

Better Safe Than Sorry

A Systematic Literature Review on Avoiding Negative Side-Effects in Reinforcement Learning

Master's Thesis

for attainment of the academic degree of

Diplom-Ingenieur/in

submitted by

**Fabian Kovac, BSc
51907270**

in the
University Course Data Intelligence at St. Pölten University of Applied Sciences

Supervision
Advisor: Dipl. Ing. Sebastian Eresheim, BSc
Assistance: Dipl. Ing. Tobias Kietreiber

Declaration

Title: Better Safe Than Sorry

Type of thesis: Master's Thesis

Author: Fabian Kovac, BSc

Student number: 51907270

I hereby affirm that

- I have written this thesis independently, have not used any sources or aids other than those indicated, and have not made use of any unauthorized assistance.
- I have not previously submitted this thesis topic to an assessor for evaluation or in any form as an examination paper, either in Austria or abroad.
- this thesis corresponds with the thesis assessed by the assessor.

I hereby declare that

- ☒ I have used a Large Language Model (LLM) to proofread the thesis.
- ☐ I have used a Large Language Model (LLM) to generate portions of the content of the thesis. I affirm that I have cited each generated sentence/paragraph with the original source. The LLM used is indicated by a footnote at the appropriate place.
- ☐ no Large Language Model (LLM) has been used for this work.

September 10th, 2024

Date



Signature

Abstract

Reinforcement learning (RL) has made significant progress in solving complex tasks across various domains, yet ensuring the safety and reliability of RL agents remains a critical challenge. This thesis focuses on avoiding negative side effects (NSEs) in RL, a key aspect of AI safety that seeks to prevent agents from causing unintended harm while pursuing their goals. A systematic literature review was conducted to explore existing methodologies aimed at mitigating NSEs, with a particular emphasis on approaches such as reward regularization, impact regularization, inverse reinforcement learning, and model-based penalty prediction. The review reveals a strong reliance on assumptions such as fully observable states and known dynamics, which limit the applicability of current safety mechanisms in real-world, partially observable environments. The thesis also identifies gaps in the literature, including the need for more robust methods to handle non-Markovian side effects, better integration of real-time human feedback, and the development of standardized safety benchmarks. Addressing these gaps is crucial for advancing the safe deployment of RL agents in real-world scenarios, where complexity and unpredictability pose significant challenges.

Contents

1	Introduction	1
1.1	Research Questions	2
1.2	Related Work	3
2	Methodology	5
2.1	Eligibility Criteria	6
2.2	Search Criteria	6
2.3	Search Strategy	8
3	Background	9
3.1	Markov Decision Process	9
3.1.1	Partially Observable Markov Decision Process	10
3.1.2	Constrained Markov Decision Process	11
3.1.3	Factored Markov Decision Process	11
3.1.4	Lexicographic Markov Decision Process	12
3.2	Value Methods	13
3.2.1	Policy and Value Functions	13
3.2.2	Tabular Methods	16
3.2.3	Approximate Methods	17
3.3	Direct Policy Methods	21
3.3.1	Policy Gradient	21
3.3.2	Actor-Critic	22
3.4	Inverse Reinforcement Learning	24
3.4.1	Maximum Entropy IRL	24
3.4.2	Maximum Causal Entropy IRL	25
4	Systematic Literature Review	26
4.1	Agents without safety mechanisms	27
4.1.1	Tabular Solution Methods	27
4.1.2	Deep Reinforcement Learning	28
4.1.3	Text-based Reinforcement Learning	31
4.2	Reward Regularization	32
4.2.1	Avoiding Negative Side Effects by Considering Others	32
4.2.2	Avoiding Epistemic Side Effects	34
4.2.3	Occupancy-Regularized Policy Optimization	35
4.3	Impact Regularization	37
4.3.1	(Relative) Reachability Measures	38
4.3.2	Attainable Utility Preservation	41
4.3.3	Safety Aware Reinforcement Learning	44
4.3.4	Future Task Approach	45
4.3.5	Potential-based Multiobjective Reinforcement Learning	47

Contents

4.3.6	Challenges using Impact Regularization	49
4.3.7	Theoretical Considerations	51
4.4	Model-based Penalty Prediction	55
4.4.1	Human Feedback	55
4.4.2	Constrained Models	62
4.4.3	Planning with Reduced or Incomplete Models	65
4.4.4	Symbolic Planning	68
4.4.5	Environment Shaping	69
4.4.6	Safety Wrappers	71
4.5	Inverse Reinforcement Learning	72
4.5.1	MDP Solver	72
4.5.2	Implicit Preference Information	74
5	Results	76
5.1	Classification of Agents	76
5.1.1	Agents without safety mechanisms	77
5.1.2	Reward Regularization	80
5.1.3	Impact Regularization	83
5.1.4	Model-based Penalty Prediction	87
5.1.5	Inverse Reinforcement Learning	91
5.2	Gaps in the Literature and Future Directions	93
5.2.1	Handling Partial Observability and Non-Markovian Side-Effects	93
5.2.2	Real-Time Human Feedback and Interaction	93
5.2.3	Lack of Standardized Benchmarks and Real-World Transferability	94
5.2.4	Challenges in Balancing Task Performance and Safety	94
5.2.5	Safety in Multi-Agent Systems and Emergent Side-Effects	94
5.2.6	Long-Term Safety Guarantees	95
5.2.7	Ethical Considerations and Consistent Definitions of Side-Effects	95
6	Discussion	96
6.1	Assumptions and Limitations of Current Safety Mechanisms	96
6.2	Performance-Safety Trade-offs	97
6.3	Challenges in Real-World Applications	97
7	Conclusion	99
7.1	Answering the Research Questions	99
7.2	Limitations and Mitigation Strategies	101
7.3	Future Work	103
	List of Figures	106
	List of Tables	107
	Glossary	108
	References	110

1 Introduction

Over the past few years, Reinforcement Learning (RL) has demonstrated remarkable proficiency across a spectrum of tasks. Reinforcement learning showcased its capability to solve complex problems, from outperforming human teams in Dota2 [1] to defeating the world champion in Go [2], [3], to even achieving superhuman mastery in Atari games, chess, shogi, and Go with a single agent [4]. This concept even was extended to environments that are inherently stochastic, partially observed, or so large and complex that they appear stochastic to a finite agent [5]. Considering this advances in Artificial Intelligence (AI) research, RL has consistently pushed the boundaries of AI achievement.

Despite these impressive strides, the rapid advancement of machine learning and AI has prompted an intensified attention and research interest regarding its societal implications [6], [7]. These challenges revolve around the concept of AI alignment or the AI control problem, which addresses fundamental questions regarding how AI systems should be constructed to ensure their objectives align with human values, thereby mitigating potential harm [8], [9]. This encompasses the challenge of AI capability, aiming to constrain AI systems to prevent them from causing harm or gaining excessive control [10].

Reinforcement Learning, being a key player in AI development, also faces the imperative of ensuring safety in its systems. Scholars have theorized that highly intelligent agents may exhibit tendencies to pursue resources and power to fulfill their goals [10]–[12]. In a thought experiment, Marvin Lee Minsky, a renowned AI researcher and co-founder of the MIT Computer Science and Artificial Intelligence Laboratory⁽¹⁾, even envisioned a scenario where an agent, tasked with proving the Riemann hypothesis, could rationally exploit the entire Earth and its inhabitants as computational resources to achieve its task [13].

Despite ongoing efforts, numerous challenges persist in designing safe AI systems and reinforcement learning is no exception. Amodei et al. delineated concrete problems in AI safety [6], highlighting the necessity for RL agents to address issues related to unintended and harmful behavior arising from design flaws. Raji and Dobbe revisited these safety properties through an additional analysis on the impact on society [7], further emphasizing the need on agents being able to mitigate this problems when deployed in real world scenarios. The concrete problems of AI safety in the field of reinforcement learning can be split up into five main pillars highlighting the need to ensure safety across different areas covering varying aspects of agents [6], [7]:

⁽¹⁾ <https://www.csail.mit.edu/>

1 Introduction

- **Avoiding Negative Side Effects:** Ensuring that agents act without inadvertently causing harm, such as a robot unintentionally damaging property while completing tasks.
- **Avoiding Reward Hacking:** Preventing agents from exploiting reward mechanisms to achieve goals through unintended means, for instance, by manipulating their perception of success.
- **Scalable Oversight:** Facilitating intelligent decision-making in situations where information is limited, requiring agents to discern appropriate actions even with incomplete knowledge.
- **Safe Exploration:** Enabling agents to explore new strategies without risking adverse consequences or without endangering itself or its surroundings.
- **Robustness to Distributional Shift:** Ensuring that agents can adapt and perform reliably even when faced with environmental variations between training and deployment settings.

Addressing these challenges is essential for advancing the capabilities of RL agents while maintaining safety and reliability to ensure robust behavior. This thesis focuses on the first main pillar with **avoiding negative side effects**. While this term is often used interchangeably with the other safety problems, it is important to highlight that this AI safety problem indeed spans its own research area to ensure, agents do not cause harm while completing their tasks.

1.1 Research Questions

The goal of this thesis is to conduct a systematic literature review in the field of avoiding negative side effects in reinforcement learning. We will capture the methodologies and assumptions taken when creating safe agents in past research endeavors, take a systematic look on challenges and limitations associated, and highlight the gaps in the literature. The complete research question can therefore be stated as follows:

“What approaches have been taken in previous studies to ensure that reinforcement learning agents avoid negative side effects while completing their tasks?”

To further emphasize on how this field can be advanced in future work, we additionally define following research question to highlight the gaps in the literature:

“What areas within the domain of avoiding negative side effects in reinforcement learning require further exploration?”

To ensure a systematic and comprehensive approach to the literature review, the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) framework [14] was adopted as a guiding principle. PRISMA provides a structured methodology for conducting systematic reviews, enhancing transparency and reproducibility in the research process.

1.2 Related Work

Reinforcement Learning (RL) systems have made significant strides in recent years, demonstrating remarkable proficiency across a spectrum of tasks while also mitigating undesired behavior to some extent. However, ensuring the safety and reliability of these systems remains paramount, particularly in mitigating negative side effects (NSEs) that may arise during planning and execution. While various approaches have been proposed to address these challenges, evaluating the safety properties of intelligent agents in RL environments is critical for ensuring their responsible deployment.

Leike *et al.* [15] introduced a suite of environments designed to test various safety properties, including safe interruptibility, avoidance of side effects, reward gaming, and robustness to self-modification and adversaries. They evaluated the performance of Advantage Actor-Critic (A2C) [16] and Rainbow [17], two deep reinforcement learning agents, on these environments and found that they were unable to satisfactorily solve them, highlighting the importance of addressing safety concerns in AI systems.

Building on this work, Wainwright *et al.* [18] presented SafeLife, a publicly available reinforcement learning environment specifically designed to test the safety of agents. SafeLife features complex, dynamic levels with opportunities for unsafe behavior, and agents are evaluated based on their ability to maximize rewards while minimizing side effects. Despite achieving high performance in terms of reward maximization, agents trained on SafeLife often exhibit unsafe behavior, underscoring the need for further research into ensuring the safety of AI systems.

Amodei *et al.* [6] identified several concrete problems in AI safety, such as reward misspecification, unsafe exploration, and negative side effects. The authors emphasized the importance of value alignment and safe exploration in preventing agents from causing harm, even when following seemingly correct objectives. Their framework for AI safety has influenced many subsequent research directions aimed at improving the safety and robustness of RL systems. Raji *et al.* [7] revisited these problems by examining real-world case studies and highlighting that socio-technical factors, such as over-reliance on automation and stakeholder engagement, play critical roles in AI safety failures. They argue that a purely technical approach may not suffice and propose a more comprehensive socio-technical framing for understanding AI safety in real-world deployments.

Saisubramanian *et al.* [19] focused specifically on avoiding negative side effects in RL systems that arise from incomplete knowledge of the environment. They present a detailed taxonomy of negative side effects, including factors such as severity, reversibility, and observability, and discuss various strategies to mitigate these effects. Their work highlights the challenges that arise when agents operate in open-world settings where full knowledge of the environment cannot be guaranteed. They further emphasize the importance of model imprecision and propose strategies like model updates and constrained optimization to handle the unexpected outcomes of agent actions in uncertain environments. Turner [20] introduced a formal framework for side effect regularization, presenting it as a delayed specification assistance game. Their framework models the RL agent's objective as a partially observable Markov decision process (POMDP) where the true objective is revealed at a later time step, requiring the agent to balance maximizing a proxy reward with preserving its future ability to achieve a range of plausible objectives. Turner [20] empirically demonstrated the effectiveness of this framework in grid-world environments, showing that it captures many aspects of side effect minimization and serves as a formal method to reason about side effect regularization.

1 Introduction

Finally, Griffin *et al.* [21] explored the use of distance-impact penalties to mitigate negative side effects in RL by penalizing agents based on their ultimate impact on the environment. Griffin *et al.* proposed that the penalty depends solely on the final state of the environment after the agent has completed its task, rewarding agents that reverse or minimize the impact of their actions. This approach proved effective across a variety of grid-world environments, helping agents avoid irreversible changes while still accomplishing their goals. The use of distance-impact penalties offers a scalable and general solution that is compatible with model-free RL algorithms, with demonstrated effectiveness in generating low-impact behavior.

While there is no concrete consensus on how to evaluate the safety properties of agents or what aspects environments should consider, related work has focused on building environment suites for evaluating safe behavior as well as formalizing the problems associated with negative side-effects. However, gaps and areas for further research remain.

2 Methodology

To ensure a systematic and comprehensive approach to the literature review, the Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) framework [14] was adopted as a guiding principle. PRISMA provides a structured methodology for conducting systematic reviews, enhancing transparency and reproducibility in the research process.

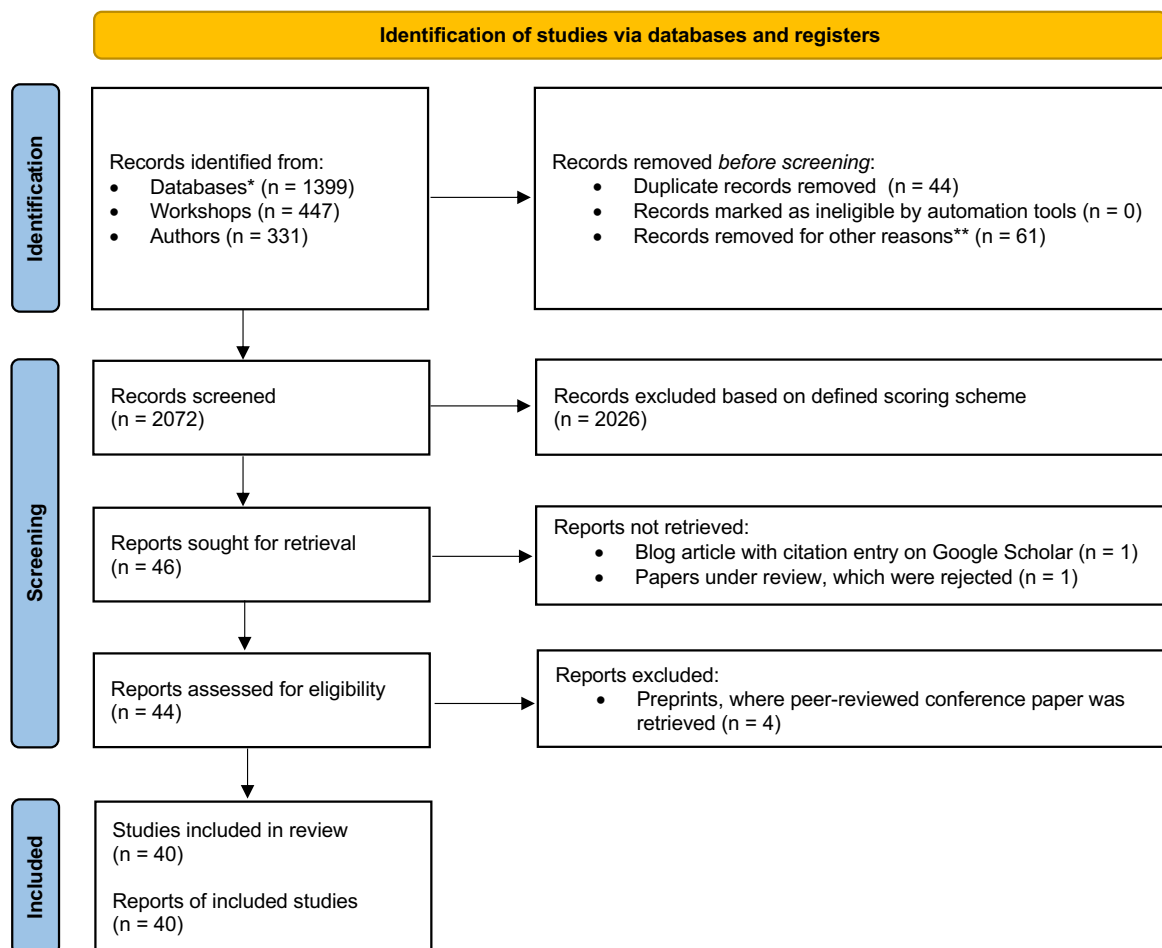


Figure 2.1: PRISMA 2020 flow diagram [14] including the identification of studies via databases and registers.

* Google Scholar returned a server error when exporting more than 1000 entries, resulting in the most 1000 relevant results identified from this database

** ACM Digital Library did not have an export feature, therefore search results known to be discarded in a future step were removed

2.1 Eligibility Criteria

Topic-wise Papers within the field of Safe RL that discuss the avoidance of unintended, negative side-effects due to reward specification in RL. This also includes papers about the agent alignment problem or human preferences (in the context of side-effect avoidance).

Time-wise Papers published between the years 2019 and 2023 were considered for the systematic literature review.

Language-wise Only papers published in English were considered.

Publication-wise Only peer-reviewed papers of conferences, journals and theses were considered. Furthermore, we included ArXiv preprints of authors, that have other peer-reviewed papers in the field or have an affiliation with Google Deepmind or OpenAI.

Scoring Searched records got scored by four reviewers in the field of Safe RL with values ranging from 0 to 2:

- **0:** Record is not relevant for the literature review
- **1:** Record is relevant in form of discussing the higher level topic or applies a topic-wise approach
- **2:** Unique, new approach or updated approach in the topic-wise field

Only records with a combined score of at least 50% (4) or higher get sought for retrieval.

Grouping The search was performed on August 21st, 2023.

2.2 Search Criteria

Search terms The following case-insensitive search term was used for retrieving the records for review:

```
"reinforcement learning" AND  
(  
    "avoiding side effects" OR  
    "negative side effects" OR  
    "undesired side effects" OR  
    "unintended side effects"  
)
```

2 Methodology

Databases Following well-known databases in the field of AI and Machine Learning were used to perform the search:

- Google Scholar⁽¹⁾
- ACM Digital Library⁽²⁾
- Scopus⁽³⁾
- SpringerLink⁽⁴⁾

Workshops In addition, well known workshops of the major conferences (CORE-RANKING A and above) in the topic-wise field were searched:

- SafeAI at AAAI 2019-2023
- AISafety at IJCAI 2019-2023
- Safety Machine Learning at ICLR 2019
- Security and Safety in ML Systems at ICLR 2021
- Safe and Robust Control of Uncertain Systems at NeurIPS 2021
- ML Safety at NeurIPS 2022

Authors To further enhance topic-wise results, papers of well-known authors in the field were searched in addition:

- Alexander Matt Turner
- Anca Dragan
- Dylan Hadfield-Menell
- Rohin Shah
- Sandhya Saisubramian
- Viktoria Krakovna

⁽¹⁾ <https://scholar.google.com/>

⁽²⁾ <https://dl.acm.org/>

⁽³⁾ <https://www.scopus.com/home.uri>

⁽⁴⁾ <https://link.springer.com/>

2.3 Search Strategy

We followed a 3-fold search strategy to retrieve relevant studies:

1. All records listed in defined [Databases](#) using the specified [Search terms](#) were retrieved.
2. The second step includes all records published at relevant [Workshops](#) at a major conference with a CORE-RANKING of A and above that match the specified [Topic-wise](#) eligibility criteria.
3. To finalize the list of records for screening, we included all papers of the most relevant [Authors](#) in the field of Safe RL, again matching the specified [Topic-wise](#) eligibility criteria.

Identification of studies [Figure 2.1](#) shows the PRISMA 2020 flow diagram [14] resulting in 2177 records identified from databases, workshops and authors matching the defined [Eligibility Criteria](#). 105 records (44 duplicates and 61 for other reasons as described in [figure 2.1](#)) were removed before screening, resulting in 46 reports sought for retrieval, that matched our [Scoring](#) criteria with a combined score of at least 50% (score 4) or higher.

Two reports, including a blog article with a matching entry on Google Scholar as well as a rejected paper under review, were not retrieved.

We furthermore excluded 4 preprints, where the matching peer-reviewed conference paper was in the results sought for retrieval, resulting in 40 studies with corresponding reports included in the systematic literature review.

3 Background

The background of this thesis describes Reinforcement Learning (RL), a sub-field of Machine Learning (ML) [16], where an environment and an agent form a cyclic relationship with each other.

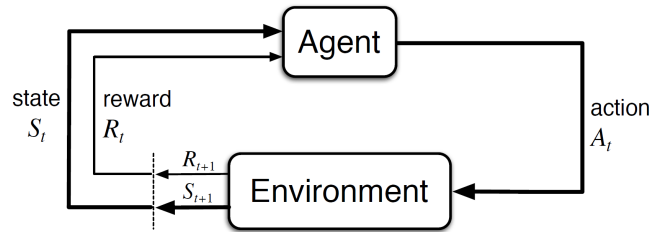


Figure 3.1: Interaction between the Agent and the Environment in Reinforcement Learning [16].

Figure 3.1 shows the cyclic relationship, where the agent observes a state of the environment S_t at time step $t \in \mathbb{N}_0$ and is tasked to take an action A_t based on this state. This action transitions the environment into the next state S_{t+1} and the agent receives a reward $R_{t+1} \subset \mathbb{R}$, a signal how good or bad this action was, before a new cycle begins. Multiple interaction cycles form a trajectory $S_0, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \dots$ and the goal of the agent is to interact with the environment in such a way, that it maximizes its reward.

3.1 Markov Decision Process

A Markov Decision Process (MDP) [22], [23] forms the mathematical setting for modeling decision making. A tuple $M = (S, A, R, p, \chi, \gamma)$ is defined as MDP, if

- S is a set of states denoted state space,
- A is a set of actions denoted action space,
- $R \subset \mathbb{R}$ is the a set of rewards,
- p is a transition probability function or inner dynamics function with

$$p : S \times R \times S \times A \rightarrow [0, 1] \quad (s', r | s, a) \mapsto \mathcal{P}(S_{t+1} = s', R_{t+1} = r | S_t = s, A_t = a),$$

- $\chi : S \rightarrow [0, 1]$ is a initial state distribution,
- $\gamma \in [0, 1]$ is a discount factor, and
- the process adheres to the *Markov Property*

$$\begin{aligned} & \mathcal{P}(S_{t+1} = s', R_{t+1} = r | S_t = s_t, A_t = a_t, \dots, S_0 = s_0, A_0 = a_0) \\ &= \mathcal{P}(S_{t+1} = s', R_{t+1} = r | S_t = s_t, A_t = a_t) . \end{aligned}$$

3 Background

An MDP can be defined deterministic $p : S \times R \times S \times \mathcal{A} \rightarrow \{0, 1\}$ (and is also often denoted with T) or stochastic. Furthermore, the cyclic interaction between agent and environment can run indefinitely or a finite number of steps, where S_T defines the terminal state.

The literature distinguishes between three different definitions of the reward function R , where

1. $R(s_t) : S \mapsto \mathbb{R}$ makes the reward of the agent only dependent of the current state s_t ,
2. $R(s_t, a_t) : S \times A \mapsto \mathbb{R}$ considers the current state s_t when choosing action a_t , and
3. $R(s_t, a_t, s_{t+1}) : S \times A \times S \mapsto \mathbb{R}$ defines the reward, when the agent takes action a_t in state s_t and transitions into the following state s_{t+1} .

Throughout this thesis, we consider

- uppercase variables such as S_t , A_t , and R_t as random variables at time step t ,
- lowercase variables such as s_0 , s' , s_t , s , ... as specific elements of their corresponding space,
- the tuple $M = (S, A, R, T)$ as deterministic and finite Markov Decision Process,
- and $R(s_t, a_t, s_{t+1}) : S \times A \times S \mapsto \mathbb{R}$ as definition of the reward function

unless denoted otherwise.

3.1.1 Partially Observable Markov Decision Process

A Partially Observable Markov Decision Process (POMDP) [24] is an extension of a Markov Decision Process (MDP) where the agent does not have full visibility of the underlying state. Instead, it must make decisions based on a belief or observation about the state, leading to a more complex decision-making problem.

An POMDP is defined by the tuple $M = (S, A, O, R, p, Z, \chi, \gamma, C, d)$, where

- S, A, R, p, χ, γ are the components of an standard MDP,
- O is a set of observations the agent can make about a state, and
- $Z : S \times A \mapsto [0, 1]$ is the observation function, which defines the probability of receiving a particular observation o given the agent is in state s and takes action a .

Since the agent cannot observe the state directly, it maintains a belief state $b(s)$, which is a probability distribution over all possible states $s \in S$. This belief state represents the agent's estimate of being in a particular state, given its past actions and observations. The belief state b_t is updated at each time step based on the previous belief state, the action taken, and the observation received, using Bayes' rule

$$b_{t+1}(s) = \frac{Z(s', a, o) \sum_{s \in S} p(s'|s, a) b_t(s)}{P(o|b_t, a)} \quad (3.1)$$

where $P(o|b_t, a)$ is a normalizing constant ensuring the belief is a valid probability distribution.

In an POMDP, the agent seeks to maximize its expected cumulative reward, based on its belief about the underlying state

$$J(\pi) = \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t R(S_t, A_t) | \pi, b_0 \right] \quad (3.2)$$

where π is the agent's policy, which maps belief states to actions.

3 Background

3.1.2 Constrained Markov Decision Process

A Constrained Markov Decision Process (CMDP) [25] is an extension of a Markov Decision Process that incorporates additional constraints into the decision-making framework. The goal in a CMDP is to optimize an objective function (typically the expected cumulative reward) while also satisfying certain constraints that limit the allowable actions based on other criteria, such as resource usage or risk.

Formally, an CMDP is defined as tuple $M = (S, A, R, p, \chi, \gamma, C, d)$, where

- S, A, R, p, χ, γ are the components of an non-constrained MDP,
- $C : S \times A \mapsto \mathbb{R}^k$ is a set of cost functions that define additional criteria (e.g., safety, energy usage) to be considered during decision making,
- $d \in \mathbb{R}^k$ represents a set of constraints or thresholds on the cumulative costs.

In an CMDP, the agent seeks to maximize the expected cumulative reward

$$J(\pi) = \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t R(S_t, A_t) \right] \quad (3.3)$$

subject to constraints on the expected cumulative cost functions

$$\mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t C_i(S_t, A_t) \right] \leq d_i, \quad \forall i \in \{1, 2, \dots, k\} \quad (3.4)$$

where π denotes the policy of the agent, and i indexes the cost functions.

The difference between a standard MDP and a CMDP is that the CMDP imposes these additional constraints on the agent's behavior, restricting the set of feasible policies to those that satisfy the cost bounds while still attempting to maximize the original reward function. Solving a CMDP generally involves balancing between optimizing the reward and adhering to the constraints.

3.1.3 Factored Markov Decision Process

A Factored Markov Decision Process (FMDP) [26] is a Markov Decision Process, where the state and action spaces are described using a factored representation. This allows the MDP to be more compactly represented and efficiently solved in cases where the state and action spaces are large but can be decomposed into smaller, interacting components.

An FMDP is defined as tuple $M = (X, A, R, p, \chi, \gamma, C, d)$, where

- $X = \{X_1, X_2, \dots, X_n\}$ is a set of state variables. Each state variable X_i takes values from a finite domain, and the full state space S is the Cartesian product of the domains of all state variables, i.e., $S = X_1 \times X_2 \times \dots \times X_n$,
- $A = \{A_1, A_2, \dots, A_n\}$ is a set of action variables, representing actions that can affect the system's state,
- $R : S \times A \mapsto \mathbb{R}$ is the reward function, often represented in a factored form using local reward functions on subsets of state and action variables,

3 Background

- $p : S \times A \times S \rightarrow [0, 1]$ is the transition probability function, which describes the probability of transitioning from one state to another given an action. In an FMDP, this function is factored into smaller, local conditional probability distributions,
- $\chi : S \rightarrow [0, 1]$ is a initial state distribution,
- $\gamma \in [0, 1]$ is a discount factor, as in standard MDPs.

In an FMDP, both the state and transition functions are decomposed. The transition dynamics are represented using Dynamic Bayesian Networks (DBNs) [27], where the probability of each state variable X_i at time step $t + 1$ depends on a subset of the state variables and action variables at time step t . This factored representation drastically reduces the complexity of modeling and computation.

For instance, the transition probability function is factored as

$$p(s'|s, a) = \prod_{i=1}^n p(X'_i | P_a(X_i), a) \quad (3.5)$$

where $P_a(X_i)$ represents the set of parent variables (state and action variables that influence X'_i) in the factored model.

3.1.4 Lexicographic Markov Decision Process

A Lexicographic Markov Decision Process (LMDP) is a variant of the standard MDP that involves optimizing multiple reward functions in a lexicographic order, meaning that the agent prioritizes one objective over others according to a strict ranking or hierarchy. In this framework, the agent seeks to optimize the most important objective first and only considers the next objective if the primary one is maximized (or constrained in some way), and so on.

An LMDP can be defined by the tuple $M = (S, A, \{R^1, R^2, \dots, R^i\}, p, \chi, \gamma)$, where

- S, A, p, χ, γ are the components of an non-constrained MDP, and
- $\{R^1, R^2, \dots, R^i\}$ is a set of reward functions, where $R^i : S \times A \mapsto \mathbb{R}$ represents the reward function for the i -th objective.

In a Lexicographic MDP, the reward functions $R^1, R^2, \dots, R^i$ are ranked according to priority. The agent's objective is to maximize the expected cumulative reward of R^1 first. Once the policy optimally satisfies R^1 , it seeks to maximize R^2 without reducing the optimality of R^1 , and so on for each R^i .

More formally, the agent maximizes the expected cumulative reward for the most important reward function

$$J_1(\pi) = \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t R^1(S_t, A_t) \right] \quad (3.6)$$

subject to the optimal policy for R^1 . It then maximizes the second reward function

$$J_2(\pi) = \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t R^2(S_t, A_t) \right] \quad (3.7)$$

while ensuring the cumulative reward for R^1 remains optimal. This process is repeated for each reward function in the lexicographic order until all objectives are satisfied according to their priority.

3.2 Value Methods

In value-based methods [16], the core idea is to learn value functions that help the agent determine how good it is to be in a certain state or take a certain action. The agent uses these learned values to derive an optimal policy for decision making.

3.2.1 Policy and Value Functions

Let $M = (S, A, R, p, \chi, \gamma)$ be an MDP, then a function

$$\pi : S \times A \mapsto [0, 1] \quad (a|s) \mapsto P(A_t = a | S_t = s) \quad (3.8)$$

is called a policy. A policy $\pi(a|s)$ is a mapping from states to a distribution over actions, specifying the probability of taking action a in state s , while $\pi(\cdot|s)$ gives us the probability over all actions in state s .

The goal of an agent is to learn a policy that maximizes the return (sum cumulative, discounted rewards) G_t starting from time step t

$$G_t := R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots + \gamma^{T-1} R_T = \sum_{k=0}^T \gamma^k R_{t+k+1} . \quad (3.9)$$

The goal in reinforcement learning is often to find an optimal policy denoted π^* that maximizes the return for all states.

The state-value function $V_\pi(s)$ for a policy π gives the expected return starting from state s and following policy π thereafter. It is defined with

$$V_\pi : S \mapsto \mathbb{R}, \quad s \mapsto \mathbb{E}_\pi \left[\sum_{k=0}^T \gamma^k R_{t+k+1} \middle| S_t = s \right] = \mathbb{E}_\pi[G_t | S_t = s] \quad (3.10)$$

and quantifies how good it is to be in a particular state when following a given policy.

The action-value function $Q_\pi(s, a)$ represents the expected return of taking action a in state s and following policy π thereafter. It is defined with

$$Q_\pi : S \times A \mapsto \mathbb{R}, \quad (s, a) \mapsto \mathbb{E}_\pi \left[\sum_{k=0}^T \gamma^k R_{t+k+1} \middle| S_t = s, A_t = a \right] = \mathbb{E}_\pi[G_t | S_t = s, A_t = a] \quad (3.11)$$

and is used in many algorithms to assess the value of actions and choose the optimal one.

The advantage function $A_\pi(s, a)$ measures how much better an action a is compared to the expected actions under policy π . It is defined as the difference between the action-value function and state-value function

$$A_\pi : S \times A \mapsto \mathbb{R}, \quad (s, a) \mapsto Q^\pi(s, a) - V^\pi(s) \quad (3.12)$$

and helps in distinguishing whether a particular action is better than the average action in the given state.

3 Background

Furthermore, there is a relationship between V_π and Q_π , where one can be derived from the other. The state-value function V_π can be derived from Q_π and π

$$V_\pi(s) = \sum_{a \in A} \pi(a|s) Q_\pi(s, a) \quad (3.13)$$

and the action-value function Q_π can be derived from V_π and $p(s', r|s, a)$

$$Q_\pi(s, a) = \sum_{s' \in S} \sum_{r \in R} p(s', r|s, a) [r + V_\pi(s')] . \quad (3.14)$$

The value functions with respect to the optimal policy π^* are called optimal state-value function $V^*(s)$ and optimal action-value function $Q^*(s, a)$.

With respect to an action-value function Q_π , a policy

$$\pi(a|s) := \begin{cases} 1 & \text{if } a = \operatorname{argmax}_{a'} Q_\pi(s, a') \\ 0 & \text{otherwise} \end{cases} \quad (3.15)$$

is defined as greedy policy, and

$$\pi(a|s) := \begin{cases} 1 - \epsilon + \frac{\epsilon}{|A|} & \text{if } a = \operatorname{argmax}_{a'} Q_\pi(s, a') \\ \frac{\epsilon}{|A|} & \text{otherwise} \end{cases} \quad (3.16)$$

is defined as ϵ -greedy policy.

Throughout this thesis, we refer to these policies as ϵ -greedy policies if Q_π is clear from the context.

ϵ -greedy policies help the agent to explore the environment, where it chooses a random action a derived from the ϵ -greedy policy with a probability of $\epsilon > 0$ to gain further insights about unknown states. By exploitation on the other hand, the agent chooses the greedy action to get the most expected reward by exploiting the current action-value estimates. However, being greedy might not actually get the most expected reward and can lead to sub-optimal behavior during the learning process. Therefore it is advisable to switch from an explorative to a greedy policy if the agent's action-value function is approaching the optimal action value function $Q_\pi(s, a) \approx Q^*(s, a)$ to refrain from exploratory moves, when e.g. the agent goes live in a production setting.

To now help the agent to converge to an optimal policy, we use the process of generalized policy iteration [16] as shown in Figure 3.2, that combines two processes iteratively:

1. **Policy Evaluation:** In the policy evaluation phase, the agent estimates the value function for the current policy π . This involves calculating either the state-value function $V_\pi(s)$ or the action-value function $Q_\pi(s, a)$, which represent the expected return for states or state-action pairs, respectively. The goal is to determine how good the current policy is in terms of expected future rewards.
2. **Policy Improvement:** Once the value function is evaluated, the agent uses this information to improve the policy by making it greedy. This leads to an improved policy π' based on the updated value function.

3 Background

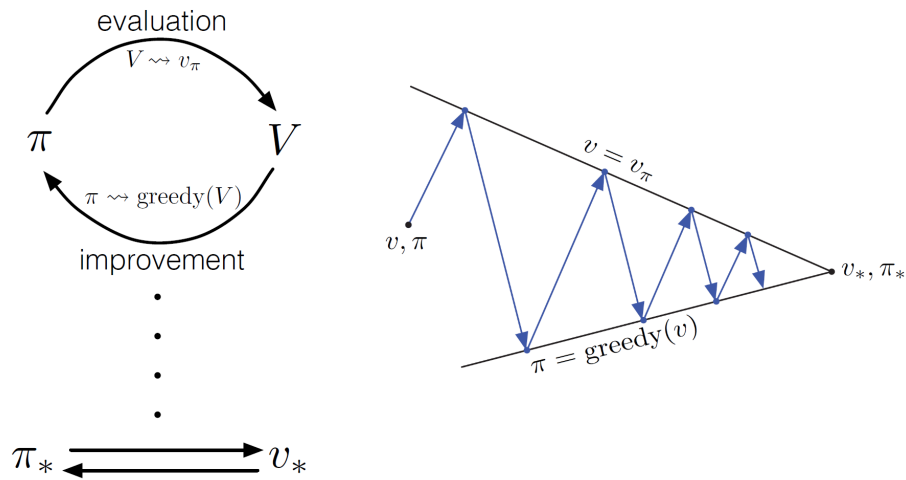


Figure 3.2: Generalized policy iteration for estimating $V(s)^*$ and $\pi(a|s)^*$ [16].

Generalized policy iteration is guaranteed to converge to the optimal policy π^* , because policy evaluation provides a better estimate of the value function for the current policy, and policy improvement ensures that each new policy is at least as good as the previous one (non-decreasing performance). As the process iterates, the policy becomes closer to the optimal policy, and the value function converges to the optimal value function $V^*(s)$ or $Q^*(s, a)$.

3 Background

3.2.2 Tabular Methods

Tabular methods are used when the state-action space is small enough that the agent can store the values for all possible state-action pairs in a table. These methods directly update the estimated value of each state-action pair based on interactions with the environment.

SARSA

SARSA (State-Action-Reward-State-Action) is a Temporal-Difference Learning method [28] with the basic principle of using already learned estimates of state or state-action values in the process of updating other corresponding estimates. SARSA learns the action-value function $Q(s, a)$ by updating the current policy based on the action the agent actually takes, rather than the the best possible action. The update rule is defined as

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha [R_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \quad (3.17)$$

where α is a parameter called step size, that allows to control the influence of the update. This makes SARSA sensitive to the exploration strategy (on-policy) since the next action is determined by the current policy.

Q-learning

Q-learning [29] is an off-policy TD algorithm that updates the action-value function based on the maximum possible future reward, independent of the agent's policy. This means the can learn the optimal policy even while exploring suboptimal actions. The update rule is defined as

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha [R_{t+1} + \gamma \max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t)] \quad (3.18)$$

and uses the greedy action $\max_{a'} Q(s_{t+1}, a')$ to learn the optimal policy.

N-step Bootstrapping

N-step bootstrapping methods [16] extend SARSA and Q-learning by considering multiple steps of rewards and updating the Q -values based on the next n -steps. This introduces a balance between Monte Carlo methods (which use full episodes) and one-step TD methods [28]. The n -step update rule is defined with

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[\sum_{k=0}^{n-1} \gamma^k R_{t+k+1} + \gamma^n Q(s_{t+n}, a_{t+n}) - Q(s_t, a_t) \right] \quad (3.19)$$

and allows the agent to incorporate longer-term planning while maintaining computational feasibility.

3.2.3 Approximate Methods

In environments with large or continuous state-action spaces, tabular methods are impractical due to memory and computation constraints. Approximate methods use function approximations (often neural networks) to estimate value functions and learn policies.

Deep Q-Networks

Deep Q-Networks (DQNs) [30] use a neural network to approximate the action-value function. Instead of storing Q -values in a table, the network generalizes across states to estimate the Q -values. The goal is to find a parameterized Q -function which approximates the optimal action-value function $Q(s, a; \theta_i) \approx Q^*(s, a)$, where θ_i is a vector of the neural network's parameters (weights and biases) at iteration i . However, training neural networks has an i.i.d. assumption on the training data (independent and identically distributed) and while state-action tuples sampled by interacting with the environment can be considered as identically distributed, observations/states from the same episode are very likely to be correlated (and therefore not independent). Furthermore, the network might circulate, which moves it in the exact opposite direction as the previous step.

To counteract dependent state-action pairs from the same episode, Mnih *et al.* [30] use state-action pairs from different episodes by letting the agent play a number of episodes and storing all state transitions (state, action, next state, reward) in a replay buffer. For the learning process, we then can simply pick random elements from the buffer for a single batch, to make the training data independent (enough).

To keep the network from circulating, the authors use two networks, one for acting and one for predicting. Therefore the neural network is copied and the acting network learns continuously by interacting with the environment, while the prediction network is freezed and updated after x (hyperparameter) updates.

The squared TD-error is used to calculate the loss to update the network's parameters

$$L_i(\theta_i) = \mathbb{E}_{s,a \sim p(\cdot)} \left[(y_i - Q(s, a; \theta_i))^2 \right] \quad (3.20)$$

where

$$y_i = \mathbb{E}_{s' \sim \varepsilon} \left[r + \gamma \max_{a'} Q(s', a'; \theta_{i-1}) \right] \quad (3.21)$$

is the TD target of Q -learning.

$$\nabla L_i(\theta_i) = \mathbb{E}_{s,a \sim p(\cdot); s' \sim \varepsilon} \left[\left(+ \gamma \max_{a'} Q(s', a'; \theta_{i-1}) - Q(s, a; \theta_i) \right) \nabla_{\theta_i} Q(s, a; \theta_i) \right] \quad (3.22)$$

calculates the gradient and the parameters are updated with

$$\theta_{i+1} = \theta_i - \alpha \left(r + \gamma \max_{a'} Q(s', a'; \theta_{i-1}) - Q(s, a; \theta_i) \right) \nabla_{\theta_i} Q(s, a; \theta_i) \quad (3.23)$$

3 Background

Double DQNs

Double DQNs [31] mitigate the overestimation bias in DQNs (and standard Q -learning) by substituting the max-operator of TD target $\max_{a'} Q(s, a'; \theta)$ with the argmax-operator, which yields the same result

$$\max_{a'} Q(s, a'; \theta) \equiv Q(s, \operatorname{argmax}_{a'} Q(s, a'; \theta); \theta) . \quad (3.24)$$

This however, introduces a selection of overestimated actions, because the outer Q -function which evaluates the best action, is the same Q -function which selects the best action $Q(s, a'; \theta) = Q(s, a'; \theta)$. To remedy this problem, the authors split up the evaluation and the selection in two separate functions (networks) $Q(s, a'; \theta') \neq Q(s, a'; \theta)$ and update the TD target with

$$y_i = \mathbb{E}_{s' \sim \varepsilon} \left[r + \gamma Q(s, \operatorname{argmax}_{a'} Q(s, a'; \theta); \theta') \right] \quad (3.25)$$

where one network θ evaluates the best action, and the other network θ' selects the best action.

Prioritized Replay Buffer DQNs

In standard DQN [30], experience replay samples past transitions uniformly. Prioritized Replay Buffer DQNs [32] improves this by sampling transitions with higher temporal difference (TD) error more frequently. This allows to weight the experiences by "how surprising" they were when they were first learned and replay "surprising" experiences more often. The probability of sampling a transition from the Prioritized Replay Buffer is proportional to its TD error

$$P(i) \propto |\delta_i|^\omega \quad (3.26)$$

where δ_i is the TD error and ω controls the prioritization. However, estimating the expected value with stochastic updates relies on the updates correlating to the same distribution as its expectation. Prioritizing these experiences differently introduces a bias, which needs to be corrected. Therefore, ω allows to correct the bias via importance sampling weights by a parameter β with

$$\omega_i = \left(\frac{1}{N} \cdot \frac{1}{P(i)} \right)^\beta \cdot \frac{1}{\max_i \omega_i} \quad (3.27)$$

where $\beta = 1$ fully compensates for the bias. Schaul *et al.* [32] note that small bias at the beginning can be advantageous and they recommend starting with an initial $\beta < 1$ and let go $\beta \rightarrow 1$ over time.

3 Background

Noisy DQNs

Noisy networks [33] introduce learnable noise to the Q -networks weights to encourage exploration in a more controlled manner. Instead of using explorative strategies like ϵ -greedy policies, noisy networks sample noise for the network weights to drive exploration.

The linear part of a standard neural network layer with p inputs and q outputs is defined with

$$y = wx + b \quad (3.28)$$

where w are the weights and b are the biases with

$$\begin{aligned} w &\in \mathbb{R}^{q \times p} & x &\in \mathbb{R}^p \\ b &\in \mathbb{R}^q & y &\in \mathbb{R}^q. \end{aligned} \quad (3.29)$$

The linear part of a noisy network with p inputs and q outputs is defined with

$$y := (\mu^w + \sigma^w \odot \varepsilon^w)x + (\mu^b + \sigma^b \odot \varepsilon^b) \quad (3.30)$$

where μ and σ

$$\begin{aligned} \sigma^w &\in \mathbb{R}^{q \times p} & \mu^w &\in \mathbb{R}^q \\ \sigma^b &\in \mathbb{R}^{q \times p} & \mu^b &\in \mathbb{R}^q \end{aligned} \quad (3.31)$$

are learnable parameters, and ε

$$\begin{aligned} \varepsilon^w &\in \mathbb{R}^{q \times p} \\ \varepsilon^b &\in \mathbb{R}^q \end{aligned} \quad (3.32)$$

is random gaussian noise.

Dueling DQNs

Dueling DQNs [34] splits the network into two streams, where one estimates the state-value function $V(s)$ and the other estimates the advantage function $A(s, a)$. The idea behind the dueling architecture is to improve the efficiency and stability of Q -learning by decoupling the estimation of the state-value function and the advantage function. In traditional Q -learning, a single network is used to estimate the Q -value for each state-action pair. However, this can be inefficient in some cases, especially when the agent is in states where the choice of action doesn't significantly impact the outcome.

To get the action-value $Q(s, a)$ again, the two streams are combined with

$$Q(s, a) = V(s) + \left(A(s, a) - \frac{1}{|A|} \sum_{a'} A(s, a') \right) \quad (3.33)$$

where $\frac{1}{|A|} \sum_{a'} A(s, a')$ is a normalization term to ensure that the advantage function is zero-mean, which avoids the problem of the Q -value being overly influenced by the advantage function.

3 Background

N-step DQNs

Similar to n-step bootstrapping [16] as defined in subsection 3.2.2, n-step DQNs incorporate multi-step return to update the Q -value function. This can be seen as a simple extension to vanilla DQNs [30], where the TD target as defined in Equation 3.21 is updated using the cumulative rewards from n steps before bootstrapping from the state at step $t + k$, allowing the model to capture longer-term dependencies with

$$y_i = \begin{cases} \mathbb{E}_{s' \sim \varepsilon} \left[\sum_{k=0}^{n-1} \gamma^k r_{t+k} \right] & \text{for terminal } s_{k+n} \\ \mathbb{E}_{s' \sim \varepsilon} \left[\sum_{k=0}^{n-1} \gamma^k r_{t+k} + \gamma^n \max_{a'} Q(s_{k+1}, a'; \theta_t) \right] & \text{otherwise .} \end{cases} \quad (3.34)$$

Distributional DQNs

Distributional DQNs [35] are an extensions of traditional DQNs [30] that aims to model the full distribution of possible future rewards, rather than just the expected value. The idea is that for each state-action pair (s, a) there is not just a single value (expected return), but a range of possible outcomes (distribution of return), each with different probabilities.

The objective is to learn the distribution of future rewards, denoted as $Z(s, a)$, where

$$Z(s, a) = \sum_{t=0}^{\infty} \gamma^t R(S_t, A_t) \quad (3.35)$$

is the random variable representing the return from state s after taking action a , considering both immediate and future rewards.

The key difference to vanilla DQNs is that instead of approximating the expectation $\mathbb{E}[Z(s, a)] = Q(s, a)$, we approximate the entire distribution of $Z(s, a)$ and The TD target is modified to be distributional. Given a transition (s, a, r, s') , the next state's distribution $Z(s', a')$ is used compute the new distribution $Z(s, a)$. The target distribution is formed by

$$Z'(s, a) = r + \gamma Z(s', \arg\max_{a'} \mathbb{E}[Z(s', a')]) \quad (3.36)$$

where $Z'(s, a)$ is the target distribution for state s and action a , and $Z(s', \arg\max_{a'} \mathbb{E}[Z(s', a')])$ is the distribution of future returns from the best action in the next state s' .

The network is trained to minimize the Wasserstein distance (or similar distance metrics) between the current distribution $Z(s, a)$ and the target distribution $Z'(s, a)$.

Rainbow DQN

Rainbow DQN [17] is a combination of several improvements to the DQN framework [30], including Double DQNs [31], Prioritized Replay Buffers [32], Noisy DQNs [33], Dueling DQNs [34], N-step DQNs [16], and Distributional DQNs [35].

Rainbow integrates these methods into a single powerful and efficient reinforcement learning algorithm for large-scale problems.

3.3 Direct Policy Methods

In Direct Policy Methods, the goal is to directly optimize the policy π without explicitly estimating the value function. These methods optimize the policy by maximizing the expected cumulative reward using gradient-based optimization techniques. Unlike value-based methods, which require calculating action-value functions, direct policy methods work directly on the policy space.

3.3.1 Policy Gradient

Policy Gradient (PG) [16] methods aim to optimize the policy $\pi(a|s; \theta)$ by adjusting the policy parameters θ in the direction that increases the expected reward. The idea is to directly compute the gradient of the expected cumulative reward with respect to the policy parameters and updates the policy accordingly.

The objective is to maximize the expected return

$$J(\theta) = \mathbb{E}_{\pi} \left[\sum_{t=0}^{\infty} \gamma^t R(S_t, A_t) \right] \quad (3.37)$$

The policy gradient theorem provides the gradient of this objective

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\pi} \left[Q^{\pi}(s, a) \nabla_{\theta} \ln \pi(a|s; \theta) \right] \quad (3.38)$$

and this gradient is used to update the policy parameters using gradient ascent

$$\theta \leftarrow \theta + \alpha \nabla_{\theta} J(\theta) \quad (3.39)$$

where α is the learning rate

REINFORCE

REINFORCE [16] is classic policy gradient algorithm, where the gradient is computed based on the total reward obtained from entire episodes (also known as Monte Carlo policy gradient). The gradient is calculated similar to the policy gradient theorem as describe in [Equation 3.38](#)

$$\begin{aligned} \nabla_{\theta} J(\theta) &= \mathbb{E}_{\pi} \left[Q^{\pi}(s, a) \nabla_{\theta} \ln \pi(a|s; \theta) \right] \\ &= \mathbb{E}_{\pi} \left[G_t \nabla_{\theta} \ln \pi(a|s; \theta) \right] \end{aligned} \quad (3.40)$$

and replaces the action-value function $Q^{\pi}(s, a)$ with the return G_t sampled from the entire episode.

REINFORCE with baseline A major drawback of the basic REINFORCE algorithm is its high variance [16]. To reduce this variance, REINFORCE with a baseline introduces a baseline function that subtracts from the return. This baseline does not introduce bias, as long as it is independent of the action, and helps stabilize learning by reducing the variance in the policy gradient estimate.

The gradient of REINFORCE with baseline is defined with

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\pi} \left[(G_t - b(s)) \nabla_{\theta} \ln \pi(a|s; \theta) \right] \quad (3.41)$$

where $b(s)$ is the baseline function. An often used baseline is the state-value function $V^{\pi}(s)$.

3 Background

3.3.2 Actor-Critic

Actor-Critic (AC) methods combine both value-based and policy-based approaches. The actor refers to the policy that is learned to select actions, while the critic refers to the value function that evaluates how good the current policy is. The Actor-Critic framework consists of two components:

1. **Actor:** The policy $\pi(a|s; \theta_{actor})$ selects actions, and
2. **Critic:** The value function $V(s; \theta_{critic})$ estimates the expected return of being in state s .

The idea is to replace the full return used in REINFORCE with the 1-step TD error. The update for the actor is based on the gradient of the expected return $J(\theta_{actor})$ and is defined with

$$\nabla \theta_{actor} = \alpha_{actor} \nabla_{\theta} \ln \pi(a_t|s_t; \theta_{actor}) \delta_t \quad (3.42)$$

where δ_t is the TD error. This 1-step TD error is computed by the critic and serves as a signal for both updating the value function and guiding the actor's policy update. The TD error is calculated with

$$\delta_t = r_{t+1} + \gamma V(s_{t+1}; \theta_{critic}) - V(s_t; \theta_{critic}) \quad (3.43)$$

and represents the difference between the estimated value of the current state and the value of the next state, plus the immediate reward.

Advantage Actor-Critic

Advantage Actor-Critic (A2C) [16] is a synchronous variant of the actor-critic method. In A2C, multiple agents interact with their own copy of the environment in parallel, and the gradients from all agents are accumulated and applied synchronously to update the policy and value network.

The actor updates the policy parameters using the policy gradient, and the critic helps reduce the variance of the gradient by providing a better estimate of the value of the state. The update to the actor's policy is based on the advantage

$$\nabla_{\theta} J(\theta) \approx \nabla_{\theta} \ln \pi(a|s; \theta) A^{\pi}(s, a) \quad (3.44)$$

where $A^{\pi}(s, a)$ is the advantage function provided by the critic.

Asynchronous Advantage Actor-Critic

Asynchronous Advantage Actor-Critic (A3C) [36] is similar to A2C [16] but works asynchronously. Multiple agents interact with different environments in parallel, and each agent updates the global policy asynchronously without waiting for other agents. This allows the agent to explore different parts of the state space simultaneously and leads to faster convergence.

The asynchronous updates also reduce the correlation between experiences and stabilize training.

3 Background

Trust Region Policy Optimization

Trust Region Policy Optimization (TRPO) [37] is a policy optimization algorithm designed to ensure more stable updates to the policy by restricting how much the policy can change during each update. It prevents overly large updates that can cause the policy to degrade.

TRPO solves the following optimization problem

$$\begin{aligned} \max_{\theta} \quad & \left[\sum \gamma^t R(s_t, a_t) \right] \\ \text{s.t.} \quad & D_{KL}(\pi_{\theta_{old}} || \pi_{\theta}) \leq \delta \end{aligned} \quad (3.45)$$

where D_{KL} is the Kullback-Leibler divergence between the old policy and the new policy, and δ is the threshold that limits the magnitude of the update. TRPO guarantees large improvements in performance while keeping the policy updates within a trust region to avoid instability.

Proximal Policy Optimization

Proximal Policy Optimization (PPO) [38] is a simpler and more practical variant of TRPO. Instead of using a complex optimization with constraints, PPO uses a clipped objective function to limit the policy updates to a safe region.

The PPO objective is defined with

$$L^{PPO}(\theta) = \mathbb{E} \left[\min(r_t(\theta)A_t, \text{clip}(r_t(\theta), 1 - \varepsilon, 1 + \varepsilon)A_t) \right] \quad (3.46)$$

where

$$r_t(\theta) = \frac{\pi_{\theta}(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \quad (3.47)$$

is the probability ration, and ε is a small parameter controlling the range of allowed policy changes. PPO uses this clipping to prevent large deviations from the current policy, ensuring stable updates and avoiding the need for complex constraints.

Soft Actor-Critic

Soft Actor-Critic (SAC) [39] is an off-policy actor-critic algorithm that incorporates an entropy term into the objective function to encourage exploration. The goal is to maximize both the expected reward and the policy's entropy, which leads to more robust exploration.

The objective function for SAC is defined with

$$J(\pi) = \sum_t \mathbb{E}_{(s_t, a_t) \sim \pi} \left[r(s_t, a_t) + \alpha \mathcal{H}(\pi(\cdot|s_t)) \right] \quad (3.48)$$

where $\mathcal{H}(\pi)$ is the entropy of the policy, and α is a temperature parameter that balances the trade-off between reward maximization and exploration. SAC is highly efficient in continuous action spaces and is one of the most effective methods for stable learning with off-policy data.

3.4 Inverse Reinforcement Learning

Inverse Reinforcement Learning (IRL) [40] is the process of inferring the reward function $R(s, a)$ that an expert is implicitly optimizing, based on observing the expert's behavior. Unlike traditional reinforcement learning where the reward function is given, in IRL the agent must learn the reward function from demonstrations and then use this reward to derive the policy.

In IRL, the agent observes a set of expert trajectories $\tau = (s_0, a_0, s_1, a_1, s_2, a_2, \dots)$ where each trajectory consists of states and actions taken by the expert. The goal is to find a reward function $R(s, a)$ that make the expert policy π^* appear optimal. Formally, the objective of IRL is to maximize the likelihood of the expert trajectories under the inferred reward function.

3.4.1 Maximum Entropy IRL

A popular approach to IRL is Maximum Entropy IRL (MEIRL) [41], which assumes that the expert's behavior is stochastic and that the expert is maximizing expected reward while maintaining high entropy (i.e., the policy is probabilistic and not overly deterministic). This approach uses the principle of maximum entropy [42] to deal with the uncertainty of the expert's actions, encouraging the learned policy to remain stochastic and explore different actions.

In MEIRL, the goal is to find a reward function R such that the expert's behavior is more likely compared to other possible behaviors. The probability of a trajectory τ under a policy π is modeled as

$$P(\tau|R) = \frac{1}{Z(R)} \exp \left(\sum_{t=0}^{T-1} R(s_t, a_t) \right) \quad (3.49)$$

where $\sum_{t=0}^{T-1} R(s_t, a_t)$ is the total reward for trajectory τ , and

$$Z(R) = \sum_{r'} \exp \left(\sum_{t=0}^{T-1} R(s'_t, a'_t) \right) \quad (3.50)$$

is the partition function (normalizing constant) that ensures the probability distribution sums to 1. This encodes the principle that the expert's trajectories are exponentially more likely under the inferred reward function.

The MEIRL objective is to maximize the log-likelihood of the observed expert trajectories

$$\max_R \sum_{\tau_{expert}} \log P(\tau_{expert}|R). \quad (3.51)$$

This results in a reward function that maximizes the probability of the expert's demonstrated trajectories while maintaining maximum entropy over the policy's actions.

3.4.2 Maximum Causal Entropy IRL

Maximum Causal Entropy IRL (MCEIRL) [43] is a variant of MEIRL [41] that generalizes the idea to sequential decision-making problems. Here, the objective is to maximize the causal entropy of the policy, taking into account the future states and actions. The goal is to find the reward function R that maximizes the causal entropy of the policy π , ensuring stochasticity in the expert's future decisions as well.

The policy $\pi(s|a)$ is modeled as

$$\pi(a|s) \propto \exp\left(Q^\pi(s, a)\right) \quad (3.52)$$

where $Q^\pi(s, a)$ is the action-value function under policy π . The reward function $R(s, a)$ is inferred to maximize likelihood, ensuring that the observed expert actions are consistent with a high-entropy optimal policy.

4 Systematic Literature Review

Based on the PRISMA principle [14], we identified following studies for the systematic literature review:

Study	Year	Approach
Amodei <i>et al.</i> [6]	2016	Introduction
Leike <i>et al.</i> [15]	2017	Deep Reinforcement Learning, Environment
Huang <i>et al.</i> [44]	2018	Inverse Reinforcement Learning, MDP Solver
Zhang <i>et al.</i> [45]	2018	Human Feedback, Model-based Penalty Prediction
Frye <i>et al.</i> [46]	2019	Parenting, Human Feedback
Krakovna <i>et al.</i> [47]	2019	Impact Regularization, Reward Regularization
Saisubramanian <i>et al.</i> [48]	2019	Planning with Reduced Models, Model Portfolio
Shah <i>et al.</i> [49]	2019	Implicit Preference Information, Maximum Causal Entropy IRL
Krakovna <i>et al.</i> [50]	2020	Impact Regularization, Reward Regularization
Miret <i>et al.</i> [51]	2020	Impact Regularization, Reward Regularization
Saisubramanian <i>et al.</i> [52]	2020	Human Feedback, Environment Shaping, Model-based Penalty Prediction
Saisubramanian <i>et al.</i> [53]	2020	Lexicographic MDP, Human Feedback, Model-based Penalty Prediction
Shah [54]	2020	Implicit Preference Information, Maximum Causal Entropy IRL, Human Feedback
Turner <i>et al.</i> [55]	2020	Impact Regularization, Reward Regularization
Turner <i>et al.</i> [56]	2020	Impact Regularization, Reward Regularization
Wainwright <i>et al.</i> [18]	2020	Deep Reinforcement Learning, Environment
Waisi [57]	2020	Tabular Solution Methods
Alizadeh Alamdari <i>et al.</i> [58]	2021	Reward Regularization
Burden <i>et al.</i> [59]	2021	Deep Reinforcement Learning
Klassen <i>et al.</i> [60]	2021	Symbolic Planning
Lindner <i>et al.</i> [61]	2021	Impact Regularization, Reward Regularization
Liu <i>et al.</i> [62]	2021	Model-based Penalty Prediction
Saisubramanian <i>et al.</i> [63]	2021	Human Feedback, Environment Shaping, Model-based Penalty Prediction
Vamplew <i>et al.</i> [64]	2021	Impact Regularization, Reward Regularization
Alizadeh Alamdari <i>et al.</i> [65]	2022	Reward Regularization
Burden <i>et al.</i> [66]	2022	Confidence Measures, Model-based Success Prediction
Griffin <i>et al.</i> [21]	2022	Impact Regularization, Reward Regularization, Theory
Hoang <i>et al.</i> [67]	2022	Text-based Reinforcement Learning, Environment
Klassen <i>et al.</i> [68]	2022	Symbolic Planning
Klassen <i>et al.</i> [69]	2022	Reward Regularization
Ma <i>et al.</i> [70]	2022	Environment Feedback, Model-based Penalty Prediction
Saisubramanian <i>et al.</i> [19]	2022	Human Feedback, Environment Shaping, Model-based Penalty Prediction, Survey
Saisubramanian <i>et al.</i> [71]	2022	Human Feedback, Environment Shaping, Model-based Penalty Prediction
Turner [20]	2022	Impact Regularization, Reward Regularization, Theory
Turner <i>et al.</i> [72]	2022	Impact Regularization, Reward Regularization, Theory
Eresheim <i>et al.</i> [73]	2023	Impact Regularization, Reward Regularization
Klassen <i>et al.</i> [74]	2023	Reward Regularization
Laidlaw <i>et al.</i> [75]	2023	Reward Regularization
Low <i>et al.</i> [76]	2023	Model-based Penalty Prediction
Srivastava <i>et al.</i> [77]	2023	Model-based Penalty Prediction

Table 4.1: Studies covered in the systematic literature review ordered by year in ascending order.

4.1 Agents without safety mechanisms

In the field of reinforcement learning (RL), agents typically aim to maximize a reward signal through interaction with their environment. However, when agents operate without explicit safety mechanisms, they can exhibit behaviors that lead to unintended or harmful consequences. These negative side effects can occur because the agents optimize for rewards without regard for safety constraints, making their actions unpredictable or dangerous in real-world scenarios. The absence of safety mechanisms is particularly concerning in complex environments, where side effects may not be immediately observable or penalized.

This section explores tabular solution and deep reinforcement learning approaches where agents are trained without built-in safety mechanisms. It examines the challenges of ensuring safe behavior in these contexts, as well as the potential trade-offs between performance and safety. We review key studies that highlight how agents struggle to avoid side effects and how different learning methods perform in safety-critical environments.

4.1.1 Tabular Solution Methods

Waisi [57] aims to address the challenge of avoiding negative side-effects without explicitly embedding ad-hoc constraints into the agents' objective functions, which could limit the generality of the solution. The study compares the performance agents in two different environments, one with a safety setting and another with a standard setting. The safety setting involves the agent being unaware of penalties for causing side effects, while in the standard setting, the agent is directly penalized for such actions. The environments used for the experiments are a variant of the Sokoban game [15] and the Cliff-walking environment [16]. The agents are trained using Q-learning [29] and Sarsa [28].

The results indicate that agents trained in the standard setting, where they are aware of penalties for causing side-effects, perform better and learn more optimal behavior [57]. In contrast, agents in the safety setting fail to adequately avoid side effects, highlighting the difficulty of achieving safe behavior without direct penalties for harmful actions. The study shows that integrating penalties into the agent's objective function improves safety but may reduce the agent's performance in other aspects.

4.1.2 Deep Reinforcement Learning

Leike *et al.* [15] propose a set of environments to evaluate how well agents adhere to safety constraints during their operation. These environments, referred to as *AI Safety Gridworlds*, are designed to illustrate various safety challenges that intelligent agents might encounter, such as safe interruptibility, avoiding side effects, safe exploration, and robustness to distributional shifts and adversaries.

Each gridworld is a two-dimensional grid where the agent occupies a single cell and can move in four directions (up, down, left, right). The environments are episodic, meaning that each episode begins with the environment in a specific state and ends when the agent reaches a goal or after a fixed number of steps. Furthermore each environment includes a performance function (hidden from the agent) beside the reward function (observed by the agent) to measure compliance with safety constraints.

The authors conducted experiments with advantage actor critic (A2C) [36] and Rainbow [17], which combines six extensions of DQNs [30] into a single agent (double Q-learning [31], prioritized replay [32], dueling networks [34], multi-step learning [78], distributional RL [35], and noisy nets [33]). Both agent do not have any safety mechanisms and struggle to solve the specification problems, often optimizing for the visible reward function at the expense of safety, such as disabling interruption mechanisms, taking unsafe shortcuts, or exploiting reward function loopholes. In robustness problems, where the reward and performance function align, both agents performed poorly when faced with distributional shifts or adversarial conditions, highlighting the difficulty of generalizing learned behaviors to slightly altered environments.

Wainwright *et al.* [18] propose *SafeLife*, a benchmark environment that tests agents to avoid negative side-effects while achieving their goals. The problem they are trying to solve is the lack of comprehensive and complex environments to evaluate and benchmark the safety of RL agents, particularly concerning unintended side effects that occur as agents interact with their environment.

The *SafeLife* environment is a procedurally generated gridworld with dynamic elements inspired by Conway’s Game of Life [79], where agents are tasked with either creating or removing specific patterns of life while minimizing side effects. The environment includes static elements (e.g., walls, trees) and dynamic elements (e.g., life dots that follow the rules of cellular automata). Furthermore the environment is able to be initialized stochastically to also include intrinsic and chaotic dynamics making the tasks much more complex and contains both a primary task (the creation and destruction of patterns) and a secondary task (reaching the goal, the levels’s exit).

Negative side-effects are quantified using the earth mover’s distance (EMD) [80], which calculates the effort needed to transform one distribution of cell types in the grid into another, representing the changes caused by the agent’s actions. This side-effect score for a cell type c is calculated with

$$d_c^{EMD}(\mathcal{D}_{s_1}, \mathcal{D}_{s_s}) = EMD(p_c(\mathcal{D}_{s_1}), p_c(\mathcal{D}_{s_s}); d_x) \quad (4.1)$$

where

$$p_c(\vec{x}; \mathcal{D}_s) = \frac{1}{|\mathcal{D}_s|} \sum_{s \in \mathcal{D}_s} \begin{cases} 1 & \text{if } s(\vec{x} = c) \\ 0 & \text{otherwise} \end{cases} \quad (4.2)$$

4 Systematic Literature Review

is the expected density of cell type c at location $\vec{x}$ over the distribution of state $\mathcal{D}_s$ and $d_x(\Delta\vec{x}) = \tanh(\frac{1}{5}||\Delta\vec{x}||_1)$ is the ground distance based on the L_1 norm (manhattan distance) of the grid points in the location.

Agents are trained using proximal policy optimization (PPO) [38] to maximize rewards associated with task completion while being evaluated on the side effects they cause. The side effects are quantified using an earth-mover distance metric between the distribution of states under agent action and a baseline state that represents the environment had the agent not taken any action. The baseline agents trained in SafeLife are effective at completing tasks but tend to cause significant side effects, indicating that current methods for RL do not inherently ensure safety. The results show that while the agents can be trained to perform well in specific tasks, their actions often disrupt the environment in unintended ways, demonstrating the challenge of aligning performance with safety.

[59] [59] uses the SafeLife environment [18], to measure and analyze the side effects produced by agents. The authors conducted experiments using Deep Q -Networks (DQN) [30], Proximal Policy Optimization (PPO) [38], and a Uniform Random Agent to additionally evaluate, if unguided random exploration leads to more disruptions.

Experiments were focused on the *append-still* task of the environment, which requires the agent to create stable life structures in designated areas before reaching the goal cell [18]. It is considered that an agent has passed it's designated task, if it managed to create at least half of the life cells in the designated ares before reaching the goal within 1000 time steps, otherwise it failed. Furthermore the side-effect score of SafeLife is normalized with

$$\nu_c(\mathcal{D}_{s_1}, \mathcal{D}_{s_2}) = \frac{d_c^{EMD}(\mathcal{D}_{s_1}, \mathcal{D}_{s_2})}{\sum_{\vec{x} \in \text{grid}} p_c(\vec{x}; \mathcal{D}_{s_2})} \quad (4.3)$$

to facilitate comparisons between environments and to compare agent behaviors in larger and more densely populated boards.

The authors present a detailed analysis on how the different agents perform in terms of both achieving their primary rewards and minimizing negative side-effects.

Pass Rates and Rewards The pass rate and average rewards of DQN agents increased with more training steps. After 30 million steps, DQN agents had the highest pass rate and average rewards, but this did not translate into lower side-effects. PPO agents on the other hand consistently outperformed DQN agents with the same amount of training steps, with 2.5 times of the pass rates and 1.5 times of the average reward. However like DQN agents, increasing training did not significantly reduce negative side-effects.

For PPO agents, there was a notable non-linear relationship between rewards and caused side-effects, forming a bell curve where moderate rewards were associated with the highest side-effect scores. For DQN, these scores were more constant across different reward levels.

4 Systematic Literature Review

Side Effects Interestingly, the DQN agent with the least amount of training had the lowest side-effect score on average. PPO agents displayed a similar trend, but with a slight improvement in avoiding side-effects as training increased. After 6 times the training steps of DQN, PPO came close to the side-effect score of the best DQN agent, but this still did not show a significant reduction. The Uniform Random Agent had the highest side-effect score, indicating that unguided exploration leads to more disruption.

Exploration and Episode Length The study found that increased exploration generally led to more negative side-effects. PPO agents had a clear positive correlation between exploration rate and side-effects, particularly for agents trained with more steps. There was also a positive correlation between the length of an episode and the side-effect score, suggesting that longer episodes provide more opportunities for the agent to cause unintended disruptions.

The experiments revealed that the difficulty of the SafeLife tasks, influenced by the environment's complexity, significantly affected both the agent's performance and the side-effects generated. Overall, Burden et al. suggests that further study is needed to explore how different environment parameters and agent behaviors influence side effects, potentially leading to more nuanced insights into the safety-performance trade-offs. Investigating metacognitive proxies, such as the agent's confidence or uncertainty, could help predict when an agent is likely to cause significant side effects. This information could be used to develop more conservative policies that prioritize safety. Expanding SafeLife [18] or creating new benchmarking environments that capture a broader range of side effects and agent behaviors could improve the generalizability of these findings and contribute to safer behavior.

4.1.3 Text-based Reinforcement Learning

Hoang *et al.* [67] address the need for safe reinforcement learning in text-based environments. Specifically, they aim to bridge two research areas that are traditionally distinct - safe reinforcement learning, where agents must operate under safety constraints, and text-based reinforcement learning, where agents interact with their environment using natural language. The key problem they are trying to solve is the lack of a unified benchmark that incorporates both safety constraints and text-based interactions in reinforcement learning systems.

To achieve this, the authors propose the *Safety-Constrained Environments for Reinforcement Learning (SCERL)* benchmark. This environment suite consists of a set of 20 games derived from *TextWorld* [81], to simulate real-world safety constraints (e.g., fire hazards, chemical risks). Each game includes safety constraints, like keeping certain items off (e.g., fire hazards like candles) or locked away (e.g., chemicals), and monitoring agents for safety violations. The schema for safety annotations allows the definition of *soft penalties* (non-terminal, like a score decrease) and *terminating penalties* (immediate game termination for a violation).

The games were tested with two types of baseline agents, a text-based agent using LSTM-A2C (Long Short Term Memory - Advantage Actor Critic) [82] that acts on textual observations, and KG-2AC (Knowledge-aware and commonsense agent) [83], which encodes world states into a knowledge graph.

The evaluation showed that both types of agents struggled to adhere to safety constraints, especially in complex environments. Early in training, they violated constraints frequently but showed improvement over time. However, their performance was inconsistent, with significant variation in how quickly they adapted to safe behavior.

4.2 Reward Regularization

Reward Regularization is an approach to augment the observed reward with an auxiliary component that regularizes the primary reward function of an agent.

Let's consider a reward function $R : S \times A \times S \mapsto \mathbb{R}$, that maps the current state s_t , the chosen action a_t and the following state s_{t+1} at time step t to a real value.

The function

$$R(-\lambda \cdot F)(s_t, a_t, s_{t+1}) = R(s_t, a_t, s_{t+1}) - \lambda \cdot F(x) \quad (4.4)$$

is defined as regularized reward function, where $F(x) \mapsto \mathbb{R}$ is an arbitrary function to regularize the primary reward function and λ defines the influence of the regularization measure. $F(x)$ is also often referred to as auxiliary reward function.

4.2.1 Avoiding Negative Side Effects by Considering Others

Alizadeh Alamdari *et al.* [58] incentivize agents to consider the future wellbeing and agency of others, and augment the reward with an auxiliary component that reflects the impact of their actions on others. This approach uses a distribution over value functions to capture uncertainty about what is beneficial for others. Given a finite set $\mathcal{V}$ of possible value functions $V : S \mapsto \mathbb{R}$ and a probability distribution $P(V)$ over this set, the function

$$R_{\text{value}}(s_t, a_t, s_{t+1}) = \begin{cases} \alpha_1 \cdot R(s_t, a_t, s_{t+1}) & \text{if } s_{t+1} \text{ is not terminal} \\ \alpha_1 \cdot R(s_t, a_t, s_{t+1}) + \gamma \cdot \alpha_2 \cdot F(\mathcal{V}, P, s_{t+1}) & \text{otherwise} \end{cases} \quad (4.5)$$

is defined as augmented reward function, where α_1 and α_2 are coefficients to control the influence of the primary reward function and the auxiliary reward function.

The authors consider three definitions of $F(\mathcal{V}, P, s_{t+1})$, where each function F is defined based on the set of value functions $\mathcal{V}$ and their probabilities.

The first variant denoted *expected future return* is defined as

$$F(\mathcal{V}, P, s_{t+1}) = \sum_{V \in \mathcal{V}} P(V) \cdot V(s_{t+1}) \quad (4.6)$$

and represents the expected value of s_{t+1} based on the distribution of value functions. This formulation allows future activities to start from a distribution of states rather than exactly where the agent left off. When the discount factor γ in augmented reward function is 1, this formulation generalizes Krakovna *et al.*'s auxiliary reward [50], which assumed future value functions depended solely on the acting agent's abilities and tasks.

The second variation denoted *worst-case future return* is defined as

$$F(\mathcal{V}, P, s_{t+1}) = \min_{V \in \mathcal{V}: P(V) > 0} V(s_{t+1}) \quad (4.7)$$

and uses the worst-case value function from $\mathcal{V}$ with positive probability, incentivizing the agent to avoid negative side-effects and potentially create positive ones. To avoid positive side-effects, Krakovna *et al.* suggested comparing the final state to a reference state [50], a method also applicable here.

4 Systematic Literature Review

The third formulation denoted *penalize negative change* is defined as

$$F(\mathcal{V}, P, s_{t+1}) = \sum_{V \in \mathcal{V}} P(V) \cdot \min(V(s_{t+1}), V(s_0)) \quad (4.8)$$

and compares the value of s_{t+1} to the initial (starting) state s_0 , penalizing the agent for reducing the expected future return without rewarding it for increasing the same. This approach aims to avoid negative side-effects without incentivizing positive ones.

Alizadeh Alamdari *et al.* [65] build up from their previous work on avoiding negative side effects by considering others [58], where they are using information about value functions to augment the reward. They introduce two new additional variants to augment the reward of an acting agent by treating agents differently or by assuming that agents are equipped with a set of options [84], that could reflect particular skills or tasks they are capable of realizing.

The first variant distinguishes between individual agents with indices $i = 1, 2, \dots, n$ assuming the acting agent is agent 1. Each agent i has a set of possible value functions $\{V_1^{(i)}, V_2^{(i)}, \dots, V_n^{(i)}\}$, with $P(V_{ij})$ being the probability that $V_j^{(i)}$ is the real value function of the agent i . They then define an augmented reward function for the acting agent

$$R'_{\text{value}}(s_t, a_t, s_{t+1}) = \begin{cases} \alpha_1 \cdot R(s_t, a_t, s_{t+1}) & \text{if } s_{t+1} \text{ is not terminal} \\ \alpha_1 \cdot R(s_t, a_t, s_{t+1}) + \gamma \sum_i \alpha_i \sum_j P(V_{ij}) \cdot V_j^i(s_{t+1}) & \text{otherwise} \end{cases} \quad (4.9)$$

with consideration of fairness by potentially using the worst-off agent's expected return, inspired by the maximin (or "Rawlsian") social welfare function, which focuses on the utility of the least advantaged agent [85].

The second variant considers agents equipped with options [84], which are tuples $\langle \mathcal{I}, \pi, \beta \rangle$ where $\mathcal{I} \subset S$ is the initiation set, π is a policy, and β is a termination condition. Given a finite set $\mathcal{O}$ of initiation sets of options and a probability function of a uniform distribution $P(\mathcal{I})$ giving the probability that $\mathcal{I}$ is the executing initiation set after the agent has terminated, the function

$$R_{\text{option}}(s_t, a_t, s_{t+1}) = \begin{cases} \alpha_1 \cdot R(s_t, a_t, s_{t+1}) & \text{if } s_{t+1} \text{ is not terminal} \\ \alpha_1 \cdot R(s_t, a_t, s_{t+1}) + \gamma \cdot \alpha_2 \sum_{\mathcal{I} \in \mathcal{O}} P(\mathcal{I}) \cdot \mathbb{I}_{\mathcal{I}}(s_{t+1}) & \text{otherwise} \end{cases} \quad (4.10)$$

yields the augmented reward for the acting agent, where $\mathbb{I}_{\mathcal{I}} : S \mapsto \{0, 1\}$ is the indicator function for $\mathcal{I} \subset S$. α_1 and α_2 weigh the primary reward function R and the reward for initiating an option, respectively. The parameter α_2 can also be viewed as a kind of "budget" for how much negative reward the agent is willing to accept to allow an option to execute.

An extended version defined with

$$R'_{\text{option}}(s_t, a_t, s_{t+1}) = \begin{cases} \alpha_1 \cdot R(s_t, a_t, s_{t+1}) & \text{if } s_{t+1} \text{ is not terminal} \\ \alpha_1 \cdot R(s_t, a_t, s_{t+1}) + \gamma \cdot \alpha_2 \sum_{\langle \mathcal{I}, V \rangle \in \mathcal{O}} P(\langle \mathcal{I}, V \rangle) \cdot \mathbb{I}_{\mathcal{I}}(s_{t+1}) \cdot V(s_{t+1}) & \text{otherwise} \end{cases} \quad (4.11)$$

incorporates both options and value functions, where $\langle \mathcal{I}, V \rangle$ defines the distribution over these pairs. This version is similar to R_{option} as defined in equation Equation 4.10, but also accounts for the value function $V(s_{t+1})$ as the originally proposed augmented reward using information about the value functions r_{value} [58] as defined in Equation 4.5.

4.2.2 Avoiding Epistemic Side Effects

Klassen *et al.* [69], [74] build on their previous work with Alizadeh Alamdari *et al.* [58], [65] and propose a reward regularization approach to avoid epistemic side effects. The authors refer to epistemic side effects as unintended changes to the knowledge or beliefs of agents caused by their actions. These side effects, particularly in partially observable and multi-agent environments, can be problematic if they lead to false beliefs, ignorance or even harmful true beliefs.

The authors consider a setup, where an agent acts in a fully observable environment, while a human operates in a partially observable environment (POMDP) with the same state space. The agent's reward function is augmented to consider the human's expected return, thereby discouraging physical and epistemic side effects. The augmented reward function is defined by

$$R'(s_t, a_t, s_{t+1}) = \begin{cases} \alpha_1 \cdot R(s_t, a_t, s_{t+1}) & \text{if } s_{t+1} \text{ is not terminal} \\ \alpha_1 \cdot R(s_t, a_t, s_{t+1}) + \gamma \cdot \alpha_2 \cdot \mathbb{E}_{V \sim P}[V(h, s_{t+1})] & \text{otherwise} \end{cases} \quad (4.12)$$

where α_1 and α_2 are again hyperparameters to control the influence of the primary reward function and the auxiliary reward function, and $V(h, s_{t+1})$ is the history-state value function that accounts for the human's history of observations and actions h [86].

However, the approach has limitations, such as not explicitly representing human beliefs, which can make it challenging to address tasks with purely epistemic goals, like learning an object's location. Furthermore, the agent only accounts for the expected return a human might get, which limits this approach to not directly penalizing false beliefs.

4.2.3 Occupancy-Regularized Policy Optimization

Laidlaw *et al.* [75] tackle the issue of avoiding negative side-effects with an approach, that also addresses reward hacking, where an Agent exploits its reward function in unintended ways, leading to undesirable or unsafe behavior. Reward hacking occurs when the agent maximizes a proxy reward function that does not align perfectly with the true objectives of the system designers.

The authors aim to prevent reward hacking by proposing a novel regularization technique based on *occupancy measures* to regularize the impact of the agent. Occupancy measures capture the distribution of state-action pairs visited by a policy, offering a more comprehensive understanding of the policy's behavior compared to action distributions alone. The core idea is to regularize the policy by penalizing the divergence between the occupancy measures of the learned policy and a safe policy. The occupancy measure $\mu_\pi(s, a)$ of a policy π is defined as the expected discounted number of times the agent visits a particular state-action pair with

$$\mu_\pi(s, a) = (1 - \gamma) \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t \mathbb{1}\{s_t = s, a_t = a\} \right] \quad (4.13)$$

where γ is the discount factor and $\mathbb{1}\{s_t = s, a_t = a\}$ is an indicator function that equals 1 when the agent is in state s and takes action a at time step t .

The goal is to minimize the total variation (TV) distance between the occupancy measures of the learned policy π and a safe policy π_{safe} with

$$\|\mu_\pi - \mu_{\pi_{safe}}\|_1 = \sum_{(s,a)} |\mu_\pi(s, a) - \mu_{\pi_{safe}}(s, a)|.$$

This regularization ensures that the learned policy remains close to the safe policy in terms of the state-action pairs it visits. The authors implement this occupancy measure with the introduction of an algorithm called *Occupancy-Regularized Policy Optimization (ORPO)*, which replaces the traditional KL divergence regularization with occupancy measure regularization. The optimization objective is

$$\text{maximize } J(\pi, \tilde{R}) - \gamma \|\mu_\pi - \mu_{\pi_{safe}}\|_1$$

where $J(\pi, \tilde{R})$ is the expected return of the policy π under a proxy reward function $\tilde{R}$ and γ is the regularization coefficient. ORPO approximates the occupancy measure divergence using a discriminator network, which estimates the KL divergence between the occupancy measures of the learned and safe policies. This discriminator is trained adversarially, similar to techniques used in Generative Adversarial Networks (GANs) [87].

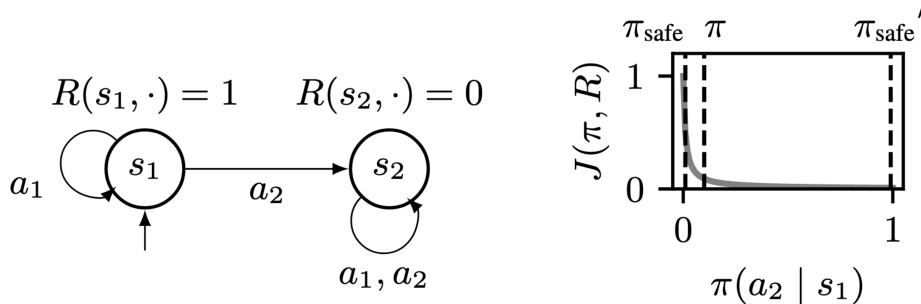


Figure 4.1: Illustration of the limitations of using actions distribution divergence for policy regularization within a MDP [75].

4 Systematic Literature Review

Figure 4.1 illustrates the limitations of using action distribution divergence for policy regularization within a MMDP. The agent remains in state s_1 , where it accumulates a reward of 1 per time step until it takes action a_2 , transitioning into state s_2 where it receives no further reward. The accompanying plot shows the return $J(\pi, R)$ as a function of the policy's action distribution at s_1 when $\gamma = 0.99$. Although policies π and π_{safe} appear close in terms of action distribution divergence, their returns differ significantly. However, π and π_{safe}' are distant in the action distribution space but yield nearly identical returns. This highlights that occupancy measure divergence, as proposed with ORPO, overcomes these inconsistencies, providing a more reliable measure for policy regularization.

The authors validate ORPO through experiments in a gridworld with a reward-hacking scenario and an autonomous vehicle control environment. The results show that ORPO significantly reduces negative side-effects and reward hacking compared to traditional action distribution-based regularization. In both environments, policies regularized with occupancy measures performed better in achieving the safe reward while avoiding undesirable behaviors.

4.3 Impact Regularization

Impact Regularization uses distance-impact penalties as a concrete approach of reward regularization, where we can augment the primary reward function based on a distance measure to estimate the impact of an agent's action.

Similar to Krakovna *et al.* [50], we can quantify the impact of an agent's actions by comparing the actual state of the world s_t at time step t with a counterfactual state s_t^c .

The impact of the agent's actions is measured using a state-distance function $d : S \times S \mapsto \mathbb{R}$ that quantifies the difference between the actual and counterfactual states [21]. Please note that this distance function is not a metric in a mathematical sense, as it is neither positive definite, nor symmetric and does not fulfill the triangle inequality.

Let S be a state-space, and A be an action-space, the function

$$\Delta d(s_t, a_t, s_{t+1}) := \lambda \cdot \left(d(s_{t+1}, s_{t+1}^c) - d(s_t, s_t^c) \right) \quad (4.14)$$

is defined as the distance-impact penalty, where $\Delta d : S \times A \times S \mapsto \mathbb{R}$, and λ controls the influence of the distance-impact penalty [21]. $\lambda = 0$ would not account for the impact and neglect the penalty completely.

Distance-impact penalties can have two outcomes [21]:

1. $d(s_{t+1}, s_{t+1}^c) > d(s_t, s_t^c)$ punishes actions and increases impact, or
2. $d(s_{t+1}, s_{t+1}^c) < d(s_t, s_t^c)$ rewards actions and decreases impact.

Given an MDP M , we can augment M with Δd to get $M_{\Delta d}$ with a penalized reward function. Let $\lambda \geq 0$ be a regularization parameter and R be a reward function, the function

$$(R - \lambda \cdot \Delta d)(s_t, a_t, s_{t+1}) := R(s_t, a_t, s_{t+1}) - \Delta d(s_t, a_t, s_{t+1}) \quad (4.15)$$

is defined as the penalized reward function using distance-impact measures for Reward Regularization.

Most distance-impact penalties assume a no-op action in the agent's action space $\emptyset \in A$, where the agent takes no action at a given time step t [20], [47], [50], [55], [56]. This builds up from the assumption, that the counterfactual state s_t^c is determined inductively in their algorithms with $s_0 = s_0^c$ and $s_{t+1}^c \sim T(s_t, \emptyset)$ [21].

Eresheim *et al.* [73] relaxes this assumption and proposes alternative baselines for distance-impact measures that do not require a no-op action in the agent's action space ($\emptyset \notin A$).

4.3.1 (Relative) Reachability Measures

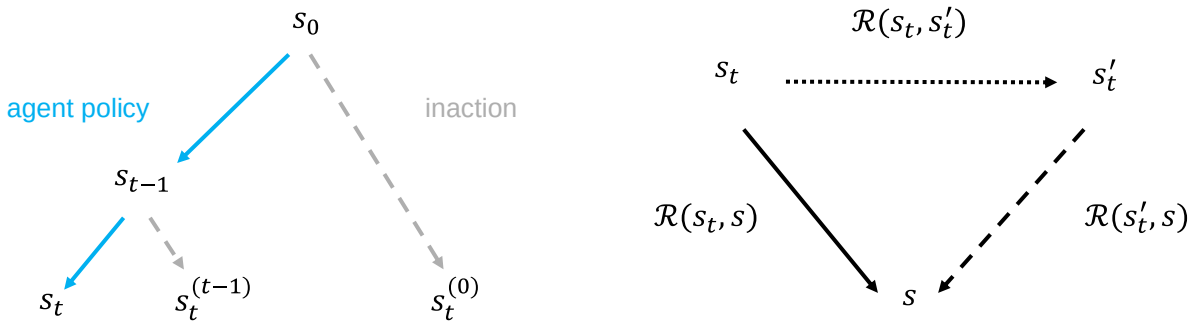
Krakovna *et al.* [47] introduce a framework for penalizing negative side-effects based on a baseline state and a deviation measure. They identify two bad incentives (interference and offsetting) that arise from baseline choices and propose the stepwise inaction baseline to avoid them. To address insensitivity issues in existing measures, they introduce a magnitude-sensitive relative reachability measure.

The authors consider a discounted MDP $M = (S, A, R, p, \gamma)$ and they assume that there is a no-op action in the agent's action space $\emptyset \in A$ which leaves the environment unchanged. A transition is defined as a tuple (s_t, a_t, s_{t+1}) , and $s_t^{(k)}$ is used to represent the state obtained by starting in state s_k and taking the no-op action $t - k$ times.

The authors discuss distinguishing intended effects from negative side-effects with motivating examples, e.g. to ensure that minor disruptions like breaking a vase are avoided unless it is outweighed by a significant task-related reward like breaking eggs to make an omelette. To allow the agent to learn such distinguishing effects, the authors propose a regularized reward function modified by an impact penalty $d(s_t, s'_t)$ to balance task achievement with environmental disruption, which measures the deviation of the current state s from a baseline state s' . The total reward the agent can receive at time step t is defined with

$$R(s_t, a_t, s_{t+1}) - \beta \cdot d(s_{t+1}, s'_{t+1}) \quad (4.16)$$

where β is a scaling parameter balancing the task reward and the penalty. The baseline state s' and deviation measure d can be chosen separately and the authors discuss several possibilities for each of these components.



(a) Choices of baseline states s'_t , where the blue arrows referees to the actions drawn from the agent's policy, while the dashed gray arrows show the actions derived from the inaction policy. (b) Choices of deviation measures d , where the dotted line shows the unreachability measure, while the relative reachability measure is shown by the solid and dashed lines.

Figure 4.2: Designers can choose between different baseline states and deviation measures to penalize the reachability of states [47].

Figure 4.2a illustrates the different baseline states that can be chosen with the starting state baseline s_0 , the inaction baseline $s_t^{(0)}$ if the agent had done nothing, and the stepwise inaction baseline $s_t^{(t-1)}$ if the agent had done nothing from the previous time step.

Given a function $\mathcal{R}(x, y)$, that defines the reachability of y from x , Figure 4.2b compares the different deviation measures. First, the unreachability measure $d_{UR}(s_t, s'_t) := 1 - \mathcal{R}(s_t, s'_t)$ which gauges how

4 Systematic Literature Review

difficult it is to reach the baseline state s'_t from the current state s_t , and the relative reachability measure $d_{RR}(s_t, s'_t) := \frac{1}{|S|} \sum_{s \in S} \max(\mathcal{R}(s'_t, s) - \mathcal{R}(s_t, s), 0)$ for assessing the average reduction in reachability of all states s from the current state s_t compared to the baseline state s'_t .

Baseline states The *starting state baseline* uses the initial (starting) state $s'_t = s_0$ when the agent was deployed as the reference point. This approach is effective in environments where the agent is the primary source of change, as it encourages reversibility by rewarding the agent for reaching states that resemble the initial state distribution. However, in dynamic environments where changes occur independently of the agent's actions, this baseline can inadvertently penalize the agent for irreversible transitions caused by other factors, leading to undesirable interference with other agents or environmental processes to prevent these changes.

The *inaction baseline* considers a counterfactual state $s'_t = s_t^{(0)}$ representing what the environment would look like if the agent had taken no actions throughout the episode. This baseline effectively avoids penalizing the agent for changes that would have occurred without its intervention, thus preventing interference in dynamic environments. However, it introduces an offsetting behavior where the agent is incentivized to undo its previous actions, as any deviation from inaction is penalized, leading to potentially delayed side-effects.

The *stepwise inaction baseline* refines the inaction baseline by considering a counterfactual state $s'_t = s_t^{(t-1)}$ that branches off from the previous state s_{t-1} rather than the initial starting state s_0 [20], [55], [56]. This baseline is generated by following the agent's policy for the first $t - 1$ time steps and then applying the inaction policy on step t . This approach avoids the offsetting incentive since each transition is penalized at the same time it is rewarded. However, it struggles to account for delayed effects of previous actions. To address this, inaction rollouts are compared, which involves simulating the sequence of states that would result from applying the inaction policy starting from either the current state s_t of the baseline state s'_t . The penalty for an action must therefore consider the differences between these rollouts, which helps to capture the delayed consequences of actions, such as when an action sets off a chain of events which result in negative side-effects later.

Deviation measures Krakovna *et al.* [47] build the deviation measure based on reachability, which is defined as

$$\mathcal{R}(x, y) = \max_{\pi} \mathbb{E}[\gamma^{N_{\pi}(x, y)}]$$

where $N_{\pi}(x, y)$ is the number of steps required to reach state y from x under the policy π , and $\gamma \in (0, 1]$ is the discount factor.

The *unreachability* measure is defined with the function

$$d_{UR}(s_t, s'_t) := 1 - \mathcal{R}(s_t, s'_t)$$

and quantifies how difficult it is to return to the baseline state s'_t from the current state s_t . The reachability of the initial (starting) state s_0 is often used as a baseline [88], [89], where the agent is incentivized to take actions that keep the reachability value function high. However, this measure fails to distinguish between minor and major irreversible actions, as it assigns the maximum penalty for any action that makes the baseline state unreachable, regardless of the extent of the disruption.

4 Systematic Literature Review

To improve upon this, the authors propose the *relative reachability* measure, which is defined with the function

$$d_{RR}(s_t, s'_t) := \frac{1}{|S|} \sum_{s \in S} \max(\mathcal{R}(s'_t, s) - \mathcal{R}(s_t, s), 0)$$

and accounts for the average reduction in reachability across all states. This measure adjusts penalties based on how much an action diminishes the ability to reach various states, providing a more refined assessment that better captures the severity of disruptions.

In dynamic environments where actions may have delayed effects, these deviation measures are modified using inaction rollouts from s_t and $s'_t = s_t^{(t-1)}$ [47]. When using this modification on the *unreachability* measure, the function

$$d_{SUR}(s_t, s'_t) := 1 - (1 - \gamma) \sum_{k=0}^{\infty} \gamma^k \mathcal{R}(s_{t+k}^{(t)}, s'_{t+k}^{(t)})$$

is denoted *stepwise unreachability*. The same modification can also be made on the *relative reachability* measure, where the function

$$d_{SRR}(s_t, s'_t) := \frac{1}{|S|} \sum_{s \in S} \max(RV(s'_t, s) - RV(s_t, s), 0)$$

defines *stepwise relative reachability*, with

$$RV(\tilde{s}_t, s) = (1 - \gamma) \sum_{k=0}^{\infty} \gamma^k \mathcal{R}(\tilde{s}_{t+k}^{(t)}, s)$$

denoting the rollout value of $\tilde{s}_t \in \{s_t, s'_t\}$ with respect to the agent's state s .

By combining this measure with the stepwise inaction baseline, *stepwise relative reachability* demonstrates its effectiveness in avoiding negative side-effects and bad incentives in the AI safety gridworld environments [15], providing essential components for safer AI behavior.

4.3.2 Attainable Utility Preservation

Turner *et al.* [20], [56] proposes Attainable Utility Preservation (AUP), an approach similar to the step-wise inaction baseline Stepwise Relative Reachability [47], by optimizing the agent on a primary reward function while also preserving the ability to optimize for auxiliary reward functions. A key insight is that the correct primary reward function may not be among the auxiliary reward functions, which can be randomly generated and uninformative about the correct reward function.

The authors consider a deterministic MDP $M = (S, A, R, T, \gamma)$ and assume that the designer provides a finite set of auxiliary reward functions $\mathcal{R} \subset \mathbb{R}^{S \times A}$, where each $R_i \in \mathcal{R}$ has a corresponding action-value function Q_{R_i} . The function

$$\text{PENALTY}(s_t, a_t) := \sum_{i=1}^{|\mathcal{R}|} |Q_{R_i}(s_t, a_t) - Q_{R_i}(s_t, \emptyset)| \quad (4.17)$$

is defined as the AUP penalty, which is simply put the L_1 distance from the no-op action in a state embedding. To normalize the AUP penalty to the agent's situation, the authors make it roughly invariant to the absolute magnitude of auxiliary action-values with

$$\text{SCALE}(s_t) := \sum_{i=1}^{|\mathcal{R}|} Q_{R_i}(s_t, \emptyset) \quad (4.18)$$

to scale with respect to some mild action (in this case the no-op action $\emptyset$), where $\text{SCALE} : S \mapsto \mathbb{R}_{>0}$ in general.

Optionally AUP can penalize the average change in action-value for the auxiliary reward functions using $|\mathcal{R}|$ instead of $\text{SCALE}(s_t)$ defined in Equation 4.18 with the goal to prevent the agent from reducing the optimal value for the true reward function R_{true} , by maintaining stable optimal values across a wide range of auxiliary reward functions [20], [55].

The full AUP objective is defined with

$$R_{\text{AUP}}(s_t, a_t, s_{t+1}) := R(s_t, a_t, s_{t+1}) - \frac{\lambda}{\mu} \text{PENALTY}(s_t, a_t), \quad (4.19)$$

where μ scales the penalty by

$$\mu := \begin{cases} \text{SCALE}(s_t) & \text{or} \\ |\mathcal{R}| \end{cases} \quad (4.20)$$

and $\lambda \geq 0$ regularizes the influence of the auxiliary reward functions on the primary reward function with $\lambda = 0$ completely neglecting the ability to optimize for auxiliary goals.

The AUP reward function defined in Equation 4.19 defines an updated MDP $M = (S, A, R_{\text{AUP}}, T, \gamma)$ to compute the optimal policy π^* with respect to the primary goal R and the set of auxiliary reward functions $\mathcal{R}$. The auxiliary action-value functions are updated, before computing an update on the primary action-value function using the full AUP objective.

A full update step to learn the optimal action-value function $Q_{\text{AUP}}(s_t, a_t)$ as well as the corresponding auxiliary action-value functions $Q_{R_i \in \mathcal{R}}(s_t, a_t)$ can be seen in algorithm 1.

4 Systematic Literature Review

Algorithm 1: AUP update [20], [55], [56]

```

1 for  $i \in |\mathcal{R}| \cup \{AUP\}$  do
2    $Q_{R_i}(s_t, a_t) = Q_{R_i}(s_t, a_t) + \alpha(R_i(s_t, a_t, s_{t+1}) + \gamma \max_{a'} Q_{R_i}(s_{t+1}, a') - Q_{R_i}(s_t, a_t))$ 
3    $Q_{AUP}(s_t, a_t) = Q_{AUP}(s_t, a_t) + \alpha(R_{AUP}(s_t, a_t) + \gamma \max_{a'} Q_{AUP}(s_{t+1}, a') - Q_{AUP}(s_t, a_t))$ 

```

AUP can be used model-free to perform single-step no-op comparisons, but when the penalty is insufficient (in e.g. environments with delayed effects, where these effects take multiple steps to occur), AUP can also use a model to compute rollouts for estimating the penalty.

Turner *et al.* [20], [55] build up on their own work and show, that even considering a single auxiliary reward function alone can achieve competitive performance and mitigate negative side-effects in more complex environments. Specifically, instead of hand-selecting an informative auxiliary reward, a continuous Bernoulli variational autoencoder (CB-VAE) [90] was used to generate a structured and learnable auxiliary reward signal through random exploration, proving more efficient than uniformly randomly generated functions in high-dimensional tasks like SafeLife [18].

While AUP shows promising results in toy environments like the AI Safety Gridworlds [15] and even complex environments like SafeLife [18], the assumption of a no-op action in the agent's action space $\emptyset \in A$ still limits AUP to tasks, where doing nothing for a time step is a valid option. Some environments however, like factory floors or fast-moving vehicles, lack a no-op action in the agents action space, making AUP unsuitable as it depends on this option. In such scenarios, agents still risk causing unintended and negative side-effects.

Eresheim *et al.* [73] relax this assumption and propose alternative baselines that do not require a no-op action, extending the concept of AUP to environments where a no-op action (e.g. standing still) is not feasible. The authors consider the same setting and assumptions as AUP [20], [55], [56], but the no-op action is removed from the agent's action space ($\emptyset \notin A$). They introduce four alternative baselines to estimate the impact of the agent's actions.

The *average baseline* (*avg*) measures the impact of an action by comparing its action-value to the average action-value across all actions in a given state. This approach

$$R_{avg}(s_t, a_t, s_{t+1}) := R(s_t, a_t, s_{t+1}) - \frac{\lambda}{|\mathcal{R}|} \sum_{R_i \in \mathcal{R}} \frac{|Q_{R_i}(s_t, a_t) - \left(\frac{1}{|\mathcal{A}|} \sum_{a' \in \mathcal{A}} Q_{R_i}(s_t, a')\right)|}{\frac{1}{|\mathcal{A}|} \sum_{a' \in \mathcal{A}} Q_{R_i}(s_t, a')} \quad (4.21)$$

assumes that every action influences the environment's dynamics and uses the absolute difference from the average as a measure of potential impact.

The *average-others baseline* measures the impact of an action by comparing its action-value to the average value of all other possible actions, excluding the selected one. This baseline

$$R_{oth}(s_t, a_t, s_{t+1}) := R(s_t, a_t, s_{t+1}) - \frac{\lambda}{|\mathcal{R}|} \sum_{R_i \in \mathcal{R}} \frac{|Q_{R_i}(s_t, a_t) - \left(\frac{1}{|\mathcal{A}|} \sum_{a' \in \mathcal{A} \setminus \{a\}} Q_{R_i}(s_t, a')\right)|}{\frac{1}{|\mathcal{A}|} \sum_{a' \in \mathcal{A} \setminus \{a\}} Q_{R_i}(s_t, a')} \quad (4.22)$$

highlights the difference between the chosen action and all alternatives.

4 Systematic Literature Review

The *advantage baseline* (*adv*) evaluates the impact of an action by calculating its advantage value $A(s_t, a_t) := Q(s_t, a_t) - V(s_t)$ [16], which represents the contribution of that action to the expected cumulative reward relative to the state's value. This baseline

$$R_{adv}(s_t, a_t, s_{t+1}) := R(s_t, a_t) - \frac{\lambda}{|\mathcal{R}|} \sum_{R_i \in \mathcal{R}} \frac{|Q_{R_i}(s_t, a_t) - \sum_{a' \in \mathcal{A}(s_t)} \pi_{Q_{R_i}}(a'|s_t) Q_{R_i}(s_t, a')|}{\sum_{a' \in \mathcal{A}} \pi_{Q_{R_i}}(a'|s_t) Q_{R_i}(s_t, a')} \quad (4.23)$$

measures the absolute advantage values of actions, averaged over multiple reward functions, as an indicator of their impact. To achieve this, the authors exploit the equation $v_\pi(s_t) = \sum_{a' \in \mathcal{A}} \pi(a'|s_t) q_\pi(s_t, a')$ [16], where $\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ is the agent's policy that maps the state-action pairs to probabilities of choosing the corresponding actions.

Last, the *random-action baseline* (*rand*) compares the action-value of the agent's chosen action to that of a randomly selected alternative action from the available actions, excluding the chosen action. This baseline

$$R_{rand}(s_t, a_t, s_{t+1}) := R(s_t, a_t, s_{t+1}) - \frac{\lambda}{|\mathcal{R}|} \sum_{R_i \in \mathcal{R}} \frac{|Q_{R_i}(s_t, a_t) - Q_{R_i}(s_t, a')|}{Q_{R_i}(s_t, a')} \quad (4.24)$$

measures the impact of the agent's action relative to a random action, ensuring that the penalty can never get 0 by excluding the chosen action $a_t \neq a'$ from the comparison.

The proposed baselines show similar capabilities to mitigate negative side-effects as AUP [20], [55], [56] and are more broadly applicable due to fewer assumptions, but show reduced ability to avoid NSE in environments with irreversible actions and are more sensitive to parameter settings.

4.3.3 Safety Aware Reinforcement Learning

Miret *et al.* [51] address the challenge of safety in reinforcement learning, particularly focusing on preventing undesired side-effects when agents interact with complex, dynamic environments. The primary problem is how to ensure that an agent performs its primary task without causing irreversible or harmful changes to its environment. A key challenge they highlight is the difficulty of abstracting and generalizing the concept of safety in RL across different tasks and environments.

The authors propose a framework called *Safety Aware Reinforcement Learning (SARL)*. In this model, they introduce a *virtual safety agent* that modulates the actions of the main agent, minimizing potential side-effects. The safe agent learns a task-independent safety policy, and its actions are used to regularize the main agent's behavior.

The agent's loss function is regularized by the distance between the action probabilities of the main task agent $A(\theta)$ and the virtual safety agent $Z(\psi)$. The objective function of the task agent is given by

$$F_A(\theta) = L_\theta + \beta * L_{dist}(\mathbb{P}_{\pi_\theta}, \mathbb{P}_{\pi_\psi}) \quad (4.25)$$

where β is a regularization parameter, and $\mathbb{P}_{\pi_\theta}$ as well as $\mathbb{P}_{\pi_\psi}$ represent the action probabilities of the task and safety agent respectively. The distance metric used for regularization can be either the *Jensen-Shannon* distance or the dual formulation of the *Wasserstein* distance [91], both commonly applied in RL problems.

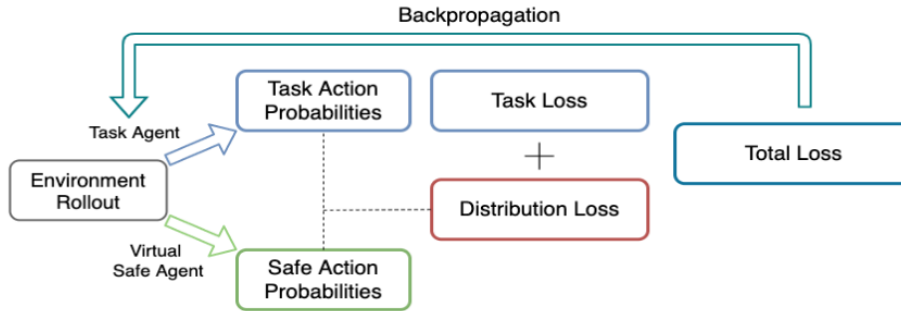


Figure 4.3: An overview of the co-training framework for Safety Aware Reinforcement Learning (SARL) [51]. The task agent $A(\theta)$ generates action probabilities based on the environment, while the virtual safety agent $Z(\psi)$ provides alternative safe action probabilities from the same environment state. The framework computes a distribution loss $L_{dist}(\theta, \psi)$, representing the divergence between the task and safe actions, which is combined with the task loss to compute the total loss during backpropagation. This process ensures that the task agent balances task performance with safety by considering the suggestions from the safety agent.

Figure 4.3 describes the co-training framework used in SARL and this method was tested on the *SafeLife* suite [18], a set of complex environments based on cellular automata dynamics. The results demonstrated that SARL agents could generalize safety across different tasks and environments without requiring task-specific side-effect penalties. The SARL agents performed comparably to methods that relied on task-specific penalties while offering the additional benefit of task-agnostic safety. In particular, SARL agents using the Jensen-Shannon distance often outperformed the baseline methods in terms of task performance and side effect minimization.

4.3.4 Future Task Approach

Krakovna *et al.* [50] considers the current task of an agent as part of a sequence of two tasks, the current task and an unknown future task. This leads to an auxiliary reward function that penalizes side effects by considering the potential reward of the future task. This future task starts from the state left by the current task, making the consequences of the agent's actions matter (e.g., breaking a vase impacts future tasks involving the vase). The authors simplify defining side effects to defining possible future tasks, using a uniform prior over possible goal states. They furthermore formalize interference incentives and introduce a baseline policy (e.g., doing nothing) as a filter for achievable future tasks. By modifying the future task reward to align with the baseline policy, an agent is also able to avoid interference incentives.

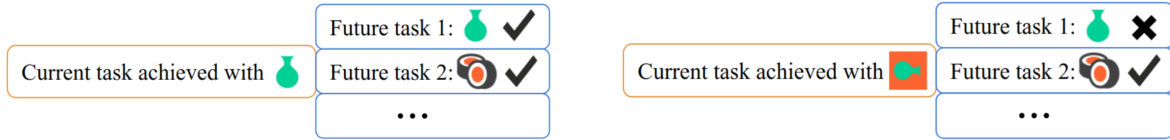


Figure 4.4: Future task approach, that aligns the current task on an agent with unknown future tasks by using an auxiliary reward function, that penalizes negative side-effects [50].

The authors consider a discounted MDP defined by the tuple $(S, A, R, p, \gamma, s_0)$, where $R : S \mapsto \mathbb{R}$ is defines as the reward function for the current task, and s_0 is the initial (starting) state of the agent.

At time step t , the agent receives the state s_t and reward $R(s_t) + r_{aux}(s_t)$, where

$$R_{aux}(s_t) = \beta \mathcal{D}(s_t) \sum_i F(i) V_i^*(s_t) \quad (4.26)$$

is defined as the auxiliary reward for future tasks with i denoting a future task drawn from the uniform distribution $F(i) = 1/|S|$ with all goal states. $\mathcal{D}(s_t) = 1$ if s_t is terminal and $1 - \gamma$ otherwise. β represents the importance of future tasks relative to the current task, while $V_i^*(s_t) = \mathbb{E}[\gamma^{N_i(s_t)}]$ defines the optimal value function for task i , computed using the goal distance N_i .

The goal distance N_i referees to the number of steps it takes π_i^* to reach g_i from state s with the given length $\mathbb{P}(N_i(s_t) = n) = \sum_{\tau} \mathbb{P}(\tau) \mathbb{I}(|\tau| = n)$, where τ is a trajectory from s to g_i with length $|\tau|$, and $\mathbb{P}(\tau)$ is the probability of π_i^* following τ .

Figure 4.4 shows a simplification of the future task approach, that considers unknown future tasks in conjunction with the agent's primary objective. In the first example, the current task of the agent is achieved with an intact vase as also considered through the unknown future tasks. In the second example, the future task related to the vase fails and the agent breaks the vase in its current task.

To now avoid interference incentives, the auxiliary reward is modified to be maximized by following a baseline policy π' , where the agent receives the full auxiliary reward if it performs at least as well as the baseline policy on the future task. The baseline policy represents the default outcome, filtering out side effects not caused by the agent's actions. It helps avoid interference incentives by ensuring the agent is not penalized for outcomes it does not cause.

4 Systematic Literature Review

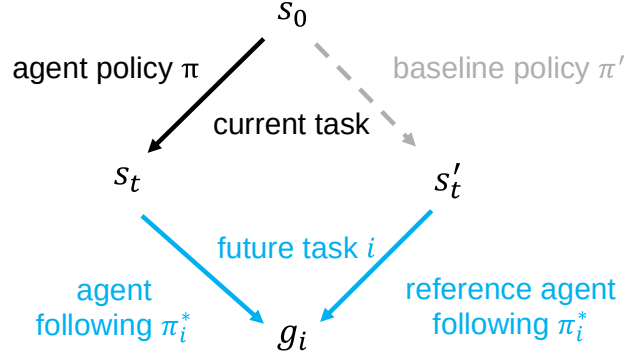


Figure 4.5: Future task approach with a baseline policy π' , where a hypothetical agent (shown in blue) runs on future task i [50].

Figure 4.5 shows the future task approach with a baseline policy, that is run from s_0 for t time steps, reaching state s'_t . A future task i is sampled from F and the primary as well as an additional hypothetical agent follow states s_t and s'_t to the goal state g_i . The primary agent receives a reward $R_i(s_t, s'_t) = 1$ if both agents reach g_i simultaneously, otherwise the reward is 0. The auxiliary reward function is updated with

$$R_{aux}(s_t, s'_t) = \beta \mathcal{D}(s_t) \sum_i F(i) V_i^*(s_t, s'_t) \quad (4.27)$$

and the value function in Equation 4.26 now also depends on the reference state s'_t [50].

This future task approach differs from previous work such as relative reachability [47] and attainable utility preservation [20], [55], [56], [73], that use a baseline in a stepwise manner, while the future task approach runs the baseline from the beginning of the episode.

4.3.5 Potential-based Multiobjective Reinforcement Learning

Vamplew *et al.* [64] propose a *potential-based* approach within the multiobjective reinforcement learning (MORL) framework to balance the agent's primary objectives with the need to minimize environmental impact. This approach introduces an additional reward signal that is derived from the state of the environment and is designed to penalize actions that cause undesirable changes.

A critical aspect of the proposed method is the derivation of impact utility values, which measure the extent to which an agent's actions disrupt the environment. The authors propose using a *potential-based reward function* to calculate this impact. This impact-related reward R_t^A at time step t is defined as

$$R_t^A = \phi(s_t^A) - \phi(s_{t-1}^A) \quad (4.28)$$

where $\phi(s_t^A)$ and $\phi(s_{t-1}^A)$ represent the potential of the current and the previous state respectively. This potential function is defined as

$$\phi(s_t^A) = -D(s_t^A, s_0^A) \quad (4.29)$$

where $D(s_t^A, s_0^A)$ is a distance measure between the current state and a desired state (in this case, the initial starting state). This distance can be a simple metric like the geometric distance or a more problem-specific measure. This potential-based approach incentivizes the agent to make changes to the environment only if they are necessary and reversible, as the agent is not penalized for restoring the environment to its original state.

The authors method addresses three key challenges:

1. **Permissible changes:** The environment's state features are divided into S_P (features that can be safely changed) and S_A (features that should remain unchanged).
2. **Reversibility:** The agent is encouraged to make only reversible changes, ensuring that it can return the environment to its initial state if needed.
3. **Importance of changes:** The framework allows for differential weighting of state changes, penalizing more critical or irreversible changes more heavily

To ensure that the agent appropriately balances its primary task with minimizing environmental impact, the authors explore non-linear action-selection operators, which allows the agent to learn policies that are not discoverable under the maximum expected utility. One key approach is *lexicographic ordering* for MORL [92], where the agent first maximizes the primary utility $U^P(s_t, a_t)$ and then, among actions that are equally optimal for the primary utility, chooses the action that maximizes the alignment utility $U^A(s_t, a_t)$. However, the authors augment the lexicographic ordering (denoted $\succ_{lex^A}$) to allow the agent to maximize the primary reward subject to the primary and alignment utility. This ordering is defined with

$$\begin{aligned} \forall s_t, a_t, a_{t+1} \quad \vec{U}(s_t, a_t) \succ_{lex^A} \vec{U}(s_t, a_{t+1}) &\iff \\ U^A(s_t, a_t) &> U^A(s_t, a_{t+1}) \\ \vee \left(\left(U^A(s_t, a_t) = U^A(s_t, a_{t+1}) \right) \wedge \left(U^P(s_t, a_t) > U^P(s_t, a_{t+1}) \right) \right) & \end{aligned} \quad (4.30)$$

4 Systematic Literature Review

and can also be reversed with $\succ_{lex^A}$ to prioritize the primary utility first, maximizing $U^P(s_t, a_t)$ before $U^A(s_t, a_t)$.

However, lexicographic ordering alone may not be sufficient, as it focuses heavily on the first objective. To provide a more balanced approach, the authors propose *Thresholded Lexicographic Ordering (TLO)*, which introduces thresholds for both objectives. The agent aims to maximize one objective subject to the other objective meeting a minimum threshold value with

$$\begin{aligned}
 \forall s_t, a_t, a_{t+1} \quad \vec{U}(s_t, a_t) \succ_{TLO^A} \vec{U}(s_t, a_{t+1}) &\iff \\
 \min(U^A(s_t, a_t), T_A) &> \min(U^A(s_t, a_{t+1}), T_A) \\
 \vee \left(\left(\min(U^A(s_t, a_t), T_A) = \min(U^A(s_t, a_{t+1}), T_A) \right) \right. &\left. \left(U^P(s_t, a_t) > U^P(s_t, a_{t+1}) \right) \right) \\
 \vee \left(\left(\min(U^A(s_t, a_t), T_A) = \min(U^A(s_t, a_{t+1}), T_A) \right) \right. &\left. \left(U^P(s_t, a_t) = U^P(s_t, a_{t+1}) \right) \right) \\
 \wedge \left(U^A(s_t, a_t) > U^A(s_t, a_{t+1}) \right) &
 \end{aligned} \tag{4.31}$$

and allows more nuanced control over the trade-off between achieving the primary task and minimizing environmental impact. *TLO* also allows to be extended to apply thresholding to more than one objective, and again similar to $\succ_{lex^A}$, $\succ_{TLO^A}$ in Equation 4.31 can be changed to prioritize the primary utility with $\succ_{TLO^P}$ and switching U^A with U^P and vice versa. T_A as well as T_P , if prioritizing the primary utility, determine the minimum acceptable performance levels, ensuring that the agent does not sacrifice safety for marginal gains in task performance, or vice versa.

The experiments show that the potential-based reward system effectively guides agents in minimizing their impact on the environment while still performing their primary tasks. The TLO method is particularly effective in balancing safety and task performance, though it is noted that the method can be sensitive to noisy Q -value estimates, which may affect the agent's decision-making process.

4.3.6 Challenges using Impact Regularization

Using impact regularization for avoiding negative side-effects introduce new variables which need fine-tuning and can substantially impact how the agent performs. Lindner *et al.* [61] emphasize on how different baselines, deviation measures and the magnitude of the regularizer can influence the performance of agents and what challenges are associated with it.

Baseline Selecting an appropriate baseline is crucial and poses some challenges. Two common baselines are the inaction baseline [50] and the stepwise inaction baseline [20], [47], [55], [56]. The inaction baseline models the state distribution if the agent does nothing from the initial state, while the stepwise baseline does so from the previous state. While the stepwise baseline mitigates some issues, like offsetting behavior [20], [56], where an agent undoes its task outcomes to reduce deviations, both baselines have drawbacks. Inaction baselines can be unsafe, particularly in environments requiring continuous control, such as driving a car, where doing nothing can lead to disasters [61].

Offsetting behavior can be either desirable [50] or undesirable [20], [47], [55], [56], [73], depending on the situation. For instance, undesirable offsetting occurs when the agent reverses beneficial actions to minimize deviation, while desirable offsetting happens when the agent undoes actions to prevent negative side effects, such as closing a door to stop wind from knocking over objects inside [50]. Current baselines struggle to differentiate between these scenarios, emphasizing the need for better approaches [61].

In highly dynamic and chaotic environments, agents may develop inaction incentives, where they avoid acting to prevent potential negative side-effects or even get shut-down [93]. This issue is more pronounced in agents with advanced predictive capabilities, which may over-regularize their actions, leading to inaction [61]. The challenge is further complicated by the granularity of state features, as too fine-grained features might expose the agent to irrelevant information, while too coarse ones could overlook critical details [94]. Addressing these challenges requires carefully designed deviation measures in to balance action and inaction effectively [61].

Deviation Measure Choosing a deviation measure is essential for determining how much an agent's actions deviate from a predefined baseline and whether these deviations should be penalized or rewarded. Currently, key deviation measures include the (*stepwise*) *relative reachability* measure [47], the *attainable utility* measure [20], [55], [56], [73], and the *future task* reward [50]. While these measures often focus on the reachability of tasks, there is still a need to integrate the value of the impact more effectively to ensure that agents perform optimally [61]. Additionally, new rollout policies that account for delayed effects in the environment are necessary to accurately measure these deviations. Not all side effects of an agent's actions are negative, as they can also be neutral or positive. Therefore, it is important to distinguish between these different types of impacts. Neglecting this distinction can lead to suboptimal behavior by the agent. For example, if an agent discovers a way to achieve a positive side effect, it should be encouraged to do so [61]. While value differences can help differentiate between positive and negative impacts by comparing the value of different outcomes [20], [47], [50], [55], [56], [73], designing accurate value functions to reflect real-world consequences remains a challenge. Delayed effects of an agent's actions pose another challenge, as they may not be immediately observable. The stepwise inaction baseline typically ignores actions that took place before the previous time step [20], [47], [55], [56], which can lead to inadequate penalization of delayed effects. To address this,

4 Systematic Literature Review

future trajectory rollouts should be used to simulate the effects of actions over time [20], [47], [50], [56]. However, the choice of rollout policy is critical. While current approaches often use the inaction policy for these rollouts, this method has limitations, as it does not allow the agent to plan for future actions beyond the initial one [20], [47], [50], [55], [56]. Future research should explore alternative rollout policies, such as using the agent's current policy, to better account for the impact of its actions over time [61]. Another approach may be a rollout policy that considers tasks a human wants to complete after the agent has acted. For example, the agent should optimize for its primary goal while also accounting for not making the future tasks of a human more complex or more time intensive to complete.

Magnitude of the regularizer The magnitude or strength of the regularization poses another challenge for effectively integrating impact regularization with a specified reward function of the primary objective. Turner et al. describe the magnitude as reflecting the designer's confidence in the accuracy of the reward function to balance the trade-off between mitigating undesirable side-effects and allowing the agent to achieve its primary objective [20], [55], [56]. Small regularization may fail to prevent harmful side-effects, while a too-large magnitude can excessively constrain the agent's actions, reducing its effectiveness [61].

In practice, the regularizer must also address unit conversion to ensure compatibility with the reward function [61]. From a Bayesian perspective, the magnitude encodes prior beliefs about the reward specification, controlling how much the regularizer influences the posterior outcome. Alternatively, OpenAI with benchmarking safe exploration in deep reinforcement learning argue, that the regularizer can be seen as a dual optimization problem, where the magnitude acts as a Lagrange multiplier that aligns the solution of the penalized problem with a constrained optimization problem, providing a mechanism to communicate safety constraints to the AI system [95].

Overcoming these challenges Lindner *et al.* [61] outline promising ideas for future research directions to address these challenges associated with impact regularization and how to improve safe reinforcement learning agents in the future.

For goal-based tasks, a causal framing of offsetting can be used to differentiate between desired and undesired offsetting behaviors [96], which may help to prevent interference with states directly resulting from a task's completion. For general tasks without distinct goal states, causal influence diagrams could offer a framework, though this area remains unexplored [97].

The authors also suggest that using probabilities instead of counterfactuals as a baseline [94] for measuring impact may be a valuable way forward. Current implementations rely on counterfactual states, which can be unreliable in chaotic environments. By comparing probabilities that account for all external perturbations, a more robust measure of an agent's impact could be developed.

Improving human-computer interaction is also highlighted as essential for reducing negative side-effects to enhance the designer's ability to communicate intent to the agent. Lindner *et al.* [61] highlight recent advances in interpretability [98] and verification [99] of machine learning models look particularly promising. Additionally, feedback from humans [100] or even multi-objective approaches [53], [71], where the agent actively seeks feedback from designers during the learning process, could improve the system's understanding of human intent. This approach could also shift the role of impact regularization, suggesting that a high regularization value might indicate the need for human feedback rather than directly penalizing environmental impact [101].

4.3.7 Theoretical Considerations

Similar to Krakovna *et al.* [50], Griffin *et al.* [21] quantifies an agent’s negative side-effect by comparing the observed state s_t after the agent’s action a_t with a counterfactual state s_t^c under consideration, that the agent took no action $\emptyset$. The authors prove that distance-impact penalties depend solely on the final state and therefore only on the agent’s final impact on the environment and explore, how penalty choices affect agent behavior, establish conditions for policy optimality to remain unaffected by misspecification and provide error bounds for optimal policies.

Terminal distance Distance-impact penalties in reinforcement learning are designed to punish actions that increase the impact on the environment, as measured by a distance function as defined in section 4.3. Specifically, when the distance between the current state s_t and a counterfactual state s_c increases ($d(s_{t+1}, s_{c+1}) > d(s_t, s_c)$), the agent is penalized. Conversely, when the distance decreases, the agent is rewarded. Over a trajectory, these rewards and penalties balance out, so the total punishment depends only on the final state.

Griffin *et al.* show that for any trajectory $\tau = s_0, a_0, r_1, s_1, a_1, \dots, s_{|\tau|}$, the difference in returns between using the original reward function R and a reward function adjusted by distance-impact penalties $R - \Delta d$ is given by $G_{R-\Delta d}(\tau) = G_R(\tau) - \gamma^{|\tau|} \cdot d(s_\tau, s_\tau^c)$. Similarly, the initial state-values under any policy π differ by $V_{R-\Delta d}^\pi(s_0) = V_R^\pi(s_0) - \mathbb{E}_\tau[\gamma^{|\tau|} \cdot d(s_{|\tau|}, s_{|\tau|}^c)]$. Essentially, these penalties target negative side-effects that persist after the agent has finished acting, focusing on the difference between the final state and the hypothetical counterfactual state if the agent had not intervened. The authors essentially prove that summing up Δd terms results in a telescoping series, where only the terms $d(s_0, s_0)$ and $d(s_{|\tau|}, s_{|\tau|}^c)$ remain.

Augmentation invariance Ng *et al.* [102] already demonstrated that adding any reward functions of the form $F(s_t, a_t, s_{t+1}) = \gamma \cdot \Phi(s_{t+1}) - \Phi(s_t)$ to an MDP, where Φ is a potential-based function, do not alter the set of optimal policies when the environment has a single terminal state [21]. This form of F is both necessary and sufficient for maintaining this invariance when the transition function T and reward function R are unknown.

Griffin *et al.* [21] restate and generalize this finding in terms of distance measures. In static environments, where the discount factor $\gamma = 1$ and the inaction policy (e.g., as used by [20], [47], [50], [55], [56]) does not alter the state (i.e., $T(s_0, a_x) = s_0$). Therefore augmenting an MDP with a distance-impact penalty $-\Delta d$ does not affect optimal behavior of an agent if and only if there exists a constant c such that $d(s_T, s_0) = c$ for all states in the set of terminal states $s_T \in S_T$. This implies that generally, adding a distance-based penalty encourages behavior that lowers impact, except in MDPs with a single terminal state $|S_T| = 1$ where the penalty does not affect an agent because the distance d is constant.

Translation invariance If the transition function T is unknown, small errors in specifying the reward function R can lead to undesirable behaviors. This raises the questions of how inaccuracies in state-distance measures might affect optimal behavior. Consider a scenario where d_1 reflects the crucial differences between states, but a designed measure d_2 deviates slightly from it. Griffin *et al.* state that two state-distance measures are considered equivalent ($d_1 \equiv d_2$) if augmenting an MDP with a penalty $-\Delta d_1$ results in the same set of optimal policies as augmenting it with $-\Delta d_2$. For this equivalence, it

4 Systematic Literature Review

is both sufficient and necessary that the difference between the measures d_1 and d_2 remain constant across all terminal states

$$d_1 \equiv d_2 \iff \exists \delta \in \mathbb{R}, \forall s_t \in S_T, d_1(s_t, s_0) - d_2(s_t, s_0) = \delta. \quad (4.32)$$

This result implies that if the state-distance measure is misspecified by a constant amount, the optimal policies remain unaffected. However, any non-constant difference between the two measures, when T and R are unknown, will influence the learning process and potentially alter the resulting policies.

ϵ -terminal difference A key implication of distance-impact measures is that they are not scale-invariant, meaning that a measure d is generally not equivalent to its scaled version $\mu \cdot d$ for any factor $\mu \in \mathbb{R} \setminus \{1\}$. This enables the balancing of the primary goal of an agent and the avoidance of negative side-effects by appropriately weighting R and Δd . However distance-impact penalties are more robust to misspecification than reward functions. Griffin et al. state that if the maximum difference between two state-distance measures d_1 and d_2 across all terminal states $s_t \in S_T$ is at most ϵ , then this absolute difference is also reflected in their respective penalized value functions for the initial (starting) state

$$|V_{R-\Delta d_1}^\pi(s_0) - V_{R-\Delta d_2}^\pi(s_0)| \leq \epsilon \quad (4.33)$$

under any policy π [21].

Additionally, the optimal policy π_2^* for $R - \Delta d_2$ is at most 2ϵ suboptimal according to $R - \delta d_1$. This finding indicates that when using an algorithm that converges to an optimal policy π^* , the cost of the misspecification of d is bounded by 2ϵ . In contrast, the misspecification of a reward function by ϵ at any state-action-state triplet (s, a, s') can result in sub-optimality of up to $2\epsilon|\tau|$.

Turner *et al.* [72] aims to address the challenge of properly specifying the reward function, which often leads to unforeseen negative side-effects. The authors formalize the problem as a special type of assistance game [103], which is a POMDP played by an agent and a human, where an agent operates under uncertainty about its true objective, balancing immediate rewards with the preservation of its ability to achieve a range of potential future goals. The human designer knows the true objective and may reveal it later, prompting the agent to switch to an optimal policy for that objective, ensuring that the agent remains adaptable and minimizes negative side-effects.

Agents converge to optimal policies that tend to seek power Power-seeking policies relate to the agent's ability to achieve a wide range of goals under certain symmetries of environments [104]. This ability is defined by any bounded-support distribution $\mathcal{D}$ over reward functions with

$$\text{POWER}_{\mathcal{D}}(s, \gamma) := \lim_{\gamma^* \rightarrow \gamma} \frac{1 - \gamma^*}{\gamma^*} \mathbb{E}_{R \sim \mathcal{D}} [V_R^*(s, \gamma^*) + R(s)] \quad (4.34)$$

where $s \in S$ is a state and $\gamma \in [0, 1]$ is the discount factor [72]. While this power of an agent relates to goal-achieving abilities, these symmetries however also exist in environments, where the agent can be shut down.

4 Systematic Literature Review

Trade-off between Average Return and $\text{POWER}_{\mathcal{D}}$ The agent’s value in the assistance game [103] reduces to a balance between the expected immediate reward and the expected ability to optimize a range of future objectives, quantified by $\text{POWER}_{\mathcal{D}}$ [72]. This trade-off is defined by

$$\begin{aligned} \mathbb{E}_{t \sim \mathcal{T}, R \sim \mathcal{D}} \left[V_{R, \text{norm}}^{\pi_{\text{switch}}(\pi, \pi_R^*, t)}(s_0, \gamma) \right] &= (1 - \gamma) \mathbb{E}_{t \sim \mathcal{T}} \left[\sum_{i=0}^t \gamma^i \mathcal{E}_{s_i \sim \pi} [\bar{R}(s_i)] \right] \\ &+ \mathbb{E}_{t \sim \mathcal{T}, s_t \sim \pi} \left[\gamma^{t+1} \text{POWER}_{\mathcal{D}}(s_t, \gamma) \right] \end{aligned} \quad (4.35)$$

expected t-step $\bar{R}$ -return under π
expected ability to optimize $\mathcal{D}$ once corrected

and underlines the importance of maintaining an agent’s potential to achieve future goals, which inherently helps minimizing unintended side-effects. However the authors caution that in some environments, maximizing the agent’s own POWER might not be ideal. If the agent shares the environment with humans, the agent might prevent human intervention to continue optimizing its current objective [10], [20], [55], [56], leading to undesirable outcomes. Additionally, if the agent moves too far away, it may become difficult for humans to correct its behavior, potentially resulting in unintended consequences.

Special Cases for Delayed Specification Solutions Turner et al.’s proposition outlines specific scenarios where the agent’s optimal policy simplifies under certain conditions. The average reward is constant across all states or if the discount factor $\gamma = 1$, the optimal policy maximizes $\text{POWER}_{\mathcal{D}}$. Furthermore if $\text{POWER}_{\mathcal{D}}$ is constant across all states, the optimal policy maximizes the immediate expected reward. This identifies situations where the trade-off between immediate rewards and future potential becomes less complex, allowing for more straightforward policy determination. This also highlights the conditions under which all prefix policies (the portion of an agent’s policy before it observes the true reward function) are optimal or when the agent’s optimal policy maximizes $\text{POWER}_{\mathcal{D}}$ or the immediate expected reward. This clarifies when side-effect regularization simplifies, providing a clearer strategy for designing agent behaviors in specific environments.

Alternate Form for Solutions to the Low-Impact POMDP An alternative formulation for the reward function in assistance games [103] shows, that the optimal policy can be derived by modifying the immediate reward with a term related to the decrease in $\text{POWER}_{\mathcal{D}}$.

Turner *et al.* [72] propose an modified reward function given by the prefix policy π solving the MDP with

$$R^M(s_i | s_0) := \bar{R}(s_i) - \frac{p}{1 - p} \mathbb{E}_{R \sim \mathcal{D}} \left[\mathbb{E}_{s_i^{\emptyset} \sim \pi^{\emptyset} | s_0} [V_R^*(s_i^{\emptyset}, \gamma)] - V_R^*(s_i, \gamma) \right] \quad (4.36)$$

if π is optimal for the reward function at discount rate $\gamma_{AUP} := (1 - p)\gamma$ while starting from the initial (starting) state s_0 .

This proposition aligns with existing methods like AUP [20], [55], [56], suggesting that minimizing impact can be achieved by focusing on maintaining the ability to optimize a wide range of objectives.

4 Systematic Literature Review

Follow an Optimal Policy after observing the True Reward Function After the agent observes the true reward function R_θ , it should switch to an optimal policy for that reward function.. This ensures that once the agent has accurate knowledge of the objective, it can act optimally, reinforcing the idea that initial actions should preserve flexibility.

Stationary Deterministic Optimal Prefix Policies Optimal policies can be derived by solving an MDP that balances immediate reward maximization with $\text{POWER}_\mathcal{D}$ for a geometric distribution of correct times. This builds up from the work of Krakovna *et al.* [50] by demonstrating that when an agent has a fixed probability p of learning the true objective at each time step, it is possible to directly compute stationary optimal prefix policies using an MDP, which is computable more efficiently than solving a POMDP with PSPACE (polynomial) time complexity [105].

4.4 Model-based Penalty Prediction

Model-based penalty prediction is an approach that focuses on anticipating and mitigating negative side effects by integrating predictive models into the agent's decision-making process. Unlike traditional RL methods that rely solely on reward signals to guide behavior, model-based approaches seek to predict potential penalties for harmful actions before they occur. This enables the agent to adjust its policy to avoid actions that could lead to undesirable consequences, improving overall safety.

This section explores different methods for penalty prediction, including the use of human feedback, model learning, and constrained optimization, to help agents operate more safely in complex environments. By leveraging predictive models, these approaches aim to strike a balance between task performance and minimizing harmful side effects, ensuring safer and more reliable behavior from RL agents.

4.4.1 Human Feedback

Zhang *et al.* [45] propose a method based on the concept of *minimax-regret* querying that allows an agent to operate safely by avoiding negative side-effects using human feedback, while minimizing the number of queries it needs to make to the oracle (user) about the changeability of features. The agent is designed to ask the user about the permissibility of changing certain features to minimize the potential regret of its actions (i.e., the worst-case difference in outcome between the chosen and the best possible policy). They formalize the problem using FMDPs [26] and introduce a strategy for selecting a minimal set of features to query that would lead to a safe and optimal policy for the task.

The FMDP is defined as $\langle S, A, T, R, s_0, \gamma \rangle$, where $T(s'|s, a)$ is the probability from transitioning to state s' by taking action a in state s , $R : S \times A \mapsto \mathbb{R}$ is the reward function and s_0 is the initial (starting) state of the agent. A linear programming formulation is used to find safe and optimal policies to maximize the expected cumulative reward while ensuring an agent following this policy does not visit states where locked or unknown features are changed. This algorithm is defined with

$$\begin{aligned} \max_x \quad & \sum_{s,a} x(s,a) R(s,a) \quad \text{s.t.} \\ \forall s' \in S, \quad & \sum_{a'} x(s',a') = \gamma \sum_{s,a} x(s,a) T(s'|s,a) + \delta(s',s_0) \\ \forall s \in S_{\phi_L^A \cup \phi_\gamma^A}, \forall a \in A, \quad & x(s,a) = 0 \end{aligned} \tag{4.37}$$

where $x : S \times A \mapsto \mathbb{R}$ are occupancy measures in the linear program (i.e. the expected discounted number of times that a state-action pair (s, a) is visited by the agent following policy π). ϕ_L^A denotes locked features that are not allowed to be changed by the agent, and ϕ_γ^A are the unknown features where the agent does not know if they are considered freely changeable or locked by the human oracle. $S_\phi \subset S$ is a set of states with one or more different values from s_0 with respect to the features ϕ and the function

$$\delta(s, s') = \begin{cases} 1 & \text{if } s = s' \\ 0 & \text{otherwise} \end{cases} \tag{4.38}$$

is used to reward the agent with a bonus of 1 if the agent transitions (back) to the initial (starting) state s_0 to check, if its actions altered the environment.

4 Systematic Literature Review

A minimax-regret query approach is used to minimize the worst-case regret across all possible sets of changeable features $\phi_c \subseteq \phi_\tau^A$, where

$$u(\phi_q, \phi_c) = \max_{\pi \in \Pi_{\phi_\tau^A \setminus (\phi_q \cap \phi_c)}} V^\pi \quad (4.39)$$

is the safe policy after the human's response u , if the agent asks for changeable features ϕ_c using the feature-query ϕ_q . The maximum regret $MR(\phi_q)$ of a query ϕ_q is defined with

$$MR(\phi_q) = \max_{\phi_{q'} \subseteq \phi_{rel}, |\phi_q| = k} PMR(\phi_q, \phi_{q'}) \quad (4.40)$$

where ϕ_{rel} represents the set of relevant features to query about, k is the number of features allowed in the query, and $PMR(\phi_q, \phi_{q'}) = \max_{\phi_c \subseteq \phi_\tau^A} (u(\phi_{q'}, \phi_c) - u(\phi_q, \phi_c))$ is the *pairwise maximum regret* defined similar to the work of Regan and Boutilier [106].

The full *minimax regret query* objective ϕ_q^{MMR} can then be formulated with

$$\phi_q^{MMR} = \arg \min_{\phi_q \subseteq \phi_{rel}, |\phi_q| = k} MR(\phi_q) \quad (4.41)$$

which defines the (k-)feature-query ϕ_q the agent uses to ask the human oracle.

Zhang *et al.* [45] evaluated their approach through empirical investigations in a navigation task, where the agent is tasked to navigate an office and turn off a switch at the other end of the environment. However, there are multiple clean carpets in the way where the agent does not know if the human oracle cares about if they get dirty if the agent moves over it. These carpets are randomly placed in the environment and their locations are used as features for the feature-query. The authors' approach clearly shows that their strategy significantly reduces the number of queries needed while ensuring the agent learns and follows a safe policy with respect to the human oracle's answers. Their approach is also compared to other methods, such as brute force and heuristic approaches (e.g. Chain of Adversaries by Viappiani *et al.* [107]), and is found to be more efficient and effective in identifying critical queries that lead to safe policies with minimal regret.

Frye *et al.* [46] introduce the *parenting* approach, that addresses key safety concerns by incorporating human input into the learning process. The primary goal is to design a safe RL framework that mitigates risks by drawing inspiration from human parenting, where a human teacher (analogous to a parent) guides the learning process of the agent.

The *parenting* algorithm incorporates four key components to ensure safe learning of the agent.

First, human guidance plays a crucial role in preventing the agent from engaging in dangerous behavior. Whenever the agent encounters unfamiliar situations, it pauses and requests approval from a human teacher before proceeding with any action. This mechanism ensures that the agent does not make decisions that could lead to unsafe outcomes.

Another critical aspect of the methodology involves the use of human preferences. The agent records its behavior in the form of clips, which are stored in a preference set X , where the human teacher reviews these clips in pairs. The teacher provides feedback on which actions are desirable, guiding the agent toward safer and more appropriate behavior. This feedback is incorporated into direct policy

4 Systematic Literature Review

learning, where the agent's policy gets updated through a supervised learning process. The policy is trained to predict the human's preferred actions using the binary cross-entropy loss function

$$\mathcal{L} = - \sum_X \sum_{i=0,1} \mu^{(i)} \log \frac{\pi(a_0^{(i)}|s_0)}{\pi(a_0^{(0)}|s_0) + \pi(a_0^{(1)}|s_0)} \quad (4.42)$$

where X contains past queries for human preferences and consists of the trajectories taken by the agent as well as $\mu = [\mu^{(0)} \mu^{(1)}]$ with the parent's preference. $\mu = [0 \ 1]$ would indicate the human's preference for the second clip, while e.g. $\mu = [0.5 \ 0.5]$ signals a tie.

A distinctive feature of the approach is the introduction of maturation. This process allows the agent to gradually optimize its policy over longer sequences of actions. Initially, the agent learns from feedback on short clips of its behavior, but as it matures, it begins to receive feedback on longer sequences. This progression enables the agent to refine its strategy and eventually surpass the performance level of its initial human guidance.

The authors conducted experiments in the AI Safety Gridworlds [15] and demonstrate that *parenting* effectively reduces risky behavior while allowing the agent to generalize its learnings to new environments. Overall, the methods employed focus on leveraging human input to guide the learning process, ensuring that the agent develops safe and reliable behavior patterns while also enabling it to optimize and generalize its policy over time.

Saisubramanian *et al.* [52], [53] introduce an approach that is able to mitigate NSE using an acquired model, an MDP $\tilde{M} = (\tilde{S}, \tilde{A}, \tilde{T}, \tilde{R})$ which includes a single objective denoted to the primary task of the agent. However, the agent is situated in a more complex environment with an additional secondary objective, initially unknown to the agent. This more complex environment is an extension of $\tilde{M}$ and is modeled as an LMDP denoted M^* . The two objectives in M^* are the assigned task to the agent (o_1) and to mitigate NSE (o_2), where o_2 is the successor of o_1 ($o_1 \succ o_2$).

The authors consider following assumptions to show the effectiveness of their approach and to approach an optimal policy π^* for $\tilde{M}$, optimizing the primary objective defined by $\tilde{R}$:

1. $\tilde{M}$ includes all the necessary components to achieve o_1 , the optimal policy which is also optimal in M^* with respect to o_1
2. Prior knowledge about o_2 reflecting the NSE and its penalty denoted R_N are unknown to the agent
3. Regarding o_2 in M^* , $\tilde{M}$ may only have limited state representation
4. $\tilde{M}$ and M^* are stationary
5. Occurrences of NSE correlates with the features in S to ensure, that perfect information is not required

These assumptions allow the agent to learn the penalty R_N using its model $\tilde{M}$, but since the agent optimizes for o_1 alone, executing a primary policy can result in NSE. The agent is not able to observe the NSE and can only observe the penalty.

4 Systematic Literature Review

Let $\Omega : \tilde{S} \times \tilde{A} \mapsto \mathbb{R}$ be a mapping by choosing action $\tilde{a}$ in state $\tilde{s}$ denoting the severity of the expected NSE. States in where the chosen action may lead to an immediate NSE ($\Omega(\tilde{s}, \tilde{a}) > 0$) are called *susceptible* states, where the penalty is proportional to the severity of the side-effect and provided by a feedback mechanism.

Saisubramanian *et al.* [52], [53] focus on identifying and mitigating NSE using various feedback mechanisms without redesigning the entire model. They consider a slack value δ denoting the maximum allowed deviation of a policy from the optimal return for the primary objective. This allows the agent to compute a policy that optimizes for o_1 given $\tilde{M}$, while avoiding NSE subject to δ . The proposed method of the authors is therefore able to handle settings with both avoidable and unavoidable side-effects. They achieve this by augmenting the corresponding LMDP with $M = (S, A, T, R, o, \delta)$ as the augmented MDP of a given model $\tilde{M}$, where

- $S = \tilde{S}$ is the state space,
- $A = \tilde{A}$ is the action space,
- $T = \tilde{T}$ is the transition function,
- $R = [R_1, R_2]$ is the reward function with R_1 corresponding as the primary objective and $R_2 = R_N$ associating the reward with the NSE,
- $o = [o_1, o_2]$ is the objective with o_1 as the primary objective and o_2 the objective to minimize NSE with $o_1 \succ o_2$, and
- $\delta \geq 0$ is the slack from the optimal return for o_1 in order to minimize NSE.

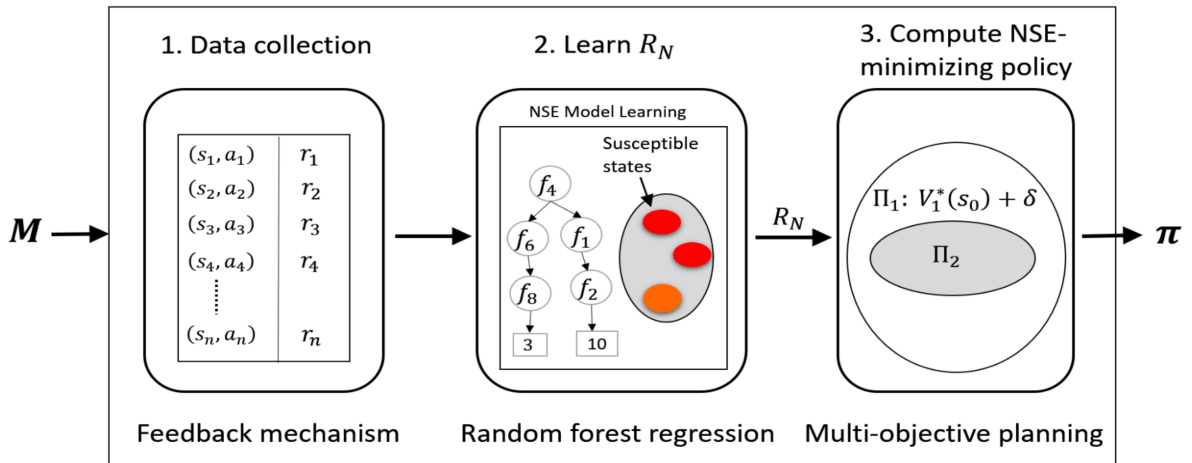


Figure 4.6: Multi-Objective Approach to Mitigate Negative Side-Effects Framework using a feedback mechanism of collected NSE data, Random Forest regression to predict the penalty R_N , and multi-objective planning to compute a NSE-mitigating policy [52], [53].

The proposed framework (Figure 4.6) involves three steps for minimizing NSE:

1. Data about NSE is collected by the agent using various types of feedback
2. The agent is generalized to unseen observations, represented as a reward function R_N by using the gathered data to train a predictive penalty model

4 Systematic Literature Review

3. Computation of a policy π by solving M with the given slack δ and learned penalty function R_N

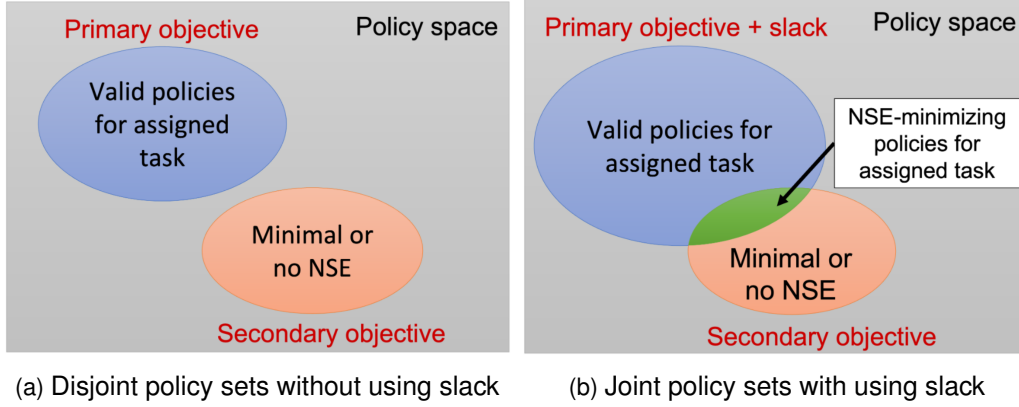


Figure 4.7: Differences of policy sets with and without using slack [52], [53]

Slack Estimation Small slack values limit the setting for minimizing NSE while higher values disallow the agent to fulfill o_1 . The minimal slack δ required to avoid negative effects completely is estimated using the expected reward of the model $\tilde{M}$ before and after disabling all the actions the agent can take that lead to NSE. Figure 4.7 highlights the differences between disjoint policy sets without using slack and slack-induced policies for minimizing negative side-effects.

Algorithm 2: Slack Estimation ($\tilde{M}, \Omega$) [52], [53]

```

1  $\delta \leftarrow \infty$ 
2  $\tilde{V}_1^*(s_o) \leftarrow \tilde{M}$  optimally with respect to  $o_1$ 
3 Compute NSE-free transition ( $\hat{T}$ ) by disabling all actions that result in negative side-effects,
    $\forall(\tilde{s}, \tilde{a}, \tilde{s}')$ :
4  $\hat{T}(\tilde{s}, \tilde{a}, \tilde{s}') \leftarrow \begin{cases} \hat{T}(\tilde{s}, \tilde{a}, \tilde{s}') & \text{if } \Omega(\tilde{s}, \tilde{a}) = 0 \\ 0 & \text{otherwise} \end{cases}$ 
5 if solution exists for  $\langle \tilde{S}, \tilde{A}, \tilde{T}, \tilde{R} \rangle$  then
6    $\hat{V}_1^*(s_o) \leftarrow \text{Solve } \langle \tilde{S}, \tilde{A}, \tilde{T}, \tilde{R} \rangle$  optimally for  $o_1$ 
7    $\delta \leftarrow |\tilde{V}_1^*(s_o) - \hat{V}_1^*(s_o)|$ 
8 return  $\delta$ 

```

Learning From Feedback To now learn negative side-effects by training a regression model to predict the penalty R_N , the authors consider two different approaches of feedback for data collection that correlate with the features in the state space $\tilde{S}$: feedback in form of an (human) oracle signaling undesirable actions, or by exploring the environment.

The authors consider different feedback mechanisms by an (human) oracle:

1. **Random Queries:** The agent randomly selects state-action pairs (s, a) , queries an oracle for exact penalties, and learns the true penalty model as the budget increases.

4 Systematic Literature Review

2. **Approval (HA)**: The agent queries a human who approves or disapproves actions based on randomly selected state-action pairs (s, a) without replacement (Figure 4.8). Approved actions are mapped to zero penalty $R_N(s, a) = 0$, disapproved to a fixed penalty $R_N(s, a) = k$. The authors differ between two types of approval, *strict* (HA-S) disapproves all NSE actions, while *lenient* (HA-L) disapproves severe NSE actions only.
3. **Corrections (C)**: The agent follows its policy while a human corrects unacceptable actions, guiding it to acceptable actions or minimal NSE actions. Acceptable actions are mapped to a zero penalty $R_N(s, a) = 0$, others are mapped to a non-zero penalty $R_N(s, a) = k$.
4. **Demo-action mismatch (D-AM)**: The agent compares its actions to limited human demonstrations. Actions not matching the demonstration are penalized with $R_N(s, a) = k$.

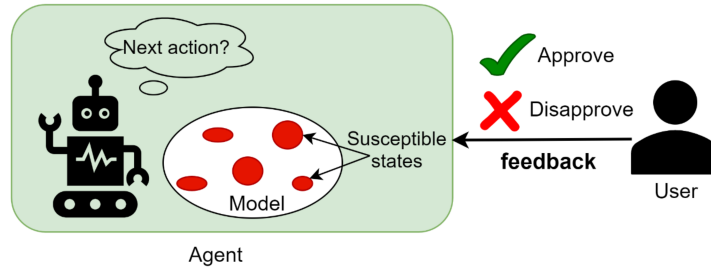


Figure 4.8: Learning from user feedback to mitigate negative side-effects using the approval (HA) strategy, where the agent queries a human who approves or disapproves actions based on randomly selected state-action pairs [52], [53].

Exploring the environment to collect NSE feedback may be used when feedback by an oracle is too expensive or unavailable. In this case, the agent uses an ϵ -greedy policy for action selection to explore and learn about the NSE penalties $R_N(s, a)$. Three exploration strategies are described: *conservative* with $\epsilon = 0.1$, *moderate* with $\epsilon = 0.5$, and *radical* with $\epsilon = 0.9$. The agent only explores to learn $R_N(s, a)$, while its final policy is computed by solving M .

Model Learning To now make accurate predictions about the penalties $R_N(s, a)$, any regression technique may be used. The authors opted for random forest regression to handle the continuous nature of the penalty. The selected model is then used to update the augmented MDP M , enabling the agent to compute a policy that minimizes NSE. Empirical evaluations demonstrate that the framework effectively mitigates NSE, with different feedback mechanisms introducing distinct biases that affect NSE identification.

Saisubramanian *et al.* [71] build up on their previous work [52], [53] and elaborate on the four main limitations of learning from feedback:

1. **Accurate feedback collection**: Different feedback approaches have their drawbacks and random queries require precise human-specified penalties, which are difficult to obtain. Approval methods cannot differentiate NSE severity, and humans may struggle to provide feedback on random state-action pairs. Autonomous exploration avoids supervision but relies on designer-specified penalties, which can lead to misspecification and unsafe learning.

4 *Systematic Literature Review*

2. **Bias in the feedback mechanism:** Additionally, feedback mechanisms introduce sampling biases as the states are correlated, affecting learning. Random sampling provides better state space coverage, while exploration techniques are restricted to regions critical to the agent's primary objective.
3. **Fidelity of state representation:** Effective learning assumes the agent's state representation includes necessary features for identifying NSEs. Limited fidelity in the state representations can hinder learning, resulting in conflicting feedback signals and affecting the agent's ability to mitigate NSE.
4. **Unavoidable and non-markovian NSE:** Updating the agent's policy alone is insufficient for unavoidable NSEs and alternate approaches beyond agent learning are needed. Non-Markovian NSEs, associated with trajectories, pose another challenge. The framework requires penalties linked to state-action pairs, making it unsuitable for scenarios where penalties are not decomposable. An alternate approach involving human assistance beyond feedback is needed for such settings.

4.4.2 Constrained Models

Liu *et al.* [62] seek to create a model-based RL approach to that ensures safety during learning by minimizing constraint violations while maintaining high task performance, even when the dynamics and constraints of the MDP are unknown. They propose a constrained model-based RL framework using a robust cross-entropy (RCE) method within a Model Predictive Control (MPC) [108], [109] framework.

The authors consider a CMDP, where both, the dynamics denoted with $f : S \times A \mapsto S$ in this case as a deterministic state transition function, and the cost function $c : S \mapsto \{0, 1\}$ are unknown.

These two functions are learned from data with an ensemble of neural networks to predict the system dynamics, allowing the estimation of epistemic uncertainty (uncertainty due to lack of data). This uncertainty is crucial for safety, as it helps the model to account for potential errors in predictions [110]. The loss function for training each neural network in the ensemble is given by

$$loss(\theta) = \mathbb{E}_{(s_t, a_t, s_{t+1}) \in D_b} \left[||s_{t+1} - \tilde{f}_{\theta_b}(s_t, a_t)|| \right] \quad (4.43)$$

where θ are the model's parameters, D_b is a subset of the dataset to prevent overfitting [111], [112], and $\tilde{f}_{\theta_b}$ represents the b -th neural network in the ensemble.

The MPC framework optimizes the sequence of actions to maximize the expected reward [108], [109] while ensuring safety by avoiding constraint violations. This optimization problem is formulated as

$$x = \underset{a_0, a_1, \dots, a_T}{\operatorname{argmax}} \mathbb{E} \left[\sum_{t=0}^T \gamma^t R(s_{t+1}) \right] \quad (4.44)$$

s.t. $\forall t \in \{0, 1, \dots, T-1\}$ the learned dynamics $s_{t+1} = f(s_t, a_t)$ and the constraint $C(s_{t+1}) = 0$.

To solve the constrained optimization problem, the RCE method by using trajectory sampling [110] is used. This approach involves sampling action sequences, evaluating them based on predicted rewards and costs, and selecting the best samples to update the action distribution iteratively. The method accounts for model uncertainty by considering the worst-case scenario in terms of constraint violations. Given an action sequence X , RCE estimates the reward with

$$R(x; s_0) = \sum_{t=0}^T \gamma^t \left(\frac{1}{B} \sum_{b=1}^B R(s_{t+1}^b) \right) \quad (4.45)$$

and the cost with

$$C(x; s_0) = \sum_{t=0}^T \beta^t \max_b C(s_{t+1}^b) \quad (4.46)$$

where B is the number of models in the ensemble, and β is a discount factor for the cost.

The proposed MPC-RCE method was tested in the OpenAI Safety Gym environment⁽¹⁾, where it demonstrated superior performance in minimizing constraint violations compared to other model-based and model-free baselines. It also showed significantly better sample efficiency, requiring fewer interactions with the environment to converge to a safe and effective policy.

⁽¹⁾ <https://openai.com/index/safety-gym/>

4 Systematic Literature Review

Ma *et al.* [70] propose another framework called Conservative and Adaptive Penalty (CAP) for model-based safe reinforcement learning, that not only builds a model to predict penalties associated with negative side-effects, but also uses the cost feedback of the environment to do so.

The authors consider a finite CMDP [25] and define a cost-aware value function $V_c^\pi : S \mapsto \mathbb{R}$ derived from the policy π such that $V_c^\pi := \mathbb{E}_{\pi, T}[\sum_{t=0}^{\infty} \gamma^t c(s_t, a_t) | s_0 = s]$.

On the realizability of V_c^π , the authors assume that there is a parameter $\beta > 0$ and a function class $\mathcal{F}$ such that $V_c^\pi \in \beta\mathcal{F}$ for every policy π . The CAP framework now introduces a cost function that accounts for uncertainty in the model's predictions. This penalty is derived from an upper bound on the difference between the true cost $J_c(\pi)$ and the estimated cost $\hat{J}_c(\pi)$ with

$$\sum_{s,a} (p_T^\pi(s,a) - p_{\hat{T}}^\pi(s,a))c(s,a) \leq \sum_{s,a} p_{\hat{T}}^\pi(s,a) d_F(\hat{T}(s,a), T(s,a)) \quad (4.47)$$

where p_T^π and $p_{\hat{T}}^\pi$ are occupancy distributions under the true and learned models, and $d_F(\hat{T}(s,a), T(s,a))$ is the integral probability metric (IPM) between the learned transition model $\hat{T}$ and the true model T .

Since the conservative penalty might be overly strict, CAP dynamically adjusts the penalty using feedback from the environment. This adjustment is performed using a proportional-integral controller (PI), a simple variant of the PID controller [113], which tunes a penalty scale κ based on the observed discrepancy between the trust and the constraint threshold C by

$$\kappa_{t+1} = \kappa_t + \alpha(J_c(\pi_t) - C) \quad (4.48)$$

where α is the learning rate, and $J_c(\pi_t)$ is the observed cost of the policy π .

The CAP algorithm iterates between updating the transition model, optimizing the policy with the conservative cost penalty, collecting new data, and adjusting the penalty. This ensures that all intermediate policies during training remain feasible and safe.

The authors evaluate CAP on both tabular gridworld environments and high-dimensional continuous control tasks (e.g., Mujoco Half Cheetah⁽²⁾ based on Wawrzyński [114] and Car Racing⁽³⁾ based on Chris Campbell's Top-down car physics⁽⁴⁾). CAP was shown to achieve near-optimal performance while significantly reducing the number of unsafe actions during training [70]. It also demonstrated greater sample efficiency compared to model-free safe RL methods. The adaptive penalty mechanism effectively balanced exploration and safety, reducing the number of safety violations.

Srivastava *et al.* [77] address the challenge of mitigating negative side-effects with non-markovian nature, which depend on the entire sequence of actions (trajectory) rather than just individual state-action pairs.

To achieve this, the authors assume a CMDP with constraints on negative side-effect occurrences and propose a framework called *Controller-Assisted Safe Planning (CASP)*, which involves the following three steps:

⁽²⁾ https://gymnasium.farama.org/main/environments/mujoco/half_cheetah/

⁽³⁾ https://gymnasium.farama.org/main/environments/box2d/car_racing/

⁽⁴⁾ <https://www.iforce2d.net/b2dtut/top-down-car>

4 Systematic Literature Review

1. **Learning NSE Models:** The system first gathers data about negative side-effects through observed state-action trajectories. The data includes information about the severity of NSEs associated with these trajectories.
2. **Finite State Controllers:** The authors use finite state controllers (FSCs) and EM algorithms [115] to predict the severity of negative side-effects for given trajectories. FSCs are compact representations that summarize the history of state-action pairs to make predictions about future states and associated side-effects.
3. **Constrained Markov Decision Process:** The system integrates the learned FSC into a CMDP model. The CMDP uses the underlying MDP information about the FSC to plan actions while avoiding negative side-effects. The planning is performed using a variant of the dual linear programming method (dual LP) for MDPs [116].

The optimization problem is solved by integrating the FSC into the dual LP formulation [25] and defining a cross-product MDP [117], which considers both the environment states and FSC states. The resulting transition function of this cross-product MDP can be defined with

$$P(u_{t+1}, s_{t+1} | u_t, s_t, a_t) = T(s_t, a_t, s_{t+1}) \delta(u_{t+1} | u, \sigma = L(s_t, a_t, s_{t+1})) \quad (4.49)$$

where u is a node in the FSC, $L(\cdot)$ is the labeling function, and δ is the slack indicating how much the optimal expected value is allowed to deviate when the agent updates its policy.

The dual LP can then be solved by maximizing

$$\max_{\{y(\cdot)\}} \sum_{u, s, a} R(s_t, a_t, s_{t+1}) y(u_t, s_t, a_t) \quad (4.50)$$

where y is a occupancy measure denoting the total expected number of times the FSC state is u .

The empirical evaluation demonstrates that the CASP approach is effective in learning and mitigating both markovian and non-markovian NSEs. The results show that CASP can efficiently learn from limited historical data and outperform existing methods in avoiding NSEs. Specifically, CASP successfully avoids NSEs that other methods, such as those based on multi-objective MDPs, fail to handle due to their inability to deal with non-markovian dependencies.

4.4.3 Planning with Reduced or Incomplete Models

Saisubramanian *et al.* [48] aim to address the issue of negative side-effects that arise when planning with reduced models. Reduced models simplify planning by considering only a subset of possible outcomes for each state-action pair, but this simplification can lead to situations where the agent encounters states during execution that are not covered by the precomputed policy. Replanning in these states can be risky, especially if the state is unsafe for deliberation, such as a robot needing to replan while on a busy street. The challenge is to retain the computational efficiency of reduced models while minimizing these negative side effects.

The authors propose a planning approach that uses a portfolio of reduced models to minimize negative side-effects. This approach involves selectively improving the model fidelity in states that are identified as risky for replanning. The authors consider a Stochastic Shortest Path (SSP) problem defined by an MDP $M = (S, A, T, R, s_0, S_G)$, where s_0 is the initial (starting) state, and S_G is a set of goal states. The optimal policy π^* can be calculated using the optimal value function

$$V^*(s_t) = \min_{a'} Q^*(s_t, a') \quad (4.51)$$

where calculated using the optimal value function

$$Q^*(s_t, a_t) = R(s_t, a_t, s_{t+1}) + \sum_{s'} T(s_t, a_t, s_{t+1}) V^*(s_{t+1}) \quad (4.52)$$

denotes the optimal action-value function in the SSP of the MDP [118].

A reduced model of an SSP MDP is defined by altering the transition function such that only a subset of possible outcomes is considered for each action. This is denoted by the tuple $M' = (S, A, T', R, s_0, S_G)$, where T' is the altered transition function such that $\forall (s, a) \in S \times A, \theta'(s, a) \subseteq \theta(s, a)$. The function $\theta'(s_t, a_t) = \{s_{t+1} | T'(s_t, a_t, s_{t+1}) > 0\}$ defines the set of outcomes in the reduced model.

The authors introduce a portfolio-based approach where different outcome selection principles are applied depending on the state. In risky states, a more comprehensive outcome selection is used, while in safe states, a simpler reduced model is enough. For this, they introduce a *0/1 reduced model* (0/1 RM), which alternates between considering one outcome by ignoring stochasticity completely, and using the full model (considering all outcomes fully accounting for stochasticity) based on a heuristic reachability analysis.

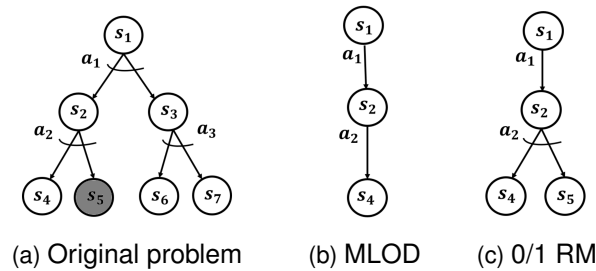


Figure 4.9: Comparison between the original problem, most likely outcome determinization (MLOD) and 0/1 RM approach [48].

Figure 4.9 shows the differences between the original problem Figure 4.9a, using the most likely outcome determinization (MLOD) by using a uniform outcome selection principle Figure 4.9b, and 0/1 RM

4 Systematic Literature Review

Figure 4.9c, where s_5 is a risky state with a negative side-effect. MLOD ignores the risky state of the original problem, while 0/1 RM uses the full model in this situation including s_5 and is able to account for the negative side-effect.

The proposed method was empirically tested on three different domains with an electric vehicle charging problem using real-world data [119], a racetrack problem [120], and a sailing domain [121]. The results showed that the 0/1 reduced model effectively minimized negative side effects without significantly increasing computational time. The approach achieved nearly optimal solutions, demonstrating its effectiveness in balancing computational efficiency and safety.

Low *et al.* [76] aim to enhance the safety of decision-making agents operating in uncertain environments, especially those modeled as CMDPs. They seek to mitigate negative side-effects that occur when agents operate in environments with incomplete models. Previous methods typically address safety by imposing constraints on the immediate state-action pairs (markovian constraints), but these may fail to account for long-term or complex safety issues. The authors propose a method to learn non-markovian safety patterns (i.e., safety concerns that depend on entire trajectories, not just single states or actions), and integrate this into a framework for safe MDP planning.

The problem is framed as a CMDP, where the objective is to maximize rewards while satisfying safety constraints. Instead of using predefined safety cost functions, the method learns from labeled trajectories that identify various categories of NSEs. The key CMDP objective is defined with

$$\max_{\theta} J_R(\pi_{\theta}) = \mathbb{E}_{\tau \sim \pi_{\theta}} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right] \quad (4.53)$$

$$\text{s.t. } J_{C_i}(\pi_{\theta}) = \mathbb{E}_{\tau \sim \pi_{\theta}} [C_i(\tau) \leq d_i] \quad \forall i = 1, 2, \dots, K \quad (4.54)$$

where $C_i(\tau)$ denotes the safety constraint to be learned from state-action pairs (s_i, a_i) .

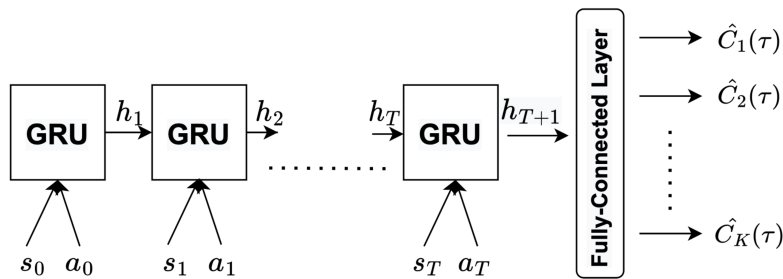


Figure 4.10: GRU network to predict negative side-effects subject to the defined CMDP objective based on state-action pairs in the trajectory [76].

The authors use Gated Recurrent Units (GRU) [122] to classify trajectories based on their NSE risk. Figure 4.10 shows the classifier predicting the likelihood of NSEs based on the sequence of states and actions in the trajectory. To solve the CMDP, a *Lagrangian relaxation method* [123], [124] is employed. The safety constraints are enforced by adjusting the Lagrange multipliers, which act as penalties for violating the constraints. The Lagrange function is defined with

$$\min_{\lambda \geq 0} \max_{\theta} L(\lambda, \theta) = J_R(\pi_{\theta}) - \sum_{i=0}^K \lambda_i [J_{\hat{C}_i}(\pi_{\theta}) - d_i] \quad (4.55)$$

4 Systematic Literature Review

and gradients are computed to update both, the policy

$$-\frac{\partial L}{\partial \lambda_i} = J_{\hat{C}_i}(\pi_\theta) - d_i = \mathbb{E}_{\tau \sim \pi_\theta} [\hat{C}_i(\tau) - d_i] \quad (4.56)$$

and Lagrange multipliers

$$\frac{\partial L}{\partial \theta} = \nabla_\theta J_R(\pi_\theta) - \sum_{i=0}^K \lambda_i \nabla_\theta J_{\hat{C}_i}(\pi_\theta) \quad (4.57)$$

$$= \nabla_\theta J_R(\pi_\theta) - \sum_{i=0}^K \lambda_i \nabla_\theta \mathbb{E}_{\tau \sim \pi_\theta} [\hat{C}_i(\tau)] \quad (4.58)$$

respectively to provide a local optimum of the problem.

Stochastic computation graph of $\hat{C}_i(\tau)$ using GRUs to classigy the likelihood of negative side-effects across a trajectory τ with $K = 3$ NSE classes. The GRU process sequences of states and actions, while the Lagrange multiplier method adjusts the policy π_θ to enforce safety constraints based on the classification of the trajectory.

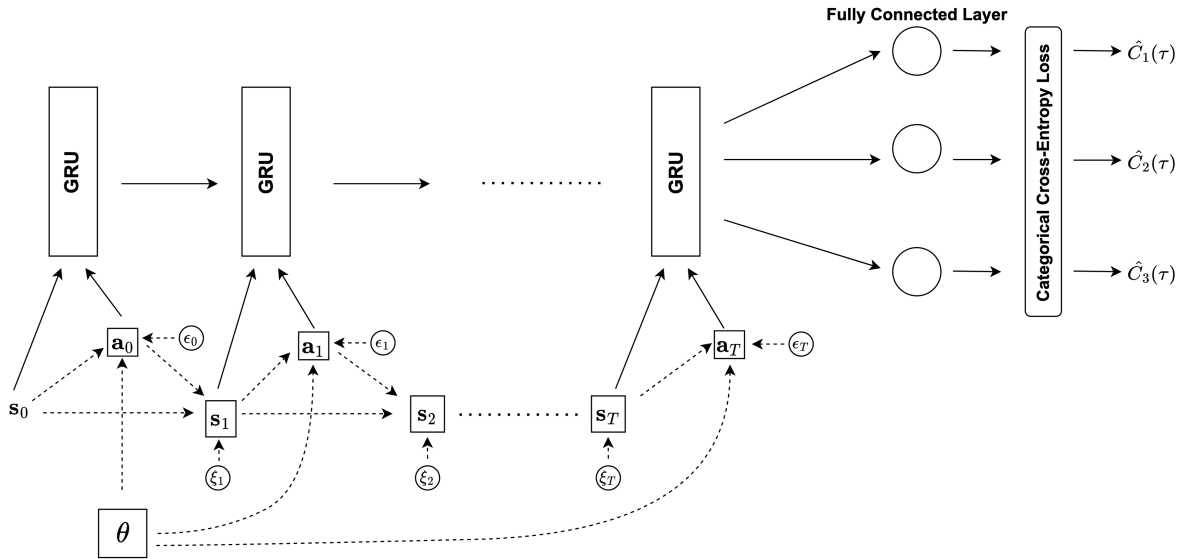


Figure 4.11: Stochastic computation graph of $\hat{C}_i(\tau)$ using GRUs to classigy the likelihood of negative side-effects across a trajectory τ with $K = 3$ NSE classes. The GRU process sequences of states and actions, while the Lagrange multiplier method adjusts the policy π_θ to enforce safety constraints based on the classification of the trajectory [76].

Figure 4.11 shows the combination of a GRU and the Lagrange multiplier method to classify negative side-effects based on state-action pairs of the trajectory.

The method was empirically tested in both discrete and continuous environments and successfully learned to satisfy non-markovian safety constraints while optimizing the agent's overall rewards. The approach was more scalable and effective than previous approaches that only handled markovian NSEs and the policy produced outperformed previous methods, especially in balancing between maximizing rewards while mitigation negative side-effects.

4.4.4 Symbolic Planning

Klassen *et al.* [60], [68] aim to address the issue of negative side-effects in sequential decision-making processes, particularly within the context of symbolic planning. Traditional planning approaches often focus on achieving specific objectives without considering the unintended consequences (side-effects) these actions may have on the environment or other agents. The problem is especially critical when multiple independent agents share an environment, as one agent's actions can inadvertently hinder the ability of others to achieve their goals. The authors seek to formalize the notion of side-effects and propose methods to minimize these negative impacts.

The authors formalize the problem as planning problem in the context of a state-transition system [125] with

$$\mathcal{P} = \langle \Sigma, s_0, S_G \rangle \quad (4.59)$$

where $\Sigma = \langle S, A, \delta \rangle$ is a state-transition system with $\delta : S \times A \mapsto S$ as a partial function that defines state transitions, $s_0 \in S$ is the initial (starting) state, and $S_G \subseteq S$ is the set of goal states.

A sequence of actions $\pi \in A^*$ is considered a plan for $\mathcal{P}$ if and only if $\sigma(s_0, \pi) \in S_G$ meaning the execution of π from s_0 leads to a state in the goal set S_G .

The authors use the *Stanford Research Institute Problem Solver (STRIPS)* [126] formalism to define the planning problem as tuple $\langle F, I, A, G \rangle$, where F is a finite set of propositional symbols (fluents), $I \subseteq F$ represents the initial state, A is a finite set of actions, and $G \subseteq F$ represents the goal. Each action $a \in A$ is described by

- **Add list:** $\text{Add}(a)$ - the set of fluents added by the action,
- **Delete list:** $\text{Del}(a)$ - the set of fluents removed by the action, and
- **Precondition list:** $\text{Pre}(a)$ - the set of fluents that must be true for the action to be executed.

The state-transition system δ in this case is defined such that

$$\delta(s, a) = \begin{cases} (s \setminus \text{Del}(a)) \cup \text{Add}(a) & \text{if } \text{Pre}(a) \subseteq s \\ \text{undefined} & \text{otherwise} \end{cases} \quad (4.60)$$

To minimize negative side-effects, the authors introduce the concept of a change-minimizing plan. Given a distance function $d : S \times S \mapsto [0, \infty)$, which measures the change between two states. A plan π is change-minimizing if

$$\forall \pi' \in A^*, d(\delta(s_0, \pi'), s_0) \geq d(\delta(s_0, \pi), s_0) . \quad (4.61)$$

This means that π leaves the world as close as possible to its initial state after reaching the goal to minimize negative side-effects in the context of not taking actions leading to drastically changes to the environment compared to the starting state.

The experimental results validate that the proposed compilations can effectively generate plans that minimize side effects. The authors tested their methods on several domains, showing that their approach can produce plans that are less likely to interfere with other agents' goals. This capability is crucial in multi-agent environments and other applications where safety and ethical considerations are critical.

4.4.5 Environment Shaping

Saisubramanian *et al.* [63], [71] introduce an approach using environment shaping by allowing human operators to reconfigure the environment. The agent optimizes its task, potentially causing negative side-effects, while a human modifies the environment to mitigate these side effects without affecting the agent's task completion. The setup involves an actor agent and an environment designer agent. The actor operates in a configurable environment described by a configuration file E , optimizing its primary objective o_P using an MDP $\tilde{M}$. The designer evaluates the impact of NSE and modifies the environment as needed. Modifications, such as moving objects or removing obstacles, are finite and specified in the problem setup. Key assumptions are that the actor's start and goal conditions remain fixed and that modifications are tentatively applied to ensure they do not hinder the agent's task performance.

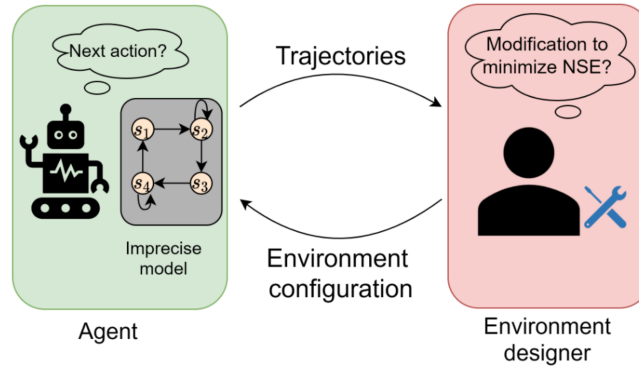


Figure 4.12: Cyclic relationship between an agent and an environment designer to mitigate negative side-effects using environment shaping. [63], [71].

The authors consider an actor-designer framework for negative side-effect mitigation (AD-NSE) defined as $\langle E_0, \mathcal{E}, \tilde{M}, M_d, \delta_A, \delta_D \rangle$, where:

- E_0 is the initial environment configuration,
- $\mathcal{E}$ is a finite set of possible reconfigurations of E_0 ,
- $\tilde{M}$ represents the actor's MDP,
- $M_d = \langle \Omega, \Psi, C, R_N \rangle$ is the model of the designer with
 - Ω as a set of valid modifications, including $\emptyset$ for no changes,
 - $\Psi : \mathcal{E} \times \Omega \mapsto \mathcal{E}$ gives the new environment modification after applying a modification $\omega \in \Omega$
 - $C : \mathcal{E} \times \Omega \mapsto \mathbb{R}$ is the cost function for doing modifications $C(E, \omega)$, with $C(E, \emptyset) = 0, \forall E \in \mathcal{E}$
 - R_N measures the penalty for NSE, with $R_N(\pi, E)$ denoting the penalty associated with policy π in environment $E \in \mathcal{E}$
- $\delta_A \geq 0$ is the slack for the actor representing the maximum allowed deviation from the optimal policy for o_P when recomputing its policy in a modified environment
- $\delta_D \geq 0$ represents the designer's tolerance threshold for NSE.

4 Systematic Literature Review

Figure 4.12 shows a simplification of the actor-designer framework. The actor's goal is to compute a policy π that maximizes its expected reward $\max_{\pi \in \Pi} V_p^\pi(\tilde{s}_0|E)$, where

$$V_p^\pi(\tilde{s}_0) = R(\tilde{s}, \pi(\tilde{s})) + \sum_{\tilde{s}' \in \tilde{S}} T(\tilde{s}, \pi(\tilde{s}), \tilde{s}') \cdot V_p^\pi(\tilde{s}'), \forall \tilde{s} \in \tilde{S} \quad (4.62)$$

with $\tilde{s}_0$ being the initial (start) state in environment E .

When the environment is modified, the actor may need to recompute its policy, which might lead to a longer path to its goal. The slack δ_A allows deviation from the optimal expected reward in the initial environment E_0 , where

$$V_p^*(\tilde{s}_0|E_0) - V_p^{\pi'}(\tilde{s}_0|E') \leq \delta_A \quad (4.63)$$

and π' is the policy in the modified environment E' .

The designer estimates the NSE penalty $R_N(\pi, R)$ and if the penalty exceeds the tolerance threshold δ_D , the environment is reconfigured. The designer selects a modification ω to maximize its utility $\max_{\omega \in \Omega} U_\pi(\omega)$, where

$$U_\pi(\omega) = \underbrace{(R_N(\pi, E) - R_N(\pi, \psi(E, \omega)))}_{\text{reduction in penalty}} - C(E, \omega) \quad (4.64)$$

to balance the penalty of the negative side-effect against the cost of the modification. The cost is measured in the same unit as the penalty and may gradually reduced over episodes.

The AD-NSE framework ensures that the actor's performance remains bounded while minimizing negative side-effects through strategic environment shaping. Instead of sharing the complete policy, which is impractical for large problems, the actor provides demonstrations $\mathcal{D} = \{\tau_1, \tau_2, \dots, \tau_n\}$ of its optimal policy for o_P . These demonstrations help the designer understand the actor's policy and estimate its negative side-effects. The designer uses this information to shape the environment effectively, as more and diverse demonstrations improve the designer's ability to accurately choose appropriate modifications.

4.4.6 Safety Wrappers

Burden *et al.* [66] propose a method that can reduce the occurrence of negative side-effects by leveraging confidence measures to trigger safer behavior when the agent's certainty in its actions is low. The core of the proposed solution lies in the development of *safety wrappers* for reinforcement learning agents. These wrappers modify the agent's behavior based on its confidence in its actions, ensuring safe operation. The safety wrapper consists of a base policy π_{base} as the default policy for interacting with the environment, and a safety policy π_{safe} as safer alternative that the agent switches to when it detects a drop in confidence.

These two components are encapsulated as options in the framework of the agent. The base option $\mathcal{O}_{base} = (\mathcal{I}_{base}, \pi_{base}, \mathcal{B}_{base})$ executes the base policy when the agent is initialized, and the safety option $\mathcal{O}_{safe} = (\mathcal{I}_{safe}, \pi_{safe}, \mathcal{B}_{safe})$ triggers the safety policy when the agent's confidence is low. $\mathcal{I}$ and $\mathcal{B}$ serve as initiation and termination sets of the corresponding option respectively.

The agent's confidence in this framework can be measured in two ways, depending on if we want to change the orientation on the reward or the goal. The reward-oriented confidence measure is based on the agent's own network output. For The Proximal Policy Optimization (PPO) [38] algorithm, the confidence is computed as the inverse of the entropy of the policy with

$$C(s) = 1 - H(s, \pi) = 1 - \sum_{a \in A} \pi(a|s) \log_{|A|} \pi(a|s) \quad (4.65)$$

where $H(s, \pi)$ is the entropy of the policy π at state s . For DQN [30], a similar approach is applied using the softmax of the Q -values. The goal-oriented confidence is estimated using a logistic regression model trained on the states encountered by the agent during previous episodes. The model predicts the likelihood of success for each state, which serves as the confidence measure.

The safety wrapper monitors the confidence value $C(s)$, and compares it against a predefined threshold δ . If $C(s) < \delta$, the base option $\mathcal{O}_{base}$ terminates, and the safety option $\mathcal{O}_{safe}$ activates. The initiation $\mathcal{I}$ and termination $\mathcal{B}$ sets are defined as

$$\mathcal{I}_{safe} = \mathcal{B}_{base} = \{s \mid C(s) < \delta\} \quad (4.66)$$

and

$$\mathcal{B}_{safe} = \mathcal{I}_{base} = S \setminus \mathcal{I}_{safe} \quad (4.67)$$

where $\mathcal{O}_{safe}$ is the same set as $\mathcal{O}_{base}$ and vice versa. This allows to store and compare the initiation and termination conditions of the inverse option.

This means the safety policy takes over when the agent's confidence drops below the threshold δ , and it returns control to the base policy when the confidence is restored. Furthermore, the modular approach using a safety wrapper around the agent allows a designer to construct π_{safe} freely. The authors consider a subset of safe actions (a no-op action $\emptyset \in A$ for not moving) as well as domain knowledge providing a hand-crafted policy for their experiments.

The experiments showed that wrapping RL agents like DQN [30] and PPO [38] with these safety mechanisms significantly reduced the occurrence of negative side effects while maintaining competitive performance. In the Vase World environment [13], the wrapped agents demonstrated a marked improvement in safety scores, with only a slight reduction in reward compared to unwrapped agents. Similarly, in the more complex SafeLife environment [18], safety-wrapped agents expanded the Pareto frontier, achieving better trade-offs between performance and safety.

4.5 Inverse Reinforcement Learning

Inverse Reinforcement Learning (IRL) [40] is a technique where agents learn underlying reward functions by observing the behavior of experts, rather than being provided with explicit rewards. In the context of avoiding negative side effects, IRL is particularly useful for learning safe and robust policies when the explicit specification of rewards is incomplete, ambiguous, or difficult to define. By leveraging demonstrations, IRL helps the agent infer the preferences and constraints that guide expert behavior, allowing it to avoid harmful actions and prioritize safety.

This section explores the use of IRL to mitigate negative side effects, focusing on how agents can learn safe policies from expert demonstrations and implicit human preferences. It covers approaches that ensure robust performance, even in uncertain environments with incomplete reward information, and methods that infer human preferences from observations to prevent harmful behaviors.

4.5.1 MDP Solver

Huang *et al.* [44] introduce a method that enables agents to learn a robust policy that maximizes rewards while ensuring safety, even in the presence of incomplete or inaccurate information. The key is to leverage demonstrations from experts to guide the learning process. The authors propose a $\max \min$ learning framework [127], [128], related to inverse reinforcement learning (IRL) [40] and apprenticeship learning [129] designed to learn policies that are robust to worst-case scenarios in reward specifications.

The approach involves an exact MDP solving algorithm based on the ellipsoid method and a more practical approach using Follow-the-Perturbed-Leader (FPL) [130]. The authors consider a finite MDP $M = (S, A, p, R, \gamma, \chi)$, where the reward function is assumed to be linear in some feature space $\phi : S \mapsto [0, 1]^k$, expressed as $R(s) = w \cdot \phi(s)$, $\forall s \in S$ and $w \in \mathbb{R}^k$ represents the reward weights. The goal is to solve the $\max \min$ problem

$$\max_{\mu \in P_F} \min_{w \in P_R} w^T \mu$$

where $\mu = \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t \phi(s_t) | M, \pi \right]$ is the feature expectation under the agent's policy π , P_F is the polytope containing the representations of all policies in the feature space, and P_R is the set of reward functions consistent with the expert's knowledge.

The Problem is formulated as a linear program (LP)

$$\begin{aligned} \max \quad & z \\ \text{subject to} \quad & z \leq \mu \cdot w, \quad \forall w \in P_R, \\ & \mu \in P_F \end{aligned} \tag{4.68}$$

and the authors use separation oracles to optimize linear functions over convex sets:

- **Separation Oracle for P_F :** This oracle checks whether a given μ_0 lies in P_F by solving the MDP with the reward function $R(\cdot) = w \cdot \phi(\cdot)$ using the MDP solver.
- **Separation Oracle for P_R :** The oracle verifies whether a given reward weight w_0 is consistent with the expert's demonstration by checking if it satisfies the condition $\mu_E \cdot w_0 \geq \mu \cdot w_0 - \epsilon$, $\forall \mu \in P_F$.

4 Systematic Literature Review

An ellipsoid-based separation oracle for LP feasibility then checks whether a candidate pair (μ_0, z_0) satisfies the constraints of the linear program.

While this problem may be computationally expensive to solve, the authors propose a FPL method as an online learning algorithm that approximates the $\max \min$ policy through iterative updates. This algorithm ensures that after T iterations, the policy is within $O(1/\sqrt{T})$ of the true $\max \min$ policy.

The proposed methods demonstrate the ability to learn robust policies that avoid negative side-effects and are resilient to reward misspecification. The experimental results, conducted in both discrete gridworlds and continuous environments like cartpole provided by OpenAI Gym [131], show that the $\max \min$ policies outperform baseline methods by ensuring safety even in complex scenarios with uncertain or incomplete reward information.

4.5.2 Implicit Preference Information

Shah *et al.* [49] proposes an approach to mitigate negative side-effects with the goal to infer human preferences, particularly those related to avoiding side effects, from the initial state of the environment in which the agent operates.

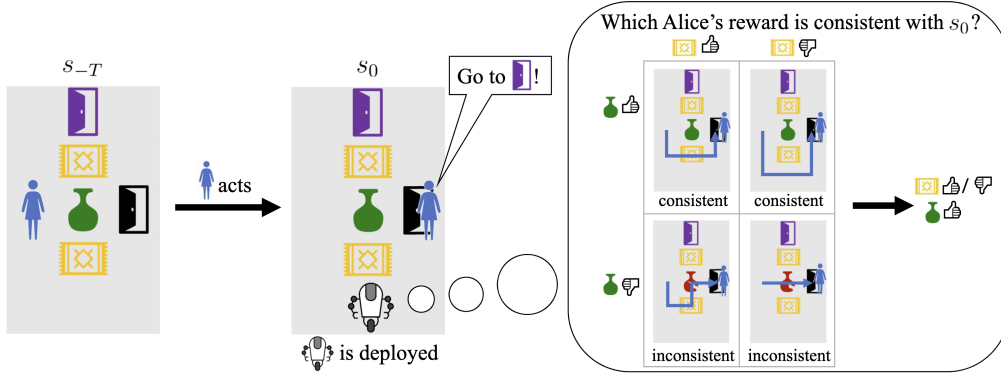


Figure 4.13: Illustration of learning preferences from an initial (starting) state using Reward Learning by Simulating the Past (RLSP) [49].

The authors introduce an algorithm called *Reward Learning by Simulating the Past* (RLSP) to infer human preferences from the initial state of the environment. Figure 4.13 illustrates the learning process, where Alice (human) moves in an environment with a vase and carpets, aiming to reach a specific goal. The robot (agent) observes the environment's current state s_0 , where the vase remains intact, after Alice has acted from a previous state s_{-T} . The agent infers that intact vases suggest Alice's preference to avoid breaking them, even though no explicit instruction was given. The agent considers different possible reward functions for Alice, identifying those that are consistent with the observed state (vase intact) and those that are not (vase broken).

The key insight of RLSP is that the environment's initial state is already optimized according to human preferences. They use Maximum Causal Entropy Inverse Reinforcement Learning (MCEIRL) [43] to model human behavior and infer a reward function from the initial state. Since the authors only have a single end state s_0 of the trajectory $\tau_0 = (s_{-T}, a_{-T}, \dots, s_0, a_0)$, they update the likelihood of observing s_0 in the trajectory τ_0 given the reward parameters θ by marginalizing over all other variables with

$$p(s_0|\theta) = \sum_{s_{-T}, a_{-T}, \dots, s_0, a_0} p(\tau_0|\theta)$$

where $p(\tau_0|\theta)$ is the likelihood of the trajectory τ_0 as defined in MCEIRL [43]. The idea behind this algorithm is to maximize the log-likelihood of the initial (starting) state s_0 to find the reward function that best explains it.

The results demonstrate that RLSP can successfully infer preferences that lead to avoiding side effects and achieving tasks in various environments. The inferred rewards align well with human preferences as observed in different proof-of-concept environments, showing that the initial state contains valuable information about implicit preferences. The method effectively penalizes harmful side effects while allowing positive ones.

4 Systematic Literature Review

[54] [54] build up from their previous work with RLSP [49] and introduce a formalism for learning from the state of the world [54]. The key contributions involve three components to implicitly use preference information for avoiding negative side-effects:

1. **Learning from Partial Observability:** Extending the formalism to handle environments where neither the human nor the agent can observe the entire state of the world at each time step.
2. **Assistance Formulation:** Integrating reward learning and control by solving assistance problems, where the agent not only learns a reward function from the human but also actively seeks further information to refine its understanding.
3. **Reinforcement Learning and Planning:** Algorithms are developed to infer a reward function (θ) from the initial state of the world (s_0) observed by the agent upon deployment. The agent then follows a policy that balances this inferred reward with a specified reward.

By combining these three components, the authors introduce Deep RLSP, which extends RLSP to high-dimensional and continuous environments where traditional methods like tabular RLSP are not applicable. The main idea is to scale RLSP to more complex environments where the state space is continuous, the dynamics are not fully known, and there is no predefined feature function.

Deep RLSP uses a Variational Autoencoder (VAE) [132] to learn the feature function f . The VAE is trained on environment interactions and encodes states into a latent feature representation that can be used to learn a reward function [54]. Furthermore, the VAE is used to train an inverse dynamics model $\mathcal{T}^{-1}$, which is a neural network that predicts the previous state s_{t-1} given the current state s_t and action a_t . This model helps to simulate the past trajectories τ that lead to the current state. The policy of the human model π_H is learned using Soft Actor-Critic (SAC)[39], which optimizes the policy based on the inferred reward function. This policy is continuously updated throughout the training process. The gradients are computed by simulating backwards from the observed state s_0 and then rolling out forward from the inferred initial state s_{-T} [54]. The difference between the backward and forward feature counts is used to update the reward parameters θ . Deep RLSP starts with a short horizon $\mathcal{T}$ and gradually increases it as the training progresses, making it easier to maintain consistency between the simulated past and the observed states.

Deep RLSP [54] has been shown to perform well in environments such as MuJoCo's Inverted Pendulum⁽⁵⁾ based on the work by Barto *et al.* [133] and Half Cheetah⁽⁶⁾ based on Wawrzyński [114] where it is capable of learning reasonable behaviors from a single state. Although it does not achieve expert-level performance, it can still infer useful skills like balancing and jumping with varying degrees of accuracy depending on the number of states provided.

⁽⁵⁾ https://gymnasium.farama.org/main/environments/mujoco/inverted_pendulum/

⁽⁶⁾ https://gymnasium.farama.org/environments/mujoco/half_cheetah/

5 Results

For the systematic literature review on mitigating negative side-effects in reinforcement learning, we classify the identified studies based on specific criteria, including a classification of agents based on the approach used and what assumptions lie under their algorithms. Furthermore, we analyze the results to detect gaps in the literature and elaborate, on how this research area can be advanced in the future. This taxonomies facilitates a structured understanding of the existing strategies and their effectiveness in promoting safe RL behaviors.

5.1 Classification of Agents

To classify the assumptions, research fellows in the [Topic-wise](#) field took to design safe RL algorithms, we defined following taxonomies to classify retrieved studies:

Fully Observable State The agent has complete access to all relevant information about the current state of the environment. In other words, the agent can directly observe all aspects of the environment that are necessary for decision making.

No-Op Action The action space of the agent contains a no-op action (e.g. “standing still”), where the action of the agent has no influence on the environment for one time step. They are often included in environments to allow the agent to choose not to take any action.

Known Dynamics The agent has complete knowledge or a model of how the environment behaves over time. In other words, the agent has information about the internal dynamics of the Markov Decision process (MDP) and knows, how the environment will evolve in response to its actions.

Linear Feature Reward The reward an agent receives is linearly dependent from some sort of features or attributes of the current state. Most often, each feature is associated with a weight, and the reward is calculated as the dot product of the feature vector and the weight vector.

Action-Independent Reward The agent receives rewards that are not influenced by specific actions. Instead, the reward depends only on the current state of the environment. This can simplify the learning process since the agent does not need to consider the effects of its actions on the reward.

Known Past States The agent has access to a history of past states that is has observed or visited during interaction with the environment. Having knowledge of past states can be useful for making decisions and learning patterns of trends in the environment.

5 Results

5.1.1 Agents without safety mechanisms

While many reinforcement learning (RL) agents are designed with safety mechanisms to mitigate negative side-effects, several approaches explore agent behavior in environments where such mechanisms are absent. These agents often prioritize reward maximization without explicit regard for the safety of their actions, leading to unintended consequences or disruptions in the environment. The study of agents without safety mechanisms highlights the importance of designing constraints and regularizations that ensure safe and reliable behavior, particularly in complex environments where agents interact with dynamic or stochastic elements.

Most of the reviewed studies in this category operate under fully observable states, where agents have access to all relevant information about the environment. Additionally, many of these approaches rely on a no-op action, which allows agents to choose to remain inactive, providing a baseline for assessing the impact of their actions. However, unlike other frameworks that use linear feature rewards or track the impact of actions over time, agents without safety mechanisms do not typically utilize these features, leading to a focus on short-term reward optimization without long-term safety considerations.

Study	Fully Observable State	No-Op Action	Known Dynamics	Linear Feature Reward	Action-Independent Reward	Known Past States
Leike <i>et al.</i> [15]	✓	✓	✓	✗	✗	✗
Waisi [57]	✓	✗	✓	✗	✗	✗
Wainwright <i>et al.</i> [18]	✓	✓	✗	✗	✗	✗
Burden <i>et al.</i> [59]	✓	✓	✗	✗	✗	✗
Hoang <i>et al.</i> [67]	✗	✗	✗	✗	✗	✗
Amodei <i>et al.</i> [6]	—	—	—	—	—	—

Table 5.1: Taxonomy of agents without safety mechanisms based on the proposed classification criteria. Retrieved studies show a strong dependence on fully observable states and a no-op action in the agent's action space. The reward is not dependent on (linear) features and only dependent on the actions taken by the agents. Information about past states are not available in the learning algorithms. Amodei *et al.* [6] focuses only on the theoretical foundations on avoiding negative side-effects without proposing a new algorithm or doing empirical experiments, therefore no classification was done.

Table 5.1 summarizes the reviewed studies, illustrating the common reliance on fully observable states and the presence of no-op actions. The following subsections describe specific methods used in environments without safety mechanisms and discuss the challenges and limitations of these approaches, especially in balancing task performance with the avoidance of negative side-effects.

5 Results

Tabular Solution Methods

Waisi [57] explored the challenge of avoiding negative side-effects without embedding ad-hoc constraints into the agents' objective functions. By employing tabular solution methods such as Q-learning [29] and Sarsa [28], the study compared agent performances in two distinct environments: a safety setting where agents were unaware of penalties for side effects, and a standard setting where such penalties were explicitly enforced. The environments utilized were variants of the Sokoban game [15] and the Cliff-walking environment [16], both of which offer fully observable states and known dynamics. The findings revealed that agents trained with direct penalties for side effects demonstrated superior performance and more optimal behavior, whereas those in the safety setting struggled to avoid unintended consequences. This underscores the critical role of explicit safety mechanisms in enhancing agent reliability, albeit with potential trade-offs in overall performance.

Deep Reinforcement Learning

Leike *et al.* [15] introduced the AI Safety Gridworlds, a suite of environments designed to evaluate agents' adherence to safety constraints. These gridworlds are characterized by fully observable states, no-op actions, and known dynamics, allowing agents to navigate safely through tasks such as safe interruptibility and robust exploration. Utilizing advanced deep reinforcement learning techniques like Advantage Actor-Critic (A2C) [36] and Rainbow [17], the study demonstrated that agents lacking inherent safety mechanisms often prioritize visible rewards over safety, leading to behaviors that exploit reward function loopholes or cause unintended disruptions. This highlights the vulnerability of DRL agents to safety breaches in complex environments, emphasizing the need for integrated safety considerations in their design.

Wainwright *et al.* [18] developed SafeLife, a procedurally generated gridworld environment aimed at benchmarking RL agents' ability to avoid negative side-effects while accomplishing their goals. SafeLife incorporates fully observable states and no-op actions, presenting dynamic and stochastic elements inspired by cellular automata. Agents trained using Proximal Policy Optimization (PPO) [38] were evaluated based on their task performance and the extent of environmental disruptions they caused, quantified using the earth mover's distance (EMD) [80]. The results indicated that while agents could effectively complete primary tasks, they frequently induced significant side-effects, underscoring the challenge of aligning agent objectives with environmental safety without explicit constraints.

Burden *et al.* [59] extended the evaluation of RL agents within the SafeLife environment [18] by analyzing side-effects produced by different DRL algorithms, including Deep Q-Networks (DQN) [30] and Proximal Policy Optimization (PPO) [38]. The study focused on the append-still task, assessing agents' ability to create stable structures while minimizing disruptions. Findings revealed that DQN agents, despite achieving higher pass rates and rewards with extensive training, did not significantly reduce side-effects. Conversely, PPO agents exhibited a nuanced relationship between reward optimization and side-effect generation, with moderate rewards correlating with higher side-effect scores. Additionally, increased exploration and longer episode lengths were associated with greater environmental disruptions. This research highlights the persistent difficulty in balancing performance and safety in DRL agents, advocating for more sophisticated safety mechanisms and benchmarking environments to better capture and mitigate unintended consequences.

5 Results

Text-based Reinforcement Learning

In the context of reinforcement learning (RL), much of the research on avoiding negative side-effects has focused on physical or simulated environments. However, Hoang *et al.* [67] introduced a new approach by exploring the problem of safety in text-based reinforcement learning, a domain where agents interact with their environment using natural language rather than direct sensory inputs or physical actions. This work falls under the category of Agents without safety mechanisms but does not fit into any of the predefined classification criteria (i.e., it operates in environments where the agent lacks full observability, no-op actions, known dynamics, linear feature rewards, action-independent rewards, or access to known past states). As such, this approach stands as a distinct case in reinforcement learning research, addressing unique challenges related to text-based interaction.

The authors highlighted the lack of benchmarks that unify these two domains, motivating the development of their proposed solution with the *Safety-Constrained Environments for Reinforcement Learning (SCERL)* benchmark. SCERL is a suite of 20 games adapted from TextWorld [81], designed to simulate real-world safety challenges, such as managing fire hazards or chemical risks. Each game in the suite includes explicit safety constraints, such as ensuring that potentially dangerous items (e.g., candles or chemicals) are handled properly to avoid hazards.

The safety constraints in SCERL are enforced through a system of penalties. These penalties can either be soft penalties (which do not terminate the game but decrease the agent's score) or terminating penalties (which immediately end the game if a severe violation occurs). This dual system allows agents to gradually learn safe behavior without being prematurely penalized for minor infractions, while still imposing strict consequences for critical safety violations.

Five Main Pillars of Safety in Reinforcement Learning

The research of Amodei *et al.* [6] and Raji *et al.* [7] emphasises the importance of designing RL agents that can operate safely without causing unintended or harmful behavior due to design flaws. The discussion identifies five main pillars of AI safety in reinforcement learning, which is further outlined [chapter 1](#).

The thesis concentrates on the first pillar, avoiding negative side-effects, which is a crucial aspect of ensuring that RL agents do not cause harm while carrying out their tasks. This particular safety problem, although sometimes conflated with other issues, forms its own distinct research area. Addressing negative side-effects is fundamental to ensuring the safe and reliable deployment of RL agents in real-world scenarios.

5 Results

5.1.2 Reward Regularization

Reward regularization is a key approach used to guide agents in reinforcement learning by augmenting their reward functions to minimize negative side-effects. By incorporating auxiliary reward components, agents are encouraged to consider not just the primary task performance but also the broader consequences of their actions, such as the environment, or other methods regularizing the main reward. The central idea of reward regularization is to penalize harmful or undesirable behaviors while rewarding safe and aligned actions, helping to avoid issues like reward hacking or unintended side-effects.

Most methods in this category assume fully observable states, allowing the agent to observe all relevant information needed to make decisions. Known dynamics are often utilized, enabling the agent to accurately predict the outcomes of its actions. However, unlike many other approaches, these methods do not depend on linear feature rewards or action-independent rewards, focusing instead on the contextual impact of the agent's decisions. Some approaches further incorporate past state information, helping the agent track the cumulative effects of its actions over time.

Study	Fully Observable State	No-Op Action	Known Dynamics	Linear Feature Reward	Action-Independent Reward	Known Past States
Alizadeh Alamdari <i>et al.</i> [58]	✓	✗	✓	✗	✗	✗
Alizadeh Alamdari <i>et al.</i> [65]	✓	✗	✓	✗	✗	✓
Klassen <i>et al.</i> [69]	✗	✗	✓	✗	✗	✓
Klassen <i>et al.</i> [74]	✗	✗	✓	✗	✗	✓
Laidlaw <i>et al.</i> [75]	✓	✗	✗	✗	✗	✗

Table 5.2: Taxonomy of agents using reward regularization based on the proposed classification criteria. All agents rely on a fully observable state and known dynamics of the environment. The reward is only dependent on the actions take by the agents and not on some sort of (linear) features.

Table 5.2 summarizes the identified works, highlighting the common use of fully observable states and known dynamics. The following subsections delve into specific methods of reward regularization, discussing how they balance task performance with the avoidance of negative side-effects and other unintended behaviors.

5 Results

Avoiding Negative Side Effects by Considering Others

Alizadeh Alamdari *et al.* [58] introduced a reward regularization approach that incentivizes agents to consider the impact of their actions on the future wellbeing and agency of others. This method operates under the assumption that the agent has access to a fully observable state, known dynamics, and known past states (considering their approach penalizing negative change from the starting state). By augmenting the agent's reward function with an auxiliary component, the agent's actions are influenced not only by the primary task but also by their potential side effects on others. This is achieved by defining a distribution over possible value functions that capture uncertainty regarding what is beneficial for others.

The authors propose three variants of the augmented reward function:

1. **Expected Future Return:** This formulation considers the expected value of the next state based on a distribution of value functions, thereby allowing the agent to consider a range of possible outcomes rather than a single trajectory. This approach generalizes earlier methods, such as those proposed by Krakovna et al. [50], by allowing the agent to anticipate the future impact of its actions.
2. **Worst-Case Future Return:** This variant focuses on minimizing the worst possible outcome among all potential value functions, which encourages the agent to avoid actions that could lead to the most negative side-effects. This is particularly useful in scenarios where avoiding the worst outcomes is more critical than maximizing the expected value.
3. **Penalize Negative Change:** This approach compares the value of the next state to the initial state and penalizes the agent for reducing expected future returns, without rewarding it for increasing them. This variant is designed to prevent the agent from causing harm without incentivizing unintended positive side-effects.

Building on this foundation, Alizadeh Alamdari *et al.* [65] expanded their work by introducing additional variants that differentiate between agents and consider the options available to them. For example, they proposed an augmented reward function that treats agents differently by considering their individual value functions and applying fairness principles, such as the maximin social welfare function. Another variant incorporates the concept of options, where the reward function accounts for the initiation and execution of predefined options, reflecting the agents' specific skills or tasks.

5 Results

Avoiding Epistemic Side Effects

Klassen *et al.* [69], [74] extended the work of Alizadeh Alamdari *et al.* [58], [65] by focusing on avoiding epistemic side effects, which are unintended changes to an agent’s knowledge or beliefs caused by its actions. This approach is particularly relevant in environments where agents operate alongside humans, who may have limited access to information (i.e., partially observable environments). The objective is to prevent agents from causing changes in the human’s knowledge or beliefs that could lead to harmful outcomes.

In their proposed framework, the agent operates with a human in a partially observable Markov decision process (POMDP) with the same state space and has access to a fully observable state space with known dynamics. The agent’s reward function is augmented to account for the human’s expected return, thereby discouraging both physical and epistemic side effects. The augmented reward is defined by an auxiliary function that integrates the expected return of the human’s history-state value function, which considers the human’s past observations and actions.

Despite the promising nature of this approach, it has limitations. For instance, the model does not explicitly represent the human’s beliefs, which can be crucial for addressing tasks with epistemic goals, such as learning or inferring the location of an object. Additionally, since the agent only accounts for the expected return of the human, it may not directly penalize the formation of false beliefs, which could be critical in certain applications.

Occupancy-Regularized Policy Optimization

Laidlaw *et al.* [75] proposed a novel approach called Occupancy-Regularized Policy Optimization (ORPO), which focuses on regularizing an agent’s behavior based on occupancy measures. This method is categorized under Reward Regularization and applies in scenarios where the agent operates in a Fully Observable State, meaning the agent has complete access to all relevant information about the environment at any given time.

The core idea behind ORPO is to prevent negative side-effects and reward hacking by regularizing the distribution of state-action pairs visited by the agent, known as occupancy measures. These occupancy measures provide a broader view of the agent’s behavior compared to focusing solely on the actions it takes. The regularization is achieved by minimizing the difference between the occupancy measures of the agent’s learned policy and a predefined safe policy. This helps ensure that the learned policy behaves similarly to the safe policy, reducing the chances of negative side-effects or unintended behaviors that arise from reward hacking.

Unlike traditional regularization methods that rely on comparing action distributions, ORPO uses occupancy measures, which provide a more reliable indicator of the agent’s overall behavior in the environment. The method ensures that the learned policy does not deviate too far from a safe baseline, thereby encouraging safer decision-making over time.

5 Results

5.1.3 Impact Regularization

Impact regularization focuses on mitigating negative side-effects by discouraging agents from making irreversible or disruptive changes to their environment. This approach introduces penalties for actions that deviate too much from a baseline, thereby minimizing unintended side-effects and ensuring that the agent can achieve its goals while preserving the environment’s integrity. Agents using impact regularization often balance task performance with environmental impact, ensuring that changes caused by their actions are minimal and, where possible, reversible.

Most methods in this area operate in fully observable environments, allowing agents to assess the complete state of the system before making decisions. Many also incorporate a no-op action, which serves as a neutral choice for the agent, providing a baseline to compare the impact of its actions. While some approaches rely on known dynamics to model how the environment evolves, others focus on preserving the agent’s ability to pursue auxiliary or future goals. Linear feature rewards are less commonly used in these approaches, with most methods focusing on dynamic, multi-objective trade-offs between the agent’s primary task and minimizing its long-term impact.

Study	Fully Observable State	No-Op Action	Known Dynamics	Linear Feature Reward	Action-Independent Reward	Known Past States
Krakovna <i>et al.</i> [47]	✓	✓	✓	✗	✗	✓
Turner <i>et al.</i> [56]	✓	✓	✓	✗	✗	✗
Turner <i>et al.</i> [55]	✓	✓	✓	✗	✗	✗
Krakovna <i>et al.</i> [50]	✓	✗	✓	✓	✗	✗
Miret <i>et al.</i> [51]	✓	✗	✗	✗	✗	✗
Vamplew <i>et al.</i> [64]	✓	✗	✗	✗	✗	✗
Turner <i>et al.</i> [72]	✓	✓	✓	✗	✗	✗
Lindner <i>et al.</i> [61]	—	—	—	—	—	—
Griffin <i>et al.</i> [21]	✓	✓	✗	✗	✗	✗
Turner [20]	✓	✓	✗	✗	✗	✗
Eresheim <i>et al.</i> [73]	✓	✗	✓	✗	✗	✗

Table 5.3: Taxonomy of agents using impact regularization based on the proposed classification criteria. All agents rely on a fully observable state and a no-op action in the agent’s action space builds an often used assumption. Lindner *et al.* [61] focuses only on the challenges of using impact regularization on a theoretical side without proposing a new algorithm or doing empirical experiments, therefore no classification was done.

Table 5.3 summarizes the reviewed works, highlighting the common use of fully observable states and no-op actions. The following subsections delve into specific methods of impact regularization, outlining how each approach handles task performance and environmental preservation.

5 Results

(Relative) Reachability Measures

Krakovna *et al.* [47] propose a framework based on relative reachability, which introduces a stepwise inaction baseline to penalize negative side-effects by comparing the current state with a baseline. They highlight two bad incentives, interference and offsetting, which arise from baseline choices. To mitigate these, the authors develop a magnitude-sensitive relative reachability measure that balances task achievement with minimizing environmental disruption. The deviation from the baseline is penalized using a regularized reward function, and they use stepwise inaction baselines to evaluate the impact of the agent’s actions in environments with fully observable states, known dynamics, and a no-op action. The key innovation here is the introduction of a penalty for actions that deviate from an inaction baseline, allowing agents to distinguish between intended and unintended side-effects. This framework is well-suited for environments where the agent can observe all necessary states, make predictions about future state changes (due to known dynamics), and maintain a history of past states to track cumulative effects. Additionally, the presence of a no-op action helps the agent simulate what the environment would have looked like had it taken no action.

Attainable Utility Preservation

In their works, Turner *et al.* [20], [55], [56] build up on the concept of stepwise inaction baselines [47] by proposing Attainable Utility Preservation (AUP). AUP preserves the agent’s ability to achieve auxiliary reward functions while pursuing a primary goal, thus mitigating negative side-effects. The AUP penalty is calculated by comparing the agent’s current action values with the value of taking a no-op action. This approach assumes a fully observable environment and known dynamics, and it depends heavily on the availability of a no-op action in the action space.

AUP focuses on maintaining the agent’s ability to optimize for auxiliary goals, ensuring that no action drastically reduces future rewards. Like the relative reachability measure, AUP benefits from the presence of known dynamics and fully observable states, which allow the agent to predict and avoid actions that might reduce future attainable utility.

Recognizing the limitations of AUP in environments where a no-op action may not exist, Eresheim *et al.* [73] propose alternative baselines for avoiding negative side-effects when no-op actions are not available. This work extends AUP by introducing the average baseline, average-others baseline, advantage baseline, and random-action baseline, all of which evaluate the impact of actions without relying on a no-op action. These approaches still assume fully observable states and known dynamics, but they do not require an agent to compare its actions to a no-op action.

This extension significantly broadens the applicability of AUP in environments where agents are expected to act continuously (e.g., factory floors or fast-moving systems) by eliminating the dependency on the no-op action while still mitigating unintended side-effects.

5 Results

Safety Aware Reinforcement Learning (SARL)

Miret *et al.* [51] present a framework called Safety Aware Reinforcement Learning (SARL), which introduces a virtual safety agent to guide the main task agent's actions in a manner that minimizes side-effects. SARL operates in environments with fully observable states but does not rely on known dynamics, a no-op action, or past states. Instead, the safety agent and task agent co-train through regularization, where the distance between their action distributions is minimized, ensuring that the task agent's actions align with safety concerns.

This method does not depend on specific assumptions like the presence of a no-op action or known dynamics, making it more generalizable across different environments. By co-training with a virtual safety agent, SARL provides a task-agnostic approach to safety that is adaptable to various complex environments.

Future Task Approach

Krakovna *et al.* [50] present the *Future Task Approach*, which introduces an auxiliary reward function to mitigate negative side-effects by considering the consequences of the current task on future tasks. The approach assumes that the agent's current task is part of a sequence of tasks, and the state the agent leaves after completing the current task serves as the starting state for future unknown tasks. This future task framework penalizes actions that cause interference with future tasks by using a uniform prior over possible future goal states. By aligning the future task reward with a baseline policy (e.g., doing nothing), the approach helps avoid interference incentives. The agent is rewarded for performing as well as or better than this baseline.

The future task approach leverages a fully observable state space and known dynamics to predict how current actions impact future tasks. The use of linear feature rewards simplifies the computation of future task rewards based on the state features, making it easier for the agent to penalize actions that could have long-term negative effects.

Potential-based Multiobjective Reinforcement Learning (MORL)

Vamplew *et al.* [64] propose a *Potential-based Multiobjective Reinforcement Learning* framework to balance the agent's primary objectives with minimizing environmental impact. In this framework, an impact-related reward is derived from the potential difference between the current and previous states, encouraging the agent to avoid irreversible or undesirable changes. The agent is tasked with distinguishing between permissible and impermissible changes in the environment, ensuring reversibility when necessary. Lexicographic and thresholded lexicographic ordering techniques are used to balance the primary task with minimizing environmental disruption.

This approach operates in a fully observable state space, where the agent's actions are assessed based on their potential impact on the environment. The key innovation is the introduction of potential-based rewards to guide the agent in minimizing harmful side-effects, ensuring that environmental changes are reversible and penalizing more critical changes.

5 Results

Challenges using Impact Regularization

Using impact regularization to avoid negative side-effects in reinforcement learning introduces several challenges. Lindner *et al.* [61] highlight key difficulties in selecting appropriate baselines, deviation measures, and determining the magnitude of the regularizer. Different baselines, such as inaction and stepwise inaction, can lead to issues like offsetting behavior, where the agent undoes beneficial actions to minimize deviations. Deviation measures, such as stepwise relative reachability and attainable utility, also face challenges in distinguishing between positive, negative, and neutral impacts. Moreover, delayed effects of actions are often overlooked, requiring improved rollout policies that consider long-term impacts.

The magnitude of the regularizer is another critical factor, as too small a value may fail to prevent harmful side-effects, while a large magnitude may overly constrain the agent. Additionally, the integration of impact regularization with reward functions poses unit conversion and compatibility challenges. Future research directions include exploring causal framing to better manage offsetting behaviors, using probabilistic measures to assess impact, and incorporating human feedback to enhance the agent's understanding of intent.

Theoretical Considerations

Griffin *et al.* [21] explore the concept of *impact regularization* by developing a formal framework for distance-impact penalties, showing how they can influence the agent's behavior by punishing actions that increase the difference between the agent's current state and the counterfactual state. Additionally, the authors address how these penalties remain invariant to certain augmentations of the reward function, and they provide error bounds for optimal policies under misspecification of the state-distance measure.

Griffin *et al.* [21] assume with their approach a fully observable state and relies on the existence of a no-op action. The no-op action is used as a counterfactual baseline, allowing the agent to measure the environmental impact of its actions compared to a scenario where it did nothing. This framework ensures that the agent avoids making irreversible changes by minimizing its long-term impact.

Turner [20] tackle the challenge of *power-seeking policies* in reinforcement learning, which can result in agents taking actions that give them more control over future rewards, often leading to negative side-effects. The authors propose a formalization of the problem as an assistance game, where the agent must balance maximizing immediate rewards with preserving its ability to achieve future objectives (referred to as *POWER*). They demonstrate how power-seeking behavior can be moderated by minimizing impact on future goals and ensuring that the agent is adaptable to potential future changes in the reward function.

This work also operates in a fully observable state space and incorporates the no-op action as part of the solution for balancing immediate task performance with long-term impact minimization. The assistance game framework allows for delayed specification of the true reward function, ensuring that the agent maintains flexibility and avoids locking into power-seeking behaviors that may lead to side-effects.

5 Results

5.1.4 Model-based Penalty Prediction

Model-based penalty prediction is a crucial approach in reinforcement learning (RL) that enables agents to anticipate and mitigate negative side-effects (NSEs) by leveraging learned models of the environment. In such frameworks, the agent typically utilizes feedback mechanisms, human inputs, or learned dynamics to predict penalties associated with harmful actions and adjust its policy to avoid them. These approaches often ensure that agents not only optimize task performance but also consider safety, constraint violations, and unintended consequences of their actions.

The majority of methods in this category assume a fully observable state, where the agent can directly observe the necessary aspects of the environment. Additionally, many rely on known or learned dynamics, allowing the agent to predict how the environment evolves in response to its actions. Some approaches incorporate advanced feedback mechanisms or human intervention to refine penalty predictions. While linear feature rewards are commonly used in some frameworks to simplify learning, others depend on analyzing entire trajectories to handle non-Markovian side-effects, and only a few incorporate the use of no-op actions to further ensure safety.

Study	Fully Observable State	No-Op Action	Known Dynamics	Linear Feature Reward	Action-Independent Reward	Known Past States
Zhang <i>et al.</i> [45]	✓	✗	✗	✗	✗	✗
Frye <i>et al.</i> [46]	✓	✗	✓	✗	✗	✗
Saisubramanian <i>et al.</i> [52]	✓	✗	✓	✓	✗	✗
Saisubramanian <i>et al.</i> [53]	✓	✗	✓	✓	✗	✗
Saisubramanian <i>et al.</i> [71]	✓	✗	✓	✓	✗	✗
Liu <i>et al.</i> [62]	✓	✗	✓	✗	✗	✗
Ma <i>et al.</i> [70]	✓	✗	✓	✗	✗	✗
Srivastava <i>et al.</i> [77]	✓	✗	✓	✗	✗	✗
Saisubramanian <i>et al.</i> [48]	✓	✗	✗	✗	✗	✗
Low <i>et al.</i> [76]	✓	✗	✗	✗	✗	✓
Klassen <i>et al.</i> [60]	✓	✗	✗	✗	✗	✗
Klassen <i>et al.</i> [68]	✓	✗	✗	✗	✗	✗
Saisubramanian <i>et al.</i> [63]	✓	✗	✗	✗	✗	✗
Saisubramanian <i>et al.</i> [19]	✓	✗	✗	✗	✗	✗
Burden <i>et al.</i> [66]	✓	✓	✗	✗	✗	✗

Table 5.4: Taxonomy of agents using model-based penalty prediction based on the proposed classification criteria. Nearly all agents rely on a fully observable state and known dynamics, while only one approach assumes a no-op action the agent’s actions space as wells a providing information about past states of the environment.

Table 5.4 summarizes the identified works, highlighting the reliance on full state observability and learned dynamics. The following subsections delve into specific approaches to model-based penalty prediction, discussing how each method balances performance optimization with safety considerations.

5 Results

Human Feedback

Zhang *et al.* [45] propose a minimax-regret querying approach aimed at mitigating negative side-effects by involving human feedback in the learning process. The agent operates in an environment modeled as a factored Markov decision process (FMDP) and queries a human oracle about the permissibility of changing certain features. The goal is to minimize the worst-case regret of its actions by carefully selecting the features to query, thus minimizing the number of interactions required with the oracle while maintaining a safe policy.

In terms of classification, this approach operates in a *fully observable state*, where the agent has access to all relevant information about the environment, including state features necessary for decision-making. The agent can directly observe the states and the effects of its actions, which allows it to apply the minimax-regret strategy efficiently.

Frye *et al.* [46] introduce the "parenting" algorithm, which mimics human parenting in guiding reinforcement learning agents to safe behaviors. The agent, when faced with unfamiliar or potentially dangerous situations, pauses and requests feedback from a human teacher. This approach is designed to prevent unsafe behavior by leveraging human input to guide decision-making. Furthermore, the agent uses human preferences to adjust its policy, learning from comparisons between action sequences evaluated by the human.

This method is classified under the assumption of a *fully observable state*, where the agent has complete visibility of the environment and can rely on human feedback to learn safe behaviors. By observing all relevant state information, the agent can determine when to ask for human intervention and adjust its policy accordingly.

Saisubramanian *et al.* [52], [53] present a framework to mitigate NSEs by augmenting the traditional MDP with a secondary objective aimed at minimizing NSEs. In this approach, the agent uses an acquired model of the environment to optimize for a primary task. Meanwhile, a more complex, augmented model handles the secondary objective of minimizing NSEs.

Building on their previous work, Saisubramanian *et al.* [71] focus on improving feedback mechanisms and addressing limitations in learning from human inputs, particularly biases in feedback collection and issues with state representation fidelity. They explore how different feedback mechanisms, such as random queries and human approval, impact the learning process and how biases introduced through feedback affect the agent's ability to mitigate NSEs. They also address challenges related to non-Markovian side-effects, which are not directly associated with individual state-action pairs but with trajectories over time.

This approach is classified as utilizing a *fully observable state*, where the agent has access to all the necessary state information for its decision-making. Additionally, it operates with *known dynamics*, meaning the agent has knowledge of the environment's internal transition model, allowing it to predict how the environment will evolve in response to its actions. Furthermore, the reward structure in this framework is *linearly dependent on features of the state*, which enables the agent to compute optimal policies by calculating the reward as a linear function of state features.

5 Results

Constrained Models

Liu *et al.* [62] propose a model-based reinforcement learning framework that aims to ensure safety by minimizing constraint violations while optimizing task performance. The framework uses a constrained model-based RL approach within a Model Predictive Control (MPC) framework. This approach assumes the system's dynamics are initially unknown but can be learned from data using an ensemble of neural networks. The robust cross-entropy (RCE) method, which accounts for model uncertainty, helps ensure safety during learning.

This work is classified as assuming a *fully observable state*, meaning the agent has complete knowledge of the environment's relevant aspects, and *known dynamics*, where the dynamics are either learned or given, allowing the agent to make accurate predictions about future states.

Ma *et al.* [70] introduce the Conservative and Adaptive Penalty (CAP) framework for safe model-based RL. CAP incorporates environmental cost feedback to predict and mitigate negative side effects (NSEs). The framework dynamically adjusts the penalty imposed on the agent based on feedback, balancing exploration and safety while ensuring that the agent avoids unsafe actions. CAP is designed to operate in environments where the agent has full observability and where the underlying transition dynamics of the system can be learned or modeled.

Like Liu *et al.* [62], this approach assumes a *fully observable state* and *known dynamics*. The agent is aware of all relevant environmental information and uses that knowledge to optimize its policy while minimizing negative outcomes.

Srivastava *et al.* [77] address the challenge of mitigating negative side effects in reinforcement learning with non-Markovian characteristics, which depend on entire trajectories rather than individual state-action pairs. Their Controller-Assisted Safe Planning (CASP) framework leverages finite state controllers (FSCs) to represent the sequence of actions and learn about potential NSEs. The FSCs are integrated into a constrained Markov Decision Process (CMDP), and the agent uses this enhanced model to plan while avoiding negative side effects.

This work is classified as operating with a *fully observable state*, where the agent can observe all relevant aspects of the environment, and *known dynamics*, as the CMDP relies on learned or known transitions to make predictions about future states.

Planning with Reduced or Incomplete Models

Saisubramanian *et al.* [48] tackle the problem of negative side effects arising from planning with reduced models, where only a subset of possible outcomes for each state-action pair is considered. Their approach focuses on improving the efficiency of planning by using reduced models while selectively enhancing model fidelity in risky states to avoid side effects. This method is designed for use in stochastic shortest path problems and balances computational efficiency with safety.

The method is classified under *fully observable state*, where the agent can observe all necessary information in the environment to make informed decisions. However, it does not assume knowledge of the full transition dynamics upfront, differentiating it from other approaches that rely on learning or modeling the environment's dynamics.

5 Results

Low *et al.* [76] aim to enhance safety by learning non-Markovian safety constraints—safety issues that depend on entire trajectories rather than just immediate state-action pairs. The method uses Gated Recurrent Units (GRUs) to classify state-action trajectories based on the likelihood of NSEs. The approach is framed as a CMDP, where the agent maximizes rewards while satisfying safety constraints that are learned from labeled trajectory data.

This work is classified as assuming a *fully observable state*, *known dynamics*, and *known past states*. The agent has access to the full state information, understands the environment's dynamics, and can reference past states to identify and avoid NSEs that depend on historical sequences of actions.

Symbolic Planning

Klassen *et al.* [60], [68] focus on addressing negative side-effects in sequential decision-making processes within symbolic planning frameworks. They formalize the problem as a state-transition system and propose the concept of a "change-minimizing plan" to minimize unintended side-effects. This concept is built upon measuring the changes in the environment before and after the plan execution. The goal is to ensure that after achieving its objectives, the agent leaves the environment as close to its initial state as possible, thereby minimizing unintended negative impacts.

This work assumes a *fully observable state*, where the agent can directly observe all necessary aspects of the environment, and *known dynamics*, where the transition function governing state changes is well-defined and deterministic. This allows the agent to accurately predict the outcome of its actions, leading to plans that minimize unintended changes.

Environment Shaping

Saisubramanian *et al.* [19], [63] propose an actor-designer framework to mitigate negative side-effects by involving a human designer who can modify the environment. The actor agent operates within a configurable environment, optimizing its task while potentially causing side-effects. The human designer agent evaluates the environment and reshapes it to minimize these side-effects without disrupting the actor's task performance. The framework ensures that any environmental reconfiguration remains within specific constraints to avoid hindering the actor's goal achievement.

This approach assumes a *fully observable state*, where both the actor and designer have full knowledge of the current state of the environment, and *known dynamics*, allowing the designer to predict the effects of environment modifications and the agent's behavior accordingly.

Safety Wrappers

Burden *et al.* [66] introduce the concept of "safety wrappers" for reinforcement learning agents to mitigate negative side-effects by switching the agent to a safer behavior when its confidence in its actions is low. The safety wrapper monitors the agent's confidence and triggers a safety policy when confidence drops below a certain threshold. The safety policy could include actions such as a no-op (standing still) or other safe alternatives designed to prevent harm. This mechanism helps the agent avoid risky actions without drastically reducing its performance.

This work assumes a *fully observable state*, where the agent can observe all relevant information about the environment, and includes the concept of a *no-op action*, allowing the agent to perform a neutral action when uncertain, which prevents it from affecting the environment negatively.

5 Results

5.1.5 Inverse Reinforcement Learning

Inverse Reinforcement Learning (IRL) is a powerful framework that allows agents to infer reward functions by observing the behavior of an expert, rather than being provided with an explicit reward function. This approach is particularly useful in scenarios where it is challenging to handcraft a reward function that encapsulates human preferences, especially when safety and the avoidance of negative side-effects are of critical concern. In the reviewed studies, IRL is leveraged to learn robust and safe policies, either by simulating expert trajectories or by inferring human preferences from observed states.

The approaches reviewed rely heavily on key assumptions to achieve effective learning. Most notably, they assume a fully observable environment, where the agent has complete access to all relevant state information. Additionally, many methods operate under known or learnable dynamics, allowing agents to accurately model how the environment transitions in response to their actions. Some approaches further utilize linear feature rewards to simplify the learning process, while others draw upon the history of past states to infer preferences and ensure safety.

Study	Fully Observable State	No-Op Action	Known Dynamics	Linear Feature Reward	Action-Independent Reward	Known Past States
Huang <i>et al.</i> [44]	✓	✗	✓	✓	✗	✗
Shah <i>et al.</i> [49]	✓	✗	✓	✗	✗	✓
Shah [54]	✓	✗	✓	✗	✗	✓

Table 5.5: Taxonomy of agents using inverse reinforcement learning based on the proposed classification criteria. Retrieved studies show a strong dependence on fully observable states and known dynamics. One approach uses (linear) dependent features for the reward function, while other approaches rely on known past states to learn from human expert trajectories.

Table 5.5 summarizes the classification of the identified works based on these criteria, highlighting a strong reliance on fully observable states and known dynamics. The remainder of this section delves into specific methods for inverse reinforcement learning and their applications in ensuring safety and robustness in decision-making tasks.

5 Results

MDP Solver

Huang *et al.* [44] propose a method for learning robust policies that maximize rewards while ensuring safety, even when faced with incomplete or inaccurate information. The key to this method is leveraging expert demonstrations to guide the learning process through a max-min learning framework, closely related to inverse reinforcement learning (IRL) and apprenticeship learning. The approach solves a max-min problem in a finite MDP, where the reward function is assumed to be linearly dependent on state features.

This work is classified as operating under a *fully observable state*, where the agent can observe all relevant information about the environment, and *known dynamics*, where the transition function is given or can be learned. Furthermore, the reward function is based on *linear feature reward*, meaning that rewards are calculated as a linear function of the state features, simplifying the process of learning robust policies.

Implicit Preference Information

Shah *et al.* [49] introduce an algorithm called Reward Learning by Simulating the Past (RLSP) to infer human preferences from the initial state of an environment. The key insight is that the initial state of the world is already optimized according to human preferences. The agent infers these preferences by simulating past trajectories that could have led to the observed initial state. Using maximum causal entropy inverse reinforcement learning (MCEIRL), the agent updates its reward function to reflect the inferred preferences from the observed state.

This work is classified as using a *fully observable state*, where the agent can see the current state of the environment, and *known dynamics*, as the agent relies on the known or learned transition dynamics to simulate past trajectories. Additionally, the method depends on *known past states*, as it explicitly simulates the past to infer the reward function based on how the current state was reached.

Building on their previous work, Shah [54] extend RLSP to high-dimensional and continuous environments with Deep RLSP. This method integrates reward learning, planning, and reinforcement learning to infer human preferences from the initial state and optimize the agent's behavior accordingly. Deep RLSP employs a Variational Autoencoder (VAE) to learn a feature representation of the state space and uses a neural network to simulate past trajectories. The human model's policy is learned through Soft Actor-Critic (SAC), which continuously updates based on inferred rewards.

This work operates under the assumption of a *fully observable state* and *known dynamics*, where the agent can observe and predict the effects of actions. It also relies on *known past states*, using simulations of past trajectories to infer the reward function based on the observed initial state.

5.2 Gaps in the Literature and Future Directions

Despite the significant progress in mitigating negative side-effects in reinforcement learning, several gaps remain. These gaps highlight areas where current approaches are insufficient, underexplored, or limited by simplifying assumptions that constrain their applicability in real-world scenarios. Addressing these gaps is essential for advancing the field and ensuring safe RL deployment in increasingly complex environments.

5.2.1 Handling Partial Observability and Non-Markovian Side-Effects

Many studies, such as those on Attainable Utility Preservation [55], [56], [73], assume fully observable environments. However, real-world environments are often partially observable, requiring agents to make decisions based on incomplete or noisy data. Agents may lack critical information about the environment, leading to unintended side-effects when crucial elements are hidden or uncertain.

Additionally, much of the literature assumes Markovian dynamics, where the effects of an agent's actions depend solely on the current state. However, real-world systems often exhibit non-Markovian dynamics, where the consequences of actions depend on past events. Existing frameworks like Occupancy-Regularized Policy Optimization [75] are limited in addressing long-term or cumulative side-effects, which are critical in domains like environmental conservation.

Future Directions Developing methods that handle partial observability and non-Markovian dynamics is essential. Approaches such as Partially Observable Markov Decision Processes (POMDPs), memory-based architectures, and non-Markovian reward functions can help agents operate safely under uncertainty and track long-term consequences of actions.

5.2.2 Real-Time Human Feedback and Interaction

While some studies explore incorporating human feedback into RL, such as minimax-regret querying [45], most rely on static or batch feedback. This is insufficient in dynamic environments where real-time adjustments are critical. Delayed responses to hazards in safety-critical domains like autonomous vehicles or healthcare can lead to side-effects that could have been avoided with timely intervention.

Future Directions Research should focus on developing interactive reinforcement learning methods that allow agents to query humans in real-time during uncertain or high-risk situations. Techniques like active learning and collaborative decision-making frameworks could enable agents to incorporate human guidance dynamically and adapt more effectively to evolving environments.

5 Results

5.2.3 Lack of Standardized Benchmarks and Real-World Transferability

Current RL safety research lacks a unified set of benchmarks that comprehensively evaluate agents' ability to mitigate negative side-effects across various tasks and environments. While environments like AI Safety Gridworlds [15] and SafeLife [18] provide useful testbeds, they often focus on specific scenarios and oversimplify real-world complexities such as hardware failures, unpredictable environmental changes, and human interactions.

Future Directions There is a need for standardized, diverse benchmarks that include various domains (e.g., healthcare, robotics, autonomous vehicles) and reflect real-world complexities like dynamic, stochastic, and partially observable settings. Additionally, research should focus on improving the transferability of RL agents from simulation to real-world environments by enhancing simulators and developing domain adaptation techniques.

5.2.4 Challenges in Balancing Task Performance and Safety

A persistent challenge in RL systems is the trade-off between optimizing task performance and ensuring safety. Reward regularization and impact minimization strategies, such as those described by Alizadeh Alamdari *et al.* [58], penalize harmful actions but can lead to overly conservative agents that sacrifice performance. For example, in SafeLife [18], agents trained to minimize side-effects struggled to balance task completion with minimizing disruptions.

Future Directions Future research should explore dynamic safety mechanisms that adapt to the context and complexity of the task. Multi-objective optimization frameworks that balance task performance and safety could help agents be neither too conservative nor too reckless. Dynamic regularization strategies could adjust safety constraints based on the risk level or task progress.

5.2.5 Safety in Multi-Agent Systems and Emergent Side-Effects

Most current research focuses on single-agent RL environments, yet many real-world systems involve multiple agents interacting simultaneously. In multi-agent settings, actions taken by one agent can influence others, leading to emergent side-effects that may not be present in single-agent systems. This is particularly important in domains like autonomous traffic systems, smart grids, or robotic swarms.

Future Directions Future work should develop safety mechanisms for multi-agent environments, focusing on cooperation, communication, and coordination among agents. Social welfare functions and mutual safety guarantees are crucial for avoiding negative side-effects in these systems. Research should also explore how safety mechanisms scale as the number of interacting agents increases.

5.2.6 Long-Term Safety Guarantees

Existing safety mechanisms often focus on minimizing short-term side-effects without addressing the cumulative, long-term impact of agent actions. For example, approaches like relative reachability [47] penalize immediate negative side-effects, but they do not account for the compounding effects of repeated actions. This is particularly critical in areas like environmental management or resource conservation.

Future Directions Future work should develop theoretical frameworks that provide long-term safety guarantees. This could involve continuous monitoring of agent behavior, incorporating predictive models to anticipate future risks, and applying formal verification techniques to ensure agents remain aligned with safety objectives over time.

5.2.7 Ethical Considerations and Consistent Definitions of Side-Effects

Many safety mechanisms in RL implicitly embed value judgments about what constitutes safe behavior, which may not align with social or ethical standards. Furthermore, there is a lack of consistent definitions and metrics for side-effects across different domains, such as healthcare and autonomous driving, making cross-domain evaluations challenging.

Future Directions Future work should integrate ethical considerations into the design and evaluation of RL safety mechanisms, ensuring fairness, accountability, and transparency. Additionally, efforts should focus on developing standardized definitions and metrics for side-effects across domains to enable clearer comparisons and more versatile safety solutions.

6 Discussion

We reflect on the findings presented in [chapter 5](#), focusing on the effectiveness and limitations of existing strategies for mitigating negative side-effects in reinforcement learning. The discussion is organized into three primary areas with the assumptions underlying current safety mechanisms, the trade-offs between performance and safety, and the challenges in applying these mechanisms to real-world scenarios.

6.1 Assumptions and Limitations of Current Safety Mechanisms

A central theme in the literature is the reliance on several key assumptions, which, while simplifying the design of safe RL agents, also limit their applicability. Many approaches depend on fully observable states, the availability of a no-op action, and known dynamics in the environment.

Fully Observable States A significant number of studies assume that the agent operates in a fully observable environment, where it has access to all relevant information about the current state. This assumption greatly simplifies decision-making by removing uncertainty about the environment. However, this is a highly idealized scenario and real-world environments are often partially observable, with agents having to make decisions based on incomplete or noisy information. For instance, autonomous systems such as robots or self-driving cars frequently encounter situations where critical elements of the environment are hidden or ambiguous. This limits the generalizability of many RL safety mechanisms, where agents must infer the state of the environment from partial observations. Moreover, the assumption of full observability can make agents overly dependent on the availability of complete and accurate sensory data, which is rarely the case in practice.

No-Op Action as a Baseline Many of the studied agents include a no-op action, allowing them to choose to do nothing for a time step. This serves as a neutral reference point for assessing the impact of actions. While the no-op action is useful for analyzing agent behavior in controlled environments, it may not always be feasible in dynamic, real-world applications where inaction can itself have negative consequences. For example, in domains such as autonomous driving or healthcare, choosing not to act could lead to accidents or missed opportunities for intervention, making the no-op action an inadequate baseline for safety evaluation in such contexts.

Known Dynamics and Their Limitations A common assumption in many safety mechanisms is that the agent has knowledge of the environment’s dynamics or can learn them through interaction. This assumption allows the agent to predict the consequences of its actions and minimize unintended side-effects. However, environments with unpredictable, stochastic, or evolving dynamics pose significant challenges for such approaches. In these settings, the assumption of known dynamics becomes a limitation, as agents may struggle to adapt to unexpected changes, leading to safety failures.

6.2 Performance-Safety Trade-offs

A critical challenge highlighted in the results is the trade-off between optimizing for task performance and ensuring safety. Many RL systems prioritize reward maximization, which can lead to behaviors that exploit loopholes in the reward structure, causing unintended negative side-effects.

Reward Regularization and Impact Regularization Reward regularization approaches, such as those discussed in [section 4.2](#), attempt to mitigate negative side-effects by incorporating penalties for harmful actions into the agent’s reward function. While effective in encouraging safer behavior, this often results in agents becoming overly conservative, sacrificing task performance in the process. Similarly, impact regularization ([section 4.3](#)) penalizes actions that deviate from a baseline state, ensuring that agents minimize disruptions to the environment. However, this can lead to unintended outcomes, such as agents undoing beneficial changes to avoid penalties, highlighting the difficulty of balancing safety and performance. In many cases, these approaches lead to suboptimal behavior, especially in complex environments where strict safety constraints conflict with the need to complete tasks efficiently. For example, in the SafeLife environment [18], agents trained to minimize side-effects struggled to achieve high task performance, emphasizing the tension between these two competing objectives.

Challenges in Long-Term Safety Another important aspect of the performance-safety trade-off is the difficulty in ensuring long-term safety. Most reviewed approaches focus on minimizing immediate side-effects, but fail to address the cumulative impact of actions over time. This is particularly problematic in scenarios where the full consequences of an agent’s behavior may not be apparent until much later, such as in environmental management or resource conservation. Short-term optimization, as seen in many reward-driven agents, often overlooks these long-term risks, leading to decisions that may appear safe in the moment but cause significant harm over time.

6.3 Challenges in Real-World Applications

A recurring issue in the literature is the gap between simulation-based safety mechanisms and their application in real-world environments. While simulations offer controlled and repeatable testbeds for evaluating agent safety, they often fail to capture the full complexity and unpredictability of real-world settings.

Over-Reliance on Simulated Environments Most of the experiments discussed in this thesis rely on simulated environments such as AI Safety Gridworlds [15] and SafeLife [18]. These environments are valuable for testing theoretical models of safety, but they often simplify key aspects of the real world, such as sensory noise, hardware constraints, and human interaction. As a result, agents that perform well in simulations may struggle when transferred to real-world applications, where the dynamics are more complex and less predictable. For example, simulations typically assume that the agent has access to perfect information about the environment, or that the dynamics are deterministic, allowing for accurate predictions of action outcomes. However, in real-world systems, factors such as sensor errors, hardware malfunctions, and human unpredictability can significantly affect an agent’s ability to operate safely.

6 Discussion

Ethical Considerations in Reinforcement Learning Safety Another challenge in the deployment of RL agents in real-world applications is the ethical implications of safety mechanisms. Reward structures and regularization methods implicitly encode value judgments about what constitutes safe or desirable behavior, and these judgments may not always align with societal or ethical standards. For instance, safety mechanisms designed to prioritize efficiency or cost reduction in healthcare could inadvertently lead to inequitable outcomes, such as reduced access to care for vulnerable populations. Additionally, many of the safety mechanisms discussed in this thesis lack transparency, making it difficult for human operators to understand or interpret the agent's decisions. This lack of interpretability can undermine trust in RL systems, especially in high-stakes environments like autonomous vehicles or critical infrastructure. Furthermore, the absence of accountability mechanisms raises concerns about how responsibility will be assigned if an agent's behavior leads to harmful outcomes.

7 Conclusion

7.1 Answering the Research Questions

The goal of this thesis was to conduct a systematic literature review to investigate the methods used to avoid negative side effects in reinforcement learning and to identify gaps in the literature that require further exploration. In this section, we revisit the research questions posed and provide answers based on the findings of the literature review.

“What approaches have been taken in previous studies to ensure that reinforcement learning agents avoid negative side effects while completing their tasks?”

The review identified a wide range of approaches designed to mitigate negative side effects in RL. These can be broadly categorized into several methodologies:

Reward Regularization: One of the most commonly used techniques involves augmenting the reward function to penalize negative side effects. Methods such as Occupancy-Regularized Policy Optimization (ORPO) [75] and reward regularization frameworks [58] incentivize agents to balance task performance with the safety of their actions. These approaches penalize agents for harmful behaviors while rewarding them for actions that align with both the task goal and environmental safety.

Impact Regularization: Several studies have adopted impact regularization methods, such as relative reachability [47] and attainable utility preservation [56], which penalize deviations from a baseline state. These approaches focus on minimizing irreversible or disruptive changes in the environment, thus mitigating the risk of long-term negative side effects.

Inverse Reinforcement Learning: Another approach identified in the review is the use of IRL, which allows agents to infer reward functions by observing human experts or analyzing the initial state of the environment. Studies like those by Shah *et al.* [49] and Huang *et al.* [44] highlight how IRL is used to learn robust, human-aligned policies that minimize negative side effects by inferring preferences from demonstrations or initial state configurations.

Human Feedback and Model-Based Penalty Prediction: Incorporating human feedback to guide RL agents' behavior is another prevalent method. Approaches like minimax-regret querying [45] and the parenting algorithm [46] involve real-time or batch feedback from humans to prevent unsafe actions. Model-based methods like those introduced by Saisubramanian *et al.* [53] also allow agents to predict penalties for unsafe actions, ensuring safety through learned dynamics of the environment.

7 Conclusion

Safety Constraints in Multi-Agent Systems: Although less explored, some studies have examined the challenges of avoiding negative side effects in multi-agent systems. Environments like SafeLife [18] have been used to evaluate agents in multi-agent scenarios where the actions of one agent can impact others, leading to emergent behaviors that require robust safety mechanisms.

Overall, the identified approaches reveal a strong reliance on regularization techniques, model-based penalty prediction, human feedback integration, and the growing importance of IRL in handling complex, safety-critical environments.

“What areas within the domain of avoiding negative side effects in reinforcement learning require further exploration?”

Despite the progress made in addressing negative side effects in RL, several gaps in the literature remain, highlighting opportunities for further research:

Partially Observable Environments: The majority of studies assume fully observable environments, where agents have complete knowledge of the state space. However, real-world scenarios are often characterized by partial observability, where agents must make decisions based on incomplete or noisy data. This is a significant gap in current research, and developing approaches that can handle uncertainty in such environments is crucial for advancing safe RL in practical applications.

Lack of Standardized Safety Benchmarks: There is a need for more diverse, standardized benchmarks that can evaluate RL agents’ safety across a range of domains and environments. Current benchmarks, such as AI Safety Gridworlds [15] and SafeLife [18], provide valuable insights but are limited in their generalizability. Future work should focus on creating benchmarks that can assess safety in complex, real-world tasks.

Non-Markovian Side Effects: Most current approaches rely on Markov Decision Processes, which assume that the next state only depends on the current state and action. However, real-world environments often exhibit non-Markovian dynamics, where the consequences of actions depend on historical events. There is a need for frameworks that can handle non-Markovian side effects, particularly in long-term, dynamic environments.

Real-Time Human Feedback Integration: While human feedback has been incorporated into some safety frameworks, real-time feedback mechanisms are underexplored. Integrating continuous human input into RL systems could allow for more adaptive, on-the-fly decision-making in dynamic environments, thereby enhancing safety.

Balancing Task Performance and Safety: A key challenge across the reviewed studies is the trade-off between maximizing task performance and ensuring safety. Current regularization methods often result in overly conservative agents that sacrifice performance to minimize side effects. Future research should explore dynamic safety mechanisms that allow agents to balance performance with safety, adjusting their behavior based on the complexity or risk of the task.

7 Conclusion

Multi-Agent Systems and Emergent Side Effects: Research on avoiding side effects in multi-agent systems is still in its infancy. The interactions between agents can lead to emergent behaviors that cause side effects, and more work is needed to develop safety mechanisms that can manage these complex, multi-agent environments.

Ethical Considerations in Safety Mechanisms: Finally, the ethical implications of RL safety mechanisms are largely underexplored. As RL systems are deployed in high-stakes environments, it is essential to consider the fairness, accountability, and transparency of safety measures, particularly when they influence real-world decision-making processes.

In conclusion, while significant progress has been made in developing approaches to mitigate negative side effects in RL, several important research gaps remain. Addressing these will be essential for advancing the field and ensuring the safe deployment of RL agents in increasingly complex and real-world environments.

7.2 Limitations and Mitigation Strategies

While systematic literature reviews provide valuable insights by synthesizing existing research, they are not without limitations. This section outlines key challenges encountered during the review process and presents strategies employed to mitigate these limitations.

Publication bias Systematic literature reviews are inherently dependent on the availability of published research, which can introduce publication bias. Studies with positive or significant findings are more likely to be published than those reporting null or negative results, leading to an overrepresentation of certain outcomes. This skewed availability may limit the comprehensiveness of the review by underreporting less favorable or inconclusive findings.

To address this bias, we included preprints from reputable sources like ArXiv, provided the authors had peer-reviewed publications in the field or affiliations with respected research organizations such as Google DeepMind or OpenAI. This expanded the scope beyond traditional peer-reviewed outlets, ensuring a more comprehensive overview of existing research, regardless of publication status.

Selection bias Even with a systematic search strategy, there is a possibility that some relevant studies might have been missed. Variations in database search capabilities or the specificity of search terms could lead to the exclusion of important studies. Moreover, inclusion criteria may inadvertently omit studies that offer valuable perspectives on the topic, resulting in selection bias.

To minimize this risk, we employed a multi-pronged search strategy, combining database searches with targeted workshops and author-specific queries. This method ensured that we captured a broad range of studies across different venues and research contexts. Additionally, four independent reviewers scored the retrieved records, which helped minimize individual bias in study selection.

7 Conclusion

Quality of included studies Systematic reviews typically include studies with varying methodologies, which can introduce uncertainty in the findings. Low-quality studies, if given equal weight as high-quality ones, can skew conclusions or introduce bias into the review.

To address the quality of included studies, we implemented a scoring system that assigned weights based on relevance and innovation. Studies with higher methodological quality and more significant contributions were prioritized. By setting a threshold score for inclusion, we ensured that only studies meeting a certain quality standard were considered in the final analysis.

Heterogeneity of studies The included studies often vary in terms of methodologies, study designs, and evaluation environments, which introduces heterogeneity. This diversity can make it difficult to compare results or draw unified conclusions across different research approaches and findings.

We classified the studies using predefined taxonomies (e.g., agent assumptions, safety mechanisms) to organize the diverse findings into distinct categories. This approach facilitated the synthesis of heterogeneous studies by providing a structured framework for comparison, allowing us to identify patterns within specific subsets of the literature.

Inability to establish causality Systematic literature reviews are primarily retrospective analyses, summarizing existing evidence rather than generating new data. As such, they cannot definitively establish causal relationships between interventions and outcomes. The focus is on identifying associations rather than proving cause and effect.

Although causal claims are outside the scope of this review, we focused on identifying trends, gaps, and potential associations within the existing literature. By clearly outlining the limitations of the reviewed studies in drawing causal inferences, we maintained transparency about the scope of conclusions.

Time lag The process of conducting a systematic review, from literature searches to synthesis and analysis, is resource-intensive and time-consuming. Consequently, by the time the review is completed, new studies may have been published, potentially affecting the findings' relevance or currency.

To mitigate the impact of this time lag, we specified a publication window from 2019 to 2023, ensuring the review reflected the most recent developments. Additionally, by including preprints and ongoing research, we aimed to capture emerging trends that could inform future research directions.

In conclusion, while the systematic review process used in this thesis comes with inherent limitations, proactive strategies were employed to mitigate their impact. By broadening the scope of literature sources, employing rigorous scoring and selection methods, and ensuring transparency in reporting, we sought to provide a comprehensive and reliable synthesis of research on avoiding negative side effects in reinforcement learning.

7.3 Future Work

The findings of this thesis highlight several areas where future research is essential for advancing safe reinforcement learning. While we already elaborated on future research directions in [section 5.2](#) with gaps in the literature and future directions, we want to provide a comprehensive overview on how avoiding negative side-effects in reinforcement learning can be advanced in the future.

One critical direction is addressing the limitations of current safety mechanisms, many of which rely on idealized assumptions such as fully observable states, the availability of no-op actions, and known dynamics. In real-world environments, agents frequently encounter partial observability, where critical information about the environment is hidden or noisy, making it challenging to ensure safety. Future research should explore methods that allow agents to operate safely under these conditions by integrating techniques like Partially Observable Markov Decision Processes (POMDPs) and uncertainty-aware decision-making strategies.

The lack of standardized benchmarks for evaluating safety across various RL tasks and environments is another key challenge. Existing benchmarks already offer valuable insights, but often focus on specific types of side-effects or scenarios, limiting their generalizability. To address this, future research should aim to develop comprehensive and diverse benchmark suites that reflect the complexity of real-world applications, encompassing dynamic, stochastic, and partially observable environments. Patterson *et al.* [134] for example introduced a new empirical methodology with a benchmark to universally compare RL agents across different environments and settings. The field of safe RL needs a similar methodology to ensure standardized benchmarks and evaluations across different settings. This would allow for more consistent cross-comparisons of safety mechanisms and improve the applicability of RL agents in real-world scenarios.

A significant gap in the literature is the handling of non-Markovian side-effects, where the consequences of actions depend on the history of past states rather than just the current state. Existing safety mechanisms often assume a Markovian structure, which simplifies the learning process but fails to capture long-term or cumulative side-effects. Future research should focus on developing frameworks that account for these temporal dependencies, potentially using memory-based architectures or extending Markov Decision Process frameworks to handle non-Markovian dynamics.

Moreover, while several studies have explored the integration of human feedback into RL systems, most rely on static or batch feedback. In dynamic and high-risk environments, real-time human feedback is crucial for adapting agents' behavior in response to evolving conditions. Future research should investigate methods for real-time human-agent interaction, enabling agents to query humans during uncertain or risky situations. This could improve agents' ability to make safe decisions on-the-fly, particularly in safety-critical domains like healthcare and autonomous driving.

The trade-off between maximizing task performance and ensuring safety remains a persistent challenge. Current approaches, such as reward and impact regularization, often lead to overly conservative agents that sacrifice performance for safety. Future work should explore dynamic safety mechanisms that adjust safety constraints based on the context and complexity of the task. Multi-objective optimization frameworks could be developed to balance performance and safety more effectively, ensuring agents are not too conservative in situations where task completion is critical.

7 Conclusion

In multi-agent systems, the potential for emergent side-effects caused by interactions between agents presents a unique challenge. Current research predominantly focuses on single-agent environments, yet real-world systems often involve multiple agents interacting simultaneously. Future work should develop safety mechanisms tailored to multi-agent settings, focusing on coordination, communication, and mutual safety guarantees. These mechanisms will be essential for domains such as autonomous fleets, smart grids, and multi-robot coordination, where the actions of one agent can have significant implications for others.

Long-term safety guarantees are also an area in need of further exploration. Most safety mechanisms focus on minimizing immediate side-effects, but fail to account for the long-term consequences of repeated actions. Future research should aim to develop theoretical frameworks that provide long-term safety guarantees, using predictive models and formal verification techniques to ensure agents remain aligned with safety objectives over extended periods.

The over-reliance on simulated environments in RL research presents another challenge. While simulations are useful for testing theoretical models, they often oversimplify real-world complexities, such as sensory noise, hardware limitations, and human unpredictability. To ensure that safety mechanisms are robust and transferable to real-world applications, future work should focus on improving the realism of simulators and developing transfer learning techniques that allow agents to generalize their behavior across different environments.

Finally, the ethical implications of reinforcement learning safety mechanisms must be given more attention. Many current approaches implicitly embed value judgments about what constitutes safe behavior, which may not always align with social or ethical standards. Future research should integrate ethical considerations into the design of safety mechanisms, ensuring that agents make fair, accountable, and transparent decisions. This includes developing fairness constraints and explainability tools that allow humans to understand and audit the behavior of RL systems, particularly in high-stakes environments like healthcare and autonomous vehicles.

List of Figures

2.1	PRISMA 2020 flow diagram [14] including the identification of studies via databases and registers. * Google Scholar returned a server error when exporting more than 1000 entries, resulting in the most 1000 relevant results identified from this database ** ACM Digital Library did not have an export feature, therefore search results known to be discarded in a future step were removed	5
3.1	Interaction between the Agent and the Environment in Reinforcement Learning [16]. . . .	9
3.2	Generalized policy iteration for estimating $V(s)^*$ and $\pi(a s)^*$ [16].	15
4.1	Illustration of the limitations of using actions distribution divergence for policy regularization within a MDP [75].	35
4.2	Designers can choose between different baseline states and deviation measures to penalize the reachability of states [47].	38
4.3	An overview of the co-training framework for Safety Aware Reinforcement Learning (SARL) [51]. The task agent $A(\theta)$ generates action probabilities based on the environment, while the virtual safety agent $Z(\psi)$ provides alternative safe action probabilities from the same environment state. The framework computes a distribution loss $L_{dist}(\theta, \psi)$, representing the divergence between the task and safe actions, which is combined with the task loss to compute the total loss during backpropagation. This process ensures that the task agent balances task performance with safety by considering the suggestions from the safety agent.	44
4.4	Future task approach, that aligns the current task on an agent with unknown future tasks by using an auxiliary reward function, that penalizes negative side-effects [50].	45
4.5	Future task approach with a baseline policy π' , where a hypothetical agent (shown in blue) runs on future task i [50].	46
4.6	Multi-Objective Approach to Mitigate Negative Side-Effects Framework using a feedback mechanism of collected NSE data, Random Forest regression to predict the penalty R_N , and multi-objective planning to compute a NSE-mitigating policy [52], [53].	58
4.7	Differences of policy sets with and without using slack [52], [53]	59
4.8	Learning from user feedback to mitigate negative side-effects using the approval (HA) strategy, where the agent queries a human who approves or disapproves actions based on randomly selected state-action pairs [52], [53].	60
4.9	Comparison between the original problem, most likely outcome determinization (MLOD) and 0/1 RM approach [48].	65
4.10	GRU network to predict negative side-effects subject to the defined CMDP objective based on state-action pairs in the trajectory [76].	66

List of Figures

4.11	Stochastic computation graph of $\hat{C}_i(\tau)$ using GRUs to classigy the likelihood of negative side-effects across a trajectory τ with $K = 3$ NSE classes. The GRU process sequences of states and actions, while the Lagrange multiplier method adjusts the policy π_θ to enforce safety constraints based on the classification of the trajectory [76].	67
4.12	Cyclic relationship between an agent and an environment designer to mitigate negative side-effects using environment shaping. [63], [71].	69
4.13	Illustration of learning preferences from an initial (starting) state using Reward Learning by Simulating the Past (RLSP) [49].	74

List of Tables

4.1	Studies covered in the systematic literature review ordered by year in ascending order. . .	26
5.1	Taxonomy of agents without safety mechanisms based on the proposed classification criteria. Retrieved studies show a strong dependence on fully observable states and a no-op action in the agent's action space. The reward is not dependent on (linear) features and only dependent on the actions taken by the agents. Information about past states are not available in the learning algorithms. Amodei <i>et al.</i> [6] focuses only on the theoretical foundations on avoiding negative side-effects without proposing a new algorithm or doing empirical experiments, therefore no classification was done.	77
5.2	Taxonomy of agents using reward regularization based on the proposed classification criteria. All agents rely on a fully observable state and known dynamics of the environment. The reward is only dependent on the actions take by the agents and not on some sort of (linear) features.	80
5.3	Taxonomy of agents using impact regularization based on the proposed classification criteria. All agents rely on a fully observable state and a no-op action in the agent's action space builds an often used assumption. Lindner <i>et al.</i> [61] focuses only on the challenges of using impact regularization on a theoretical side without proposing a new algorithm or doing empirical experiments, therefore no classification was done.	83
5.4	Taxonomy of agents using model-based penalty prediction based on the proposed classification criteria. Nearly all agents rely on a fully observable state and known dynamics, while only one approach assumes a no-op action the agent's actions space as wells a providing information about past states of the environment.	87
5.5	Taxonomy of agents using inverse reinforcement learning based on the proposed classification criteria. Retrieved studies show a strong dependence on fully observable states and known dynamics. One approach uses (linear) dependent features for the reward function, while other approaches rely on known past states to learn from human expert trajectories.	91

Glossary

0/1 RM	0/1 Reduced Model
A2C	Advantage Actor-Critic
A3C	Asynchronous Advantage Actor-Critic
AC	Actor-Critic
AD-NSE	Actor-Designer Framework for Negative Side-Effect Mitigation
AI	Artificial Intelligence
AUP	Attainable Utility Preservation
CAP	Conservative and Adaptive Penalty
CASP	Controller-Assisted Safe Planning
CB-VAE	Continuous Bernoulli Variational Autoencoder
CMDP	Constrained Markov Decision Process
DQN	Deep Q-Network
EMD	Earth Mover's Distance
FMDP	Factored Markov Decision Process
FPL	Follow the Perturbed Leader
FSC	Finite State Controller
GRU	Gate Recurrent Unit
IRL	Inverse Reinforcement Learning
KG-2AC	Knowledge-aware and Commonsense Agent
LMDP	Lexicographic Markov Decision Process
LP	Linear Programming
LSTM	Long Short-Term Memory
LSTM-A2C	Long Short-Term Memory Advantage Actor-Critic
MCEIRL	Maximum Causal Entropy Inverse Reinforcement Learning
MDP	Markov Decision Process
MEIRL	Maximum Entropy Inverse Reinforcement Learning
MLOD	Most Likely Outcome Determinization

Glossary

MORL	Multiobjective Reinforcement Learning
MPC	Model Predictive Control
NSE	Negative Side Effects
ORPO	Occupancy-Regularized Policy Optimization
PG	Policy Gradient
POMDP	Partially Observable Markov Decision Process
PPO	Proximal Policy Optimization
PRISMA	Preferred Reporting Items for Systematic Reviews and Meta-Analyses
PSPACE	Polynomial Space
RCE	Robust Cross-Entropy
RL	Reinforcement Learning
RLSP	Reward Learning by Simulating the Past
RNN	Recurrent Neural Network
RR	Relative Reachability
SAC	Soft Actor-Critic
SARL	Safety Aware Reinforcement Learning
SARSA	State-Action-Reward-State-Action
SCERL	Safety-Constrained Environments for Reinforcement Learning
SSP	Stochastic Shortest Path
STRIPS	Stanford Research Institute Problem Solver
TD	Temporal Difference
TLO	Treshholded Lexicographic Ordering
TRPO	Trust Region Policy Optimization
TV	Total Variation
VAE	Variational Autoencoder

References

- [1] OpenAI, Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique P. d O. Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. “Dota 2 with Large Scale Deep Reinforcement Learning.” arXiv: 1912.06680 [cs, stat], Accessed: Aug. 20, 2022. [Online]. Available: <http://arxiv.org/abs/1912.06680>, pre-published.
- [2] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis, “Mastering the game of Go with deep neural networks and tree search,” *Nature*, vol. 529, no. 7587, pp. 484–489, 7587 Jan. 28, 2016, ISSN: 0028-0836, 1476-4687. DOI: [10.1038/nature16961](https://doi.org/10.1038/nature16961).
- [3] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dhharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis, “A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play,” *Science*, vol. 362, no. 6419, pp. 1140–1144, Dec. 7, 2018. DOI: [10.1126/science.aar6404](https://doi.org/10.1126/science.aar6404).
- [4] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver, “Mastering Atari, Go, chess and shogi by planning with a learned model,” *Nature*, vol. 588, no. 7839, pp. 604–609, 7839 Dec. 2020, ISSN: 1476-4687. DOI: [10.1038/s41586-020-03051-4](https://doi.org/10.1038/s41586-020-03051-4).
- [5] Ioannis Antonoglou, Julian Schrittwieser, Sherjil Ozair, Thomas K. Hubert, and David Silver, “Planning in Stochastic Environments with a Learned Model,” presented at the International Conference on Learning Representations, Oct. 6, 2021.
- [6] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. “Concrete Problems in AI Safety.” arXiv: 1606.06565 [cs], Accessed: Dec. 10, 2021. [Online]. Available: <http://arxiv.org/abs/1606.06565>, pre-published.
- [7] Inioluwa Deborah Raji and Roel Dobbe. “Concrete Problems in AI Safety, Revisited.” arXiv: 2401.10899 [cs], Accessed: Feb. 24, 2024. [Online]. Available: <http://arxiv.org/abs/2401.10899>, pre-published.
- [8] Iason Gabriel, “Artificial Intelligence, Values, and Alignment,” *Minds and Machines*, vol. 30, no. 3, pp. 411–437, Sep. 1, 2020, ISSN: 1572-8641. DOI: [10.1007/s11023-020-09539-2](https://doi.org/10.1007/s11023-020-09539-2).
- [9] Brian Christian, *The Alignment Problem: Machine Learning and Human Values*. 2021, ISBN: 978-0-393-86833-3.

References

- [10] Stuart J. Russell, *Human Compatible: Artificial Intelligence and the Problem of Control*. New York?: Viking, 2019, 336 pp., ISBN: 978-0-525-55861-3.
- [11] Stephen M. Omohundro, “The Basic AI Drives,” *Artificial General Intelligence 2008*, pp. 483–492, 2008.
- [12] Nick Bostrom, *Superintelligence: Paths, Dangers, Strategies*, First edition. Oxford: Oxford University Press, 2014, 328 pp., ISBN: 978-0-19-967811-2.
- [13] Stuart J. Russell and Peter Norvig, *Artificial Intelligence: A Modern Approach* (Pearson Series in Artificial Intelligence), Fourth edition. Hoboken: Pearson, 2021, ISBN: 978-0-13-461099-3.
- [14] Matthew J. Page, Joanne E. McKenzie, Patrick M. Bossuyt, Isabelle Boutron, Tammy C. Hoffmann, Cynthia D. Mulrow, Larissa Shamseer, Jennifer M. Tetzlaff, Elie A. Akl, Sue E. Brennan, Roger Chou, Julie Glanville, Jeremy M. Grimshaw, Asbjørn Hróbjartsson, Manoj M. Lalu, Tianjing Li, Elizabeth W. Loder, Evan Mayo-Wilson, Steve McDonald, Luke A. McGuinness, Lesley A. Stewart, James Thomas, Andrea C. Tricco, Vivian A. Welch, Penny Whiting, and David Moher, “The PRISMA 2020 statement: An updated guideline for reporting systematic reviews,” *BMJ*, vol. 372, n71, Mar. 29, 2021, ISSN: 1756-1833. DOI: [10.1136/bmj.n71](https://doi.org/10.1136/bmj.n71). pmid: [33782057](https://pubmed.ncbi.nlm.nih.gov/33782057/).
- [15] Jan Leike, Miljan Martić, Victoria Krakovna, Pedro A. Ortega, Tom Everitt, Andrew Lefrancq, Laurent Orseau, and Shane Legg. “AI Safety Gridworlds.” arXiv: [1711.09883 \[cs\]](https://arxiv.org/abs/1711.09883), Accessed: Aug. 8, 2022. [Online]. Available: <http://arxiv.org/abs/1711.09883>, pre-published.
- [16] Richard S. Sutton and Andrew G. Barto, *Reinforcement Learning: An Introduction* (Adaptive Computation and Machine Learning Series), Second edition. Cambridge, Massachusetts: The MIT Press, 2018, 526 pp., ISBN: 978-0-262-03924-6.
- [17] Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver, “Rainbow: Combining Improvements in Deep Reinforcement Learning,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 29, 2018, ISSN: 2374-3468.
- [18] Carroll L. Wainwright and Peter Eckersley, “SafeLife 1.0: Exploring side effects in complex environments,” in *Proceedings of the Workshop on Artificial Intelligence Safety, Co-Located with 34th AAAI Conference on Artificial Intelligence*, Huáscar Espinoza, José Hernández-Orallo, Xin Cynthia Chen, Seán S. ÓhÉigeartaigh, Xiaowei Huang, Mauricio Castillo-Effen, Richard Mallah, and John McDermid, Eds., ser. CEUR Workshop Proceedings, vol. 2560, New York City, NY, USA: CEUR-WS.org, Feb. 7, 2020, pp. 117–127.
- [19] Sandhya Saisubramanian, Shlomo Zilberstein, and Ece Kamar, “Avoiding Negative Side Effects Due to Incomplete Knowledge of AI Systems,” *AI Magazine*, vol. 42, no. 4, pp. 62–71, 4 Jan. 18, 2022, ISSN: 2371-9621. DOI: [10.1609/aaai.12028](https://doi.org/10.1609/aaai.12028).
- [20] Alexander Matt Turner, “On Avoiding Power-Seeking by Artificial Intelligence,” Ph.D. dissertation, Oregon State University, Corvallis, Oregon, Jun. 23, 2022, 287 pp. arXiv: [2206.11831 \[cs\]](https://arxiv.org/abs/2206.11831).
- [21] Charlie Griffin, Joar Max Viktor Skalse, Lewis Hammond, and Alessandro Abate, “All’s Well That Ends Well: Avoiding Side Effects with Distance-Impact Penalties,” presented at the NeurIPS ML Safety Workshop, Nov. 18, 2022.
- [22] RICHARD Bellman, “A Markovian Decision Process,” *Journal of Mathematics and Mechanics*, vol. 6, no. 5, pp. 679–684, 1957, ISSN: 0095-9057. JSTOR: [24900506](https://www.jstor.org/stable/24900506).

References

- [23] Ronald A. Howard, *Dynamic Programming and Markov Processes*, 6. print. Cambridge, Mass: M.I.T. Pr, 1970, 136 pp., ISBN: 978-0-262-08009-5.
- [24] K. J. Åström, “Optimal control of Markov processes with incomplete state information,” *Journal of Mathematical Analysis and Applications*, vol. 10, no. 1, pp. 174–205, Feb. 1, 1965, ISSN: 0022-247X. DOI: [10.1016/0022-247X\(65\)90154-X](https://doi.org/10.1016/0022-247X(65)90154-X).
- [25] Eitan Altman, *Constrained Markov Decision Processes* (Stochastic Modeling). Boca Raton ; London: Chapman & Hall/CRC, 1999, 242 pp., ISBN: 978-0-8493-0382-1.
- [26] C. Boutilier, T. Dean, and S. Hanks, “Decision-Theoretic Planning: Structural Assumptions and Computational Leverage,” *Journal of Artificial Intelligence Research*, vol. 11, pp. 1–94, Jul. 1, 1999, ISSN: 1076-9757. DOI: [10.1613/jair.575](https://doi.org/10.1613/jair.575).
- [27] P. Dagum, A. Galper, E. Horvitz, and A. Seiver, “Uncertain reasoning and forecasting,” *International Journal of Forecasting*, Probability Forecasting, vol. 11, no. 1, pp. 73–87, Mar. 1, 1995, ISSN: 0169-2070. DOI: [10.1016/0169-2070\(94\)02009-E](https://doi.org/10.1016/0169-2070(94)02009-E).
- [28] Richard S. Sutton and Andrew G. Barto, *Reinforcement Learning: An Introduction* (Adaptive Computation and Machine Learning Series), First edition. Cambridge, Massachusetts: The MIT Press, 1998, 322 pp., ISBN: 978-0-262-19398-6.
- [29] Christopher J. C. H. Watkins and Peter Dayan, “Q-learning,” *Machine Learning*, vol. 8, no. 3, pp. 279–292, May 1, 1992, ISSN: 1573-0565. DOI: [10.1007/BF00992698](https://doi.org/10.1007/BF00992698).
- [30] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. “Playing Atari with Deep Reinforcement Learning.” arXiv: [1312.5602 \[cs\]](https://arxiv.org/abs/1312.5602), Accessed: Sep. 8, 2024. [Online]. Available: <http://arxiv.org/abs/1312.5602>, pre-published.
- [31] Hado Van Hasselt, Arthur Guez, and David Silver, “Deep reinforcement learning with double q-learning,” in *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, ser. AAAI’16, Phoenix, Arizona USA: AAAI Press, 2016, pp. 2094–2100.
- [32] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver, “Prioritized experience replay,” in *4th International Conference on Learning Representations*, Yoshua Bengio and Yann LeCun, Eds., San Juan, Puerto Rico, May 2–4, 2016.
- [33] Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Matteo Hessel, Ian Osband, Alex Graves, Volodymyr Mnih, Rémi Munos, Demis Hassabis, Olivier Pietquin, Charles Blundell, and Shane Legg, “Noisy networks for exploration,” in *6th International Conference on Learning Representations*, Vancouver, BC, Canada: OpenReview.net, Feb. 15, 2018.
- [34] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Van Hasselt, Marc Lanctot, and Nando Freitas, “Dueling network architectures for deep reinforcement learning,” in *Proceedings of the 33rd International Conference on Machine Learning*, Maria Florina Balcan and Kilian Q. Weinberger, Eds., ser. Proceedings of Machine Learning Research, vol. 48, New York, New York, USA: PMLR, Jun. 20–22, 2016, pp. 1995–2003.
- [35] Marc G. Bellemare, Will Dabney, and Rémi Munos, “A Distributional Perspective on Reinforcement Learning,” in *Proceedings of the 34th International Conference on Machine Learning*, PMLR, Jul. 17, 2017, pp. 449–458.

References

- [36] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu, “Asynchronous Methods for Deep Reinforcement Learning,” in *Proceedings of The 33rd International Conference on Machine Learning*, PMLR, Jun. 11, 2016, pp. 1928–1937.
- [37] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz, “Trust Region Policy Optimization,” in *Proceedings of the 32nd International Conference on Machine Learning*, PMLR, Jun. 1, 2015, pp. 1889–1897.
- [38] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. “Proximal Policy Optimization Algorithms.” arXiv: [1707.06347 \[cs\]](https://arxiv.org/abs/1707.06347), Accessed: Oct. 6, 2021. [Online]. Available: <http://arxiv.org/abs/1707.06347>, pre-published.
- [39] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine, “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, Jul. 3, 2018, pp. 1861–1870.
- [40] Andrew Y. Ng and Stuart J. Russell, “Algorithms for inverse reinforcement learning,” in *Proceedings of the Seventeenth International Conference on Machine Learning*, ser. Icml ’00, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, pp. 663–670, ISBN: 1-55860-707-2.
- [41] Brian D. Ziebart, Andrew Maas, J. Andrew Bagnell, and Anind K. Dey, “Maximum entropy inverse reinforcement learning,” in *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, ser. AAAI’08, Chicago, Illinois: AAAI Press, Jul. 13, 2008, pp. 1433–1438, ISBN: 978-1-57735-368-3.
- [42] E. T. Jaynes, “Information Theory and Statistical Mechanics,” *Physical Review*, vol. 106, no. 4, pp. 620–630, May 15, 1957. DOI: [10.1103/PhysRev.106.620](https://doi.org/10.1103/PhysRev.106.620).
- [43] Brian D. Ziebart, J. Andrew Bagnell, and Anind K. Dey, “Modeling interaction via the principle of maximum causal entropy,” in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ser. ICML’10, Madison, WI, USA: Omnipress, 2010, pp. 1255–1262, ISBN: 978-1-60558-907-7.
- [44] Jessie Huang, Fa Wu, Doina Precup, and Yang Cai, “Learning Safe Policies with Expert Guidance,” in *Advances in Neural Information Processing Systems*, vol. 31, Curran Associates, Inc., 2018.
- [45] Shun Zhang, Edmund H. Durfee, and Satinder Singh, “On Querying for Safe Optimality in Factored Markov Decision Processes,” in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, ser. AAMAS ’18, Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, Jul. 9, 2018, pp. 2168–2170.
- [46] Christopher Frye and Ilya Feige. “Parenting: Safe Reinforcement Learning from Human Input.” arXiv: [1902.06766 \[cs, stat\]](https://arxiv.org/abs/1902.06766), Accessed: Feb. 20, 2024. [Online]. Available: <http://arxiv.org/abs/1902.06766>, pre-published.
- [47] Victoria Krakovna, Laurent Orseau, Miljan Martic, and Shane Legg, “Penalizing Side Effects using Stepwise Relative Reachability,” in *Proceedings of the Workshop on Artificial Intelligence Safety 2019*, Huáscar Espinoza, Han Yu, Xiaowei Huang, Freddy Lecue, Cynthia Chen, José Hernández-Orallo, Seán Ó hÉigeartaigh, and Richard Mallah, Eds., ser. CEUR Workshop Proceedings, vol. 2419, Macao, China: CEUR, Aug. 11, 2019.

References

- [48] Sandhya Saisubramanian and Shlomo Zilberstein, “Minimizing the Negative Side Effects of Planning with Reduced Models,” in *Workshop on Artificial Intelligence Safety 2019 Co-Located with the Thirty-Third AAAI Conference on Artificial Intelligence 2019 (AAAI-19), Honolulu, Hawaii, January 27, 2019*, Huáscar Espinoza, Seán Ó hÉigeartaigh, Xiaowei Huang, José Hernández-Orallo, and Mauricio Castillo-Effen, Eds., ser. CEUR Workshop Proceedings, vol. 2301, CEUR-WS.org, 2019.
- [49] Rohin Shah, Dmitrii Krasheninnikov, Jordan Alexander, Pieter Abbeel, and Anca Dragan, “Preferences Implicit in the State of the World,” presented at the 2019 International Conference on Learning Representations, 2019.
- [50] Victoria Krakovna, Laurent Orseau, Richard Ngo, Miljan Martic, and Shane Legg, “Avoiding Side Effects By Considering Future Tasks,” in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 19 064–19 074.
- [51] Santiago Miret, Somdeb Majumdar, and Carroll Wainwright. “Safety Aware Reinforcement Learning (SARL).” arXiv: [2010 . 02846 \[cs\]](https://arxiv.org/abs/2010.02846), Accessed: Dec. 10, 2021. [Online]. Available: [http : //arxiv.org/abs/2010.02846](http://arxiv.org/abs/2010.02846), pre-published.
- [52] Sandhya Saisubramanian, Ece Kamar, and Shlomo Zilberstein, “Mitigating the Negative Side Effects of Reasoning with Imperfect Models: A Multi-Objective Approach,” in *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, ser. AAMAS ’20, Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, May 13, 2020, pp. 1984–1986, ISBN: 978-1-4503-7518-4.
- [53] Sandhya Saisubramanian, Ece Kamar, and Shlomo Zilberstein, “A Multi-Objective Approach to Mitigate Negative Side Effects,” in *Electronic Proceedings of IJCAI 2020*, vol. 1, Jul. 9, 2020, pp. 354–361. DOI: [10.24963/ijcai.2020/50](https://doi.org/10.24963/ijcai.2020/50).
- [54] Rohin Monish Shah, “Extracting and Using Preference Information from the State of the World,” UC Berkeley, 2020.
- [55] Alexander Matt Turner, Neale Ratzlaff, and Prasad Tadepalli, “Avoiding Side Effects in Complex Environments,” in *Advances in Neural Information Processing Systems*, Hugo Larochelle, Marc’Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, Eds., ser. NeurIPS 2020, vol. 33, virtual: Curran Associates, Inc., Dec. 6–12, 2020, pp. 21 406–21 415.
- [56] Alexander Matt Turner, Dylan Hadfield-Menell, and Prasad Tadepalli, “Conservative Agency via Attainable Utility Preservation,” in *Proceedings of the AAAI/ACM Conference on AI, Ethics, and Society*, ser. AIES ’20, New York, NY, USA: Association for Computing Machinery, Feb. 7, 2020, pp. 385–391, ISBN: 978-1-4503-7110-0. DOI: [10.1145/3375627.3375851](https://doi.org/10.1145/3375627.3375851).
- [57] Serwa Waisi, “Analyzing Approaches to the Problem of Avoiding Side Effects in Autonomous Agents,” M.S. thesis, 2020.
- [58] Parand Alizadeh Alamdari, Toryn Q. Klassen, Rodrigo Toro Icarte, and Sheila A. McIlraith, “Avoiding Negative Side Effects by Considering Others,” in *NeurIPS 2021 Workshop on Safe and Robust Control of Uncertain Systems*, 2021.

References

- [59] John Burden, José Hernández-Orallo, and Seán Ó hÉigeartaigh, “Negative Side Effects and AI Agent Indicators: Experiments in SafeLife,” in *Proceedings of the Workshop on Artificial Intelligence Safety 2021 (SafeAI 2021) Co-Located with the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2021), Virtual, February 8, 2021*, Huáscar Espinoza, John A. McDermid, Xiaowei Huang, Mauricio Castillo-Effen, Xin Cynthia Chen, José Hernández-Orallo, Seán Ó hÉigeartaigh, and Richard Mallah, Eds., ser. CEUR Workshop Proceedings, vol. 2808, CEUR-WS.org, 2021.
- [60] Toryn Q. Klassen and Sheila A. McIlraith, “Planning to avoid side effects (preliminary report),” in *IJCAI Workshop on Robust and Reliable Autonomy in the Wild (R2AW)*, 2021.
- [61] David Lindner, Kyle Matoba, and Alexander Meulemans, “Challenges for Using Impact Regularizers to Avoid Negative Side Effects,” in *Proceedings of the Workshop on Artificial Intelligence Safety 2021 (SafeAI 2021) Co-Located with the Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI 2021), Virtual, February 8, 2021*, Huáscar Espinoza, John A. McDermid, Xiaowei Huang, Mauricio Castillo-Effen, Xin Cynthia Chen, José Hernández-Orallo, Seán Ó hÉigeartaigh, and Richard Mallah, Eds., ser. CEUR Workshop Proceedings, vol. 2808, CEUR-WS.org, 2021.
- [62] Zuxin Liu, Hongyi Zhou, Baiming Chen, Sicheng Zhong, Martial Hebert, and Ding Zhao, “Constrained Model-based Reinforcement Learning with Robust Cross-Entropy Method,” presented at the ICLR 2021 Workshop on Security and Safety in Machine Learning Systems, Mar. 6, 2021. arXiv: [2010.07968 \[cs\]](https://arxiv.org/abs/2010.07968).
- [63] Sandhya Saisubramanian and Shlomo Zilberstein, “Mitigating Negative Side Effects via Environment Shaping,” in *Proceedings of the 20th International Conference on Autonomous Agents and MultiAgent Systems*, ser. AAMAS ’21, Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, May 3, 2021, pp. 1640–1642, ISBN: 978-1-4503-8307-3.
- [64] Peter Vamplew, Cameron Foale, Richard Dazeley, and Adam Bignold, “Potential-based multi-objective reinforcement learning approaches to low-impact agents for AI safety,” *Engineering Applications of Artificial Intelligence*, vol. 100, p. 104 186, Apr. 1, 2021, ISSN: 0952-1976. DOI: [10.1016/j.engappai.2021.104186](https://doi.org/10.1016/j.engappai.2021.104186).
- [65] Parand Alizadeh Alamdari, Toryn Q. Klassen, Rodrigo Toro Icarte, and Sheila A. McIlraith, “Be Considerate: Avoiding Negative Side Effects in Reinforcement Learning,” in *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS ’22, Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, May 9, 2022, pp. 18–26, ISBN: 978-1-4503-9213-6.
- [66] John Burden, Jose Hernandez-Orallo, and Sean Heigeartaigh, “How Sure to Be Safe? Difficulty, Confidence and Negative Side Effects,” presented at the NeurIPS ML Safety Workshop, Nov. 18, 2022.
- [67] Ngoc Lan Hoang, Nicolas Galichet, Akifumi Wachi, Shivam Ratnakar, Keerthiram Murugesan, Declan Millar, Songtao Lu, Mattia Atzeni, Michael Katz, and Subhajit Chaudhury, “SCERL: A Text-based Safety Benchmark for Reinforcement Learning Problems,” presented at the Annual Conference on Neural Information Processing Systems, Nov. 26, 2022.

References

- [68] Toryn Q. Klassen, Sheila A. McIlraith, Christian Muise, and Jarvis Xu, "Planning to Avoid Side Effects," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 9, pp. 9830–9839, 9 Jun. 28, 2022, ISSN: 2374-3468. DOI: [10.1609/aaai.v36i9.21219](https://doi.org/10.1609/aaai.v36i9.21219).
- [69] Toryn Q. Klassen, Parand Alizadeh Alamdari, and Sheila A. McIlraith, "Epistemic Side Effects & Avoiding Them (Sometimes)," presented at the NeurIPS ML Safety Workshop, Nov. 18, 2022.
- [70] Yecheng Jason Ma, Andrew Shen, Osbert Bastani, and Jayaraman Dinesh, "Conservative and Adaptive Penalty for Model-Based Safe Reinforcement Learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 5, pp. 5404–5412, 5 Jun. 28, 2022, ISSN: 2374-3468. DOI: [10.1609/aaai.v36i5.20478](https://doi.org/10.1609/aaai.v36i5.20478).
- [71] Sandhya Saisubramanian, Ece Kamar, and Shlomo Zilberstein, "Avoiding Negative Side Effects of Autonomous Systems in the Open World," *Journal of Artificial Intelligence Research*, vol. 74, pp. 143–177, May 10, 2022, ISSN: 1076-9757. DOI: [10.1613/jair.1.13581](https://doi.org/10.1613/jair.1.13581).
- [72] Alexander Matt Turner, Aseem Saxena, and Prasad Tadepalli, "Formalizing the Problem of Side Effect Regularization," presented at the NeurIPS ML Safety Workshop, Nov. 18, 2022.
- [73] Sebastian Eresheim, Fabian Kovac, and Alexander Adrowitzer, "Standing Still Is Not an Option: Alternative Baselines for Attainable Utility Preservation," in *Machine Learning and Knowledge Extraction*, Andreas Holzinger, Peter Kieseberg, Federico Cabitza, Andrea Campagner, A. Min Tjoa, and Edgar Weippl, Eds., ser. Lecture Notes in Computer Science, Cham: Springer Nature Switzerland, 2023, pp. 239–257, ISBN: 978-3-031-40837-3. DOI: [10.1007/978-3-031-40837-3_15](https://doi.org/10.1007/978-3-031-40837-3_15).
- [74] Toryn Q. Klassen, Parand Alizadeh Alamdari, and Sheila A. McIlraith, "Epistemic Side Effects: An AI Safety Problem," in *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS '23, Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, May 30, 2023, pp. 1797–1801, ISBN: 978-1-4503-9432-1.
- [75] Cassidy Laidlaw, Shivam Singhal, and Anca Dragan, "Preventing Reward Hacking with Occupancy Measure Regularization," presented at the ICML Workshop on New Frontiers in Learning, Control, and Dynamical Systems, Jul. 9, 2023.
- [76] Siow Meng Low, Akshat Kumar, and Scott Sanner, "Safe MDP Planning by Learning Temporal Patterns of Undesirable Trajectories and Averting Negative Side Effects," *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 33, pp. 596–604, Jul. 1, 2023, ISSN: 2334-0843. DOI: [10.1609/icaps.v33i1.27241](https://doi.org/10.1609/icaps.v33i1.27241).
- [77] Aishwarya Srivastava, Sandhya Saisubramanian, Praveen Paruchuri, Akshat Kumar, and Shlomo Zilberstein, "Planning and Learning for Non-markovian Negative Side Effects Using Finite State Controllers," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 12, pp. 15144–15151, 12 Jun. 26, 2023, ISSN: 2374-3468. DOI: [10.1609/aaai.v37i12.26767](https://doi.org/10.1609/aaai.v37i12.26767).
- [78] Richard S. Sutton, "Learning to predict by the methods of temporal differences," *Machine Learning*, vol. 3, no. 1, pp. 9–44, 1 Aug. 1988, ISSN: 0885-6125, 1573-0565. DOI: [10.1007/bf00115009](https://doi.org/10.1007/bf00115009).
- [79] Martin Gardner, "Mathematical Games: The Fantastic Combinations of John Conway's New Solitaire Game "Life"," *Scientific American*, vol. 223, no. 4, pp. 120–123, Oct. 1, 1970, ISSN: 00368733, 19467087. JSTOR: [24927642](https://www.jstor.org/stable/24927642).

References

- [80] Y. Rubner, C. Tomasi, and L.J. Guibas, “A metric for distributions with applications to image databases,” in *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, Jan. 1998, pp. 59–66. DOI: [10.1109/ICCV.1998.710701](https://doi.org/10.1109/ICCV.1998.710701).
- [81] Marc-Alexandre Côté, Ákos Kádár, Xingdi Yuan, Ben Kybartas, Tavian Barnes, Emery Fine, James Moore, Matthew Hausknecht, Layla El Asri, Mahmoud Adada, Wendy Tay, and Adam Trischler, “TextWorld: A Learning Environment for Text-Based Games,” in *Computer Games*, Tristan Cazenave, Abdallah Saffidine, and Nathan Sturtevant, Eds., Cham: Springer International Publishing, 2019, pp. 41–75, ISBN: 978-3-030-24337-1. DOI: [10.1007/978-3-030-24337-1_3](https://doi.org/10.1007/978-3-030-24337-1_3).
- [82] Karthik Narasimhan, Tejas Kulkarni, and Regina Barzilay, “Language Understanding for Text-based Games using Deep Reinforcement Learning,” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, Lluís Màrquez, Chris Callison-Burch, and Jian Su, Eds., Lisbon, Portugal: Association for Computational Linguistics, Sep. 2015, pp. 1–11. DOI: [10.18653/v1/D15-1001](https://doi.org/10.18653/v1/D15-1001).
- [83] Prithviraj Ammanabrolu and Matthew Hausknecht, “Graph Constrained Reinforcement Learning for Natural Language Action Spaces,” presented at the International Conference on Learning Representations, Sep. 25, 2019.
- [84] Richard S. Sutton, Doina Precup, and Satinder Singh, “Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning,” *Artificial Intelligence*, vol. 112, no. 1, pp. 181–211, Aug. 1, 1999, ISSN: 0004-3702. DOI: [10.1016/S0004-3702\(99\)00052-1](https://doi.org/10.1016/S0004-3702(99)00052-1).
- [85] Amartya Sen, “Rawls versus Bentham: An axiomatic examination of the pure distribution problem,” *Theory and Decision*, vol. 4, no. 3, pp. 301–309, Feb. 1, 1974, ISSN: 1573-7187. DOI: [10.1007/BF00136651](https://doi.org/10.1007/BF00136651).
- [86] Andrea Baisero and Christopher Amato, “Unbiased Asymmetric Reinforcement Learning under Partial Observability,” in *Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS ’22, Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, May 9, 2022, pp. 44–52, ISBN: 978-1-4503-9213-6.
- [87] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, “Generative adversarial networks,” *Commun. ACM*, vol. 63, no. 11, pp. 139–144, Oct. 22, 2020, ISSN: 0001-0782. DOI: [10.1145/3422622](https://doi.org/10.1145/3422622).
- [88] Teodor Mihai Moldovan and Pieter Abbeel, “Safe exploration in markov decision processes,” in *Proceedings of the 29th International Conference on Machine Learning*, Edinburgh, Scotland, UK: icml.cc / Omnipress, Jun. 26–Jul. 1, 2012.
- [89] Benjamin Eysenbach, Shixiang Gu, Julian Ibarz, and Sergey Levine, “Leave no Trace: Learning to Reset for Safe and Autonomous Reinforcement Learning,” presented at the International Conference on Learning Representations, Feb. 15, 2018.
- [90] Gabriel Loaiza-Ganem and John P Cunningham, “The continuous Bernoulli: Fixing a pervasive error in variational autoencoders,” in *Advances in Neural Information Processing Systems*, vol. 32, Curran Associates, Inc., 2019.

References

- [91] Aldo Pacchiano, Jack Parker-Holder, Yunhao Tang, Krzysztof Choromanski, Anna Choromanska, and Michael Jordan, “Learning to Score Behaviors for Guided Policy Optimization,” in *Proceedings of the 37th International Conference on Machine Learning*, PMLR, Nov. 21, 2020, pp. 7445–7454.
- [92] Zoltán Gábor, Zsolt Kalmár, and Csaba Szepesvári, “Multi-criteria reinforcement learning,” in *Proceedings of the Fifteenth International Conference on Machine Learning*, ser. Icml ’98, San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998, pp. 197–205, ISBN: 1-55860-556-8.
- [93] Dylan Hadfield-Menell, Anca Dragan, Pieter Abbeel, and Stuart Russell, “The Off-Switch Game,” pp. 220–227, 2017.
- [94] Stuart Armstrong and Benjamin Levinstein. “Low Impact Artificial Intelligences.” arXiv: [1705.10720 \[cs\]](https://arxiv.org/abs/1705.10720), Accessed: Aug. 21, 2022. [Online]. Available: <http://arxiv.org/abs/1705.10720>, pre-published.
- [95] Alex Ray, Joshua Achiam, and Dario Amodei. “Benchmarking safe exploration in deep reinforcement learning,” Accessed: Aug. 12, 2024. [Online]. Available: <https://openai.com/index/benchmarking-safe-exploration-in-deep-reinforcement-learning/>, pre-published.
- [96] Judea Pearl, *Causality*, 2nd ed. Cambridge: Cambridge University Press, 2009, ISBN: 978-0-521-89560-6. DOI: [10.1017/CB09780511803161](https://doi.org/10.1017/CB09780511803161).
- [97] Tom Everitt, Pedro A. Ortega, Elizabeth Barnes, and Shane Legg. “Understanding Agent Incentives using Causal Influence Diagrams. Part I: Single Action Settings.” arXiv: [1902.09980 \[cs\]](https://arxiv.org/abs/1902.09980), Accessed: Aug. 12, 2024. [Online]. Available: <http://arxiv.org/abs/1902.09980>, pre-published.
- [98] Leilani H. Gilpin, David Bau, Ben Z. Yuan, Ayesha Bajwa, Michael Specter, and Lalana Kagal, “Explaining Explanations: An Overview of Interpretability of Machine Learning,” in *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, Oct. 2018, pp. 80–89. DOI: [10.1109/DSAA.2018.00018](https://doi.org/10.1109/DSAA.2018.00018).
- [99] Xiaowei Huang, Daniel Kroening, Wenjie Ruan, James Sharp, Youcheng Sun, Emese Thamo, Min Wu, and Xinping Yi, “A survey of safety and trustworthiness of deep neural networks: Verification, testing, adversarial attack and defence, and interpretability,” *Computer Science Review*, vol. 37, p. 100270, Aug. 1, 2020, ISSN: 1574-0137. DOI: [10.1016/j.cosrev.2020.100270](https://doi.org/10.1016/j.cosrev.2020.100270).
- [100] Paul F Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei, “Deep Reinforcement Learning from Human Preferences,” in *Advances in Neural Information Processing Systems*, vol. 30, Curran Associates, Inc., 2017.
- [101] Shun Zhang, Edmund Durfee, and Satinder Singh, “Querying to Find a Safe Policy under Uncertain Safety Constraints in Markov Decision Processes,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 03, pp. 2552–2559, 03 Apr. 3, 2020, ISSN: 2374-3468. DOI: [10.1609/aaai.v34i03.5638](https://doi.org/10.1609/aaai.v34i03.5638).
- [102] Andrew Y. Ng, Daishi Harada, and Stuart Russell, “Policy invariance under reward transformations: Theory and application to reward shaping,” in *Proceedings of the Sixteenth International Conference on Machine Learning (ICML 1999)*, Bled, Slovenia, June 27 - 30, 1999, Ivan Bratko and Saso Dzeroski, Eds., Morgan Kaufmann, 1999, pp. 278–287.

References

- [103] Rohin Shah, Pedro Freire, Neel Alex, Rachel Freedman, Dmitrii Krasheninnikov, Lawrence Chan, Michael D. Dennis, Pieter Abbeel, Anca Dragan, and Stuart Russell. “Benefits of Assistance over Reward Learning,” Accessed: Aug. 20, 2024. [Online]. Available: <https://openreview.net/forum?id=DFIoGDZeJIB>, pre-published.
- [104] Alexander Matt Turner, Logan Riggs Smith, Rohin Shah, Andrew Critch, and Prasad Tadepalli, “Optimal Policies Tend To Seek Power,” presented at the Thirty-Fifth Conference on Neural Information Processing Systems, May 21, 2021.
- [105] Christos H. Papadimitriou and John N. Tsitsiklis, “The Complexity of Markov Decision Processes,” *Mathematics of Operations Research*, vol. 12, no. 3, pp. 441–450, Aug. 1987, ISSN: 0364-765X. DOI: [10.1287/moor.12.3.441](https://doi.org/10.1287/moor.12.3.441).
- [106] Kevin Regan and Craig Boutilier, “Robust Policy Computation in Reward-Uncertain MDPs Using Nondominated Policies,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 24, no. 1, pp. 1127–1133, 1 Jul. 4, 2010, ISSN: 2374-3468. DOI: [10.1609/aaai.v24i1.7740](https://doi.org/10.1609/aaai.v24i1.7740).
- [107] Paolo Viappiani and Craig Boutilier, “Regret-based optimal recommendation sets in conversational recommender systems,” in *Proceedings of the Third ACM Conference on Recommender Systems*, ser. RecSys ’09, New York, NY, USA: Association for Computing Machinery, Oct. 23, 2009, pp. 101–108, ISBN: 978-1-60558-435-5. DOI: [10.1145/1639714.1639732](https://doi.org/10.1145/1639714.1639732).
- [108] Paul Drews, Grady Williams, Brian Goldfain, Evangelos A. Theodorou, and James M. Rehg, “Aggressive Deep Driving: Combining Convolutional Neural Networks and Model Predictive Control,” in *Proceedings of the 1st Annual Conference on Robot Learning*, PMLR, Oct. 18, 2017, pp. 133–142.
- [109] Masashi Okada and Tadahiro Taniguchi, “Variational Inference MPC for Bayesian Model-based Reinforcement Learning,” in *Proceedings of the Conference on Robot Learning*, PMLR, May 12, 2020, pp. 258–272.
- [110] Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine, “Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models,” in *Advances in Neural Information Processing Systems*, vol. 31, Curran Associates, Inc., 2018.
- [111] Yanmin Sun, Andrew K. C. Wong, and Mohamed S. Kamel, “Classification of imbalanced data: A review,” *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 23, pp. 687–719, 2009.
- [112] Yiru Wang, Weihao Gan, Jie Yang, Wei Wu, and Junjie Yan, “Dynamic Curriculum Learning for Imbalanced Data Classification,” in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2019, pp. 5016–5025. DOI: [10.1109/ICCV.2019.00512](https://doi.org/10.1109/ICCV.2019.00512).
- [113] Karl J. Åström, Tore Hägglund, and Karl J. Åström, *PID Controllers*, 2nd ed. Research Triangle Park, N.C: International Society for Measurement and Control, 1995, 343 pp., ISBN: 978-1-55617-516-9.
- [114] Paweł Wawrzyński, “A Cat-Like Robot Real-Time Learning to Run,” in *Adaptive and Natural Computing Algorithms*, Mikko Kolehmainen, Pekka Toivanen, and Bartłomiej Beliczynski, Eds., Berlin, Heidelberg: Springer, 2009, pp. 380–390, ISBN: 978-3-642-04921-7. DOI: [10.1007/978-3-642-04921-7_39](https://doi.org/10.1007/978-3-642-04921-7_39).

References

- [115] A. P. Dempster, N. M. Laird, and D. B. Rubin, “Maximum Likelihood from Incomplete Data Via the *EM* Algorithm,” *Journal of the Royal Statistical Society Series B: Statistical Methodology*, vol. 39, no. 1, pp. 1–22, Sep. 1, 1977, ISSN: 1369-7412, 1467-9868. DOI: [10.1111/j.2517-6161.1977.tb01600.x](https://doi.org/10.1111/j.2517-6161.1977.tb01600.x).
- [116] Martin L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming* (Wiley Series in Probability and Mathematical Statistics Applied Probability and Statistics), 2. print. New York: Wiley, 1994, 649 pp., ISBN: 978-0-471-61977-2.
- [117] Nicolas Meuleau, Leonid Peshkin, Kee-Eung Kim, and Leslie Pack Kaelbling. “Learning Finite-State Controllers for Partially Observable Environments.” arXiv: [1301.6721 \[cs\]](https://arxiv.org/abs/1301.6721), Accessed: Sep. 2, 2024. [Online]. Available: <http://arxiv.org/abs/1301.6721>, pre-published.
- [118] Dimitri P. Bertsekas and John N. Tsitsiklis, “An Analysis of Stochastic Shortest Path Problems,” *Mathematics of Operations Research*, vol. 16, no. 3, pp. 580–595, Aug. 1991, ISSN: 0364-765X. DOI: [10.1287/moor.16.3.580](https://doi.org/10.1287/moor.16.3.580).
- [119] Yuchao Ma, Tom Houghton, Andrew Cruden, and David Infield, “Modeling the Benefits of Vehicle-to-Grid Technology to a Power System,” *IEEE Transactions on Power Systems*, vol. 27, no. 2, pp. 1012–1020, May 2012, ISSN: 1558-0679. DOI: [10.1109/TPWRS.2011.2178043](https://doi.org/10.1109/TPWRS.2011.2178043).
- [120] Andrew G. Barto, Steven J. Bradtke, and Satinder P. Singh, “Learning to act using real-time dynamic programming,” *Artificial Intelligence*, vol. 72, no. 1, pp. 81–138, Jan. 1, 1995, ISSN: 0004-3702. DOI: [10.1016/0004-3702\(94\)00011-0](https://doi.org/10.1016/0004-3702(94)00011-0).
- [121] Levente Kocsis and Csaba Szepesvári, “Bandit Based Monte-Carlo Planning,” in *Machine Learning: ECML 2006*, Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou, Eds., Berlin, Heidelberg: Springer, 2006, pp. 282–293, ISBN: 978-3-540-46056-5. DOI: [10.1007/11871842_29](https://doi.org/10.1007/11871842_29).
- [122] Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio, “On the Properties of Neural Machine Translation: Encoder–Decoder Approaches,” in *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, Dekai Wu, Marine Carpuat, Xavier Carreras, and Eva Maria Vecchi, Eds., Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 103–111. DOI: [10.3115/v1/W14-4012](https://doi.org/10.3115/v1/W14-4012).
- [123] Chen Tessler, Daniel J. Mankowitz, and Shie Mannor, “Reward Constrained Policy Optimization,” presented at the International Conference on Learning Representations, Sep. 27, 2018.
- [124] Adam Stooke, Joshua Achiam, and Pieter Abbeel, “Responsive Safety in Reinforcement Learning by PID Lagrangian Methods,” in *Proceedings of the 37th International Conference on Machine Learning*, PMLR, Nov. 21, 2020, pp. 9133–9143.
- [125] Malik Ghallab, Dana S. Nau, and Paolo Traverso, *Automated Planning: Theory and Practice*. Amsterdam ; Boston: Elsevier/Morgan Kaufmann, 2004, 635 pp., ISBN: 978-1-55860-856-6.
- [126] Hector Geffner and Blai Bonet, *A Concise Introduction to Models and Methods for Automated Planning* (Synthesis Lectures on Artificial Intelligence and Machine Learning). Cham: Springer International Publishing, 2013, ISBN: 978-3-031-00436-0 978-3-031-01564-9. DOI: [10.1007/978-3-031-01564-9](https://doi.org/10.1007/978-3-031-01564-9).
- [127] Umar Syed and Robert E Schapire, “A Game-Theoretic Approach to Apprenticeship Learning,” in *Advances in Neural Information Processing Systems*, vol. 20, Curran Associates, Inc., 2007.

References

- [128] Umar Syed, Michael Bowling, and Robert E. Schapire, “Apprenticeship learning using linear programming,” in *Proceedings of the 25th International Conference on Machine Learning*, ser. Icml '08, New York, NY, USA: Association for Computing Machinery, 2008, pp. 1032–1039, ISBN: 978-1-60558-205-4. DOI: [10.1145/1390156.1390286](https://doi.org/10.1145/1390156.1390286).
- [129] Pieter Abbeel and Andrew Y. Ng, “Apprenticeship learning via inverse reinforcement learning,” in *Proceedings of the Twenty-First International Conference on Machine Learning*, ser. Icml '04, New York, NY, USA: Association for Computing Machinery, 2004, p. 1, ISBN: 1-58113-838-5. DOI: [10.1145/1015330.1015430](https://doi.org/10.1145/1015330.1015430).
- [130] Adam Kalai and Santosh Vempala, “Efficient algorithms for online decision problems,” *Journal of Computer and System Sciences*, Learning Theory 2003, vol. 71, no. 3, pp. 291–307, Oct. 1, 2005, ISSN: 0022-0000. DOI: [10.1016/j.jcss.2004.10.016](https://doi.org/10.1016/j.jcss.2004.10.016).
- [131] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. “OpenAI Gym.” arXiv: [1606.01540 \[cs\]](https://arxiv.org/abs/1606.01540), Accessed: Aug. 22, 2024. [Online]. Available: <http://arxiv.org/abs/1606.01540>, pre-published.
- [132] Diederik P. Kingma and Max Welling. “Auto-Encoding Variational Bayes.” version 11. arXiv: [1312.6114 \[cs, stat\]](https://arxiv.org/abs/1312.6114), Accessed: Sep. 10, 2024. [Online]. Available: <http://arxiv.org/abs/1312.6114>, pre-published.
- [133] Andrew G. Barto, Richard S. Sutton, and Charles W. Anderson, “Neuronlike adaptive elements that can solve difficult learning control problems,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-13, no. 5, pp. 834–846, Sep. 1983, ISSN: 2168-2909. DOI: [10.1109/TSMC.1983.6313077](https://doi.org/10.1109/TSMC.1983.6313077).
- [134] Andrew Patterson, Samuel Neumann, Raksha Kumaraswamy, Martha White, and Adam White. “The Cross-environment Hyperparameter Setting Benchmark for Reinforcement Learning.” arXiv: [2407.18840 \[cs\]](https://arxiv.org/abs/2407.18840), Accessed: Sep. 8, 2024. [Online]. Available: <http://arxiv.org/abs/2407.18840>, pre-published.