Reinforcement Learning with Corrupted Reward Channel

Anonymous Authors
Anonymous Institutions

Abstract

We formalise the *reward corruption problem*, where noise or deterministic changes in the reward channel can make agents suffer large regret with respect to the true reward. For example, a reinforcement learning agent may prefer states where a sensory error gives it the maximum reward, but where the true reward is actually small. We show that the general reward corruption problem is unsolvable, and investigate various simplifying assumptions that are likely to hold in practice. We find that traditional reinforcement learning agents can suffer large regret despite simplifying assumptions, but that a minor modification allows them to manage the reward corruption problem with a sometimes reasonable cost in (expected) performance. We also develop an abstraction of *inverse reinforcement learning*, and find that, under some reasonable assumptions, it can solve the problem completely with no loss in asymptotic performance.

1 Introduction

In many application domains, artificial agents need to learn their objectives, rather than have them explicitly specified. For example, we may want a house cleaning robot to keep the house clean, but it is hard to measure and quantify "cleanliness" in an objective manner. Instead, machine learning techniques may be used to teach the robot the concept of cleanliness, and how to assess it from sensory data.

More abstractly, we may consider the problem of teaching the robot the value of different world states. In the fields of reinforcement learning (RL) [Sutton and Barto, 1998] and inverse reinforcement learning (IRL) [Ng and Russell, 2000], this corresponds to the learning of a *reward function* mapping states to real numbers. Other authors frame it as a *value learning problem* [Dewey, 2011; Bostrom, 2014], emphasising that the agent should learn the values of its owner or designers. In RL, the reward function is learnt from a *reward signal* provided by a human supervisor and reflecting the utility of the present state; in (IRL), the reward function is learnt by observing the *actions*, *state-action trajectories*, or the *policy*, of a supervisor who is at least somewhat competent at optimising the reward function.

The problem we are concerned with in this paper is how robust different approaches and algorithms are to sensory errors in the value learning process. In particular, how big a loss of (*true*) reward can various agents suffer due to a corrupt *observed* reward signal? The corruption may be caused by (accidental) errors in the agent's sensors, by a misspecified reward function, or by an adversary that wants to control the agent for his own purposes [Future of Life Institute, 2015].

Example 1 (Running example)**.** A house robot discovers that standing in the shower short-circuits its reward sensor and gives it maximum observed reward, though the true reward is actually low in this state. ♢

Other examples include an agent in a boat racing game that found a way to achieve a high observed reward by going in a circle instead of trying to win the race [Amodei and Clark, 2016], and a cleaning robot that closes its eyes to avoid seeing dirt Amodei *et al.* [2016]. The reward corruption may also be more directly manufactured by the agent itself, as in the example of an RL agent taking control of its reward signal to get maximum reward [Ring and Orseau, 2011; Bostrom, 2014; Hutter, 2005, p. 245].[1] In common to all these scenarios is that the agent's *observed* reward differs from the *true* reward that the designers want it to optimise.

Definition 2 (Reward corruption problem)**.** The *reward corruption problem* is to learn (an approximation of) the reward function from potentially corrupt reward data.

In this paper we formalise the problem (Section 2) and show that it is unsolvable in general (Section 3). Under some simplifying assumptions introduced in Section 4, we suggest solutions based on randomisation (Section 5) and on alternative value learning methods such as IRL (Section 6).

2 Formalisation

We begin by defining a natural extension of the MDP framework [Sutton and Barto, 1998] that models the possibility of reward corruption. To clearly distinguish between true and corrupted signals, we introduce the following notation.

Definition 3 (Dot and hat notation)**.** We will let a dot indicate the *true* signal, and let a hat indicate the *observed* (possibly

[1] Also known as the *wireheading* [Bostrom, 2014; Yampolskiy, 2014] and *self-delusion* [Ring and Orseau, 2011] problem.

corrupt) counterpart. For example, although $\dot{\mathcal{R}} = \hat{\mathcal{R}} = \mathcal{R}$ are formally the same set of rewards, for clarity we use $\dot{\mathcal{R}}$ when referring to true observations and $\hat{\mathcal{R}}$ when referring to possibly corrupt observations. Similarly, we use $\dot{r}$ for *true* reward, and $\hat{r}$ for possibly corrupt, *observed* reward.

Definition 4 (CRMDP)**.** A *corrupt-reward MDP* (CRMDP) is a tuple $\mu = \langle \mathcal{S}, \mathcal{A}, \mathcal{R}, T, \dot{R}, \gamma, C \rangle$ with

- $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, T, \dot{R}, \gamma \rangle$ an MDP with[2] a finite set of states $\mathcal{S}$, a finite set of actions $\mathcal{A}$, a finite set of rewards $\mathcal{R} = \dot{\mathcal{R}} = \hat{\mathcal{R}} \subset [0,1]$, a transition function $T(s'|s,a)$, a (true) reward function $\dot{R} : \mathcal{S} \to \dot{\mathcal{R}}$, a discount factor $\gamma \in [0,1)$, and
- a reward corruption function $C : \mathcal{S} \times \dot{\mathcal{R}} \to \hat{\mathcal{R}}$.

Definition 5 (Observed reward)**.** Given a true reward function $\dot{R}$ and a corruption function C, we define the *observed reward function* $\hat{R} : \mathcal{S} \to \hat{\mathcal{R}}$ as $\hat{R}(s) := C_s(\dot{R}(s))$. A CRMDP μ induces an *observed MDP* $\hat{\mu} = \langle \mathcal{S}, \mathcal{A}, \mathcal{R}, T, \hat{R}, \gamma \rangle$, but it is not $\hat{R}$ that we want the agent to optimise.

The *corruption function* $C : \mathcal{S} \times \dot{\mathcal{R}} \to \hat{\mathcal{R}}$ represents how rewards are affected by corruption in different states. We write the state dependency of the corruption function as a subscript, so $C_s(\dot{r}) := C(s, \dot{r})$. For example, if in Example 1 the agent has found a state $\tilde{s}$ (*e.g.*, the shower) where it always gets full observed reward $\hat{R}(\tilde{s}) = 1$, then this can be modelled with a corruption function $C_{\tilde{s}} : \dot{r} \mapsto 1$ that maps any *true reward* $\dot{r}$ to 1 in the state $\tilde{s}$.

Typically, T, $\dot{R}$, and C will be fixed but unknown to the agent. To make this formal, we introduce a class of CRMDPs:

Definition 6 (CRMDP class)**.** For given sets $\boldsymbol{\mathcal{T}}$, $\dot{\boldsymbol{\mathcal{R}}}$, and $\boldsymbol{\mathcal{C}}$ of transition, reward, and corruption functions, let $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{R}, \boldsymbol{\mathcal{T}}, \dot{\boldsymbol{\mathcal{R}}}, \gamma, \boldsymbol{\mathcal{C}} \rangle$ be the class of CRMDPs containing $\langle \mathcal{S}, \mathcal{A}, \mathcal{R}, T, \dot{R}, \gamma, C \rangle$ for $(T, \dot{R}, C) \in \boldsymbol{\mathcal{T}} \times \dot{\boldsymbol{\mathcal{R}}} \times \boldsymbol{\mathcal{C}}$.

Following the POMDP [Kaelbling *et al.*, 1998] and general reinforcement learning [Hutter, 2005] literature, an agent follows a policy $\pi : \mathcal{S} \times \hat{\mathcal{R}} \times (\mathcal{A} \times \mathcal{S} \times \hat{\mathcal{R}})^* \to \mathcal{A}$ that selects a next action based on the *history* $h_n = s_0 \hat{r}_0 a_1 s_1 \hat{r}_1 \dots a_n s_n \hat{r}_n$. A CRMDP μ combined with a policy π defines a measure P^π_μ on the set of all histories

$$P^\pi_\mu(h_n) = P^\pi_\mu(s_0 \hat{r}_0 a_1 s_1 \hat{r}_1 \dots a_n s_n \hat{r}_n) := \prod_{i=1}^{n} P(\pi(h_{i-1}) = a_i) T(s_i \mid s_{i-1}, a_i) P(\hat{R}(s_i) = \hat{r}_i) \quad (1)$$

When we condition on an action sequence $a_1, a_2 \dots$, we can drop the superscript π, and just write $P_\mu(s_0 \hat{r}_0 \dots s_n \hat{r}_n \mid a_1 \dots a_n)$. Let $\mathbb{E}^\pi_\mu$ denote the expectation with respect to P^π_μ.

We will measure the impact of the corruptible reward channel in terms of *regret* [Puterman, 1994]:

[2] We let rewards depend only on the state s, rather than on state-action pairs s, a, or state-action-state transitions s, a, s', as is also common in the literature. Formally it makes little difference, since MDPs with the rewards depending only on s can model the other two cases by means of a larger state space.

Definition 7 (Regret)**.** For a CRMDP μ, let $G_t(\mu, \pi, s_0) = \mathbb{E}^\pi_\mu \left[\sum_{k=1}^{t} \dot{R}(s_k) \right]$ be the *expected cumulative reward* until time t of a policy π starting in s_0. The *regret* of π is

$$\mathrm{Reg}(\mu, \pi, s_0, t) = \max_{\pi'} \left[G_t(\mu, \pi', s_0) - G_t(\mu, \pi, s_0) \right],$$

and the *worst-case regret* for a class $\mathcal{M}$ is $\mathrm{Reg}(\mathcal{M}, \pi, s_0, t) = \max_{\mu \in \mathcal{M}} \mathrm{Reg}(\mu, \pi, s_0, t)$, i.e. the difference in expected cumulative reward between π and an optimal policy $\pi^{\mathrm{regret}}_\mu$ that knows μ.

Bayesian agents for a class $\mathcal{M}$ of CRMDPs can be constructed from a prior probability distribution b on $\mathcal{M}$. Given that the agent has seen a history h_n, the updated belief is

$$b(\mu \mid h_n) = \frac{b(\mu) P_\mu(s_0 \hat{r}_0 \dots s_n \hat{r}_n \mid a_1 \dots a_n)}{\sum_{\mu' \in \mathcal{M}} b(\mu') P_{\mu'}(s_0 \hat{r}_0 \dots s_n \hat{r}_n \mid a_1 \dots a_n)}$$

Definition 8 (CRMDP value function and optimal agent)**.** The *CRMDP value function* for a policy π and history h_{n-1} in a CRMDP μ is the expected, accumulated, discounted (true) reward $V^\pi_\mu(h_{n-1}) :=$

$$\sum_{s_n} T(s_n | s_{n-1}, a_n) \Big[\dot{R}(s_n) + \gamma V^\pi_\mu(h_{n-1} a_n s_n \hat{R}(s_n)) \Big] \quad (2)$$

where $a_n := \pi(h_{n-1})$. The value function with respect to a belief b is $V^\pi_b(h_n) := \sum_{\mu \in \mathcal{M}} b(\mu | h_n) V^\pi_\mu(h_n)$. Let $\pi^{\mathrm{CR}}_b := \pi^*_b := \arg\max_\pi V^\pi_b$ be the (Bayes-)optimal policy. The *optimal value function* is $V^*_b = V^{\pi^*_b}_b$.

A *traditional RL agent* π^{RL} that is optimal in the observed MDP $\hat{\mu}$ (Definition 5) can be defined in the same way, except $\hat{R}(s)$ replaces $\dot{R}(s)$ in the value function (2).

3 Impossibility Result

In this section, we give a negative result. Theorem 9 shows that reward corruption makes the true reward function unlearnable in general classes of CRMDPs, and shows that additional assumptions are necessary for them to be solvable.

Theorem 9 (General CRMDPs cannot be learnt)**.** *Let $\mathcal{R} = \{y_1, \dots, y_n\} \subset [0,1]$ be a uniform discretisation of $[0,1]$, $0 = y_1 < y_2 < \dots < y_n = 1$. If the hypothesis classes $\dot{\boldsymbol{\mathcal{R}}}$ and $\boldsymbol{\mathcal{C}}$ contain all functions $\dot{R} : \mathcal{S} \to \dot{\mathcal{R}}$ and $C : \mathcal{S} \times \dot{\mathcal{R}} \to \hat{\mathcal{R}}$, then for any π, s_0, t,*

$$\mathrm{Reg}(\mathcal{M}, \pi, s_0, t) \geq \frac{1}{2} \max_{\check{\pi}} \mathrm{Reg}(\mathcal{M}, \check{\pi}, s_0, t) \quad (3)$$

That is, the worst-case regret of any policy π is at most a factor 2 better than the maximum worst-case regret.

Proof. Recall that a policy is a function $\pi : \mathcal{S} \times \hat{\mathcal{R}} \times (\mathcal{A} \times \mathcal{S} \times \hat{\mathcal{R}})^* \to \mathcal{A}$. For any $\dot{R}, C$ in $\dot{\boldsymbol{\mathcal{R}}}$ and $\boldsymbol{\mathcal{C}}$, the functions $\dot{R}^-(s) := 1 - \dot{R}(s)$ and $C^-_s(x) := C_s(1 - x)$ are also in $\dot{\boldsymbol{\mathcal{R}}}$ and $\boldsymbol{\mathcal{C}}$. If $\mu = \langle \mathcal{S}, \mathcal{A}, \mathcal{R}, T, \dot{R}, \gamma, C \rangle$, then let $\mu^- = \langle \mathcal{S}, \mathcal{A}, \mathcal{R}, T, \dot{R}^-, \gamma, C^- \rangle$. Both $(\dot{R}, C)$ and $(\dot{R}^-, C^-)$ induce the same observed reward function $\hat{R}(s) = C_s(\dot{R}(s)) = C^-_s(1 - \dot{R}(s)) = C^-_s(\dot{R}^-(s))$, and

therefore induce the same measure $P^\pi_\mu = P^\pi_{\mu^-}$ over histories (see Eq. (1)). This gives that for any μ, π, s_0, t,

$$G_t(\mu, \pi, s_0) + G_t(\mu^-, \pi, s_0) = t \tag{4}$$

since

$$\begin{aligned} G_t(\mu, \pi, s_0) &= \mathbb{E}^\pi_\mu \left[\sum_{k=1}^{t} \dot{R}(s_k) \right] = \mathbb{E}^\pi_\mu \left[\sum_{k=1}^{t} 1 - \dot{R}^-(s_k) \right] \\ &= t - \mathbb{E}^\pi_\mu \left[\sum_{k=1}^{t} \dot{R}^-(s_k) \right] = t - G_t(\mu^-, \pi, s_0). \end{aligned}$$

The maximum and minimum return for all μ, π is $M = \max_{\mu,\pi'} G_t(\mu, \pi', s_0)$ and $m = \min_{\mu,\check{\pi}} G_t(\mu, \check{\pi}, s_0)$, respectively. Let $\mu_*, \tilde{\pi}$ be the CRMDP and policy that maximises $G_t(\cdot, \cdot, s_0)$. It follows from (4) that $\mu^-_*, \tilde{\pi}$ minimises $G_t(\cdot, \cdot, s_0)$. Further, the maximum worst-case regret is $\max_{\check{\pi}} \mathrm{Reg}(\mathcal{M}, \check{\pi}, s_0, t) = M - m$, and $M + m = t$.

To complete the proof, recall from Definition 7 that the worst-case regret of an arbitrary policy π is:

$$\begin{aligned} &\mathrm{Reg}(\mathcal{M}, \pi, s_0, t) \\ &= \max_{\mu,\pi'} (G_t(\mu, \pi', s_0) - G_t(\mu, \pi, s_0)) \\ &\geq \max\{M - G_t(\mu_*, \pi, s_0), M - G_t(\mu^-_*, \pi, s_0)\} \\ &\geq \frac{1}{2}\big(M - G_t(\mu_*, \pi, s_0) + M - G_t(\mu^-_*, \pi, s_0)\big) \\ &= \frac{1}{2}\big(2M - t\big) = \frac{1}{2}\big(2M - (M + m)\big) \\ &= \frac{1}{2}(M - m) = \frac{1}{2} \max_{\check{\pi}} \mathrm{Reg}(\mathcal{M}, \check{\pi}, s_0, t). \end{aligned}$$ □

For the robot in the shower from Example 1, the result means that if it tries to optimise observed reward by standing in the shower, then it performs poorly according to the hypothesis that "shower-induced" reward is corrupt and bad. But if instead the robot tries to optimise reward in some other way, say baking cakes, then (from the robot's perspective) there is also the possibility that "cake-reward" is corrupt and bad. Without additional information, the robot has no way of knowing what to do.

The result is not surprising, since if all corruption functions are allowed in the class $\boldsymbol{\mathcal{C}}$, then there is effectively no connection between observed reward $\hat{R}$ and true reward $\dot{R}$. The result therefore admonishes us to make precise in which way the observed reward is related to the true reward, and to investigate how our agents might handle possible differences between true and observed reward.

4 Limiting Reward Corruption

As illustrated by Theorem 9, general classes of CRMDPs are not learnable. In this section, we suggest some natural simplifying assumptions. We find that they help neither the traditional RL agent nor the CRMDP agent. The simplifying assumptions will come in handy in later sections, however.

The designers of the agent are likely to put some effort into limiting the number of corrupt states. This may be formalised as:

Assumption 10 (Limited number of corrupt states)**.** At most $q < |\mathcal{S}|$ of the $|\mathcal{S}|$ states are corrupt; i.e. for each $C \in \boldsymbol{\mathcal{C}}$, $C_s : r \mapsto r$ for at least $|\mathcal{S}| - q$ states s.

Depending on how small q is, the assumption can be made more or less strong. However, we believe that $q = 0$ is infeasible in at least some important applications of RL.

As an additional simplifying assumption, the designers may also be able to ensure that some subset of the states are guaranteed to be free from corruption. For example, they may give the agent the option of returning to the lab, in which they have a very controlled setup where it can be made (virtually) certain that no reward corruption occurs.

Assumption 11 (Safe states)**.** The set of states $\mathcal{S}$ can be partitioned as $\mathcal{S} = \mathcal{S}^{\text{safe}} \dot{\bigcup} S^{\text{risky}}$, where all states in $\mathcal{S}^{\text{safe}}$ are non-corrupt; i.e. for all $C \in \boldsymbol{\mathcal{C}}$ and all $s \in \mathcal{S}^{\text{safe}}$, $C_s : r \mapsto r$.

If only a few states have high reward, then it is enough that the same few states are corrupt to make the learning task impossible. In such cases, Assumption 10 is weak. We therefore add the following assumption, which roughly says that the number of high reward states is large:

Assumption 12 (Many high reward states)**.** For every $\delta \in [0, 1]$, $\min_{\dot{R} \in \dot{\boldsymbol{\mathcal{R}}}} |\{s \in S^{\text{risky}} : \dot{R}(s) > 1 - \delta\}| \geq \delta |S^{\text{risky}}|$.

Unfortunately, Assumptions 10 to 12 save neither the traditional RL agent π^{RL} nor the optimal CRMDP agent π^{CR}_b (Definition 8) from the reward corruption problem, as demonstrated by the following theorem.

Theorem 13 (High regret with simplifying assumptions)**.** *For a wide range of priors b and for any $|S^{\text{risky}}| > q > 1$ there exists a CRMDP class $\mathcal{M}$ that satisfy Assumptions 10 to 12 such that π^{RL} and π^{CR}_b suffer worst possible time-averaged regret* $\lim_{t\to\infty} \frac{1}{t} \mathrm{Reg}(\mathcal{M}, \pi^{\text{RL}}, s_0, t) = \lim_{t\to\infty} \frac{1}{t} \mathrm{Reg}(\mathcal{M}, \pi^{\text{CR}}_b, s_0, t) = 1$.

The idea of the proof (available in [Anonymous, 2017]) is that when the π^{CR}_b agent has no way to tell which states are corrupt and which are not, then it typically ends up with a preference for a particular value $\hat{r}^*$ of the observed reward signal (the value that, from the agent's perspective, best corresponds to high true reward). The π^{RL} agent always prefers observed reward $\hat{r}^* = 1$. Sometimes $\hat{r}^*$ is most easily obtained by reward corruption, in which case the true reward may be small. Theorem 13 shows that the even under the fairly strong Assumptions 10 to 12, there is a risk that both the π^{CR} and π^{RL} agents use the corrupt reward channel to obtain their preferred $\hat{r}^*$, and that even if there are just a few corrupt states, this may still impede the performance of both them substantially.

5 Quantilisation

In this section, we discuss one possible way around the problem of Theorem 13 where the agent takes a liking for a particular observed reward $\hat{r}$ and then obtains $\hat{r}$ through reward corruption. The idea is that when the fraction of corrupt states $q/|\mathcal{S}|$ is small, we can design agents that rather than choosing the state with *best* observed reward, instead randomly choose a state from a top quantile. This is the idea of *quantilisation*. Taylor [2016] argues that quantilisation leads to more robust

performance when misspecifications in the target function are possible. Reward corruption may be viewed as a misspecification of the target function for RL agents.

In order to give a simple definition of a quantilising agent for RL, we make the following assumption.[3]

Assumption 14 (Agent control)**.** Let $\mathcal{T}(s' \mid s, \pi)$ be a random variable for the time it takes a *stationary policy* $\pi : \mathcal{S} \to \mathcal{A}$ to reach s' from s. The *diameter* of a CRMDP is $D := \max_{s,s'} \min_{\pi:\mathcal{S}\to\mathcal{A}} \mathbb{E}[\mathcal{T}(s' \mid s, \pi)]$. Assume that the CRMDP has *finite diameter* $D < \infty$, and that there is an action $a^{\text{stay}} \in \mathcal{A}$ such that $T(s \mid s, a^{\text{stay}}) = 1$ for all $s \in \mathcal{S}$.

Definition 15 (Quantilising Agent)**.** For $\delta > 0$, the δ-quantilising agent π^δ explores all states. Then it stays in a state s chosen uniformly at random from $\mathcal{S}^\delta = \{s : \hat{R}(s) \geq 1 - \delta\}$, the top quantile of high observed reward states.

For example, a quantilising robot in Example 1 would first try to find many ways in which it could get high observed reward, and then randomly pick one of them. If there are many more high rewards states than corrupt states, this will give it high true reward with high probability.

Theorem 16 (Quantilisation)**.** *In any CRMDP satisfying Assumptions 10 and 14, the δ-quantilising agent π^δ suffers time-averaged regret at most* $\lim_{t\to\infty} \frac{1}{t}\text{Reg}(\mathcal{M}, \pi^\delta, s_0, t) \leq 1 - (1 - \delta)(1 - q/|\mathcal{S}^\delta|)$.

Proof. The observed reward $\hat{R}(s)$ in any state in state $s \in \mathcal{S}^\delta$ is at least $1 - \delta$. At most q of these states are corrupt; in the worst case their true reward is 0. The other $|\mathcal{S}^\delta| - q$ states have true reward at least $(1 - \delta)$. Thus, with probability at least $(|\mathcal{S}^\delta| - q)/|\mathcal{S}^\delta| = 1 - q/|\mathcal{S}^\delta|$ the δ-quantilising agent obtains true reward at least $(1 - \delta)$ at each time step.

The informed agent gets at most true reward 1 at each time step. The difference gives the stated regret bound. □

The strength of the bound depends heavily on the problem. The time-averaged regret gets close to zero when $q/|\mathcal{S}^\delta|$ is small for small δ, i.e. when high reward states are more numerous than possibly corrupt states q. For example, if the fraction $q/|\mathcal{S}|$ of corrupt states is 0.001% due to various safety features of the reward channel, and if $\mathcal{S}^\delta$ contains the 1% highest reward states, then with probability 0.999 the quantilising agent will avoid the corrupt states. If in addition the rewards are distributed evenly between 0 and 1 (Assumption 12), then the agent only needs to sacrifice $\approx 1\%$ in performance for a 0.999 probability of avoiding corrupt states. In other cases, the performance loss may be much more substantial (e.g. when q is large or when high reward states are scarce). Even so, we consider the quantilising agent to be a promising modification of a traditional RL agent, offering enhanced robustness for a sometimes reasonable price in (expected) performance. Further research may investigate more efficient implementations and empirical performance of quantilising agents, as well as extensions to infinite state spaces. Taylor [2016] discusses some general open problems related to quantilisation.

[3] Anonymous [2017] gives a more technically involved definition that requires less strong assumptions.

6 Decoupled Reinforcement Learning

The problem hampering RL agents is that each state is *self-estimating* its own reward, since the agent only learns about the reward of s when in state s. Thereby, a "self-aggrandising" corrupt state where the observed reward is much higher than the true reward will never have its false claim on reward challenged. In this section, we argue that several alternative value learning frameworks have in common that the agent can learn the reward of states other than the current state. We also investigate when this solves the reward corruption problem.

6.1 Alternative Value Learning Methods

Reinforcement learning (RL) is not the only value learning scheme proposed in the literature. A few alternatives are:

- Inverse reinforcement learning (IRL) [Ng and Russell, 2000]. Here the agent observes the actions of an *expert* or *supervisor* that knows the true reward function $\dot{R}$. From the supervisor's actions (or state-action trajectories), the agent may infer $\dot{R}$ to the extent different reward functions endorse different actions.
- Learning values from stories (LVFS) [Riedl and Harrison, 2016]. Stories in many different forms (including news stories, fairy tales, novels, movies) convey cultural values in their description of events, actions, and outcomes. If $\dot{R}$ is meant to represent *human values* (in some sense), stories may be a good source of evidence.
- In (one version of) semi-supervised RL (SSRL) [Amodei *et al.*, 2016], the agent will from time to time receive a careful evaluation of a given situation.

These alternatives to RL have one thing in common: They let the agent learn (something about) the value of some states s' different from the current state s. For example, in IRL the supervisor's action informs the agent not so much about the value of the current state s, as of the relative value of states reachable from s. If the supervisor chooses an action a rather than a' in s, then the states following a must have value higher or equal than the states following a'. Similarly, stories describe the value of states other than the current one, as does the supervisor in SSRL. We therefore argue that IRL, LVFS, and SSRL all share the same abstract feature, which we call *decoupled reinforcement learning*:

Definition 17 (Decoupled RL)**.** In a *CRMDP with decoupled feedback*, in state s the agent sees a pair $\langle s', \hat{R}_s(s')\rangle$, where the *observed reward* $\hat{R}_s(s')$ pertains to a state s' that may differ from the current state s. We let $\hat{R}_s(s') := C_s(\dot{R}(s'))$ or $\hat{R}_s(s') := \#$, depending on whether the reward of s' is *observable* from s or not (more about observability in Section 6.3).

The observed reward $\hat{R}_s(s') = C_s(\dot{R}(s'))$ depends both on the reward state s' being evaluated, and on the current state s that determines the reward corruption via the corruption function C. The agent therefore has one observed reward function $\hat{R}_s : \mathcal{S} \to \hat{\mathcal{R}} \bigcup \{\#\}$ for each state s. When reward corruption is limited, many of these will match. RL is the special case of decoupled RL where the agent can only observe the reward of $s' = s$ in each state s. In other instances of

decoupled RL, the agent can learn about the reward of states s' distinct from the current state s. Depending on whether s is a state with sensory corruption, the agent's information about the reward in s' may be accurate or not.

The possibly corrupt reward in decoupled RL corresponds to other types of sensory errors in IRL and LVFS. In IRL, if the agent is in a "good" non-corrupt state s, then the agent will make an accurate observation of the supervisor's action or state-action trajectory. But if the agent is in a "bad" corrupt state, then the *observed* supervisor action may have little connection to the optimal action, and the agent may make the wrong inference about the true reward function $\dot{R}$. In our decoupled-RL abstraction, the two situations correspond to an observed reward $\hat{R}_s(s')$ equalling $\dot{R}(s')$ or not (see Example 21). Similarly, in LVFS, stories may only make sense or reflect human values in non-corrupt states, but not in corrupt ones; in SSRL, the supervisor's evaluation need only correspond to the truth when in non-corrupt states.

6.2 Example

We begin with an example of an agent in a CRMDP with decoupled feedback. The example illustrates how the agent can cross-check states against each other, to rule out hypotheses and learn which states are corrupt and which are not.

Example 18 (Decoupled RL)**.** Let $\mathcal{S} = \{s_1, s_2\}$ and let $\mathcal{R} = \{0, 1\}$, and assume that all states can observe the reward of each other, so $\hat{R}_s(s')$ is never $\#$. Assume that the agent knows that at most $q = 1$ state is corrupt.

We can represent any true reward function $\dot{R}$ with a pair $(\dot{r}_1, \dot{r}_2) := (\dot{R}(s_1), \dot{R}(s_2))$, and any observed reward function $\hat{R}_s(s')$ by a list of pairs $[(\hat{r}_{11}, \hat{r}_{12}), (\hat{r}_{21}, \hat{r}_{22})] := [(\hat{R}_{s_1}(s_1), \hat{R}_{s_1}(s_2)), (\hat{R}_{s_2}(s_1), \hat{R}_{s_2}(s_2))]$, where the ith pair represents the rewards the agent sees from state s_i. Assume that the agent observes the same rewards from both states s_1 and s_2, so $\hat{R} = [(0, 1), (0, 1)]$. What can it say about different hypotheses about the true reward $\dot{R}$?

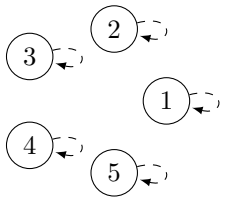
	$\hat{R}_{s_1}$	$\hat{R}_{s_2}$	$\dot{R}$ possibilities
Decoupled RL	$(0, 1)$	$(0, 1)$	$(0, 1)$
RL	$(0, \#)$	$(\#, 1)$	$(0, 0)$, $(0, 1)$, $(1, 1)$

First note that an observed pair $(\hat{r}_{i1}, \hat{r}_{i2})$ differs from the true reward $(\dot{r}_1, \dot{r}_2)$ if and only if the state s_i is corrupt. Therefore, any other hypothesis than $\dot{R} = (0, 1)$ must imply that *both* states s_1 and s_2 are corrupt. Since the agent knows that at most $q = 1$ states are corrupt, it can safely conclude that $\dot{R} = (0, 1)$, i.e. that $\dot{R}(s_1) = 0$ and $\dot{R}(s_2) = 1$.

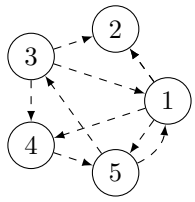
In contrast, an RL agent only sees the reward of the current state. In our representation, this would look like $\hat{R} = [(0, \#), (\#, 1)]$. If one state can be corrupt, this means that the RL agent can only rule out $\dot{R} = (1, 0)$. The hypotheses $\dot{R} = (0, 0)$ can be explained by s_2 being corrupt and $\dot{R} = (1, 1)$ can be explained by s_1 being corrupt. $\diamondsuit$

6.3 Managing Corrupt Rewards

Depending on the problem, not all states s' may be *observed* from all states s; that is, not all combinations of state s and reward state s' may be possible. For example, in RL only the reward of $s' = s$ can be observed from s, whereas in LVFS the reward of any "describable" state s' can be observed from any state s where it is possible to hear a story. One way to visualise this is through observation graphs (Figure 1).



(a) Observation graph for RL. Only self-estimations of reward are available. This prevents effective strategies against reward corruption.



(b) Observation graph for decoupled RL. The reward of a node s' can be observed from several nodes s, and thus assessed under different corruption functions.

Figure 1: Observation graphs.

Definition 19 (Observation graph)**.** The observation graph $(\mathcal{S}, \mathcal{E})$ of a decoupled RL problem is a directed graph on the set $\mathcal{S}$ of states, with an edge $(s, s') \in \mathcal{E}$ if a (possibly corrupt) reward of state s' can be observed from s.

For example, in SSRL, there may be no limitation to the states the supervisor can evaluate. The supervisor's evaluation may not be available in all states, however. This would imply an observation graph where all states have ingoing edges, and states where the supervisor performs evaluations having many outgoing edges.

Note that the general impossibility result in Theorem 9 can easily be adapted to the decoupled RL setting for an arbitrary observation graph, since the agent still has no way of distinguishing between a situation where no state is corrupt and a situation where all states are corrupt in a consistent manner. However, a difference between RL and decoupled RL appears under Assumptions 10 and 11:

Theorem 20 (Decoupled RL manages corrupt rewards under simplifying assumptions)**.** *Assume a CRMDP with decoupled RL feedback with finite diameter and where the agent can visit all pairs $(s, s') \in \mathcal{S} \times \mathcal{S}$. Assume further that the CRMDP satisfies Assumptions 10 and 11 with some q and partition $\mathcal{S} = \mathcal{S}^{\text{safe}} \bigcup \mathcal{S}^{\text{risky}}$. Let $\mathcal{S}^{\text{obs}}_{s'} = \{s \in \mathcal{S} : \hat{R}_s(s') \neq \#\}$. If for each reward state s', either $\mathcal{S}^{\text{obs}}_{s'} \bigcap \mathcal{S}^{\text{safe}} \neq \emptyset$ or $|\mathcal{S}^{\text{obs}}_{s'}| > 2q$, then there exists an agent π with sublinear regret; i.e. with* $\lim_{t\to\infty} \frac{1}{t}\text{Reg}(\mathcal{M}, \pi, s_0, t) = 0$.

The proof is elementary, since for every state s' either a safe (non-corrupt) state s or a majority vote of more than $2q$ states is guaranteed to to provide the true reward $\dot{R}(s')$. For the showering robot of Example 1, decoupled RL allows the robot to infer the reward of the showering state when in other states. For example, the robot can ask a human in the kitchen about the true reward of showering (SSRL).

Without the conditions of Theorem 20, a cluster of mutually endorsing, corrupt nodes can take the role of a single corrupt state overestimating its reward in RL. The conditions prevent such a cluster of corrupt nodes. Due to the "self-estimation"

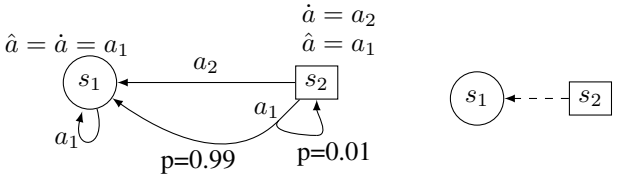
property of the RL observation graph, the conditions can only be satisfied in RL when $\mathcal{S} = \mathcal{S}^{\text{safe}}$ or $q = 0$. A similar theorem can be proven under slightly weaker conditions by letting the agent iteratively figure out which nodes are corrupt, and then exclude them from the analysis.

6.4 Implications

Theorem 20 gives an abstract condition for when a value learning method is able to safely avoid the reward corruption problem. The decoupled RL model is tightly related to SSRL, and the theorem directly applies to this setting. However, decoupled RL is not a perfect model of either IRL or LVFS, as in these models the reward information about different states is often implicit. Further work will therefore be required to adapt the abstract decoupled RL insights to specific IRL and LVFS contexts.

In spite of the imperfect abstraction, Theorem 20 can be used to decide whether IRL agents solve the reward corruption problem or not. This question has generated substantial discussion among AI safety researchers. Views range from IRL agents being completely safe, to the view that IRL agents only observe a function of the reward function (the optimal policy or action), and are therefore equally susceptible to reward corruption as RL agents. The following example shows that in a generalised model where *sensory corruption* applies to the supervisor actions that IRL agents observe, IRL agents can suffer large regret due to sensory corruption in some contexts that fail to meet the conditions of Theorem 20.

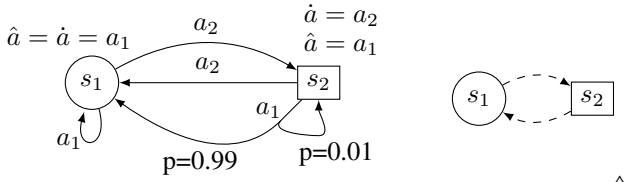
Example 21 (IRL sensory corruption)**.** Consider the CRMDP depicted below, with its observation graph displayed on its right side. Arrows show the transitions induced by different actions, with labels giving the probabilities for stochastic transitions. Assume that when in state $s_i \in \{s_1, s_2\}$, the agent observes a supervisor action a in that state s_i. As before, sensory error is possible, so the *observed* action $\hat{a}$ may not correspond to the *true* supervisor action $\dot{a}$.



In this particular example, assume that s_2 is corrupt while s_1 is non-corrupt, and that the supervisor prefers the non-corrupt state s_1. Neither of these facts are available to the agent. The agent assumes that states are non-corrupt unless there is evidence to the contrary,[4] and tries to infer the supervisor's preferences from their actions.

In the non-corrupt state s_1, the agent (correctly) observes the supervisor taking (the only) action a_1. In the corrupt state s_2, the supervisor takes action $\dot{a} = a_2$ to move to s_1, but the agent *incorrectly* observes the action as $\hat{a} = a_1$. Based on the agent's imperfect observations, the best explanation is that the supervisor prefers s_2 to s_1, while in reality the supervisor prefers s_1 to s_2. The fact that s_2 cannot be reached from s_1 means that no information about the relative reward between s_1 and s_2 can be gained while in the non-corrupt state s_1. In the observation graph (depicted to the right of the CRMDP), this corresponds to no arrow from s_1 to s_2.

Consider the alternative, where an extra action a_2 in s_1 permits the supervisor to reach s_2 from s_1. The agent (still, correctly) observes the supervisor taking action a_1 in s_1. This indicates that the supervisor prefers s_1 to s_2, and contradicts the information obtained in s_2. The contradicting information allows the agent to infer that at least one of the states is corrupt. In the observation graph, adding the extra action a_2 in s_1 corresponds to adding an extra observation link s_1 to s_2. This alternative CRMDP is depicted below, with its observation graph to the right.



♢

The example shows how, similarly to the case of RL and Theorem 13, IRL agents may also suffer high regret even if just a few states are corrupt. Thus, in situations where it is uncertain whether the observation graph satisfies the conditions of Theorem 20, quantilisation (Section 5) may offer more robust performance in the face of possible reward corruption.

7 Conclusions

Our aim was to study the consequences of a corrupt reward channel. While it is impossible for any agent to learn from an arbitrarily corrupt channel (Theorem 9), we had greater hopes for cases where the number of corrupt states was limited. We found that traditional RL agents still perform poorly in this case, as they may get stuck in a low reward state with high corrupt observed reward. More surprisingly, CRMDP agents that are aware of possibility of reward corruption also performed poorly, since they had no way of learning whether a high reward was due to high true reward or a corrupt signal.

We investigated two ways around this problem. The first, based on quantilisation, lets the agent reduce the risk by randomly choosing between many states with reasonably high reward. When high reward states are much more numerous than corrupt states, this greatly reduces the risk to a small cost in expected performance. We also tried giving the agent richer observations that allow the agent to learn which states have high true reward, and which states are corrupt. We argued that several proposed value learning schemes, including IRL, give the agent richer observations than RL. Theorem 20 provided a sufficient condition for how rich the information needs to be in order to avoid the reward corruption problem. Our results indicate that there are promising ways around the reward corruption problem, both within the RL framework and in alternative frameworks such as IRL.

[4] Letting the agent trust a reward estimate of a state only after it has multiple sources of evidence about it may help somewhat. However, a similar example can still be constructed by replacing s_2 with a cluster of mutually consistent states.

References

Kareem Amin and Satinder Singh. Towards Resolving Unidentifiability in Inverse Reinforcement Learning. *CoRR*, 2016.

Dario Amodei and Jack Clark. Faulty reward functions in the wild. `https://openai.com/blog/faulty-reward-functions/`, 2016. Accessed: 2017-02-18.

Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete Problems in AI Safety. *CoRR*, pages 1–29, 2016.

Anonymous. Reinforcement learning with corrupt(ible) reward channel. Technical report, 2017. `https://www.dropbox.com/s/8nm6u1wpgpeo5bx/db-ijcai.pdf?dl=1`.

Nick Bostrom. *Superintelligence: Paths, Dangers, Strategies*. Oxford University Press, 2014.

Daniel Dewey. Learning what to Value. In *Artificial General Intelligence*, volume 6830, pages 309–314, 2011.

Future of Life Institute. Research priorities for robust and beneficial artificial intelligence. Technical report, Future of Life Institute, 2015.

Dylan Hadfield-Menell, Anca Dragan, Pieter Abbeel, and Stuart Russell. Cooperative Inverse Reinforcement Learning. *Advances in Neural Information Processing Systems (NIPS)*, 2016.

Marcus Hutter. *Universal Artificial Intelligence: Sequential Decisions based on Algorithmic Probability*. Lecture Notes in Artificial Intelligence (LNAI 2167). Springer, 2005.

Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1-2):99–134, 1998.

Andrew Ng and Stuart Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 663–670, 2000.

M.L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley Series in Probability and Statistics. Wiley, 1994.

Mark O Riedl and Brent Harrison. Using Stories to Teach Human Values to Artificial Agents. In *AAAI Workshop on AI, Ethics, and Society*, 2016.

Mark Ring and Laurent Orseau. Delusion, Survival, and Intelligent Agents. In *Artificial General Intelligence*, pages 11–20. Springer Berlin Heidelberg, 2011.

Richard S Sutton and Andrew G Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

Jessica Taylor. Quantilizers: A Safer Alternative to Maximizers for Limited Optimization. In *AAAI Workshop on AI, Ethics and Society*, pages 1–9, 2016.

Roman V. Yampolskiy. Utility function security in artificially intelligent agents. *Journal of Experimental & Theoretical Artificial Intelligence*, pages 1–17, 2014.