LEARNING DIVERSE AND EFFECTIVE POLICIES WITH NON-MARKOVIAN REWARDS

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning a set of diverse and high-quantity policies is a difficult problem in Reinforcement Learning since the diversity of policies is demanded to be achieved without dampening their effectiveness. This problem becomes more challenging when the rewards are non-Markovian, i.e., the rewards depend on the history of states and actions, which are quite sparse and returned over a long period. The sparse supervision signals and the non-Markovian properties of the rewards hinder the learning of policy embeddings and thus the learning of diverse and high-quality policies. In this paper, we propose to use a diversity matrix to quantify policy diversity and theoretically prove that if the diversity matrix is positive definite, then the diversity of policies can be achieved without sacrificing their effectiveness. The policy diversity matrix stems from policy embeddings. To obtain high-quality embeddings, we adopt a transformer to capture mutual dependencies between states and actions and design pseudo tasks to overcome sparse rewards. Experimental results show that our method can achieve a set of policies with more effective diversity and better performance than multiple recently proposed baseline methods in a variety of non-Markovian and Markovian environments.

1 INTRODUCTION

Reinforcement learning (RL) has achieved great success in learning an efficient policy for a given task (Mnih et al., 2015; Schulman et al., 2017; 2015; Lillicrap et al., 2015; Fujimoto et al., 2018; Haarnoja et al., 2018). To achieve better exploration and adaptation to complex environments, a diverse set of high-quality policies are required in many real-world scenarios with non-Markovian rewards. For example, an autonomous taxi traveling from the passenger zone to the destination should learn diverse routes in the case that a regular path is blocked, and the taxi can get a positive reward only when taking the passenger to the correct destination. We usually need to learn diverse policies and ensure their diversity does not dampen their effectiveness. This problem becomes becomes more challenging when the rewards are non-Markovian, in which each reward is returned over a long period and depends on the history of states and actions (Camacho et al., 2017; Gaon & Brafman, 2020; Brafman et al., 2018).

Some prior works are developed to learn diverse policies. A series of existing Quality-Diversity (QD) (Pugh et al., 2016) related evolutionary algorithms have achieved good performance in exploring diverse behaviors and diverse policies (Cully & Demiris, 2017; Gangwani et al., 2020; Hong et al., 2018; Masood & Doshi-Velez, 2019; Peng et al., 2020; Zhang et al., 2019; Conti et al., 2017). Even though the policies achieve diverse behaviors, different policies will frequently switch between existing behaviors during the policy update process, which degrades the performance of policies. The MAP-Elites algorithms (Cully et al., 2015; Mouret & Clune, 2015) solve this problem by discretizing the behavior description space into a grid of cells. Nonetheless, in this method, the gird of cells needs to be pre-defined. To break this limitation, policy embedding-based methods are developed in the literature. The P3S (Jung et al., 2019) algorithm embeds policy behaviors and searches an area in the policy space guided by the best policy to obtain diverse solutions. The DvD (Parker-Holder et al., 2020b) algorithm pre-defines the behavioral embeddings as a collection of actions and uses a kernel function to transform the behavioral embeddings into a matrix for diversity policy learning. However, when the reward is non-Markovian, it is challenging to learn the policy embeddings since the non-Markovian rewards depend on the history of states and actions and provide quite sparse

supervision signals, and in the above methods, the absence of the history of states and actions can result in ineffective policy embeddings.

Some theoretical conclusions are also developed to ensure that diverse policies are not obtained by sacrificing their effectiveness. The DvD (Parker-Holder et al., 2020b) algorithm proves that under tabular MDP, multiple distinct optimal solutions can be obtained by maximizing the proposed loss function. The ridge rider algorithm (Parker-Holder et al., 2020a) proposes to use the eigenvectors of the Hessian matrix to discover diverse local optima with theoretical guarantees. Nieves et al. (2021) theoretically shows that maximizing the diversity metric based on the decision point process can guarantee to enlarge the convex polytopes spanned by the policies of agents. However, these works cannot be used in our settings, the theory in Nieves et al. (2021) is for two-player zero-sum games, which is not the concern of this paper, and the theory in Parker-Holder et al. (2020a) assumes the Hessian matrix can be accurately estimated, which is challenging when the rewards are sparse. The embeddings and the proposed theory in Parker-Holder et al. (2020b) are only for the tabular Markov Decision Process.

A natural question emerges: How to learn policy embeddings in environments with non-Markovian rewards and ensure the learned diverse policies are effective?

In the paper, we propose a novel method called **D**iverse Policy Learning via **D**iversity **M**atrix (DDM), which can efficiently find high-quality policies with diverse behaviors in non-Markovian reward environments. Specifically, to learn high-quality embeddings, we use a Transformer Encoder (Vaswani et al., 2017) to capture mutual dependencies between actions and states. Moreover, to overcome the challenge of sparse rewards, we design pseudo tasks to train the Transformer Encoder, and the obtained embeddings are further employed to construct a *diversity matrix* to measure the degree of policy diversity. We also give the theoretical conditions such that the policy diversity will not reduce their effectiveness.

Our contributions are summarized as follows: (1)We develop a Transformer-based policy representation model, which is capable of learning policy embeddings in non-Markovian reward environments. To overcome the sparse rewards, we design pseudo tasks to train the policy representation model. (2) We propose a diversity matrix to measure the degree of policy diversity and prove that if the diversity matrix is positive definite, then the diversity of policies can be achieved without sacrificing their effectiveness. (3) Our policy diversity learning scheme can be combined with any off-policy reinforcement learning algorithm, and experimental results in a variety of environments show it outperforms multiple recently proposed baselines in both non-Markovian and Markovian environments.

2 PRELIMINARIES

In this section, we present the necessary background relevant to the problem setting of this work.

Markov Decision Process (MDP) is a mathematical framework to describe an environment in reinforcement learning. MDP is a tuple $(S, A, P, R, \rho, \gamma)$, where S is the state space, A is the action space. The state transition dynamic function is given by $P : S \times A \to S$, mapping from the current state $s \in S$ to the next state $s' \in S$. The reward function is given by $R : S \times A \to \mathbb{R}$, mapping from state $s \in S$ and action $a \in A$ to $\mathbb{R}$. $\gamma \in [0, 1)$ denotes the discount factor. Policy π is a mapping from S to A, and the objective of the policy is to maximize the expected cumulative reward $R(\tau) = \sum_{t=1}^{T} \gamma^{t-1} r_t$. In deep reinforcement learning, policy π is typically a neural network, encoded by parameter vectors θ, and the objective function of policy π is as follows:

$$J(\pi) = \mathbb{E}_{\tau \sim \pi}[R(\tau)]. \tag{1}$$

Non-Markovian Reward Decision Processes (Bacchus et al., 1996; Gaon & Brafman, 2020)(NMRDPs) extend MDPs and assume each reward depends on the history of states and actions. Specifically, the Non-Markovian Reward (NMR) function R is a mapping from the finite history of states and actions to $\mathbb{R}$, denoted as $(S \times A)^* \to \mathbb{R}$.

Here we use $\mathrm{Div}(\{\pi_{\mathrm{m}}\}_{\mathrm{m}=1}^{\mathrm{M}})$ to denote the diversity of M policies and its specific form will be discussed in Section 3.3. In this paper, for each π_m ($m \in \{1, ..., M\}$), we aim at maximizing

$$\tilde{J}(\pi_m) = (1 - \beta) J(\pi_m) + \beta \mathrm{Div}(\{\pi_{\mathrm{m}}\}_{\mathrm{m}=1}^{\mathrm{M}}), \tag{2}$$

where $\tilde{J}(\pi_m)$ is an objective function of policy π_m, and $\beta \in (0, 1)$ controls the trade-off between $J(\pi_m)$ and $\mathrm{Div}(\{\pi_\mathrm{m}\}_{\mathrm{m}=1}^{\mathrm{M}})$. However, the effectiveness of policy π_m in collecting rewards may be dampened when maximizing the objective in Eq. (2). To overcome this challenge, we propose Theorem 3.1 to justify that the objective in Eq. (2) can only be maximized when all policies $\{\pi_m\}_{m=1}^{M}$ are optimal.

3 METHODOLOGY

In this paper, we aim to learn the policies satisfying the following two requirements: diversity and effectiveness. However, when rewards are non-Markovian, it is difficult to obtain effective policy embeddings since the rewards depend on the history of states and actions and the supervision signals are quite sparse. Moreover, it is also a challenge to ensure the diversity of effective policies. In the following, we shall provide the method to overcome these two challenges.

3.1 OVERVIEW

Our proposed method, DDM (Diverse Policy Learning via Diversity Matrix), comprises two primary submodules: the policy representation module and the policy diversity qualification module, which are developed to answer the question in Introduction 1. The schematic illustration of our method is shown in Fig. 1, and the pseudo code is given in Appendix A.1.

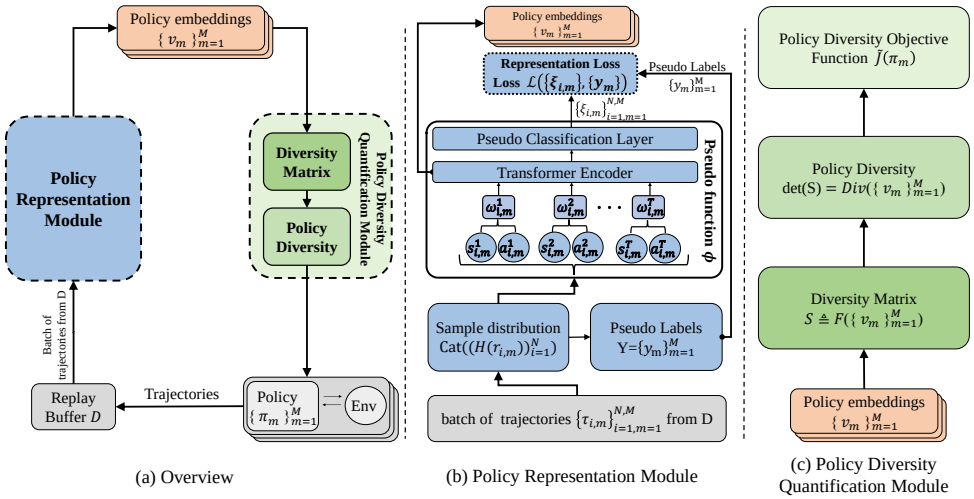

Figure 1: **A schematic illustration of DDM: (a)** The overall structure of DDM. DDM comprises the policy representation module and the policy diversity quantification module. M learners execute parallelly and store trajectories into replay buffer D. **(b)** The policy representation module samples a batch of trajectories from replay buffer D and then obtains the sample distribution using Eq. (3). A pseudo task is proposed, and the pseudo function ϕ is trained using sampled trajectories and their pseudo labels Y by minimizing representation loss $\mathcal{L}$ (Eq. (7)). Outputs are policy embeddings $\{v_m\}_{m=1}^{M}$. **(c)** The policy diversity quantification module utilizes policy embeddings to construct a diversity matrix. The determinant of the diversity matrix is regarded as policy diversity, which is further included in the policy diversity objective $\tilde{J}(\pi_m)$ (Eq. (8)). We prove this policy diversity objective can only be maximized when all policies are optimized.

The representation module is shown in Fig. 1(b), which is utilized to obtain effective policy embeddings. Non-Markovian rewards are returned over a long period and thus are quite sparse. Besides, they depend on the history of actions and states, not only the last ones. A Transformer encoder is adopted to learn effective policy embeddings by capturing mutual dependencies between states and actions. We design pseudo tasks to solve the lack of supervision signals for the Transformer encoder. For the m-th ($m \in \{1, ..., M\}$) policy π_m, the representation module inputs a state-action trajectory collected by π_m and outputs a policy embedding vector v_m for π_m.

The policy diversity quantification module is shown in Fig. 1(c), which aims to construct a diverse matrix using the policy embeddings. It is challenging to ensure that policy diversity can be achieved without sacrificing their performance. We prove that our method based on diversity matrix can overcome this challenge by: 1) utilizing the policy embeddings to construct positive definite diversity matrix and 2) adding the determinant of the diversity matrix as a term to the objective function.

In our method, M parallel learners learn M distinct policies and share a common replay buffer D. The M learners execute parallelly in different copies of the same environment and employ a common base algorithm which can be any off-policy RL algorithm.

3.2 Policy Representation Module

The policy representation module is employed to generate effective and diverse embeddings for different policies. However, in the non-Markovian scenarios, each reward depends not only on the current state and action but also on the history of states and actions, and thus we use a Transformer to capture this long-horizon dependencies. To train the policy representation module with sparse supervision signals, we design a self-defined pseudo task and use the policy indexes as the pseudo labels. Moreover, we use sparse rewards as the criterion to select training trajectories.

3.2.1 Pseudo task

The non-Markovian rewards are quite sparse, which hinders the learning of policy embeddings directly using states, actions, and rewards. Moreover, we require the learned embeddings to be diverse and effective, i.e., the embeddings for different policies are distinguished while they have good performance.

Let $\tau_{i,m}$ be the i-th trajectory of states and actions collected by policy π_m. To obtain different embeddings for different policies, we model the problem as a straightforward pseudo task and predict the indexes of policies from the history trajectories, i.e., design a function $\phi : \mathcal{T} \to Y$, where $\mathcal{T} = \{\tau_{i,m}\}_{i=1,m=1}^{N,M}$ denotes the set of state-action trajectories collected by policies $\{\pi_m\}_{m=1}^{M}$, and $Y = \{y_m\}_{m=1}^{M}$ is the set of the policy indexes, namely the pseudo labels. As our goal is to have more diverse embeddings when they have better performance, which facilitates us in obtaining high quality policies by increasing their diversity. Hence, we need to design a mapping from $(\tau_{i,m}, r_{i,m})_{i=1,m=1}^{N,M}$ to Y, where $r_{i,m}$ denotes the accumulated reward of $\tau_{i,m}$. Here, instead of directly concatenating $r_{i,m}$ with $\tau_{i,m}$, we design a more effective way to incorporate $r_{i,m}$. We regard $r_{i,m}$ as the parameter of a categorical distribution and use this distribution to sample the trajectories for policy π_m. Mathematically,

$$i_m \sim \text{Cat}\left((H(r_{i,m}))_{i=1}^{N}\right), \tag{3}$$

where i_m is the trajectory index sampled by the categorical distribution for policy π_m, and $H(r_{i,m})$ is a heaten equation given by

$$H(r_{i,m}) := \sigma(r_{i,m})^{\frac{1}{T}} / \sum_{j=1}^{N} \sigma(r_{j,m})^{\frac{1}{T}}, \tag{4}$$

where σ is the sigmoid function, N is the batch size, and T is a hyperparameter that is used to adjust the "temperature" of the sampling distribution. With the heaten equation, trajectories with higher cumulative rewards will be sampled to train the model at a higher frequency.

It is worthy noting that the function $\phi : \mathcal{T} \to Y$ needs to capture mutual dependencies between states and actions, and we will discuss its details in Section 3.2.2. Now we obtain the pseudo labels of trajectories sampled from Eq. (3), which will be used to train the representation model by minimizing the loss Eq. (7).

3.2.2 Capture mutual dependencies between actions and states

The non-Markovian rewards depend on the history of states and actions rather than solely on the last state and action. Thus, it is essential to understand and mine the complex characteristics in the trajectories and extract the embeddings of policies. In this section, we use a Transformer to

capture mutual dependencies between actions and states. Because the states and actions have different feature dimensions and can not be directly input into the Transformer encoder, for a trajectory $\tau_{i,m} = (s^1_{i,m}, a^1_{i,m}, ..., s^T_{i,m}, a^T_{i,m})$ collected by policy π_m, we let $\omega^t_{i,m} = (s^t_{i,m}, a^t_{i,m})$ for each time step $t \in \{1, ..., T\}$. Then the Transformer encoder can take $\{\omega^t_{i,m}\}^{T,N,M}_{t=1,i=1,m=1}$ as input and extract the mutual dependencies between multiple time steps. We give a specific form of function $\phi(\cdot)$ mentioned in Section 3.2.1, which is a combination of Eq. (5) and Eq. (6). Subsequently, we let $\omega_{\mathbf{i},\mathbf{m}} = (\omega^t_{i,m})^T_{t=1}$ be a sequence, which can be processed with the Transformer encoder (i.e., a Layernorm (LN), a multiheaded self-attention (MSA) layer, and a residual connections (Wang et al., 2019; Baevski & Auli, 2018)), to obtain the policy embedding:

$$v_{i,m} = \mathrm{MSA}(\mathrm{LN}(\omega_{\mathbf{i},\mathbf{m}} + \mathrm{E}_{\mathrm{pos}})) + \omega_{\mathbf{i},\mathbf{m}}, \tag{5}$$

where $v_{i,m} \in \mathbb{R}^K$ is the i-th embedding for policy π_m and E_{pos} is the positional encoding. We then randomly select M embeddings $\{v_m\}^M_{m=1}$ for M policies from $\{v_{i,m}\}^{N,M}_{i=1,m=1}$ and obtain the diversity matrix by stacking the M selected embeddings. The determinant of the diversity matrix will be included in the general loss (see Eq. (8) in Section3.3) for training the basic off-policy RL models.

To train the policy representation module with the pseudo labels, $v_{i,m}$ are processed with a pseudo classification layer (SC). Our method can achieve a good result when SC is the combination of a Layernorm (LN) and a fully connected (FC) layer, which is formulated by:

$$\xi_{i,m} = \mathrm{SC}(v_{i,m}), \tag{6}$$

where $\xi_{i,m}$ is a vector in which the j-th element ($j \in \{1, 2, ..., M\}$) represents the probability that the policy embedding $v_{i,m}$ is assigned to the j-th policy. Then we train the policy representation module and learn the policy embeddings by minimizing the loss function:

$$\mathcal{L} = -\frac{1}{N}\frac{1}{M}\sum_{i=1}^{N}\sum_{m=1}^{M}({y_m}^T \log(\xi_{i,m})), \tag{7}$$

where y_m is the pseudo label of $\tau_{i,m}$ obtained in Section 3.2.1.

3.3 Policy Diversity Qualification Module

Diverse policy embeddings can be learned by minimizing Eq. (7) in Section 3.2. But, we still cannot guarantee that the diversity of learned policies is achieved without sacrificing their performance. In this section, we first theoretically provide the conditions such that optimal policies can be obtained in our diverse settings. Then, we present the method to ensure the conditions.

Definition 3.1. ***(Diversity Matrix)*** *Consider M policies $\{\pi_m\}^M_{m=1}$, and their embeddings are denoted as $\{v_m\}^M_{m=1}$, where $v_m \in \mathbb{R}^K$. Let $V \triangleq [v_1, ..., v_M]$, where $[\cdot]$ is the operator that stacks vectors into a Matrix. We define Diversity Matrix of policies as $\mathbf{S} \triangleq F(V)$, where $F(\cdot)$ is a function that transforms the $M \times K$ matrix V to a $K \times K$ positive-definite matrix.*

Definition 3.2. ***(Policy Diversity)*** *We define the diversity of policies as the determinant of their diversity matrix, denoted as* $\mathrm{Div}(\{\mathrm{v_m}\}^{\mathrm{M}}_{\mathrm{m=1}}) = \det(\mathbf{S})$.

We explain these two definitions from a geometric perspective. Matrix V in Definition 3.1 is an $M \times K$ matrix. The embeddings of M policies will construct a polyhedron in the space, and the polyhedron volume represents the embeddings dispersion, which is regarded as the diversity of M policies. Suppose the number of policies M is equal to the dimension K of the policy embeddings ($M = K$). In this case, the determinant of the square matrix V can directly represent the volume of a parallelepiped spanned by policy embeddings. In general cases, $M \neq K$, function $F(\cdot)$ is adopted to transform the polyhedron spanned by M K-dimensional embeddings to a parallelepiped spanned by a K-by-K positive-definite matrix in another space. Then the determinant of the square matrix also represents the diversity of M policies.

Now consider the objective function $\tilde{J}(\pi_m)$ of policy π_m, which is the summation of two terms: the first term is the same objective function $J(\pi_m)$ as the base algorithm, and the second term is the policy diversity $\mathrm{Div}(\{\mathrm{v_m}\}^{\mathrm{M}}_{\mathrm{m=1}})$. Specifically, we train $\mathrm{Div}(\{\mathrm{v_m}\}^{\mathrm{M}}_{\mathrm{m=1}})$ and the basic reinforcement learning algorithm by maximizing:

$$\tilde{J}(\pi_m) = (1-\beta)J(\pi_m) + \beta\mathrm{Div}(\{\mathrm{v_m}\}^{\mathrm{M}}_{\mathrm{m=1}}) \tag{8}$$

where and $\beta \in (0, 1)$ controls the trade-off between $J(\pi_m)$ and $\mathrm{Div}(\{\pi_\mathrm{m}\}_{\mathrm{m}=1}^{\mathrm{M}})$. Next, we will give a theory to justify that our method can ensure both the diversity and quality of policies.

Theorem 3.1. *Consider M policies for an environment characterized by finite NMRDP. Suppose optimal policy π_m achieves a cumulative reward of R and suboptimal policy $\tilde{\pi}$ achieves a cumulative reward of $R(\tilde{\pi})$ with $R(\tilde{\pi}) + \Delta < R$ for some $\Delta > 0$. If diversity matrix $\mathbf{S}$ is positive definite, and $\Lambda \triangleq \prod_{i=1}^{K} s_{ii} < \frac{(1-\beta)}{\beta}\Delta$, then the objective in Eq. (8) can only be maximized when all policies are optimal.*

When Δ is small, we can choose a small β to guarantee $\Lambda < \frac{1-\beta}{\beta}\Delta$. The proof of Theorem 3.1 is in Appendix A.3.

A method to construct a diversity matrix. The theory holds if the diversity matrix is positive definite, and we give a method to construct a positive definite diversity matrix. In general, we compute the covariance matrix of the policy embeddings and then regard the determinant of the covariance matrix as the diversity of policies in Definition 3.2. Specifically, we construct the covariance matrix $\mathbf{S}$ of $(v_m)_{m=1}^{M}$, where $v_m \triangleq (v_{m,k})_{k=1}^{K}$. The (i, k)-th element of $\mathbf{S}$ is $s_{ik} = \frac{1}{M-1}\sum_{m=1}^{M}(v_{mi} - \bar{v}_i)(v_{mk} - \bar{v}_k)$, where M is the number of policies, and $\bar{v}_i \triangleq \frac{1}{M}\sum_{j=1}^{M} v_{ji}$. However, the covariance matrix is semi-positive definite, in this section, we use the following method to make it positive definite. When $\det(\mathbf{S}) = 0$, we replace $\mathbf{S}$ with $\tilde{\mathbf{S}}$, where (i, i)-th element is $\tilde{s}_{ii} = s_{ii} + \sum_{j \neq i} |s_{ij}|$. According to Gershgorin circle theorem (GERSCHGORIN, 1931), the modified matrix $\tilde{\mathbf{S}}$ is positive definite. The determinant of the covariance matrix is also called the generalized variance (Wilks, 1932) in statistics, which is a one-dimensional measure of multidimensional scatter.

Intuitive explanations of our method. The larger the determinant of $\mathbf{S}$ is, the more dispersed the data points are. From a geometric perspective: the generalized variance of the samples is proportional to the volume of the ellipsoid (Anderson, 1962; Johnson et al., 2014). Thus, the diversity of policies $\det(\mathbf{S})$ captures the volume of policy embedding space. The detailed explanation is in Appendix A.4.

4 Related Work

Learning diverse policies via Evolution Strategy. These methods are based on the ES (Salimans et al., 2017) algorithm, which is a broad class of population-based black-box optimization algorithms and directly searches in the parameter space of a neural network to find an effective policy. In the DvD (Parker-Holder et al., 2020b) algorithm, behavioral embeddings are generated to characterize the behaviors of each agent, and policy diversity is measured by the determinant of the embedding matrix. The MAP-Elites (Mouret & Clune, 2015) algorithm strives to find different solutions by discretizing behavior descriptor space into a grid of cells. The EDO-CS (Wang et al., 2021) uses a clustering-based selection method to optimize evolutionary diversity. Policies are divided into many clusters based on their behaviors in each iteration, and a high-quality policy is chosen for reproduction from each cluster.

Learning diverse policies via Reinforcement Learning. Our method can be grouped into this category. Some Reinforcement Learning (RL) based methods have been developed to explore diverse behaviors (Eysenbach et al., 2018; Hartikainen et al., 2019). The GEP-PG algorithm (Colas et al., 2018) uses Goal Exploration Processes (Forestier et al., 2017) to generate diverse policies and combines them with the Deep RL algorithm DDPG (Lillicrap et al., 2015), which performs well in continuous control tasks. The RSPO (Zhou et al., 2022) explores diverse policies by solving a filtering-based objective, which restricts RL policies from converging to a solution that differs from a set of local optimal policies. P3S-TD3 (Jung et al., 2019) method periodically determines the best policy among all learners and assigns the best policy parameters to all learners so that the learner can search for a better policy under the guidance of the best policy.

Policy representation learning. A generative method is proposed in Grover et al. (2018), which imitates the policy and generates policy embeddings in the generative module, and maximizes

the distances between embeddings in the discrimination module. Two meta learning methods are proposed in Rusu et al. (2018); Ghosh & Bellemare (2020), and they both regard the latent generative representation of learning model parameters as the policy representation, and the method in Ghosh & Bellemare (2020) shows more stable performance.

Learning diverse policies and effective policy representations is difficult with non-Markovian rewards since the rewards depend on the history of states and actions, and the supervision signals are quite sparse. Moreover, diverse policies should be obtained without sacrificing their effectiveness. To the best of our knowledge, our method is the first method that discusses policy diversity learning with non-Markovian rewards.

5 EXPERIMENTS

To examine the performance of DDM, we conduct experiments in a variety of tasks from the *OpenAI Gym* library (Brockman et al., 2016), including non-Markovian reward and Markovian reward environments. We compare DDM against DvD (Parker-Holder et al., 2020b), QD-RL (Cideron et al., 2020), and P3S (Jung et al., 2019). For a fair comparison, we combine DDM and baselines with the same base algorithm (i.e., DQN, PPO, TD3) and use the same hyperparameters for each environment. The policy set size M of these algorithms is always set to 5. Other experiments details can be found in Appendix A.2. We report the mean and standard deviations across five identical seeds for all algorithms and all tasks. The experiments are performed using the ray (Moritz et al., 2018) library for multi-process parallel computation on a computer with 16-cores.

5.1 NON-MARKOVIAN REWARD ENVIROMENTS

In this experiment, we consider two environments, Point-v1 and FrozenLake-v1, with non-Markovian rewards and multi-solutions to explicitly examine whether DDM can efficiently find high-quality policies with diverse behaviors. In the Point-v1 environment, which is modified from the Point-v0 (Parker-Holder et al., 2020b) and has continuous state and action space, an agent only obtains a non-Markovian reward at the end of each episode, the agent and the target (i.e., green cuboid) are separated by three U-shape walls (Fig. 2(left)), and the current episode ends immediately if the agent hits the wall or reaches the target. The reward is the negative distance between the agent and the target at the end of the episode. The agent will get an additional reward of +100 if it reaches the target and an additional reward of -100 if it hits the wall. Obviously, following the gradient of only the expected cumulative rewards will guide the agent to walk forward directly and hit the wall.

We combine our method DDM and baselines with PPO (Schulman et al., 2017). The results are shown in Fig. 2 (right). It can be observed that DvD-PPO achieves the second best performance but is still 20% worse than our method. That is because DvD-PPO neglects the long-horizon dependencies from the history of actions and states in non-Markovian rewards environments, leading to imperfect policy embeddings. QD-PPO shows unstable performance and unsatisfied policy diversity, which is due to its instability in handling sparse rewards. P3S-PPO finally gets trapped into a less-attractive local optimal. This is because P3S-PPO searches policies only according to the previous best policy, which can cause a rapid improvement of a particular policy but may also lead to search in small-range policy space. It implies that the DDM-PPO can achieve better performance with much fewer training iterations, which shows the advantage of DDM in non-Markovian reward environments.

To measure the policy diversity of different methods, we visualize 5 trajectories of policies in Fig. 3, which are learned after 500 training iterations in the Point-v1 environment. The lines represent the trajectories, where the orange point and the green square represent the start position and target position, respectively, and the black polylines represent the walls that separate the agent and the target. Overall, all policies of our method reach the target successfully in the Point-v1 environment. Moreover, our DDM method manages to bypass the three walls over the upper wall, beneath the bottom wall, and through two gaps between walls, i.e., the gap between the middle wall and the upper wall and the gap between the middle wall and the bottom wall, which demonstrates the excellence of our method in learning diverse policies. Besides, trajectories of our method from the same side are also different from each other. In comparison, the trajectories of P3S-PPO overlap over a long period, and most policies get stuck in front of the walls, which demonstrates that P3S-PPO can not properly handle non-Markovian rewards. For QD-PPO, only two policies can bypass the wall from the gap

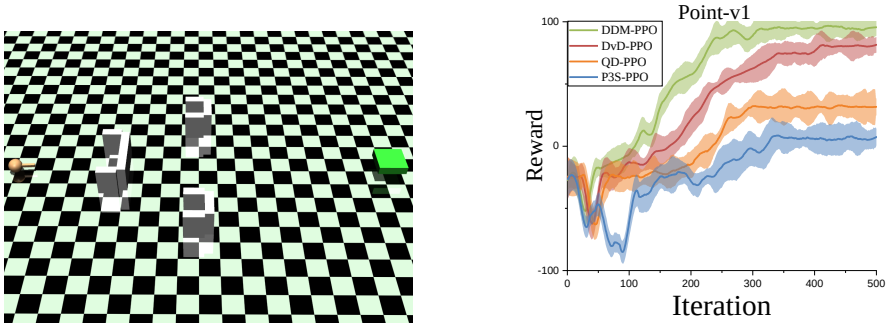

Figure 2: **Left:** the Point-v1 environment. **Right:** the performance of different algorithms

between the middle wall and the upper wall, and the others are stuck in front of the walls. One of the policies hit the edge of the upper wall and stumble to the target. DvD-PPO shows similar results. Only a small part of the policies in the policy set of DvD-PPO, QD-PPO, and P3S-PPO can reach the target, and these policies can only move through two gaps. In comparison, most of the policies of our method can reach the target from different paths, including gaps between walls and spaces over or beneath the walls, demonstrating that our DDM is capable of achieving better performance in learning diverse policies.

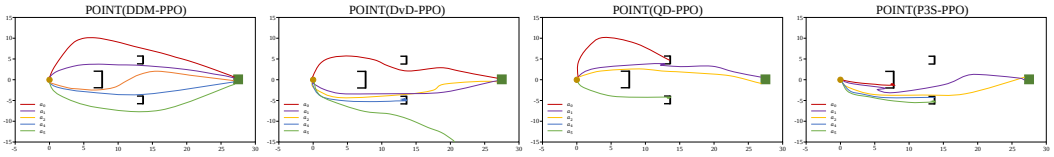

Figure 3: Trajectory visualization of our proposed DDM and baselines after 500 training iterations in the Point-v1 environment.

The FrozenLake-v1 environment is a grid world game, where some tiles are walkable, and some tiles are holes that will make the agent fall into the water. If the agent reaches the goal tile, the returned reward is 1, and if the goal tile is not reached within limited time steps or the agent falls into water, the returned reward is 0. In this environment, the non-Markovian rewards are only obtained at the end of the episodes and thus are extremely sparse. As the size of grids and the density of holes increases, it becomes increasingly difficult for agents to reach the target.

Table 1: The number of times to reach the target in three FrozenLake-v1 environments.

Environment	DDM-DQN	P3S-DQN	DvD-DQN	QD-DQN
FrozenLake-v1 4×4 (4holes)	**2175**	2034	1874	1582
FrozenLake-v1 5×5 (10holes)	**1895**	1798	1603	1489
FrozenLake-v1 8×8 (28holes)	**1040**	770	689	634

In three FrozenLake-v1 environments, we combine our method DDM and baselines with DQN (Mnih et al., 2013) and compare their performance in 4×4, 5×5 and, 8×8 grids, respectively. For each agent, we first conduct 4000 episodes in the experiment. Then, we show the average number of times that each method reaches the target in three FrozenLake-v1 environments in Table 1. DDM-DQN reaches the target more times than DvD-DQN, P3S-DQN, and QD-DQN in all three environments, which shows the advantage of DDM in non-Markovian reward environments. The trajectory visualization of the FrozenLake-v1 is in Appendix A.2.1.

5.2 MARKOVIAN REWARD ENVIRONMENTS

To examine the performance of DDM in environments with Markovian reward, we conduct experiments in standard MuJoCo environments. We combine our method DDM and baselines with TD3 (Fujimoto et al., 2018). The accumulated rewards versus the number of training iterations are shown in Fig. 4. In the three environments, the convergence speed of DDM-TD3 is faster than DvD-TD3, QD-TD3, and P3S-TD3. We can observe that DDM-TD3 can always achieve the best final performance. Although P3S-TD3 can sometimes learn faster at the early stage of optimization, e.g., in the Humanoid-v2 environment, its final performance is worse than the other methods. The results show our method can achieve better performance than baselines in environments with Markovian rewards. Moreover, the results also reflect that in the process of policy learning, policy diversity is not achieved by sacrificing effectiveness.

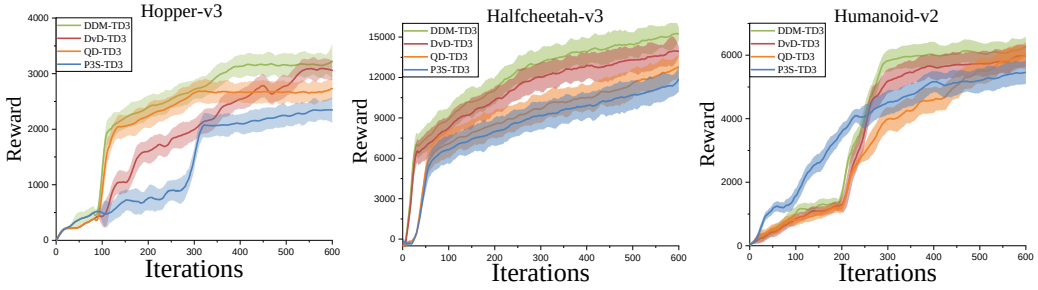

Figure 4: Performance of different algorithms in the three standard MuJoCo environments.

We change the original single-mode environments HalfCheetah and Ant into two two-mode environments by assigning rewards for both Forward and Backward tasks. These changed environments are employed to examine the effectiveness of DDM in learning diverse and high-quality policies. All the methods are trained in two-mode environments and tested in single-mode environments.

Table 2: Rewards obtained by different algorithms in two-mode environments.

Envrionment	Mode	DDM-TD3	P3S-TD3	DvD-TD3	QD-TD3
HalfCheetah	Forward	**5325**	5014	-3591	4897
	Backward	**6716**	-4452	6380	6045
Ant	Forward	**4740**	4454	4437	4033
	Backward	**4734**	-3273	4090	4108

From Table 2, we observe PS3-TD3 can learn to handle Forward tasks, but fails in Backward tasks of both environments. DvD-TD3 can learn to handle both tasks of Ant environment and the Backward task of HalfCheetah Environment. Only QD-TD3 and our method can handle both tasks in both environments, but our method can obtain larger cumulative reward than QD-TD3, demonstrating its good performance in learning high-quality and diverse policies. This demonstrate DDM can learn high-quality and diverse policies in Markovian reward environments.

6 CONCLUSION

In this paper, we introduced DDM, a method for promoting policy diversity for control tasks with non-Markovian rewards. DDM addresses the issue of achieving diverse policies without sacrificing their effectiveness by including the determinant of diversity matrix to policy objective function. Furthermore, DDM develops a Transformer-based policy representation model, which is capable of learning policy embeddings in non-Markovian reward environments. We design a pseudo task in the policy representation module to overcome the challenge of sparse rewards. We demonstrate across a variety of tasks that DDM not only finds diverse, high-quality policies in non-Markovian rewards environments but also manages to maintain strong performances in Markovian rewards environments.

REFERENCES

Theodore Wilbur Anderson. *An Introduction to Multivariate Statistical Analysis*. Job Wiley Sons, Inc. New York-London Sydney, 1962.

Fahiem Bacchus, Craig Boutilier, and Adam J Grove. Rewarding behaviors. In *AAAI/IAAI, Vol. 2*, 1996.

Alexei Baevski and Michael Auli. Adaptive input representations for neural language modeling. *arXiv preprint arXiv:1809.10853*, 2018.

Ronen Brafman, Giuseppe De Giacomo, and Fabio Patrizi. Ltlf/ldlf non-markovian rewards. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Alberto Camacho, Oscar Chen, Scott Sanner, and Sheila A McIlraith. Decision-making with non-markovian rewards: From ltl to automata-based reward shaping. In *Proceedings of the Multi-disciplinary Conference on Reinforcement Learning and Decision Making (RLDM)*, pp. 279–283, 2017.

Geoffrey Cideron, Thomas Pierrot, Nicolas Perrin, Karim Beguir, and Olivier Sigaud. Qd-rl: Efficient mixing of quality and diversity in reinforcement learning. *arXiv preprint arXiv:2006.08505*, 2020.

Cédric Colas, Olivier Sigaud, and Pierre-Yves Oudeyer. Gep-pg: Decoupling exploration and exploitation in deep reinforcement learning algorithms. In *International conference on machine learning*, pp. 1039–1048. PMLR, 2018.

Edoardo Conti, Vashisht Madhavan, Felipe Petroski Such, Joel Lehman, Kenneth O Stanley, and Jeff Clune. Improving exploration in evolution strategies for deep reinforcement learning via a population of novelty-seeking agents. *arXiv preprint arXiv:1712.06560*, 2017.

Antoine Cully and Yiannis Demiris. Quality and diversity optimization: A unifying modular framework. *IEEE Transactions on Evolutionary Computation*, 22(2):245–259, 2017.

Antoine Cully, Jeff Clune, Danesh Tarapore, and Jean-Baptiste Mouret. Robots that can adapt like animals. *Nature*, 521(7553):503–507, 2015.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2018.

Sébastien Forestier, Rémy Portelas, Yoan Mollard, and Pierre-Yves Oudeyer. Intrinsically motivated goal exploration processes with automatic curriculum learning. *arXiv preprint arXiv:1708.02190*, 2017.

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pp. 1587–1596. PMLR, 2018.

Tanmay Gangwani, Jian Peng, and Yuan Zhou. Harnessing distribution ratio estimators for learning agents with quality and diversity. *arXiv preprint arXiv:2011.02614*, 2020.

Maor Gaon and Ronen Brafman. Reinforcement learning with non-markovian rewards. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 3980–3987, 2020.

S GERSCHGORIN. Uber die abgrenzung der eigenwerte einer matrix. *Izv. Akad. Nauk. USSR. Otd. Fiz-Mat. Nauk*, 7:749–754, 1931.

Dibya Ghosh and Marc G Bellemare. Representations for stable off-policy reinforcement learning. In *International Conference on Machine Learning*, pp. 3556–3565. PMLR, 2020.

Aditya Grover, Maruan Al-Shedivat, Jayesh Gupta, Yuri Burda, and Harrison Edwards. Learning policy representations in multiagent systems. In *International conference on machine learning*, pp. 1802–1811. PMLR, 2018.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

Jacques Hadamard. Étude sur les propriétés des fonctions entières et en particulier d'une fonction considérée par riemann. *Journal de mathématiques pures et appliquées*, pp. 171–216, 1893.

Kristian Hartikainen, Xinyang Geng, Tuomas Haarnoja, and Sergey Levine. Dynamical distance learning for semi-supervised and unsupervised skill discovery. In *International Conference on Learning Representations*, 2019.

Zhang-Wei Hong, Tzu-Yun Shann, Shih-Yang Su, Yi-Hsiang Chang, Tsu-Jui Fu, and Chun-Yi Lee. Diversity-driven exploration strategy for deep reinforcement learning. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pp. 10510–10521, 2018.

Richard Arnold Johnson, Dean W Wichern, et al. *Applied multivariate statistical analysis*, volume 6. Pearson London, UK:, 2014.

Whiyoung Jung, Giseung Park, and Youngchul Sung. Population-guided parallel policy search for reinforcement learning. In *International Conference on Learning Representations*, 2019.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Muhammad A Masood and Finale Doshi-Velez. Diversity-inducing policy gradient: Using maximum mean discrepancy to find a set of diverse policies. *arXiv preprint arXiv:1906.00088*, 2019.

V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning. *Computer Science*, 2013.

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

Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I Jordan, et al. Ray: A distributed framework for emerging {AI} applications. In *13th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 18)*, pp. 561–577, 2018.

Jean-Baptiste Mouret and Jeff Clune. Illuminating search spaces by mapping elites. *arXiv preprint arXiv:1504.04909*, 2015.

Nicolas Perez Nieves, Yaodong Yang, Oliver Slumbers, David Henry Mguni, Ying Wen, and Jun Wang. Modelling behavioural diversity for learning in open-ended games. *arXiv preprint arXiv:2103.07927*, 2021.

Jack Parker-Holder, Luke Metz, Cinjon Resnick, Hengyuan Hu, Adam Lerer, Alistair Letcher, Alexander Peysakhovich, Aldo Pacchiano, and Jakob Foerster. Ridge rider: Finding diverse solutions by following eigenvectors of the hessian. *Advances in Neural Information Processing Systems*, 33:753–765, 2020a.

Jack Parker-Holder, Aldo Pacchiano, Krzysztof M Choromanski, and Stephen J Roberts. Effective diversity in population based reinforcement learning. *Advances in Neural Information Processing Systems*, 33, 2020b.

Zhenghao Peng, Hao Sun, and Bolei Zhou. Non-local policy optimization via diversity-regularized collaborative exploration. *arXiv preprint arXiv:2006.07781*, 2020.

Justin K Pugh, Lisa B Soros, and Kenneth O Stanley. Quality diversity: A new frontier for evolutionary computation. *Frontiers in Robotics and AI*, 3:40, 2016.

Andrei A Rusu, Dushyant Rao, Jakub Sygnowski, Oriol Vinyals, Razvan Pascanu, Simon Osindero, and Raia Hadsell. Meta-learning with latent embedding optimization. In *International Conference on Learning Representations*, 2018.

Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F Wong, and Lidia S Chao. Learning deep transformer models for machine translation. *arXiv preprint arXiv:1906.01787*, 2019.

Yutong Wang, Ke Xue, and Chao Qian. Evolutionary diversity optimization with clustering-based selection for reinforcement learning. In *International Conference on Learning Representations*, 2021.

Samuel S Wilks. Certain generalizations in the analysis of variance. *Biometrika*, pp. 471–494, 1932.

Yunbo Zhang, Wenhao Yu, and Greg Turk. Learning novel policies for tasks. In *International Conference on Machine Learning*, pp. 7483–7492. PMLR, 2019.

Zihan Zhou, Wei Fu, Bingliang Zhang, and Yi Wu. Continuously discovering novel strategies via reward-switching policy optimization. *arXiv preprint arXiv:2204.02246*, 2022.