost constraint with which they trained in all environments. In particular, only IQN-0.5 satisfies the risk-averse threshold on cost-CVaR-0.5, demonstrating its suitability for risk-averse agents. PPO-Lag has competitive safety performance in all environments, but fails to achieve a high return in StaticEnv and CarButton. Compared to the safe RL methods (CPO, PPO-Lag, and WCSAC), SAC has an excellent performance in expected return, but, naturally it does not satisfy the safety constraint, this shows that a safe agent must find a tradeoff between safety and performance. Although the final policies of the remaining algorithms CPO, GS-1.0 (SAC-Lag), and GS-0.5 may show better expected return, these methods are not safe in PointGoal and CarButton.

In Figure 4.8, we can observe that the distributions in PointGoal and CarButton are not



(a) PointGoal (b) CarButton

Figure 4.8: Cost-return distributions in PointGoal and CarButton (based on the same data as Table 4.1). The x-axis depicts the cost-return, while the y-axis depicts the binned probability density and the cumulative density functions. The gray dashed line indicates the safety threshold.

Gaussian, which justifies the use of a quantile regression algorithm and the safe behavior of the WCSAC-IQN algorithms in these more complex environments. Compared to GS-1.0, GS-0.5, and IQN-1.0, the distribution of IQN-0.5 displays a smaller range of costs, most of which are within the safety bound. Although the policies from GS-0.5 still generate some unsafe trajectories, the likelihood is much lower.

BEHAVIOR DURING TRAINING

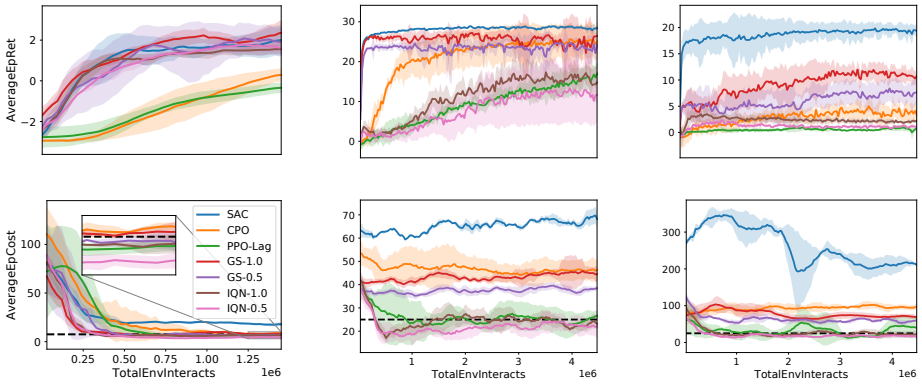
Figure 4.9 shows the behavior of the agents during training. The top row shows the expected return, while the bottom row shows the expected cost-return.

We can see that all safe RL methods manage to make some safety improvements, while SAC obviously has better and more stable performance in average episodic return across all the environments since it ignores the safety constraints.

In StaticEnv (Figure 4.9(a)), we notice that all safe RL algorithms converge toward the optimal policy. However, compared to the off-policy WCSAC, the on-policy baselines CPO and PPO-Lag take more time to do so. When we look closely, we notice that CPO and GS-1.0 exceed slightly the cost bound at the end of the training, while PPO-Lag, GS-0.5, and IQN-1.0 end slightly below the cost bound. In particular, we highlight that IQN-0.5 achieves a lower expected cost without sacrificing much performance in terms of return.

In PointGoal (Figure 4.9(b)), we see a different behavior: only the PPO-Lag and WCSAC-IQN algorithms manage to find a satisfactory policy. Although WCSAC-GS and CPO manage to find policies with high returns, they fail to achieve safe behavior.

Finally, in the most complex environment, CarButton (Figure 4.9(c)), we see that the cost constraints severely limit the ability to find high-reward policies: PPO-Lag, IQN-1.0, and IQN-0.5 manage to find safe policies, however, they cannot improve significantly

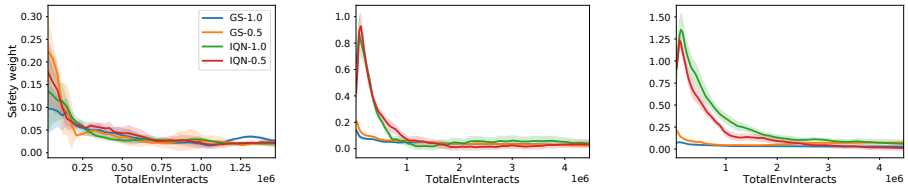


(a) StaticEnv (b) PointGoal (c) CarButton

Figure 4.9: Performance of the algorithms during training in terms of mean (solid lines) and ±1 standard deviation (shaded area) of the runs within an epoch. The black dashed lines indicate the safety thresholds.

in terms of return; GS-1.0 and GS-0.5 also approach a safe policy and manage to get some improvements in terms of return; and CPO does not find a safe policy whilst simultaneously struggling to improve returns.

As to the unsafe performance of CPO in PointGoal and CarButton, Ray, Achiam, and Amodei (2019) conclude that the function (value and policy network) approximation error and sampling error (caused by using a finite number of samples) in CPO greatly degrade its performance on these Safety Gym environments. PPO-Lag has competitive performance in safety, but it converges slowly compared to the off-policy baselines. This phenomenon is even more obvious in relatively simple StaticEnv. For WCSAC-GS in the relatively more complex environments PointGoal and CarButton (Figures 4.9(b) and 4.9(c)), we can see that the return and cost-return of WCSAC-GS start to stabilize near a certain value, instead of making continuous improvement until satisfying the constraint. However, in Figure 4.10, the safety weights of GS-1.0 and GS-0.5 quickly converge to a small value. It appears to be the case that the algorithm mistakenly takes the policy as safe, which means we get a convergent safety approximation (CVaR or expectation) that is below the safety threshold, but the safety of the policy is not truly reflected. Then, the algorithm stops making any progress in safety. Compared to the Gaussian approximation, the safety weights of IQN-1.0 and IQN-0.5 have drastic changes at the beginning of training (Figures 4.10(b) and 4.10(c)), but they finally converge to a safe policy according to the training process in Figure 4.9. We hypothesize that WCSAC-IQN benefits from the quantile regression to enhance exploration and avoid overfitting. That may also explain why distributional RL can converge to a better policy than traditional RL (Dabney, Ostrovski, et al., 2018).



(a) StaticEnv (b) PointGoal (c) CarButton

Figure 4.10: Change of the adaptive safety weights ω in WCSAC during training. A small stable weight means that the constraint is approximately satisfied. In StaticEnv, the safety weights in the four WCSAC algorithms change similarly, which accord with the convergence process of cost-return in Figure 4.9(a). In PointGoal and CarButton, compared to IQN-1.0 and IQN-0.5, the safety weights of GS-1.0 and GS-0.5 quickly converge to a small value, but they fail to obtain a constraint-satisfying policy from the results in Figures 4.9(b) and 4.9(c). Although the safety weights of IQN-1.0 and IQN-0.5 have drastic changes at the beginning of training (Figures 4.10(b) and 4.10(c)), they finally converge to a safe policy, according to the training process in Figure 4.9.

Given the additional intricacy arising from the estimation of the distribution, a more sophisticated neural network architecture would be necessary to accurately approximate the underlying distribution in WCSAC-IQN. Despite the improved performance and enhanced sample efficiency exhibited by WCSAC-IQN, it should be noted that, with the hyperparameters in this dissertation, each iteration of the gradient step incurs approximately double the required computational time for the gradient step when compared to WCSAC-GS in all the environments.

TRAJECTORY ANALYSIS

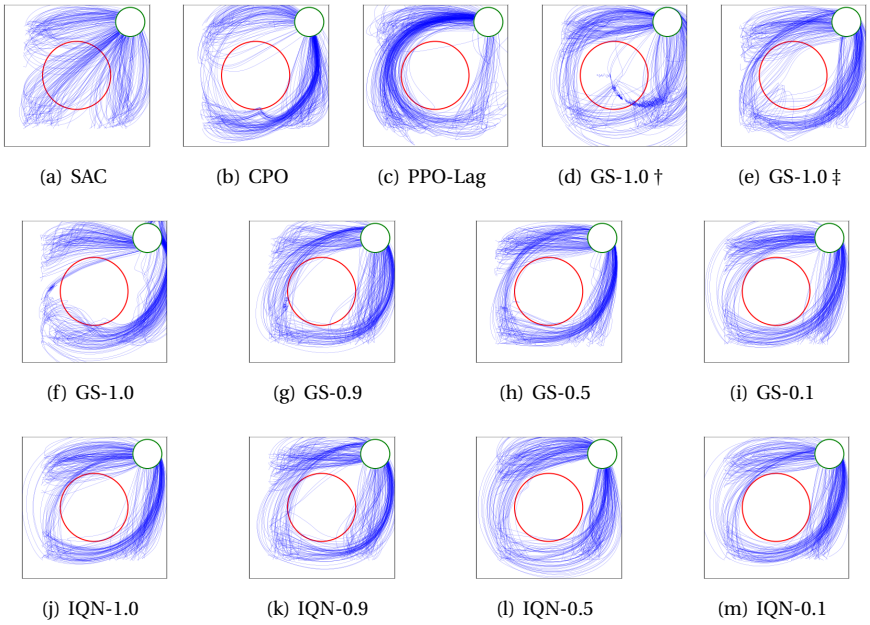
Finally, we execute a trajectory analysis in StaticEnv, see Figure 4.11. Specifically, we will compare SAC, CPO, PPO-Lag, WCSAC with different risk levels in safety, i.e., $\alpha = 0.1$ (highly risk-averse), $\alpha = 0.5$, $\alpha = 0.9$, and $\alpha = 1.0$ (risk-neutral).

The behavior of the SAC agent is presented in Figure 4.11(a), the agent chooses the shortest path to reach the target directly since the safety constraint is not considered.

We also consider the training process of GS-1.0 at different stages[1]. At the beginning of learning (Figure 4.11(d)), it is possible that the agent cannot get out of the hazard, and gets stuck before arriving at the goal area. We can observe the number of constraint violations being reduced over time (Figures 4.11(e) and 4.11(f)).

The final policies from CPO, PPO-Lag, GS-1.0, GS-0.9, IQN-1.0, and IQN-0.9 perform better than before, but still prefer to take a risk within the budget to get a larger return (Figures 4.11(b), 4.11(f), 4.11(g), 4.11(j) and 4.11(k)). Conversely, the agents GS-0.5 and IQN-0.5 are more risk-averse (Figures 4.11(h) and 4.11(l)). Finally, in Figures 4.11(i) and 4.11(m), we can see that the agents GS-0.1 and IQN-0.1 prefer to avoid the hazardous area more strictly given its risk level setting.

[1] Other methods show similar behavior during training.



(a) SAC (b) CPO (c) PPO-Lag (d) GS-1.0 † (e) GS-1.0 ‡

(f) GS-1.0 (g) GS-0.9 (h) GS-0.5 (i) GS-0.1

(j) IQN-1.0 (k) IQN-0.9 (l) IQN-0.5 (m) IQN-0.1

Figure 4.11: Trajectory analysis. (c)-(e) show the trajectories generated by policies from GS-1.0 (SAC-Lag) at the beginning, middle stage and end of training respectively. (a), (b), and (f)-(l) show the final trajectories from SAC, CPO, PPO-Lag, and WCSAC separately.

Overall, we observe that with a higher risk level α (around 1.0), WCSAC can attain risk-neutral performance similar to expectation-based methods. Both WCSAC algorithms can become more risk-averse by setting lower risk level α.

4.4. RELATED WORK

Risk-averse methods have commonly been used in RL problems with a single signal (reward or cost). Although CVaR in the context of IQN was used in the work by Dabney, Ostrovski, et al. (2018) (with only a reward signal) to get risk-sensitive policies, the implementation is significantly different from ours. We deploy IQN in the safety-constrained RL problems (with separate reward and cost signals) with continuous action spaces, instead of problems with a discrete action space.

Based on the work by Bellemare, Dabney, and Munos (2017), Keramati et al. (2020) propose to perform strategic exploration to quickly obtain the optimal risk-averse policy. Following Dabney, Ostrovski, et al. (2018), Urpí, Curi, and Krause (2021) propose a new actor critic method to optimize a risk-averse criterion in terms of return, where only samples previously collected by safe policies are available for training. Although their paper

provides the off-policy algorithm version, it is not clear how the exploration-exploitation tradeoff is handled, while we explicitly define a SAC-based method with an entropy-related mechanism for exploration. Chow et al. (2017) propose efficient RL algorithms for risk-constrained MDPs, but their goal is to minimize an expected cumulative cost while keeping the cost CVaR below a given threshold, instead of maintaining reward and cost signals independently. To some extent, the way they update the Lagrange multiplier inspired our use of adaptive safety weights. However, in the real world, safe RL problems typically involve multiple objectives, some of which may be contradictory, like collision avoidance and speed on an autonomous driving task (Kamran, Lopez, et al., 2020). Therefore, the setting with an explicit safety signal can be more practical (Dulac-Arnold et al., 2021).

The safe RL setting with separate reward and cost signals has also been studied in several works (Achiam et al., 2017; Liu, Ding, and Liu, 2020; Yang, Rosca, et al., 2020; Bharadhwaj et al., 2021). Specifically, Achiam et al. (2017); Liu, Ding, and Liu (2020); Yang, Rosca, et al. (2020) propose a series of on-policy constrained policy optimization methods with trust-region property, where the worst case performance is bounded at each update. However, they do not present a clear risk aversion mechanism for the intrinsic uncertainty, captured by the distribution over the cost-return. In addition, on-policy methods (with worse sample efficiency compared to off-policy methods) are usually not favored in safe RL domains. Under a similar problem setting, Bharadhwaj et al. (2021) work on a conservative estimate of the expected cost-return (Kumar, Zhou, et al., 2020) for each candidate state-action tuple, which is used in both safe-exploration and policy updates. With the conservative safety estimate, their proposed method can learn effective policies while reducing the rate of catastrophic failures. However, they only focus on the parametric uncertainty over the value estimate instead of the intrinsic uncertainty. On the other hand, their paper focuses on catastrophic events, which is a binary signal. While our work considers safety according to the accumulated cost in a trajectory. Overall, our approach gives more freedom to the designer of the system to indicate which behaviors are more or less desirable.

Finally, we did not find state-of-the-art distributional RL techniques had been used in safety-constrained RL with separate reward and safety signals prior to our work.

4.5. CONCLUSION

In this chapter, we propose to improve the WCSAC framework by employing an implicit quantile network as safety critic to overcome considerable errors caused by the Gaussian assumption. The experiments show that both WCSAC-GS and WCSAC-IQN can attain better risk control compared to expectation-based methods. In complex environments, WCSAC-GS does not show improvements in safety, where the safety weight is not up-

dated fast enough to truly reflect the current policy. However, WCSAC-IQN has strong performance with the benefits from IQN, which provides a stronger safety signal than the one from the Gaussian approximation. The novel use of IQN for safety constraints can potentially be extended to other safe RL methods. Without any knowledge about the environment, it is hard to strictly fulfill the safety constraint during exploration. Thus, WCSAC still focuses more on the performance of the final policy. While our method has good risk control for the safety-constrained RL problems, one limitation is that we cannot ensure a safe training process. Also, although our method shows good performance in practice, our work has not established theoretical proof of convergence.

5

GUIDED SAFE EXPLORATION

In this chapter, we propose to leverage a safe guide (SaGui) to ensure safety during training. While WCSAC (Chapters 3 and 4) achieves risk control for the safety-constrained RL problems, safety during training is not guaranteed. In the real world, it is impossible to have training safety assurance if learning from scratch. Often, RL agents are trained in a controlled environment, such as a laboratory, before being deployed. However, the real-world target task might be unknown prior to deployment. Reward-free RL addresses this problem by training an agent without the reward to adapt quickly once the reward is revealed. We consider the constrained *reward-free setting, where an agent (the guide) learns to explore safely without the reward signal. This agent is trained in a controlled environment, which allows unsafe interactions and still provides the safety signal. After the target task is revealed, safety violations are not allowed anymore. Thus, the guide is leveraged to compose a safe behavior policy. Drawing from transfer learning, we also regularize a target policy (the student) towards the guide while the student is unreliable and gradually eliminate the influence from the guide as training progresses. The empirical analysis shows that this method can achieve safe transfer learning and helps the student solve the target task faster.*

5.1. INTRODUCTION

Developments in safe RL have allowed us to learn safe policies by solving constrained Markov decision processes (CMDPs; Altman, 1999). For instance, SAC-Lag (Ha et al., 2020)

This chapter has been published in ALA (2022) (Yang, Simão, Jansen, et al., 2022).

combines the SAC (Haarnoja, Zhou, Abbeel, et al., 2018; Haarnoja, Zhou, Hartikainen, et al., 2018) algorithm with Lagrangian methods to learn a safe policy in an off-policy way. This algorithm solves high-dimensional problems with a sample complexity lower than on-policy algorithms. Unfortunately, it only finds a safe policy at the end of the training process and may be unsafe while learning.

Some knowledge about the safety dynamics can ensure safety during learning. One can precompute unsafe behavior and mask unsafe actions using a so-called shield (Alshiekh et al., 2018; Jansen et al., 2020; Carr et al., 2023), or start from an initially safe baseline policy and gradually improve its performance while remaining safe (Achiam et al., 2017; Tessler, Mankowitz, and Mannor, 2019; Yang, Rosca, et al., 2020). However, these approaches may necessitate numerous interactions with the environment before they find an adequate policy (Zanger, Daaboul, and Zöllner, 2021). Moreover, reusing a pre-trained policy can have a detrimental effect, since the agent encounters a new trajectory distribution as the policy changes (Igl et al., 2021). Therefore, we investigate *how to efficiently solve a task without violating the safety constraints.*

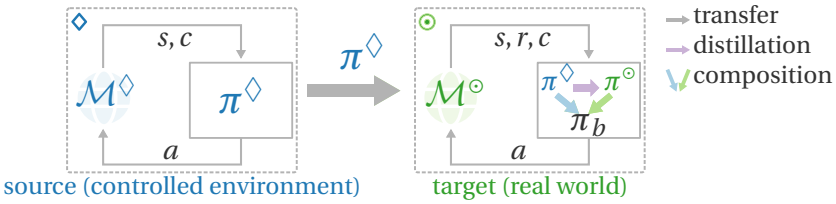


Figure 5.1: Transferring the Safe Guide (SaGui) policy $\pi^{\diamondsuit}$ from the source task ($\diamondsuit$) to the target task ($\odot$) with three steps: *i*) training the SaGui policy; *ii*) *distilling* the guide's policy into a student policy, and *iii*) *composing* a safe behavior policy.

We make two key observations. First, RL agents often learn in a controlled environment, such as a laboratory or a simulator before being deployed in the real world (García and Fernández, 2015). Second, an agent can often benefit from expert guidance instead of solely relying on trial-and-error (Peng et al., 2022). For instance, in autonomous driving, the driver agent can quickly learn by mimicking an expert's behavior to handle dangerous situations. Such a process is referred to as *policy distillation*. Furthermore, under expert guidance, the agent can safely explore before taking dangerous actions (composite sampling).

Transfer learning (Taylor and Stone, 2009) investigates how to improve the learning of a target task with some knowledge from a source task. In these settings, the source task may provide only partial knowledge of the target task. We adopt a transfer learning framework Taylor and Stone (2009) and refer to (i) the controlled environment as the *source task* ($\diamondsuit$) and (ii) the real world as the *target task* ($\odot$). In our setting, the controlled environment provides only the cost signals related to safety but not the reward signals of

the target task in the real world. The central problem is then to avoid any safety violations after the target task is revealed.

Our approach. We show how to transfer knowledge encoded by a policy to enhance safety. Here, we refer to the policy that has been learned in the source task as the *safe guide* (SaGui, Figure 5.1). The intuition is that, in the real world, the agent is guided to accomplish the target task in a safe manner. We propose to transfer SaGui from the source task to the target task. Our approach has three central steps:

- train the SaGui policy and *transfer* it to the target task;
- *distill* the guide's policy into a *student policy* which is dedicated to the target task, and
- *compose* a behavior policy that balances safe exploration (using the guide) and exploitation (using the student).

As we train the guide in a reward-free constrained RL setting (Miryoosefi and Jin, 2021), the agent only observes the costs related to safety, and it does not access the reward signals. This task-agnostic approach allows us to train a guide independently of the reward of the target task, so this guide can be useful for different reward functions. Furthermore, we assume the source task preserves the safety dynamics, which allows us to train a guide that can act safely when transferred to the target task. Inspired by advances in robotics where an agent is trained under strict supervision, we assume the source task is a simulated/controlled environment (Schuitema et al., 2010; Xie et al., 2019). Therefore, safety is not required while training the SaGui policy. Once the target task is revealed, SaGui safely collects the initial trajectories in the target environment and the student starts learning based on these trajectories. To ensure that the new policy quickly learns how to act safely, we also employ a policy distillation method, encouraging the student to imitate the guide.

Contributions. Our main contributions are: we *i*) formalize transfer learning for RL from a safety perspective, *ii*) propose to guide learning using a task-agnostic agent with exploration benefits, *iii*) show how to adaptively regularize the student policy to the guide policy based on the student's safety, *iv*) investigate when to sample from the student or from the guide to ensure safe behavior in the target environment and fast convergence of the student policy, and *v*) demonstrate empirically that, compared to learning from scratch and adapting a pre-trained policy, our method can learn to solve the target task faster without violating the safety constraints on the target task.

5.2. SAFE AND EFFICIENT EXPLORATION

Naturally, to train RL agents without violating the safety constraints, some prior knowledge is required (Sui et al., 2015). Often, a safe initial policy collects the initial trajectories (Achiam et al., 2017; Tessler, Mankowitz, and Mannor, 2019; Yang, Rosca, et al., 2020). However, these approaches largely neglect how this policy is computed or what makes it effective. Therefore, we explicitly consider the problem of how to obtain an initial policy that can safely expedite learning in the target task. This section formalizes this problem and provides an overview of our approach.

5.2.1. PROBLEM STATEMENT

We formalize our problem setting using the transfer learning (TL) framework. In general, TL allows RL agents to use expertise from *source* tasks to speed up the learning process on a *target* task (Taylor and Stone, 2009; Zhu, Lin, and Zhou, 2020). The source tasks $\{\mathcal{M}^{\diamondsuit}\}$ should provide some knowledge $\mathcal{K}^{\diamondsuit}$ to an agent learning in the target task $\mathcal{M}^{\odot}$, such that, by leveraging $\mathcal{K}^{\diamondsuit}$, the agent learns the target task $\mathcal{M}^{\odot}$ faster.

As we are particularly interested in the safety properties of the transfer, we consider a reward-free source task, which only provides information about the safety dynamics. Moreover, we use a policy to encode the knowledge transferred. Formally, given a source task $\mathcal{M}^{\diamondsuit} = \langle \mathcal{S}^{\diamondsuit}, \mathcal{A}^{\diamondsuit}, \mathcal{P}^{\diamondsuit}, \emptyset, c^{\diamondsuit}, d^{\diamondsuit}, \iota^{\diamondsuit}, \gamma \rangle$, we compute the policy $\pi^{\diamondsuit}$ in the absence of a reward signal. This provides knowledge $\mathcal{K}^{\diamondsuit} = \{\pi^{\diamondsuit}\}$ to help solving the target task $\mathcal{M}^{\odot} = \langle \mathcal{S}^{\odot}, \mathcal{A}^{\odot}, \mathcal{P}^{\odot}, r^{\odot}, c^{\odot}, d^{\odot}, \iota^{\odot}, \gamma \rangle$.

To apply the source policy $\pi^{\diamondsuit}$ in the target task $\mathcal{S}^{\odot}$, we have a mapping from the source state space to the target state space $\Xi : \mathcal{S}^{\odot} \to \mathcal{S}^{\diamondsuit}$. Then, we can define a target policy $\pi^{\diamondsuit \to \odot}$ as follows: $\pi^{\diamondsuit \to \odot}(s) = \pi^{\diamondsuit}(\Xi(s))$. Furthermore, we assume the source task $\mathcal{M}^{\diamondsuit}$ and target task $\mathcal{M}^{\odot}$ share the same action space. To build the source task based on a target task and a mapping Ξ from the target state space to the source state space, we assume Ξ is a state abstraction function (Li, Walsh, and Littman, 2006).

Let $\Xi : \mathcal{S}^{\odot} \to \mathcal{S}^{\diamondsuit}$ be the state abstraction function. We define Ξ^{-1} as the inverse of the abstraction function such that $\Xi^{-1}(s^{\odot}) = \{s^{\diamondsuit} \in \mathcal{S}^{\diamondsuit} | \Xi(s^{\diamondsuit}) = s^{\odot}\}$. We assume a weighting function $w : \mathcal{S} \mapsto [0, 1]$, where

$$\sum_{s^{\odot} \in \Xi^{-1}(s^{\diamondsuit})} w(s^{\odot}) = 1, \forall s^{\diamondsuit} \in \mathcal{S}^{\diamondsuit}. \tag{5.1}$$

Now we can define the transition and cost function of the target task:

$$\mathcal{P}^{\Diamond}(s^{\Diamond\prime} \mid s^{\Diamond}, a) = \sum_{s^{\odot}\in\Xi^{-1}(s^{\Diamond})} \sum_{s^{\odot\prime}\in\Xi^{-1}(s^{\Diamond\prime})} w(s^{\odot})\mathcal{P}^{\odot}(s^{\odot\prime} \mid s^{\odot}, a) \tag{5.2}$$

$$c^{\Diamond}(s^{\Diamond}, a) = \sum_{s^{\odot}\in\Xi^{-1}(s^{\Diamond})} w(s^{\odot})c^{\odot}(s^{\odot}, a) \tag{5.3}$$

$$\iota^{\Diamond}(s^{\Diamond}) = \sum_{s^{\odot}\in\Xi^{-1}(s^{\Diamond})} w(s^{\odot})\iota^{\odot}(s^{\odot}). \tag{5.4}$$

Assumption 5.1. $\mathcal{A}^{\Diamond} = \mathcal{A}^{\odot} = \mathcal{A}$.

To enable the knowledge transferable between tasks, having the same action spaces ensures that the policy learned in the source task is directly applicable to the target task.

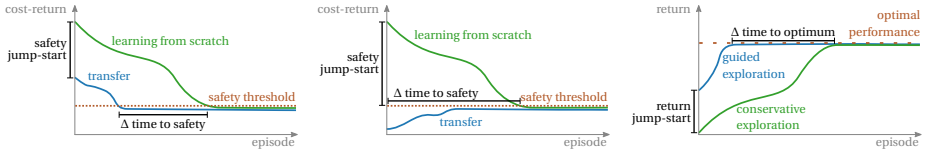
5.2.2. TRANSFER METRICS

To evaluate a safe transfer RL algorithm, Figure 5.2(a) presents a schematic of transfer metrics related to safety (inspired by (Taylor and Stone, 2009)): *safety jump-start* indicates how much closer to the safety threshold the expected cost-return of an agent learning with the source knowledge is compared to the expected cost-return of an agent learning from scratch in the first episodes, and Δ *time to safety* is the difference in the number of interactions required to become safe.

Notice that a trained agent might start with an expected cost-return lower than the safety threshold, for instance, when the safety threshold in the source task is lower than in the target task (Figure 5.2(b)). In this case, *safety jump-start* would be the difference between the safety threshold and the cost-return of an agent learning from scratch. Similarly, the Δ *time to safety* would be the number of interactions an agent learning from scratch needs to become safe.

In the case of two methods that can solve the target task without violating the safety constraints, we can also consider the usual metrics of transfer learning with respect to the reward (Taylor and Stone, 2009). For instance, Figure 5.2(c) shows the initial improvement in terms of performance which we call *return jump-start*, and the time necessary to reach an optimum performance, which we call the Δ *time to optimum*.

Problem statement. We aim to maximize the *safety jump-start* (potentially preventing safety violations in the target task) and to reduce the *time to optimum* (improving exploration) when transferring a policy $\pi^{\Diamond}$ from a source task $\mathcal{M}^{\Diamond}$ to a target task $\mathcal{M}^{\odot}$.



(a) Unsafe transfer. (b) Fully safe transfer. (c) Return transfer.

Figure 5.2: Pictorial illustration showing different transfer metrics for safe reinforcement learning. Usually, we consider *safety jump-start* and Δ *time to safety*. If we can develop agents that learn without violating the safety constraint, we can also consider *return jump-start* and Δ *time to optimum*.

5.2.3. METHOD OVERVIEW

Recall that for our transfer setting, we consider a single source task that only provides the safety signals, which we use to train the guide. Without the reward signal, the guide aims to explore the world safely and efficiently. We are interested in using the guide's safe exploration capabilities to train the student on the target task without violating the safety constraints. Notably, *i*) the guide and the student are trained separately; *ii*) the guide is only trained once and can support the training of different students; and *iii*) the guide only has access to safety information and no knowledge about the student's task.

To ensure the source policy is safe when deployed in the target task, we assume that the source task has a safety threshold lower than or equal to the target task, and Ξ is a state abstraction that preserves the safety dynamics, as formalized next.

Assumption 5.2. *The safety threshold of the target task upper bounds the safety threshold of the source task:* $d^{\diamondsuit} \leq d^{\odot}$.

Assumption 5.3. Ξ *is a* Q^c_π*-irrelevance abstraction (Li, Walsh, and Littman, 2006), therefore*

$$\Xi(s) = \Xi(s') \Rightarrow Q^c_{\pi^\odot}(s,a) = Q^c_{\pi^\odot}(s',a), \forall s, s' \in \mathcal{S}^\odot, a \in \mathcal{A}, \pi^\odot.$$

This assumption allows us to connect the expected cost-return of a policy on the source task to the expected cost-return on the target task.

Given Assumption 5.1 and Assumption 5.3, we have $Q^{c,\diamondsuit}_{\pi^\diamondsuit}(\Xi(s),a) = Q^{c,\odot}_{\pi^{\diamondsuit\to\odot}}(s,a) \quad \forall s \in \mathcal{S}^\odot, a \in \mathcal{A}, \pi^\diamondsuit$. That is, the expected cost of a source policy is the same in the source task and in the target task. With additional Assumption 5.2, any policy that is safe on the source task $\mathcal{M}^\diamondsuit$ is also safe when deployed on the target task $\mathcal{M}^\odot$.

It is important to note, however, that the reward function $r^\odot$ in the target task may not be related to the state space of the source task $\mathcal{S}^\diamondsuit$. Therefore, although a policy that is safe on the source task is also safe on the target task, the behavior required to accomplish the target task may not be defined on the source task. Consider, for instance, an agent with access to its position and the position of a threat. In each target task, the agent might

need to visit a different goal position, which is not defined on the source task. Then, a safe policy may be conditioned only on the positions of the agent and the threat, but to achieve the target the agent must consider the goal position. This highlights the need to compute a policy dedicated to the target task.

5.3. GUIDED SAFE EXPLORATION

In this section, we consider how to train the *safe guide* (SaGui) policy. Then, we describe how the student learns to imitate the SaGui policy after the task is revealed, while learning to complete the target task. Finally, we investigate how to prevent safety violations while the student has not yet learned how to act safely.

5.3.1. TRAINING THE SAFE GUIDE

Since the source task does not provide information regarding the reward of the target task, we adopt a reward-free exploration approach to train the guide. To efficiently explore the world, we first consider maximizing the policy entropy under safety constraints. Then, we can solve the problem defined in (2.4) with $r(s,a) = 0 : \forall s \in \mathcal{S}, a \in \mathcal{A}$ to get a guide MAXENT. However, although MAXENT tends to have diverse behaviors, that does not imply efficient exploration of the environment. Especially for continuous state and action spaces, it is possible that a policy provides limited exploration even if it has high entropy.

To enhance the exploration of the guide, we adopt an auxiliary reward that motivates the agent to visit novel states. To measure the novelty, we first define the metric space $(\mathcal{S}^\ddagger, \varsigma)$, where $\mathcal{S}^\ddagger$ is an abstracted state space, and $\varsigma : \mathcal{S}^\ddagger \times \mathcal{S}^\ddagger \to [0, \infty)$ is a distance function, that is

$$\varsigma(s,s') = 0 \Leftrightarrow s = s', \varsigma(s,s') = \varsigma(s',s), \text{and } \varsigma(s',s'') \leq \varsigma(s,s') + \varsigma(s,s''), \forall s, s', s'' \in \mathcal{S}^\ddagger.$$

Notice that $\mathcal{S}^\ddagger$ may not be the original state space $\mathcal{S}$. Especially when $\mathcal{S}$ is high dimensional, $\mathcal{S}^\ddagger$ can be some selected dimensions from $\mathcal{S}$, or a latent space from representation learning. Next, we define the auxiliary rewards as the expected distance between the current state and the successor state:

$$r_t^\varsigma = \mathop{\mathbb{E}}_{s_{t+1} \sim \mathcal{P}(\cdot|s_t,a_t)} \left[\varsigma(f^\ddagger(s_t), f^\ddagger(s_{t+1})) \right], \qquad \forall s_t, a_t \in \mathcal{S} \times \mathcal{A}, \tag{5.5}$$

where we may apply a potential abstraction $f^\ddagger : \mathcal{S} \to \mathcal{S}^\ddagger$. So, we train the *guide* agent by solving the constrained optimization problem (2.4) based on the auxiliary reward r^ς. Then, the SAC-Lag algorithm can be directly employed to solve the problem (2.4), as Algorithm 3. Our auxiliary reward does not explicitly promote exploration, but we find

Algorithm 3 Maximum exploration RL for *safe guide*

Require: Task $\mathcal{M}^{\Diamond}$
Require: Hyperparameters β, d
initialize $\mathcal{D} \leftarrow \emptyset$
initialize $\theta_x^{\Diamond}$ for $x \in \{\pi, R, C, \omega\}$
for each iteration **do**
  for each environment step **do**
    $a_t \sim \pi^{\Diamond}(\cdot \mid s_t)$
    $s_{t+1} \sim \mathcal{P}(\cdot \mid s_t, a_t)$
    $r_t^{\varsigma} \leftarrow \varsigma(f^{\ddagger}(s_t), f^{\ddagger}(s_{t+1}))$ ▷ Auxiliary task (5.5)
    $c_t^{\Diamond} \leftarrow c^{\Diamond}(s_t, a_t)$
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t, a_t, r_t^{\varsigma}, c_t^{\Diamond}, s_{t+1})\}$ ▷ Replay buffer
  end for
  for each gradient step **do**
    Sample experience from replay buffer $\mathcal{D}$
    for $x \in \{\pi, R, C, \omega\}$ **do**
      $\theta_x^{\Diamond} \leftarrow \theta_x^{\Diamond} - \lambda_x \hat{\nabla}_{\theta_x^{\Diamond}} J_x(\theta_x^{\Diamond})$ ▷ Parameter updating
    end for
  end for
end for
Output: Optimized parameters $\theta_{\pi}^{\Diamond}$ for $\pi^{\Diamond}$

that increasing both the step size and policy entropy significantly improves exploration in practice. Overall, our experiment with the auxiliary reward aimed to evaluate the impact of the exploration of the guide on how safely and quickly the student learns.

We could also consider more sophisticated reward-free exploration strategies such as maximizing the the entropy of the state occupancy distribution (Hazan et al., 2019; Seo et al., 2021; Svidchenko and Shpilman, 2021). However, we address it in Chapter 6 and focus on how to use the guide to improve how the student learns in this chapter.

5.3.2. POLICY DISTILLATION FROM THE SAFE GUIDE

When the agent is trained for a certain task, it is difficult to generalize when faced with a new task (Igl et al., 2021). Similarly, it is not trivial to adjust the guide's policy that was trained to explore the environment to perform the target task. Therefore, we train a new policy, referred as the student, dedicated to the target task.

We can leverage the *guide* agent to make the student quickly learn how to act safely. Through the mapping function Ξ, the transferred policy can be used by most constrained RL algorithms to regularize the student policy $\pi^{\odot}$ towards the guide policy $\pi^{\Diamond}$ using KL divergences, as shown in Figure 5.3. So, with $\pi^{\Diamond}$ fixed, we have an augmented reward

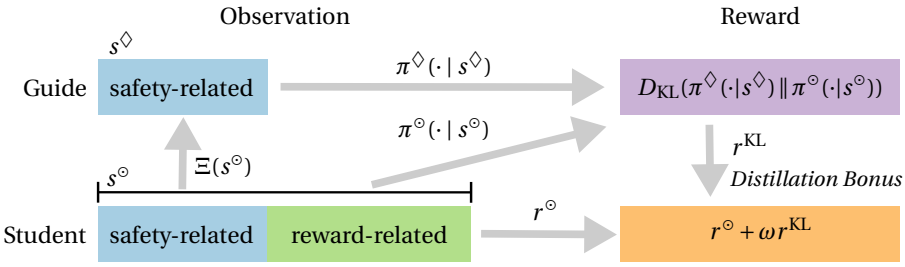


Figure 5.3: Overview of the policy distillation. We can leverage the *guide* agent to make the student quickly learn how to act safely. The target policy $\pi^{\odot}$ may have an input space different from the guide policy $\pi^{\Diamond}$. Through the mapping function Ξ, the transferred policy can be used to regularize the student policy $\pi^{\odot}$ towards the guide policy.

function

$$r'_t = r^{\odot}_t + \varpi r^{\mathrm{KL}}_t + \beta r^{\mathcal{H}}_t,$$

where

$$r^{\mathrm{KL}}_t = \log \frac{\pi^{\Diamond}(a_t \mid \Xi(s_t))}{\pi^{\odot}(a_t \mid s_t)}, \text{and } r^{\mathcal{H}}_t = -\log \pi^{\odot}(a_t \mid s_t).$$

The weights ϖ and β indicate the strengths of the KL and entropy regularization (respectively). Then, setting $r^{\Diamond}_t = \log \pi^{\Diamond}(a_t \mid \Xi(s_t))$, we can define the student's objective:

$$\max_{\pi^{\odot}} \mathop{\mathbb{E}}_{\tau \sim \rho_{\pi^{\odot}}} \sum_{t=0}^{\infty} \gamma^t \left[r^{\odot}_t + \varpi r^{\Diamond}_t + (\beta + \varpi) r^{\mathcal{H}}_t \right]. \tag{5.6}$$

To find an appropriate ϖ, our goal is to follow the guide more for safer exploration if the student's policy is unsafe, but eliminate the influence from the guide and focus more on the performance if the student's policy is safe. Therefore, we propose to set $\varpi = \omega$ to determine the strength of the KL regularization since the adaptive safety weight ω reflects the safety of the current policy.

In summary, we have an entropy regularized expected return with redefined (regularized) reward $r''_t = r^{\odot}_t + \omega r^{\Diamond}_t$. This augmented reward encourages the student to yield actions that are more likely to be generated by the guide. Then, SAC-Lag can be directly used to solve (5.6) with the additional entropy constraint (Algorithm 4, lines 18-23).

5.3.3. COMPOSITE SAMPLING

To enhance safety and improve the student during training (Algorithm 4, lines 4-17), we leverage a *composite sampling* strategy, which means our behavior policy (π_b) is a mixture of the guide's policy ($\pi^{\Diamond}$) and the student's policy ($\pi^{\odot}$). So, at each environment step,

Algorithm 4 Guided Safe Exploration

Require: Task $\mathcal{M}^{\odot}$
Require: The guide's policy $\pi^{\diamondsuit}$
Require: Hyperparameters $\overline{\mathcal{H}}$, d
initialize $\mathcal{D} \leftarrow \emptyset$
initialize $\theta_x^{\odot}$ for $x \in \{\pi, R, C, \beta, \omega\}$
for each iteration **do**
  for each environment step **do**
    if *linear-decay* **then**
      $b \leftarrow f_{\text{ld}}(\diamondsuit, \odot)$ ▷ linearly eliminate the effect of $\pi^{\diamondsuit}$
    else if *control-switch* **then**
      $b \leftarrow f_{\text{cs}}(\diamondsuit, \odot)$ ▷ $\pi^{\diamondsuit}$ takes control if unsafe
    end if
    $a_t \sim \pi_b(\cdot \mid s_t)$ ▷ Composite sampling (5.7)
    $\mathcal{I}_t \leftarrow \mathcal{I}(s_t, a_t)$ ▷ IS ratio (5.8)
    $r_t^{\odot} \leftarrow r^{\odot}(s_t, a_t)$
    $r_t^{\diamondsuit} \leftarrow \log \pi^{\diamondsuit}(a_t \mid \Xi(s_t))$
    $c_t^{\odot} \leftarrow c^{\odot}(s_t, a_t)$
    $s_{t+1} \sim \mathcal{P}^{\odot}(\cdot \mid s_t, a_t)$
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t, a_t, r_t^{\odot}, r_t^{\diamondsuit}, c_t^{\odot}, \mathcal{I}_t, s_{t+1})\}$
  end for
  for each gradient step **do**
    Sample experience from $\mathcal{D}$
    for $x \in \{\pi, R, C, \beta, \omega\}$ **do**
      $\theta_x^{\odot} \leftarrow \theta_x^{\odot} - \lambda_x \hat{\nabla}_{\theta_x^{\odot}} \mathcal{I} J_x(\theta_x^{\odot})$ ▷ Parameter updating
    end for
  end for
end for
output: Optimized parameters $\theta_{\pi}^{\odot}$ for $\pi^{\odot}$

$a_t \sim \pi_b(\cdot \mid s_t), s_t \in \mathcal{S}^{\odot}$ where

$$\pi_b(\cdot \mid s_t) = \begin{cases} \pi^{\diamondsuit}(\cdot \mid \Xi(s_t)), & \text{if } b = \diamondsuit, \\ \pi^{\odot}(\cdot \mid s_t), & \text{otherwise.} \end{cases} \tag{5.7}$$

We investigate two strategies to define b:

Linear-decay (Algorithm 5). This strategy, denoted as $b = f_{\text{ld}}(\diamondsuit, \odot)$, linearly decreases the probability of using $\pi^{\diamondsuit}$ with a constant decay rate after each iteration of the algorithm, conversely increasing the probability of using $\pi^{\odot}$. We have two modes with *linear-decay*: *step-wise*, where in each time step we may change π_b; and *trajectory-wise*, where π_b only changes at the start of a trajectory. The mode is decided before executing an episode,

Algorithm 5 Composite sampling (linear-decay)

Require: $\pi^{\diamondsuit}$, $\pi^{\odot}$, υ
1: **initialize** $P_\pi \leftarrow 1$, $P_{wise} \leftarrow 1$
2: **for** each iteration **do**
3: $P_b(\diamondsuit) = P_\pi$ ▷ The probability of using $\pi^{\diamondsuit}$
4: $P_b(\odot) = 1 - P_\pi$ ▷ The probability of using $\pi^{\odot}$
5: Sample $\kappa_{wise} \sim U(0,1)$
6: **if** $\kappa_{wise} < P_{wise}$ **then**
7: *step-wise* $\leftarrow$ *true*
8: **else**
9: *step-wise* $\leftarrow$ *false*
10: $b \leftarrow\sim P_b$ ▷ Choose behavior policy
11: **end if**
12: $P_{wise} = P_{wise} - \upsilon$ ▷ Decrease the probability of step-wise
13: **for** each environment step **do**
14: **if** *step-wise* **then**
15: $b \leftarrow\sim P_b$ ▷ Choose behavior policy
16: **end if**
17: **end for**
18: $P_\pi = P_\pi - \upsilon$ ▷ Decrease the probability of using $\pi^{\diamondsuit}$
19: **end for**
Output: π_b

and smoothly switches from the complete *step-wise* to the complete *trajectory-wise* over the training process. We linearly decrease the probability to execute the *step-wise* and use the *guide* with a constant decay rate after each iteration of the algorithm, conversely increasing the probability of executing the *trajectory-wise* and using the student policy. So, we initialize the probabilities $P_\pi = 1$ to determine π_b, and $P_{wise} = 1$ to determine the mode at the beginning (line 1). We linearly decrease them with a constant decay rate υ (lines 12 and 18), determined by the training length. At the beginning of each episode, we sample $\kappa_{wise} \sim U(0,1)$, so if $\kappa_{wise} < P_{wise}$, we will execute *step-wise*, or we are in *trajectory-wise* (lines 5-11). Under *step-wise*, at each time step, we sample from the *guide* $\pi^{\diamondsuit}$ with probability P_π, and sample from the student $\pi^{\odot}$ with probability $1 - P_\pi$ (lines 14-16). Under *trajectory-wise*, we only make a decision once at the beginning of the trajectory (line 10).

Control-switch (Algorithm 6). To balance between the safe exploration and the sample efficiency (the samples from the target policy is relatively more valuable), the student policy keeps sampling, i.e., $\pi_b = \pi^{\odot}$ at the start of a trajectory (line 3); after we meet the first $c_{t-1} > 0$, we have $\pi_b = \pi^{\diamondsuit}$ until the end of the trajectory (lines 13-16). Therefore, the

Algorithm 6 Composite sampling (control-switch)

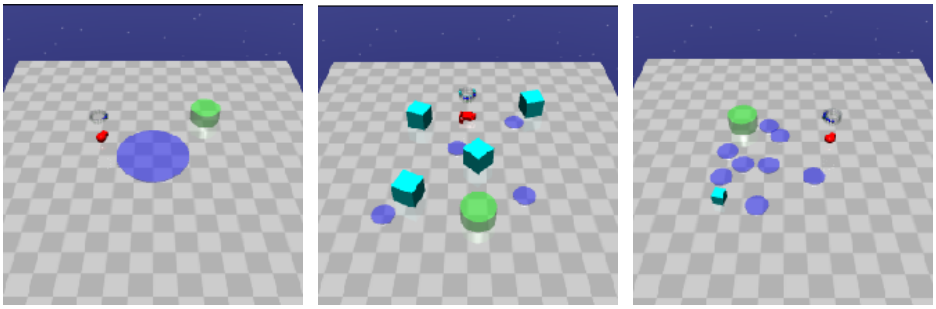
Require: $\pi^{\Diamond}, \pi^{\odot}$
1: **initialize** $\mathcal{D}^{\Diamond} \leftarrow \emptyset, \mathcal{D}^{\odot} \leftarrow \emptyset$
2: **for** each iteration **do**
3: $b \leftarrow \odot$ ▷ Start sampling from the student
4: *control-switch*$(t) \leftarrow false$
5: **for** each environment step **do**
6: $a_t \sim \pi_b(\cdot \mid s_t)$
7: $E \leftarrow (s_t, a_t, r_t^{\odot}, r_t^{\Diamond}, c_t, \mathcal{I}_t, s_{t+1})$ ▷ Generate experience
8: **if** $b = \Diamond$ **then**
9: $\mathcal{D}^{\Diamond} \leftarrow \mathcal{D}^{\Diamond} \cup \{E\}$ ▷ Save the guide samples
10: **else**
11: $\mathcal{D}^{\odot} \leftarrow \mathcal{D}^{\odot} \cup \{E\}$ ▷ Save the student samples
12: **end if**
13: **if** $\neg$ *control-switch*$(t) \wedge c_t > 0$ **then**
14: $b \leftarrow \Diamond$ ▷ Switch behavior policy
15: *control-switch*$(t) \leftarrow true$
16: **end if**
17: **end for**
18: **end for**
Output: π_b

guide policy serves as a *rescue policy* to improve safety during sampling. In addition, we leverage two replay buffers $\mathcal{D}^{\Diamond}$ and $\mathcal{D}^{\odot}$ to save the guide and student samples separately (lines 8-12), so as to control the probability $P_{\mathcal{D}^{\odot}}$ to use the more on-policy samples in $\mathcal{D}^{\odot}$. Thus, we have the probability $P_{\mathcal{D}^{\Diamond}} = 1 - P_{\mathcal{D}^{\odot}}$ to sample from $\mathcal{D}^{\Diamond}$. In practice, we train the safe guide to achieve $Q^c_{\pi^{\Diamond}}(s,a) \leq d, s \sim \mathcal{D}, a \sim \pi^{\Diamond}(\cdot \mid s)$. From the definition of $Q^c_{\pi^{\Diamond}}(s,a)$, we can basically ensure $\mathbb{E}_{(s_t,a_t)\sim\mathcal{T}'_{\pi^{\Diamond}}}\left[\sum_{t=0}^{\infty}\gamma^t c_t \middle| s_0 = s, a_0 = a\right] \leq d$ even starting with $c_0 > 0$. We denote this strategy as $b = f_{\text{cs}}(\Diamond, \odot)$.

With the *composite sampling* strategy, the function approximation may diverge, because $\pi^{\odot}$ and π_b are too different, especially when we collect most data following $\pi^{\Diamond}$. This phenomenon is related to the *deadly triad* (Sutton, Mahmood, and White, 2016). To eliminate its negative effect, we endow each sample with an *importance sampling* (IS) ratio:

$$\mathcal{I}(s,a) = \min\left(\max\left(\frac{\pi^{\odot}(a \mid s)}{\pi_b(a \mid s)}, \mathcal{I}_l\right), \mathcal{I}_u\right). \tag{5.8}$$

The clipping hyper-parameters $\mathcal{I}_u$ and $\mathcal{I}_l$ are introduced to reduce the variance of the off-policy TD target. Notice that if π_b is using the student $\pi^{\odot}$ then $\mathcal{I}(s,a) = 1$. Here, in addition to use the IS ratio $\mathcal{I}$ for learning values (the *critics*), we also use it in the policy update, as shown in line 21 of Algorithm 4.



(a) Static (b) Semi-Dynamic (c) Dynamic

Figure 5.4: Navigation tasks with different complexity levels: static, semi-dynamic and dynamic. In these environments, a robot (red) navigates in a 2D map to reach the goal position (green) while trying to avoid hazards (blue) and vases (cyan).

5.4. EMPIRICAL ANALYSIS

We evaluate how well our method transfers from the reward-free setting using the Safety-Gym engine (Ray, Achiam, and Amodei, 2019), where a random-initialized robot navigates in a 2D map to reach target positions while trying to avoid dangerous areas and obstacles (Figure 5.4). These tasks are particularly complex due to the observation space; instead of observing its location, the agent observes the other objects with a lidar sensor. We considered three environments with different complexity levels:

Static. A static environment with a point robot, and a hazard (Figure 5.4(a)). The locations of the hazard and goal are the same in all episodes.

Semi-Dynamic. A semi-dynamic environment with a car robot, four hazards, and four vases (Figure 5.4(b)). The locations of the hazards and vases are the same in all episodes. The location of the goal is random-initialized in each episode.

Dynamic. A dynamic environment with a point robot, eight hazards, and a vase (Figure 5.4(c)). The locations of the goal, vase, and hazards are random-initialized in each episode.

The *guide* agent is trained without the goals, and its auxiliary reward is the magnitude of displacement at each time step. We provide a detailed description of the safety-mapping function in Section 5.4.1. Since our focus is on the target task and the guide is trained in a controlled environment, we do not consider the guide's training in the evaluation. In the target tasks, we use the original reward signal from Safety Gym, i.e., the distance towards the goal plus a constant for finishing the task (Ray, Achiam, and Amodei, 2019). In all environments: $c = 1$ if an unsafe interaction happens, and $c = 0$

otherwise. All experiments are performed over 10 runs with different random seeds and the plots show the mean and standard deviation of all runs.

To evaluate the performance of the algorithms during training, we use the following metrics: safety of the behavior policy (Cost-Return π_b), performance of the behavior policy (Return π_b), safety of the target policy (Cost-Return $\pi^{\odot}$), and performance of the target policy (Return $\pi^{\odot}$). To check the convergence of the target policy, we have a test process with 100 episodes after each epoch (in parallel to the training) to evaluate Return $\pi^{\odot}$ and Cost-Return $\pi^{\odot}$.

5.4.1. HYPERPARAMETERS

We list the hyperparameters used in SAGUI, which are summarized in Table 5.1. As to the baselines, we use the default hyperparameters in `https://github.com/openai/safety-starter-agents`. All runs in the experiment use separate feedforward Multi-layer Perceptron (MLP) actor and critic networks. The size of the neural network (all actors and critics of the algorithms) depends on the complexity of the tasks. We use a replay buffer of size 10^6 for each off-policy algorithm to store the experience. The discount factor is set to be $\gamma = 0.99$, the target smoothing coefficient is set to be 0.005 to update the target networks, and the learning rate to 0.001. The clipping interval hyper-parameters $[\mathcal{I}_l, \mathcal{I}_u]$ are set to $[0.1, 2.0]$, while the sampling probabilities $P_{\mathcal{D}^{\diamond}}$ and $P_{\mathcal{D}^{\odot}}$ are set to 0.25 and 0.75, respectively. The maximum episode length is 1000 steps in all experiments. We set the safety constraint d based on the problem. All experiments are performed on an Intel(R) Xeon(R) CPU@3.50GHz with 16 GB of RAM.

Parameter	Static	Semi-Dynamic	Dynamic	Note
Size of networks	$(32, 32)$	$(64, 64)$	$(256, 256)$	
Size of replay buffer	10^6	10^6	10^6	$\lvert\mathcal{D}\rvert$
Batch size	32	64	256	
Number of epochs	50	100	150	
Safety constraint	5	8	25	d

Table 5.1: Summary of hyperparameters in SAGUI.

Safety-mapping function. The state spaces of the source and target task differ by the presence of the LiDAR observation of the target location. While the source task only has a safety-related signal x_c, the target task has an additional goal-related signal x_r. Thus, following the definition in Section 5.2.1, we can map the target state $[x_c, x_r]$ to the source state ignoring the target-related signal: $\Xi([x_c, x_r]) = [x_c]$.

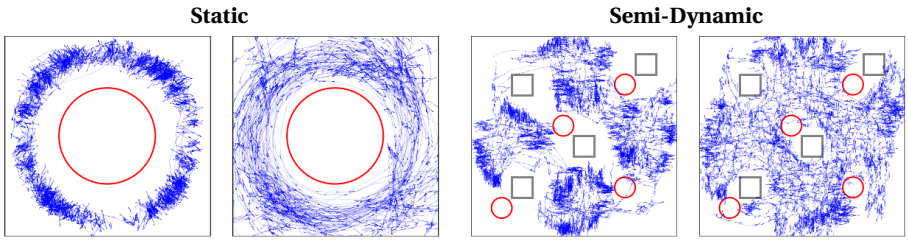


Figure 5.5: Exploration analysis. Trajectories collected by the guide agent, with and without the distance bonus, after training.

5.4.2. ABLATION STUDY

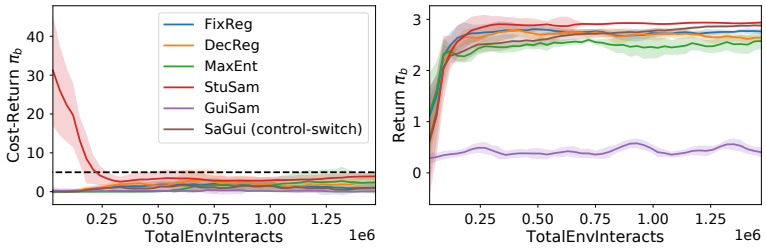
We investigate each component of the proposed SAGUI algorithm individually to answer the following questions: *i*) Does the *auxiliary reward* enlarge the exploration range? *ii*) Does a better *guide* agent result in a better student in the target task? *iii*) How does the *adaptive strength* of the KL regularization affect the performance? *iv*) How does the *composite sampling* benefit the safe transfer learning?

i) Auxiliary reward leads to more diverse trajectories. We performed an ablation of our approach where no auxiliary reward is added while training the *guide* agent, called MAXENT. We refer to the agent with the auxiliary reward as SAGUI. This teases apart the role the designed auxiliary task plays in the exploration. In Figure 5.5, we can see that SAGUI can explore larger areas in *Static* and *Semi-Dynamic*, which have the same layout in each episode. We notice that MAXENT is safe, but the explored space is limited.

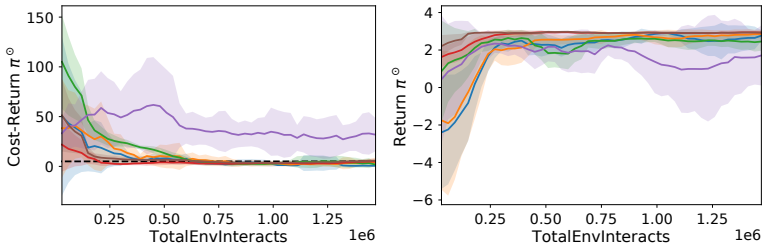
ii) An effective guide can speed up the student's training. We compare how these guides (MAXENT and SAGUI) affect the learning in the target task. In Figure 5.6, we notice that both methods can collect samples safely but the agent using the auxiliary reward needs fewer interactions to find highly performing policies.

iii) Safety-adaptive regularization improves the student's convergence rate. To combine the original reward with the bonus to follow the guide (ω), we have the following choices: fix the weights of the bonus and make it to be a hyperparameter to tune (FIXREG); apply a decay rate to linearly decrease the weights during training (DECREG); and, adapt the weights of the bonus based on the safety performance (SAGUI). In Figure 5.6(a) we observe that this weight does not affect the safety of the agent, but both FIXREG and DECREG cause the student to converge slower in terms of performance (Figure 5.6(b)).

iv) Composite sampling enhances safety and final performance. We modify the composite sampling approach, sampling only from the guide (GUISAM) or the student (STUSAM) instead. From the results in Figure 5.6(a), we can see that GUISAM can ensure safety, but the student does not learn a safe optimal policy (Figure 5.6(b)). Compared to



(a) Behavior policy



(b) Target policy

Figure 5.6: Ablation study in *Static* showing the safety and performance of the behavior policy (a) and target policy (b). The black dashed line indicates the safety threshold.

our method, STUSAM performs similarly converging to a safe target policy, but fails to satisfy the constraint at the early stage of training. So, *composite sampling* is necessary to avoid the dangerous actions from a naive policy and to ensure the target task is solved.

5.4.3. COMPARISON WITH BASELINES

Finally, we compare our algorithms SAGUI (control-switch) and SAGUI (linear-decay) with five baselines, divided into three groups.

Learning from scratch. **(1)** SAC-LAG (Ha et al., 2020) shows the performance when starting to learn from scratch, representing an off-policy algorithm. Similarly, **(2)** CPO (Achiam et al., 2017) is an on-policy algorithm that maximizes the reward in a small neighbourhood to enforce the safety constraints.

Pre-training. **(3)** CPO-PRE and **(4)** SAC-LAG-PRE are used to show how CPO and SAC-Lag perform after being pre-trained in a task that replaces the original reward function of the target task by the auxiliary reward. So, we also encourage exploration in the task for pre-training, which shares the same observation space with the target task.

Expert-in-the-loop. **(5)** As an upper bound, we also consider the Expert Guided Policy Optimization (EGPO; Peng et al., 2022) algorithm, which uses knowledge from the

target task in the form of an expert to train a student policy. EGPO proposes a guardian mechanism that shows replaces the actions of the student by the expert when the student takes actions too different from the expert. In summary, EGPO uses an expert policy as a demonstrator as well as a safety guardian.

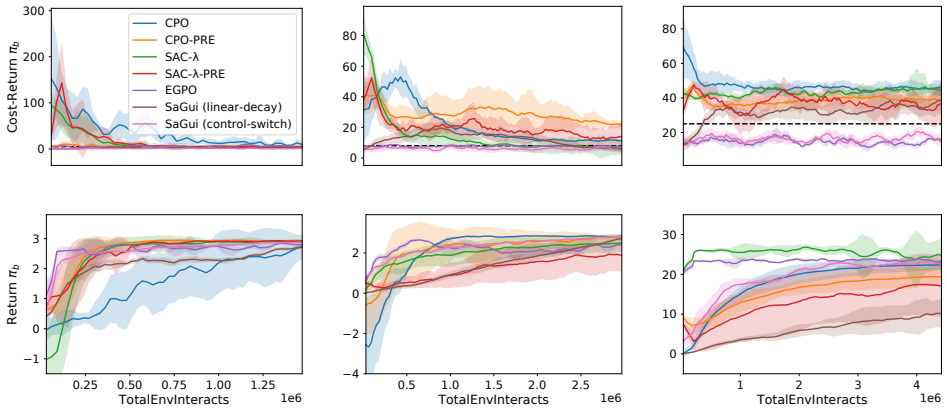
EGPO constrains safety behaviors at each timestep, which is different from our safety defined on long-term cost-return. In terms of the safe guide, EGPO assumes access to the well-performing expert policy, but our safe guide is task-agnostic. Thus, the expert in EGPO depends on the target task and does not undertake the task of exploration, while our safe guide can be useful for different reward functions and enhance the exploration capabilities of the student. Even though, EGPO can be easily adapted to our setting. The constraint of EGPO on the guardian intervention frequency can be directly transferred to be our safety constraint. Also, we do not minimize intervention anymore. Once the EGPO agent starts to take unsafe actions, the expert policy can take over the control until the end. Notice, for CPO-PRE, SAC-LAG-PRE and EGPO we adapt the source task to have the same observation space as the target task, which gives them some advantage compared to SAGUI. Even further, EGPO has access to a policy trained on the target task, while SAGUI only has access to the source task without the observations of the goal.

Safety during training. In Figure 5.7, we observe that SAGUI (control-switch) and EGPO are the only methods that exhibit safe behavior during the full training process.

Learning from scratch is unsafe and may converge to sub-optimal and even unsafe policies. SAC-LAG and CPO can learn safe policies in relatively simpler environments (*Static* and *Semi-Dynamic*) but they violate the safety constraints at the beginning of training, which is expected. In *Dynamic*, SAC-LAG and CPO fail to attain safe performance. However, with benefits from the *guide*, SAGUI (control-switch), on the basis of SAC-LAG, attains a better balance between safety and performance.

Pre-training is insufficient. With pre-training, a safe initialization cannot benefit CPO-PRE and SAC-LAG-PRE in safety, and may have negative effects. We infer that it is difficult to generalize a task when faced with a new reward signal (Igl et al., 2021). Especially for SAC-LAG-PRE with an initialized Q^r, the difficulty to adapt is evident.

Fast convergence rates. Benefiting from the targeted expert policy, the behavior policy of EGPO has a high return throughout the training in the target environment. We notice that SAGUI (control-switch) quickly finds policies with similar performance even though it has no knowledge of the target task (Figure 5.7).



(a) Static (b) Semi-Dynamic (c) Dynamic

Figure 5.7: Evaluation of π_b for CPO, CPO-PRE, SAC-Lag, SAC-LAG-PRE, EGPO, SAGUI (linear-decay), and SAGUI (control-switch) over 10 seeds. The solid lines are the average of all runs, and the shaded area is the standard deviation. The black dashed lines indicate the safety thresholds. The behavior policy (π_b), a mixture of the guide's policy ($\pi^{\Diamond}$) and the student's policy ($\pi^{\odot}$) in SAGUI, reflects the real interactions with the environment during training.

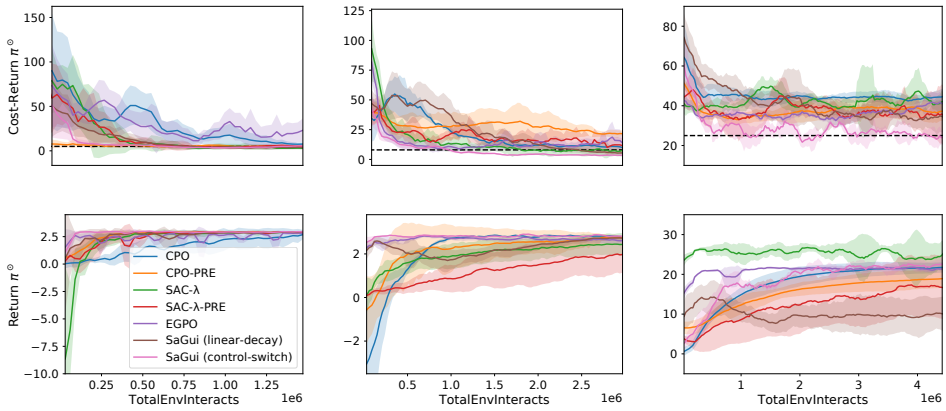
The distillation mechanism ensures the safety of the target policy. Figure 5.8 shows that SAGUI (control-switch) can learn a well-performing target policy in a safe way. Without the policy distillation mechanism like SAGUI, EGPO (learning only from the expert demonstrations) fails to find a safe target policy. This indicates that the target policy computed with SAGUI may eventually take full control of the target task, while the policy computed by EGPO may still require interventions from the expert.

Control-switch can be more effective than linear-decay. SAGUI (linear-decay), which lacks samples from $\pi^{\odot}$ at the early stage of training, does not achieve similar performance as SAGUI (control-switch). Figures 5.7(b) and 5.7(c) show that *linear-decay* fails to compose the behavior policy π_b in a safe way.

Summary. Overall, SAGUI (control-switch) does not violate the safety constraints on the target environment, quickly finds high-performing policies, and can train a student that can act independently from the teacher.

5.5. RELATED WORK AND FUTURE DIRECTIONS

As we discussed in the introduction, safe RL has multiple facets (García and Fernández, 2015), considering alternative optimization criteria (Chow et al., 2017; Yang, Simão, Tindemans, et al., 2021), and different types of prior knowledge to ensure safe exploration



(a) Static (b) Semi-Dynamic (c) Dynamic

Figure 5.8: Evaluation of $\pi^\odot$ for CPO, CPO-PRE, SAC-Lag, SAC-LAG-PRE, EGPO, SAGUI (linear-decay), and SAGUI (control-switch) over ten seeds. To check the convergence of the target policy, we have a test process with 100 episodes after each epoch. The solid lines are the average of all runs, and the shaded area is the standard deviation. The black dashed lines indicate the safety thresholds.

(Sui et al., 2015; Achiam et al., 2017; Alshiekh et al., 2018; Jansen et al., 2020; Yang, Rosca, et al., 2021). We will discuss alternatives to train the guide and how to adapt to new tasks using a pre-trained policy.

Multiple algorithms have been proposed for generalising policies from reward-free RL for better performance in target tasks (Srinivasan et al., 2020; Zhang, Cheung, et al., 2020; Gimelfarb et al., 2021). However, only (Savas et al., 2018; Miryoosefi and Jin, 2021) have considered the reward-free RL with constraints, focusing on tabular and linear settings, while we consider general function approximation algorithms.

While we considered a relatively simple strategy to achieve rich exploration, our framework allows the translation of any progress in reward-free RL into training the *guide* agent. For instance, we could adopt works with the entropy of the state density (Hazan et al., 2019; Islam, Ahmed, and Precup, 2019; Vezzani et al., 2019; Zhang, Cheung, et al., 2020; Qin, Chen, and Fan, 2021; Seo et al., 2021; Svidchenko and Shpilman, 2021). Another option to improve exploration is to find a set of diverse policies to the same problem (Kumar, Kumar, et al., 2020; Ghasemi et al., 2021; Zahavy et al., 2021). Our framework could easily combine multiple guides.

Work in transfer learning has leveraged meta-RL (Finn, Abbeel, and Levine, 2017) for safe adaptation (Grbic and Risi, 2020; Luo et al., 2021; Lew et al., 2022). Our work is also related to curriculum learning (Bengio et al., 2009; Turchetta, Kolobov, et al., 2020; Marzari et al., 2021). We first train an agent to be safe and later solve a target task. However, our approach focuses on safe exploration and is able to transfer to tasks with different

reward functions, so the guide's training is ignored. For curriculum learning, it would be interesting to consider when to stop training the guide and start training the student.

Our work also has similarities to SPACE (Yang, Rosca, et al., 2021), an on-policy constrained RL algorithm that uses different baseline policies to help the agent learn faster. In SPACE, the baseline policies are often dedicated to the target task, although they might be sub-optimal, while in our framework the guide is task-agnostic. Since we use an off-policy approach to train the student we can sample using the guide, while SPACE is an on-policy method so it always samples from the student. Finally, SPACE assumes that the state and action spaces are equal for both policies. Under different state spaces, it is not clear how to adapt SPACE even with a mapping function.

As to composite sampling strategies, recovery and shielding mechanisms (Alshiekh et al., 2018; Thananjeyan et al., 2021) could be further explored to combine with a safe guide, in particular using the control-switch mechanism that we evaluated. Nevertheless, we would like to highlight that while a student using a recovery policy would need to perform exploration alone, the safe guide can enhance the exploration capabilities of the student, speeding up the learning of the target task.

5.6. CONCLUSION

This chapter handles multiple challenges of reinforcement learning with safety constraints. It shows how we can use a safe policy (the guide) during data collection and gradually switch to a policy that is dedicated to the target task (the student). It tackles the off-policy issue that arises from collecting data with a policy different from the target policy. It shows how the student can make the best use of the guide's policy using an incentive to imitate the guide, which makes the student learn faster how to behave safely. It demonstrates that simply initializing an agent with a safe policy may not be as effective as learning a new policy dedicated to the target task through policy distillation. Finally, it proposes a method that can collect diverse trajectories, which reduces the sample complexity of the student on the target task. In summary, the framework proposed is a safe and sample-efficient way of training the agent on a target task.

6

SAFE UNSUPERVISED EXPLORATION

In this chapter, we propose to train the safe exploration policy in a more principled way. The pursuit of exploration will inevitably bring more safety risks for an agent. An under-explored aspect of reinforcement learning is how to achieve safe efficient exploration when the task is unknown. We propose the Constrained Entropy Maximization (CEM) algorithm to solve task-agnostic safe exploration problems, which naturally require a finite horizon and undiscounted constraints on safety costs. The CEM algorithm aims to learn a policy that maximizes state entropy under the premise of safety. To avoid approximating the state density in complex domains, CEM leverages a k-nearest neighbor entropy estimator to evaluate the efficiency of exploration. In terms of safety, CEM minimizes the safety costs, and adaptively tradeoffs safety to exploration based on the current safety performance. The empirical analysis shows that CEM enables the acquisition of a safe exploration policy in complex environments, resulting in improved performance in both safety and sample efficiency for target tasks.

6.1. INTRODUCTION

In RL, exploration is critical to avoid the learning agent finally converging into a suboptimal policy. However, in safety-critical domains, unlimited exploration is unacceptable (García and Fernández, 2015; Dulac-Arnold et al., 2021). For instance, while running a power network, an agent trying unlimited exploration could cause a blackout (Marot et al., 2020; Subramanian et al., 2021). Hence, encouraging exploration is bound to increase

This chapter has been published in AAAI (2023) (Yang and Spaan, 2023).

safety risks.

Many learning problems may start from an unsupervised setting. The knowledge gained can make an agent easier to achieve a variety of tasks later. For instance, when employing a safe exploration policy as a safe guide (SaGui; Yang, Simão, Jansen, et al., 2022), an agent can adapt safely and quickly to a revealed task (Chapter 5), especially when the task's reward signal is sparse. In this chapter, we focus on learning such a task-agnostic safe exploration policy. While task-agnostic exploration has been given attention (Badia et al., 2019; Hazan et al., 2019; Tao, François-Lavet, and Pineau, 2020; Liu and Abbeel, 2021; Mutti, Pratissoli, and Restelli, 2021; Seo et al., 2021), its safety aspects are still under-explored.

Prior approaches to boosting exploration usually shape the reward signal using an exploration bonus (Stadie, Levine, and Abbeel, 2015; Bellemare, Srinivasan, et al., 2016; Ostrovski et al., 2017; Pathak, Agrawal, et al., 2017; Tang, Houthooft, et al., 2017; Fox, Choshen, and Loewenstein, 2018; Haarnoja, Zhou, Abbeel, et al., 2018; Haarnoja, Zhou, Hartikainen, et al., 2018; Burda, Edwards, Pathak, et al., 2019; Burda, Edwards, Storkey, et al., 2019; Pathak, Gandhi, and Gupta, 2019; Sun et al., 2019; Seo et al., 2021). Most of them are based on a measure of state novelty to lead the agent to new unseen states. However, these typically heuristic measures are not part of the optimization objectives. They are designed to only transiently affect the process of learning, but not the final result. In contrast, to quantify exploration in a more principled way, Badia et al. (2019); Hazan et al. (2019); Tao, François-Lavet, and Pineau (2020); Liu and Abbeel (2021); Mutti, Pratissoli, and Restelli (2021); Seo et al. (2021) propose to encourage uniform coverage of the state space. With an explicit target to maximize the entropy of the state density, the interpretability of the learned exploration policy is improved significantly (Seo et al., 2021).

In safe RL, it is natural to formulate safety concerns by constraints (Achiam et al., 2017; Qin, Chen, and Fan, 2021). In this case, safety can be decoupled from reward to mitigate the issue of constructing a single reward signal that must carefully trade-off task performance and safety. When our focus is solely on efficient exploration, however, it is not clear how we can design a traditional reward to maximize the state entropy. With additional safety concerns, it is even more challenging to construct a single reward signal that is sensible for both safety and exploration. Therefore, in task-agnostic safe exploration, the need to treat safety as a constraint is exacerbated.

In safety-constrained RL problems, the discounted long-term costs are usually constrained within a pre-defined cost limit (Achiam et al., 2017; Liu, Ding, and Liu, 2020; Yang, Rosca, et al., 2020). However, for industrial and robotic settings (Jardine, Lin, and Banjevic, 2006; De Nijs, Spaan, and De Weerdt, 2015; Boutilier and Lu, 2016), the safety constraints are always built on the real costs within a finite horizon instead of the dis-

counted cost-return. For instance, a safety constraint for an electric vehicle is based on its real battery capacity, so the battery consumption cannot be discounted.

In this chapter, we aim to achieve safe and efficient exploration when the target task is unknown. We propose to formulate the problem by maximizing the entropy of the state density under safety constraints. Specifically, we designed the Constrained Entropy Maximization (CEM) algorithm, which leverages the k-nearest neighbor state entropy estimator to avoid approximating the full state density, which hardly scales to complex domains (Hazan et al., 2019). Based on the real safety costs, CEM leverages an adaptive safety weight (Lagrangian multiplier) to automatically trade off exploration and safety during policy updates instead of taking it as a reward-shaping factor. We improve the safety of the policy by calculating the gradient on the discounted cost-return but updating the safety weight following the undiscounted real costs.

Summarizing, the main contributions of this chapter can be summarized as follows: *i*) we propose a practical and approximately convergent CEM algorithm for task-agnostic safe exploration problems with convergence guarantees, *ii*) and we empirically show that CEM enables the acquisition of a safe exploration policy in complex domains, and that the policy benefits the target tasks.

6.2. TASK-AGNOSTIC SAFE EXPLORATION

In this section, we formulate the task-agnostic safe exploration (TASE) problems, where only safety signals are provided. Without the reward signal, the agent aims to explore the world safely and efficiently. The obtained policy with safe exploration capabilities may provide useful prior knowledge required to enhance the safety in potential target tasks.

In this chapter, we focus on a finite-horizon setting like the work by Mutti, Pratissoli, and Restelli (2021). Most real-world constrained RL problems are finite-horizon, and the constraints on cumulative safety costs do not include discounts, which also mitigates the problem of designing a safety threshold based on the discounted cost-return (Walraven and Spaan, 2018). For instance, an electric vehicle can take its battery capacity as the cost limit d. Naturally, we can select the horizon T in alignment with the horizon of the target task that the policy is expected to confront. When the target task is not clear, we can tune T to balance the exploration efficiency and quality (Mutti, Pratissoli, and Restelli, 2021).

In contrast to the policy entropy in Section 2.2, we address the state entropy in this chapter. We use a state density function $\rho : \mathcal{S} \mapsto \mathbb{R}_{\geq 0}$ that quantifies the distribution of states within the state space $\mathcal{S}$. When a policy π is applied to a CMDP, it influences the state distribution over time. For each time step t, the state density function $\rho_t^\pi(s)$ calculates the concentration of states at that moment. The initial state distribution ι serves as the starting point, and the policy interaction with the CMDP produces the state

density $\rho_t^\pi(s) = \rho(s_t = s|\pi)$ for each subsequent time step $t > 0$. For CMDPs with finite horizon T, the stationary density of state s can be expressed as:

$$\rho_T^\pi(s) = \frac{1}{T}\sum_{t=1}^{T}\rho_t^\pi(s), \tag{6.1}$$

which is the average state density, and $\int_{\mathcal{S}}\rho_T^\pi(s)\mathrm{d}s = 1$.

Then, we formulate the TASE problem as maximizing the entropy of the *average state density* under the premise of safety (Definition 2.1 in Chapter 2):

$$\max_{\pi\in\Pi}\mathcal{H}(\rho_T^\pi) \text{ s.t. } \mathop{\mathbb{E}}_{(s_t,a_t)\sim\mathcal{T}_\pi}\left[\sum_{t=1}^{T}c_t\right] \le d. \tag{6.2}$$

We are particularly interested in problems where the set of initial states is small since they are more challenging for task-agnostic exploration. If the trajectory can start at any state, it will be meaningless to maximize the state entropy. The best policy for the agent might then be to stay still in the initial state.

We choose to use the k-nearest neighbors (k-NN) entropy estimator $\hat{\mathcal{H}}_N^k(\rho)$ by a group of particles $\{s_i\}_{i=1}^N$ to avoid estimating the state density directly (Singh et al., 2003). In the RL process, we may need to use the samples from the current policy to estimate the state entropy of the target policy, for which we can employ an Importance-Weighted (IW) k-NN estimator $\hat{\mathcal{H}}_N^k(\rho|\rho')$ (Ajgl and Šimandl, 2011). We refer the reader to Mutti, Pratissoli, and Restelli (2021) for the detailed expression of $\hat{\mathcal{H}}_N^k(\rho|\rho')$.

6.3. SAFETY-CONSTRAINED ENTROPY MAXIMIZATION

In this section, we first clarify that traditional value function based methods are not suitable for maximizing the state entropy when the original environment reward does not exist. To achieve task-agnostic safe exploration (TASE), we will establish the duality of the original problem, then propose a practical algorithm called Constrained Entropy Maximization (CEM) for TASE with convergence guarantees.

6.3.1. VULNERABLE RELIANCE ON RETURN

Traditional RL agents learn from the reward signal when interacting with the environment. We call this original environment signal the *extrinsic reward*, and the signal designed for encouraging exploration the *intrinsic reward*. When we have no access to the extrinsic reward, it is important to ask whether we can design an intrinsic reward, such that we can solve the TASE problems by traditional RL methods.

When learning is only for exploration without extrinsic rewards, we need to design

an intrinsic reward that is stationary and implies efficient exploration of the environment in the standard RL framework. Many different intrinsic rewards are designed in previous works, e.g., count-based exploration (Bellemare, Srinivasan, et al., 2016; Ostrovski et al., 2017), prediction-based exploration (Stadie, Levine, and Abbeel, 2015; Pathak, Agrawal, et al., 2017), and auxiliary task (Fox, Choshen, and Loewenstein, 2018; Burda, Edwards, Pathak, et al., 2019). However, they are not easy to be generalized to explicitly maximize the state entropy in the task-agnostic setting. Even though we may finally get better exploration by maximizing the long-term (intrinsic) rewards, the interpretability of the exploration policy is not clear. Thus, before we have an intrinsic reward that can incentivize exploration in a principled way, it is not suitable to enhance the exploration for a policy based on the traditional optimization objective in RL.

6.3.2. DUALITY OF CONSTRAINED ENTROPY MAXIMIZATION

The standard RL algorithms that optimize long-term rewards cannot solve the TASE problem (6.2) directly. Even for constrained RL algorithms, traditional RL rewards are also necessary. Without a reward signal, the TASE problem (6.2) is dual to a problem that is solvable in a Lagrangian way. We denote the Lagrangian multiplier for $\mathop{\mathbb{E}}_{(s_t,a_t)\sim\mathcal{T}_\pi}\left[\sum_{t=1}^{T} c_t\right] \leq d$ as $\omega : \Pi \to \mathbb{R}_{\geq 0}$. Note that ω is an overall safety evaluation of the current policy and does not depend on the state. Then we consider the following optimization problem:

$$\min_{\omega\geq 0}\max_{\pi}\mathcal{F}(\pi,\omega) \doteq f(\pi) - \omega g(\pi), \tag{6.3}$$

$$\begin{aligned} &\text{where} \quad f(\pi) = \mathcal{H}(\rho_T^\pi) \\ &\text{and} \quad g(\pi) = \mathop{\mathbb{E}}_{(s_t,a_t)\sim\mathcal{T}_\pi}\left[\sum_{t=1}^{T} c_t\right] - d. \end{aligned}$$

We can solve the problem by alternating between updating the policy π and updating the safety weight ω until the Karush-Kuhn-Tucker (KKT; Gordon and Tibshirani, 2012) condition $\omega g(\pi) = 0$ is satisfied.

We search for a policy within a parametric space $\Pi_\Theta = \{\pi_\theta : \theta \in \Theta\}$. Ideally, we have two loss functions for the constrained optimization problem (6.2), i.e.,

$$\begin{aligned} J_\pi(\theta) &= \omega g(\theta) - f(\theta), \\ J_s(\omega) &= -\omega g(\theta). \end{aligned} \tag{6.4}$$

In practice, if we calculate the policy gradient based on $\mathbb{E}_{(s_t,a_t)\sim\mathcal{T}_{\pi_\theta}}\left[\sum_{t=1}^{T} c_t\right]$, the training is likely to be unstable because of the high variance in policy evaluation, especially for complex and long-horizon problems (Kakade, 2001; Peters and Bagnell, 2010). Instead,

we will optimize the policy π by using the gradient of its induced long-term discounted costs, i.e.,

$$\overline{g}(\theta) = \mathbb{E}_{(s_t,a_t)\sim\mathcal{T}_{\pi_\theta}}\left[\sum_{t=0}^{\infty}\gamma^t c_t\right] - \overline{d}, \tag{6.5}$$

where $\overline{d} = \frac{d}{T(1-\gamma)}$ is the discounted approximation of d (Section 2.2). We propose to replace $g(\theta)$ in $J_\pi(\theta)$ by $\overline{g}(\theta)$, so that we have the adapted loss function

$$J'_\pi(\theta) = \omega\overline{g}(\theta) - f(\theta), \tag{6.6}$$

but $J_s(\omega)$ remains as in (6.4), because the constraint satisfaction of a policy can be easily estimated by the real costs of the sampled trajectories. In the following, we argue that it is valid to optimize the policy π by minimizing the discounted cumulative costs until the original undiscounted cost constraint is satisfied.

Theorem 6.1. *Let the constrained optimization in (6.2) be feasible with a solution $\mathscr{S}^* = (\theta^*, \omega^*)$ that satisfies the KKT conditions, which is found by minimizing the loss functions (6.4). Then, $\overline{\mathscr{S}}^* = (\theta^*, \overline{\omega}^*)$ with $\overline{\omega}^* = \frac{\omega^*}{h'(0)}$ is a solution to the problem obtained by replacing $g(\theta)$ in $J_\pi(\theta)$ with $h(g(\theta))$, where $h : \mathbb{R} \to \mathbb{R}$ is a strictly monotone increasing function. The reverse also holds.*

Proof. For (6.2), when the KKT conditions are satisfied,

$$\nabla f(\theta^*) - \omega^* \nabla g(\theta^*) = 0.$$

By complementary slackness, $\omega^* g(\theta^*) = 0$. When $\omega^* = 0$, the constraint is not effective, and the replacement of $g(\theta)$ does not affect the results.

If $\omega^* > 0$, then $g(\theta^*) = 0$, and ω^* is the solution of $\nabla f(\theta^*) = \omega^* \nabla g(\theta^*)$. When $h(g(\theta))$ is a strictly monotone increasing function, we have

$$\nabla h(g(\theta^*)) = h'(g(\theta^*))\nabla g(\theta^*).$$

Then,

$$\nabla f(\theta^*) = \frac{\omega^*}{h'(0)}\nabla h(g(\theta^*)).$$

Therefore, $(\theta^*, \overline{\omega}^*)$ with $\overline{\omega}^* = \frac{\omega^*}{h'(0)}$ is a solution of the adjusted problem where $h(g(\theta))$ is used in the loss function. In addition, it follows that if $(\theta^*, \overline{\omega}^*)$ is a solution to the amended problem, then $(\theta^*, \overline{\omega}^* h'(0))$ is a minimizer of (6.4). □

We argue that $\overline{g}(\theta)$ is approximately an increasing monotone map of $g(\theta)$ if we have a long episode length $T \gg 1/(1-\gamma)$ and on-policy sampling at each gradient step. Then we

Algorithm 7 Constrained Entropy Maximization

Require: Initial parameters T, N, δ, λ, k, and d

initialize θ, ω, $\mathcal{D} \leftarrow \emptyset$, $\theta' \leftarrow \theta$
for each epoch **do**
  for each environment step **do**
    $a_t \sim \pi_{\theta'}(a_t|s_t)$ ▷ Sample trajectories with current policy
    $c_t \sim c(a_t|s_t)$
    $s_{t+1} \sim \mathcal{P}(s_{t+1}|s_t, a_t)$
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s_t, a_t, c_t, s_{t+1})\}$
  end for
  $\omega \leftarrow \max(0, \omega + \lambda_\omega \hat{g}(\theta'))$ ▷ Update safety weight (6.10)
  while $D_{KL}(\rho_T(\theta)||\rho_T(\theta')) \leq \delta$ **do**
    $\theta \leftarrow \theta + \lambda_\pi \nabla_\theta J_\pi(\theta)$ ▷ Policy gradient (6.13)
  end while
  $\theta' \leftarrow \theta$
  $\mathcal{D} \leftarrow \emptyset$
end for

Output: Safe exploration policy π_θ

have

$$\begin{aligned}
\overline{g}(\theta) &= \mathbb{E}_{(s_t,a_t)\sim\mathcal{T}_{\pi_\theta}}\left[\sum_{t=0}^{\infty}\gamma^t c_t\right] - \overline{d} \\
&\approx \mathbb{E}_{(s_t,a_t)\sim\mathcal{T}_{\pi_\theta}}[c_t]\cdot\sum_{t=0}^{\infty}\gamma^t - \frac{d}{T(1-\gamma)} \\
&= \mathbb{E}_{(s_t,a_t)\sim\mathcal{T}_{\pi_\theta}}[c_t]\,\frac{1}{(1-\gamma)} - \frac{d}{T(1-\gamma)} \\
&= \mathbb{E}_{(s_t,a_t)\sim\mathcal{T}_{\pi_\theta}}\left[\sum_{t=1}^{T} c_t\right]\frac{1}{T(1-\gamma)} - \frac{d}{T(1-\gamma)} \\
&= \frac{g(\theta)}{T(1-\gamma)}.
\end{aligned} \tag{6.7}$$

We see that $\overline{g}(\theta)$ is approximately an affine function of $g(\theta)$. Therefore, invoking Theorem 6.1, we can optimize the policy π for (6.2) by calculating the gradient on the discounted cost-return $\overline{g}(\theta)$, but updating the safety weight ω based on the undiscounted real costs.

6.3.3. THE CEM ALGORITHM

To solve the safety-constrained entropy maximization problem (6.2) in complex domains, we propose a CEM method (Algorithm 7) to optimize the policy within Π_Θ. At each gradient step, CEM will perform a series of fine-tuned optimizations centered around the current policy (Schulman, Levine, et al., 2015). We take the trust region as a constraint to

ensure that the optimizations are conducted within a reliable and stable neighborhood of the current policy θ'. Considering the trust-region threshold δ, we are presented with a constrained optimization problem as follows:

$$\begin{aligned} &\max_{\theta\in\Theta} \hat{\mathcal{H}}_k(\rho_T(\theta)) \\ &\text{s.t.} \begin{cases} D_{KL}(\rho_T(\theta)||\rho_T(\theta')) \leq \delta \\ \mathbb{E}_{(s_t,a_t)\sim\mathcal{T}_{\pi_\theta}}\left[\sum_{t=1}^{T} c_t\right] \leq d. \end{cases} \end{aligned} \tag{6.8}$$

Before updating the policy, we can determine the safety weight ω by evaluating the current safety performance. With the current policy parameters θ', we can sample a batch of trajectories of length T (Algorithm 7, lines 3-8). We use λ_π and λ_ω to represent the learning rate for the policy π and safety weight ω respectively. Then, we can update the safety weight (Algorithm 7, line 9) by

$$\omega \leftarrow \max(0, \omega + \lambda_\omega \hat{g}(\theta')), \tag{6.9}$$

where

$$\hat{g}(\theta') = \frac{1}{N_T}\sum_{n=0}^{N_T}\left[\sum_{t=1}^{T} c_t|(s_t,a_t)\sim\mathcal{T}_{\pi_{\theta'}}\right] - d, \tag{6.10}$$

where N_T is the number of trajectories. Then, we construct the loss function for the policy

$$J_\pi(\theta) = J_{\mathcal{H}}(\theta) + \omega J_g(\theta), \tag{6.11}$$

$$\begin{aligned} \text{where} \quad & J_{\mathcal{H}}(\theta) = -\hat{\mathcal{H}}_k(\rho_T(\theta)|\rho_T(\theta')) \\ \text{and} \quad & J_g(\theta) = \overline{g}(\theta) - \overline{g}(\theta'). \end{aligned}$$

The loss function for the state entropy $J_{\mathcal{H}}(\theta)$ is based on the IW k-NN estimator $\hat{\mathcal{H}}_N^k(\rho|\rho')$. Note that the entropy cannot be calculated directly, but needs to be estimated based on the current policy, the target policy, and the sampled particles from the current policy. We can first compute the normalized importance weight for each sample, then approximate the state density (Mutti, Pratissoli, and Restelli, 2021). We use the states in the replay buffer to estimate the entropy $\hat{\mathcal{H}}_N^k(\rho|\rho')$. Although these states are not all independent (trajectories are sampled independently, but states within a trajectory are correlated), we observe satisfactory behavior when k and the number of trajectories is sufficiently large.

Notice that we use the surrogate advantage $\overline{g}(\theta) - \overline{g}(\theta')$ to approximate our objective in minimizing the discounted safety costs, and build our loss function for safety J_g. The surrogate advantage is a measure of how the target policy π_θ performs in safety relative to

the current policy $\pi_{\theta'}$ using data from $\pi_{\theta'}$ (Schulman, Levine, et al., 2015), i.e.,

$$J_g(\theta) = \overline{g}(\pi_\theta) - \overline{g}(\pi'_\theta) \doteq \mathop{\mathbb{E}}_{(s_t,a_t)\sim\mathcal{T}_{\pi_{\theta'}}} \left[\frac{\pi_\theta(a|s)}{\pi_{\theta'}(a|s)} A^c_{\pi_{\theta'}}(s,a) \right], \tag{6.12}$$

where $A^c_\pi(s,a) = Q^c_\pi(s,a) - V^c_\pi(s)$ is the advantage function for costs, defined in Section 2.1. The surrogate advantage is designed for maximizing the long-term return, but it can be easily adapted to minimize the discounted safety costs $\overline{g}(\pi)$ in our setting. We refer the reader to Schulman, Levine, et al. (2015) for the proof of (6.12).

At each gradient step, we exploit a KL estimator $\hat{D}_{KL}(\rho\|\rho')$ to compute the trust-region constraint. We refer the reader to Ajgl and Šimandl (2011); Mutti, Pratissoli, and Restelli (2021) for the detailed derivation and expression of $\hat{D}_{KL}(\rho\|\rho')$. While the updated policy satisfies $\hat{D}_{KL}(\rho_T(\theta)||\rho_T(\theta')) \leq \delta$, we can optimize the policy several times (Algorithm 7, lines 10-12) by

$$\begin{aligned} \theta &\leftarrow \theta + \lambda_\pi \nabla_\theta J_\pi(\theta) \\ &= \theta + \lambda_\pi \nabla_\theta J_{\mathcal{H}}(\theta) + \lambda_\pi \omega \nabla_\theta J_g(\theta), \end{aligned} \tag{6.13}$$

where

$$\nabla_\theta J_g(\theta) = \frac{1}{N} \sum_{n=1}^{N} \nabla_\theta \left[\frac{\pi_\theta(a_n|s_n)}{\pi_{\theta'}(a_n|s_n)} A^c_{\pi_{\theta'}}(s_n,a_n) \right].$$

We employ the Theorem 5.1 by Mutti, Pratissoli, and Restelli (2021) to compute the gradient of the IW entropy estimator in $\nabla_\theta J_{\mathcal{H}}(\theta)$, where θ is updated without constraints. Even though we use $A^c_\pi(s,a)$ in $J_g(\theta)$, we still need to train the value functions $Q^c_\pi(s,a)$ and $V^c_\pi(s)$ during the learning process. We refer the reader to Schulman, Levine, et al. (2015) for more details of the TRPO method. Using the Lagrangian cost constraint, we leverage ω to balance safety during policy updates instead of a reward-shaping factor.

6.3.4. BOUNDS ON APPROXIMATE CONVERGENCE

In this section, we demonstrate the approximate convergence of CEM to the optimal solution by transforming the problem (6.3) into the framework by Qin, Chen, and Fan (2021). At each gradient step, CEM updates the policy based on the gradient descent algorithm with the modifications shown by Theorem 6.1, which finds the direction of the maximum increase in the entropy of the average state density but only considers the immediate surroundings of the current policy. Thus, the policy ascent is noisy due to limited samples and constrained due to the trust-region constraint. Then, the question is whether the gradient descent of the weight ω is sufficiently perturbed to no longer find a solution. Theorem 2 by Qin, Chen, and Fan (2021) has shown that a density-constrained RL algorithm can eventually converge around the optimal policy even under suboptimal policy

updates at each gradient step.

The amended Lagrangian optimization in Section 6.3.2 can be written as:

$$\begin{aligned} &\max_{\omega \geq 0} \mathcal{F}(\omega), \\ &\text{where } \mathcal{F}(\omega) = \omega g(\rho^*(\omega)) - \mathcal{H}(\rho^*(\omega)) \\ &\text{and } \rho^*(\omega) = \arg\min_{\rho} \omega \overline{g}(\rho) - \mathcal{H}(\rho). \end{aligned} \tag{6.14}$$

In this representation, the state density $\rho = \rho_T^\pi$ is implicitly generated by the policy π. The associated discounted safety costs can be expressed as

$$\overline{g}(\rho) = \int_S \rho_T^\pi(s) V_\pi^c(s) \mathrm{d}s - \overline{d} \approx \frac{1}{1-\gamma} \left[\int_S \rho_T^\pi(s) c(s) \mathrm{d}s - \frac{d}{T} \right],$$

where $\overline{g}(\rho)$ is (approximately) affine in ρ, and we assumed that costs $c(s)$ are incurred by the presence in states, not by actions. We note that the optimized expression $\omega \overline{g}(\rho) - \mathcal{H}(\rho)$ in Eq. 6.14 is strongly convex, because $-\mathcal{H}$ is strongly convex and $\overline{g}(\rho)$ is (approximately) affine in ρ. We set its modulus to be μ, which measures the degree of convexity.

6

We optimize ω to achieve $\max_{\omega \geq 0} \mathcal{F}(\omega)$. The set of its optimal solutions is denoted as $\Omega^* = \{\omega | \mathcal{F}(\omega) = \max_{\omega \geq 0} \mathcal{F}(\omega)\}$, with $\overline{\omega}^* \in \Omega^*$ in line with Theorem 6.1. For a given ω, we use the TRPO method (with discounted safety costs) to solve the state density optimization problem. For a suboptimal update in policy, we assume the imperfect solution $\hat{\rho}$ satisfies

$$\omega g(\hat{\rho}) - \mathcal{H}(\hat{\rho}) - \mathcal{F}(\omega) \leq \epsilon.$$

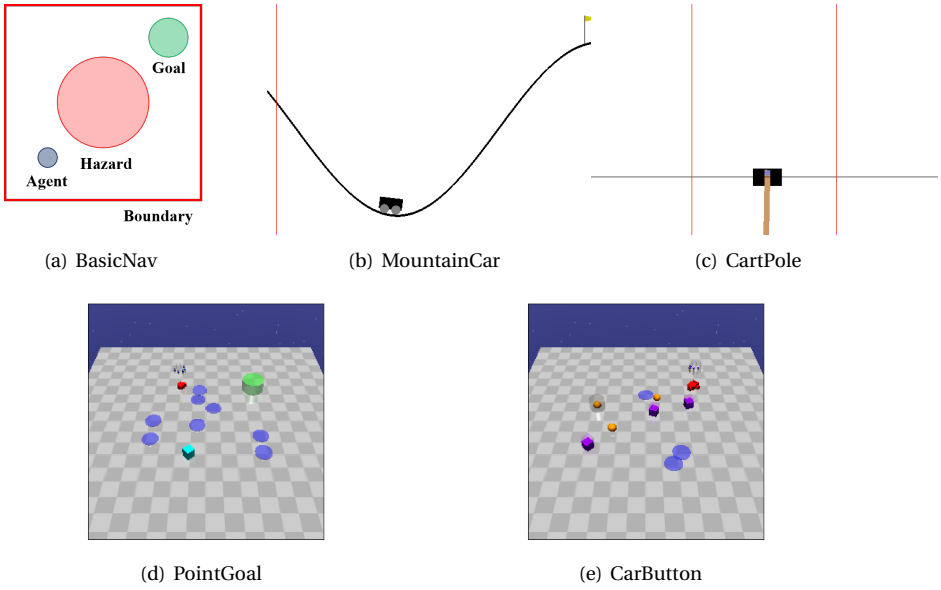
The corresponding update in the safety weight is $\omega \leftarrow \max(0, \omega + \lambda_\omega \nabla \hat{\mathcal{F}}(\omega))$, where $\nabla \hat{\mathcal{F}}(\omega) = g(\hat{\rho})$. Then, we can invoke Lemma 2 and Theorem 2 by Qin, Chen, and Fan (2021) to get the following convergence result.

Convergence result. For a step size $\lambda_\omega \leq \mu$, CEM with suboptimal policy updates will converge to a $\hat{\omega}$ that satisfies

$$\min_{\omega' \in \Omega^*} \|\hat{\omega} - \omega'\| \leq \psi \sqrt{\epsilon / \mu}$$

with constant $\psi > 0$. With another constant $\xi > 0$, $\mathcal{F}(\hat{\omega})$ also converge to a bounded neighborhood of its optimal value:

$$\min_{\omega' \in \Omega^*} \|\mathcal{F}(\hat{\omega}) - \mathcal{F}(\omega')\| \leq \xi \epsilon / \mu^2.$$



(a) BasicNav (b) MountainCar (c) CartPole

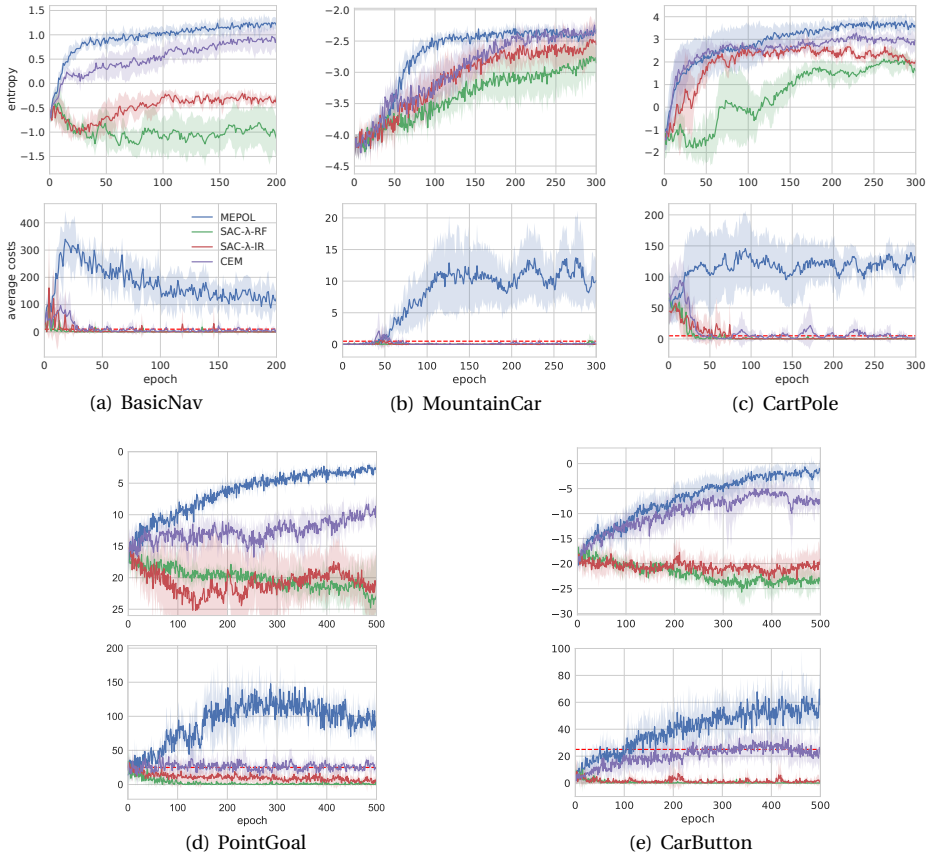
(d) PointGoal (e) CarButton

Figure 6.1: Safety-constrained exploration tasks with different complexity levels, i.e., the state spaces, the type of the obstacles, and the potential downstream tasks.

6.4. EMPIRICAL ANALYSIS

We evaluate our method based on a wide variety of TASE benchmarks. We organize our empirical analysis as follows: 1) We demonstrate that CEM can facilitate learning a safe exploration policy in various complex environments; 2) We reveal that the safe exploration policy can benefit the target tasks in safety and sample efficiency.

Benchmarks. We first evaluate our safe unsupervised exploration in a 2D navigation domain BasicNav (2D states, Figure 6.1(a)), where a hazard in the center should be avoided. Then, we consider two simple environments: MountainCar (2D, Figure 6.1(b)) and CartPole (4D, Figure 6.1(c)). Note that they are different from the original versions in OpenAI Gym (Brockman et al., 2016) because of the additional constraints. In MountainCar, the constraint is to not go too far to the left (indicated by the red line in Figure 6.1(b)), every step the cart is too far on the left a cost of 1 is received. In CartPole, the constraint is to keep the cart in a certain region. Finally, we test our method in two complex environments from the Safety Gym suite (Todorov, Erez, and Tassa, 2012; Ray, Achiam, and Amodei, 2019): PointGoal (36D, Figure 6.1(d)), CarButton (56D, Figure 6.1(e)). In PointGoal, we control the point robot to navigate in the 2D map to reach a goal while trying to avoid a vase and several hazards. In CarButton, we control a more complex car robot to



(a) BasicNav (b) MountainCar (c) CartPole

(d) PointGoal (e) CarButton

Figure 6.2: Comparison of MEPOL, SAC-Lag-RF, SAC-Lag-IR, and CEM during training in exploration (top row) and safety (bottom row). The solid lines are the average of all runs, and the shaded area is the standard deviation. The red dashed lines indicate the safety thresholds.

push the right button while trying to avoid the wrong button, several moving gremlins, and several fixed hazards. In all environments: $c = 1$ if an unsafe interaction happens, and $c = 0$ otherwise. All experiments are performed over 10 runs with different random seeds and the plots show the mean and standard deviation of all runs.

Environment builder. We use the Safety Gym engine (https://github.com/openai/safety-gym/blob/master/safety_gym/envs/engine.py) to build the PointGoal and CarButton environments (Ray, Achiam, and Amodei, 2019). Two configuration dictionaries are used to specify the size of the map, the type of the robot, the task to finish, the size and location of the goal, the signals the agent can receive, and the size and location of the hazard. The environments are created through the configuration file:

```
import safety_gym
from gym.envs.registration import register

register(id='PG-v0'/'CB-v0',entry_point='safety_gym.envs.mujoco:Engine',kwargs={'
    config': config_pg/config_cb})

config_pg = {'task': 'goal',
    'robot_base': 'xmls/point.xml',
    'observe_goal_lidar': True,
    'observe_box_lidar': True,
    'lidar_max_dist': 3,
    'lidar_num_bins': 8,
    'goal_size': 0.3,
    'goal_keepout': 0.305,
    'hazards_size': 0.2,
    'hazards_keepout': 0.1,
    'constrain_hazards': True,
    'observe_hazards': True,
    'observe_vases': True,
    'placements_extents': [-1.5, -1.5, 1.5, 1.5],
    'hazards_num': 8,
    'vases_num': 1}

config_cb = {'task': 'button',
    'robot_base': 'xmls/car.xml',
    'observe_goal_lidar': True,
    'observe_box_lidar': True,
    'lidar_max_dist': 3,
    'lidar_num_bins': 8,
    'buttons_num': 3,
    'buttons_size': 0.1,
    'buttons_keepout': 0.1,
    'observe_buttons': True,
    'hazards_size': 0.2,
    'hazards_keepout': 0.1,
    'gremlins_travel': 0.1,
    'gremlins_keepout': 0.1,
    'constrain_hazards': True,
    'constrain_buttons': True,
    'constrain_gremlins': True,
    'observe_hazards': True,
    'observe_gremlins': True,
    'placements_extents': [-1.5, -1.5, 1.5, 1.5],
    'hazards_num': 3,
    'gremlins_num': 3}
```

Hyperparameters. We list the hyperparameters used in CEM, which are summarized in Table 6.1. The specific hyperparameters of the SAC-λ baselines are listed in Table 6.2. As to MEPOL, we use the same parameters as CEM except for the safety-related ones. The discount factor for the cost-return in CEM is set to be $\gamma = 0.99$, which is also used in the SAC-λ baselines. In CEM, the maximum iterations to reach the trust-region threshold is 30, and the activation functions in the policy networks and value networks are ReLU. In all experiments, the number of episodes is 20. We set the safety constraint d based on the problem. As to the evaluation of safe transfer learning, we use the hyperparameters for the Dynamic environment in SaGui, listed in Table 5.1. The rest of the hyperparameters are explained in the Empirical Analysis part of the paper. All experiments are performed on an Intel(R) Xeon(R) CPU@3.50GHz with 16 GB of RAM.

Parameter	BasicNav	MountainCar	CartPole	PointGoal	CarButton	Note
Number of epochs	200	300	300	500	500	
Number of neighbors	50	4	4	4	4	k
Safety constraint	10	0.5	5	25	25	d
Learning rate of policy	0.00001	0.0001	0.0001	0.00001	0.00001	λ_π
Learning rate of safety	0.001	0.01	0.01	0.01	0.01	λ_ω
Trajectory length	1200	400	300	500	500	T
Trust-region threshold	1.0	0.5	0.5	0.1	0.1	δ
Size of policy networks	[300,300]	[300,300]	[300,300]	[400,300]	[400,300]	π
Size of value networks	[64,64]	[64,64]	[64,64]	[64,64]	[64,64]	V_C

Table 6.1: Summary of hyperparameters in CEM.

Parameter	BasicNav	MountainCar	CartPole	PointGoal	CarButton
Policy entropy weight	0.2	0.2	0.2	0.2	0.2
Learning rate	0.001	0.001	0.001	0.001	0.001
Batch size	32	32	32	256	256
Size of networks	[32,32]	[32,32]	[32,32]	[256,256]	[256,256]
Size of replay buffer	10^6	10^6	10^6	10^6	10^6
Target smoothing coefficient	0.005	0.005	0.005	0.005	0.005

Table 6.2: Specific hyperparameters in the SAC-λ baselines.

6.4.1. EVALUATION OF SAFE EXPLORATION

During training, the agent is not aware of the extrinsic environment reward, i.e., $r(s,a) = 0, \forall s \in \mathcal{S}, a \in \mathcal{A}$. We have identified the entropy value $\hat{\mathcal{H}}_k(\rho_T(\theta))$ and the average episodic costs over each epoch as the key metrics to assess the effectiveness of the policy. We hand-tuned hyperparameter k to attain reasonable performance of the entropy estimator. Similar to the work by Mutti, Pratissoli, and Restelli (2021), we chose the horizon T according to the potential task for the agent in each specific environment. To evaluate

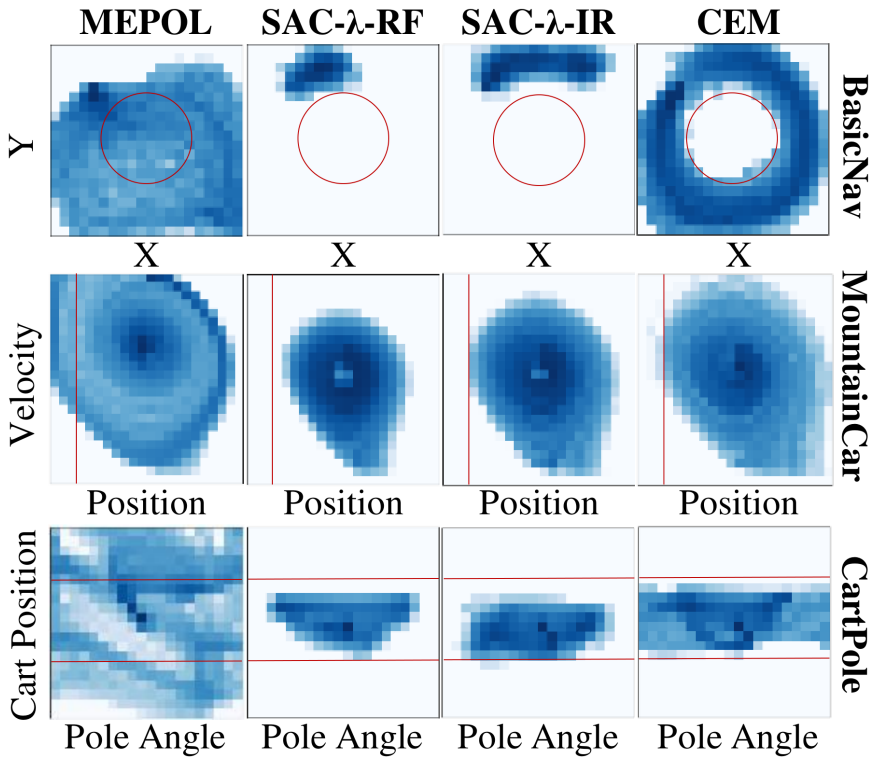


Figure 6.3: Exploration analysis after training. The heat maps show the final state density of the learned policies. The red line indicates the dangerous area.

how our method performs in pure safe exploration tasks, we compare CEM with three baselines:

MEPOL To show how well the agent can explore the world without taking into account any safety concerns, we also take MEPOL as a baseline, which is the state-of-the-art algorithm in maximizing the state entropy (Mutti, Pratissoli, and Restelli, 2021).

SAC-Lag-RF To efficiently explore the world, we first consider SAC-Lag (Ha et al., 2020) to maximize the policy entropy under the safety constraints with $r(s,a) = 0 : \forall s \in \mathcal{S}, a \in \mathcal{A}$, rather than optimize the state entropy directly.

SAC-Lag-IR Inspired by the off-policy version for efficient exploration in the work by (Seo et al., 2021), we introduce an auxiliary reward $r(s) := \log(\|s - s^{k\text{-NN}}\|_2 + 1)$ to further enhance the exploration under the framework of SAC-Lag (Ha et al., 2020).

As we show in Figure 6.2, compared to the safe methods (SAC-Lag-RF, SAC-Lag-IR, and

CEM), MEPOL shows the ability to acquire policies with remarkably strong exploration across all domains, but it does not satisfy the safety constraint. Both SAC-Lag-RF and SAC-Lag-IR can converge to safe policies even in more complex environments. In the classic control environments MountainCar and CartPole, the two SAC-Lag methods can also make prominent improvements in state entropy, but failed in BasicNav (Figure 6.2(a)), PointGoal (Figure 6.2(d)), and CarButton (Figure 6.2(e)). In general, with the benefits from the intrinsic reward, SAC-Lag-IR attained higher state entropy than SAC-Lag-RF. Compared to all the baselines, only CEM managed to learn a policy that finally gets remarkable results in exploration and satisfies the safety constraint.

After training, we leverage the heat maps in Figure 6.3 to show the exploration of the final policies in the illustrative environments BasicNav, MountainCar, and CartPole. Note that the states in CartPole are 4D, but we just focus on the cart position and pole angle. We can see that MEPOL can always achieve efficient exploration in all environments. However, the unsafe areas are also covered by the learned agents. The exploration heat maps also show that the two SAC-Lag methods are too conservative in safety. Even though SAC-Lag-IR is generally better than SAC-Lag-RF, the learned agent cannot cover the safe areas well, especially in BasicNav and CartPole. Only CEM can efficiently explore the safe areas in all the illustrative environments.

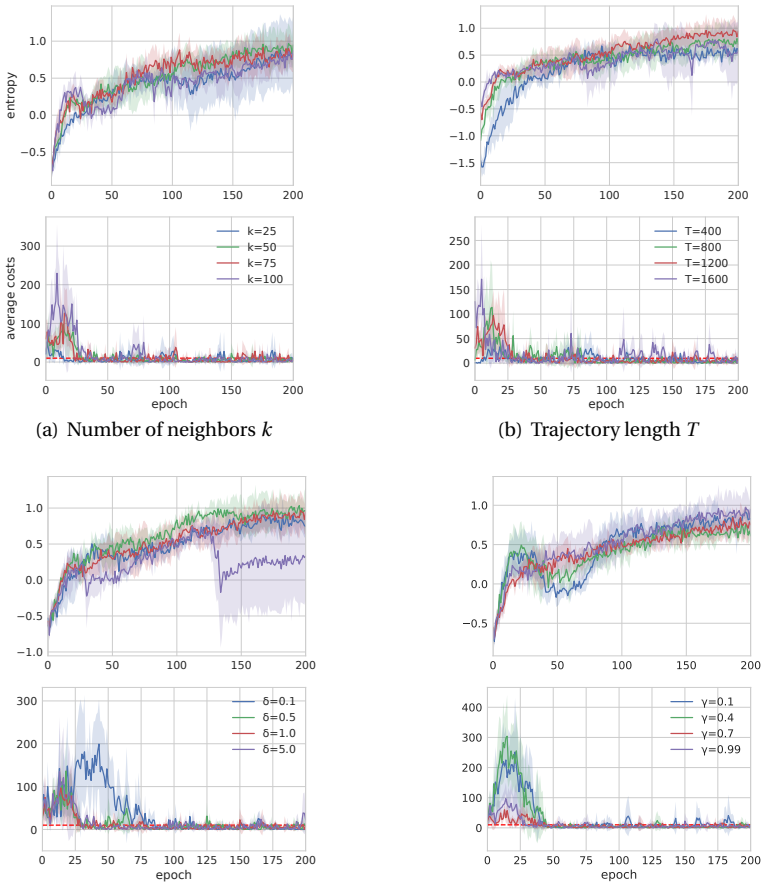
6.4.2. PARAMETER SENSITIVITY

In this section, we examine the effect of altering the parameters k (number of neighbors for entropy estimation), T (finite horizon/trajectory length), δ (trust-region threshold to determine the gradient step size), and γ (discount factor) on both the entropy and safety costs in the BasicNav environment.

From Figures 6.4(a) and 6.4(d), it is obvious that CEM is not sensitive to the number of neighbors k and the discount factor γ in exploration and safety. In terms of the trajectory length (Figure 6.4(b)), a short T makes the learning in exploration slow compared to the curves with much longer trajectories, but the difficulty to be safe will be increased if we have a longer T. A high trust-region threshold δ negatively impacts learning stability, and could cause sudden performance degradation in exploration, as shown in Figure 6.4(c). A low δ will make the learning very slow in safety, but the performance of the final policy is not affected.

6.4.3. EVALUATION OF SAFE TRANSFER LEARNING

In this section, we evaluate how a safe exploration policy learned by CEM can benefit the target tasks in safety and sample efficiency. To evaluate the policy in safety, we use the safety costs generated during the interaction with the environment. In terms of



(a) Number of neighbors k

(b) Trajectory length T

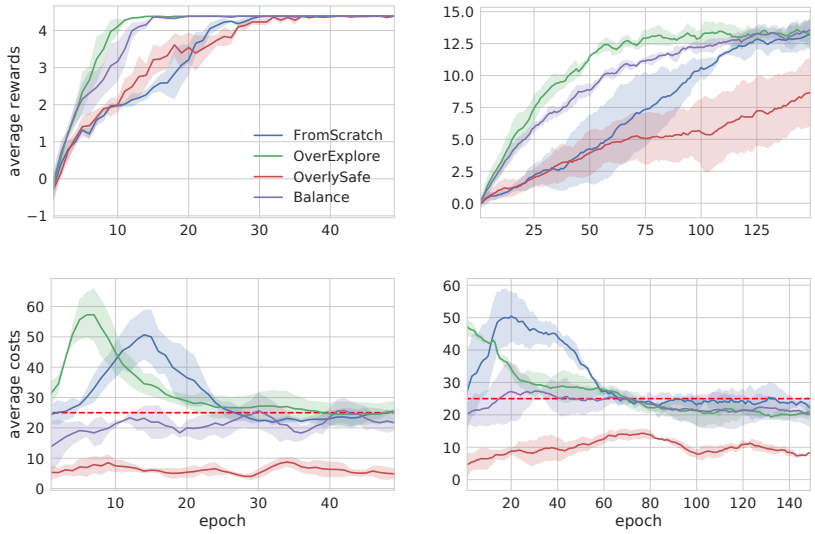
(c) Trust-region threshold δ

(d) Discount factor γ

Figure 6.4: Parameters $\langle k, T, \delta, \gamma \rangle$ sensitivity analysis in BasicNav. The solid lines are the average of all runs, and the shaded area is the standard deviation. The red dashed lines indicate the safety thresholds.

performance, we use the average episodic rewards over 100 episodes in an extra test process after each epoch. In the downstream tasks, the extrinsic environment reward is revealed to the agent. We leverage the safe exploration policy to guide learning in the off-policy safe guide (SaGui; Yang, Simão, Jansen, et al., 2022) framework (control-switch version in Chapter 5), which achieves safe transfer learning by two mechanisms:

- Adaptively regularize the student policy to the guide policy based on the student's safety;
- Use the safe exploration policy as a recovery policy when the student starts to take unsafe actions.



(a) PointGoal (b) CarButton

Figure 6.5: Effects of the safe exploration policies on the downstream tasks. The solid lines are the average of all runs, and the shaded area is the standard deviation. The red dashed lines indicate the safety thresholds.

To show how the quality of the safe exploration policies plays a role in learning, we use the policy learned by CEM to represent a teacher with a good balance between safety and exploration (Balance). For comparison, we use the policy learned by MEPOL to represent an unsafe teacher but with efficient exploration over the whole state space (OverExplore), so we deactivate the recovery mechanism in SaGui to avoid worse results from this unsafe guide. On the other hand, we also use the policy learned SAC-Lag-RF, which is safe but very conservative in exploration (OverlySafe). We also take the agent that starts learning from scratch (FromScratch) as a baseline.

In Figure 6.5, we show how the quality of the safe exploration policies influences the learning in the downstream tasks, where the agent needs to reach the goal in PointGoal, and push the right button in CarButton. In general, we can observe that the different safe exploration policies benefit the downstream tasks in different ways. The agent guided by the OverExplore policy can learn to get high rewards quickly, but cannot get obvious improvement in safety compared to learning from scratch. The OverlySafe policy can stay the agent to be absolutely safe when interacting with the environment. However, the resulting performance is even worse than learning from scratch. The policy learned by CEM (Balance) can guide the agent to obtain high rewards quickly under the condition of ensuring the safety of training.

In this experiment, we opt to assess our proposed method in two selected environ-

ments out of the five available, specifically targeting those deemed more complex. This deliberate choice is motivated by the objective of conducting a comprehensive analysis of the method's performance in challenging scenarios characterized by intricate dynamics and higher-dimensional state-action spaces. While the remaining three environments are not included in this particular experiment, they present potential avenues for future investigations within the broader problem domain.

6.5. RELATED WORK

Task-agnostic exploration has been studied in three different directions (Mutti, Pratissoli, and Restelli, 2021), estimating the environment dynamics (Jin et al., 2020; Tarbouriech, Shekhar, et al., 2020), learning a transferable meta-reward function (Zheng, Oh, et al., 2020; Bechtle et al., 2021), and learning an efficient exploration policy (Hazan et al., 2019; Tarbouriech and Lazaric, 2019; Mutti and Restelli, 2020; Guo et al., 2021; Mutti, Pratissoli, and Restelli, 2021; Nedergaard and Cook, 2022). These works made impressive progress in exploring the environment efficiently without a reward signal (Laskin, Yarats, et al., 2021). Nevertheless, task-agnostic exploration with safety concerns is still under-explored. Compared to our method, their learned policies cannot explore safely, which is important when we need to explore the real world and the downstream tasks are safety critical.

The constrained cross-entropy method proposed by Wen and Topcu (2018) could be extended to the TASE problem, but its efficiency under the state-entropy maximization objective has not yet been tested. To some extent, SAC-λ can also be used to solve our problem. By maximizing the policy entropy, the agent trained by SAC-λ tends to have diverse behaviors, but it does not imply efficient exploration of the environment. With an additional intrinsic reward, the exploration of SAC-λ can be enhanced (Yang, Simão, Jansen, et al., 2022), but the interpretability of the learned policy in exploration is not clear. Achiam et al. (2017); Liu, Ding, and Liu (2020); Yang, Rosca, et al. (2020) propose a series of constrained policy optimization methods, where the constraints are built on long-term costs instead of real costs within a finite horizon. To apply their methods in our domain, more work is needed to process the different optimization objectives and constraints.

6.6. CONCLUSION

In this chapter, we propose the CEM algorithm to solve safety-constrained entropy maximization problems in a completely reward-free manner. We argue that it is more practical to formulate the problem to be finite-horizon without discounting, which mitigates the problem of designing a safety threshold based on the discounted cost-return. To trade off exploration with safety, we adaptively change the safety weights based on the undis-

counted real costs. Accordingly, we can update the policy under the adjusted balance between safety and exploration. The learned policy can maximize exploration under the premise of safety even in complex continuous-control domains, and benefit the potential downstream tasks in sample efficiency and safety.

7

CONCLUSION

This dissertation focuses on two challenges that hinder the application of RL in the real world. Firstly, potential safety risks exist because of the randomness in long-term costs, which is generated by the stochastic policy and the dynamics of the environment. A worst-case analysis is required to limit the frequency of very unsafe outcomes. Thus, we propose to define safety based on varying risk requirements that the user can indicate. Accordingly, we present the safety-constrained RL algorithms with risk control. Secondly, many real-world RL problems do not have simulators of sufficient fidelity, so interactions with safety-critical environments are inevitable. In this case, safety during training needs to be guaranteed. However, it is not possible to be absolutely safe if learning from scratch. We propose to train and transfer a safe exploration policy to ensure safety during training and also improve sample efficiency. In Section 7.1, we first answer the research questions raised in Section 1.4, and summarise our contributions to the main research goal. Then, in Section 7.2, we elaborate on the limitations of our work and suggest future directions. Finally, in Section 7.3, we expound our final thoughts on some other aspects that hinder RL to be more applicable in the real world.

7.1. ANSWERS TO THE RESEARCH QUESTIONS

We briefly summarise the answers to the research question and sub-questions below.

How to control risks and ensure safety during training in reinforcement learning?

The overall aim of this dissertation is to answer this main question, so as to promote the application of RL in the real world. However, it is impossible for us to cover all

the critical factors for real-world RL applications. We select two perspectives, i.e., risk control in safety-constrained RL, and safety assurance during training, to answer the main question, which is addressed by answering the following four sub-questions.

Q1 How to formulate the safe RL problem with risk control?

This sub-question is answered in Chapter 3, where we analyze the vulnerable reliance on expected cost-return. In this case, the learned policies are unaware of the potential risks because of the randomness generated by the stochastic policy and the dynamics of the environment. To have a lower risk of hazardous events even under stochastic or heavy-tailed cost-return, we consider the uncertainty in safety. Given the risk level, we can define the safety based on the corresponding upper tail of the cost distribution, represented by the conditional Value-at-Risk (CVaR). In this way, policies can be optimized under different levels of CVaR, which determine the degree of risk aversion from a safety perspective.

Q2 How to optimize the policy under the premise of safety?

In Chapters 3 and 4, we present two versions of Worst-Case Soft Actor Critic algorithms with different ways to estimate the cost-return distribution, namely a Gaussian approximation and a quantile regression algorithm. The Gaussian approximation is simple and easy to implement, but may underestimate the safety cost, and the quantile regression leads to a more conservative behavior. Thus, under the given risk level, we can approximate the CVaR from the distribution to guide the change of adaptive safety weights to achieve a trade-off between reward and safety. As a result, we can compute policies whose worst-case performance satisfies the constraints. The empirical analysis shows that both versions of WCSAC attain better risk control compared to the expectation-based methods, and the quantile regression version shows strong adaptability in complex safety-constrained environments.

Q3 How to ensure safety during training by transferring safe exploration policies?

This sub-question was answered in Chapter 5, where we present the safe guide (SaGui) framework to leverage a safe exploration policy to ensure safety during training and also improve sample efficiency. A safe exploration policy, trained in a constrained reward-free setting, can be taken as a universal guide to enhancing safety and learning in the potential target tasks, where safety violations are not allowed during training. The safe exploration policy is leveraged to compose a safe behavior policy that directly interacts with the environment for sampling. Also, we regularize the target policy towards the safe exploration policy while the student is unreliable, and gradually eliminate the influence from the guide as training progresses. The empirical analysis shows that this method can achieve safe transfer learning and helps the student solve the target task faster.

Q4 How to train a safe exploration policy in a principled way?

This sub-question was answered in Chapter 6, where we present the practical Con-

strained Entropy Maximization (CEM) algorithm to solve task-agnostic safe exploration (TASE) problems. The CEM algorithm aims to learn a policy that maximizes state entropy under the premise of safety. To avoid approximating the state density in complex domains, CEM leverages a k-nearest neighbor (k-NN) entropy estimator to evaluate the efficiency of exploration. In terms of safety, CEM minimizes the advantage in cost-return, and adaptively tradeoffs safety to exploration based on the current safety performance. We empirically show that CEM allows learning a safe exploration policy in complex domains, and under the SaGui framework, the learned policy benefits downstream tasks in safety and sample efficiency.

Conclusions on the main research goal

To address each of the four subquestions identified in response to the two perspectives of the main research question, we have proposed novel formulations for safety-constrained RL problems, and designed corresponding algorithms. Since our proposed formulations and algorithms cover different perspectives in safe RL, they can complement each other. With the WCSAC algorithms for risk-averse constrained RL, we can learn safe policies under different risk levels, but safety during training is ignored. Instead, the SaGui framework aims to ensure safety during training by leveraging a safe exploration policy. Under different domain conditions, the proposed algorithms can be used alone or in combination. In summary, by addressing safety from the two perspectives, our work significantly improves the reliability and practicality of RL for real-world use.

7.2. LIMITATIONS AND FUTURE WORK

This dissertation only focuses on the two perspectives in safety-constrained RL, but more extensive research is necessary to bring RL closer to real-world applications. We also made assumptions and simplifications on the proposed formulations and algorithms, thereby leaving room for further development. Below we suggest future research directions that can further consolidate the safety mechanism in RL.

More complex scenarios

In complex safe RL scenarios, such as those with multiple constraints, our proposed methods rely on more networks for policy optimization and constraint enforcement. However, updating several networks can be computationally expensive and time-consuming. To improve scalability, approaches such as network architecture optimization, parallel computing, approximate worst-case estimation, and hardware acceleration can be employed. These techniques reduce computational and memory requirements, speed up training through parallelization, and utilize specialized hardware. Ongoing research is also exploring scalable safe RL algorithms. When faced with contradictory constraints that do not allow for a feasible solution, we infer that our Lagrangian methods will increase

the penalty terms associated with the conflicting constraints. This leads to a higher cost or objective function value. The algorithms adjust the Lagrange multipliers to balance the penalties, striving to find a compromise between the conflicting objectives. The methods aim to minimize the overall cost while considering the trade-offs imposed by the conflicting constraints, seeking a policy that satisfies the constraints as much as possible while still maximizing the expected reward. However, in cases of severe conflicts where no feasible solution exists, our methods may converge to suboptimal or infeasible solutions. We can further explore the constrained RL problems with multiple constraints, especially when they are conflicting.

Intrinsic uncertainty and parametric uncertainty

We can further explore modeling the uncertainty of the cost-return in different ways. We have shown the versatility of WCSAC by using two methods to approximate the cost-return distribution. Therefore, WCSAC can be further improved with any progress in distributional RL. Our method captures the randomness in the cost-return intrinsic to the CMDP, but not the uncertainty in approximating the value function or the cost-return distribution, i.e., the *parametric uncertainty* (Dearden, Friedman, and Russell, 1998; Engel, Mannor, and Meir, 2005). Approximation error of parameters also brings potential risks for the learned policies. Hence, the safety in RL can be further strengthened if we can also take the parametric uncertainty into account.

Difference between transfer tasks

Safe transfer learning can be formulated with larger differences between the tasks. In the SaGui framework, we assume the safe exploration policy is trained in a controlled environment, which allows unsafe interactions. Mostly, we have no access to a simulator with enough fidelity. Thus, the learned policy cannot be deployed in the real world directly. Even though we consider the observation spaces are different between source and target tasks, there is still considerable space to explore the settings with larger differences between the tasks (Cutler, Walsh, and How, 2014), e.g., incompatible environment dynamics. In this case, the SaGui framework needs to meta-learn the safe guide on a diverse set of environments to adapt to a wide range of tasks.

Knowledge for safe transfer learning

Training and transferring safe exploration policies is not the only way to safe adaptation. The SaGui framework leverages the knowledge learned from a reward-free setting. In a simulator, it is also possible that we can freely generate different tasks of the same type. Meta-RL addresses the fast adaptation challenge by leveraging knowledge learned from training different tasks to perform well in previously unseen tasks (Finn, Abbeel, and Levine, 2017; Rakelly et al., 2019). Thus, we may meta-learn some knowledge, e.g., a transferable meta-reward function, or a meta policy, from the limited or unlimited amount of tasks in the simulator to benefit the real-world learning process (Grbic and

Risi, 2020; Luo et al., 2021; Lew et al., 2022). Another direction is to leverage the epistemic uncertainty about the safety dynamics to ensure safety also during training (Zheng and Ratliff, 2020; Simão, Jansen, and Spaan, 2021).

State density estimation and state abstraction

We leverage the k-NN entropy estimator to avoid approximating the state density in high-dimensional observation spaces. On the one hand, it is still an open problem to scale the state density estimation to complex domains (Hazan et al., 2019). With more efficient ways to approximate the state density, TASE problems can be further extended by more complex convex constraints (Miryoosefi, Brantley, et al., 2019; Qin, Chen, and Fan, 2021). On the other hand, we treat all dimensions in the observation equally, which may reduce the efficiency of optimization, and not accurately capture the concerns of the task. Instead, we could learn a low-dimensional latent representation space to better express our exploration focus, but the complexity and instability of the algorithm will increase, especially when we have additional safety concerns. For this reason, it is essential to investigate how to efficiently abstract the observation space in TASE problems (Tao, François-Lavet, and Pineau, 2020; Liu and Abbeel, 2021; Seo et al., 2021; Yarats et al., 2021).

Different exploration objectives

The safe exploration policy is trained by maximizing the state entropy under the premise of safety. The resulting policy is able to explore the state space in a diffuse manner without compromising safety. However, a policy that maximizes the state entropy may not visit all transitions (Zhang, Cai, and Li, 2021). In some cases, e.g., the reward is related to the state-action pairs, covering all the transitions is more important than covering all the states. It is a promising direction for future work to consider more different pretraining settings, e.g., maximum entropy over the state-action pairs (Zhang, Cai, and Li, 2021), maximum mutual information between tasks and policy-induced states (Liu and Abbeel, 2021) or between state-transitions and latent skill vectors (Laskin, Liu, et al., 2021), pretraining in a class of multiple environments (Mutti, Mancassola, and Restelli, 2022), and pretraining for history-based policies (Mutti, De Santi, and Restelli, 2021).

Risk-averse safe transfer learning

The proposed approaches in this dissertation cover different perspectives and complement each other. Nevertheless, it is worth to further explore how to use them in combination. The safety in SaGui is defined to be risk-neutral, but SaGui can be easily generalized to the risk-averse case. In the first place, for the guide in SaGui (safe exploration policy), we can update the adaptive safety weights based on the approximated CVaR by the undiscounted real costs. In the second place, the internal algorithm of SaGui is SAC-Lagrangian, therefore, risk control can be augmented to SaGui by using the distributional safety critic rather than the expectation safety critic. In general, the combi-

nation just means a different safety requirement in SaGui. But, additional theoretical and empirical proofs of convergence need to be established.

7.3. FINAL THOUGHTS

In this dissertation, we made efforts to enhance the safety mechanism in RL. The proposed formulations and algorithms make RL more applicable in the real world. However, except for the aspects we addressed, there are still many problems to be solved. In this section, we will discuss some other factors that have a significant impact on the successful application of RL.

Uncontrolled data collection. Unlike supervised learning, the data of RL comes from the interactions between the agent and the environment. To achieve data balance without skewed class proportions in a classification data set, supervised learning can resample the training set, cluster the abundant class, etc. However, it is challenging to define the training data balance in RL so as to maintain it, while data collection is executed by the policy. Also, data balance during training may not have a positive effect on the long-term objectives. No matter which RL algorithm we use, it is difficult to have a perfect understanding of the reward signal, with the result that the RL agent is collecting repeated data of little value most of the time. To some extent, a prioritized replay buffer can solve the problem of training priority, but it actually discards the repeated experience data (Schaul et al., 2015). To finally obtain feasible policies in the face of complex tasks, it is still necessary to collect a lot of redundant data. However, in practice, the data collection is not always free, especially when the simulation is not possible. In summary, data balance is a challenge in RL due to the dynamic nature of the interactions between the agent and the environment, and the difficulty of understanding the reward signal.

Environmental restrictions. Many real-world environments start from the initial states, which limits many possible optimization directions. In the state space, we may prioritize certain states or novel states and aim to visit them more frequently. However, the state transition functions (represented by probability functions) of many real-world environments are stochastic (Puterman, 2014). It can be difficult to optimize for certain critical states in RL due to the uncertainty of the environment, even when the previous action sequence is recorded. This becomes even more challenging when using a stochastic policy, which adds an additional layer of uncertainty through the double stochastic superposition. In addition, this limitation makes impossible some test scenarios that are essential for some applications. For instance, automatic driving needs to test a certain bend, but it is hardly possible to repeatedly test the robustness of the policy in this state.

Poor interpretability. To solve complex RL problems, we often rely on deep RL methods, which typically involve the use of value functions to evaluate the quality of a policy. Originally, value function learning is a process to estimate the impact of current actions on the future return through gradual learning (Gaskett, Wettergreen, and Zelinsky, 1999). For deep RL algorithms in large continuous problems, deep neural networks are introduced to approximate the value functions. In this case, it is challenging to verify the convergence of the value function, and the correctness of the policy. Both of these are not problems for tabular Q-learning (Sutton and Barto, 2018). For deep RL algorithms, we can only demonstrate their effectiveness by rendering the policy and checking the learning processes of return and parameters. Accordingly, careful tuning of hyperparameters and reward functions may be necessary to achieve good performance, but this can come at the cost of interpretability. Overall, deep RL methods offer a promising approach for solving complex RL problems, but they can be challenging to verify and interpret due to the use of value functions and deep neural networks.

Difficult encapsulation. In some visual classification frameworks, users do not need to care about model building and parameter setting, but only need to collect their own data sets and load them (Voulodimos et al., 2018). Also, the whole training process is transparent to users. However, this is still impossible in RL, partly because of the reward design (Dewey, 2014; Hadfield-Menell et al., 2017). The original intention of RL research was to let the machine learn by itself without human intervention. But now, there is a lot of human engineering in RL, which goes against the original intention of RL. The most time-consuming work to solve practical problems with RL is not in algorithm selection but in reward design. We need to consider various mechanisms to guide the agent to learn the knowledge we expect it to learn, and make the reward meaningful. The current RL is not intelligent enough to quickly self-evolve without human participation. Therefore, the diversity of scenarios and the complexity of reward design make RL algorithms difficult to be encapsulated. To address this, we can improve encapsulation by using modular design, standard interfaces, abstraction, etc. By taking a structured approach and minimizing dependencies between components, RL algorithms can be easier to understand and use.

To sum up, the research progress of making RL more applicable is slower than the expectation of the public. Even though it is widely believed that real AI can be further realized by combining RL with deep learning, deep RL has not been extensively applied in the real world. In highly complex problems, the cost of trial and error is high, the task objective is not easy to be quantified by the reward function, data collection is demanding, etc. Hence, to apply RL in the real world, we still have a long way to go.

BIBLIOGRAPHY

Abel, David, John Salvatier, Andreas Stuhlmüller, and Owain Evans (2017). *Agent-agnostic human-in-the-loop reinforcement learning*. arXiv:1701.04079.

Achiam, Joshua, David Held, Aviv Tamar, and Pieter Abbeel (2017). "Constrained Policy Optimization". In: *Proceedings of the 34th International Conference on Machine Learning*. PMLR, pp. 22–31.

Ajgl, Jiřı and Miroslav Šimandl (2011). "Particle based probability density fusion with differential Shannon entropy criterion". In: *14th International Conference on Information Fusion*. IEEE, pp. 1–8.

Alshiekh, Mohammed, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu (2018). "Safe Reinforcement Learning via Shielding". In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*. AAAI Press, pp. 2669–2678.

Altman, Eitan (1999). *Constrained Markov decision processes*. Vol. 7. CRC Press.

Badia, Adrià Puigdomènech, Pablo Sprechmann, Alex Vitvitskyi, Daniel Guo, Bilal Piot, Steven Kapturowski, Olivier Tieleman, Martin Arjovsky, Alexander Pritzel, Andrew Bolt, et al. (2019). "Never Give Up: Learning Directed Exploration Strategies". In: *International Conference on Learning Representations*, pp. 1–9.

Bechtle, Sarah, Artem Molchanov, Yevgen Chebotar, Edward Grefenstette, Ludovic Righetti, Gaurav Sukhatme, and Franziska Meier (2021). "Meta learning via learned loss". In: *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE, pp. 4161–4168.

Bellemare, Marc, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos (2016). "Unifying count-based exploration and intrinsic motivation". In: *Advances in Neural Information Processing Systems* 29.

Bellemare, Marc G, Salvatore Candido, Pablo Samuel Castro, Jun Gong, Marlos C Machado, Subhodeep Moitra, Sameera S Ponda, and Ziyu Wang (2020). "Autonomous navigation of stratospheric balloons using reinforcement learning". In: *Nature* 588(7836), pp. 77–82.

Bellemare, Marc G, Will Dabney, and Rémi Munos (2017). "A Distributional Perspective on Reinforcement Learning". In: *Proceedings of the 34th International Conference on Machine Learning*. PMLR, pp. 449–458.

Bellemare, Marc G, Yavar Naddaf, Joel Veness, and Michael Bowling (2013). "The arcade learning environment: An evaluation platform for general agents". In: *Journal of Artificial Intelligence Research* 47, pp. 253–279.

Bengio, Yoshua, Jérôme Louradour, Ronan Collobert, and Jason Weston (2009). "Curriculum Learning". In: *Proceedings of the 26th Annual International Conference on Machine Learning*. ACM, pp. 41–48.

Bertsekas, Dimitri P (1982). *Constrained optimization and Lagrange multiplier methods*. Vol. 1. Academic press.

Bharadhwaj, Homanga, Aviral Kumar, Nicholas Rhinehart, Sergey Levine, Florian Shkurti, and Animesh Garg (2021). "Conservative Safety Critics for Exploration". In: *International Conference on Learning Representations*, pp. 1–9.

Borkar, Vivek S (2005). "An actor-critic algorithm for constrained Markov decision processes". In: *Systems & control letters* 54(3), pp. 207–213.

Boutilier, Craig and Tyler Lu (2016). "Budget Allocation using Weakly Coupled, Constrained Markov Decision Processes". In: *Proceedings of the Thirty-Second Conference on Uncertainty in Artificial Intelligence*, pp. 52–61.

Brockman, Greg, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba (2016). *Openai gym*. *arXiv:1606.01540*.

Burda, Yuri, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A Efros (2019). "Large-Scale Study of Curiosity-Driven Learning". In: *International Conference on Learning Representations*, pp. 1–9.

Burda, Yuri, Harrison Edwards, Amos Storkey, and Oleg Klimov (2019). "Exploration by random network distillation". In: *International Conference on Learning Representations*, pp. 1–9.

Carr, Steven, Nils Jansen, Sebastian Junges, and Ufuk Topcu (2023). "Safe Reinforcement Learning via Shielding under Partial Observability". In: *Thirty-Fifth AAAI Conference on Artificial Intelligence*. AAAI Press.

Chow, Yinlam, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone (2017). "Risk-Constrained Reinforcement Learning with Percentile Risk Criteria ". In: *The Journal of Machine Learning Research* 18(1), pp. 6070–6120.

Cutler, Mark, Thomas J Walsh, and Jonathan P How (2014). "Reinforcement learning with multi-fidelity simulators". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 3888–3895.

Dabney, Will, Georg Ostrovski, David Silver, and Rémi Munos (2018). "Implicit Quantile Networks for Distributional Reinforcement Learning". In: *Proceedings of the 35th International Conference on Machine Learning*, pp. 1096–1105.

Dabney, Will, Mark Rowland, Marc G. Bellemare, and Rémi Munos (2018). "Distributional reinforcement learning with quantile regression". In: *Thirty-Second AAAI Conference on Artificial Intelligence*. AAAI Press, pp. 2892–2901.

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

De Nijs, Frits, Matthijs T. J. Spaan, and Mathijs De Weerdt (2015). "Best-response planning of thermostatically controlled loads under power constraints". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 29.

Dearden, Richard, Nir Friedman, and Stuart Russell (1998). "Bayesian Q-learning". In: *Proceedings of the AAAI-98*, pp. 761–768.

Dewey, Daniel (2014). "Reinforcement learning and the reward engineering principle". In: *2014 AAAI Spring Symposium Series*.

Di Noia, Tommaso, Nava Tintarev, Panagiota Fatourou, and Markus Schedl (2022). "Recommender systems under European AI regulations". In: *Communications of the ACM* 65(4), pp. 69–73.

Dick, Stephanie (2019). "Artificial Intelligence". In: *Harvard Data Science Review* 1(1).

Duan, Jingliang, Yang Guan, Shengbo Eben Li, Yangang Ren, and Bo Cheng (2020). *Distributional soft actor-critic: Off-policy reinforcement learning for addressing value estimation errors*. arXiv:2001.02811.

Dulac-Arnold, Gabriel, Nir Levine, Daniel J Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester (2021). "Challenges of real-world reinforcement learning: definitions, benchmarks and analysis". In: *Machine Learning*, pp. 2419–2468.

Dulac-Arnold, Gabriel, Nir Levine, Daniel J. Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester (2021). "Challenges of real-world reinforcement learning: definitions, benchmarks and analysis". In: *Machine Learning* 110(9), pp. 2419–2468.

Engel, Yaakov, Shie Mannor, and Ron Meir (2005). "Reinforcement learning with Gaussian processes". In: *Proceedings of the 22nd international conference on Machine learning*, pp. 201–208.

Eysenbach, Benjamin, Shixiang Gu, Julian Ibarz, and Sergey Levine (2018). "Leave no Trace: Learning to Reset for Safe and Autonomous Reinforcement Learning". In: *International Conference on Learning Representations*, pp. 1–9.

Finn, Chelsea, Pieter Abbeel, and Sergey Levine (2017). "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks". In: *Proceedings of the 34th International Conference on Machine Learning*. PMLR, pp. 1126–1135.

Fox, Lior, Leshem Choshen, and Yonatan Loewenstein (2018). "DORA The Explorer: Directed Outreaching Reinforcement Action-Selection". In: *International Conference on Learning Representations*, pp. 1–9.

García, Javier and Fernando Fernández (2015). "A Comprehensive Survey on Safe Reinforcement Learning". In: *The Journal of Machine Learning Research* 16(1), pp. 1437–1480.

Gaskett, Chris, David Wettergreen, and Alexander Zelinsky (1999). "Q-learning in continuous state and action spaces". In: *Australasian joint conference on artificial intelligence*. Springer, pp. 417–428.

Ghasemi, Mahsa, Evan Scope Crafts, Bo Zhao, and Ufuk Topcu (2021). "Multiple Plans are Better than One: Diverse Stochastic Planning". In: *Proceedings of the International Conference on Automated Planning and Scheduling*. AAAI Press, pp. 140–148.

Gimelfarb, Michael, André Barreto, Scott Sanner, and Chi-Guhn Lee (2021). "Risk-Aware Transfer in Reinforcement Learning using Successor Features". In: *Advances in Neural Information Processing Systems* 34, pp. 17298–17310.

Gordon, Geoff and Ryan Tibshirani (2012). "Karush-Kuhn-Tucker conditions". In: *Optimization* 10(725/36), p. 725.

Grbic, Djordje and Sebastian Risi (2020). "Safe Reinforcement Learning through Meta-learned Instincts". In: *ALIFE 2020 : The 2020 Conference on Artificial Life*. Artificial Life Conference Proceedings. MIT Press: United States, pp. 183–291.

Guo, Zhaohan Daniel, Mohammad Gheshlaghi Azar, Alaa Saade, Shantanu Thakoor, Bilal Piot, Bernardo Avila Pires, Michal Valko, Thomas Mesnard, Tor Lattimore, and Rémi Munos (2021). *Geometric entropic exploration*. arXiv:2101.02055.

Ha, Sehoon, Peng Xu, Zhenyu Tan, Sergey Levine, and Jie Tan (2020). *Learning to Walk in the Real World with Minimal Human Effort*. arXiv:2002.08550.

Haarnoja, Tuomas, Sehoon Ha, Aurick Zhou, Jie Tan, George Tucker, and Sergey Levine (2019). "Learning to Walk Via Deep Reinforcement Learning." In: *Robotics: Science and Systems*.

Haarnoja, Tuomas, Aurick Zhou, Pieter Abbeel, and Sergey Levine (2018). "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor". In: *Proceedings of the 35th International Conference on Machine Learning*. PMLR, pp. 1861–1870.

Haarnoja, Tuomas, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. (2018). *Soft Actor-Critic Algorithms and Applications*. arXiv:1812.05905.

Hadfield-Menell, Dylan, Smitha Milli, Pieter Abbeel, Stuart J Russell, and Anca Dragan (2017). "Inverse reward design". In: *Advances in Neural Information Processing Systems* 30.

Hans, Alexander, Daniel Schneegaß, Anton Maximilian Schäfer, and Steffen Udluft (2008). "Safe exploration for reinforcement learning." In: *ESANN*. Citeseer, pp. 143–148.

Hazan, Elad, Sham Kakade, Karan Singh, and Abby Van Soest (2019). "Provably Efficient Maximum Entropy Exploration". In: *Proceedings of the 36th International Conference on Machine Learning*. PMLR, pp. 2681–2691.

Hoehl, Stefanie, Kahl Hellmer, Maria Johansson, and Gustaf Gredebäck (2017). "Itsy bitsy spider…: Infants react with increased arousal to spiders and snakes". In: *Frontiers in psychology* 8, p. 1710.

Huber, Peter J (1964). "Robust Estimation of a Location Parameter". In: *The Annals of Mathematical Statistics*, pp. 73–101.

Igl, Maximilian, Gregory Farquhar, Jelena Luketina, Wendelin Boehmer, and Shimon Whiteson (2021). "Transient Non-stationarity and Generalisation in Deep Reinforcement Learning". In: *International Conference on Learning Representations*, pp. 1–9.

Islam, Riashat, Zafarali Ahmed, and Doina Precup (2019). *Marginalized State Distribution Entropy Regularization in Policy Optimization*. *arXiv:1912.05128*.

Jansen, Nils, Bettina Könighofer, Sebastian Junges, Alex Serban, and Roderick Bloem (2020). "Safe Reinforcement Learning Using Probabilistic Shields (Invited Paper)". In: *31st International Conference on Concurrency Theory*. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, pp. 1–16.

Jardine, Andrew KS, Daming Lin, and Dragan Banjevic (2006). "A review on machinery diagnostics and prognostics implementing condition-based maintenance". In: *Mechanical systems and signal processing* 20(7), pp. 1483–1510.

Jin, Chi, Akshay Krishnamurthy, Max Simchowitz, and Tiancheng Yu (2020). "Reward-free exploration for reinforcement learning". In: *International Conference on Machine Learning*. PMLR, pp. 4870–4879.

Kakade, Sham M (2001). "A natural policy gradient". In: *Advances in Neural Information Processing Systems* 14.

Kamran, Danial, Carlos Fernandez Lopez, Martin Lauer, and Christoph Stiller (2020). "Risk-Aware High-level Decisions for Automated Driving at Occluded Intersections with Reinforcement Learning". In: *IEEE Intelligent Vehicles Symposium, IV*. IEEE, pp. 1205–1212.

Kamran, Danial, Thiago D Simão, Qisong Yang, Canmanie T Ponnambalam, Johannes Fischer, Matthijs T. J. Spaan, and Martin Lauer (2022). "A Modern Perspective on Safe Automated Driving for Different Traffic Dynamics using Constrained Reinforcement Learning". In: *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 4017–4023.

Kelly, Michael, Chelsea Sidrane, Katherine Driggs-Campbell, and Mykel J Kochenderfer (2019). "HG-DAgger: Interactive imitation learning with human experts". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 8077–8083.

Keramati, Ramtin, Christoph Dann, Alex Tamkin, and Emma Brunskill (2020). "Being optimistic to be conservative: Quickly learning a cvar policy". In: *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 4436–4443.

Khokhlov, Valentyn (2016). "Conditional value-at-risk for elliptical distributions". In: *Evropskỳ časopis ekonomiky a managementu* 2(6), pp. 70–79.

Kullback, Solomon and Richard A. Leibler (1951). "On information and sufficiency". In: *The Annals of Mathematical Statistics* 22(1), pp. 79–86.

Kumar, Aviral, Aurick Zhou, George Tucker, and Sergey Levine (2020). "Conservative Q-Learning for Offline Reinforcement Learning". In: *Advances in Neural Information Processing Systems* 33, pp. 1179–1191.

Kumar, Saurabh, Aviral Kumar, Sergey Levine, and Chelsea Finn (2020). "One Solution is Not All You Need: Few-Shot Extrapolation via Structured MaxEnt RL". In: *Advances in Neural Information Processing Systems 33*. Curran Associates, Inc., pp. 8198–8210.

Laskin, Michael, Hao Liu, Xue Bin Peng, Denis Yarats, Aravind Rajeswaran, and Pieter Abbeel (2021). "CIC: Contrastive Intrinsic Control for Unsupervised Skill Discovery". In: *Deep Reinforcement Learning Workshop at NeurIPS 2021*.

Laskin, Michael, Denis Yarats, Hao Liu, Kimin Lee, Albert Zhan, Kevin Lu, Catherine Cang, Lerrel Pinto, and Pieter Abbeel (2021). "URLB: Unsupervised Reinforcement Learning Benchmark". In: *Deep Reinforcement Learning Workshop at NeurIPS 2021*.

Lew, Thomas, Apoorva Sharma, James Harrison, Andrew Bylard, and Marco Pavone (2022). "Safe active dynamics learning and control: A sequential exploration–exploitation framework". In: *IEEE Transactions on Robotics* 38(5), pp. 2888–2907.

Li, Lihong, Thomas J Walsh, and Michael L Littman (2006). "Towards a Unified Theory of State Abstraction for MDPs". In: *International Symposium on Artificial Intelligence and Mathematics*. ISAIM, pp. 1–10.

Lillicrap, Timothy, Jonathan Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra (2015). "Continuous control with deep reinforcement learning". In: *International Conference on Learning Representations*, pp. 1–10.

Liu, Hao and Pieter Abbeel (2021). "Aps: Active pretraining with successor features". In: *International Conference on Machine Learning*. PMLR, pp. 6736–6747.

Liu, Hao and Pieter Abbeel (2021). "Behavior from the void: Unsupervised active pre-training". In: *Advances in Neural Information Processing Systems* 34, pp. 18459–18473.

Liu, Yongshuai, Jiaxin Ding, and Xin Liu (2020). "IPO: Interior-point policy optimization under constraints". In: *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 4940–4947.

Luo, Michael, Ashwin Balakrishna, Brijen Thananjeyan, Suraj Nair, Julian Ibarz, Jie Tan, Chelsea Finn, Ion Stoica, and Ken Goldberg (2021). *MESA: Offline Meta-RL for Safe Adaptation and Fault Tolerance*. *arXiv:2112.03575*.

Ma, Xiaoteng, Qiyuan Zhang, Li Xia, Zhengyuan Zhou, Jun Yang, and Qianchuan Zhao (2020). *Distributional Soft Actor Critic for Risk Sensitive Learning*. arXiv:2004.14547.

Marot, Antoine, Benjamin Donnot, Camilo Romero, Balthazar Donon, Marvin Lerousseau, Luca Veyrin-Forrer, and Isabelle Guyon (2020). "Learning to run a power network challenge for training topology controllers". In: *Electric Power Systems Research* 189, p. 106635.

Marzari, Luca, Davide Corsi, Enrico Marchesini, and Alessandro Farinelli (2021). *Curriculum Learning for Safe Mapless Navigation*. arXiv:2112.12490.

Mihatsch, Oliver and Ralph Neuneier (2002). "Risk-Sensitive Reinforcement Learning". In: *Machine Learning* 49(2-3), pp. 267–290.

Mills, John A (1998). *Control: A history of behavioral psychology*. Vol. 14. NYU Press.

Miryoosefi, Sobhan, Kianté Brantley, Hal Daume III, Miro Dudik, and Robert E Schapire (2019). "Reinforcement learning with convex constraints". In: *Advances in Neural Information Processing Systems* 32.

Miryoosefi, Sobhan and Chi Jin (2021). *A Simple Reward-free Approach to Constrained Reinforcement Learning*. arXiv:2107.05216.

Mitchell, Tom M and Tom M Mitchell (1997). *Machine learning*. Vol. 1. 9. McGraw-hill New York.

Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin A Riedmiller, Andreas Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis (2015). "Human-level control through deep reinforcement learning". In: *Nature* 518(7540), pp. 529–533.

Moldovan, Teodor Mihai and Pieter Abbeel (2012). "Safe Exploration in Markov Decision Processes". In: *International Conference on Machine Learning*. PMLR, pp. 1451–1458.

Morimura, Tetsuro, Masashi Sugiyama, Hisashi Kashima, Hirotaka Hachiya, and Toshiyuki Tanaka (2010). "Parametric Return Density Estimation for Reinforcement Learning". In: *Twenty-Sixth Conference on Uncertainty in Artificial Intelligence*. AUAI Press, pp. 368–375.

Mutti, Mirco, Riccardo De Santi, and Marcello Restelli (2021). "The Importance of Non-Markovianity in Maximum State Entropy Exploration". In: *Unsupervised Reinforcement Learning Workshop at ICML 2021*.

Mutti, Mirco, Mattia Mancassola, and Marcello Restelli (2022). "Unsupervised reinforcement learning in multiple environments". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 36, pp. 7850–7858.

Mutti, Mirco, Lorenzo Pratissoli, and Marcello Restelli (2021). "Task-agnostic exploration via policy gradient of a non-parametric state entropy estimate". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 10, pp. 9028–9036.

Mutti, Mirco and Marcello Restelli (2020). "An intrinsically-motivated approach for learning highly exploring and fast mixing policies". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 04, pp. 5232–5239.

Nedergaard, Alexander and Matthew Cook (2022). *k-Means Maximum Entropy Exploration*. *arXiv:2205.15623*.

Ng, Andrew Y, Stuart Russell, et al. (2000). "Algorithms for inverse reinforcement learning." In: *International Conference on Machine Learning*. PMLR.

Olkin, I and F Pukelsheim (1982). "The distance between two random vectors with given dispersion matrices". In: *Linear Algebra and its Applications* 48, pp. 257–263.

Ostrovski, Georg, Marc G Bellemare, Aäron Oord, and Rémi Munos (2017). "Count-based exploration with neural density models". In: *International conference on machine learning*. PMLR, pp. 2721–2730.

Papini, Matteo, Matteo Pirotta, and Marcello Restelli (2019). *Smoothing policies and safe policy gradients*. *arXiv:1905.03231*.

Pathak, Deepak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell (2017). "Curiosity-driven exploration by self-supervised prediction". In: *International conference on machine learning*. PMLR, pp. 2778–2787.

Pathak, Deepak, Dhiraj Gandhi, and Abhinav Gupta (2019). "Self-supervised exploration via disagreement". In: *International conference on machine learning*. PMLR, pp. 5062–5071.

Peng, Zhenghao, Quanyi Li, Chunxiao Liu, and Bolei Zhou (2022). "Safe driving via expert guided policy optimization". In: *Conference on Robot Learning*. PMLR, pp. 1554–1563.

Peters, Jan and J Andrew Bagnell (2010). "Policy Gradient Methods." In: *Scholarpedia* 5(11), p. 3698.

Pham, Tu-Hoa, Giovanni De Magistris, and Ryuki Tachibana (2018). "Optlayer-practical constrained optimization for deep reinforcement learning in the real world". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 6236–6243.

Pirotta, Matteo, Marcello Restelli, Alessio Pecorino, and Daniele Calandriello (2013). "Safe policy iteration". In: *International Conference on Machine Learning*. PMLR, pp. 307–315.

Prakash, Bharat, Mohit Khatwani, Nicholas Waytowich, and Tinoosh Mohsenin (2019). "Improving safety in reinforcement learning using model-based architectures and human intervention". In: *The Thirty-Second International Flairs Conference*.

Puterman, Martin L (2014). *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Qin, Zengyi, Yuxiao Chen, and Chuchu Fan (2021). "Density Constrained Reinforcement Learning". In: *Proceedings of the 38th International Conference on Machine Learning*. PMLR, pp. 8682–8692.

Rakelly, Kate, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen (2019). "Efficient Off-Policy Meta-Reinforcement Learning via Probabilistic Context Variables". In: *Proceedings of the 36th International Conference on Machine Learning*. PMLR, pp. 5331–5340.

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

Rockafellar, R. Tyrrell and Stanislav Uryasev (2000). "Optimization of conditional value-at-risk". In: *Journal of risk* 2(3), pp. 21–41.

Rowland, Mark, Robert Dadashi, Saurabh Kumar, Rémi Munos, Marc G Bellemare, and Will Dabney (2019). "Statistics and samples in distributional reinforcement learning". In: *Proceedings of the 36th International Conference on Machine Learning*. PMLR, pp. 5528–5536.

Roy, Benjamin Van (2002). "Neuro-dynamic programming: Overview and recent trends". In: *Handbook of Markov decision processes*, pp. 431–459.

Saunders, William, Girish Sastry, Andreas Stuhlmüller, and Owain Evans (2018). "Trial without Error: Towards Safe Reinforcement Learning via Human Intervention". In: *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2067–2069.

Savas, Yagiz, Melkior Ornik, Murat Cubuktepe, and Ufuk Topcu (2018). "Entropy Maximization for Constrained Markov Decision Processes". In: *56th Annual Allerton Conference on Communication, Control, and Computing*. IEEE, pp. 911–918.

Schaul, Tom, John Quan, Ioannis Antonoglou, and David Silver (2015). *Prioritized experience replay*. *arXiv:1511.05952*.

Schuitema, Erik, Martijn Wisse, Thijs Ramakers, and Pieter Jonker (2010). "The design of LEO: A 2D bipedal walking robot for online autonomous Reinforcement Learning". In: *International Conference on Intelligent Robots and Systems*. IEEE, pp. 3238–3243.

Schulman, John, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz (2015). "Trust Region Policy Optimization". In: *Proceedings of the 32nd International Conference on Machine Learning*. PMLR, pp. 1889–1897.

Schulman, John, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov (2017). *Proximal Policy Optimization Algorithms*. *arXiv:1707.06347*.

Seo, Younggyo, Lili Chen, Jinwoo Shin, Honglak Lee, Pieter Abbeel, and Kimin Lee (2021). "State entropy maximization with random encoders for efficient exploration". In: *International Conference on Machine Learning*. PMLR, pp. 9443–9454.

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

Simão, Thiago D, Nils Jansen, and Matthijs T J Spaan (2021). "AlwaysSafe: Reinforcement Learning Without Safety Constraint Violations During Training". In: *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*. IFAAMAS, pp. 1226–1235.

Simão, Thiago D and Matthijs TJ Spaan (2019). "Safe policy improvement with baseline bootstrapping in factored environments". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 33. 01, pp. 4967–4974.

Singh, Harshinder, Neeraj Misra, Vladimir Hnizdo, Adam Fedorowicz, and Eugene Demchuk (2003). "Nearest neighbor estimates of entropy". In: *American journal of mathematical and management sciences* 23(3-4), pp. 301–321.

Sobel, Matthew J (1982). "The Variance of Discounted Markov Decision Processes". In: *Journal of Applied Probability* 19(4), pp. 794–802.

Spencer, Jonathan, Sanjiban Choudhury, Matthew Barnes, Matthew Schmittle, Mung Chiang, Peter Ramadge, and Siddhartha Srinivasa (2020). "Learning from interventions: Human-robot interaction as both explicit and implicit feedback". In: *16th Robotics: Science and Systems, RSS 2020*. MIT Press Journals.

Srinivasan, Krishnan, Benjamin Eysenbach, Sehoon Ha, Jie Tan, and Chelsea Finn (2020). *Learning to be Safe: Deep RL with a Safety Critic*. *arXiv:2010.14603*.

Stadie, Bradly C, Sergey Levine, and Pieter Abbeel (2015). *Incentivizing exploration in reinforcement learning with deep predictive models*. *arXiv:1507.00814*.

Subramanian, Medha, Jan Viebahn, Simon H. Tindemans, Benjamin Donnot, and Antoine Marot (2021). "Exploring grid topology reconfiguration using a simple deep reinforcement learning approach". In: *2021 IEEE Madrid PowerTech*. IEEE, pp. 1–6.

Sui, Yanan, Alkis Gotovos, Joel Burdick, and Andreas Krause (2015). "Safe Exploration for Optimization with Gaussian Processes". In: *Proceedings of the 32nd International Conference on Machine Learning*. PMLR, pp. 997–1005.

Sun, Yifan, Yaqi Duan, Hao Gong, and Mengdi Wang (2019). "Learning low-dimensional state embeddings and metastable clusters from time series data". In: *Advances in Neural Information Processing Systems* 32.

Sutton, Richard S and Andrew G Barto (2018). *Reinforcement Learning: An Introduction*. Vol. 2. MIT press.

Sutton, Richard S., A. Rupam Mahmood, and Martha White (2016). "An Emphatic Approach to the Problem of Off-policy Temporal-Difference Learning". In: *The Journal of Machine Learning Research* 17(1), pp. 2603–2631.

Svidchenko, Oleg and Aleksei Shpilman (2021). "Maximum Entropy Model-based Reinforcement Learning". In: *Deep Reinforcement Learning Workshop at NeurIPS 2021*.

Tamar, Aviv, Dotan Di Castro, and Shie Mannor (2016). "Learning the Variance of the Reward-To-Go". In: *The Journal of Machine Learning Research* 17(1), pp. 361–396.

Tang, Haoran, Rein Houthooft, Davis Foote, Adam Stooke, OpenAI Xi Chen, Yan Duan, John Schulman, Filip DeTurck, and Pieter Abbeel (2017). "# exploration: A study of count-based exploration for deep reinforcement learning". In: *Advances in Neural Information Processing Systems* 30.

Tang, Yichuan Charlie, Jian Zhang, and Ruslan Salakhutdinov (2020). "Worst Cases Policy Gradients". In: *3rd Annual Conference on Robot Learning*. PMLR, pp. 1078–1093.

Tao, Ruo Yu, Vincent François-Lavet, and Joelle Pineau (2020). "Novelty search in representational space for sample efficient exploration". In: *Advances in Neural Information Processing Systems* 33, pp. 8114–8126.

Tarbouriech, Jean and Alessandro Lazaric (2019). "Active exploration in markov decision processes". In: *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR, pp. 974–982.

Tarbouriech, Jean, Shubhanshu Shekhar, Matteo Pirotta, Mohammad Ghavamzadeh, and Alessandro Lazaric (2020). "Active model estimation in markov decision processes". In: *Conference on Uncertainty in Artificial Intelligence*. PMLR, pp. 1019–1028.

Tassa, Yuval, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. (2018). *Deepmind control suite*. *arXiv:1801.00690*.

Taylor, Matthew E and Peter Stone (2009). "Transfer Learning for Reinforcement Learning Domains: A Survey". In: *The Journal of Machine Learning Research* 10(56), pp. 1633–1685.

Tessler, Chen, Daniel J. Mankowitz, and Shie Mannor (2019). "Reward Constrained Policy Optimization". In: *International Conference on Learning Representations*, pp. 1–9.

Thananjeyan, Brijen, Ashwin Balakrishna, Suraj Nair, Michael Luo, Krishnan Srinivasan, Minho Hwang, Joseph E Gonzalez, Julian Ibarz, Chelsea Finn, and Ken Goldberg (2021). "Recovery RL: Safe Reinforcement Learning With Learned Recovery Zones". In: *IEEE Robotics and Automation Letters* 6(3), pp. 4915–4922.

Théate, Thibaut, Antoine Wehenkel, Adrien Bolland, Gilles Louppe, and Damien Ernst (2021). *Distributional Reinforcement Learning with Unconstrained Monotonic Neural Networks*. *arXiv:2106.03228*.

Todorov, Emanuel, Tom Erez, and Yuval Tassa (2012). "MuJoCo: A physics engine for model-based control". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033.

Turchetta, Matteo, Felix Berkenkamp, and Andreas Krause (2016). "Safe exploration in finite markov decision processes with gaussian processes". In: *Advances in Neural Information Processing Systems* 29.

Turchetta, Matteo, Andrey Kolobov, Shital Shah, Andreas Krause, and Alekh Agarwal (2020). "Safe Reinforcement Learning via Curriculum Induction". In: *Advances in Neural Information Processing Systems 33*. Curran Associates, Inc., pp. 12151–12162.

Urpí, Núria Armengol, Sebastian Curi, and Andreas Krause (2021). *Risk-averse offline reinforcement learning*.

Vezzani, Giulia, Abhishek Gupta, Lorenzo Natale, and Pieter Abbeel (2019). *Learning latent state representation for speeding up exploration*. arXiv:1905.12621.

Voulodimos, Athanasios, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis (2018). "Deep learning for computer vision: A brief review". In: *Computational intelligence and neuroscience* 2018.

Walraven, Erwin and Matthijs T. J. Spaan (2018). "Column generation algorithms for constrained POMDPs". In: *Journal of Artificial Intelligence Research* 62, pp. 489–533.

Wen, Min and Ufuk Topcu (2018). "Constrained cross-entropy method for safe reinforcement learning". In: *Advances in Neural Information Processing Systems* 31.

Xie, Zhaoming, Patrick Clary, Jeremy Dao, Pedro Morais, Jonathan W. Hurst, and Michiel van de Panne (2019). "Learning Locomotion Skills for Cassie: Iterative Design and Sim-to-Real". In: *3rd Annual Conference on Robot Learning*. PMLR, pp. 317–329.

Yang, Derek, Li Zhao, Zichuan Lin, Tao Qin, Jiang Bian, and Tie-Yan Liu (2019). "Fully Parameterized Quantile Function for Distributional Reinforcement Learning". In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., pp. 6193–6202.

Yang, Qisong, Thiago D Simão, Simon H Tindemans, and Matthijs TJ Spaan (2023). "Safety-constrained reinforcement learning with a distributional safety critic". In: *Machine Learning* 112(3), pp. 859–887.

Yang, Qisong, Thiago D. Simão, Nils Jansen, Simon H. Tindemans, and Matthijs T. J. Spaan (2022). "Training and transferring safe policies in reinforcement learning". In: *Adaptive Learning Agents Workshop at AAMAS 2022*.

Yang, Qisong, Thiago D. Simão, Simon H. Tindemans, and Matthijs T. J. Spaan (2021). "WCSAC: Worst-Case Soft Actor Critic for Safety-Constrained Reinforcement Learning". In: *Thirty-Fifth AAAI Conference on Artificial Intelligence*. AAAI Press, pp. 10639–10646.

Yang, Qisong and Matthijs T. J. Spaan (2023). "CEM: Constrained Entropy Maximization for Task-Agnostic Safe Exploration". In: *Thirty-Fifth AAAI Conference on Artificial Intelligence*. AAAI Press.

Yang, Tsung Yen, Justinian Rosca, Karthik Narasimhan, and Peter J Ramadge (2020). "Projection-Based Constrained Policy Optimization". In: *International Conference on Learning Representations*, pp. 1–9.

Yang, Tsung-Yen, Justinian Rosca, Karthik Narasimhan, and Peter J Ramadge (2021). "Accelerating Safe Reinforcement Learning with Constraint-mismatched Baseline Policies". In: *International Conference on Machine Learning*. PMLR, pp. 11795–11807.

Yarats, Denis, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto (2021). "Reinforcement learning with prototypical representations". In: *International Conference on Machine Learning*. PMLR, pp. 11920–11931.

Zahavy, Tom, Brendan O'Donoghue, André Barreto, Sebastian Flennerhag, Volodymyr Mnih, and Satinder Singh (2021). *Discovering Diverse Nearly Optimal Policies with Successor Features*. Unsupervised Reinforcement Learning Workshop at ICML 2021.

Zanger, Moritz A., Karam Daaboul, and J. Marius Zöllner (2021). "Safe Continuous Control with Constrained Model-Based Policy Optimization". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems IROS*. IEEE, pp. 3512–3519.

Zhang, Chuheng, Yuanying Cai, and Longbo Huang Jian Li (2021). "Exploration by Maximizing Rényi Entropy for Reward-Free RL Framework". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 12, pp. 10859–10867.

Zhang, Jesse, Brian Cheung, Chelsea Finn, Sergey Levine, and Dinesh Jayaraman (2020). "Cautious Adaptation For Reinforcement Learning in Safety-Critical Settings". In: *Proceedings of the 37th International Conference on Machine Learning*. PMLR, pp. 11055–11065.

Zhao, Wenshuai, Jorge Peña Queralta, and Tomi Westerlund (2020). "Sim-to-real transfer in deep reinforcement learning for robotics: a survey". In: *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, pp. 737–744.

Zheng, Liyuan and Lillian Ratliff (2020). "Constrained Upper Confidence Reinforcement Learning". In: *Proceedings of the 2nd Conference on Learning for Dynamics and Control*. PMLR: online, pp. 620–629.

Zheng, Zeyu, Junhyuk Oh, Matteo Hessel, Zhongwen Xu, Manuel Kroiss, Hado Van Hasselt, David Silver, and Satinder Singh (2020). "What can learned intrinsic rewards capture?" In: *International Conference on Machine Learning*. PMLR, pp. 11436–11446.

Zhu, Zhuangdi, Kaixiang Lin, and Jiayu Zhou (2020). *Transfer Learning in Deep Reinforcement Learning: A Survey*. *arXiv:2009.07888*.

List of Publications

Related to this dissertation:

1. Yang Q, Simão TD, Tindemans SH, Spaan MTJ. WCSAC: Worst-Case Soft Actor Critic for Safety-Constrained Reinforcement Learning. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI-21)*. 2021

2. Yang Q, Simão TD, Jansen N, Tindemans SH, Spaan MTJ. Training and Transferring Safe Policies in Reinforcement Learning. In *Proceedings of the Adaptive and Learning Agents Workshop*. 2022

3. Yang Q, Simão TD, Tindemans SH, Spaan MTJ. Refined Risk Management in Safe Reinforcement Learning with a Distributional Safety Critic. In *Safe RL Workshop at IJCAI*. 2022

4. Yang Q, Simão TD, Tindemans SH, Spaan MTJ. Safety-constrained reinforcement learning with a distributional safety critic. *Machine Learning*. 2023

5. Yang Q, Spaan MTJ. CEM: Constrained Entropy Maximization for Task-Agnostic Safe Exploration. In *Proceedings of the 37th AAAI Conference on Artificial Intelligence (AAAI-23)*. 2023

6. Yang Q, Simão TD, Jansen N, Tindemans SH, Spaan MTJ. Reinforcement Learning by Guided Safe Exploration. (under review)

Other publications:

1. Kamran D, Simão TD, Yang Q, Ponnambalam CT, Fischer J, Spaan MTJ, Lauer M. A Modern Perspective on Safe Automated Driving for Different Traffic Dynamics using Constrained Reinforcement Learning. In *Proceedings of the IEEE International Conference on Intelligent Transportation Systems (ITSC-22)*. 2022