DGPO: Discovering Multiple Strategies with Diversity-Guided Policy Optimization

Wentse Chen¹, Shiyu Huang², Yuan Chiang³, Tim Pearce⁴, Wei-Wei Tu², Ting Chen³, Jun Zhu³
¹Carnegie Mellon University, Pittsburgh, USA
²4Paradigm Inc., Beijing, China
³Tsinghua University, Beijing, China
⁴Microsoft Research, Cambridge, United Kingdom
wentsec@andrew.cmu.edu, huangsy1314@163.com, yjennice2001@gmail.com, tim.pearce@microsoft.com,
tuweiwei@4paradigm.com, tingchen@tsinghua.edu.cn, dcszj@tsinghua.edu.cn

Abstract

Most reinforcement learning algorithms seek a single optimal strategy that solves a given task. However, it can often be valuable to learn a diverse set of solutions, for instance, to make an agent’s interaction with users more engaging, or improve the robustness of a policy to an unexpected perturbation.

We propose Diversity-Guided Policy Optimization (DGPO), an on-policy algorithm that discovers multiple strategies for solving a given task. Unlike prior work, it achieves this with a shared policy network trained over a single run. Specifically, we design an intrinsic reward based on an information-theoretic diversity objective. Our final objective alternatively constrains on the diversity of the strategies and on the extrinsic reward. We solve the constrained optimization problem by casting it as a probabilistic inference task and use policy iteration to maximize the derived lower bound. Experimental results show that our method efficiently discovers diverse strategies in a wide variety of reinforcement learning tasks. Compared to baseline methods, DGPO achieves comparable rewards, while discovering more diverse strategies, and often with better sample efficiency.

Introduction

Reinforcement Learning (RL) has pioneered breakthroughs in various domains ranging from video games (Vinyals et al. 2019; Berner et al. 2019; Huang et al. 2019a, 2021) to robotics (Raffin et al. 2018; Yu et al. 2021b). While its achievements are remarkable, RL is not devoid of challenges. A paramount issue is the innate pursuit of RL algorithms for a singular optimal solution, even when a myriad of equally viable strategies exists. This tunnel vision for optimization can inadvertently introduce weaknesses.

For instance, RL algorithms are known for “overfitting” tasks. By zeroing in on just one strategy, they often miss out on exploring a wealth of high-quality alternative solutions. This over-specialization renders the agent vulnerable to unpredictable environmental changes, as it lacks the robustness multiple strategies could have offered (Kumar et al. 2020). In competitive arenas, predictability can be an Achilles heel, with adversaries exploiting the agent’s inflexibility. A diversified approach would obfuscate the agent’s strategies, improving its competitive edge (Lanctot et al. 2017). Furthermore, in domains like dialogue systems, monotony can dull user interactions, whereas varied responses could significantly enhance user experience (Li et al. 2016; Gao et al. 2019: Pavel, Budulan, and Rebedea 2020; Xu et al. 2022; Chow et al. 2022).

For instance, RL algorithms are known for “overfitting” tasks. By zeroing in on just one strategy, they often miss out on exploring a wealth of high-quality alternative solutions. This over-specialization renders the agent vulnerable to unpredictable environmental changes, as it lacks the robustness multiple strategies could have offered (Kumar et al. 2020). In competitive arenas, predictability can be an Achilles heel, with adversaries exploiting the agent’s inflexibility. A diversified approach would obfuscate the agent’s strategies, improving its competitive edge (Lanctot et al. 2017). Furthermore, in domains like dialogue systems, monotony can dull user interactions, whereas varied responses could significantly enhance user experience (Li et al. 2016; Gao et al. 2019; Pavel, Budulan, and Rebedea 2020; Xu et al. 2022; Chow et al. 2022).

We identify two key scenarios where multiple strategies are beneficial: 1. The margin for error is inconsequential to the task’s success. In such cases, agents can operate optimally without adhering strictly to the best strategy, enabling a spread of near-optimal strategies (Zahavy et al. 2021). 2. The task inherently allows multiple optimal solutions, such as a maze offering two equally efficient paths (Osa, Tangkaratt, and Sugiyama 2021; Zhou et al. 2022).

Crafting an algorithm that harnesses diverse high-reward solutions efficiently is intricate. With the Diversity-Guided Policy Optimization (DGPO) we propose, we aim to address several requisites for such an algorithm:

Strategy Representation: the diversity of a policy suite is evaluated in DGPO using a metric grounded on the mutual information between states and the latent variable, implemented through a learned discriminator.

Diversity Evaluation: The diversity of a policy suite is evaluated in DGPO using a metric grounded on the mutual information between states and the latent variable, implemented through a learned discriminator.

Diversity Exploration: DGPO embarks on exploration that encourages deviation from familiar strategies while
safeguarding performance. This is achieved using a constrained optimization method that harmonizes performance and diversity.

**Sample Efficiency**: Unlike predecessors like RSPO (Zhou et al. 2022) and RPG (Tang et al. 2021), which necessitated multiple networks and training phases, DGPO employs a shared network for concurrent learning of strategies, resulting in superior sample efficiency.

In encapsulation, this work makes three pivotal contributions: (1) We introduce a structured approach to discover diverse high-reward policies by framing it as two constrained optimization problems, coupled with tailored diversity rewards to guide the policy learning. (2) DGPO, a novel on-policy algorithm, is unveiled, designed to seamlessly uncover a diverse suite of high-quality strategies. (3) Our empirical evaluations elucidate that DGPO not only holds its own against benchmarks but frequently surpasses them in terms of diversity, performance, and sample efficiency.

**Related Works**

In this section, we provide an overview of prior research that encompasses two main aspects: the representation of reinforcement learning (RL) as a probabilistic graphical model (PGM), and the explicit integration of diversity learning with RL.

**Reinforcement Learning as Probabilistic Graphical Model**

PGM’s have proven to be a useful way of framing the RL problem (Ziebart et al. 2008; Furmston and Barber 2010; Levine 2018). The soft actor-critic (SAC) algorithm (Haarnoja et al. 2018b) formalizes RL as probabilistic inference and maximizes an evidence lower bound by adding an entropy term to the training objective, encouraging exploration. PGM’s also serve as useful tools for studying partially observable Markov decision processes (POMDP’s) (Igl et al. 2018; Huang et al. 2019b; Lee et al. 2019). Haarnoja et al. (Haarnoja et al. 2018a) use PGM’s to construct a hierarchical reinforcement learning algorithm. Hausman et al. (Hausman et al. 2018) optimize a multi-task policy through a variational bound, allowing for the discovery of multiple solutions with a minimum number of distinct skills. In this work, we modify the PGM of a Markov decision process (MDP) by introducing a latent variable to induce diversity into the MDP. We then derive the evidence lower bound of the new PGM, allowing us to construct a novel RL algorithm.

**Diversity in Reinforcement Learning**

Achieving diversity has been studied in various contexts in RL. (Mouret and Doncieux 2009; Mohamed and Rezende 2015; Eysenbach et al. 2018; Osa, Tangkaratt, and Sugiyama 2021; Derek and Isola 2021). Eysenbach et al. (Eysenbach et al. 2018) proposed DIAYN to maximize the mutual information between states and skills, which results in a maximum entropy policy. Osa et al. (Osa, Tangkaratt, and Sugiyama 2021) proposed a method that can learn infinitely many solutions by training a policy conditioned on a continuous or discrete low-dimensional latent variable. Their method can learn diverse solutions in continuous control tasks via variational information maximization. There is also a growing corpus of work on diversity in multi-agent reinforcement learning (Mahajan et al. 2019; Lee, Yang, and Lim 2019; He, Shao, and Ji 2020). Mahajan et al. (Mahajan et al. 2019) proposed MAVEN, a method that overcomes the detrimental effects of QMIX’s (Rashid et al. 2018) monotonicity constraint on exploration by maximizing the mutual information between latent variables and trajectories. However, their method can not find multiple diverse strategies for a specified task. He et al. (He, Shao, and Ji 2020) investigated multi-agent algorithms for learning diverse skills using information bottlenecks with unsupervised rewards. However, their method operates in an unsupervised manner, without external rewards. More recently, RSPO (Zhou et al. 2022) was proposed to derive diverse strategies. However, it requires multiple training stages, which results in poor sample efficiency – our method trains diverse strategies simultaneously which reduces sample complexity.

**Preliminaries**

RL can be formalized as an MDP. An MDP is a tuple \((S, A, P, r, \gamma)\), where \(S\) and \(A\) represent state and action space respectively, \(P(s, a) : S \times A \rightarrow S\) is the transition probability density, \(r(s, a) : S \times A \rightarrow \mathbb{R}\) is a reward function, and \(\gamma \in [0, 1]\) is the discount factor.

**Latent conditioned policy**: We consider a policy \(\pi_{\theta}\) that is conditioned on latent variable \(z\) to model diverse strategies, where \(\theta\) represents policy parameters. For compactness, we will omit \(\theta\) in our notation. We denote the latent-conditioned policy as \(\pi(a|s, z)\) and a latent conditioned critic network as \(V^\pi(s, z)\). For each episode, a single latent variable is sampled, \(z \sim p(z)\) from a categorical distribution with \(n_z\) categories. In our work, we choose \(p(z)\) to be a uniform distribution. The agent then conditions on this latent code \(z\), to produce a trajectory \(s_t\).

**Discounted state occupancy**: The discounted state occupancy measure for policy \(\pi\) is defined as \(\rho^\pi(s) = (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t P_t^\pi(s)\), where \(P_t^\pi(s)\) is the probability that policy \(\pi\) visit state \(s\) at time \(t\). The goal of the RL agent is to...
train a policy \( \pi \) to maximize the discounted accumulated reward \( J(\theta) = \mathbb{E}_{z \sim p(z), \tau \sim \rho^\pi(s)|z} \left[ \sum_t \gamma^t r(s_t, a_t) \right] \).

**RL as probabilistic graphical model:** An MDP can be framed as a probabilistic graphical model as shown in Fig. 1(a) and the optimal control problem can be solved as a probabilistic inference task (Levine 2018). In this paper, we propose a new probabilistic graphical model, denoted as the diverse MDP, as shown in Fig. 1(b). We introduce a binary random variable \( O_t \) and an integer random variable \( z \) into the model. \( O_t = 1 \) denotes that action \( a_t \) is optimal at time step \( t \) and \( O_t = 0 \) denotes it is not. Previous work (Levine 2018) has defined, \( p(O_t = 1|s_t, a_t, z) \propto \exp(r(s_t, a_t)) \). The evidence lower bound (ELBO) is given by:

\[
\log p(O_{1:T}) \\
\geq \mathbb{E}_{\tau \sim D_\tau} [\log p(O_{1:T} = 1|s_{1:T}, a_{1:T}, z)] \\
- \log p(a_{1:T} = 1|s_{1:T}, z) \\
= \mathbb{E}_{\tau \sim D_\tau} [\log p(O_{1:T} = 1|s_{1:T}, a_{1:T}, z)] \\
+ \log p(z|s_t, a_t) - \log p(a_t|s_t, z),
\]

where the trajectory \( \tau = \{a_{1:T} = 1|s_{1:T}, z\} \) is sampled from a trajectory dataset \( D_\tau \). The proof of Eq. 1 can be found in Appendix C. Note that the introduction of \( z \) gives rise to the term \( p(z|s_t, a_t) \), which represents how identifiable the latent code is from the current policy behavior. This will be a crucial ingredient of DGPO, guiding the policy to explore and discover a set of diverse strategies.

**Methodology**

In this section, we will introduce our algorithm in detail. Our algorithm can be divided into two stages. In the first stage, we will focus on improving the agent’s performance while maintaining its diversity. In the second stage, we will focus more on enhancing diverse strategies. Finally, we will introduce our final algorithm, denoted as Diversity-Guided Policy Optimization (DGPO), that unifies the two-stage training processes and also the implementation details.

**Diversity Measurement**

In this section, we present a diversity score capable of evaluating the diversity of a given set of policies. We then derive a diversity objective from this score to facilitate exploration. Eysenbach et al. (Eysenbach et al. 2018) proposed a diversity score based on mutual information between states and latent codes \( z \),

\[
I(s; z) = \mathbb{E}_{z \sim p(z), s \sim \rho^\pi(s)} [\log p(z|s) - \log p(z)],
\]

where \( I(\cdot; \cdot) \) stands for mutual information. As we can not directly calculate \( p(z|s) \), we approximate it with a learned discriminator \( q_\phi(z|s) \) and derive the ELBO as \( \mathbb{E}_{z \sim p(z), s \sim \rho^\pi(s)} [\log q_\phi(z|s) - \log p(z)] \), where \( \phi \) are the parameters of the discriminator network.

Mutual information is equal to the KL distance between the state marginal distribution of one policy and the average state marginal distribution, i.e., \( I(s; z) = \mathbb{E}_{z \sim p(z)} [D_{KL}(\rho^\pi(s|z)||\rho^\pi(s))] \) (Eysenbach, Salakhutdinov, and Levine 2021). This means that \( I(s; z) \) captures the diversity averaged over the whole set of policies. In DGPO, we wish to ensure that any given pair of strategies is different, rather than on average. As such, we define a novel, stricter diversity score,

\[
\text{DIV}(\pi_\theta) = \mathbb{E}_{z \sim p(z)} \min_{z' \neq z} D_{KL}(\rho^\pi(s|z)||\rho^\pi(s|z')) \quad (3)
\]

Instead of comparing policy with the average state marginal distribution, we compare it with the nearest policy in \( \rho^\pi(s) \) space. In this way, setting \( \text{DIV}(\pi_\theta) \geq \delta \) means that each pair of policies have at least \( \delta \) distance in terms of expectation. In order to optimize Eq. 3, we first derive a lower bound,

\[
\text{DIV}(\pi_\theta) \geq \mathbb{E}_{z, s} \min_{z' \neq z} \log \frac{p(z|s)}{p(z|s) + p(z'|s)}.
\]

The proof is given in Appendix D. To maximize this lower bound, we first assume we can learn a latent code discriminator, \( q_\phi(z|s) \), to approximate \( p(z|s) \). We then define an intrinsic reward,

\[
r_t^{in} = \min_{z' \neq z} \log \frac{q_\phi(z|s_{t+1})}{q_\phi(z'|s_{t+1}) + q_\phi(z'|s_{t+1})}. \quad (5)
\]

This allows us to define our final diversity objective,

\[
J_{\text{Div}}(\theta) = \mathbb{E}_{z \sim p(z), s \sim \rho^\pi(s), a \sim \pi|s, z} \left[ \sum_t \gamma^t r_t^{in} \right]. \quad (6)
\]

A straightforward way to incorporate the diversity metric into a PGM is defining elements in Eq. 1 as,

\[
p(O_{1:T} = 1|s_t, a_t, z) = \exp \left( r(s_t, a_t) \right),
\]

\[
p(z|s_t, a_t) = \exp \left( r_t^{in} \right).
\]

However, the simple combination of extrinsic and intrinsic rewards may lead to poor performance. In the following paragraph, we formulate the algorithm as constrained optimization problems and mask elements in Eq. 7 based on constraints to guide the policy to explore.

**Stage 1: Diversity-Constrained Optimization**

Strategies that solve a given RL task may be very distinct. One can think of these as a set of discrete points in \( \rho^\pi(s) \) space – if the distance between the points is large, perturbing around one single solution may not allow the discovery of all optimal strategies. Thus, we formulate the policy optimization process as a diversity-constrained optimization problem,

\[
\max_{\pi_\theta} J(\theta), \quad \text{s.t.} \quad J_{\text{Div}}(\theta) \geq \delta. \quad (8)
\]

Under this objective, individual policies are constrained to keep a certain distance \( \delta \) apart from each other, and systematically explore their own regions. We can introduce a Lagrange multiplier \( \lambda \) to tackle this constrained optimization problem,

\[
\max_{\pi_\theta} \min_{\lambda \geq 0} J(\theta) + \lambda (J_{\text{Div}}(\theta) - \delta) \geq \max_{\pi_\theta} \mathbb{E}_{z \sim p(z), s \sim \rho^\pi(s)} \min_{\lambda \geq 0} \mathbb{E}_{a \sim \pi(s, z)} \left[ \sum_t \gamma^t r_t^{in} \right] + \lambda \left( \sum_t \gamma^t r_t^{in} - \delta \right).
\]

11392
Figure 2: The overall framework of the DGPO algorithm. The top illustrates the way of calculating the intrinsic rewards objective, which can be written, 
\[ r_{\text{in}}^t \] can be defined as: 
\[ p(\mathcal{O}_t = 1|s_t, a_t, z) = \exp(r(s_t, a_t)) \]
\[ p(z|s_t, a_t) = \exp(1 - I[J_{\text{Div}}(\theta) \geq \delta]) \cdot r_{\text{in}}^t \]

Stage 2: Extrinsic-Reward-Constrained Optimization

The objective developed in the previous section can return a set of discrete optimal points. However, sometimes two strategies may still converge to the same sub-optimal point. This can destabilize the training process since both strategies may still converge to the same sub-optimal point. They are allowed to find their destiny in the neighborhood of the optimal point to further maximize the level of diversity. On the other hand, for those policies that are already sufficiently distinct, the diversity objective serves as an intrinsic reward to encourage exploration. Similar to how we deal with diversity-constrained optimization, we implement Eq. 11 by injecting an extrinsic-rewards-constraint into the framework. The updated elements in Eq. 1 can be defined as:

\[ p(\mathcal{O}_t = 1|s_t, a_t, z) = \exp(r(s_t, a_t)) \]
\[ p(z|s_t, a_t) = \exp[I[J(\theta) \geq R_{\text{target}}] \cdot r_{\text{in}}^t] \]

Diversity-Guided Policy Optimization

In this section, we will introduce our final algorithm which unifies the two-stage training processes into one unified algorithm. We develop a new variation of PPO (Schulman et al. 2017) by considering policy network and critic network that are conditioned on latent variable \( z \), i.e., \( \pi(a_t|s_t, z) \). The critic network is divided into two parts, i.e., \( V_{\psi_{\text{ex}}}^\pi(o_{1:t}, z) \) and \( V_{\psi_{\text{in}}}^\pi(o_{1:t}, z) \), where \( \psi_{\text{ex}} \) and \( \psi_{\text{in}} \) are their parameters. The parameters of critic networks can be trained by a temporal difference (TD) loss (Sutton and Barto 2018):

\[ L(\psi_{\text{ex}}) = MSE\left(V_{\psi_{\text{ex}}}^\pi(o_{1:t}, z)\right) \]
\[ L(\psi_{\text{in}}) = MSE\left(V_{\psi_{\text{in}}}^\pi(o_{1:t}, z), \sum_{t'=t}^{\infty} \gamma^{t'-t} \cdot I[J_{\text{Div}}(\theta) \geq \delta] \right) \]

where \( L(\cdot) \) stands for loss function and \( MSE(\cdot) \) stands for mean square error. We maintain a running average of \( V_{\psi_{\text{ex}}}^\pi(s_t, z) \) to approximate \( \mathbb{E}_{s \sim \rho(s), a \sim \pi(\cdot|s)} \sum_t \gamma^t r(s_t, a_t) \). We also construct a
discriminator $q_{\phi}(z|s_t)$ that takes the state as input and predict the probability of latent variable $z$, where $\phi$ is the parameter of the discriminator network. And the discriminator can be trained in a supervised manner:

$$L(\phi) = \mathbb{E}_{(s_t, a_t, z) \sim D_{\pi}}[CE(q_{\phi}(s_t), z)],$$

where $CE(\cdot, \cdot)$ stands for cross entropy loss. We implement DGPO by incorporating diversity-constrained optimization and extrinsic-reward-constrained optimization into the same framework, i.e., the total value of the state and the total reward can be defined as below:

$$V_{\pi,t}^{\text{target}}(s_t, z) = V_{\psi_{\text{in}}}(s_t, z) + V_{\psi_{\text{rec}}}(s_t, z),$$

$$r_{t}^{\text{total}} = \mathbb{I}[J_{\text{Div}}(\theta) \geq \delta] r(s_t, a_t) + [1 - \mathbb{I}[J_{\text{Div}}(\theta) \geq \delta]] + \mathbb{I}[J(\theta) \geq R_{\text{target}}] r_{t}^{\text{in}}.$$ (15)

In theory, it is not feasible to simultaneously conduct diversity-constrained optimization and extrinsic-reward-constrained optimization. As a result, the aforementioned implementation serves as an approximation to the original objective. We posit that this approximation is reasonable, as the empirical findings demonstrate that the two training stages are not concurrent. Fig. 2 shows the overall framework of the DGPO algorithm. The detailed training process of DGPO can be found in the Appendix G.

### Experiments

In this section, we evaluate our algorithm on several RL benchmarks — Multi-agent Particle Environment (MPE) (Mordatch and Abbeel 2018), StarCraft Multi-Agent Challenge (SMAC) (Samvelyan et al. 2019), and Atari (Bellemare et al. 2013). We compare our algorithm to four baseline algorithms:

- **MAPPO** (Yu et al. 2021a): MAPPO adapts the single-agent PPO (Schulman et al. 2017) algorithm to the multi-agent setting by using a centralized value function with shared team-based rewards.
- **DIAYN** (Eysenbach et al. 2018): DIAYN trains agents with a mutual-information based intrinsic reward to discover a diverse set of skills. In our setup, these intrinsic rewards are combined with extrinsic rewards.
- **SMERL** (Kumar et al. 2020): SMERL maximizes a weighted combination of intrinsic rewards and extrinsic rewards when the return of extrinsic reward is greater than some given threshold.
- **RSPO** (Zhou et al. 2022): RSPO is an iterative algorithm for discovering a diverse set of quality strategies. It toggles between extrinsic and intrinsic rewards based on a trajectory-based novelty measurement.

As far as possible, all methods use the same hyperparameters. However, there is some difference for RSPO, for which we use the open-source implementation. (Full hyperparameters are listed in the Appendix B.) All experiments were performed on a machine with 128 GB RAM, one 32-core CPU, and one GeForce RTX 3090 GPU.

### Multi-Agent Particle Environment

We evaluate on two scenarios shown per Fig. 5, Spread (easy) and Spread (hard). In Spread (easy), there are four landmarks and one agent. The agent starts from the center and aims to reach one of the landmarks, giving four optimal solutions. In Spread (hard), there are three agents and three landmarks. Agents cooperate to cover all the landmarks and avoid colliding with others, giving two optimal solutions. Model weights are shared across agents.

Similar to (Parker-Holder et al. 2020), we introduce a metric to quantitatively evaluate the diversity score of the given set of policies, $\Pi$: $M_{\text{Div}}(\Pi) = \frac{1}{n_{\pi}} \sum_{i=1}^{n_{\pi}} \sum_{j=i+1}^{n_{\pi}} \ln(\|\Phi(\pi_i) - \Phi(\pi_j)\|_2)$, where $\Phi(\pi)$ is the behavior embedding of the policy $\pi$. For MPE, $\Phi(\pi)$ is represented by the concatenated agents’ positions over an episode.

We set $n_z = 4$ in Spread (easy) and $n_z = 2$ in Spread (hard) to test whether an algorithm can discover all optimal
Figure 4: Plots showing extrinsic reward performance vs. the diversity of the set of discovered strategies. (a) In two Atari games. (b) In two SMAC scenarios.

Figure 5: The initial state of Spread (easy) and Spread (hard). In both scenarios, Agents (orange dots) aim to reach one of the destinies (blue dots). We highlight the optimal solutions with arrows of different colors.

Figure 6: Screen shots and heat maps of agents’ trajectories on Pong and Boxing. (a) In the Pong game, our agent controls the paddle (green block) to hit the ball (white dot). (b) In the Boxing game, our agent (in white) has a boxing match with the opponent (in black).

Atari

We evaluate DGPO on the Atari games Pong and Boxing, to test the performance for tasks with image observations. DGPO’s diversity metric is defined similarly to the one used in MPE (full details in Appendix F). In each environment, we set \( n_z = 2 \). Results are summarized in Fig. 4(a), averaged over five seeds. We also reported the experimental results for \( n_z = 10 \) in Appendix I. DGPO again delivers a favorable trade-off between external reward and strategy diversity. Fig. 6(a) visualizes the results of the two strategies obtained by DGPO on Pong. In this game, the agent controls the green paddle. When \( z = 0 \) the agent holds the paddle at the bottom of the screen until the ball is near, while when \( z = 1 \), the default position of the paddle is at the top of the screen. Fig. 6(b) similarly shows the different strategies obtained by DGPO on Boxing, where the agent controls the white character, and is rewarded for punching the black opponent. The heatmap shows DGPO learns to attack from different sides. Other baseline algorithms tend to remain in a single corner.
StarCraft II

We conduct experiments on two StarCraft II maps (2s,vs, Jsc & 3m) from SMAC. We set \( n_z = 3 \) and measure the mean win rates over five seeds. Fig. 4(b) shows that, relative to other algorithms, DGPO discovers sets of strategies that are both diverse and achieve good win rates. Fig. 7 visualizes three strategies obtained by DGPO on 3m map. In this map, we control the three green agents to combat the red built-in agents. We visualize the trajectories of our agents with green arrows. When \( z = 0 \), the policy produces an aggressive strategy, with agents moving directly forward to attack the enemies. When \( z = 1 \), the agents display a kiting strategy, alternating between attacking and moving. This allows them to attack enemies while limiting the taken damage. When \( z = 2 \), the policy produces another kiting strategy but now with a downwards, rather than upwards drift.

Ablation Study

We performed ablation studies on MPE Spread (hard) tasks, systematically removing each element of our algorithm to assess its impact on diversity. The empirical result is shown in Fig. 8(a). Throughout this section, we set \( n_z = 3 \) and the result is averaged over 5 seeds. Change-Diversity-Measurement uses mutual information as shown in Eq. 2 as diversity metric. Experimental results indicate that in the Spread (hard) scenario, the diversity score of Change-Diversity-Measurement is lower than that of DGPO. While optimizing three policies simultaneously, the limited number of optimal solutions (only two) leads to one policy behaving differently while the other two exhibit similar behavior. Consequently, the diversity score based on mutual information becomes artificially high (as it reflects the overall diversity level of the policy set), causing the policy to stop optimizing diversity, even though two policies continue to behave similarly. No-Diversity-Constrained-Optimization excludes diversity-constrained optimization. Empirical results reveal that it only identifies one optimal solution. This suggests that utilizing the diversity metric as a constraint, rather than blending it with extrinsic rewards during the initial stages of training, significantly enhances the algorithm’s effectiveness. No-Extrinsic-Reward-Constrained-Optimization omits extrinsic-reward-constrained optimization. While DGPO continues to enhance the diversity score as the expected return surpasses \( R_{\text{target}} \), No-Extrinsic-Reward-Constrained-Optimization fails to do so. Consequently, it ultimately attains a lower diversity score compared to DGPO. Our algorithm can be divided into two distinct stages, as depicted in Fig. 8(b). In the initial stage, we focus on diversity-constrained optimization until policies exhibit sufficient behavioral differences. Once the expected return reaches \( R_{\text{target}} \), we transition to the extrinsic-reward-constrained optimization stage. Here, we maintain a fixed performance level at \( R_{\text{target}} \) while maximizing the diversity score. These stages are clearly separated and non-overlapping.

Conclusions

In conclusion, this paper introduced the Diversity-Guided Policy Optimization (DGPO) algorithm, which demonstrates its capability to efficiently uncover diverse strategies that yield high rewards. By framing the training process as a pair of constrained optimization problems and solving them through probabilistic inference, DGPO stands out as a promising on-policy algorithm. Through experiments, we observed that DGPO strikes a favorable balance between diversity scores and rewards, all while exhibiting improved sample efficiency. Moving forward, we envision delving deeper into its potential to handle challenges such as exploration, self-play, and robustness.


