CONSEQUENTIALIST CONDITIONAL COOPERATION IN SOCIAL DILEMMAS WITH IMPERFECT INFORMATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Social dilemmas, where mutual cooperation can lead to high payoffs but participants face incentives to cheat, are ubiquitous in multi-agent interaction. We wish to construct agents that cooperate with pure cooperators, avoid exploitation by pure defectors, and incentivize cooperation from the rest. However, often the actions taken by a partner are (partially) unobserved or the consequences of individual actions are hard to predict. We show that in a large class of games good strategies can be constructed by conditioning one's behavior solely on outcomes (ie. one's past rewards). We call this consequentialist conditional cooperation. We show how to construct such strategies using deep reinforcement learning techniques and demonstrate, both analytically and experimentally, that they are effective in social dilemmas beyond simple matrix games. We also show the limitations of relying purely on consequences and discuss the need for understanding both the consequences of and the intentions behind an action.

1 INTRODUCTION

Deep reinforcement learning (RL) is concerned with constructing agents that start as blank slates and can learn to behave in optimal ways in complex environments.[1] A recent stream of research has taken a particular interest in social dilemmas, situations where individuals have incentives to act in ways that undermine socially optimal outcomes (Leibo et al., 2017; Perolat et al., 2017; Lerer & Peysakhovich, 2017; Kleiman-Weiner et al., 2016). In this paper we consider RL-based strategies for social dilemmas in which information about a partner's actions or the underlying environment is only partially observed.[2]

The simplest social dilemma is the Prisoner's Dilemma (PD) in which two players choose between one of two actions: cooperate or defect. Mutual cooperation yields the highest payoffs, but no matter what one's partner is doing, one can get a higher reward by defecting. A well studied strategy for maintaining cooperation when the PD is repeated is tit-for-tat (TFT, Axelrod (2006)). TFT behaves by copying the prior behavior of their partner, rewarding cooperation today with cooperation tomorrow. Thus, if an agent commits to TFT it makes cooperation the best strategy for the agent's partner. TFT has proven to be a heavily studied strategy because it has intuitive appeal: 1) it is easily explainable, 2) it begins cooperating, 3) it rewards a cooperative partner, 4) it avoids being exploited, 5) it is forgiving.

[1]This approach has been applied to domains including: single agent decision problems (Mnih et al., 2015), board and card-based zero-sum games (Tesauro, 1995; Silver et al., 2016; Heinrich & Silver, 2016), video games (Kempka et al., 2016; Wu & Tian, 2016; Ontanón et al., 2013; Usunier et al., 2016; Foerster et al., 2017a), multi-agent coordination problems (Lowe et al., 2017; Foerster et al., 2017b; Riedmiller et al., 2009; Tampuu et al., 2017; Peysakhovich & Lerer, 2017), and the emergence of language (Lazaridou et al., 2017; Das et al., 2017; Evtimova et al., 2017; Havrylov & Titov, 2017; Jorge et al., 2016).

[2]Game theorists have studied the emergence of cooperation in bilateral relationships under both perfect and imperfect observation (Green & Porter, 1984; Fudenberg & Maskin, 1986; Fudenberg et al., 1994; Axelrod, 2006; Kamada & Kominers, 2010; Abreu et al., 1990). However, this research program almost exclusively studies repeated matrix games such as the two action, stochastically repeated Prisoner's Dilemma (PD) and focuses mostly on proving the existence of equilibria which maintain cooperation rather than on constructing simple strategies that do well across many complex situations. We take some of the insights developed here and use them in the service of agent design.

In Markov games cooperation and defection are not single actions, but rather temporally extended policies. Recent work has considered expanding TFT to more complex Markov games either as a heuristic, by learning cooperative and selfish policies and switching between them as needed (Lerer & Peysakhovich, 2017), or as an outcome of an end-to-end procedure (Foerster et al., 2017c). TFT is an example of a *conditionally cooperative* strategy - that is, it cooperates when a certain condition is fulfilled (ie. the partner's last period action was cooperative). Standard TFT requires perfect observability of a partner's behavior and perfect understanding of each action's future consequences.

Our main contribution is to use RL methods to construct conditionally cooperative strategies for games with imperfect information. When information is imperfect, the agent must use what they can observe to try to estimate whether a partner is acting cooperatively (or not) and determine how to respond. We show that when the game is ergodic, observed rewards can be used as a summary statistic - if the current total (or time averaged) reward is above a time-dependent threshold (where the threshold values are computed using RL and a form of self play) the agent cooperates, otherwise the agent does not[3]. We call this consequentialist conditional cooperation (CCC). We show analytically that this strategy cooperates with cooperators, avoids exploitation, and guarantees a good payoff to the CCC agent in the long run.

We study CCC agents in a partially observed Markov game which we call Fishery. In Fishery two agents live on different sides of a lake in which fish appear. The game has partial information because agents cannot observe what happens across the lake. Fish spawn randomly, starting young and swim to the other side and become mature. Agents can catch fish on their side of the lake. Catching any fish yields payoff but mature fish are worth more. Therefore, cooperative strategies are those which leave young fish for one's partner. However, there is always a temptation to defect and catch both young and mature fish. We show that CCC agents cooperate with cooperators, avoid exploitation, and get high payoffs when matched with themselves.

Second, we show that CCC is an efficient strategy for more complex games where implementing conditional cooperation by fully modeling the effect of an action on future rewards (eg. amTFT (Lerer & Peysakhovich, 2017)) is computationally demanding. To demonstrate this we follow the method of Tampuu et al. (2017) to construct a version of Atari Pong which makes the game into a social dilemma. In what we call the Pong Player's Dilemma (PPD) when an agent scores they gain a reward of 1 but the partner receives a reward of -2. Thus, in the PPD the only (jointly) winning move is not to play, but selfish agents are again tempted to defect and try to score points even though this decreases total social reward. We see that CCC is a successful, robust, and simple strategy in this game.

Conditioning on only outcomes has limitations. We consider a version of the Pong Players' Dilemma where when a player scores, instead of their partner losing 2 points deterministically they lose $2/p$ points with probability p. Here the *expected* rewards of non-cooperation are the same as in the PPD and so expected-future-reward based methods (eg. amTFT) will act act identically. However, when p is low it may take a long time for consequentialist agents to detect a defector. Empirically we see that in short risky PPD games CCC agents can be exploited by defectors but that amTFT agents cannot. We close by discussing limitations and progress towards agents that can effectively use both intention and outcome information effectively in navigating the world.

2 Consequentialist Conditionally Cooperative Strategies

We work with partially observed Markov games (POMG), which are multi-agent generalizations of partially observed Markov decision problems:

Definition 1 *A (two-player, finite)* ***partially observed Markov game*** *(POMG) consists of: a finite set of states S; a set of actions for each player $\mathcal{A}_1, \mathcal{A}_2$; a transition function $\tau : S \times \mathcal{A}_1 \times \mathcal{A}_2 \to \Delta(S)$; an observation function that tells us what each player observes $_i : S \times \mathcal{A}_1 \times \mathcal{A}_2 \to \Delta(\Omega_i)$ where Ω_i is a set of possible observations; and a reward function that maps states and actions to each player's reward $R_i : S \times \mathcal{A}_1 \times \mathcal{A}_2 \to \Delta(\mathbb{R})$.*

[3]In an ideal world we may want to construct a full posterior using Bayesian methods. However, this is often computationally difficult in practice.

We assume that per turn rewards are bounded above and below. Agents choose a policy

$$\pi_i : \Omega_i \to \Delta(\mathcal{A}_i)$$

which takes as input the observation and outputs a probability distribution on actions - this is similar to the notion of 'belief free strategies' in the study of repeated games (Ely et al., 2005). Given a pair of policies, one for each agent, and a starting state, we define each player's value function as the average (undiscounted) reward if players start in state s and follow policies π_1, π_2, i.e.

$$V_i(s, \pi_1, \pi_2) = \lim_{t\to\infty} E\left[\frac{1}{t}\sum_{k=0}^{t} r_k^i\right]$$

Note that while we will prove results in the undiscounted setting, for δ sufficiently close to 1, the optimal policy is the same in the undiscounted and discounted setting, so standard discounted policy gradient techniques can still be used (Schwartz, 1993).

We restrict ourselves to reward-ergodic POMGs:

Assumption 1 (Reward Ergodicity in POMG) *Say a POMG is reward-ergodic if for any pair of policies π_1, π_2 the long-run reward has a well defined rate independent of the starting state almost surely. Formally, this means for any starting state s, and either player i, there exists a rate $\rho^i_{\pi_1\pi_2}$ such that*

$$\lim_{t\to\infty} V_i(s, \pi_1, \pi_2) \xrightarrow{a.s.} \rho^i_{\pi_1\pi_2}$$

Any pair of policies applied to a POMG creates a Markov chain of underlying states. If for any pair of policies that Markov chain is 'unichain', that it, it has a single positive recurrent chain (Puterman, 2014) then the POMG will be reward ergodic(Meyn & Tweedie, 2012)[Thm. 17.0.1]. The unichain assumption is often used in applications of RL methods in the undiscounted RL problem (Schwartz, 1993).

Definition 2 ***Cooperative Policies*** *are those that maximize the joint rate of reward:*

$$(\pi_1^C, \pi_2^C) \in \textit{argmax}_{\Pi_1,\Pi_2}(V_1(\pi_1, \pi_2) + V_2(\pi_1, \pi_2)).$$

Let Π^C be the set of such tuples.

We look at the class of POMGs which have two restrictions:

Assumption 2 (Social Dilemma) *For any player i and any $(\pi_1^C, \pi_2^C) \in \Pi^C$ we have that*

$$\pi_i^C \notin \textit{argmax}_{\Pi_i}(V_i(s, \pi_i, \pi_j^C)).$$

Assumption 3 (Exchangeability of Cooperative Strategies) *For any $(\pi_1, \pi_2) \in \Pi^C$ and $(\pi_1', \pi_2') \in \Pi^C$ we have that $(\pi_i', \pi_j) \in \Pi^C$.*

Note that if this exchangeability assumption is not satisfied we have both a cooperation problem (should agents defect?) and also a coordination problem (if we choose to cooperate, how do we choose among the multiple potential ways to cooperate?) We point the reader to Kleiman-Weiner et al. (2016) for further discussion of this issue. Solving the coordination problem (eg. by introducing communication) is beyond the scope of this paper though it is an important avenue for future work.

To construct a CCC agent we need access to a policy pair (π_1^D, π_2^D) that forms a Nash equilibrium of the game. We assume that these strategies generalize two properties of defection in the Prisoner's Dilemma: 1) the have lower rates of payoff than the socially optimal strategies in the long run for both players and 2) if we take a mixed policy π^{C+D} which behaves according to π^C at some periods and π^D at others then $V_i(\pi_i^{C+D}, \pi_j^C) \geq V_i(\pi_i^C, \pi_j^C)$. This last condition is essentially saying that π^D is a selfish policy that, even if used some of the time, still increases the payoffs of the player choosing it (while decreasing total social efficiency). We explain a weaker condition in the appendix.

To behave according to CCC our agent maintains a persistent state at each time period C_t^i which is the current time-averaged reward it has received.

The agent plays according to π^C if $C_t^i > T_t$ and π^D otherwise. Let ρ_{CC} be the rate associated with both players behaving according to π^C and let ρ_{CD} be the rate associated with our agent playing according to π^C and the other agent behaving according to π^D. Let $T = (1-\alpha)\rho_{CC} + \alpha\rho_{CD}$, where $0 < \alpha < 1$ is a slack parameter that specifies the agent's leniency. We present the following result:

Theorem 1 *Consider a strategy where agent* 1 *acts according to* π_1^C *if* $C_i^T > T$ *and* π_1^D *otherwise. This gives two guarantees in the long run:*

1. ***Cooperation Wins:*** *If agent* 2 *acts according to* π_2^C *then for both agents* $\lim_{t\to\infty} \frac{1}{t}\sum r_t^i = \rho_{CC}^i$.
2. ***Defecting Doesn't Pay:*** *If agent* 2 *acts according to a policy that gives agent* 1 *a payoff of less than* T *in the long run then* $\lim_{t\to\infty} \frac{1}{t}\sum r_t^2 \leq \rho_{DD}^2 < \rho_{CC}^2$.

Thus, a simple threshold based strategy for player 1 makes cooperation a better strategy than defection in the long-run for player 2 in any ergodic game. CCC satisfies the desiderata we set out at the beginning: it is simple to understand, cooperates with cooperators, does not get exploited by pure defectors, incentivizes rational partners to cooperate, and, importantly, gives a way to return to cooperation if it has failed in the past.[4]

The CCC strategy also provides a payoff guarantee against rational learning agents: if one's partner is a learning agent who best responds in the long-run then, since π^D forms an equilibrium, a CCC agent can always guarantee themselves a payoff of at least ρ_{DD} in the long-run. Unfortunately, CCC does not give any adversarial guarantees in general. Extending conditionally cooperative strategies to be robust to not only selfish agents trying to cheat but adversarial agents trying to actively destroy value is an interesting direction for future work.

However, we note that the simple construction above only gives long-run guarantees. We now focus on generalizing this strategy to work well in finite time as well as how to use RL methods to compute its components. Finally, we note that this strategy can be extended to some non-payoff ergodic games. For example, if there is a persistent but unknown state which affects both rewards equally (say, multiplies them by some factor) then the amount of inequality (eg. the ratio or difference of rewards or more complex function such as those used by Fehr & Schmidt (1999)) can be used as a summary statistic.

3 RL Implementation of CCC

To construct the $\hat{\pi}^C$ and $\hat{\pi}^D$ policies used by CCC, we follow the training procedure of Lerer & Peysakhovich (2017). We perform self-play with modified reward schedules to compute the two policies:

1. Selfish - here both players get rewards equal to their own reward. This is the standard self-play paradigm used in multi-agent zero-sum settings. We refer to the learned policies here as $\hat{\pi}^D$
2. Prosocial - here both players get rewards at each time step not only for their own reward, but also for the reward their partner receives. We refer to the learned polices as $\hat{\pi}^C$

Our agents are set up as standard deep RL agents which take game state (e.g. pixels) as input and pass them through a convolutional neural network to compute a distribution over actions. The architectures of the agents as well as the training procedures are standard and we put them in the appendix.

We note that learning policies via RL in POMDPs has unique challenges (Jaakkola et al., 1995). The correct choice of learning algorithm will depend on the situation; policy gradient is preferred for POMDPs because optimal policies may be nondeterministic, and a common approach is to perform a variant of policy gradient with a function approximator augmented with an RNN 'memory' that keeps track of past states (Heess et al., 2015). In our Fishery game, the policy (but not the value) is

[4]Cooperation returns when CCC returns to a time averaged payoff above the threshold, this means a partner can accelerate this process by taking actions to give the CCC agent extra payoff. We refer to this process as 'giving flowers to apologize.'

independent of the unobserved state, so RNN augmentation was unnecessary; however, since our policy was stateless we had to avoid value-based methods (e.g. actor-critic) because the aliasing of *values* prevents these methods from finding good policies.

Having computed the policies, we need to compute thresholds for conditional cooperation. There are 3 important sources of finite time variance that a threshold needs to account for: first, a partner's π^C may not be the same as our agent's due to function approximation. Second, initial states of the game may matter in finite time. Third, there may be inherent stochasticity in the rewards.

We compute the per-turn threshold $\hat{T}_t^i$ as follows: first we take our learned $\hat{\pi}^C$ and $\hat{\pi}^D$ and perform k rollouts of the full game assuming cooperate (we call the resulting per period cumulative payoffs to our agent $\hat{R}_{CC}^{tk}$ where k corresponds to the iteration and t to the time). We also compute batches of rollouts of a paired cooperator and defector $\hat{R}_{CD}^{tk}$. We let $\bar{R}_{CC}^t$ be the bottom q^{th} percentile of these sample paths and we define our time dependent threshold in terms of cumulative reward as

$$\hat{T}^t = (1-\alpha)\bar{R}_{CC}^t + \alpha \frac{1}{k} \sum_k R_{CD}^{tk}.$$

If the CCC agent's current cumulative reward is above $\hat{T}^t$ they behave according to $\hat{\pi}^C$, otherwise they use $\hat{\pi}^D$.

This process gives us slack to account for the three sources of error described above. Tuning the parameters q, α allows us to trade off between the importance of false positives (detecting defection when one hasn't occurred) and false negatives (missing a defecting opponent). The algorithm is formalized into pseudocode below. We also show an example of threshold computation as well as associated precision/recall with actual opponents in the example below.

Algorithm 1 CCC as Agent 1

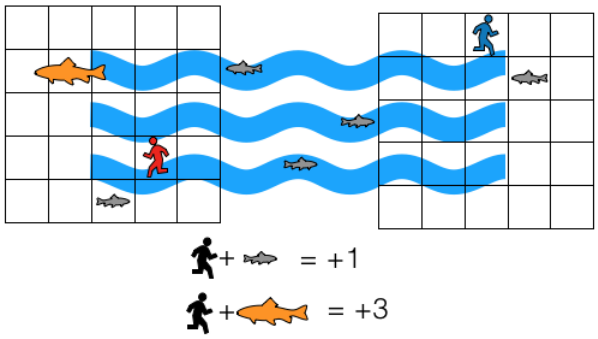
```
Input: π̂^C, π̂^D, α, q, k
for b in range(0, k) do
    s_CC[b] ← NewGame()
    s_CD[b] ← NewGame()
while Game do
    for b in range(0, k) do
        s_CC[b], R_CC[b] ← Step(s_CC[b], π̂_1^C, π̂_2^C)      ▷ Step returns next state and total reward
        s_CD[b], R_CD[b] ← Step(s_CD[b], π̂_1^C, π̂_2^D)
    R̄_CC ← quantile(R_CC, q)
    R̄_CD ← mean(R_CD)
    T ← (1 − α)R̄_CC + αR̄_CD
    if CurrentTotalReward < T then
        Choose a = π̂_1^D(o)
    else
        Choose a = π̂_1^C(o)
```

4 EXPERIMENTS

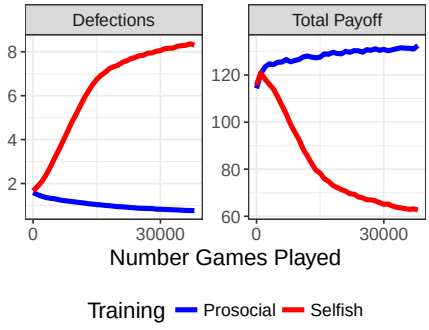
4.1 EXPERIMENT: FISHERY

Our first example is a common pool resource game which we call Fishery. In Fishery two agents live on different sides of a lake in which fish appear. Each side of the lake is instantiated as a 5×5 grid and agents can walk in all cardinal directions. Fish spawn randomly, starting young and swim to the other side and become mature. Agents can catch fish on their side of the lake by walking over them. Catching young fish yields 1 reward while mature fish yield a reward of 3.

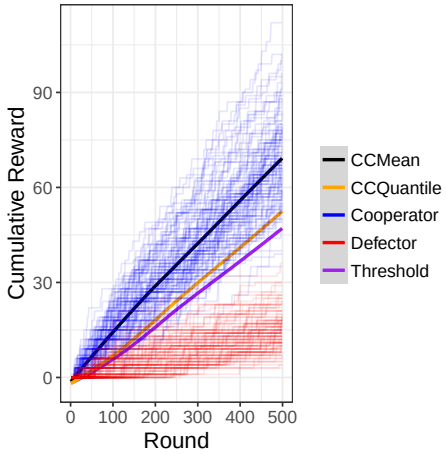
In Fishery cooperative strategies are those which leave young fish for one's partner. However, there is always a temptation to defect and catch both young and mature fish. Fishery is an imperfect observation game because agents cannot see the behavior of their partners across the lake. Figure 1 shows an example of a threshold in our Fishery game with $q = .1$ and $\alpha = .05$, we see that CC



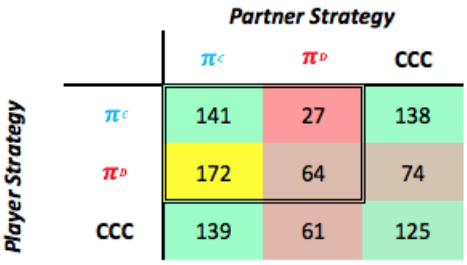
(a) Fishery Game



(b) Training Curves



(c) CCC Threshold



(d) Tournament Results

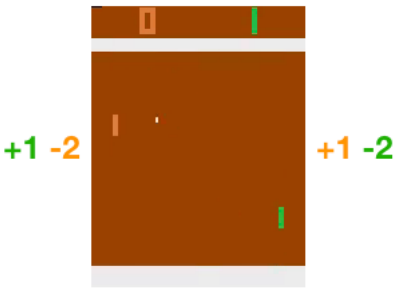
Figure 1: In Fishery two agents live on opposite sides of a lake and cannot observe each other's actions directly. Each time step fish can spawn on their side of the lake and begin to swim to the other side. Fish start young and become mature if they are allowed to enter the middle of the lake. Training using selfish self-play leads to agents that try to eat all the fish and thus cannot reach optimal payoffs, while social training finds cooperative strategies. CCC agents are able to cooperate with cooperators, avoid exploitation by defectors, incentivize cooperation, and get good payoffs when matched with other CCC agents. Panel C shows example trajectories of payoffs as well as the CCC per-round threshold. Panel D shows the numerical payoffs for an agent choosing the row strategy with a partner who chooses the column strategy.

trajectories remain mostly above the threshold (meaning low false positives) and CD trajectories mostly lie below even after a short time period (meaning low false negatives). The game as well as the experimental results are shown in Figure 1.

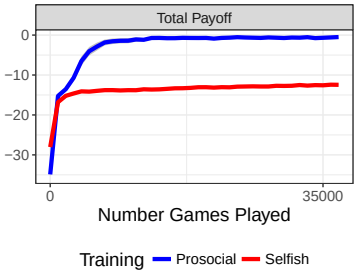
We train 50 pairs of agents under the selfish and prosocial reward schemes using a policy gradient method with simple CNN policy approximators (see Appendix for details). We see that selfish training leads to agents that defect and choose greedy, suboptimal strategies whereas prosocial training finds good policies. We then compute thresholds as described above and implement CCC agents. To see how CCC would perform in a mixed population we consider a tournament where we draw random policies from our trained pool of 50 and have them play a 1000 time step Fishery game (we use fixed lengths to normalize the payoffs across games). We find that CCC agents cooperate with cooperators, are not exploited by defectors, and do well when matched with other CCC agents.

4.2 EXPERIMENT: PONG PLAYERS' DILEMMA

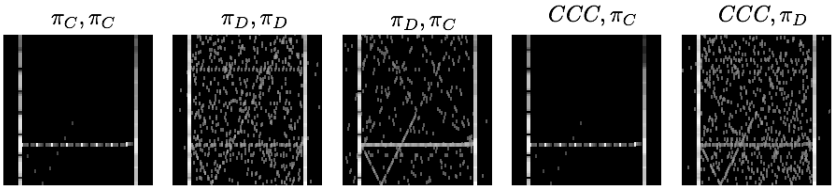
CCC can also be used in games of perfect information such as the Pong Players' Dilemma (PPD).[5] In the PPD we alter the reward structure of Atari Pong so that whenever an agent scores a point they receive a reward of 1 and the other player receives -2 (Tampuu et al., 2017). In the PPD the only (jointly) winning move is not to play. However, selfish agents are again tempted to defect and try to score points even though this decreases total social reward.



(a) Pong Player's Dilemma



(b) Training Curves



(c) Examples of Play

Player Strategy \ Partner Strategy: PPD	π^C	π^D	CCC	amTFT
π^C	0	-24.3	-0.26	-0.9
π^D	12.2	-5.9	-3.6	-3.5
CCC	0.2	-10.5	0	-1.56
amTFT	1.3	-11.1	-0.4	-1.6

(d) Results (PPD)

Player Strategy \ Partner Strategy: RPPD	π^C	π^D	CCC	amTFT
π^C	-0.74	-28.4	-0.8	-0.34
π^D	12.1	-5.8	4.9	-2.9
CCC	0.3	-18	-0.2	-1.3
amTFT	1.2	-8.9	-0.7	-3.6

(e) Results (Risky PPD)

Figure 2: In the Pong Player's Dilemma we alter the reward structure of Atari Pong so that whenever an agent scores a point they receive a reward of 1 and the other player receives -2. Agents are trained directly from the pixels with no pre-existing conceptions of how to play. Selfish training leads to agents that try hard to score and thus end up with bad payoffs. Cooperators learn to gently hit the ball back and forth. CCC agents behave like cooperators when faced with cooperators and prevent themselves from being exploited by defectors and match the performance of the more complex amTFT in the PPD but not the risky PPD. Panel C shows example PPD games between different strategies with brightness of a pixel indicating proportion of time ball spends in that location. Panel D and E show the numerical payoffs for an agent choosing the row strategy against a partner playing the column strategy for both versions of the PPD.

[5]We also study CCC in a perfectly observed grid based social dilemma used in other work (Lerer & Peysakhovich, 2017; Foerster et al., 2017c). The results mirror those of the PPD so we relegate them to the Appendix.

We train 18 pairs of agents under the selfish and prosocial reward schemes using randomly determined game lengths using a standard A3C implementation (Mnih et al. (2016); see Appendix for complete details). As with the games above we see that selfish training leads to selfish agents that try hard to score every point while prosocial training leads to cooperative agents that hit the ball directly back and forth. We construct CCC agents as in the experiments above and see how they perform against other strategies in fixed length PPD games. Again we find that CCC cooperates with cooperators (and itself) but does not get exploited by defectors.

An alternative way to construct a conditionally cooperative agent is to be forward looking rather than outcome based. This is the construction employed by approximate Markov Tit-for-Tat (amTFT Lerer & Peysakhovich (2017)). amTFT conditions its cooperation on a counterfactual future reward - the amTFT agent sees the actions taken by their partner and, if the action is not the one suggested by π^C uses the game's Q function (which is learned or simulated via rollouts) to estimate the one shot-gain to the partner from taking this action. The agent keeps track of the total 'debit' a partner has accrued over time and if that crosses a threshold the amTFT agent then behaves according to π^D for enough periods such that the partner's debit is wiped out. We call this type of strategy intention-based because it computes, at the time of the action, the expected future consequences rather than waiting for those consequences to occur as CCC agents do.

Though amTFT is more computationally expensive because it requires the use of a Q function (which can be hard to train) or rollouts (which are expensive to compute) we follow the procedure in Lerer & Peysakhovich (2017) to construct amTFT agents for the PPD. We see that CCC agents, which are much simpler, perform just as well.

However, CCC is not always as effective as forward looking strategies. In particular, intention-based strategies can be effective on much shorter timescales than CCC. To demonstrate this we modify the reward structure of the PPD so that when a player scores, instead of their partner losing 2 points deterministically they lose $2/p$ points with probability p. Here the *expected* rewards of non-cooperation are the same as in the PPD and so amTFT acts identically (though now we require large batches if using rollouts). However, we see that in intermediate length games (1000 time steps) and $p = .1$ CCC agents can be exploited by defectors.[6] A screenshot of the game as well as experimental results are reported in Figure 2.

5 Conclusion, Limitations and Future Work

In this work we have introduced consequentialist conditionally cooperative strategies and shown that they are useful heuristics in social dilemmas, even in those where information is imperfect either due to the structure of the game or due to the fact that we cannot perfectly forecast a partner's future actions. We have shown that using one's own reward stream as a summary statistic for whether to cooperate (or not) in a given period is guaranteed to work in the limit as long as the underlying game is ergodic. Note that this sometimes (but not always) gives good finite time guarantees. In particular, the time scale for a CCC agent to detect exploitation is related to the mixing time of the POMG and the stochasticity of rewards; if these are large, then correspondingly long games are required for CCC to perform well.

We have also compared consequentialist and forward-looking models. As another simple example of the difference between the two we can consider the random Dictator Game (rDG) introduced by Cushman et al. (2009). In the rDG, individuals are paired, one (the Dictator) is given an amount of money to split with a Partner, and chooses between one of two dice, a 'fair' die which yields a $50-50$ split with a high probability and an unfair split with a low probability and an 'unfair' die which yields a $50-50$ split with low probability. Consequentialist conditional cooperators would label a partner a defector if an unfair outcome came up (regardless of die choice) whereas intention-based cooperators would look at the choice of die, not the actual outcome.

For RL trained agents, conditioning purely on intentions (eg. amTFT) has advantages in that it is forward looking and doesn't require ergodicity assumptions but it is an expensive strategy that is complex (or impossible) to implement for POMDPs and requires very precise estimates of potential outcomes. CCC is simple, works in POMDPs and requires only information about payoff rates (rather

[6]For simplicity for this experiment we use the $\hat{\pi}^C$ and $\hat{\pi}^D$ strategies trained in the standard PPD for the risky PPD, this is because the risky PPD is the same (in expectation) as the PPD.

than actual policies), however it may take a long time to converge. Each has unique advantages and disadvantages. Therefore constructing agents that can solve social dilemmas will require combining consequentialist and intention-based signals.

Interestingly, experimental evidence shows that while humans combine both intentions and outcomes, we often rely much more heavily on consequences than 'optimal' behavior would demand. For example, experimental subjects rely heavily on the outcome of the die throw rather than die choice in the rDG (Cushman et al., 2009). This is evidence for the notion that rather than acting optimally in each situation, humans have social heuristics which are tuned to work across many environments (Rand et al., 2014; Hauser et al., 2014; Ouss & Peysakhovich, 2015; Arechar et al., 2016; Mao et al., 2017; Niella et al., 2016). There is much discussion of hybrid environments that include both artificial agents and humans (eg. Shirado & Christakis (2017); Crandall et al. (2017)). Constructing artificial agents that can do well in such environments will require going beyond the kinds of optimality theorems and experiments highlighted in this and related work. To be successful, AI researchers will have to understand human social heuristics and construct agents that are in tune with human moral and social intuitions (Bonnefon et al., 2016; Greene, 2014).

REFERENCES

Dilip Abreu, David Pearce, and Ennio Stacchetti. Toward a theory of discounted repeated games with imperfect monitoring. *Econometrica: Journal of the Econometric Society*, pp. 1041–1063, 1990.

Antonio A Arechar, Anna Dreber, Drew Fudenberg, and David G Rand. I'm just a soul whose intentions are good: The role of communication in noisy repeated games. *SSRN*, 2016.

Robert M Axelrod. *The evolution of cooperation: revised edition*. Basic books, 2006.

Jean-François Bonnefon, Azim Shariff, and Iyad Rahwan. The social dilemma of autonomous vehicles. *Science*, 352(6293):1573–1576, 2016.

Jacob W Crandall, Mayada Oudah, Fatimah Ishowo-Oloko, Sherief Abdallah, Jean-François Bonnefon, Manuel Cebrian, Azim Shariff, Michael A Goodrich, Iyad Rahwan, et al. Cooperating with machines. *arXiv preprint arXiv:1703.06207*, 2017.

Fiery Cushman, Anna Dreber, Ying Wang, and Jay Costa. Accidental outcomes guide punishment in a 'trembling hand' game. *PloS one*, 4(8):e6699, 2009.

Abhishek Das, Satwik Kottur, José MF Moura, Stefan Lee, and Dhruv Batra. Learning cooperative visual dialog agents with deep reinforcement learning. *arXiv preprint arXiv:1703.06585*, 2017.

Jeffrey C Ely, Johannes Hörner, and Wojciech Olszewski. Belief-free equilibria in repeated games. *Econometrica*, 73(2):377–415, 2005.

Katrina Evtimova, Andrew Drozdov, Douwe Kiela, and Kyunghyun Cho. Emergent language in a multi-modal, multi-step referential game. *arXiv preprint arXiv:1705.10369*, 2017.

Ernst Fehr and Klaus M Schmidt. A theory of fairness, competition, and cooperation. *The quarterly journal of economics*, 114(3):817–868, 1999.

Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. *arXiv preprint arXiv:1705.08926*, 2017a.

Jakob Foerster, Nantas Nardelli, Gregory Farquhar, Philip Torr, Pushmeet Kohli, Shimon Whiteson, et al. Stabilising experience replay for deep multi-agent reinforcement learning. *arXiv preprint arXiv:1702.08887*, 2017b.

Jakob N Foerster, Richard Y Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with opponent-learning awareness. *arXiv preprint arXiv:1709.04326*, 2017c.

Drew Fudenberg and Eric Maskin. The folk theorem in repeated games with discounting or with incomplete information. *Econometrica: Journal of the Econometric Society*, pp. 533–554, 1986.

Drew Fudenberg, David Levine, and Eric Maskin. The folk theorem with imperfect public information. *Econometrica: Journal of the Econometric Society*, pp. 997–1039, 1994.

Edward J Green and Robert H Porter. Noncooperative collusion under imperfect price information. *Econometrica: Journal of the Econometric Society*, pp. 87–100, 1984.

Joshua Greene. *Moral tribes: Emotion, reason, and the gap between us and them*. Penguin, 2014.

Oliver P Hauser, David G Rand, Alexander Peysakhovich, and Martin A Nowak. Cooperating with the future. *Nature*, 511(7508):220–223, 2014.

Serhii Havrylov and Ivan Titov. Emergence of language with multi-agent games: Learning to communicate with sequences of symbols. *arXiv preprint arXiv:1705.11192*, 2017.

Nicolas Heess, Jonathan J. Hunt, Timothy P. Lillicrap, and David Silver. Memory-based control with recurrent neural networks. *CoRR*, abs/1512.04455, 2015. URL `http://arxiv.org/abs/1512.04455`.

Johannes Heinrich and David Silver. Deep reinforcement learning from self-play in imperfect-information games. *arXiv preprint arXiv:1603.01121*, 2016.

Tommi Jaakkola, Satinder P Singh, and Michael I Jordan. Reinforcement learning algorithm for partially observable markov decision problems. In *Advances in neural information processing systems*, pp. 345–352, 1995.

Emilio Jorge, Mikael Kågebäck, and Emil Gustavsson. Learning to play guess who? and inventing a grounded language as a consequence. *arXiv preprint arXiv:1611.03218*, 2016.

Yuichiro Kamada and Scott Duke Kominers. Information can wreck cooperation: A counterpoint to kandori (1992). *Economics Letters*, 107(2):112–114, 2010.

Michał Kempka, Marek Wydmuch, Grzegorz Runc, Jakub Toczek, and Wojciech Jaśkowski. Vizdoom: A doom-based ai research platform for visual reinforcement learning. *arXiv preprint arXiv:1605.02097*, 2016.

Max Kleiman-Weiner, MK Ho, JL Austerweil, Michael L Littman, and Josh B Tenenbaum. Coordinate to cooperate or compete: abstract goals and joint intentions in social interaction. In *Proceedings of the 38th annual conference of the cognitive science society*, 2016.

Angeliki Lazaridou, Alexander Peysakhovich, and Marco Baroni. Multi-agent cooperation and the emergence of (natural) language. In *International Conference on Learning Representations*, 2017.

Joel Z Leibo, Vinicius Zambaldi, Marc Lanctot, Janusz Marecki, and Thore Graepel. Multi-agent reinforcement learning in sequential social dilemmas. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pp. 464–473. International Foundation for Autonomous Agents and Multiagent Systems, 2017.

Adam Lerer and Alexander Peysakhovich. Maintaining cooperation in complex social dilemmas using deep reinforcement learning. *arXiv preprint arXiv:1707.01068*, 2017.

Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *arXiv preprint arXiv:1706.02275*, 2017.

Andrew Mao, Lili Dworkin, Siddharth Suri, and Duncan J Watts. Resilient cooperators stabilize long-run cooperation in the finitely repeated prisoner's dilemma. *Nature communications*, 8:13800, 2017.

Sean P Meyn and Richard L Tweedie. *Markov chains and stochastic stability*. Springer Science & Business Media, 2012.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

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 *International Conference on Machine Learning*, pp. 1928–1937, 2016.

Tamara Niella, Nicolás Stier-Moses, and Mariano Sigman. Nudging cooperation in a crowd experiment. *PloS one*, 11(1):e0147125, 2016.

Santiago Ontanón, Gabriel Synnaeve, Alberto Uriarte, Florian Richoux, David Churchill, and Mike Preuss. A survey of real-time strategy game ai research and competition in starcraft. *IEEE Transactions on Computational Intelligence and AI in games*, 5(4):293–311, 2013.

Aurélie Ouss and Alexander Peysakhovich. When punishment doesn't pay: 'cold glow' and decisions to punish. *Journal of Law and Economics*, 58(3), 2015.

Julien Perolat, Joel Z Leibo, Vinicius Zambaldi, Charles Beattie, Karl Tuyls, and Thore Graepel. A multi-agent reinforcement learning model of common-pool resource appropriation. *arXiv preprint arXiv:1707.06600*, 2017.

Alexander Peysakhovich and Adam Lerer. Prosocial learning agents solve generalized stag hunts better than selfish ones. *arXiv preprint arXiv:1709.02865*, 2017.

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

David G Rand, Alexander Peysakhovich, Gordon T Kraft-Todd, George E Newman, Owen Wurzbacher, Martin A Nowak, and Joshua D Greene. Social heuristics shape intuitive cooperation. *Nature communications*, 5, 2014.

Martin Riedmiller, Thomas Gabel, Roland Hafner, and Sascha Lange. Reinforcement learning for robot soccer. *Autonomous Robots*, 27(1):55–73, 2009.

Anton Schwartz. A reinforcement learning method for maximizing undiscounted rewards. In *Proceedings of the tenth international conference on machine learning*, volume 298, pp. 298–305, 1993.

Hirokazu Shirado and Nicholas A Christakis. Locally noisy autonomous agents improve global human coordination in network experiments. *Nature*, 545(7654):370–374, 2017.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

Ardi Tampuu, Tambet Matiisen, Dorian Kodelja, Ilya Kuzovkin, Kristjan Korjus, Juhan Aru, Jaan Aru, and Raul Vicente. Multiagent cooperation and competition with deep reinforcement learning. *PloS one*, 12(4):e0172395, 2017.

Gerald Tesauro. Temporal difference learning and td-gammon. *Communications of the ACM*, 38(3): 58–68, 1995.

Nicolas Usunier, Gabriel Synnaeve, Zeming Lin, and Soumith Chintala. Episodic exploration for deep deterministic policies: An application to starcraft micromanagement tasks. *arXiv preprint arXiv:1609.02993*, 2016.

Yuxin Wu and Yuandong Tian. Training agent for first-person shooter game with actor-critic curriculum learning. *arXiv preprint*, 2016.

6 TECHNICAL APPENDIX

6.1 PROOF OF MAIN THEOREM

We will use this basic property of almost sure convergence. If a sequence of random variables X_n converges to X almost surely then

$$\forall \delta > 0, \epsilon > 0 \; \exists n_0 \text{ s.t. } P(\forall n > n_0, |X_n - X| > \delta) < \epsilon$$

To make notation easier we can write that $\rho_{CC} > 0$ and $\rho_{CD} = 0$.

The intuition behind the proof is as follows: first, we show that if the CCC agent's partner chooses according to π^C then for any $\epsilon > 0$ and any slack $s > 0$ the CCC agent receives a payoff of at least $R_{init} = s + tT$ with probability $1 - \epsilon$. Intuitively this is because the rate T is lower than ρ_{CC} which is weakly lower than the rate guaranteed to a CCC agent whose partner behaves according to π^C always.

Second, we will show that for any $\epsilon > 0$, if the CCC agent's partner behaves according to π^C then we can find an R_{init} such that if a CCC agent's total payoff is at period t is at least R_{init} then with probability $(1 - \epsilon)$ the CCC agent also only plays π^C from that point on (this is given by almost sure convergence).

Together this implies that if the partner plays according to π^C then with probability $(1 - 2\epsilon)$ the CCC agent behaves according to π^D only a finite amount of times and thus the rates of payoffs for both agents converge to ρ_{CC}.

(Part 1)

Let R_t be the total reward of the CCC agent at time t and pick some R_{init}. We assumed that $\rho_{C+D,C} \geq \rho_{CC} > T$. Therefore if the partner plays π_C then $\frac{R_t}{t} \to \rho_{CCC,C} > T$. We set $\delta = \frac{\rho_{CC} - T}{2}$ and use almost sure convergence to say that

$$\forall \epsilon > 0 \; \exists t_s \text{ s.t. } P\left(\forall t > t_s, \frac{R_t}{t} > T + \frac{\rho_{CC} - T}{2}\right) > 1 - \epsilon.$$

Let $t'_s = \max\left(t_s, \frac{R_{init}}{(\rho_{CC} - T)/2}\right)$. Then after t'_s turns the CCC agents total reward will be above tT by at least R_{init} with probability at least $1 - \epsilon$.

(Part 2)

If we start at some R_{init} and assume that both players choose π^C forever then by reward ergodicity and the construction that $\rho_{CC} > T$ we have

$$\forall \epsilon > 0 \; \exists t_0 \text{ s.t. } P\left(\forall t > t_0, \frac{R_t}{t} > T\right) > 1 - \epsilon.$$

Assuming bounded rewards the CCC agent's accumulated reward in the first t_0 turns is no less than $-t_0 r_{max}$. So, if the CCC agent starts with $t_0 r_{max} + tT$, then with probability $1 - \epsilon$, their time average reward will never fall below tT and thus they will **always** play π^C. This validates the initial assumption that both players play π^C.

Putting it all together, for any $\epsilon > 0$ there exists a R_{init} that guarantees that if the CCC agent starts with total reward given by R_{init} then both players play π^C and get average rewards according to ρ_{CC}. That initial balance will be achieved with probability $1 - \epsilon$ after t'_s periods. Therefore, with probability at least $1 - 2\epsilon$, both players will achieve average reward of ρ_{CC}.

Defecting Doesn't Pay:

Suppose Agent 1's average reward converges to $\rho < T$. Then for some t_0 we have that $\frac{R_t}{t} < T$ for all $t > t_0$. Starting at t_0, CCC plays a fixed (stateless) policy π_1^D. Since π^D forms an equilibrium agent 2 can achieve average reward of at most ρ_{DD}.

6.2 A NOTE ON π^{C+D} PAYOFFS

While constructing CCC strategies, we make the assumption that the payoff of the pair (π^{C+D}, π^C) to agent 1 is at least ρ_{CC}. This is an overly restrictive condition, because it precludes methods of cooperation that take more than one period to execute. For example, if cooperation is walking to the left side of the room and defection is walking to the right side of the room, then a mix of the two policies may leave me in the middle gaining no reward.

In practice, CCC likely works even when this assumption doesn't strictly hold (e.g. in Pong and Coins), but with a slight modification to CCC we can use a much weaker condition: Let π^{kC+kD} be a policy that switches between π^C and π^D at most once every k periods. Then we must only assume that there exists k such that the payoff to agent 1 of (π^{kC+kD}, π^C) is at least ρ_{CC}. In other words, as long as the Agent plays long enough stretches of π^C or π^D against π^C, their reward will be at least $\rho_C C$.

The modified CCC uses two thresholds: $T_D = (1 - \alpha_D)\rho_{CC} + \alpha_D \rho_{CD}$ is the threshold to switch from π^C to π^D, and $T_C = (1 - \alpha_C)\rho_{CC} + \alpha_C \rho_{CD}$ is the threshold to switch back, with $\alpha_D > \alpha_C$. These thresholds grow linearly as $t \to \infty$ so the intervals between CCC switching policies has to grow linearly as well given bounded rewards.

6.3 FISHERY AND COINS TRAINING

The Fishery game has a 5×10 state space, where each agent is confined to, and observes, only their 5×5 half of the game. Coins is identical to Lerer & Peysakhovich (2017) but with the board size expanded to 8×8.

We follow the model architecture and training setup from Lerer & Peysakhovich (2017), with the modification that models are trained via policy gradient. We found that value-based methods (e.g. using a neural network baseline, or actor-critic) were unable to train prosocial policies in Fishery when using a stateless model. We suspect this is because states of very different prosocial value - 'I just ate the fish' vs. 'I let the fish swim to my partner' - are aliased in the observation space. Policy gradient with standard batch baselining produced converged policies for both Coins and Fishery. We trained Coins for 100,000 games and Fishery for 40,000 games.

For the tournament, agents play against opponents they have never trained against. The CCC statistics are computed at test time by rolling out a batch of 32 games for CC and 32 games for CD, updating the rollouts by one step per game step. Alternatively, the statistics (in particular, the defect threshold) could be pre-computed once for the longest game length of interest, and used in subsequent interactions.

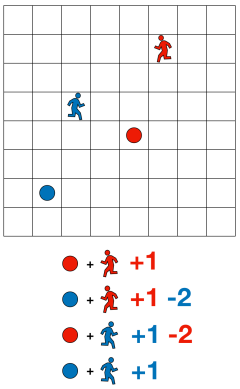
6.4 PONG PLAYER DILEMMA TRAINING

We use the ALE environment modified for 2-player play as proposed in Tampuu et al. (2017), with modified rewards of +1 for scoring a point and -2 for being scored on.

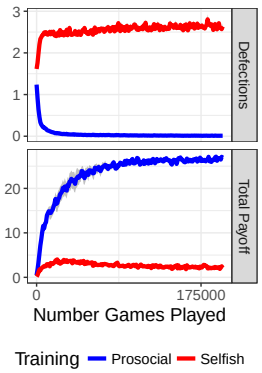
We train policies directly from pixels, using the pytorch-a3c package `https://github.com/ikostrikov/pytorch-a3c`.

Policies are trained directly from pixels via A3C (Mnih et al., 2016). Inputs are rescaled to 42x42 and normalized, and we augment the state with the difference between successive frames. We use 38 threads for A3C, over a total of 38,000 games (1,000 per thread). We use the default settings from pytorch-a3c: a discount rate of 0.99, learning rate of 0.0001, 20-step returns, and entropy regularization weight of 0.01.

The policy is implemented as a convolutional neural network with four layers, following pytorch-a3c. Each layer uses a 3x3 kernel with stride 2, followed by ELU. The network has two heads for the actor and critic. We elide the LSTM layer used in the pytorch-a3c library, as we found it to be unnecessary.



(a) Coins



(b) Training Curves

Player Strategy \ Partner Strategy	π^C	π^D	CCC
π^C	22	-16	25
π^D	40	2	9
CCC	24	0	21

(c) Tournament Results

Figure 3: In Coins two agents, Red and Blue, live on an 8×8 grid and move around in all cardinal directions. Coins randomly appear on the board and each have a color. If an agent picks up (moves over) any coin, they receive one point. However, if they pick up a coin of the other agent's color, that agent loses 2 points. Selfish agents learn to grab all coins but prosocial agents play the strategy of only picking up coins of their own color. We see that CCC agents can avoid exploitation and maintain cooperation with cooperators and other CCC agents. Panel C shows the numerical payoffs for an agent choosing one strategy (row) against a partner who chooses the strategy indicated by the column.

6.5 Experiment: Coins

We show that CCC maintains cooperation in the Coins game from Lerer & Peysakhovich (2017). In Coins two agents, Red and Blue, live on an 8×8 grid and move around in all cardinal directions[7]. Coins randomly appear on the board and each have a color. If an agent picks up (moves over) any coin, they receive one point. However, if they pick up a coin of the other agent's color, that agent loses 2 points.

In Coins, the cooperative strategy is to pick up coins of one's color only. However, there is always a temptation to grab the other agents' coins. We train 10 copies of each strategy type under our two reward schemes and again compare payoffs of various policy combinations (see Figure 3). Note that Coins is a fully observed Markov game, however we still see that CCC (which has only limit guarantees and throws away all 'forward' information contained in an action) is just as effective as the intention based TFT strategies (which incentivize cooperation at all time periods) either built using Q functions (Lerer & Peysakhovich, 2017) or learned via an end-to-end procedure (Foerster et al., 2017c).

[7]We use a larger grid size so the payoffs are not directly comparable to those reported in Lerer & Peysakhovich (2017)