DM^2: Decentralized Multi-Agent Reinforcement Learning via Distribution Matching

Caroline Wang [1*], Ishan Durugkar [1*], Elad Liebman [2*], Peter Stone [1,3]
[1] The University of Texas at Austin
[2] SparkCognition Research
[3] Sony AI
caroline.l.wang@utexas.edu, ishand@cs.utexas.edu,
eliebman@sparkcognition.com, pstone@cs.utexas.edu

Abstract

Current approaches to multi-agent cooperation rely heavily on centralized mechanisms or explicit communication protocols to ensure convergence. This paper studies the problem of distributed multi-agent learning without resorting to centralized components or explicit communication. It examines the use of distribution matching to facilitate the coordination of independent agents. In the proposed scheme, each agent independently minimizes the distribution mismatch to the corresponding component of a target visitation distribution. The theoretical analysis shows that under certain conditions, each agent minimizing its individual distribution mismatch allows the convergence to the joint policy that generated the target distribution. Further, if the target distribution is from a joint policy that optimizes a cooperative task, the optimal policy for a combination of this task reward and the distribution matching reward is the same joint policy. This insight is used to formulate a practical algorithm (DM^2), in which each individual agent matches a target distribution derived from concurrently sampled trajectories from a joint expert policy. Experimental validation on the StarCraft domain shows that combining (1) a task reward, and (2) a distribution matching reward for expert demonstrations for the same task, allows agents to outperform a naive distributed baseline. Additional experiments probe the conditions under which expert demonstrations need to be sampled to obtain the learning benefits.

1 Introduction

Multi-agent reinforcement learning (MARL) (Littman 1994) is a paradigm for learning agent policies that may interact with each other in cooperative or competitive settings (Silver et al. 2017, 2018; Barrett and Stone 2012; Leibo et al. 2017). Training multiple agents at once is challenging, since an agent updating its own strategy induces a nonstationary environment for other agents, potentially leading to training instabilities, and offsetting any theoretical guarantees single agent RL algorithms confer. To overcome these issues, agent policies can be set up as a single, centralized joint policy, be trained together but then deployed individually (Rashid et al. 2018; Foerster et al. 2018), or be coordinated through some form of communication (Lowe et al. 2017; Jaques et al. 2019; Liu et al. 2021).

Fully distributed training of agent policies remains an open problem in MARL. Distributed, or decentralized, training is desirable particularly in situations where parallelism, robustness, flexibility, or scalability is needed. Such settings include where there are a large number of agents, where agents are faced with changing environments (Marinescu, Dusparic, and Clarke 2017), where agents must perform tasks in varying team configurations over their lifetime (Thrun 1998), or where ensuring privacy is a concern (Leaute and Faltings 2013).

This paper considers the setting of cooperative tasks involving K agents, where the goal is to learn a high-performing joint policy in a fully distributed fashion. To mitigate the limitations imposed by this setting, we propose distribution matching to a target state-action distribution, as a strategy to induce coordination. We assume that this distribution is associated with the execution of some joint policy, such as demonstrations of expert teams (Song et al. 2018), or from high-performing trajectories from the agents' past interactions (Hao et al. 2019). One tempting way to utilize this distribution is to assign each agent the distribution associated with its corresponding expert, and have the agent minimize the distribution mismatch to this target distribution over states and actions.

At first glance, this approach is fraught with complications. Since the target distribution over states and actions is based on the execution of some joint policy, a single agent trying to adjust its policy might not make meaningful progress on its own, given that other agents could change their behaviors at the same time. Second, this distributed approach to distribution matching could suffer from the same destabilization that causes distributed MARL to diverge (Hernandez-Leal, Kartal, and Taylor 2019; Yang and Wang 2020).

This paper shows that despite the above complications, individual distribution matching can be combined with maximization of shared task rewards to learn effectively. In particular, the contributions of the paper are:

- Theoretical analysis showing that distributed distribution matching to the target distributions converges to the joint expert policy that generated the demonstrations.
- The DM^2 algorithm, a practical method combining the distribution matching reward with the task reward. Experimental validation shows that if demonstrations are aligned with the shared objective, DM^2 accelerates learning compared

[*]Equal contribution.

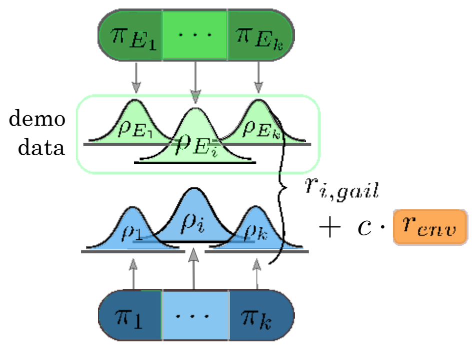


Figure 1: In DM^2, each agent $_i$ independently learns from the sum of a distribution matching reward, $r_{i;gail}$, and a shared task reward, r_{env}. The distribution matching reward is computed by comparing the marginal state visitation distribution of the agent, $_{,i}$ with the state visitation distributions implied by the corresponding expert $_{E_i}$'s demonstrations .

to a decentralized baseline learning with the task reward only.

- Ablations that empirically verify our assumption that the target distribution needs to be induced by demonstrations from coordinated policies, but do not necessarily need to be concurrently sampled.

2 Related Work

This section details related work, separated into work that relates to decentralized learning and that relates to distribution matching.

Cooperation in the Decentralized Setting: Many algorithms for multi-agent cooperation tasks require some degree of information sharing between agents. Centralized training decentralized execution (CTDE) methods use a single centralized critic that aggregates information during training, but is no longer required at execution time (Lowe et al. 2017; Sunehag et al. 2018; Rashid et al. 2018; Foerster et al. 2018; Yu et al. 2022). In practical implementations, agent networks often share parameters during training as well.

Rather than sharing model components, methods may also explicitly communicate information between agents. Agents may be allowed to directly communicate information to each other (Jaques et al. 2019; Li and He 2020; Konan, Seraj, and Gombolay 2022). There might also be a central network that provides coordinating signals to all agents (He et al. 2020; Liu et al. 2021). Knowledge of other agents' policies during training may also be assumed to limit the deviation of the joint policy (Wen et al. 2021).

This work studies the fully decentralized setting without communication or shared model components. To our knowledge, relatively few works consider this setting. Early work analyzed simple cases where two agents with similar but distinct goals could cooperate for mutual benefit under a rationality assumption (Rosenschein and Breese 1989; Genesereth, Ginsberg, and Rosenschein 1986). More recently, in the ALAN system for multi-agent navigation (Godoy et al. 2018), agents learn via a multi-armed bandits method that does not require any communication. Jiang and Lu (2021) study the decentralized multi-agent cooperation in the offline setting—in which each agent can only learn from its own data set of pre-collected behavior without communication—and propose a learning technique that relies on value and transition function error correction.

Distribution Matching in MARL: Ho and Ermon (2016) originally proposed adversarial distribution matching as a way to perform imitation learning in the single agent setting (the GAIL algorithm). Song et al. (2018) extend GAIL to the multi-agent setting in certain respects. Their analysis sets up independent imitation learning as searching for a Nash equilibrium, and assumes that a unique equilibrium exists. Their experiments focus on training the agent policies in the CTDE paradigm, rather than the fully distributed setting. This work instead leverages recent single-agent GAIL convergence theory (Guan, Xu, and Liang 2021) to demonstrate convergence to the joint expert policy, and performs experiments with distributed learning. Wang et al. (2021) study MARL using copula functions to explicitly model the dependence between marginal agent policies for multi-agent imitation learning. Durugkar, Liebman, and Stone (2020) and Radke, Larson, and Brecht (2022) show that balancing individual preferences (such as matching the state-action visitation distribution of some strategies) with the shared task reward can accelerate progress on the shared task. In contrast to these works, the goal of this paper is not to study imitation learning, but rather to study how distribution matching by independent agents can enhance performance in cooperative tasks.

Perhaps most closely related to this work, Hao et al. (2019) use self-imitation learning (SIL) (Oh et al. 2018) to encourage agents to repeat actions that led to high returns in the past. The above approach can be considered as a special case of the setting this paper studies, where the target distribution can be non-stationary, and is generated by the agents themselves. This paper further presents a theoretical analysis, showing that in the case where the target distribution is generated by demonstrations (and is therefore stationary), each agent attempting to minimize mismatch to their individual target distributions leads to convergence to the joint target policy. Due to the non-stationary nature of the target distribution in SIL, similar guarantees cannot be obtained.

3 Background

This section describes the problem setup for MARL, imitation learning, and distribution matching.

Markov games: A Markov game (Littman 1994) or a stochastic game (Gardner and Owen 1983) with K agents is defined as a tuple $hK; S; A; \,_0; T; R; i$, where S is the set of states, and $A \quad A^K$ is the product of the set of actions A available to each agent. The initial state distribution is described by $_0 : S \; ! \;\; (S)$, where $()$ indicates a distribution over the corresponding set. The transitions between states are controlled by the transition distribution $T : S \;\; A_0 \;\; A_1 \; ::: \; A_{K \;\; 1} \;\; ! \;\; (S)$. Each agent i

acts according to a parameterized policy $_i$: S ! (A $_i$), and the joint policy = [;$_1$;] is the vector of the individual agent policies. Occasionally, the policy parameters are omitted for convenience. Note that each agent observes the full state. We use subscript i to refer to all agents except i, i.e., refers to the agent policies, f ; : :$_0$; $_{i\ 1}$; $_{i+1}$; : : : ; $_K$ $_1$ g.

Each agent i is also associated with a reward function R$_i$: S A $_0$: : : A $_{K\ 1}$! R. The agent aims to maximize its expected return E [^{P} $^1_{t=0}$ ^{t}r $_{i;t}$], where r $_{i;t}$ is the reward received by agent i at time step t, and the discount factor 2 [0; 1) specifies how much to discount future rewards. In the cooperative tasks considered by this paper, the task rewards are identical across agents.

In Markov games, the optimal policy of an agent depends on the policies of the other agents. The best response policy is the best policy an agent can adopt, given the other agent's policies $_i$ = argmax E $_{i}$ $_{;i}$ [^{P} $^1_{t=0}$ ^{t}r $_{i;t}$]. If no agent can unilaterally change its policy without reducing its return, then the policies are considered to be in a Nash equilibrium. That is, 8i 2 [0; K 1]; 8^$_i$ = $_i$; E$_{i;i}$ [^{P} $^1_{t=0}$ ^{t}r$_{i;t}$] E $_{\hat{}i;\ i}$ [^{P} $^1_{t=0}$ ^{t}r $_{i;t}$].

The theoretical analysis in Section 4 deals with the above fully observable setting, and assumes a discrete and finite state and action space. However, the experiments are conducted in partially observable MDPs (POMDPs) with continuous states, which can be formalized as Dec-POMDPs in the multi-agent setting (Oliehoek 2012). Dec-POMDPs include two additional elements: the set of observations and each agent's observation function O$_i$: S ! ( ).

Distribution matching and imitation learning: Imitation learning (Bakker and Kuniyoshi 1996; Ross, Gordon, and Bagnell 2011; Schaal 1997) is a problem setting where an agent tries to mimic trajectories f $_0$; $_1$; : : :g where each trajectory = f(s $_0$;a $_0$); (s $_1$;a $_1$); : : :g is demonstrated by an expert policy $_E$. Various methods have been proposed to address the imitation learning problem. Behavioral cloning (Bain and Sammut 1995) applies supervised learning to expert demonstrations to recover the maximum likelihood policy. Inverse reinforcement learning (IRL) (Ng, Russell et al. 2000) recovers a reward function which can then be used to learn the expert policy using reinforcement learning. To do so, IRL($_E$) aims to recover a reward function under which the trajectories demonstrated by $_E$ are optimal.

Ho and Ermon (2016) formulate imitation learning as a distribution matching problem and propose the GAIL algorithm. Let the state-action visitation distribution of a joint policy = h$_1$; : : : $_K$i be:

$$(s; a) := (1\quad) \prod_{i=1}^{K} {}_i(a_i|s) \sum_{t=0}^{1} {}^t P(s_t = s|()):$$

In a multi-agent setting, for agent i,

$$_{i;i}\ (s; a) := (1\quad)_i(a|s) \sum_{t=0}^{1} {}^t p(s_t = s|_i;_i\ )$$

refers to the marginal state-action visitation distribution of agent i's policy $_i$, given the other agents' policies $_i$. In the single agent setting, a policy that minimizes the mismatch of its state-action visitation distribution to the one induced by the expert's trajectories and maximizes its causal entropy H ( ) is a solution to the RL IRL($_E$) problem (Ho and Ermon 2016). That is, distribution matching is a solution to the imitation learning problem.

Guan, Xu, and Liang (2021) showed that in the single-agent case, the GAIL algorithm converges to the expert policy under a variety of policy gradient techniques, including TRPO (Schulman et al. 2015). Let r be a reward function (based on a discriminator) parameterized by , and let ( ) be a convex regularizer. Guan, Xu, and Liang (2021) formu-late the GAIL problem as the following min-max problem:

$$\min \max L(;) \tag{1}$$

$$\text{s.t. } L(;) := V(_E; r) \quad V(; r) \quad ()$$

where V (; r) = E $_{s_0\ 0}$ E [^{P} $^1_{t=0}$ ^{t}r$_{i;t}$] is the expected return from some start state when following policy and using reward function r.

In the multi-agent setting, imitation learning has the added complexity that the expert trajectories are generated by the interaction of multiple expert policies h $_{E_0}$; : : : ; $_{E_K}$ i. Successful imitation in this setting thus involves the coordination of all K agents' policies.

4 Theoretical Analysis

This section provides theoretical grounding for the core proposition of this paper. The target distribution is assumed to be the empirical distribution of demonstrations from a set of "expert" agents in order to ensure that it is achievable by the agents. Under the conditions stated below, the analysis shows that if K agents independently minimize the distribution mis-match to their respective demonstrations in a turn-by-turn fashion, then agent policies will converge to the joint expert policy.

The three conditions are as follows. First, this joint expert policy needs to be coordinated,[1] but does not have to be a Nash equilibrium with respect to any particular task. Second, for every policy considered, there is a minimal probability of visiting each state. Third, each agent learns via a single-agent imitation learning algorithm such that it improves its distribution matching reward at each step.

Next, we establish that if the agents are learning to maximize the mixture of an extrinsic task reward and a distribution matching reward, then the agent policies will converge to a Nash equilibrium with respect to the joint reward.

Convergence of Independent GAIL Learners

This analysis considers the setting where each agent i performs independent learning updates according to the GAIL algorithm, to match the visitation distribution of the ith expert. It proposes a condition on an individual agent's GAIL objective improvement. If this condition is satisfied, it shows that a lower bound on a joint distribution matching objective is improved. Further, the lower bound objective converges, demonstrating the convergence of independent GAIL.

[1]Condition is made concrete in Section 5.

Let the parameterized (discriminator) reward of agent i be $r_i : S \; A \; !\; R$, for 2_i . At each agent's update, all the other agent policies are held fixed, and the corresponding discriminator has converged to r^{opt}_i, where $_i^{opt}$ 2 argmax $L(_i; j_i)$. Guan, Xu, and Liang (2021) showed that the learning process of a single agent repeatedly updating converges to $^{opt}_i$. The update scheme we consider for theoretical purposes is specified in Algorithm 2, located in Appendix A. [2]

Define the per-agent GAIL loss as follows:

$$L(_i; {}_i j_i) := V({}_{E_i}; r^{opt} j_{E_i}) \qquad V(_i; r^{opt} j_i) \qquad (_i)$$

$$\text{s.t. } V(_i; r^{opt} j_i) := \frac{1}{1} \underset{s \; _i}{E} \left[r^{opt}(s; a_i) \right]_{;a}$$

where $(s) = (1 \quad) \sum_{t=0}^{1} {}^t P(s_t = sj)$ is the discounted state visitation distribution.

Consider the random variable that is the indicator function $1_{a_j = a^E_j}(s)$ for the event that at state s, agent j would take an action that matched expert j's action. Note that the expectation of this indicator is the probability of matching the expert's action[3]. Define the joint action-matching objective as the probability that agent actions match their corresponding experts (plus a constant), weighted by the probability of visiting states:

$$J() = \sum_{s2S} (s) \left[(K \quad 1) + \underset{a}{E} [1^T_i \, {}_{a_i = a_i}(s)] \right] : \qquad (2)$$

where $1^T_i \, {}_{a_i = a^E_i}(s)$ indicates the event that all agents take actions that match their corresponding experts. Maximizing J() precisely corresponds to solving the multi-agent imitation learning problem because the joint expert policy $_E$ is the unique maximizer of J() (Lemma 5, Appendix A).

Theorem 1 (action-matching objective). The joint action-matching objective J() is lower bounded by the following sum over individual action-matching rewards $1_{a_i = a^E_i(s)}$:

$$L() := \sum_{s2S} (s) \left[\sum_{i=1}^{K} \underset{a_i \; i}{E} [1_{a_i = a^E_i}(s)] \right] : \qquad (3)$$

When an agent updates its policy to optimize its component of L(), the state visitation distribution might change such that the expected action rewards for other agents decrease. The next corollary introduces a lower bound on L() that is independent of the state visitation distribution .

Corollary 1 (lower bound). Let be the minimum probability of visiting any state. For all , L() is lower bounded by L():

$$L() > \sum_{s2S} \sum_{i=1}^{K} \underset{a_i \; i}{E} [1_{a_i = a_i}(s)] =: L(): \qquad (4)$$

[2]A technical appendix is included in the arXiv version of this paper, https://arxiv.org/abs/2206.00233.

[3]For the purpose of exposition, assume that the expert policy is deterministic. The theory in this section can be extended to the case where $_E$ is stochastic by comparing the distributions over actions.

With the lower bound, L we make the following assumption to relate our action-matching reward to the GAIL discriminator reward that is improved by the GAIL algorithm.

Assumption 1 (action-matching reward). For all agents i and all states s, an increase in the expected converged GAIL discriminator reward implies an increase to the expected action-matching reward function:

$$\underset{a_i \; i}{E}{}_{t+1} [r_i(s; a_i)] > \underset{i \; a_i}{E}{}_t [r_i(s; a_i)]$$

$$=) \underset{a_i \; i}{E}{}_{t+1} [1_{a_i = a^E_i}(s)] > \underset{a_i \; i}{E}{}_t [1_{a_i = a^E_i}(s)]:$$

If the reward $r_i(s; a) = \quad \log D(s; a)$, as it is in GAIL, then the assumption above is valid (see Appendix A).

Assumption 1 ensures each agent updating its policy leads to improvement in L (). This L () is a lower bound on the actual objective of interest J()—by Theorem 1 and Corollary 1. Further, L () has a unique global maximizer, which is $= \; _E$ (Lemma 7). Thus, while the action reward for the other agents r_j might decrease in the short term, the joint action matching objective across all agents will increase as the learning process continues. Since J() is bounded from above (Lemma 4), this process of improving the lower bound will converge to the optimal policy for this objective—the joint expert policy.

Theorem 2 (convergence). Each agent maximizing its individual return over the individual action rewards r_i will converge to the joint expert policy $_E$.

Multi-agent Learning with Mixed Task and Imitation Reward

Lemma 5 in Appendix A shows that the joint expert policy uniquely maximizes the joint imitation learning objective. Let $^{opt;E_i}_i$ be the optimal discriminator parameters for the ith expert, $_{E_i}$. From a game theoretic perspective, this lemma implies that these expert policies are a Nash equilibrium for the imitating agents with respect to $r^{opt}_{;iE_i}$.

Next, note that in imitation learning, it is typically not necessary for the agents to know what the demonstration actor's task reward is. However, suppose that the agents have access to both demonstrations from policies optimal at task T, and the corresponding reward function R_T.

Let $R_{I;i} = r^{opt}_{;iE_i}$, and let the expert policies maximize R_T. The expert policies that maximize R_T are in a Nash equilibrium with respect to R_T. Theorem 3 states that if the agents are trained to maximize a reward function that is a linear combination of the task reward R_T and $R_{I;i}$, then the converged agent policies are also in a Nash equilibrium with respect to R_T.

Theorem 3. Let R_T be the reward function used to train the expert policies $_E$, and let the expert policies have converged with respect to R_T (i.e., they are in a Nash equilibrium with respect to reward R_T). Then $_E$ are a Nash equilibrium for reward functions of the form, $R_T + R_{I;i}$, for any ; > 0.

Theorem 2 does not require that the demonstrations originate from optimal policies for some task. However, Theorem 3 implies that if the demonstrations do maximize the reward of a desired task, then the task reward and distribution matching reward can be combined to optimize the same task. The proposed algorithm, DM^2, takes this approach.

5 Methods

This section discusses practical considerations of fully distributed multi-agent distribution matching, and proposes DM^2, an algorithm whose performance is analyzed in Section 6.

Generating Expert Demonstrations

Section 4 shows that agents individually following demonstrations from an existing joint policy can converge to said joint policy without centralized training or communication. In practice, these demonstrations should imply an achievable joint expert policy.

For illustration, consider a four tile gridworld, where only one agent is allowed on a tile at a time. Let one of the tiles be labelled, "A". Suppose there are two agents, and each agent i is provided with a separate target state-action distribution, consisting of agent i occupying a tile "A", and the other agent occupying one of the three remaining tiles. If both agents simultaneously attempt to match their provided target distributions, then both agents will attempt to occupy tile "A". With these demonstrations, it is impossible for both agents to fully match their desired distributions.

The example above shows that for each agent to completely match its desired distribution, the state-action distributions for all agents must be compatible in some way. This notion of compatibility is defined below.

Definition 1 (Compatible demonstrations). State-action visitation distributions $_{i;\hat{}_i}$ from a collection of K policies f g $_i{}^K_{i=1}$ (where $\hat{}_i$ are the other agent policies executed with $_i$ to obtain the state-action visitation distribution $_{i;\hat{}_i}$) are compatible if for all i, s 2 S; a 2 A, there exists a joint policy $^0 = h^0_1; : : : ;^0_K i$ with the joint state-action visitation distribution (s; a) (Equation 3) such that the marginal state-action visitation distribution for agent i is:

$$_{i;_i^0}(s; a) := (1 \quad)_i(a|s)\ c \quad X^1 \quad {}^t P(s_t = s|^0;^0_i)\ {}_{t=0}$$
$$= {}_{i;\hat{}_i}(s; a):$$

Observe that K expert policies that are trained in the same environment to perform a task induce compatible individual state-action visitation distributions, providing a practical method to obtain compatible demonstrations.

Practical Multi-Agent Distribution Matching

DM^2 is inspired by the theoretical analysis in Section 4, and balances the individual objective of distribution matching with the shared task. To do so, the agents are provided a mixed reward: part cost function for minimizing individual distribution mismatch, part environment reward. This approach has been shown to be effective in balancing individual preferences with shared objectives in MARL (Durugkar, Liebman, and Stone 2020; Cui et al. 2021). The individual agent policies are learned by independently updating each agent's policy using an on-policy RL algorithm of choice. The experiments here use PPO (Schulman et al. 2017).

Algorithm 1: DM^2 (Decentralized MARL via distribution matching)

```
Input: Number of agents K, expert demonstrations
       D_0; : : : ; D_K, environment env, number of
       epochs N, number of time-steps per epoch M,
       reward mixture coefficient c
1  for k = 0; : : : ; K   1 do
2      Initialize discriminator parameters  ;k 3
       Initialize policy parameters k;
4  end
5  for n = 0; 1; : : : ; N   1 do
6      Gather m = 1; : : : ; M steps of data
       ($s^m$; $a^m$; $r^m_{env}$) from env;
7      for k = 0; : : : ; K   1 do
8          Sample M states from demonstration $D_k$;
9          Update discriminator D ;k
10         Compute GAIL reward
           $r^m_{k;GAIL}$ =   log $D_{k;}(s^m)$ for all M
           demonstration states;
11         Set agent reward $r^m_{k;mix}$ = $r^m_{env}$ + $r^m_{k;GAIL}$  c;
12         Update agent policy $^k$ with data
           ($s_m$; $a_m$; $r^m_{k;mix}$) for m = 1; : : : ; M;
13     end
14 end
Output: K  agent policies
```

In the experiments, the demonstrations used as targets for the distribution matching are compatible, state-only trajectories—i.e., originating from policies trained jointly on the task of interest. The use of state-only demonstrations enables learning purely from observations of other agents (e.g. online videos), and is supported by research in the sub-area of imitation from observation alone (Torabi, Warnell, and Stone 2019). Our experiments also validate the effective-ness of the approach in this setting. In Section 6, we show that the demonstrator policies may possess intermediate compe-tency in the task at hand, and that the demonstrations do not need to be jointly sampled for all the agents. The proposed learning scheme for training individual agents is summarized by Algorithm 1 and Figure 1.

6 Experimental Evaluation

This section presents two main experiments. The first experiment evaluates whether DM^2 may improve coordination–and therefore efficiency of learning–over a decentralized MARL baseline. A comparison against CTDE algorithms is also performed. The second experiment is an ablation study on the demonstrations that are provided to DM^2. These ablations seek to answer the question whether the faster, improved learning above is due to coordination between experts, or due to individual experts being competent.

Additional experiments in Appendix B evaluate the effect of the demonstration quality on learning, ana-

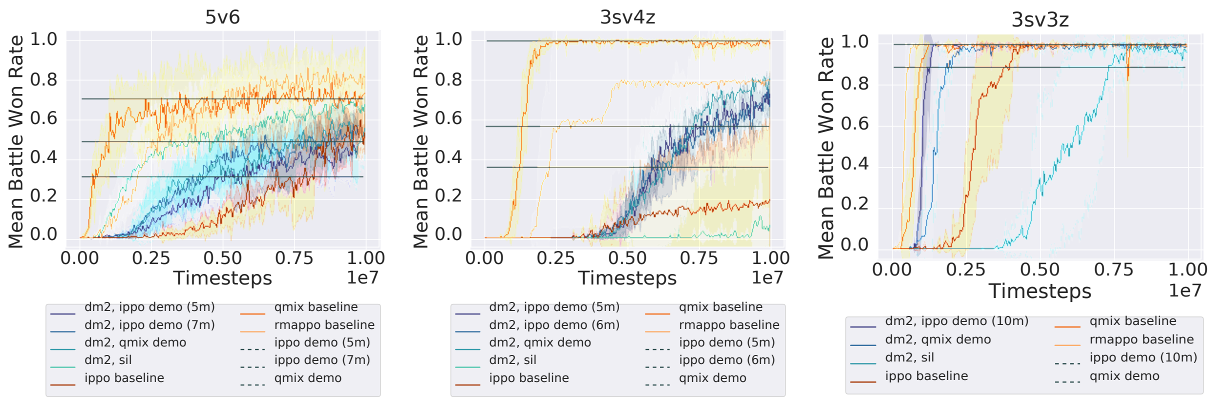


Figure 2: Learning curves of DM^2 (ours), compared to IPPO, RMAPPO, and QMIX baselines on the 5v6, 3sv4z, and 3s3z tasks. DM^2 is trained with demonstrations from IPPO and QMIX experts. IPPO demonstrations are sampled from varying points in the training process of the IPPO experts, and therefore vary in quality. A variation with SIL is also shown.

lyze the usage of demonstrations for behavioral cloning (Bain and Sammut 1995) instead of distribution matching, and examine the impact of using only distribution matching GAIL rewards for learning instead of mixing them with the task rewards. The code is provided at https://github.com/carolinewang01/dm2.

Environments: Experiments were conducted on the StarCraft Multi-Agent Challenge domain (Samvelyan et al. 2019). It features cooperative tasks where a team of controllable allied agents must defeat a team of enemy agents. The enemy agents are controlled by a fixed AI. The battle is won and the episode terminates if the allies can defeat all enemy agents. The allies each receive a team reward every time an enemy agent is killed, and when the battle is won. StarCraft is a partially observable domain, where an allied agent can observe features about itself, as well as allies and enemies within a fixed radius. The specific StarCraft tasks used here (with two additional tasks in Appendix B) are:

- 5v6: 5 Marines (allies) and 6 Marines (enemies)
- 3sv4z: 3 Stalkers (allies) and 4 Zealots (enemies)
- 3sv3z: 3 Stalkers (allies) and 3 Zealots (enemies)

Baselines: DM^2 is compared against a naive decentralized MARL algorithm, independent PPO (Schulman et al. 2017) (IPPO), where individual PPO agents directly receive the team environment reward. Although agents trained under the IPPO scheme cannot share information and see only local observations, prior work has shown that IPPO can be surprisingly competitive with CTDE methods (Yu et al. 2022). We also compare against two widely used CTDE methods, QMIX (Rashid et al. 2018) and RMAPPO (Yu et al. 2022). These CTDE methods have the advantage of a shared critic network that receives the global state during training. Thus, their performance is expected to be better than that of decentralized methods with no communication.

Setup: DM^2 uses the same IPPO implementation as the baseline, with the addition of a GAIL discriminator for each independent agent i to generate an imitation reward signal, $r_{i;\text{GAIL}}$. The scaled GAIL reward is added to the environment reward r_{env}, with scaling coefficient c 2 R: $r_{i;mix} = r_{env} + r_{i;\text{GAIL}}$ c. Learning curves of all algorithms are the mean of 5 runs executed with independent random seeds, where each run is evaluated for 32 test episodes at regular intervals during training. The shaded regions on the plots show the standard error. The evaluation metric is the mean rate of battles won against enemy teams during test episodes.

The data for the GAIL discriminator consists of 1000 joint state-only trajectories (no actions). The data is sampled from checkpoints during training runs of baseline IPPO with the environment reward, and QMIX with the environment reward. In runs of DM^2, each agent imitates the marginal observations of the corresponding agent from the dataset (i.e., agent i will imitate agent i's observations from the dataset) [4]. For each task, demonstrations are sampled from IPPO and QMIX-trained joint expert policies, executed stochastically for IPPO and with an -greedy sampling for QMIX. The win rates achieved by the demonstration policies are plotted as horizontal lines on the graphs. Additionally, SIL(Hao et al. 2019) can be seen as a variation of DM^2, with the target demonstrations being the agent's most successful prior trajectories. Experimental details such as hyperparameters are specified in Appendix C.

Main Results

Figure 2 shows that in all three tasks, DM^2 significantly improves learning speed over IPPO (the decentralized baseline). QMIX and RMAPPO (the CTDE baselines) learn faster than DM^2 and IPPO on both tasks, illustrating the challenging nature of the decentralized cooperation problem. However, on 5v6 and 3sv3z, all methods converge to a similar win rate towards the end of training. For the demonstrations from IPPO experts, DM^2 surpasses the win rate of the demonstrations. Despite the significant variance in win rates among the demonstrations for each task, DM^2 performs similarly. Similar robustness to demonstration quality is seen even with

[4] The allied agent teams in our experiments have the same state/action spaces. Thus, the mapping of agents to demonstration trajectories does not matter, as long as it is fixed

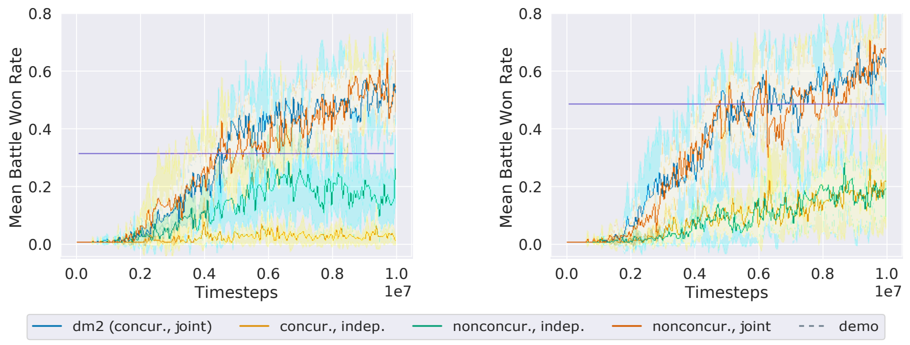


Figure 3: Ablations for IPPO trained with r_{mix} on the 5v6 task. The case where the demonstrations are concurrently sampled from co-trained (joint) expert policies corresponds to DM^2. Left: Experiments performed with lower quality IPPO demonstration (5m). Right: Experiments performed with higher quality IPPO demonstration (7m).

10 different demonstration qualities (Figure 7 in Appendix B). The relative invariance to demonstration quality suggests that the demonstrations provide a useful coordinating signal, enabling agents to discover higher-return behaviors than those portrayed in the demonstrations.

On the other hand, using trajectories from earlier in training as demonstrations (SIL) shows inconsistent performance. Its performance is comparable to RMAPPO in 5v6, but it learns more slowly in the other domains. This may occur because SIL requires examples of successful coordination by the agents, which may be rare in certain tasks. Using past trajectories as demonstrations also leads to a nonstationary target distribution for imitation, potentially negatively impacting the learning procedure.

Ablation Study

This section presents an ablation study on the demonstrations. It investigates whether the coordination of expert agents or their individual competency is more important to the success of DM^2 in the main experiment. This comparison is done by considering demonstrations that vary in two dimensions: whether they were sampled from expert teams that form a joint policy (co-trained), or whether they were sampled simultaneously (concurrently sampled).

The first dimension tests the requirement that agents are trained to coordinate with each other, while the second tests whether agents must act together when generating demonstrations. Concurrently sampled demonstrations of agents that were not co-trained, gives us examples of individually competent agents acting in the multi-agent setting.

The experiments apply DM^2 to four possible demonstration styles that vary in the aforementioned two dimensions. A detailed explanation of how these four demonstration styles were constructed is provided in Appendix C. The study is performed on the 5v6 task, with the same hyperparameters used in the experiments of the previous section.

Figure 3 shows the learning curves of the four combinations. The axis that appears to make the greatest difference in learning is whether the demonstrations originate from expert policies that were co-trained, and were thus coordinated. Whether the agent demonstrations were concurrently sampled does not appear to significantly impact learning. Similar trends are observed when DM^2 is trained with the lower quality demonstration (Figure 3, left).

7 Discussion and Future Work

This paper studies distributed MARL for cooperative tasks without communication or explicit coordination mechanisms. Fully distributed MARL is challenging, since simultaneous updates to different agents' policies can cause them to diverge. The benefits of distributed MARL are abundant. Decentralized training could make agents more robust to the presence of agents they were not trained with (e.g. humans). Decentralized training could also enable coordination while preserving the privacy of each agent.

The theoretical analysis of this paper shows that individual agents updating their policies turn-by-turn to reduce their distribution mismatch to corresponding expert distributions improves a lower bound to the joint action-matching objective against the joint expert policy. Fully maximizing the lower bound corresponds to recovering the joint expert policy. The experiments verify that mixing the task reward with the distribution matching reward accelerates cooperative task learning, compared to learning without the distribution matching objective. The ablation experiments show that expert demonstrations should be from policies that were trained together, but not necessarily concurrently sampled.

While this work is a meaningful step towards fully distributed multi-agent learning via distribution matching, some open questions remain. Future work could consider whether demonstrations sampled from expert policies with other properties, such as those trained with reward signals corresponding to different tasks, could be beneficial for distributed learning. The method proposed in this paper could also be leveraged to combine human demonstrations with a task reward for applications of MARL ranging from expert decision making (similar to that done by Gombolay et al. (2018) in the context of medical recommendation) or in the context of complex multi-agent traffic navigation (Behbahani et al. 2019). Another potential path forward would be considering human in the loop settings such as the TAMER architecture (Knox and Stone 2009), but in a fully distributed multi-agent setting.

Acknowledgements

This work has taken place in the Learning Agents Research Group (LARG) at the Artificial Intelligence Laboratory, The University of Texas at Austin, and at SparkCognition Applied Research. LARG research is supported in part by the National Science Foundation (CPS-1739964, IIS-1724157, FAIN-2019844), the Office of Naval Research (N00014-18-2243), Army Research Office (W911NF-19-2-0333), DARPA, General Motors, Bosch, and Good Systems, a research grand challenge at the University of Texas at Austin. The views and conclusions contained in this document are those of the authors alone. Peter Stone serves as the Executive Director of Sony AI America and receives financial compensation for this work. The terms of this arrangement have been reviewed and approved by the University of Texas at Austin in accordance with its policy on objectivity in research.

References

Bain, M.; and Sammut, C. 1995. A Framework for Behavioural Cloning. In Machine Intelligence 15, 103–129.

Bakker, P.; and Kuniyoshi, Y. 1996. Robot see, robot do: An overview of robot imitation. In AISB96 Workshop on Learning in Robots and Animals, 3–11.

Barrett, S.; and Stone, P. 2012. An analysis framework for ad hoc teamwork tasks. In Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1, 357–364.

Behbahani, F.; Shiarlis, K.; Chen, X.; Kurin, V.; Kasewa, S.; Stirbu, C.; Gomes, J.; Paul, S.; Oliehoek, F. A.; Messias, J.; et al. 2019. Learning from demonstration in the wild. In 2019 International Conference on Robotics and Automation (ICRA), 775–781. IEEE.

Cui, J.; Macke, W.; Yedidsion, H.; Goyal, A.; Urielli, D.; and Stone, P. 2021. Scalable Multiagent Driving Policies For Reducing Traffic Congestion. In Proceedings of the 20th International Conference on Autonomous Agents and Multi Agent Systems (AAMAS).

Durugkar, I.; Liebman, E.; and Stone, P. 2020. Balancing individual preferences and shared objectives in multiagent reinforcement learning. In Proceedings of the 29th International Joint Conference on Artificial Intelligence (IJCAI).

Foerster, J.; Farquhar, G.; Afouras, T.; Nardelli, N.; and Whiteson, S. 2018. Counterfactual Multi-Agent Policy Gradients. Proceedings of the AAAI Conference on Artificial Intelligence, 32.

Gardner, R.; and Owen, G. 1983. Game Theory (2nd Ed.). Journal of the American Statistical Association, 78: 502.

Genesereth, M. R.; Ginsberg, M. L.; and Rosenschein, J. S. 1986. Cooperation without Communication. In Proceedings of the AAAI Conference on Artificial Intelligence.

Godoy, J.; Chen, T.; Guy, S. J.; Karamouzas, I.; and Gini, M. L. 2018. ALAN: adaptive learning for multi-agent navigation. Autonomous Robots, 42: 1543–1562.

Gombolay, M.; Yang, X. J.; Hayes, B.; Seo, N.; Liu, Z.; Wadhwania, S.; Yu, T.; Shah, N.; Golen, T.; and Shah, J. 2018. Robotic assistance in the coordination of patient care. The International Journal of Robotics Research, 37(10): 1300–1316.

Guan, Z.; Xu, T.; and Liang, Y. 2021. When Will Generative Adversarial Imitation Learning Algorithms Attain Global Convergence. In AISTATS.

Hao, X.; Wang, W.; Hao, J.; and Yang, Y. 2019. Independent Generative Adversarial Self-Imitation Learning in Cooperative Multiagent Systems. In Proceedings of the 18th International Conference on Autonomous Agents and Multi Agent Systems, 1315–1323.

He, X.; An, B.; Li, Y.; Chen, H.; Wang, R.; Wang, X.; Yu, R.; Li, X.; and Wang, Z. 2020. Learning to Collaborate in Multi-Module Recommendation via Multi-Agent Reinforcement Learning without Communication. Fourteenth ACM Conference on Recommender Systems.

Hernandez-Leal, P.; Kartal, B.; and Taylor, M. E. 2019. A survey and critique of multiagent deep reinforcement learning. Autonomous Agents and Multi-Agent Systems, 33: 750 – 797.

Ho, J.; and Ermon, S. 2016. Generative adversarial imitation learning. Advances in neural information processing systems, 29: 4565–4573.

Jaques, N.; Lazaridou, A.; Hughes, E.; Gulcehre, C.; Ortega, P.; Strouse, D.; Leibo, J. Z.; and De Freitas, N. 2019. Social Influence as Intrinsic Motivation for Multi-Agent Deep Reinforcement Learning. In Chaudhuri, K.; and Salakhutdinov, R., eds., Proceedings of the 36th International Conference on Machine Learning, volume 97 of Proceedings of Machine Learning Research. PMLR.

Jiang, J.; and Lu, Z. 2021. Offline Decentralized Multi-Agent Reinforcement Learning. ArXiv, abs/2108.01832.

Knox, W. B.; and Stone, P. 2009. Interactively shaping agents via human reinforcement: The TAMER framework. In Proceedings of the fifth international conference on Knowledge capture, 9–16.

Konan, S.; Seraj, E.; and Gombolay, M. 2022. Iterated Reasoning with Mutual Information in Cooperative and Byzantine Decentralized Teaming. In ICLR.

Léauté, T.; and Faltings, B. 2013. Protecting privacy through distributed computation in multi-agent decision making. Journal of Artificial Intelligence Research, 47: 649–695.

Leibo, J. Z.; Zambaldi, V.; Lanctot, M.; Marecki, J.; and Graepel, T. 2017. Multi-agent Reinforcement Learning in Sequential Social Dilemmas. In Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, 464–473.

Li, H.; and He, H. 2020. Multi-Agent Trust Region Policy Optimization. CoRR, abs/2010.07916.

Littman, M. L. 1994. Markov games as a framework for multi-agent reinforcement learning. In Machine learning proceedings 1994, 157–163. Elsevier.

Liu, B.; Liu, Q.; Stone, P.; Garg, A.; Zhu, Y.; and Anandkumar, A. 2021. Coach-Player Multi-Agent Reinforcement Learning for Dynamic Team Composition. In International Conference on Machine Learning.

Lowe, R.; Wu, Y.; Tamar, A.; Harb, J.; Abbeel, P.; and Mordatch, I. 2017. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In NeurIPS.

Marinescu, A.; Dusparic, I.; and Clarke, S. 2017. Prediction-based multi-agent reinforcement learning in inherently non-stationary environments. ACM Transactions on Autonomous and Adaptive Systems (TAAS), 12(2): 1–23.

Ng, A. Y.; Russell, S. J.; et al. 2000. Algorithms for inverse reinforcement learning. In Icml, volume 1, 663–670.

Oh, J.; Guo, Y.; Singh, S.; and Lee, H. 2018. Self-imitation learning. In International Conference on Machine Learning, 3878–3887. PMLR.

Oliehoek, F. A. 2012. Decentralized POMDPs, 471–503. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN 978-3-642-27645-3.

Radke, D.; Larson, K.; and Brecht, T. B. 2022. Exploring the Benefits of Teams in Multiagent Learning. ArXiv, abs/2205.02328.

Rashid, T.; Samvelyan, M.; Schroeder, C.; Farquhar, G.; Foerster, J.; and Whiteson, S. 2018. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In Proceedings of the 35th International Conference on Machine Learning, volume 80 of Proceedings of Machine Learning Research. PMLR.

Rosenschein, J. S.; and Breese, J. S. 1989. Communication-Free Interactions among Rational Agents: A Probabilistic Approach. In Distributed Artificial Intelligence.

Ross, S.; Gordon, G.; and Bagnell, D. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In Proceedings of the fourteenth international conference on artificial intelligence and statistics, 627–635.

Samvelyan, M.; Rashid, T.; de Witt, C. S.; Farquhar, G.; Nardelli, N.; Rudner, T. G. J.; Hung, C.-M.; Torr, P. H. S.; Foerster, J.; and Whiteson, S. 2019. The StarCraft Multi-Agent Challenge. CoRR, abs/1902.04043.

Schaal, S. 1997. Learning from demonstration. In Advances in neural information processing systems, 1040–1046.

Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; and Moritz, P. 2015. Trust region policy optimization. In International conference on machine learning, 1889–1897. PMLR.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347.

Silver, D.; Hubert, T.; Schrittwieser, J.; Antonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; Graepel, T.; et al. 2018. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. Science, 362(6419): 1140–1144.

Silver, D.; Schrittwieser, J.; Simonyan, K.; Antonoglou, I.; Huang, A.; Guez, A.; Hubert, T.; Baker, L.; Lai, M.; Bolton, A.; et al. 2017. Mastering the game of go without human knowledge. Nature, 550(7676): 354–359.

Song, J.; Ren, H.; Sadigh, D.; and Ermon, S. 2018. Multi-Agent Generative Adversarial Imitation Learning. In Advances in Neural Information Processing Systems, volume 31.

Sunehag, P.; Lever, G.; Gruslys, A.; Czarnecki, W. M.; Zambaldi, V.; Jaderberg, M.; Lanctot, M.; Sonnerat, N.; Leibo, J. Z.; Tuyls, K.; and Graepel, T. 2018. Value-Decomposition Networks For Cooperative Multi-Agent Learning Based On Team Reward. In Proceedings of the 17th International Conference on Autonomous Agents and Multi Agent Systems, AAMAS '18.

Thrun, S. 1998. Lifelong learning algorithms. In Learning to learn, 181–209. Springer.

Torabi, F.; Warnell, G.; and Stone, P. 2019. Generative Adversarial Imitation from Observation. arXiv:1807.06158 [cs, stat]. ArXiv: 1807.06158.

Wang, H.; Yu, L.; Cao, Z.; and Ermon, S. 2021. Multi-agent Imitation Learning with Copulas. In Machine Learning and Knowledge Discovery in Databases. Research Track, 139–156.

Wen, Y.; Chen, H.; Yang, Y.; Tian, Z.; Li, M.; Chen, X.; and Wang, J. 2021. A Game Theoretic Approach to Multi-Agent Trust Region Optimization.

Yang, Y.; and Wang, J. 2020. An overview of multi-agent reinforcement learning from game theoretical perspective. ArXiv, abs/2011.00583.

Yu, C.; Velu, A.; Vinitsky, E.; Wang, Y.; Bayen, A.; and Wu, Y. 2022. The Surprising Effectiveness of MAPPO in Cooperative Multi-Agent Games. In Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks.

Appendix

A Convergence Proof Details

In Section 4, we lay out some of the conditions for our theoretical analysis. One of these conditions, that for every policy considered there is a minimal probability of visiting each state, is formalized below.

Condition 1. Let $0 < \ < 1$, and let be the state visitation distribution induced by any joint policy during training. For all agents i and for all s, suppose that (s) .

Algorithm 2: Distributed MARL with distribution matching

```
   Input: Number of agents $K$, expert demonstrations $D_0; :::; D_K$, environment env, number of time-steps per epoch $M$
1  for $k = 0; :::; K \ 1$ do
2      Initialize discriminator parameters $_k$;
3      Initialize policy parameters $_k$;
4  end
5  while any($_k$) not converged do
6      for $k = 0; :::; K \ 1$ do
7          Gather $m = 1; :::; M$ steps of data $(s^m; a^m; r^m_{env})$ from env;
8          Update agent discriminator $r_k$ to maximize Equation 1 until convergence to $r^{opt}_k$;
9          Update agent policy $_k$ using TRPO to minimize Equation 1
10     end
11 end
   Output: $K$ agent policies
```

Recall that the joint action-matching objective is defined over the expected state visitation as the probability that all agent actions match their corresponding experts (plus a constant):

$$J() = \sum_{s \in S} (s) \left[(K \ 1) + \mathbb{E}_a [1_{\wedge_i a_i = a_i^E}(s)] \right] :$$

where $1_{\wedge_i a_i = a_i^E}(s)$ indicates the event that all the agents took actions that matched their corresponding experts.

We first prove some properties of $J()$.

Lemma 4. The objective $J()$ is bounded by

$$(K \ 1) \ J() \ K :$$

Proof. As shorthand, define $f() = \sum_{s \in S} (s) \, \mathbb{E}_a [1_{\wedge_i a_i = a_i^E}(s)]$. Then,

$$J() = (K \ 1) + f():$$

First, note that $f() \ 0$ because it is a weighted sum of expectations over indicator functions, where all weights are non-negative, and it is precisely 0 if for all states, the joint agent policy does not match the correct expert actions. Thus, $J()$ is lower bounded by $(K \ 1)$. For the upper bound, notice that the outer summation is equivalent to the expectation under state visitation distribution . Inside, each expectation over the indicators can be at most 1, implying that $f()$ is at most 1. Thus, $J() \ K$. □

Lemma 5. Suppose the joint expert policy $_E$ is deterministic. Then $_E$ is the unique maximizer of $J()$.

Proof. (Maximization) Since $_E$ is deterministic, by definition, each term

$$\mathbb{E}_{a \ _E} [1_{\wedge_i a_i = a_i^E}(s)] = 1:$$

Thus, $J(_E) = K$, which means that $_E$ achieves the upper bound of J.

(Uniqueness) Suppose there exists another policy $= \ _E$ that also achieves the upper bound, i.e. $J() = K$. Let $a_i^E := {}_{E_i}(s)$. Then there must be an agent i such that with positive probability, $a_i \ _i(s)$ such that $a_i = a_i^E$. Then it is immediate that at state s, $\mathbb{E}_a[1_{\wedge_i a_i = a_i^E}(s)] < 1$. Combined with the non-zero probability of visiting every state (Condition 1), this inequality then implies $J(_E) < K$. By contradiction, $_E$ is the unique optimizer for J. □

Next, we establish that individual agents performing GAIL updates maximizes a lower bound on $J()$. We leverage the single-agent GAIL convergence result by (Guan, Xu, and Liang 2021) to show this result.

Lemma 6. Let t be the time step at which agent i's policy is updated. For all agents j, denote the optimal discriminators of Equation 1 as r^{opt}_j. Suppose agent i updates its policy parameters from $_i^t$ to $_i^{t+1}$ such that $L(^{t+1}_i; ^{opt}j^t_i) < L(^t_i; ^{opt}j^t_i)$. This decrease in loss is equivalent to increasing the agent i's expected discriminator reward.

Proof. First, note that updating a single agent policy while keeping all discriminators fixed does not alter the expert value term $V(_{E_k}; r_{opt})$ or the regularizer term $(^{opt}_k)$ in the loss definition L. Thus, the condition that agent i's loss has decreased is equivalent to the value of agent i increasing:

$$V(_i^{t+1}; ^{opt}_i j^t_i) > V(^t_i; ^{opt}_i j^t_i): \tag{5}$$

For convenience of notation, agent i's policy at time t will be written as ^t_i, and the ith discriminator r^{opt}_i implicitly indicated by the action subscript, $r(s; a_i)$. Similarly, we will write the state visitation distribution induced by the policies $^{t+1}; ^t_i$ by $^{t+1}_i$, and the distribution induced by $^t; ^t_i$ as ^{t}.

Rewriting Equation 5 in terms of visitation distributions:

$$\frac{1}{1} \underset{s^{t+1}; a_i^{t+1}_i}{E} [r_i(s; a_i)] > \frac{1}{1} \underset{s^t; a_i{}_i^t}{E} [r_i(s; a_i)]$$

$$\underset{s^{t+1}; a_i^{t+1}_i}{E} [r_i(s; a_i)] > \underset{s^t; a_i{}_i^t}{E} [r_i(s; a_i)]: \tag{6}$$

□

We next show that J ( ) is lower bounded by the sum of the individual action-matching rewards for all agents i, over all states.

Theorem 1 (action-matching objective). The joint action-matching objective J ( ) is lower bounded by the following sum over individual action-matching rewards $1_{a_i = a^E_i(s)}$:

$$L( ) := \sum_{s 2 S} (s) \left[\sum_{i=1}^{K} \underset{a_i \, i}{E} [1_{a_i = a_i}(s)] \right] : \tag{3}$$

Proof. Let us begin by rewriting J ( ) in terms of action mismatches.

$$J( ) = \sum_{s 2 S} (s) \left[(K \quad 1) + \underset{a}{E}[1 \quad 1^S_{i \, a_i = a_i}(s)]_E \right]$$

$$= \sum_{s 2 S} (s) \left[(K \quad 1) + 1 + \underset{a}{E}[\quad 1^S_{i \, a_i = a_i}(s)] \right] :$$

This formulation allows us to apply the Union Bound:

$$J( ) \quad \sum_{s 2 S} (s) \left[K + \sum_{i=1}^{K} \underset{a_i \, i}{E} [\quad 1_{a_i = a^E}(s)]_i \right]$$

$$= \sum_{s 2 S} (s) \left[\sum_{i=1}^{K} \underset{a_i \, i}{E} [1 \quad 1_{a_i = a^E_i}(s)] \right]$$

$$= L( ):$$

□

Theorem 1 relates the multi-agent imitation learning objective to a sum over single-agent imitation learning objectives. This is important because each agent updates independently to improves its own learning objective in our setting. Note also that the additive form of L( ) is similar to the value factorization assumptions made by algorithms like VDN (Sunehag et al. 2018).

It is difficult to say anything directly about the expected action-matching reward of agent j = i, $E_{s^{t+1}}[1^E_{= a_j}{}_j(s)]$, as $^{t+1}$ may be a state distribution over which agent j makes more mistakes (i.e. taking actions that don't match the expert's). While in general agent j's expected reward may decrease due to agent i's update, we show that under Assumptions 1 and 1, agent i's update increases a lower bound to L( ) that is independent of the state distribution.

Corollary 1 (lower bound). Let be the minimum probability of visiting any state. For all , L( ) is lower bounded by L( ): $_K$

$$L( ) > \sum_{s 2 S} \sum_{i=1} \underset{a_i \, i}{E} [1_{a_i = a_i}(s)] =: L( ): \tag{4}$$

Proof. The proof follows from the definition of $L()$ and that for all $s \; 2 \; S$, and all encountered in training, $(s) > $. □

Observation 1. *$L()$ is bounded by*

$$0 \quad L() \quad jSjK:$$

Lemma 7. *Suppose the joint expert policy $_E$ is deterministic. Then $_E$ is the unique maximizer of $L()$.*

Proof. Proof for this Lemma follows closely the proof for Lemma 5.

(Maximization) Since $_E$ is deterministic, by definition, for each agent i, each term

$$\underset{a_i \; E_i}{E}[1_{a_i = a_i}(s)] = 0 \qquad a_i$$
$$\underset{E_i}{E}[1 \quad 1_{a_i = a_i}(s)] = 1:$$

Summing over all agents and taking the sum weighted by over all states, $L \; (\;)_E = jSjK$, which means that achieves the upper bound of L .

(Uniqueness) Suppose there is at least one agent policy $_i \neq \quad _{E_i}$ such that the joint policy also achieves the upper bound, i.e. $L \; () = jSjK$. Then there must be a state s such that with non-zero probability, $a \quad _i(s)$ such that $a_i = a^E_i$. It follows that $E_{a_i \; E} i \; [1_{a_i = a^E_i}(s)] > 0$, meaning $L(_E) < jSjK$. By contradiction, $_E$ is the unique optimizer of L. □

Lemma 6 states that a single agent i updating its policy to improve its own GAIL loss is equivalent to increasing the expected action reward $r \; _i(s; a)$, where the expectation over states is with respect to some updated state visitation distribution $^{t+1}$ (Equation 6). Next, we make an assumption to relate our action-matching reward to the GAIL discriminator reward that is improved by the GAIL algorithm.

Assumption 1 (action-matching reward). *For all agents i and all states s, an increase in the expected converged GAIL discriminator reward implies an increase to the expected action-matching reward function:*

$$\underset{a_i \; _i^{t+1}}{E}[r \; _i(s; a_i)] > \underset{a_i \; _i^t}{E}[r \; _i(s; a_i)]$$
$$=) \quad \underset{a_i \; _i^{t+1}}{E}[1_{a_i = a_i^E}(s)] > \underset{a_i \; _i^t}{E}[1_{a_i = a_i^E}(s)]:$$

This assumption is not as strong as it may first appear.

First, note that the converged GAIL reward consists of the negative discriminator prediction, $r \; _i(s; a_i) = \quad \log D(s; a_i)$.

The discriminator predicts the likelihood ratio between the target visitation and the mix of target and agent visitation.

$$r_i (s; a_i) = \quad \log D(s; a_i) = \quad \log \frac{_E(s; a_i)}{_E(s; a_i) + (s; a_i)}$$
$$= \quad \log \frac{_E(s) \; _E(a_i js)}{_E(s) \; _E(a_i js) + (s) \; (a_i js) \; _E(s)}$$
$$\log \frac{_E(a_i js)}{_E(s) \; _E(a_i js) + \quad (a_i js)} =:$$
$$r(s; a_i):$$

The minimal state visitation assumption states that (s) for all s, allowing us to relate $r \; _i(s; a_i)$ to a state-visitation independent reward, $r \; (s; a_i)$. As we will argue next, $r \; (s; a_i)$ is a similar quantity to the action-matching indicator reward.

To see this, first suppose $a_i = a_i^E$, which would imply that the action-matching indicator function is 0. Since the expert policy is assumed to be deterministic, $_E(a_i js) = 0$, implying that $r(s; a_i) = 0$ as well. If $a_i = a^E_i$, then $r(s; a^E_i)$ is not zero, and the only way for agent i to increase $r \; (s; a_i^E)$ is to increase $(a^E_i js)$ — the probability of matching the expert's action. Thus, an increase in $r \; _i(s; a_i)$ implies an increase in $r \; (s; a_i)$, which behaves similarly to the action-matching indicator function.

Thus far, we have established that each individual agent's policy improvement under the GAIL reward improves a lower bound to the joint action-matching objective, $J()$. The following shows that within finite updates, each agent will be able to independently improve its value function until it converges to the expert policy.

Condition 2. Let $V()$ denote the value of an agent following a single-agent imitation learning algorithm. $jV(\; _t) \; V(^E)j$ is then the optimality gap at update t of the agent. Suppose that $jV(_t) \quad V(^E)j \quad (t)$, where as $t \; ! \quad 1$, $(t) \; ! \quad 0$.

This condition says that the single-agent imitation learning process should converge to the optimal (expert) policy with convergence rate dictated by (t). For our setting, Guan, Xu, and Liang (2021) shows that the single-agent GAIL algorithm converges (Theorem 3 and 4).

The next corollary shows that convergence of the single-agent imitation learning process is sufficient to guarantee the convergence of the multi-agent imitation learning scheme discussed in the main paper.

Corollary 2. There exists $H\ 2\ N^{+}; H < 1$ such that within H updates, agent i is able to improve its policy such that it increases the probability of matching the expert's action, summed over all states:

$$\sum_{s 2 S} \left[\frac{1}{c} E_{a^{t+H}_{i}} [r_{i}(s; a)] \right] > \sum_{s 2 S} \frac{1}{c} E_{a_{i}\ t} [r_{i}(s; a)] :$$

Proof. For a single agent i, define $V() = \sum_{s 2 S} (s)_{c} E_{a_i} [r_{i}(s; a)]$. Rewrite as follows:

$$\begin{aligned} V() &= \sum_{s 2 S} (s)_{a} E_{i} [r_{i}(s; a)] \\ &= \frac{1}{c} \sum_{s 2 S} ((s) \quad)_{a} E_{i} [r_{i}(s; a)] + \frac{1}{c} \sum_{s 2 S} {}_{a} E_{i} [r_{i}(s; a)]: \end{aligned}$$

Define the first term as $a() := \frac{1}{c} \sum_{s 2 S} ((s) \quad) E_{a_i} [r_{i}(s; a)]$, and the second term as $b() := \frac{1}{c} \sum_{s 2 S} E_{a_i} [r_{i}(s; a)]$. Note that $b()$ is the quantity of interest in the corollary.

The properties of the max operation directly imply that

$$\begin{aligned} \max V() &= \max[a() + b()] \\ & \max a() + \max b(): \end{aligned}$$

The expert policy $_{E}$ should maximize $V()$ for any single-agent imitation learning algorithm. Note also that $_{E}$ maximizes $a()$ and $b()$. Thus in our setting, the inequality in the above set of equations is actually an equality for the expert policy:

$$\begin{aligned} \max V() &= V(_{E}) \\ &= a(_{E}) + b(_{E}) = \max a() + \max b(): \end{aligned}$$

Assume that there is a policy $= _{E}$ such that $b() < b(_{E})$, and that $b()$ cannot be improved within finite updates. This contradicts Condition 2, which establishes that the single-agent imitation learning algorithm should be able to improve the agent policy until its value converges to the value of the expert policy. □

Corollary 2 implies that within a finite number of policy updates by agent i, the quantity $L()$ increases, because the other terms corresponding to agents $j = i$ are unchanged. By Theorem 2 and Lemma 7, $L()$ is upper bounded by the constant $jSjK$. Thus, by the monotone convergence theorem, the objective $L()$ converges. Further, the expert policy $_{E}$ is a maximizer of $L()$, $L()$, and $J()$ (Lemma 5 and Lemma 7).

Mixed Task and Imitation Reward

Theorem 3. Let R_T be the reward function used to train the expert policies $_{E}$, and let the expert policies have converged with respect to R_T (i.e., they are in a Nash equilibrium with respect to reward R_T). Then $_{E}$ are a Nash equilibrium for reward functions of the form, $R_T + R_{I;i}$, for any $; > 0$.

Proof. Let $R_{c;i} = R_T + R_{I;i}$. The following reasoning is on a per-agent basis, so we drop the i from $R_{c;i}$ and $R_{I;i}$ for convenience. For $_{E_i}$ to not be a Nash equilibrium with respect to R_c there needs to exist a policy $\tilde{}_i$ such that

$$E[R_c(\tilde{}_i(s)j_{E\ i})] > E[R_c(_{E_i}(s)j_{E\ i})]:$$

That implies

$$\begin{aligned} & E[R_T(\tilde{}_i(S)j_{E\ i})] + E[R_I(\tilde{}_i(S)j_{E\ i})] \\ & > E[R_T(_{E_i}(s)j_{E\ i})] + E[R_I(_{E_i}(s)j_{E\ i})]: \end{aligned}$$

But by definition, for all $_{E_i}(s)$,

$$E[R_T(_{E_i}(s)j_{E\ i})] \quad E[R_T(\tilde{}_i(S)j_{E\ i})]$$

and

$$E[R_I(_{E_i}(s)j_{E\ i})] \quad E[R_I(\tilde{}_i(S)j_{E\ i})];$$

which is a contradiction. □

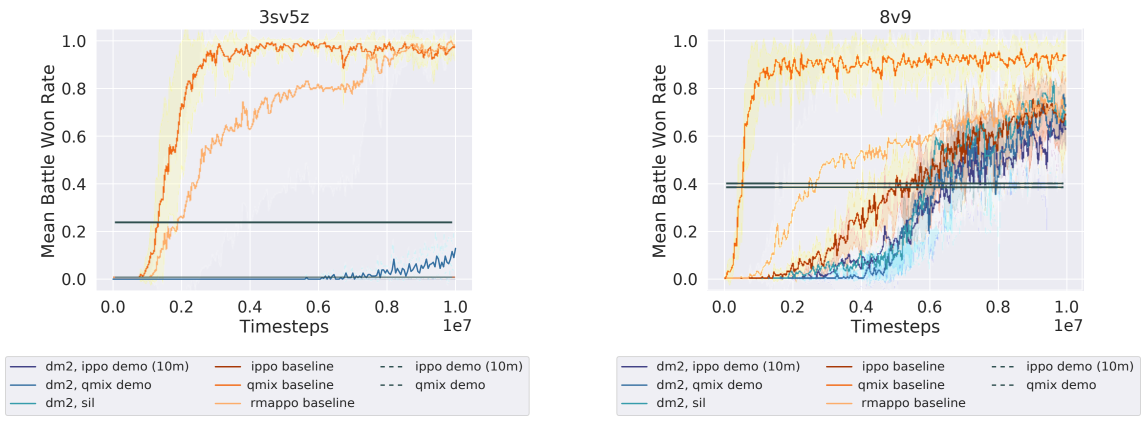


Figure 4: DM^2 comparison with baselines in additional maps in the StarCraft domain. These results are an extension of the main results from the paper.

B Supplemental Experimental Results

This section provides further commentary on results in the main paper and presents additional, supporting experiments. Figure 4 is an extension of our core results on two more maps in the Starcraft domain (3sv5z and 8v9), presented here due to space constraints.

Demonstration Styles of Ablation Study. The ablation study examines expert demonstrations that vary in two dimensions: co-trained versus concurrently sampled. For co-trained agents with demonstrations sampled non-concurrently, the demonstrations may be sampled from co-trained expert policies, but each agent's demonstrations originate from disjoint episodes. However, for agents that were not trained together but whose demonstrations are sampled concurrently, demonstrations could be obtained from expert policies that were each trained in separate teams, but executed together in the same environment. To ensure that each expert policy is of similar quality—despite not being trained together—the joint expert policies are trained with different seeds of the same algorithm.

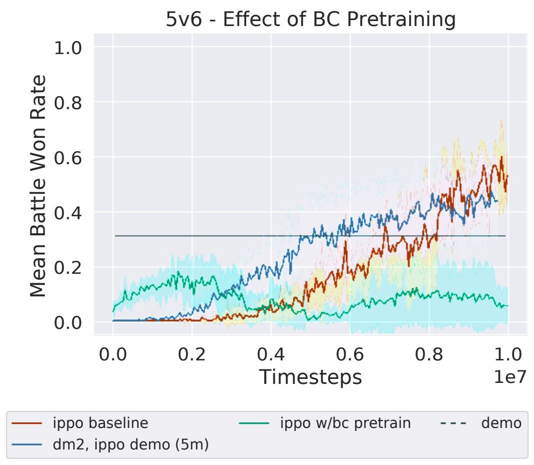
Behavioral Cloning Pretraining. Distribution matching is not the only method to use demonstrations. A more naive approach to utilize demonstrations is to use behavioral cloning (BC) (Bain and Sammut 1995) (a form of supervised learning) on the dataset of state-action pairs $D = \langle(s_0; a_0); (s_1; a_1); \ldots; (s_N; a_N)\rangle$. BC is accomplished by learning to predict the maximum likelihood action according to the dataset on the states present in the dataset. In practice, this prediction is learned by minimizing the negative log likelihood of the expert action on these states. This experiment pre-trains the agent policy with BC before the agents interact with the environment, after which point they learn using IPPO.

BC typically suffers from a distribution mismatch problem (also known as the covariate shift problem), where the agent's state visitation when interacting with the environment differs from the expert data distribution, leading to poor imitation even in single-agent settings. Behavioral cloning also requires a dataset of a size that increases quadratically with the horizon of the problem to learn successful policies (Ross, Gordon, and Bagnell 2011). These issues are likely to be exacerbated when dealing with multiple agents. The agents might minimize the supervised learning loss, but it is unlikely that the agents would learn to coordinate effectively. A second issue with using the supervised learning loss to pre-coordinate agent policies is that such coordination is unlikely to last once training with IPPO proceeds.

The result for this alternative usage of the expert demonstrations is presented in Figure 5. As detailed above, BC does not learn to imitate the demonstrations well enough to recover their performance (indicated by the dashed lines), and as training proceeds, the benefits of BC vanish as IPPO training proceeds.

Effect of GAIL Reward. DM^2 consists of IPPO trained with r_{mix}, which is a mixture of the environment and GAIL reward signal. Here, we present a brief ablation on the mixed reward (Figure 6). For both IPPO demonstration qualities, IPPO trained on the GAIL reward achieves a far lower win rate than IPPO trained on the environment reward (labelled as IPPO baseline) or DM^2. This result provides further evidence that the primary benefit derived by DM^2 comes from the coordination shown in the demonstrations, rather than from individual imitation of expert behaviors.

Sensitivity to Demonstration Quality. One finding in the experimental section of the paper is that DM^2 is relatively insensitive to demonstration quality beyond a certain baseline level of competence. To further support this claim, we train DM^2 with more demonstration qualities (where demonstrations are sampled from IPPO policies). Figure 7 shows that the algorithm improves monotonically as the demonstration quality improves, but quickly saturates. This result indicates it is important to supply good demonstrations, but not necessary to supply optimal demonstrations.



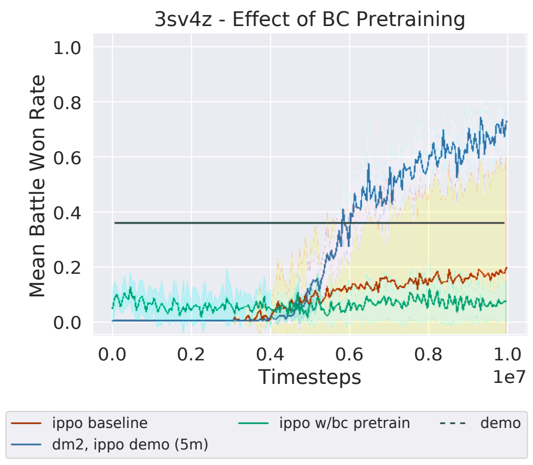


Figure 5: IPPO trained with r_{mix}, with and without behavioral cloning pretraining on the expert dataset. Demonstration qualities are shown as horizontal lines. All experiments are performed with demonstrations sampled from IPPO at 5m timesteps of training.

C Experimental Details

Implementation Details. The algorithm implementations are based on the multi-agent PPO implementations provided by Yu et al. (2022) (MIT license) and the PyMARL code base (Samvelyan et al. 2019) (Apache license). The StarCraft environment is also provided by Samvelyan et al. (2019) (MIT license).

All decentralized MARL implementations in this paper have fully separate policy/critic networks and optimizers per agent. That is, there is no parameter sharing amongst agents. For all IPPO agents, the policy architecture is two fully connected layers, followed by an RNN (GRU) layer. Each layer has 64 neurons with ReLU activation units. The critic architecture is the same as the policy architecture. For DM^2 agents, the policy and critic architectures are identical to IPPO. The discriminator architecture consists of two fully connected layers with tanh activation functions.

The centralized MARL algorithms implement agent policy networks with parameter sharing, where agents have a centralized value network. For QMIX agents, the policy architecture is the same except there is only a single fully connected layer before the RNN layer [5]. We attempted running QMIX with the the IPPO agent architecture, but found that the performance of QMIX significantly suffered (Figure 8 on 5v6). Thus, for the QMIX experiments in the main body of the paper, the better-performing policy architecture was applied. RMAPPO agents were trained directly using the code published by Yu et al. (2022).

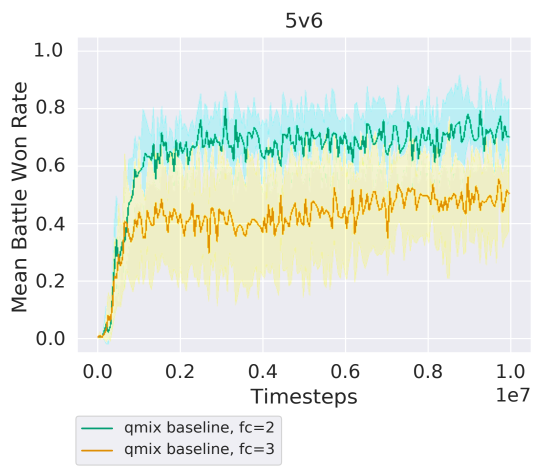
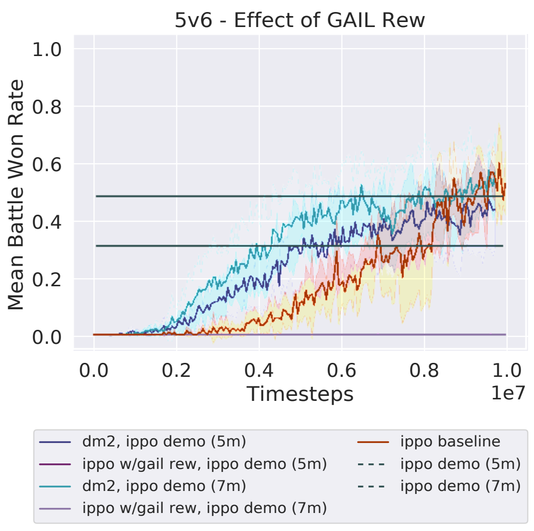


Figure 8: QMIX is sensitive to the agent policy architecture. Performance on the 5v6 task suffers significantly when an extra fully connected layer is added.

Hyperparameters. For QMIX, the default parameters specified in Rashid et al. (2018) are used for both tasks. For IPPO, and the IPPO component of DM^2, mostly default parameters (as specified in (Rashid et al. 2018; Yu et al. 2022)) were used. Algorithm hyperparameters that varied between tasks or were tuned are provided in Table 1. The remaining hyperparameters may be viewed with the code repository.

We found that for DM^2 to learn successfully from QMIX demonstrations, it was sometimes necessary to inject a small amount of random noise into the demonstration sampling process, so that the demonstrations did not constrain the exploration of the learning policies. Specifically, the demonstrations from QMIX were sampled from -greedy QMIX policies, where was chosen so that the win rate did not fall more than 10%. QMIX values are provided in Table 1.

[5]This is the architecture used in Rashid et al. (2018)



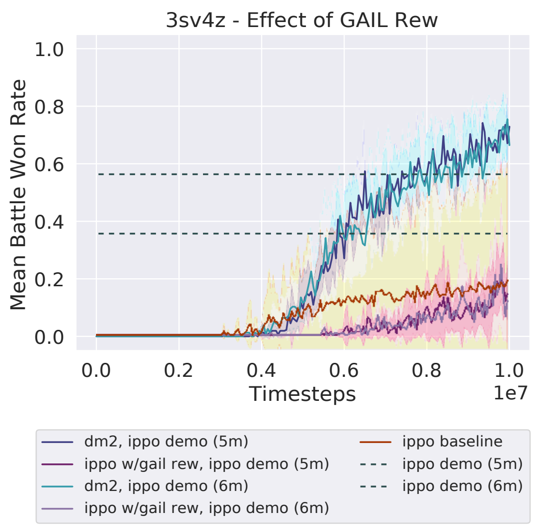


Figure 6: IPPO trained with r_{gail} and IPPO trained with r_{env} only on the 5v6 and 3sv4z tasks. DM^2 is shown for reference. Demonstrations are sampled from IPPO policies. The win rates achieved by demonstrations are plotted as horizontal lines.

We conducted a hyperparameter search over the following GAIL parameters: the GAIL reward coefficient, the number of epochs that the discriminator was trained for each IPPO update, the buffer size, and the batch size. The final selected values are given in Table 2.

Computing Architecture. All IPPO, QMIX, DM^2 experiments were performed without parallelized training; RMAPPO experiments were performed with parallelized training (as is the default in the RMAPPO codebase). The servers used in our experiments ran Ubuntu 18.04 with the following configurations:

- Intel Xeon CPU E5-2698 v4; Nvidia Tesla V100-SXM2 GPU.
- Intel Xeon CPU E5-2630 v4; Nvidia Titan V GPU.
- Intel Xeon Gold 6342 CPU; Nvidia A40 GPU.

	5v6	3sv4z	3sv3z
epochs	10	15	15
buffer size	1024	1024	1024
gain	0.01	0.01	0.01
clip	0.05	0.2	0.2
qmix epsilon	0	0	0.1

Table 1: IPPO Hyperparameters.

	5v6	3sv4z	3sv3z
gail rew coef	0.3	0.05	0.3
discr epochs	120	120	120
buffer size	1024	1024	1024
batch size	64	64	64
n exp eps	1000	1000	1000

Table 2: GAIL Hyperparameters.

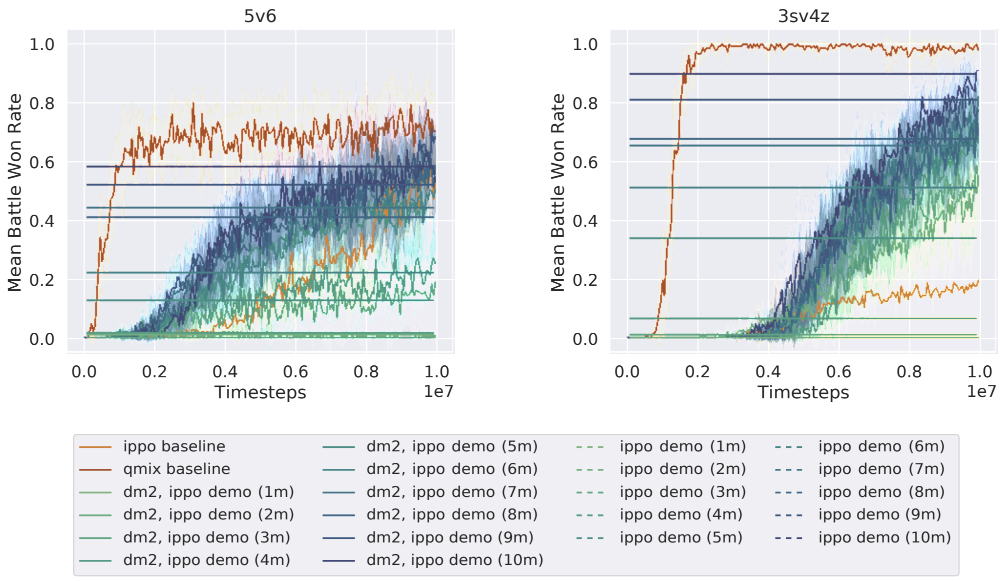


Figure 7: Learning curves of DM^2 (our method) trained with demonstrations sampled every million steps in the learning of the original demonstrator policy. Horizontal dotted lines indicate the demonstration qualities, colored to match corresponding learning curves. QMIX (centralized baseline) is included for reference.