arXiv:1707.01068v1 [cs.AI] 4 Jul 2017

Maintaining cooperation in complex social dilemmas using deep reinforcement learning

Adam Lerer*
Facebook AI Research

Alexander Peysakhovich*
Facebook AI Research

Abstract

In social dilemmas individuals face a temptation to increase their payoffs in the short run at a cost to the long run total welfare. Much is known about how cooperation can be stabilized in the simplest of such settings: repeated Prisoner's Dilemma games. However, there is relatively little work on generalizing these insights to more complex situations. We start to fill this gap by showing how to use modern reinforcement learning methods to generalize a highly successful Prisoner's Dilemma strategy: tit-for-tat. We construct artificial agents that act in ways that are simple to understand, nice (begin by cooperating), provokable (try to avoid being exploited), and forgiving (following a bad turn try to return to mutual cooperation). We show both theoretically and experimentally that generalized tit-for-tat agents can maintain cooperation in more complex environments. In contrast, we show that employing purely reactive training techniques can lead to agents whose behavior results in socially inefficient outcomes.

1 Introduction

A key component of human sociality is the ability to maintain long run cooperative relationships with others when both partners face short term incentives to defect. In such situations humans often use the 'shadow of the future' - rewarding a partner's cooperation today with our cooperation tomorrow - as a way to solve these social dilemmas (Axelrod, 2006; Nowak, 2006; Fudenberg & Maskin, 1986). Here we investigate how to construct agents that can maintain cooperation in this way while avoiding being exploited by cheaters.

Recent work in deep reinforcement learning (RL) has made great strides towards artificial agents that can learn to achieve goals in complex environments. However, the biggest successes in this literature have come in environments that are either single-agent (e.g. Atari (Mnih et al., 2015)), zero-sum[2] or coordination games without a temptation to defect.[3]

Outside of deep RL there is a large body of theoretical (Axelrod, 2006; Fudenberg & Maskin, 1986), simulation-based (Nowak, 2006; Macy & Flache, 2002) and experimental (Bó, 2005; Fudenberg et al., 2012) work studying the emergence of bilateral cooperation. A weakness of this research program is

*Equal contribution. Author order was determined via random.org.

[2]Zero sum games are those where for one player to win, another player must lose. These include Backgammon (Tesauro, 1995), Go (Silver et al., 2016), first person shooter games (Kempka et al., 2016; Wu & Tian, 2016), poker (Heinrich & Silver, 2016) and StarCraft (Ontanón et al., 2013; Foerster et al., 2017; Usunier et al., 2016).

[3]This class of games are games where both players win or lose together. It includes team-based coordination games (Lowe et al., 2017), sports like soccer (Riedmiller et al., 2009) as well as the growing RL-based literature on language emergence in games (Lazaridou et al., 2017; Das et al., 2017; Evtimova et al., 2017; Havrylov & Titov, 2017; Jorge et al., 2016).

Working paper.

that it almost exclusively studies simplified matrix-based games such as the two action, stochastically repeated Prisoner's Dilemma (PD).[4]

Our key contribution is to take the game theoretic insights gleaned from the simple PD and combine them with tools from deep reinforcement learning to construct agents that are capable of solving arbitrary bilateral social dilemmas via the shadow of the future. We do this by generalizing a simple yet successful strategy for stochastically repeated PD play: tit-for-tat (Axelrod, 2006).

Tit-for-tat (TFT) works by copying the prior behavior of their partner. If a partner cooperated last period, TFT cooperates, otherwise it defects. TFT has many desirable properties: it is clear (the partner understands it), nice (it begins by cooperating), provokable (it cannot be exploited by a pure defector), and forgiving (following a bad turn in the relationship TFT returns to cooperation if the partner returns to cooperation).

A central feature of TFT is that if one agent can credibly commit to TFT then for their partner the strategy of always cooperating dominates all other policy choices. This property is important if we are going to be designing artificial agents that will interact with others (including humans) in real situations. If the artificial agent is a form of TFT then there is a clear dominant strategy for their partner: there is no gain from coming up with ways to cheat the system, just be nice and everyone will get ahead.

Implementing a TFT-like strategy is non-trivial for general stochastic games. For example, consider what will be our workhorse situation - the Coin game. In the Coin game two players, Red and Blue, move on a 2D board. Red and blue coins appear on the board periodically, and a player receives a reward of 1 for collecting (moving over) any coin. However, if a player picks up a coin of the other players' color, the other player loses 2 points. The strategy which maximizes total payoff is for each player to only pick up coins of their own color; however each player is tempted to pick up the coins of the other color. Implementation of TFT in this environment is tantamount to maintaining a deal of 'I won't touch your coins if you don't touch mine.' Note that an agent which commits to behave only in a pro-social way (ie. always only match their own color) can be exploited by an unscrupulous cheater.

It is easy for a human to look at the game and recognize the pro-social and selfish strategies as well as come up with a mutually beneficial and stable policy. Humans are good at navigating real world social dilemmas using social heuristics that have been tuned via evolution and social learning (Tomasello, 2009; Hoffman et al., 2015; Peysakhovich & Rand, 2015; Rand et al., 2014) . Our goal will be to give our artificial agents simple versions of these heuristics.

We observe that the naive strategy of training purely reactive policies for the agents by reinforcement learning does not converge to socially optimal outcomes in the Coin game. This can be explained by two factors. First, in order to maintain cooperation from a selfish agent, there must be consequences tomorrow for defection today. In Markov games, there is not always sufficient information in the state today to sufficiently reveal past behavior. Therefore, there may not be Nash equilibria which achieve socially optimal payoffs. This can be solved via brute force by simply appending to the state a history of all past states and actions. However, this means the state and policy spaces will grow exponentially in memory size.

Second, even when there is sufficient information in the state to maintain a socially optimal payoff, there is no guarantee that standard RL methods will converge to these strategies. Even in the simple repeated PD the folk theorem states that there can be equilibrium trajectories which involve cooperation, equilibrium trajectories of pure defection and even equilibrium trajectories which only cooperate on time periods which are factorizable into small primes (Fudenberg & Maskin, 1986).

Our approach sidesteps these issues by reducing an arbitrary social dilemma to a PD-like representation. We do this by first reducing the policy space to just a few potential Markov policies. We then

[4]Some modern work has gone beyond the simple PD setup. Kleiman-Weiner et al. (2016) propose a model that infers the intention of other agents and decides whether to cooperate. Constructing this model uses some similar techniques to what we propose but the research is primarily concerned with modeling what humans actually do rather than constructing artificial agents or formally proving the properties of certain strategies. Leibo et al. (2017) study standard RL agents in more complex social dilemmas but are primarily concerned with describing how incentives affect the emergence of cooperative or non-cooperative behavior. There has also been recent interest in natural language bargaining which is far less structured than a PD but also not zero-sum (Lewis et al., 2017).

use sufficient summary statistics of past behavior to create a meta-policy which chooses which of the reduced policies to follow at any time. We construct a generalization of TFT for the theoretical case where the agent has access to the complete value function of the game under any state and set of policies. We show analytically that this generalization can maintain cooperation with selfish or pro-social partners. We call this generalization Markov TFT (mTFT),

In practice, agents do not have access to the true value functions. Instead, we often have access to the game to learn approximate value functions via RL methods. These value functions are not general, rather they are conditioned on a policy or set of policies. To deal with this we propose an approximate version of mTFT (amTFT) which retains most of its incentive properties but is tractable to compute.

amTFT is constructed as follows: at training time the agent receives the game and uses standard self-play to compute two sets of Markov policies (maps from current game state to action). The first set of policies is consistent with all players acting in a way to maximize the sum of rewards (we call these the Cooperation policies) and one which is consistent with everyone selfishly optimizing as if the game were a one-player Markov Decision problem (we call these the Selfish policies because they treat the other agent as a part of the environment rather than as something that can be reasoned with).

At test time, when faced with an actual partner, the amTFT agent keeps track of whether their partner's behavior is consistent with Cooperative (C) or Selfish (D) policies. If the partner is behaving according to C, the agent responds by also behaving according to C. If the partner behaves according to D, the agent responds by behaving according to D for long enough such that any gains the partner has gotten from trying to cheat are erased. We use learned value functions and Monte Carlo policy rollout to implement the computations required for this switching. We discuss conditions on the stochastic game under which the approximate version maintains the incentive properties of the full mTFT.

2 The Model

We now turn to formalizing the discussion above. We begin by a brief, and no means complete, introduction to the basics of cooperation in Prisoner's Dilemma (PD) games. The familiar reader can feel free to skip this subsection.

2.1 Cooperation in the Repeated Prisoner's Dilemma

The PD is the workhorse game of the cooperation literature. The simplest PD has 2 players each of which has two strategies, cooperate (C) or defect (D). If both players choose C then they both receive a payoff of 1, if both choose D, they both receive a payoff of 0. If one chooses C and the other chooses D the cooperator receives the sucker payoff $-s$ and the defector receives the payoff $1 + w$. This is illustrated in the payoff matrix below, with each element of each tuple representing the rewards accruing to the row and column players respectively.

		Player Y	
		C	D
Player X	C	$(1, 1)$	$(-s, 1 + w)$
	D	$(1 + w, -s)$	$(0, 0)$

The dominant strategy is D (meaning the only Nash equilibrium of the one-shot game is for both to defect) while the socially optimal strategy is for both to cooperate.[5] Cooperation can be maintained if the PD is repeated an infinite number of times[6] and the players discount future payoffs with rate δ.

[5]We assume $s, w > 0$ and $s > w$ otherwise the fair optimal strategy is to cycle between one player cooperating and the other defecting forever.

[6]Note that finite repetition (with known termination time) cannot sustain cooperation since by backward induction both players will defect at the last period which means both should defect at the period before which means both should defect the period before that and so on. While it is known that humans do not perform backward induction there is strong evidence that when finitely repeated games are played repeatedly (that is, each super-game is played with new partners) cooperation does unravel (Selten & Stoecker, 1986).

An alternative interpretation of these conditions is that $1-\delta$ can represent the probability that the game continues another period.

It is straightforward to see that cooperation can be maintained with the following strategy if δ is large enough:

Definition 1 *A Grim trigger strategy is one which cooperates in the first round and in any round where the opponent has always cooperated before. If the opponent has ever defected, Grim trigger defects.*

Both players playing Grim trigger is a perfect Nash equilibrium in the repeated game - after any possible history, neither player can deviate from the Grim strategy to increase their own payoff, and, on the trajectory of play Grim will indeed implement cooperation. However, note that any slight amount of noise will inevitably destabilize Grim.

Another, well studied strategy is tit-for-tat (TFT). TFT works by copying the last period play of their partner. Thus, if a partner cooperated last period, TFT cooperates, otherwise it defects. TFT is simple but came to prominence when Robert Axelrod invited game theorists to submit strategies for the repeated PD that could be implemented using computer programs. These strategies were then played against themselves and others. TFT won the tournament and has been studied relentlessly ever since. TFT has many desirable properties: it is clear (the partner understands it), nice (it begins by cooperating), provokable (it cannot be exploited by a pure defector), and forgiving (following a bad turn in the relationship TFT returns to cooperation if the partner returns to cooperation). This also implies that in the repeated PD, if one's partner is committed to TFT, the best policy is to always cooperate, no matter the current state.

Note that both individuals choosing TFT is not a subgame perfect equilibrium. For example, if two TFT players find themselves in a state where one defected last period and the other cooperated they will cycle between the (C, D) and (D, C) states forever (and one of the TFT players can do better by deviating to pure cooperation at that point). There is much work in expanding TFT or finding other equally simple strategies to deal with this issue (Nowak & Sigmund, 1993; Fudenberg et al., 2012). Nevertheless, TFT is a simple starting point for strategies that maintain cooperation in more complex environments.

2.2 Markov Games

We now move to extending the intuition developed above to more complex games. We can extend simple repeated games to subsume cover the Markov Decision Problem (MDP) setups as follows:[7]

Definition 2 (Shapley (1953)) *A (2-player) Markov game consists of*

- *A set of states* $S = \{s_1, \dots, s_n\}$
- *A set of actions for each player* $\mathcal{A}^1 = \{a^1_1, \dots, a^1_k\}$, $\mathcal{A}^2 = \{a^2_1, \dots, a^2_k\}$
- *A transition function* $\tau : S \times A_1 \times A_2 \to \Delta(S)$ *which tells us the probability distribution on the next state as a function of current state and actions*
- *A reward function for each player* $R_i : S \times A^1 \times A^2 \to \mathbb{R}$ *which tells us the utility that player gains from a state, action tuple*

A policy for each player is a map $\pi^i : S \to \mathcal{A}^i$. Note that here we abuse notation and suppress the stochastic nature of transition functions (so there are lots of places where there should be an $\mathbb{E}$) and we assume deterministic policies. It is easy to extend the discussion below to stochastic ones but only at the cost of additional notation. A value function for a player inputs a state and a pair of policies $V^i(s, \pi^1, \pi^2)$ and gives the expected discounted reward to that player from starting in state s. We assume agents discount the future with rate δ which we subsume into the value function.

Definition 3 *A policy for agent j denoted π^j is a best response to a policy π^i for agent i if for any π'^j and any s we have*

$$V^j(s, \pi^i, \pi^j) \geq V^j(s, \pi^i, \pi'^j).$$

[7] In the game theory literature such games are called stochastic games (Shapley, 1953; Dutta, 1995) while in the RL literature they are called Markov games. In this paper we use the Markov game terminology.

We denote the set of best responses to π^i as $BR^j(\pi^i)$.

Extending the standard definition of a Nash equilibrium to this game is:

Definition 4 (Dutta (1995)) *A Markov equilibrium is a pair of Markov policies (π^1, π^2) such that $\pi^1 \in BR^1(\pi^2)$ and $\pi^2 \in BR^2(\pi^1)$.*

Definition 5 *Socially optimal Markov policies (π_C^1, π_C^2) are those which, starting from any state s, maximize*

$$V^1(s, \pi^1, \pi^2) + V^2(s, \pi^1, \pi^2).$$

For simplicity we assume that the game in question has a single pair of socially optimal Markov policies. When there is a multiplicity of such policies, coordination problems can arise (Kleiman-Weiner et al., 2016). Solving these coordination problems (eg. by allowing communication within each time step) is a task beyond the scope of this paper.

The conditions for socially optimal policies to be Markov equilibria are well known in the stochastic games literature. It amounts to requiring that, if players face a short term incentive to stray from the policy, action histories can somehow be baked into the state (for example, if an action causes the destruction of something in the game environment). Note, however, that if players are only able to condition on states in the MDP we cannot implement socially optimal outcomes as equilibria in general.

Intuitively, maintaining cooperative play at time t requires knowledge that deviating from this path will induce a loss of payoffs down the road. However, in the same way that the Markov chain that walks into a bar doesn't know how it got there, agents, in general, won't know if they got to a state because their partner did one or another set of actions. This means agents cannot punish or reward past behavior appropriately. One solution is to consider an expanded set of policies which condition on a history of states and actions. However, in even moderately sized MDPs this means that the policy space will grow exponentially. We consider minimal such expansions to solve the social dilemma problem.

Definition 6 *A simple memory augmentation to a Markov game is a function*

$$M^i : \mathbb{R}^2 \times S \times \mathcal{A}^1 \times \mathcal{A}^2 \rightarrow \mathbb{R}^2.$$

This allows our agents to choose policies as functions given by

$$\pi^i : S \times M^i \rightarrow A^i.$$

We call a tuple (M^i, π^i) an augmented Markov policy.

The simple memory augmentation is itself Markov and thus can be thought of as appending a two dimensional real valued vector to the state vector (the strategies below can be implemented with a single scalar but we keep 2 to make intuition simple and notation clear). We can think of this state, for each agent, as summarizing the current state of the relationship. This allows agents to condition their actions on both the state of the environment and a summary of past behavior. We extend the definition of a Markov equilibrium to using some augmented Markov policies.

Definition 7 *Given two policies and a starting state s and policies π^1, π^2 a trajectory is the set of actions and states that result from starting in that state and having both individuals follow those policies. A cooperative trajectory is one where at each state both individuals play according to socially optimal Markov policies.*

Definition 8 *Given a pair of policies π^1, π^2 and a state s, agent i and action a for that agent the Q function is defined as*

$$Q^i(s, a, \pi^1, \pi^2) = R^i(s, a, \pi^j) + \delta V^i(\tau(s, a, \pi^j(s)), \pi^1, \pi^2).$$

Q represents the expected reward gained from choosing a now and then continuing according to π^i.

Definition 9 *Fix agent i and the policy of the other agent. Given policy π^i and state s' define a one shot deviation at s is behaving according to π^i everywhere except choosing a' at state s. We say that a one shot deviation is q-profitable if*

$$Q^i(s, a, \pi^1, \pi^2) = V^i(s, a, \pi^1, \pi^2) + q.$$

Definition 10 *We say that a trajectory t starting at s is implementable if there exists augmented Markov policies which are an equilibrium and generate the trajectory t.*

Theorem 1 *The cooperative trajectory t resulting from policies (π_C^1, π_C^2) starting from state s in a Markov game is implementable if for either agent i and any point on the cooperative trajectory and any one-shot deviation a'^i which is $q-$profitable we have that there exists a Markov equilibrium (π_M^1, π_M^2) starting from the next state $\tau(s, a')$ such that*

$$\delta V^i(\tau(s, a', \pi_C^j(s)), \pi_M^1, \pi_M^2) + q < V^i(s, \pi_C^1, \pi_C^2).$$

The proof of theorem 1 is essentially a generalization of the Grim trigger strategy above combined with the one shot deviation principle (Fudenberg & Tirole, 1991). The one shot deviation principle states that in an extensive form game if there are no one shot deviations from a given set of strategies for either player, then those strategies constitute an equilibrium. See Appendix for full details.

The Grim equilibrium proceeds as follows. Both agents play the cooperative policies. If either agent deviates, they move to playing the worst Markov equilibrium possible starting from the next state. The memory augmentation function here is trivial: M^i takes the value 1 if everything has proceeded according to the prescribed path and 0 otherwise. The prescribed augmented strategies implement a Grim trigger.[8]

As with the PD, Grim is very fragile to noise either in action implementation or in any form of policy/value function approximation. In particular, there can be many states where continuation values from any action are very close or even identical. However, π_C prescribes one trajectory exactly. This means that just a slight bit of noise in our function approximation can cause enough joint misunderstandings such that the trigger is activated and cooperation ends.

Grim's lack of robustness leads to a practical question: given a Markov game where a cooperative trajectory is implementable using the Grim policies above, can we use the Grim strategies and construct a TFT-like strategy which, though it may not be a symmetric equilibrium (for the same reasons as TFT), will have the good robustness properties if one agent can commit to it?

2.3 Oracle Markov TFT

For illustration, suppose that agent 1 has access to an oracle with infinite compute as well as access to the true value functions of the game Q^1, Q^2. In this case we define Markov tit-for-tat (mTFT) as the following construction. We use the oracle and value functions can be used to construct Markov policies (π_C^1, π_C^2) and (π_M^1, π_M^2) which are the cooperative and worst case Markov equilibrium policies respectively.

We make one additional restriction on the payoffs which generalizes a property of the PD: if one agent is cooperating and one agent is behaving according to π_M then the cooperating agent is worse off than when both cooperate. Formally, this is:

Definition 11 *Say that π_M withholds cooperation if for either agent i and any state s if $\pi^i = \pi_C^i$ and $\pi^j = \pi_M^j$ then*

$$V^i(s, \pi_C^i, \pi_M^j) < V^i(s, \pi_C^i, \pi_C^j).$$

The mTFT agent keeps a counter b which they initialize at 0. While the counter is at 0 the agent behaves according to π_C (we refer to this as the C phase). While the counter is above 0 the agent acts according to π_M and decrements the counter by one every time step (we refer to this as the D phase). If mTFT's partner deviates from π_C to some action a' at any point, the mTFT agent adds $k(s, a', b)$ to the counter. k is chosen using the partners' Q function to make sure that whatever the partner gained from choosing a' instead of the π_C action is erased by an additional k time steps of the mTFT agent choosing according to π_M and the partner continuing according to π_C.

mTFT allows us to preserve the incentive properties of the Grim Trigger above without the associated fragility – a single mistake in Grim Trigger ends cooperation forever whereas mTFT always gives

[8]The careful reader will notice that the construction can be used to implement any trajectory that yields payoffs above some 'min-max' payoffs. This is explored in the game theory literature on folk theorems (Fudenberg & Maskin, 1986) however is beyond the scope of this work.

a way to repair any deviations in finite time. Note that even if the mTFT agent is already playing according to π_M deviations from π_C still cause increments of the counter. The full pseudocode for the mTFT meta-policy is in algorithm 1.

Algorithm 1 Oracle Markov Tit For Tat (for agent 1)

```
Input: Oracle value functions and k^1(·) function
b ← 0
while Game do
    if b = 0 then
        Choose a^1 = π^1_C(s)
    if b > 0 then
        Choose a^1 = π^1_M(s)
        b = b − 1
    if a^2 ≠ π_C(s) then
        b = b + k^1(s, a^2, b)
```

2.4 Approximate Markov TFT

In most applications of RL, we do not have access to the true value functions of the game. Rather, we have access to the environment itself and then learn policies as well as their associated value and Q functions. We can use RL methods to construct the components required for mTFT by approximate the cooperative and post-deviation policies.

To construct approximations of the required policies we use self-play and two reward schedules: selfish and cooperative. In the selfish reward schedule each agent i treats the other agent just as a part of their environment and tries to maximize their own reward. This is the standard way of training RL in the various multiagent situations described in the introduction of this paper.

We assume RL training converges and we call the converged policies under the selfish reward schedule $\hat{\pi}^i_D$ and the associated Q functions $\hat{Q}^i_{DD}$. [9] Note that convergence with this training implies that $\hat{\pi}^1$ is the best response of agent 1 if agent 2 behaves according to $\hat{\pi}^2_D$ and vice-versa. Thus, they form a Markov equilibrium (up to function approximation).

We also train policies and value functions using reward schedule C. Here each learning agent gets rewards both from their own payoff and the rewards the other agent receives. That is, we modify the reward function so that it is

$$R_i^{CC} = R_1 + R_2.$$

We call the converged policy and value function approximations $\hat{\pi}^i_C$ and $\hat{Q}^i_{CC}$. Note that here convergence guarantees are much easier to get – we can think about this training schedule is that one super-agent controls both players and tries to optimize for a single scalar reward. From the point of view of the super-agent the environment is again a stationary MDP and thus we can get standard approximation and convergence guarantees.

Unfortunately, the constructed policies are not enough to construct the mTFT meta-policy in general. This is because RL methods typically do not learn the full value functions for the game, rather they learn the value function(s) conditional on a particular set of policies. These value functions may not have enough information to implement the mTFT strategy.

The reason mTFT cannot be implemented with the learned functions alone is that the mTFT agent may see their partner deviate from π_C in a D phase. When this happens, the mTFT agent needs to compute an additional k time steps to add to the current D phase. However, in order to compute what this k should be the mTFT agent needs access to not just the value functions associated with π_C and π_D but also the value functions associated with the policy given by "behave according to π_D for k

[9] We note that in general versions of this it is difficult to prove convergence results as well as stabilize the training procedure (Fudenberg & Levine, 1998). This is because if both agents are learning the environment from the perspective of each agent is not stationary. In our environment we find that standard tools appear to work though in more complex environments we may need more complex training procedures to construct these policies (cf. Foerster et al. (2017)).

periods and π_C afterwards" (we denote this as $\pi_{D_k C}$). These value functions can be computed using standard methods, but it becomes quite cumbersome.[10]

If we restrict ourselves to a subset of games which generalize another property of the Prisoner's Dilemma, we can use a much simpler training procedure and meta-policy while keeping approximately preserving mTFT's incentive properties.

Definition 12 *We say a game is $\hat{\pi}_D$ dominant if for any k we have*

$$\hat{\pi}^i_D \in BR^i(\hat{\pi}^j_{D_k C}).$$

The $\hat{\pi}_D$ dominance assumption can be loosely translated to the idea that there is a single dominant way to cheat one's partner and that the policies learned by purely reactive training find that way.

When $\hat{\pi}_D$ is dominant we can use a much simpler construction which only requires learning $\hat{\pi}^i_D$ and $\hat{\pi}^i_C$ and their associated value and Q functions. The key idea of the construction will be that due to the dominance of π_D we can use the continuation values implied by π_D to bound the length of a D phase following a deviation from π_C. We call this simplified algorithm *approximate mTFT* (amTFT).

With the value functions and policies in hand from the procedure above, we can construct an amTFT meta-policy. For the purposes of this construction, we consider agent 1 as the amTFT agent (but everything is symmetric). The amTFT agent augments the Markov state with a vector (W_0, b_0) which both start at 0. We use the algorithm below to combine pieces computed above into the approximate mTFT strategy.

The amTFT agent sees the action a' of their partner at time t and approximates the gain from this deviation as

$$D_t = \hat{Q}^2_{CC}(s, a^2_t) - \hat{Q}^2_{CC}(s, \pi^2_C(s)).$$

This is approximate total change in (both present and future) value that the partner receives from choosing action a^1_t instead of what is prescribed by $\hat{\pi}_C$.

The amTFT agent accumulates the total payoff balance of their partner as

$$W_t = W_{t-1} + D_t.$$

If W_t is below a fixed threshold T the amTFT agent chooses actions according to π_C. If W_t crosses the threshold the mTFT agent computes the expected value to their partner of the policy pair $(\pi^1_{D_k C}, \pi^2_{D_k C})$ starting at s. To do this, we can, for each k simply perform Monte Carlo policy rollouts to estimate the value of k periods of playing π_D followed by the continuation value from the resulting state implied by the learned $\hat{V}_{CC}$.

Note that because the game is π_D dominant this is an upper bound on the partners' payoff if the amTFT agent commits to playing according to $\pi_{D_k C}$.

The amTFT agent uses this rollout strategy to compute the minimal k such that the loss to the deviator from $\pi_{D_k C}$ is greater than αW^1_t where $\alpha > 1$ allows us to extend the D phase to account for approximation errors in our calculation of Q and V. If our approximations are good, it is now no longer worth it for the amTFT's partner to try to cheat while on the π_C path. The pseudocode for the meta-policy is given in algorithm 2 (see the Appendix for the algorithm described as a finite state automata).

The hyperparameters T and α trade off robustness to approximation error and noise. Tuning these parameters is important but a complete characterization is beyond the scope of this paper.

We note that amTFT has one main difference from the mTFT above: when standard mTFT is acting according to π_D it still expects the partner to play according to π_C. On the other hand the approximate version we present does not. This means that the approximate algorithm does not make cooperation a strictly dominant strategy (it is better for the other agent to play π_D during the D phase).

However, the likelihood of two mTFT agents getting locked in cycles of cooperation and defection is reduced. In this way, the approximate algorithm is more similar to the strategy of Win-Stay-Lose-Shift

[10] We note that this is learnable with the following algorithm: first learn $\hat{\pi}_D$ and $\hat{\pi}_C$, then train a new RL agent on episodes where their partner plays according to $\pi_{D_k C}$ where k is observed. This will allow us to approximate the best response policies to $\pi_{D_k C}$ which will then give us what we need to compute the responses to deviations from π_C in the D phase that incentivize full cooperation.

Algorithm 2 Approximate Markov Tit For Tat (for Agent 1)

Input: $\hat{\pi}_C, \hat{\pi_D}$ and their $\hat{Q}; \alpha, T$
$b \leftarrow 0, \bar{W} \leftarrow 0$
while $Game$ **do**
  $D \leftarrow \hat{Q}^2_{CC}(s, a^2) - \hat{Q}^2_{CC}(s, \hat{\pi}^2_C(s))$
  if $b = 0$ **then**
    Choose $a \leftarrow \hat{\pi}^1_C(s)$
    $\bar{W} = \bar{W} + D$
  if $b > 0$ **then**
    Choose $a \leftarrow \hat{\pi}^1_D(s)$
    $b = b - 1$
  if $\bar{W} > T$ **then**
    Compute $k(s)$ via batched policy rollouts and $\hat{Q}^2$
    $b = \hat{k}(s)$
    $\bar{W} = 0$

in classical repeated Prisoner's Dilemma (Nowak & Sigmund, 1993).[11] Additionally, the approximate algorithm saves on training complexity since we no longer need to compute best responses to $\pi_{D_k C}$.

2.4.1 Computational Limitations

A key component of the amTFT strategy is the computation of single action deviation gains D. One way is to use estimated $\hat{Q}$ directly. This has the issue that any approximation error $\hat{Q}$ (in particular, bias) is accumulated across periods. An unbiased but more computationally intensive estimator of D is to use Monte Carlo policy rollouts of some length m followed by the continuation value. A third way is to elide the continuation value and simply use the value from the rollout and set m large enough such that continuation values after that many time steps are approximately the same. This trades off between computational complexity and accuracy.

In our experiments we use policy rollouts to compute the approximation of Q and elide the continuation value. We tried to learn $\hat{Q}$ directly during training but noticed that our actor-critic training led to $\hat{Q}$ that was poorly-approximated anywhere substantially off of trajectories induced by the learned policy. This problem can be ameliorated by fine-tuning $\hat{Q}$ only in a second stage where more exploration is allowed around the learned policies (e.g. self-play against a combination of $\hat{\pi}_D$, $\hat{\pi}_D$ with ϵ-greedy action choice). Solving this problem is important for future applications of these ideas but is beyond the scope of this paper.

3 Experiments

We implement amTFT in a simple situation - the Coin game (figure 1). In the Coin game two players, Red and Blue, move on a 5×5 board. The game has a small probability of ending in every period. Red and blue coins appear on the board periodically, and a player receives a reward of 1 for collecting (moving over) any coin. However, if a player picks up a coin of the other players' color, the other player loses 2 points. The payoff for each agent at the end of each game is just their own point total. The strategy which maximizes total payoff is for each player to only pick up coins of their own color; however each player is tempted to pick up the coins of the other color.

We are interested in constructing general strategies which scale beyond tabular games so we approximate the policy and value function with a convolutional neural network. This network is a stack of four subunits, each of which contains a strided convolutional layers with 3×3 kernel, batch normalization, and ReLU. This produces a 104-dimensional feature vector per frame. The network has three heads: a 4-dimensional output with softmax that computes the policy π; a 4-dimensional output that computes $\hat{Q}$, and a 1-dimensional output that computes the critic $\hat{V}$ (used only for training).

[11] amTFT does not correspond to a generalization of WSLS either since WSLS only goes back to cooperation after both agents have chosen D once whereas amTFT does not require any particular behavior from their partner during any D phase.

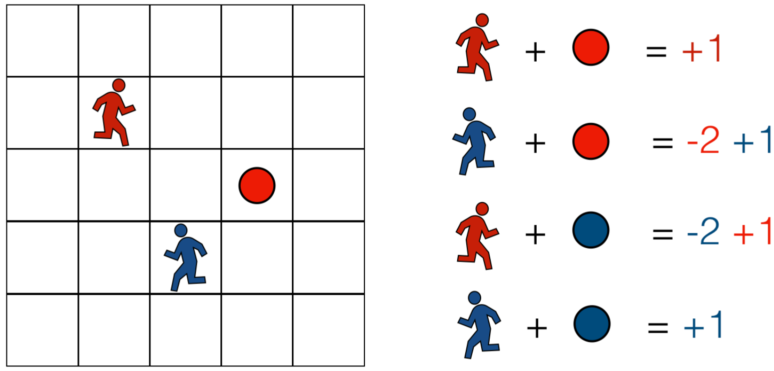

Figure 1: Example of the Coin game. Agents of two colors move on a 5×5 board and pick up coins. Picking up a coin of a matching color gives 1 point to the agent, picking up a coin of the opposing color gives 1 point but causes the other agent to lose 2 points.

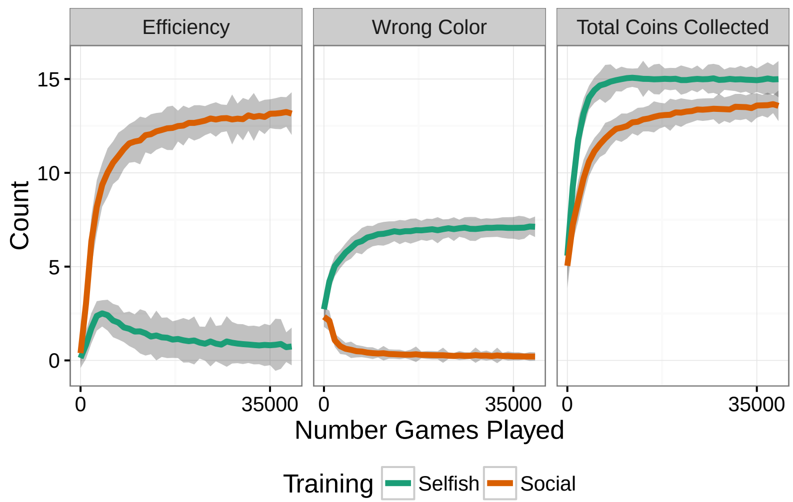

Figure 2: Selfish training leads to agents which pick up all coins (middle, right panels). This leads to poor social efficiency (sum of per game agent payoffs, left panel). On the other hand, our π_C training procedure produces policies which work together and indeed maximize social efficiency.

For each of $\hat{\pi}_C$ and $\hat{\pi}_D$, we trained a neural network function approximator via actor-critic self-play with the appropriate reward structure for the policy. We train the value function head $\hat{V}$ via the Monte-Carlo estimate where the loss of V given an episode is described by

$$\mathcal{L}(\hat{V}) = \left(\hat{V}(s) - \sum_k \delta^k r_{t+k}\right)^2.$$

Figure 2 shows selfish training leads to suboptimal behavior while cooperative training does find policies that implement socially optimal outcomes. Total reward (which we call efficiency) is near 0 when both agents play according to $\hat{\pi}_D$ even though the number of total number of coins being collected in each game is almost identical. This is because $\hat{\pi}_D$ agents converge to picking up coins of all colors while social $\hat{\pi}_C$ agents learn to only pick up matching coins.

As with the origins of TFT we evaluate the performance of various Markov social dilemma strategies in a tournament. To do so we train 20 copies of the agents described above (this gives us variation in

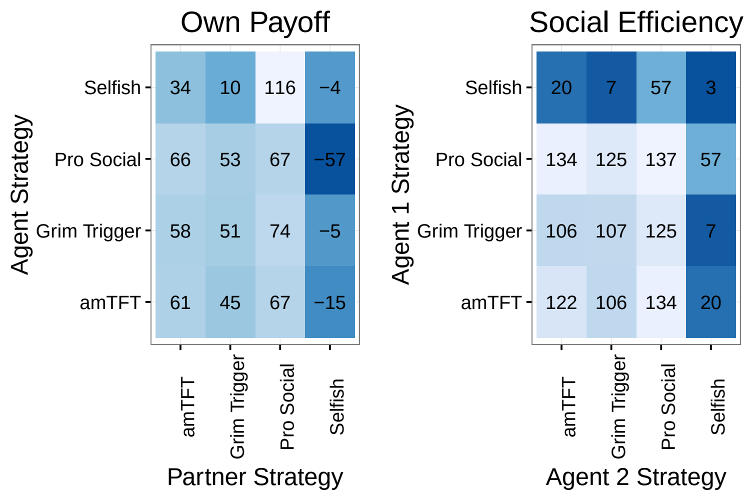

Figure 3: Left panel shows the payoffs to the row player from choosing a strategy when their partner chooses the strategy indicated by the column. We see that defectors can exploit purely cooperative players. However, cooperation is a better strategy than selfishness with Grim or amTFT partner. Left panel shows the social efficiency (total payoffs attained) when strategies are matched against each other. We see that amTFT maintains more efficient levels of cooperation than Grim Trigger strategies and can almost reach the level of pure pro-social agents.

the forms of π_C, π_D, etc... that these agents learn). We then construct matchups between augmented memory by drawing two random agents from the pool and giving them meta-strategies from the set

1. Always use π_C - we refer to this as the Pro Social strategy
2. Always use π_D - we refer to this as the Selfish strategy
3. Approximate Markov Tit-for-Tat (amTFT)
4. Markov Grim Trigger (with a threshold)

We construct Markov Grim Trigger by using the approximate mTFT algorithm above but setting $\hat{k}(s) = \infty$. That is, mGrim uses the same computation as mTFT but unlike mTFT once Grim has crossed into the D phase, cooperation never returns. We set the threshold T to 1 and multiplier α to 4. To construct a matchup between two strategies we draw agents at random from our-pretrained pool and have them play a 1000 round iteration of the game. At test time we use a fixed (rather than random) length game because this allows us to compare payoffs more efficiently. In total we generate 40 matches per pair of strategies.

The results of the tournament are visualized in figure 3. Defectors can exploit pro-social players at huge gains. However, cooperation is a better strategy than selfishness when one has a Grim or amTFT partner. Finally, we see that amTFT maintains more efficient levels of cooperation than Grim Trigger strategies and can almost reach the level of pure pro-social agents.

4 Conclusion

Humans are remarkably adapted to solving bilateral social dilemmas using heuristic strategies such as tit-for-tat. We have focused on how to give artificial agents this capability. Our theoretical results show that simple memory augmented strategies are possible in principle and we have given an algorithm to compute the required components. Finally, we have shown that this approximate Markov tit-for-tat strategy has good properties in simple experiments.

There is much more ground to be explored both in theory and in implementation. In particular, we have used off the shelf algorithms (actor-critic) to compute 'on trajectory' policies and expensive Monte Carlo rollouts to compute the 'off trajectory' components of mTFT. Finding an efficient method for computing high accuracy approximations of Q is an important future direction for this work.

From a more theoretical side, rather than thinking about mTFT in terms of policies, we can think of it in terms of a value function. Under this interpretation mTFT is an artificial implementation of the theory of conditional cooperation (Fischbacher et al., 2001) combined with warm glow altruism (Andreoni, 1990). That is, agents behave according to a policy where they put weight not just on their own utility but also on the utility of their partner. The weight on the partner's utility depends on whether the agent has inferred that their partner is acting in a way that is consistent with pure selfishness or whether their partner seems to be exhibiting warm glow altruism themselves. We note that this is extremely consistent with literature on human cooperation - we feel good when 'good people' do well (and thus strive to help them) and neutral or angry when those that have wronged us get ahead (and often are willing to pay costs to ourselves to reduce their payoffs, Ouss & Peysakhovich (2015)).

There is recent interest in building forward models of an agent's environment (eg. intuitive physics Denil et al. (2016); Lerer et al. (2016)). However, agents and objects are very different - in particular, agents have beliefs, desires and some form of optimization while objects follow simple fixed rules. An important future direction in multi-agent RL is to go beyond simple partner models such as those of mTFT and use inverse reinforcement learning (Abbeel & Ng, 2004; Ng et al., 2000) to learn more complex theories of other agents (eg. Baker et al. (2009); Kleiman-Weiner et al. (2016)).

There is a growing literature on hybrid systems which include both human and artificial agents (Crandall et al., 2017; Shirado & Christakis, 2017). We have constructed an agent which knows how to deal with perfectly selfish agents and so many of our techniques have focused on making the policy robust to approximation error that allows a selfish partner to cheat. However, human preferences are far more complex and include considerations of fairness (Fehr & Gächter, 2000), altruistic cooperation (Peysakhovich et al., 2014) and social norms (Bicchieri, 2005). One strength of the amTFT construction is that it will maintain cooperation not only with purely selfish agents but also with agents that do exhibit some from of social preferences.

However, recent work has shown that incorporating psychologically realistic models of human learning and social interaction can help those interested in designing systems that lead to good outcomes (Erev & Roth, 1998; Fudenberg & Peysakhovich, 2016). An important future direction in the design of artificial cooperative agents is to relax the purely self-interested partner assumption and human psychology into account. Indeed existing work has shown that humans tend to be honest, cooperative and forgiving in social dilemmas and this can be leveraged to maintain good social outcomes (Hauser et al., 2014; Arechar et al., 2016).

There is much modern progress in reinforcement learning in zero sum games. However, this progress has proven to be extremely computationally challenging. There is a strong amount of selection here, modern researchers study precisely the zero-sum games that are hard for humans to solve. Indeed, to be successful in a complex zero-sum game, one must outwit one's partner. However, in the case of cooperative games, especially those where humans are involved we are interested not in complex strategies which probe the weaknesses of our partner but simple, understandable strategies that allow coordination and cooperation to flourish. We hope that our work contributes to this important endeavor.

References

Abbeel, Pieter and Ng, Andrew Y. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, pp. 1. ACM, 2004.

Andreoni, James. Impure altruism and donations to public goods: A theory of warm-glow giving. *The economic journal*, 100(401):464–477, 1990.

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

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

Baker, Chris L, Saxe, Rebecca, and Tenenbaum, Joshua B. Action understanding as inverse planning. *Cognition*, 113(3):329–349, 2009.

Bicchieri, Cristina. *The grammar of society: The nature and dynamics of social norms*. Cambridge University Press, 2005.

Bó, Pedro Dal. Cooperation under the shadow of the future: experimental evidence from infinitely repeated games. *The American Economic Review*, 95(5):1591–1604, 2005.

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

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

Denil, Misha, Agrawal, Pulkit, Kulkarni, Tejas D, Erez, Tom, Battaglia, Peter, and de Freitas, Nando. Learning to perform physics experiments via deep reinforcement learning. *arXiv preprint arXiv:1611.01843*, 2016.

Dutta, Prajit K. A folk theorem for stochastic games. *Journal of Economic Theory*, 66(1):1–32, 1995.

Erev, Ido and Roth, Alvin E. Predicting how people play games: Reinforcement learning in experimental games with unique, mixed strategy equilibria. *American economic review*, pp. 848–881, 1998.

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

Fehr, Ernst and Gächter, Simon. Fairness and retaliation: The economics of reciprocity. *The journal of economic perspectives*, 14(3):159–181, 2000.

Fischbacher, Urs, Gächter, Simon, and Fehr, Ernst. Are people conditionally cooperative? evidence from a public goods experiment. *Economics letters*, 71(3):397–404, 2001.

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

Fudenberg, Drew and Levine, David K. *The theory of learning in games*, volume 2. MIT press, 1998.

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

Fudenberg, Drew and Peysakhovich, Alexander. Recency, records, and recaps: Learning and nonequilibrium behavior in a simple decision problem. *ACM Transactions on Economics and Computation (TEAC)*, 4(4):23, 2016.

Fudenberg, Drew and Tirole, Jean. Game theory, 1991.

Fudenberg, Drew, Rand, David G, and Dreber, Anna. Slow to anger and fast to forgive: Cooperation in an uncertain world. *The American Economic Review*, 102(2):720–749, 2012.

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

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

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

Hoffman, Moshe, Yoeli, Erez, and Nowak, Martin A. Cooperate without looking: Why we care what people think and not just what they do. *Proceedings of the National Academy of Sciences*, 112(6): 1727–1732, 2015.

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

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

Kleiman-Weiner, Max, Ho, Mark K, Austerweil, Joe L, Michael L, Littman, and Tenenbaum, Joshua B. 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.

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

Leibo, Joel Z, Zambaldi, Vinicius, Lanctot, Marc, Marecki, Janusz, and Graepel, Thore. 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.

Lerer, Adam, Gross, Sam, and Fergus, Rob. Learning physical intuition of block towers by example. *arXiv preprint arXiv:1603.01312*, 2016.

Lewis, Mike, Yarats, Denis, Dauphin, Yann N, Parikh, Devi, and Batra, Dhruv. Deal or no deal? end-to-end learning for negotiation dialogues. *arXiv preprint arXiv:1706.05125*, 2017.

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

Macy, Michael W. and Flache, Andreas. Learning dynamics in social dilemmas. *Proceedings of the National Academy of Sciences*, 99(suppl 3):7229–7236, 2002. doi: 10.1073/pnas.092080099. URL `http://www.pnas.org/content/99/suppl_3/7229.abstract`.

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

Ng, Andrew Y, Russell, Stuart J, et al. Algorithms for inverse reinforcement learning. In *Icml*, pp. 663–670, 2000.

Nowak, Martin and Sigmund, Karl. A strategy of win-stay, lose-shift that outperforms tit-for-tat in the prisoner's dilemma game. *Nature*, 364(6432):56, 1993.

Nowak, Martin A. *Evolutionary dynamics*. Harvard University Press, 2006.

Ontanón, Santiago, Synnaeve, Gabriel, Uriarte, Alberto, Richoux, Florian, Churchill, David, and Preuss, Mike. 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.

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

Peysakhovich, Alexander and Rand, David G. Habits of virtue: Creating norms of cooperation and defection in the laboratory. *Management Science*, 62(3):631–647, 2015.

Peysakhovich, Alexander, Nowak, Martin A, and Rand, David G. Humans display a 'cooperative phenotype' that is domain general and temporally stable. *Nature Communications*, 2014.

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

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

Selten, Reinhard and Stoecker, Rolf. End behavior in sequences of finite prisoner's dilemma supergames a learning theory approach. *Journal of Economic Behavior & Organization*, 7(1): 47–70, 1986.

Shapley, Lloyd S. Stochastic games. *Proceedings of the national academy of sciences*, 39(10): 1095–1100, 1953.

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

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

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

Tomasello, Michael. *Why we cooperate*. MIT press, 2009.

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

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

5 Appendix

5.1 Proof of Grim Trigger Theorem

Proof 1 *Let agent i play the 'Grim trigger' strategy π_G^i, which starts by playing π_C^i but switches to π_M^i forever if its partner ever deviates from π_C^{1-i}. This can be implemented with a one-bit memory augmentation: M_i takes the value 1 if there has been a deviation. During the phase where both players are playing π_C, the value function inequality provided implies that there are no profitable one-shot deviations, because any deviation would switch both players to the less profitable π_M equilibrium. During the π_M phase, there are no profitable one-shot deviations since (π_M^1, π_M^2) is an equilibrium. By the Principle of Optimality, a strategy profile is an equilibrium if there are no one-shot deviations, therefore (pi_G^1, π_G^2) is an equilibrium that produces t.*

5.2 Training Details

For the Coin game, there are four actions (up, down, left, right), and S is represented as a $4 \times 5 \times 5$ binary tensor where the first two channels encode the location of the red and blue agent and the other two channels encode the location of the red and blue apples (if any exist). In this version of the game, there is at most one apple on the board at a time. At each time step, if there are no apples on the board, an apple is generated at a random location with a random color, with probability 0.1.

We use a multi-layer convolutional neural network to jointly approximate π and $\hat{Q}$. For this small game, a simpler model could be used, but this model generalizes directly to games with higher-dimensional 2D state spaces (e.g. Atari). For a given board size, the model has $\lceil \log(2) \rceil + 1$ repeated layers, each consisting of a 2D convolution with kernel size 3, followed by batch normalization and ReLU. The first layer has stride 1, while the successive layers each have stride 2, which decreases the width and height from k to $\lceil k/2 \rceil$ while doubling the number of channels. For the 5×5 board, channel sizes are 13, 26, 52, 104. The model has three heads: $\pi, \hat{V}, \hat{Q}$. π and $\hat{V}$. The model is updated episodically, and the update has three terms:

1. MSE loss between $\hat{V}$ and the Monte Carlo reward estimate, i.e. $\left(\hat{V}(S_t) - \sum_k \gamma^k r'_{t+k}\right)^2$.

2. Policy gradient with respect to the *advantage* $\sum_k \gamma^k r'_{t_k} - \hat{V}(S_t)$, as is typically used for advantage actor-critic.

3. Off-policy Bellman iteration update of $\hat{Q}$, i.e.

$$\hat{Q}'(S_t, a_t) = \hat{Q}(S_t, a_t) + \lambda \left(r_t + \gamma \sum_{a'} \hat{Q}(S_{t+1}, a') - \hat{Q}(S_t, a_t) \right)$$

$\hat{V}$ and π are trained on the modified reward structure r' (either selfish or cooperative), while $\hat{Q}$ is trained on the original rewards r. $\hat{V}$ is only used for actor-critic training. Note that the update of $\hat{Q}$ is off-policy, since we desire to learn $\hat{Q}$ under the policy π, *not* the alternative policy implied by $\hat{Q}$.

We train with a learning rate of 0.001, continuation probability $.998$ (i.e. games last on average 500 steps), discount rate 0.98, and a batch size of 32. We train for a total of $40,000$ games.

5.3 Extra Figures

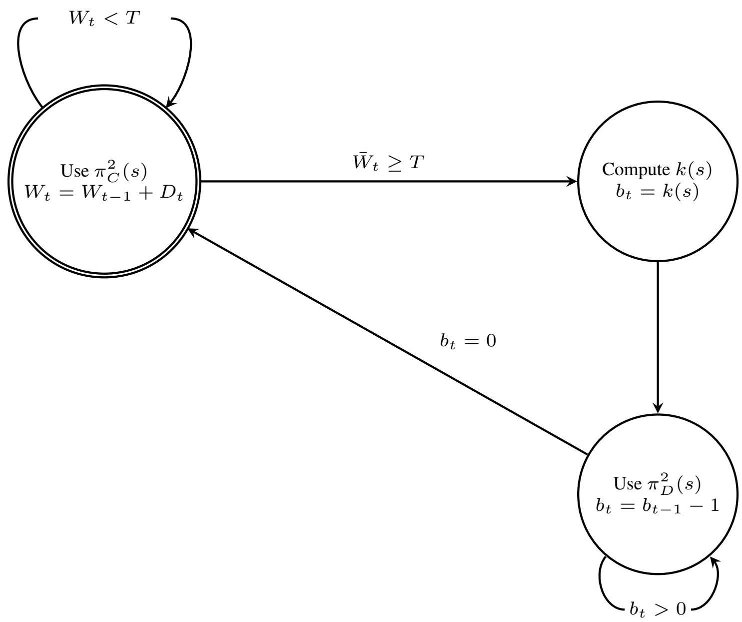

Figure 4: Approximate mTFT represented as a finite state automata.