Trajectory Diversity for Zero-Shot Coordination

Andrei Lupu [1] **Brandon Cui** [2] **Hengyuan Hu** [2] **Jakob Foerster** [2]

Abstract

We study the problem of zero-shot coordination (ZSC), where agents must independently produce strategies for a collaborative game that are compatible with novel partners not seen during training. Our first contribution is to consider the need for diversity in generating such agents. Because self-play (SP) agents control their own trajectory distribution during training, each policy typically only performs well on this exact distribution. As a result, they achieve low scores in ZSC, since playing with another agent is likely to put them in situations they have not encountered during training. To address this issue, we train a common best response (BR) to a population of agents, which we regulate to be diverse. To this end, we introduce *Trajectory Diversity* (TrajeDi) – a differentiable objective for generating diverse reinforcement learning policies. We derive TrajeDi as a generalization of the Jensen-Shannon divergence between policies and motivate it experimentally in two simple settings. We then focus on the collaborative card game Hanabi, demonstrating the scalability of our method and improving upon the cross-play scores of both independently trained SP agents and BRs to unregularized populations.

1. Introduction

In this paper, we use policy diversity as a tool to improve cross-play (XP) scores in zero-shot coordination (ZSC).

Introduced by Hu et al., ZSC is the problem of independently training two or more agents in a cooperative game such that their strategies are compatible and achieve high return when they are paired together at test time. Since it is impossible to exactly agree on an arbitrary strategy with all humans ahead of time, solving ZSC is required for AI agents that can cooperate with humans , such as rescue robots or self-driving cars.

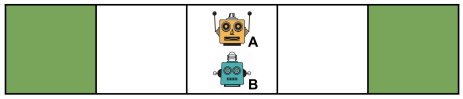


Figure 1: The corridor coordination task admits three optimal SP policies, only one of which is suitable for ZSC.

The challenge of the ZSC framework arises from the fact that many collaborative settings admit multiple joint-strategies that are optimal under *self-play* (SP) (Tesauro, 1994) yet incompatible. Thus, if we naively train two independent agents in SP, there is no guarantee that they will converge to compatible policies.

For instance, consider the task of two robots coordinating in a corridor presented in Figure 1. At each step, robot A moves first, then robot B does the same after observing A's action. The robots can occupy the same cell and receive a positive reward only if they both reach the same end of the corridor. In pure SP, robots can form conventions, and so an optimal solution is for both to agree to always go in one direction (left or right). However, another pair of robots can reach the opposite convention, in which case pairing the two in XP would result in robots A and B going in separate directions, failing to solve the task. Instead, to be optimal under ZSC, robot B must learn to follow A, regardless of the direction in which A moves.

Hu et al. address the problem of ZSC by proposing *Other-Play* (OP), a method of preventing those conventions (e.g. agreeing on a given direction in the corridor) that rely on arbitrarily breaking the symmetries of the setting. However, OP reduces to SP if the symmetries are not known a priori or if they are absent from the game. In the corridor task, this occurs if the transition probabilities in each direction are slightly different or if the rewards at the end have different variances – all while maintaining the same expected return.

Instead, we propose a complementary approach that relies on the best response (BR) relationships of the optimal SP policies. In particular, if we have access to the entire solution space (e.g. robot A either going left or right in the corridor), we can train an agent to be the common BR to the largest possible subset of that space. The resulting agent would then be robust to the maximum number of potential

[1]Mila, McGill University (Work done while at Facebook AI Research) [2]Facebook AI Research. Correspondence to: Andrei Lupu <andrei.lupu@mail.mcgill.ca>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021. Copyright 2021 by the author(s).

partners, making it a prime candidate for ZSC. In the corridor setting, this BR as robot B is precisely the policy that follows A in either direction.

This approach allows to train good policies for ZSC without additional domain knowledge, but requires access to a diverse pool of optimal policies to serve as the training set for the BR. We introduce Trajectory Diversity (*TrajeDi*), our differentiable objective that generalises the Jensen-Shannon divergence between the different policies and, unlike other methods, is especially designed to produce diverse policies in partially observable multi-agent settings.

For the rest of the paper, we first situate our work in the literature in section 2, then establish the setting and formalism in section 3. In section 4, we begin by detailing how diversity can be used jointly with reward-based training and a common BR for the purpose of ZSC. Then, we introduce the TrajeDi objective for producing diversity, justify it theoretically and show that it naturally captures the notion of diversity that we seek. We also derive an approximate policy gradient estimator and discuss the associated trade-offs. Next, in section 5, we evaluate TrajeDi experimentally. Thanks to two MDPs and a matrix game, we provide empirical insights into the shortcomings of standard approaches and show the suitability of TrajeDi in discovering multiple optimal solutions. Afterwards, we proceed to demonstrate that TrajeDi scales well to arbitrarily complex settings by using it to improve ZSC scores in the collaborative partially observable card game Hanabi. Finally, we summarize and conclude with future research directions in section 6.

Overall, our contributions are three-fold. First, we highlight the role of diversity in training robust policies for ZSC and leverage it within a population-based approach. Secondly, we introduce TrajeDi, a general and differentiable information theoretic objective for training a diverse population of optimal policies. Finally, we demonstrate empirically the merit of our method, both in small, interpretable settings and in the large-scale coordination task of Hanabi.

2. Related Work

There is a growing corpus of works featuring diversity maximization in reinforcement learning, many of which leverage it as a means of exploration. For instance, Hong et al. (2018) maximize the KL divergence between the current policy and the distribution of past experiences, effectively driving the agent towards new state-action pairs. Instead, Gangwani et al. *minimize* the difference between the policy and the high reward transitions of the replay buffer, but train a diverse set of policies for populating that buffer. Relatedly, Cohen et al. deploy diverse conjugate policies in the local policy space and Parker-Holder et al. encourage diversity through action-based embeddings, again in the scope of driving exploration. In multi-agent reinforcement learning (MARL), there has been some success in using reward randomization to drive diversity (Tang et al., 2021). However, this requires the reward function to be factored into different contributions, making this method ill-defined in settings with a single reward source (e.g. the score in Hanabi).

Diversity has also been used in hierarchical reinforcement learning (HRL) to learn useful options by preventing redundancies. This can be done in multiple ways, such as through entropy maximization (Haarnoja et al., 2018) or by employing a supervised deep learning (DL) approach (Song et al., 2019; Florensa et al., 2017). DL methods are however not restricted to HRL. For instance, Eysenbach et al. focus on learning multiple skills that are distinguishable by a classifier based on state distributions. Crucially, they do so without any reward function, as a method of pre-training.

Regardless of the application, past works tend to formulate diversity either at the action level $(\pi(a|s))$ (Hong et al., 2018; Cohen et al., 2019; Parker-Holder et al., 2020), as a function of state distributions $(\mathbb{P}(s|\pi))$ (Song et al., 2019; Eysenbach et al., 2018; Florensa et al., 2017; Kumar et al., 2020) or of state-action distributions $(\mathbb{P}(s, a|\pi))$ (Gangwani et al., 2018). Unfortunately, these formulations are inadequate for MARL, where dynamics are non-Markovian, requiring agents to condition their actions on their entire observation history. Since multi-agent settings are our main concern, we formulate TrajeDi accordingly, making it the first trajectory-based $(\mathbb{P}(\tau|\pi))$ diversity objective, to the best of our knowledge.

A related field to diversity is that of imitation learning. Although the goal is strikingly different – to train a policy to closely match the behaviour of another – the methods often rely on a measure of dissimilarity between policies, with the main distinction being that it is being minimized rather than maximized. In particular, works such as Gangwani et al. (2018); Guo et al. (2018) and Ho & Ermon (2016) base their objective on the same statistical measure we do – the Jensen-Shannon divergence. Another notable example is PoWER (Kober & Peters, 2009), which performs imitation learning with a trajectory-based KL divergence objective, although weighted in a way that makes it unusable for diversity.

Our method also shares similarities with Fictitious Self-Play (FSP), which computes a BR to a mix of snapshots of the policy's own past behaviour (Heinrich et al., 2015). While we also compute a BR, we instead do so to the most recent version of an explicit population of other policies. We additionally train each policy in SP. Also, FSP is designed exclusively for zero-sum games, whereas we are concerned only with fully cooperative games.

Our work addresses directly the challenges of the ZSC framework introduced by Hu et al., which we summarize

in section 3. For this setting, Hu et al. propose a suitable training method called "*Other-Play*" that leverages domain knowledge of the game symmetries to find unambiguous solutions that perform well. Since symmetries are not always present or known, we instead rely on the structure of the policy space to find such solutions.

Finally, our method incorporates a population of policies within which we maximize diversity. As a result, it is a natural instance of population based training (PBT) and, when used with deep reinforcement learning (DRL) models, falls loosely in the legacy of Jaderberg et al. (2017).

3. Setting and Background

Throughout this paper, we assume a decentralized partially observable Markov decision process (Dec-POMDP). The Dec-POMDP $\mathcal{M}$ is defined by a tuple $(k, \mathcal{S}, \mathcal{A}, P, r, O, \gamma_{\mathcal{M}}, T)$, where k is the number of agents, $\mathcal{S}$ is the joint-state space, $\mathcal{A} = \times_{j=1}^{k} A^j$ is the joint-action space, and P and O are respectively the transition and observation functions. Finally, r is the reward function, $\gamma_{\mathcal{M}}$ is the reward discount factor and T is the horizon.

At time t, the Dec-POMDP is in state $s_t \in \mathcal{S}$ and produces a stochastic joint observation $o_t = (o_t^1, ..., o_t^k) \sim O(\cdot|s_t)$, which is appended to the action-observation trajectory $\tau_t = (o_0, a_0, \cdots, o_{t-1}, a_{t-1}, o_t)$. Individually, every agent $j \in \{1, \cdots, k\}$ updates its own trajectory $\tau_t^j = (o_0^j, a_0^j, \cdots, o_{t-1}^j, a_{t-1}^j, o_t^j)$ and samples an action $a_t^j \in \mathcal{A}^j$ using a stochastic policy of the form $\pi^j(a^j|\tau_t^j)$. π^j represents agent j's component of the decentralized joint-policy π and so the decision process is equivalent to selecting a joint-action $a_t = (a_t^1, ..., a_t^k) \in \mathcal{A}$ with probability $\pi(a|\tau_t)$. The environment then transitions to the state s_{t+1} with unknown transition probability $P(s_{t+1}|s_t, a_t)$, upon which all agents receive a common reward $r(s_t, a_t)$. Finally, taking into account the discount $\gamma_{\mathcal{M}} \in [0, 1]$, we let $R(\tau) = \sum_{t=0}^{T} \gamma_{\mathcal{M}}^t r(s_t, a_t)$ be the discounted return.

At the core of our objective formulation is the tree $\mathcal{T}$ of all possible trajectories. The probability of a trajectory $\tau = (o_0, a_0, ..., a_{T-1}, o_T)$ under a joint-policy π is given by $\mathbb{P}(\tau|\pi) = \varepsilon(\tau)\pi(\tau)$, where $\varepsilon(\tau) := \mathbb{P}(s_0) \prod_{t=0}^{T-1} P(s_{t+1}|a_t, s_t)$ summarizes the environment dynamics and $\pi(\tau) := \prod_{t=0}^{T-1} \pi(a_t|\tau_t)$ wraps the joint-action probabilities. The behaviour of a joint-policy is therefore characterized by the distribution it produces over trajectories and it is optimal if it maximizes the expected return $J(\pi) = \mathbb{E}_{\tau\sim\pi}\{R(\tau)\}$.

3.1. Policy Populations

When training n different joint-policies on the same task, we can write the average trajectory probability as $\mathbb{P}(\tau|\hat{\pi}) = \varepsilon(\tau)\hat{\pi}(\tau)$, where $\hat{\pi}(\tau) := \sum_{i=1}^{n} \frac{1}{n}\pi_i(\tau)$ is the trajectory-wise average policy[1]. Similarly, we let $\tilde{\pi}$ be the action-wise average policy, such that for all pairs (τ_t, a) we have $\tilde{\pi}(a|\tau_t) := \sum_{i=0}^{n} \frac{1}{n}\pi_i(a|\tau_t)$. Although the two definitions are similar, we insist on the fact that generally $\hat{\pi}(\tau)$ does **not** equal $\prod_{t=0}^{T-1} \tilde{\pi}(a_t|\tau_t)$, and we use different notation (hat and tilde) as a reminder.

3.2. Zero-Shot Coordination

In the zero-shot coordination (ZSC) framework (Hu et al., 2020), two players must each *independently* train a joint-policy for a collaborative game, using a pre-agreed training algorithm of their choosing. After training, they are paired together and evaluated in cross-play (XP) by taking individual components from each joint-policy. If π_1 and π_2 are the joint-policies produced independently by the two players, their ZSC performance is the average cross-play (XP) return obtained when matching individual components from each joint-policy. This objective is formally given by

$$J_{\text{XP}}(\pi_1, \pi_2) = \frac{1}{2}(J(\pi_1^1, \pi_2^2) + J(\pi_2^1, \pi_1^2)). \quad (1)$$

Crucially, players do not have access to each other's joint-policy during training and must therefore employ a training procedure that will produce compatible policies when run separately. In particular, they cannot rely on arbitrary conventions, unlike in SP.

The procedure proposed by Hu et al., Other-Play, takes a step towards this by exploiting the known symmetries of the game. Given those symmetries, OP performs a form of asymmetric domain randomization during training by presenting each agent with a different permutation $\phi(\mathcal{S}, O, \mathcal{A})$ of the environment at every episode. For instance, in the corridor game, what robot A sees as "left" is perceived as "right" by B. These random permutations prevent coordinated symmetry breaking to form conventions, thus making the learned policies more robust for ZSC.

Because OP and our method address different facets of ZSC, they are fully compatible and so we use them jointly in our Hanabi experiments.

The ZSC framework, OP and our method can be applied to collaborative games with any number of players, but the set of possible XP teams grows exponentially in size and careful consideration has to be given to duplicate agents (e.g. 3 agents playing π_1, 2 agents playing π_2 in a five player game). For ease of exposition, we therefore restrict ourselves to two-player games throughout the paper. We also omit the "joint-" qualifier in writing, for the same reason. Indeed, the method

[1] We use superscript (π^j) to denote individual components of the same joint-policy and subscript (π_i) to denote different joint-policies within a population.

we present in section 4 is agnostic as to whether the actions are produced by a single policy or multiple decentralized ones.

4. Method

4.1. Population Based Training Setup

The core objective of this paper is applying diversity to the ZSC setting and deriving a method for training a maximally diverse population of policies with a common BR.

To do so, we first assume a population $(\pi_1, ..., \pi_n)$ of n policies, which we train in SP, but also regularize to be as diverse as possible. Additionally, we have a $(n+1)$-th policy denoted BR, which we train to be a common BR to every population member. As a result, we must also optimize for the XP performance between each policy in the population and the BR. The resulting loss is

$$\mathcal{L}(\text{BR}, \pi_1, ..., \pi_n) = -\Bigg[\sum_{i=1}^{n} \left(J_{\text{XP}}(\text{BR}, \pi_i) + J(\pi_i)\right) + J(\text{BR}) + \alpha Diversity(\pi_1, ..., \pi_n)\Bigg], \quad (2)$$

where J_{XP} is the objective defined in eq. 1, $Diversity$ is some measure of population diversity (for which we will substitute TrajeDi in later sections) and α is a tunable weight.

Our approach is informed by two core intuitions. The first is that we can consider the entire population as a training set with the sole purpose of guiding the learning of the BR. When evaluating ZSC, the BR will then be compatible with a much larger set of possible XP partners, while also hopefully generalizing to unseen ones.

The second intuition is what we call the *BR classes hypothesis*. We consider that the space of good SP policies can be partitioned into different BR classes, where two distinct policies π_1 and π_2 belong to the same class if and only if they share a common BR, π_3, satisfying $J_{\text{XP}}(\pi_1, \pi_3) = J(\pi_1)$ and $J_{\text{XP}}(\pi_2, \pi_3) = J(\pi_2)$. Furthermore, we assume that BR classes of large cardinality are much less frequent than small ones. Indeed, many settings, like Hanabi, admit a vast number of policies that are only compatible with themselves due to reliance on arbitrary conventions. Because those policies are their own unique BR, they each form their own BR class of size 1. In turn, general policies that can respond well to multiple kind of behaviours are much rarer. It follows that those policies represent natural equilibria on which to coordinate for ZSC, and our method is designed to find one of them, as we illustrate in section 5.2.

Note that in many settings, agents can only maximize SP scores by becoming highly specialized and forming arbitrary conventions (Hu et al., 2020). In such cases, the largest BR class is not necessarily the best in terms of SP. Then, minimizing the loss in eq. 2 may result in potentially lower SP scores, but will generally improve XP performance of the BR.

4.2. Trajectory Diversity

We now seek to make eq. 2 precise by specifying a $Diversity$ measure.

From a trajectory perspective, two policies π_1 and π_2 can be said to differ if they induce different trajectory distributions – an intuition which can be made precise by the use of a statistical divergence measure $\mathcal{D}(\pi_1(\tau), \pi_2(\tau))$. For a number of reasons detailed below, our measure of choice in this paper is the *Jensen-Shannon divergence* (JSD), as used in GANs (Goodfellow et al., 2014). Formally, when applied to trajectory distributions, the JSD takes the form

$$\text{JSD}(\pi_1, ..., \pi_n) = H(\hat{\pi}) - \sum_{i=1}^{n} \frac{1}{n} H(\pi_i), \quad (3)$$

where $(\pi_1, ..., \pi_n)$ is a population of n policies, H is the Shannon entropy and $\hat{\pi}$ is the trajectory average defined in section 3.1. Alternatively, we can express eq. 3 as the average Kullback-Leibler divergence $\mathcal{D}_{KL}$ from the population policies to $\hat{\pi}$:

$$\begin{aligned}\text{JSD}(\pi_1, ..., \pi_n) &= -\frac{1}{n} \sum_{i=1}^{n} \sum_{\tau} \mathbb{P}(\tau|\pi_i) \left[\log \frac{\hat{\pi}(\tau)}{\pi_i(\tau)}\right] \\ &= \frac{1}{n} \sum_{i=1}^{n} \mathcal{D}_{KL}(\pi_i||\hat{\pi}),\end{aligned}$$

where the $\varepsilon(\tau)$ factor cancels out inside the log. This latter form is particularly amenable to deriving policy gradients, which we do in section 4.4.

Some advantages of the JSD are immediately apparent. First, unlike the Kullback-Leibler divergence, which is limited to the pair-wise comparison of two distributions, the JSD naturally generalizes to an arbitrary number of policies, making it particularly relevant when seeking a population that is globally diverse. Secondly, the JSD is defined everywhere and symmetric over policies, which helps intuition and simplifies the implementation. Thirdly, the square root of the JSD is a metric when $n = 2$, in which cases maximizing eq. 3 increases the distance between the policies in a proper metric space (Endres & Schindelin, 2003).

Additionally, maximizing the JSD over a population of policies has an intuitive interpretation: the first term of eq. 3 encourages the population to cover a large subset of $\mathcal{T}$ by maximizing its collective entropy. Meanwhile, the second term drives each policy distribution to be contracted to as

few trajectories as possible. This balance results in the JSD being in fact unaffected by the individual entropy of each policy, differentiating it starkly from entropy maximization techniques. Indeed, the only thing it depends on is the degree of overlap between policies and so it is bounded between 0 (when all policies induce identical distributions) and $\log n$ (when the trajectory distributions are all disjoint). In particular, this implies that the maximum value of the JSD is independent of the number of possible trajectories. In addition, the sampling procedure underlying the JSD means that policies in any given pair are only required to act differently along trajectories on which both have non-zero probability mass. In other words, it captures the intuition that it is meaningless to evaluate a policy in the parts of the trajectory tree that it never visits naturally. As a result, the JSD measures precisely the notion of diversity that we seek to optimize.

4.3. Action Discounting

Next, consider complex settings with long horizons or many degrees of freedom in the possible solutions. For instance, suppose that along every possible trajectory, there is a *nullspace* – a small subset of states where two or more actions are equally optimal, but have little or no effect on the perceived behaviour of the agent. For instance, this could be an inconsequential action such as a robot choosing which arm to use for pressing a button, or a permutation of several actions, when order is irrelevant to the final outcome. In such settings, the JSD objective is too sensitive, as it is possible to produce a vast number of near-identical policies that nonetheless maximize the objective due to acting differently in the nullspace. This problem can theoretically be solved if the policy population is sufficiently large so as to "saturate" the nullspace, for example by having enough policies to cover all possible action permutations. In practice, however, this can require a combinatorial number of policies, which is infeasible in all but the simplest settings.

To address this issue, we introduce a generalization of the JSD objective defined in eq. 3. For a policy π_i unrolled on trajectory τ, let the local action kernel be defined as

$$\delta_{i,t}(\tau) := \prod_{t'=0}^{T} \left[\pi_i(a_{t'}|\tau_{t'})\right]^{\gamma^{|t-t'|}}, \tag{4}$$

where γ is a discounting factor in $[0, 1]$. Furthermore, similarly to what we did before, let $\hat{\delta}_t(\tau) := \sum_{i=1}^{n} \frac{1}{n} \delta_{i,t}(\tau)$ be the average local action kernel. Then, replacing π_i by $\delta_{i,t}$ and $\hat{\pi}$ by $\hat{\delta}_t$ in the logarithms of the JSD and averaging over time steps, we obtain

$$\text{JSD}_\gamma(\pi_1, ..., \pi_n) := -\frac{1}{n} \sum_{i=1}^{n} \sum_{\tau} \mathbb{P}(\tau|\pi_i) \sum_{t=0}^{T} \frac{1}{T} \log \frac{\hat{\delta}_t(\tau)}{\delta_{i,t}(\tau)} \tag{5}$$

which is the final *TrajeDi* objective.

We can better understand eq. 5 by looking at the special cases covered by it. Setting $\gamma = 1$ yields $\delta_{i,t}(\tau) = \pi_i(\tau)$ and $\hat{\delta}_t(\tau) = \hat{\pi}(\tau)$ for all t, and we retrieve the initial JSD objective over trajectory distributions, making TrajeDi a strict generalization. Consequently, we will refer to eq. 3 as JSD_1 from now on. At the other extreme, $\gamma = 0$ makes $\delta_{i,t}(\tau) = \pi_i(a_t|\tau_t)$ and $\hat{\delta}_t(\tau) = \tilde{\pi}(a_t|\tau_t)$, and so maximizing JSD_0 produces policies that differ in their action choices at every state. In the appendix, we show that this is equivalent to the KL divergence from the mean at the level of individual actions.

For two deterministic policies π and π', the innermost term of JSD_0 can only take a value of 0 (if $\pi(a|\tau_t) = \pi'(a|\tau_t) = 1$) or 1 (if $\pi(a|\tau_t) = 1$ and $\pi'(a|\tau_t) = 0$). In that case, JSD_0 measures the expected number of times π and π' will "disagree" by selecting different actions.

In summary, JSD_1 is a very responsive trajectory-level objective that is best suited for "tight" settings with few degrees of freedom in optimal solutions. Its counterpart, JSD_0, is a much more stringent action-level measure of diversity, which can be used to push policies to differ at every state. Then, by tuning γ in eq. 5, TrajeDi allows us to smoothly interpolate between the two extremes. This can be done through a hyperparameter search, using prior knowledge, or in a dynamic fashion by reducing its value during training. Thus, given our intuitions and the results from section 5.3, we emit the following conjecture:

Conjecture 4.1. *Let $\gamma_1, \gamma_2 \in [0, 1]$, with $\gamma_1 > \gamma_2$ and let $(\pi_1, ..., \pi_n)$ be a population of policies, each inducing a distribution of trajectory set $\mathcal{T}$. Then, it holds that:*

$$\log n \geq \textit{JSD}_{\gamma_1}(\pi_1, ..., \pi_n) \geq \textit{JSD}_{\gamma_2}(\pi_1, ..., \pi_n) \geq 0.$$

Finally, we use a fixed horizon only for simplicity. Since every comparison is done trajectory-wise, it suffices to change T to $T(\tau)$ in eq. 5 to accommodate for trajectories of different length. In fact, given that $\pi(a|\tau_t) > 0$ for all π, τ_t and a, TrajeDi with $\gamma < 1$ can be approximated locally even in tasks with infinite horizons.

4.4. Objective Gradient

Due to the TrajeDi objective operating directly on policy probabilities, it lends itself particularly well to being incorporated in policy gradient algorithms. Here, we provide an approximate policy gradient and show the main steps to build intuition.

Evaluating the TrajeDi objective, even when sampling only a subset of trajectories, carries a computational cost that is quadratic in the size of the population. This is because the sum in the logarithm requires that each policy be unrolled

on the trajectories generated by all of the other policies. The exact gradient, which we provide in the appendix, suffers from the same issue, which can make it prohibitively expensive in many cases. In theory, sampling very few trajectories is sufficient to keep computations manageable, but it causes high variance and is at odds with batch methods in DRL, which for large models can require the processing of a large number of trajectories for a single policy update.

To bypass the issue, we first expand eq. 5 and approximate the $\log\left(n\hat{\delta}_t(\tau)\right)$ term in the sum by its tangent to obtain $\tilde{\text{JSD}}_\gamma$, with $\tilde{\text{JSD}}_\gamma \leq \text{JSD}_\gamma$. Then, assuming each policy π_i has a parametrization $\boldsymbol{\theta}_i$, we can take the gradient of this approximate objective with respect to $\boldsymbol{\theta}_i$. After an application of the product rule, the resulting gradient estimator is

$$\nabla_{\boldsymbol{\theta}_i}\tilde{\text{JSD}}_\gamma = -\mathbb{E}_{\tau\sim\pi_i}\left\{\frac{1}{T}\sum_{t=0}^{T}\frac{\hat{\pi}(\tau)}{\pi_i(\tau)}\delta_{i,t}(\tau)\nabla_{\boldsymbol{\theta}_i}\log\delta_{i,t}(\tau) + \left(\hat{\delta}_t(\tau) - \frac{1}{n}\log\delta_{i,t}(\tau)\right)\nabla_{\boldsymbol{\theta}_i}\log\pi_i(\tau)\right\}, \quad (6)$$

where $\hat{\pi}(\tau)$ and $\hat{\delta}_t(\tau)$ are both expectations and can be sampled. Because the outermost expectation is over trajectories sampled from π_i, policies do not receive gradients from trajectories generated by other policies. This is necessary in some distributed implementations of population based training, such as the one we use for Hanabi in section 5, but comes at the cost of higher variance due to the the importance sampling ratio in the first term. We provide a different variant of the gradient estimator, along with a full derivation, in the appendix.

In the appendix, we also show how eq. 6 can be modified for use in off-policy training, and in particular batch RL methods.

4.5. TrajeDi for Zero-Shot Coordination

We can now substitute the arbitrary diversity measure from eq. 2 by TrajeDi, obtaining a fully differentiable PBT loss:

$$\mathcal{L}(\text{BR},\pi_1,...,\pi_n) = -\left[\sum_{i=1}^{n}\left(J_{\text{XP}}(\text{BR},\pi_i) + J(\pi_i)\right) + J(\text{BR}) + \alpha\text{JSD}_\gamma(\pi_1,...,\pi_n)\right]. \quad (7)$$

Then, using the gradient estimator from eq. 6, the gradient for each policy can be computed using only data from its own replay buffer, which we find necessary for large scale asynchronous PBT setups, such as our Hanabi experiments.

We present the full TrajeDi PDT procedure for ZSC in algorithm 1.

Algorithm 1 TrajeDi PBT with Common Best Response

Parameters: n, γ, α, T, λ
$\mathcal{P} \longleftarrow (\pi_{\text{BR}},\pi_1,...\pi_n)$, parametrized by $(\theta_{\text{BR}},\theta_1,...,\theta_n)$
$\text{SPB}_i, \text{XPB}_i \longleftarrow$ SP and XP replay buffers for π_i
$\text{SPB}_{\text{BR}}, \text{XPB}_{\text{BR}} \longleftarrow$ SP and XP buffers for π_{BR}
while policies have not converged **do**
  for all $i \in \{\text{BR},1,...,n\}$ **do**
    $\tau \longleftarrow$ `GetEpisodeRollout`(π_i,π_i)
    $\text{SPB}_i \longleftarrow \text{SPB}_i \bigcup \tau$
    if $i \neq$ BR **then**
      $\tau \longleftarrow$ `GetEpisodeRollout`(π_i,π_{BR})
      $\text{XPB}_i \longleftarrow \text{XPB}_i \bigcup \tau$; $\text{XPB}_{\text{BR}} \longleftarrow \text{XPB}_{\text{BR}} \bigcup \tau$
    end if
  end for
  for all $i \in \{1,...,n\}$ **do**
    Sample batch B $\sim \text{SPB}_i$ and estimate $J(\pi_i,\pi_i)$
    Sample batch B$'$ $\sim \text{XPB}_i$ and estimate $J(\pi_i,\pi_{\text{BR}})$
    $\forall(j,t,\tau) \in \{1:n\}\times\{0:T\}\times$ B compute $\delta_{j,t}(\tau)$
    Estimate $\nabla_{\boldsymbol{\theta}_i}\tilde{\text{JSD}}_\gamma$ based on the $\delta_{j,t}$'s (eq. 6)
    $\theta_i \longleftarrow \theta_i + \lambda\nabla_{\boldsymbol{\theta}_i}\mathcal{L}(\text{BR},\pi_1,...,\pi_n)$
  end for
  $\theta_{\text{BR}} \longleftarrow \theta_{\text{BR}} + \lambda\nabla_{\boldsymbol{\theta}_{\text{BR}}}\mathcal{L}(\text{BR},\pi_1,...,\pi_n)$
end while

5. Experiments

5.1. Diversity of Solutions

We first provide a proof of principle result, showing that TrajeDi can be used to find diverse solutions.

To do so, we use the two tree-like environments shown in Figure 2. Both environments are deterministic and fully observable MDPs with $\gamma_{\mathcal{M}} = 1$. In either of them, the agent starts at state s_0 (the root) and makes its way to one of the terminal states, corresponding to the leaves of the tree. The agent then receives a reward of 1.0 if it reached one of the green shaded states and 0.1 otherwise.

In both trees, the high reward states are those for which the index is a power of 2. In the complete tree, this creates a higher concentration of reward on the left half of the tree, but leaves every trajectory equally likely under a random policy. In the incomplete tree, this makes each solution exponentially harder to reach. However, because the return is not discounted, every green terminal state remains optimal.

We implement simple policy-gradient policies and train 10 populations of n agents, where we let n be the number of different solutions. We train these populations both with and without the TrajeDi objective and, for each option, we test two different ways of initializing policies. The first is the random parameter initialization that is standard in RL. The second is what we call "*entropic initialization*". That is, we initialize each policy such that $\mathbb{P}(\tau|\pi) = \frac{1}{|\mathcal{T}|}$ for all

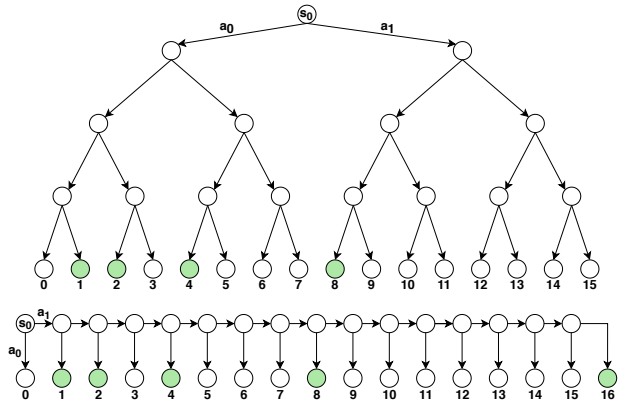
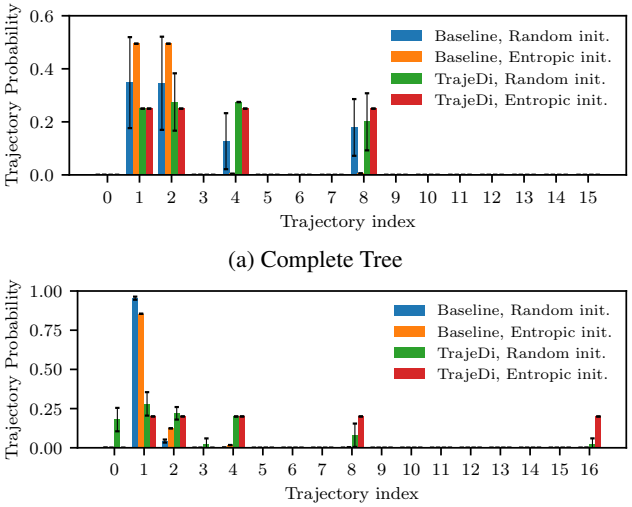


Figure 2: Two MDPs used to test the hypothesis that TrajeDi training produces diverse solutions. Both environments are fully observable, deterministic and undiscounted ($\gamma_{\mathcal{M}} = 1$). The reward for reaching a terminal state (numbered) is 1.0 for green shaded states and 0.1 otherwise. In the complete tree (top), the agent can move `left` or `right`. In the incomplete tree (bottom), the actions are `down` and `right`.

τ, thus maximizing its entropy $\mathcal{T}$. Finally, in the TrajeDi objective, we use $\gamma = 1$.

We present our results in Figure 3 by plotting the trajectory distributions ($\hat{\pi}(\tau)$) that each type of population converges to on average. In Figure 3a, we see that the baseline policies preferentially converge towards the left-most solutions of the complete tree, with the effect being significantly more pronounced when using entropic initialization. Meanwhile, the randomly initialized TrajeDi population still suffers from some variance, but outperforms both baselines. Finally, when paired with entropic initialization, TrajeDi reliably finds every solution and so maximizes diversity on all runs.

In the incomplete tree, each solution is exponentially harder to find than than the previous one, and so the difference between baseline and TrajeDi populations is more dramatic. Indeed, Figure 3b demonstrates that without a diversity objective, the baseline population collapses almost exclusively to the most accessible solution. Entropic initialization helps somewhat, but the effect is insufficient as even then the baseline puts over 85% of the probability mass on τ_1 while never reaching τ_8 and τ_{16}. The randomly initialized TrajeDi populations again outperform both baselines in terms of diversity, although this time it is at the cost of some optimality. The reason is that we put a high weight on the TrajeDi loss term ($\alpha = 4$ in eq. 7). As a result, when the population fails to find the higher index solutions altogether, some policies are pushed towards suboptimal trajectories. This effect is naturally mitigated with lower α, a smaller population size, or when using entropic initialization. In the latter case, we again retrieve perfectly diverse populations over solutions, as shown by the red bars in the plot.



(a) Complete Tree

(b) Incomplete Tree

Figure 3: Average trajectory probabilities of the population ($\hat{\pi}(\tau)$) on a) the complete tree and b) the incomplete tree environments of Figure 2. We compare baseline populations against populations enhanced with the TrajeDi objective and test both with random and entropic initialization. In both settings, TrajeDi populations with entropic initialization are by far the most reliable in finding all solutions. Values averaged over 10 populations, with error bars representing one standard deviation. Note that the y axis range is different between the two graphs.

5.2. Zero-Shot Coordination

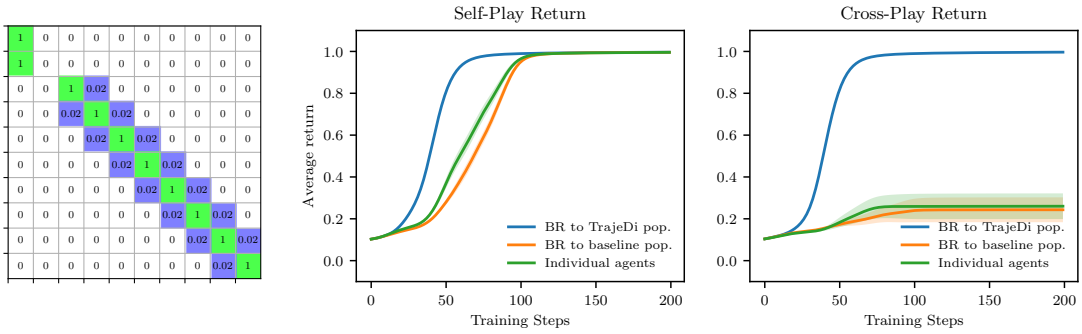
Next, we demonstrate that training a common BR as a regularizer to a TrajeDi-enhanced population results in a robust agent for ZSC[2].

We run our experiments on the single-step collaborative matrix game visualized in Figure 4a. In this game, player 1 must select a row while player 2 chooses a column independently. Once done, the actions are revealed, and both agents get the reward associated with the intersection of their choices.

We train 50 independent TrajeDi populations of two agents, each complete with a common BR, as given by eq. 7. We evaluate the BRs and compare against common BRs to unregularized populations, as well as to 50 individual agents trained independently.

We plot the performance in both SP and XP in Figure 4b. In SP, all schemes successfully achieve optimal return, but the BRs to TrajeDi populations do so faster. This is likely due to the TrajeDi objective providing a richer and more directed reward signal, but also due to the advantage diversity provides with exploration.

[2]The code is available online and can be run in-browser: https://bit.ly/33NBw5o



(a) Matrix Game

(b) Returns

Figure 4: a) Collaborative matrix game used to evaluate XP performance. b) Train and test performances on the matrix game. Shown are the results for BRs to TrajeDi-enhanced populations, BRs to baseline populations, and individual agents. TrajeDi allows faster learning in SP, and drives the BR to converge to the unique solution optimal in ZSC.

In XP, the TrajeDi curve is identical, demonstrating that BRs trained on diverse populations have indeed learned the most general solution (first or second row as the row player and first column as the column player). Meanwhile, individual agents and BRs to baseline populations perform only slightly better than chance, as off-diagonal payoffs tend to make the policies diverge from the optimal ZSC solution. Moreover, there is no statistical difference between the XP scores of the two baselines, supporting the hypothesis that PBT with a common BR provides no benefit without diversity.

Note that, because this setting is not temporally extended, the TrajeDi objective is the same regardless of the discounting term γ. This is also the only case where TrajeDi is equivalent to the diversity used in Gangwani et al. (2018). Also, because the setting is state-less, methods that rely on state distributions to define diversity (Song et al., 2019; Eysenbach et al., 2018; Florensa et al., 2017) are unusable.

5.3. TrajeDi in Hanabi

Finally, we apply algorithm 1 to improve ZSC in Hanabi. We chose this game because it was recently proposed as a challenge in artificial intelligence (Bard et al., 2020), and because it was studied by Hu et al. in the context of ZSC.

In short, a Hanabi deck is composed of 50 cards with ranks ranging from 1 to 5, duplicated in five different colors. The goal is for the players to stack the cards in five decks – one for each color – in ascending rank order. The key mechanic of the game is that each player can see the hands of other players, but never their own. Players must therefore give each other hints to provide enough information for a card to be played, which uses one of the eight available hint tokens. Those hints are regulated, and a player can only ever point out all cards of the same rank or all cards of the same color in another player's hand[3]. Finally, a player can discard one of their card, which recovers one of the hint tokens. We refer interested readers to Bard et al. (2020) for the full rules.

METHOD	SP	XP
OP	**24.24 ± 0.02**	23.65 ± 0.06
OP POOL BR	24.17 ± 0.04	23.66 ± 0.07
OP POOL BR + TRAJEDI	24.22 ± 0.01	**24.09 ± 0.02**

Table 1: SP and XP scores for OP agents in Hanabi, with the last action hidden. Results reported on either 5 individual agents (first row) or 4 BRs, each trained on their own pool (second and third rows). Following the ZSC definition, XP is only computed between agents trained with the same method. Intervals correspond to one standard error.

Hanabi is a highly complex game, even the two-player variant counts approximately 6.2×10^{13} possible initial joint states (Foerster et al., 2019) and between 10 and 20 legal actions depending on whether there are hint tokens left. Assuming players do nothing but discarding one of their five cards each turn, the game ends after 42 actions, when the deck is depleted. Ignoring all other possible actions, as well as the specifics of the deck permutations, we can take $6.2 \times 10^{13} \times 5^{42} \approx 10^{43}$ as a very loose lower bound on the number of possible trajectories.

Besides the sheer size of the setting, this game presents several challenges for successfully using TrajeDi. First, the best performing methods of Hu et al. use off-policy training which requires modifying our gradient estimator accordingly. Secondly, the models are value-based, so we first apply a softmax layer to propagate TrajeDi gradients as for a policy-based method. Thirdly, the replay buffers only track the observations of one agent at a time, hindering our kernel computation in eq. 4. Finally, training is very compute intensive, as it uses 2 GPUs per agent and takes between 48 and 72 hours to converge, depending on the

[3]E.g. "Your first and third cards are blue" or "your last card is of rank 4."

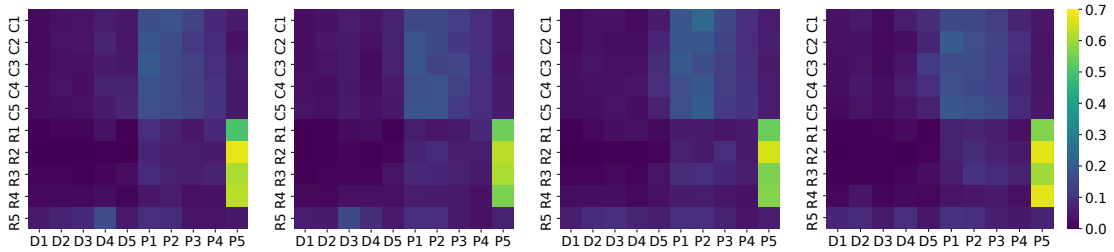


Figure 5: $P(a_{t+1}|a_t)$ matrices for the 4 BR policies to TrajeDi-regularized pools. Shown is a quadrant of each matrix corresponding to the probability of discarding and playing after a hint. D1-D5 and P1-P5 correspond to discarding and playing each of the five cards in hand, while R1-R5 and C1-C5 indicate hinting for one of the five ranks or colors, respectively. Conditional probabilities are computed empirically based on 1000 episodes of SP for each agent. Agents converge to very similar policies, explaining their high XP score when playing together.

method. The computational cost is what limited our ability to run additional seeds and baselines. For a rundown of our implementation, we refer the reader to the appendix.

Taking advantage of these adaptations to the method described in section 4, we train four independent pools of TrajeDi-regularized policies of size 3, complete with their own BRs. We compare against four BRs to unregularized (but otherwise identical) pools as well as five independently trained policies. Following the best approach of Hu et al., all our policies are trained with OP augmented with an auxiliary task consisting in predicting whether a card is playable, discardable or unknown. In addition, we prevent agents from seeing the last action used by the partner, since it can be used as a signaling method (Hu et al., 2020). Because ZSC does not allow players to coordinate on specific modifications to the observation space, hiding the last action counts as a variation upon the game itself. However, it was necessary to reduce the number of possible conventions and therefore keep our pools small enough to be computationally tractable. We also found that it substantially improved the XP scores of the individual agents baseline. In the future, we aim to forgo this modification by training larger pools.

We summarize our results in Table 1. BR policies to TrajeDi-enhanced pools achieve significantly higher XP scores than both BRs to standard pools and individual agents, showing that TrajeDi can indeed help develop more robust strategies for zero-shot coordination, even in such a complex setting. This is done at the cost of a minor drop in SP scores, which affect all BRs and is representative of the difficulty of having four policies converging to the same convention as opposed to a single one in SP. To the best of our knowledge this is the first time that a diversity objective has been shown to help in the ZSC framework and presents a step towards machines that can coordinate well with humans.

Like Hu et al., we illustrate the behaviour of the BR policies produced by our method by showing part of their conditional action matrices in Figure 5. We see that the four policies converge to a very similar behaviour focused around rank hints and playing their fifth card.

6. Conclusion and Future Work

In this work, we first studied the role of diversity in training robust policies for ZSC and leveraged it within a population-based approach. Secondly, we generalized the Jensen-Shannon Divergence over policies to derive TrajeDi, a differentiable objective for training diverse policies. We also derived a gradient estimator and argued that it is particularly well suited for multi-agent settings. Finally, we showed empirically the merit of our contributions in two simple settings and demonstrated their scalability by improving ZSC scores in the complex coordination game Hanabi.

A compelling extension of this work is the use of diversity in competitive settings, where we hypothesize it could improve robustness of a BR agent and mitigate the danger of adversarial attacks against it (Gleave et al., 2019). Moreover, due to the general formulation of the TrajeDi objective, we believe it could have a range of other applications, from HRL to pretraining skills or to exploration in sparse reward settings. Finally, inspired by our results with entropic initialization in section 5.1, we wish to study the impact of initial parameters on the convergence behavior of RL policies.

Acknowledgements

The main author was generously funded by the IVADO for part of this work. We also thank Minqi Jiang and Edward Grefenstette for their valuable feedback on the paper.

References

Bard, N., Foerster, J. N., Chandar, S., Burch, N., Lanctot, M., Song, H. F., Parisotto, E., Dumoulin, V., Moitra, S., Hughes, E., et al. The hanabi challenge: A new frontier for ai research. *Artificial Intelligence*, 280:103216, 2020.

Cohen, A., Qiao, X., Yu, L., Way, E., and Tong, X. Diverse exploration via conjugate policies for policy gradient methods. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 3404–3411, 2019.

Endres, D. M. and Schindelin, J. E. A new metric for probability distributions. *IEEE Transactions on Information theory*, 49(7):1858–1860, 2003.

Eysenbach, B., Gupta, A., Ibarz, J., and Levine, S. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*, 2018.

Florensa, C., Duan, Y., and Abbeel, P. Stochastic neural networks for hierarchical reinforcement learning. *arXiv preprint arXiv:1704.03012*, 2017.

Foerster, J., Song, F., Hughes, E., Burch, N., Dunning, I., Whiteson, S., Botvinick, M., and Bowling, M. Bayesian action decoder for deep multi-agent reinforcement learning. In *International Conference on Machine Learning*, pp. 1942–1951, 2019.

Gangwani, T., Liu, Q., and Peng, J. Learning self-imitating diverse policies. *arXiv preprint arXiv:1805.10309*, 2018.

Gleave, A., Dennis, M., Wild, C., Kant, N., Levine, S., and Russell, S. Adversarial policies: Attacking deep reinforcement learning. *arXiv preprint arXiv:1905.10615*, 2019.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 27*, pp. 2672–2680. Curran Associates, Inc., 2014. URL http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf.

Guo, Y., Oh, J., Singh, S., and Lee, H. Generative adversarial self-imitation learning. *arXiv preprint arXiv:1812.00950*, 2018.

Haarnoja, T., Hartikainen, K., Abbeel, P., and Levine, S. Latent space policies for hierarchical reinforcement learning. *arXiv preprint arXiv:1804.02808*, 2018.

Heinrich, J., Lanctot, M., and Silver, D. Fictitious self-play in extensive-form games. In *International conference on machine learning*, pp. 805–813. PMLR, 2015.

Ho, J. and Ermon, S. Generative adversarial imitation learning. In *Advances in neural information processing systems*, pp. 4565–4573, 2016.

Hong, Z.-W., Shann, T.-Y., Su, S.-Y., Chang, Y.-H., Fu, T.-J., and Lee, C.-Y. Diversity-driven exploration strategy for deep reinforcement learning. In *Advances in Neural Information Processing Systems*, pp. 10489–10500, 2018.

Hu, H., Lerer, A., Peysakhovich, A., and Foerster, J. " other-play" for zero-shot coordination. *arXiv preprint arXiv:2003.02979*, 2020.

Jaderberg, M., Dalibard, V., Osindero, S., Czarnecki, W. M., Donahue, J., Razavi, A., Vinyals, O., Green, T., Dunning, I., Simonyan, K., et al. Population based training of neural networks. *arXiv preprint arXiv:1711.09846*, 2017.

Kober, J. and Peters, J. R. Policy search for motor primitives in robotics. In *Advances in neural information processing systems*, pp. 849–856, 2009.

Kumar, S., Kumar, A., Levine, S., and Finn, C. One solution is not all you need: Few-shot extrapolation via structured maxent rl. *Advances in Neural Information Processing Systems*, 33, 2020.

Parker-Holder, J., Pacchiano, A., Choromanski, K., and Roberts, S. Effective diversity in population-based reinforcement learning. *arXiv preprint arXiv:2002.00632*, 2020.

Song, Y., Wang, J., Lukasiewicz, T., Xu, Z., and Xu, M. Diversity-driven extensible hierarchical reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 4992–4999, 2019.

Tang, Z., Yu, C., Chen, B., Xu, H., Wang, X., Fang, F., Du, S., Wang, Y., and Wu, Y. Discovering diverse multi-agent strategic behavior via reward randomization. *arXiv preprint arXiv:2103.04564*, 2021.

Tesauro, G. Td-gammon, a self-teaching backgammon program, achieves master-level play. *Neural computation*, 6 (2):215–219, 1994.