Sparse Attention Guided Dynamic Value Estimation for Single-Task Multi-Scene Reinforcement Learning

Jaskirat Singh & Liang Zheng
Australian National University
Canberra, Australia
{jaskirat.singh,liang.zheng}@anu.edu.au

Abstract

Training deep reinforcement learning agents on environments with multiple levels / scenes from the *same task*, has become essential for many applications aiming to achieve generalization and domain transfer from simulation to the real world [6, 39]. While such a strategy is helpful with generalization, the use of multiple scenes significantly increases the variance of samples collected for policy gradient computations. Current methods, effectively continue to view this collection of scenes as a single Markov decision process (MDP) and thus, learn a scene-generic value function $V(s)$. However, we show that the sample variance for a multi-scene environment is best minimized by treating each scene as a distinct MDP, and then learning a joint value function $V(s, \mathcal{M})$ dependent on both state s and MDP $\mathcal{M}$. We further demonstrate that the true joint value function for a multi-scene environment, follows a multi-modal distribution which is not captured by traditional CNN / LSTM based critic networks. To this end, we propose a dynamic value estimation (DVE) technique, which approximates the true joint value function through a *sparse* attention mechanism over multiple value function hypothesis / modes. The resulting agent not only shows significant improvements in the final reward score across a range of OpenAI ProcGen environments, but also exhibits enhanced navigation efficiency and provides an implicit mechanism for unsupervised state-space skill decomposition.

1 Introduction

While the field of reinforcement learning has shown tremendous progress in the recent years, generalization across variations in the environment dynamics remains out of reach for most state-of-the-art deep RL algorithms [29, 38, 40]. In order to achieve the generalization objective, many deep RL approaches attempt to train agents on environments comprising of multiple levels or scenes from the same task [5, 6, 20–22, 39, 42]. Although incorporating a wider source of data distribution in the training itself has shown promise in bridging the train and test performance, the inclusion of multiple scenes, each defined by a distinct underlying MDP, significantly increases the variance of samples collected for policy gradient computations [5, 34].

The current approaches using multi-scene environments for training usually deal with the high variance problem by deploying multiple actors for collecting a larger and varied range of samples. For instance, [39, 42] use multiple asynchronous actor critic (A3C) models when training on the AI2-THOR framework based visual navigation task. Similarly, [5, 6] deploy parallel workers to stabilize policy gradients for multi-level training on procedurally-generated game environments [21]. While parallel sample collection helps in stabilizing the learning process, the obvious disadvantages of lower sample efficiency and higher hardware constraints, suggest the need for specialized variance reduction techniques in multi-scene reinforcement learning.

Preprint. Under review.

Most RL generalization benchmarks [6, 18, 27, 41] *effectively* treat the collection of scenes as a *single-MDP environment*. That is, a common and scene generic value function $V(s)$ is learned across all levels. By comparison, we propose an improved variance reduction formulation, which instead shows that the sample variance for a multi-scene environment is best minimized by acknowledging each scene as a separate MDP and then learning a joint value function $V(s, \mathcal{M})$ dependent on both state s and MDP $\mathcal{M}$.

However, given the lack of information about the operational level at train / test times, estimating the joint value function $V(s, \mathcal{M})$ presents a challenging problem. To address this, we first show that the underlying true joint value function samples follow a multi-modal distribution. We then use this insight to propose a dynamic value estimation strategy, which approximates the overall value distribution through a progressively learned *sparse* attention mechanism over the corresponding distribution modes. The sparse attention guided dynamic networks not only result in huge improvements in total reward on the OpenAI ProcGen benchmark, but also exhibit semantically desirable properties like enhanced navigation efficiency and provide a framework for unsupervised state space decomposition.

To summarize, the main contributions of this paper are as follows,

- **Enhanced Variance Reduction.** We propose an improved variance reduction formulation which shows that the sample variance for a multi-scene environment is best minimized by treating the each scene as a distinct MDP, and then learning a joint value function $V(s, \mathcal{M})$ dependent on both state s and MDP $\mathcal{M}$.
- **Clustering Hypothesis.** We show that the true scene-specific value function distribution is best described using a mixture model with multiple dominant modes, which are not fully captured by the current CNN or LSTM based critic networks.
- **Novel Critic Module.** We propose a novel critic model which approximates the true multi-modal value function distribution through a progressively learned *sparse* attention mechanism over multiple value function hypothesis / modes.
- **Implicit State-Space Decomposition:** We demonstrate that the learned sparse attention divides the overall state space into distinct sets of game skills. The collection of these skills represents a curriculum that the agent must master for effective game play.
- **Navigation Efficiency.** Through both quantitative and qualitative results, we show that the sparse dynamic model leads to huge improvements in the navigation efficiency of the resulting agent. Furthermore, the high navigation efficiency of our method and its tendency to limit unnecessary exploration, presents an effective alternative to explicit reward-shaping [24, 25, 39, 42], for penalizing longer episode-lengths / reward-horizons in multi-scene reinforcement learning.

2 Problem Setup

A typical multi-scene environment is characterized by a set of possible MDPs $\boldsymbol{\mathcal{M}}$: $\{\mathcal{M}_1, \mathcal{M}_2, ...\mathcal{M}_N\}$, each defined by its own state space $\mathcal{S}_{\mathcal{M}}$, transition probabilities $P_{\mathcal{M}}(s_{t+1}|s_t, a_t)$, reward function $r_{\mathcal{M}}(s_t, a_t, s_{t+1})$ and discount factor γ. An agent with action space $\mathcal{A}$ interacts with a randomly chosen and unknown MDP $\mathcal{M} \in \boldsymbol{\mathcal{M}}$, to generate a trajectory $\tau : (s_0, a_0, s_1...s_T)$ with total reward $\mathcal{R}_\tau = \sum_{t=0}^{T-1} \gamma^t r_{\mathcal{M}}(s_t, a_t, s_{t+1})$. The goal of the agent is to maximize the expected trajectory rewards over the entire MDP set $\boldsymbol{\mathcal{M}}$, *i.e.* $\mathbf{E}_{\tau,\mathcal{M}}\left[\mathcal{R}_{\tau,\mathcal{M}}\right]$.

3 Motivation

In this section, we present a step-by-step analysis which motivates the final method presented in Sec. 4. We first begin by proposing an improved variance reduction formulation for multi-scene environments in Sec. 3.1. We then demonstrate the multi-modal nature of the joint value distribution in Sec. 3.2. Finally we outline the main idea behind dynamic value estimation in Sec. 3.3.

3.1 Variance Reduction for Multi-Scene Environments

For a single-MDP environment, with policy network π (parameterized by θ) and an action-value function $Q(s, a)$, the general expression for computing policy gradients with minimal possible sample

variance can be written as [15, 31],

$$\nabla_\theta J = \mathbf{E}_{s,a}\left[\left(\nabla_\theta \log \pi(a|s)\right)\, \psi(s,a)\right], \tag{1}$$

where $\psi(s,a) = Q(s,a) - V(s)$ is the advantage function. Similarly for a multi-scene environment, it can be shown that the optimal formulation for minimizing total sample variance is given by (please refer supplementary material for proof),

$$\nabla_\theta J = \mathbf{E}_{s,a,\mathcal{M}}\left[\left(\nabla_\theta \log \pi(a|s)\right)\, \psi(s,a,\mathcal{M})\right], \tag{2}$$

where $\psi(s,a,\mathcal{M}) = Q(s,a,\mathcal{M}) - V(s,\mathcal{M})$. Here $Q(s,a,\mathcal{M})$ and $V(s,\mathcal{M})$ represent the action-value and value function respectively for the particular MDP $\mathcal{M}$. However, since most of the time knowledge about the operational MDP $\mathcal{M}$ is unknown to the agent, the current policy gradient methods continue to use a single scene-generic value function estimate $\hat{V}(s)$ for variance reduction. However, $\hat{V}(s)$ then is essentially an estimate of the global average over the underlying scene-specific value functions $\{V_{\mathcal{M}_1}(s), V_{\mathcal{M}_2}(s), ..., V_{\mathcal{M}_N}(s)\}$, and thus gives a poor approximation of the joint value function for a given MDP. We next show that such a simplification is not necessary and present an approach for obtaining a better approximation for the joint value function $V(s,\mathcal{M})$.

3.2 Clustering Hypothesis

Training on multi-scene environments over the same domain task can lead to ambiguity in value function estimation. That is, two visually similar states could have very different value function estimates corresponding to distinct scenes / levels. In this section, we empirically demonstrate that unlike a single-scene environment, the true value function for a multi-scene environment (having scenes with similar state spaces), is better described by a multi-modal distribution.

Empirical Demonstration on OpenAI CoinRun . To test the above hypothesis, we finetune separate critic networks over a fixed policy π, to obtain the true MDP-specific value function estimates $\{V(s,\mathcal{M}_1), V(s,\mathcal{M}_2), ..., V(s,\mathcal{M}_{50})\}$ corresponding to a random selection of 50 levels from the CoinRun ProcGen environment [6]. We then use a Gaussian Mixture Model (GMM) for fitting these $V(s,\mathcal{M}_i)\{i \in [1,50]\}$ samples. Results are shown in Fig. 1. We observe that the true value function estimates form a multi-modal distribution that is not captured by traditional CNN or LSTM based critic networks. (Please refer supplementary materials for further details.)

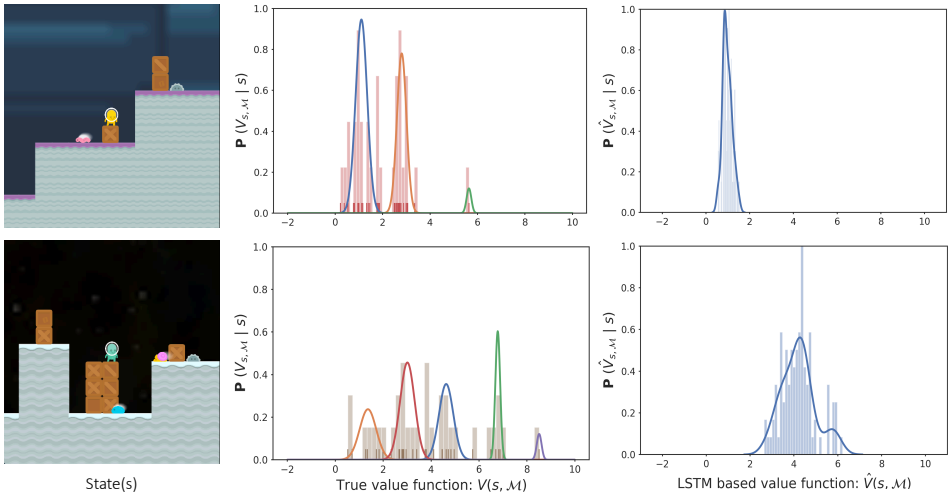

Figure 1: **Clustering Hypothesis.** Column 1-2 demonstrate multi-modal nature of the true value function distribution, for an intermediate policy π, on a set of randomly selected 50 levels from the CoinRun environment. The true value estimate for a state image shown in Column 1 can be characterized by one of the many clusters. Column 3: In contrast, the LSTM based value predictions, though showing some variance with MDP $\mathcal{M}$, fail to capture the multiple dominant modes exhibited by the true value function distribution.

3.3 Minimizing Value Prediction Error

Theorem 1. The sample variance (ν) for policy gradients defined by Eq. 2, can be minimized by reducing the prediction error ϵ between the true joint value function $V(s, \mathcal{M})$ and the predicted estimate $\hat{V}(s, \mathcal{M})$, where $\epsilon = \mathbf{E}_{s,\mathcal{M}}[V(s, \mathcal{M}) - \hat{V}(s, \mathcal{M})]^2$.

The proof is provided in supplementary materials. We now use the insight provided by Theorem 1 to propose the following solution for reducing the policy gradient sample variance.

Proposed Solution: While the exact estimation of the true joint value function $V(s, \mathcal{M})$ is infeasible without knowledge of MDP $\mathcal{M}$, we use the results of Section 3.2, to assert that the prediction error can be reduced by approximating the value function as the mean value of the cluster to which the current MDP belongs. Fig. 2 provides an overview of this idea.

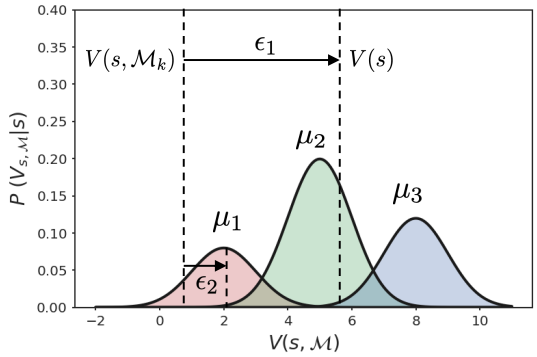

Figure 2: **Proposed Approach.** Traditional methods learn a single scene-generic value function estimate $V(s)$ which leads to high prediction error ($\epsilon_1 > \epsilon_2$). In contrast we propose that the overall prediction error can be significantly reduced by approximating the joint value function $V(s, \mathcal{M})$ as the mean of the nearest cluster.

4 Our method

The solution proposed in Section 3.3 can be implemented through a sparse attention mechanism over the value function modes μ_i, wherein the attention parameters are 1 for the closest cluster and 0 otherwise. However the sparse attention parameters are not fixed, as the cluster to which an MDP belongs is expected to change while training the RL agent. To address this, we first model the predicted value function through a generalized attention mechanism over the value function modes (refer Section 4.1) and then propose a novel loss function which progressively enforces sparse attention parameters based on the training dynamics (refer Section 4.2).

4.1 Continuous Dynamic Estimation Formalism

Mathematically, given that the *true* value function follows a Gaussian Mixture Model (GMM) like,

$$P\left(V(s_t, \mathcal{M})|s_t\right) = \sum_{i=1}^{N_b} p_i \, \mathcal{N}(V(s_t, \mathcal{M}) \mid \mu_i, \sigma_i^2), \tag{3}$$

we propose to model the *predicted* value function as,

$$\hat{V}(s_t, \mathcal{M}) = \sum_{i=1}^{N_b} \alpha_i(s_t, \mathcal{M}) \, \mu_i(s_t), \quad s.t. \quad \alpha_i > 0, \quad \sum_{i}^{N_b} \alpha_i = 1. \tag{4}$$

That is, given a state s_t, we predict N_b distinct value function hypotheses $\{\mu_1(s), \mu_2(s), ..., \mu_{N_b}(s)\}$ (one corresponding to each cluster). The final value prediction is then modelled as the weighted combination of these value hypotheses using attention parameters α_i. Wherein, the attention parameters $\alpha_i(s_t, \mathcal{M})$ are used to capture the similarity between the i^{th} value hypothesis and the true value for MDP $\mathcal{M}$. In practice, since the current MDP $\mathcal{M}$ is not known, the parameters α_i are learned from (state, episode trajectory) pairs $\{s_t, \tau^{t-}\}$ ($\tau^{t-} : \{s_0, a_0, ...s_{t-1}\}$ is the trajectory till time $t-1$), using a Long Short Term Memory (LSTM) [16] network.

4.2 Enforcing Sparse Attention

We first define two metrics to describe the attention parameter distribution, *confusion* and *contribution*. Confusion (δ) is a measure of uncertainty as to which cluster, the current state-trajectory pair

$\{s_t, \tau^{t-}\}$ belongs to. On the other hand, contribution (ρ), as the name suggests, determines the "contribution" of a cluster in the overall value function estimation across a general trajectory sequence $\tau : \{s_0, a_0, s_1, a_1, ...s_T\}$. Formally, confusion and contribution are defined as,

$$\delta(s_t, \tau^{t-}) = \frac{1}{N_b \cdot \sum_i \alpha_i^2(s_t, \tau^{t-})}, \quad \rho_i(\tau) = \frac{1}{T} \sum_{t=1}^{T} \delta(s_t, \tau^{t-})\ \alpha_i(s_t, \tau^{t-}). \tag{5}$$

We now note that an increase in sparsity of cluster assignments $\{\alpha_1, \alpha_2, \dots \alpha_{N_b}\}$ is equivalent to maximization of their l_2 norm. Thus using Eq. 5, it corresponds to minimization of the confusion (δ) metric. However, a mere enforcement of sparsity may encourage convergence to solutions where only one of the clusters is active. We also want to ensure that each cluster contributes equally in the $(s, \mathcal{M})$ space. To achieve this, we propose the following *confusion-contribution loss*,

$$L^{CC} = k_1\ \mathbf{E}_{s_t, \tau^{t-}} \left[\log \delta(s_t, \tau^{t-})\right] + k_2\ \mathbf{E}_{\tau} \left[\log \left(\sum_i^{N_b} \rho_i^2(\tau)\right)\right]. \tag{6}$$

The above sparse loss is then used in conjunction with the policy and value-function loss while training the RL agent. However, we emphasize that the state space must already be well explored[1] prior to the application of confusion-contribution loss. If applied prematurely, due to the continuous nature of neural networks, the sparse cluster assignment maybe incorrectly generalized across the entire state space. This could lead to a detrimental impact on value function estimation for the currently unexplored states. Also, such a mistake is hard to recover from, because for any state $s \in \mathcal{S}$, the sparse assignment ensures that the gradients for all but one cluster are approximately zero.

5 Experiments

In this section, we mainly aim to pursue the following three goals 1) Understand how dynamic value estimation (DVE) effects both train and test time performance on different multi-scene environments. 2) Demonstrate the advantage of DVE as a framework for learning unsupervised state space decomposition and 3) Analyse the enhanced navigation efficiency resulting after applying DVE.

We test our method on the hard distribution setting of different multi-scene environments from the OpenAI ProcGen [5] benchmark: (`Coinrun, Caveflyer, Climber, Jumper, Chaser, Bigfish, Plunder`). Following [5], we adopt the "*500 level generalization*" as our evaluation protocol. In particular, an agent is trained on a set of 500 levels from a given ProcGen environment, and evaluated for its performance on the remaining unseen levels. All models adopt an IMPALA-CNN-LSTM [11] architecture and are trained using the Proximal policy optimization (PPO) [32] algorithm, which is ran with 4 parallel workers (GPUs) for gradient computations as this is seen to enhance performance. Each worker is trained for 50M steps, thus equating to a total of 200M steps across all the 4 workers. All results and standard deviations are reported as the average across 4 random seeds. Additional evaluation on the challenging visual navigation benchmark, along with further hyperparameter and implementation details are provided in the supplementary material.

5.1 Impact on Train and Test Performance

We compare the proposed dynamic value estimation (DVE) strategy with a range of recent works aimed at increasing the generalization performance in deep reinforcement learning. In particular, we include comparisons with batch-normalization which outperforms other regularization techniques in [6], along with the cutout-color, random-crop and random-convolution based data-augmentation strategies which show high-performance according to [23]. Furthermore, we also compare our method with ITER [19], IBAC-SNI [18] and MixReg [37], which are all designed specifically with the purpose of improving generalization performance on the OpenAI Procgen benchmark.

Results are shown in Table 1. We clearly see that DVE outperforms baseline PPO by a large margin on both train and test performance. We also note that, as seen in [37], the use of data-augmentation strategies from RAD [23] while helpful on a couple of environments, often leads to significantly

[1]For OpenAI ProcGen, we consider state space to be sufficiently explored when the avg. episode length plateaus / starts decreasing. Please refer supplementary material for further details.

Train Performance							
Method	CoinRun	Caveflyer	Climber	Jumper	Chaser	Bigfish	Plunder
PPO [32]	7.78±0.3	6.79±1.1	7.59±0.9	6.61±0.5	7.41±0.7	10.57±0.9	5.82±0.5
BatchNorm [6]	9.01±0.6	5.26±0.6	8.67±0.3	6.24±0.3	8.69±0.5	14.38±2.3	8.63±0.2
RAD (Cutout-color) [23]	8.21±0.7	4.93±0.4	7.38±0.5	6.39±0.5	3.23±0.1	10.36±0.8	8.03±0.7
RAD (Random-crop) [23]	5.98±0.2	5.22±0.3	3.56±0.2	4.12±0.1	3.21±0.2	3.17±0.2	5.40±0.3
RAD (Random-conv) [23]	6.72±0.3	4.47±0.2	5.34±0.2	6.44±0.6	2.75±0.1	3.45±0.2	5.74±0.4
ITER [19]	8.34±0.5	6.42±0.4	9.21±1.2	6.84±0.7	5.52±0.8	10.54±0.7	4.86±0.5
IBAC-SNI [18]	8.09±0.3	6.07±0.5	7.67±0.4	6.37±0.3	7.29±0.2	12.50±1.2	4.58±0.6
MixReg [37]	8.75±0.4	8.64±0.2	9.29±0.5	**7.16±0.4**	7.58±0.4	13.08±1.6	7.89±0.4
DVE (Ours)	**9.57±0.1**	**11.59±0.3**	**9.95±0.7**	6.65±0.1	**9.64±0.3**	**14.70±1.1**	**9.67±0.8**
Test Performance							
Method	CoinRun	Caveflyer	Climber	Jumper	Chaser	Bigfish	Plunder
PPO [32]	6.25±0.5	5.31±1.4	5.22±0.5	3.03±0.2	6.99±0.8	5.81±0.4	3.64±0.4
BatchNorm [6]	7.12±0.3	3.87±0.5	6.17±0.3	2.95±0.2	7.47±0.4	7.25±1.6	5.96±0.3
RAD (Cutout-color) [23]	6.82±0.6	3.68±0.4	5.20±0.4	2.88±0.4	2.86±0.3	6.31±0.6	6.14±0.6
RAD (Random-crop) [23]	5.46±0.2	3.23±0.2	3.28±0.1	2.43±0.1	2.39±0.2	2.04±0.1	4.69±0.4
RAD (Random-conv) [23]	5.37±0.3	2.86±0.2	3.34±0.2	2.81±0.3	2.36±0.1	2.58±0.2	4.82±0.3
ITER [19]	7.19±0.4	5.41±0.4	**7.57±1.1**	3.53±0.5	5.47±0.8	6.83±0.2	3.73±0.3
IBAC-SNI [18]	7.04±0.1	5.17±0.6	7.29±0.4	3.14±0.1	7.14±0.3	8.28±0.7	3.62±0.4
MixReg [37]	6.72±0.4	5.86±0.3	7.12±0.1	**4.01±0.4**	7.36±0.4	8.09±1.1	5.41±0.1
DVE (Ours)	**7.43±0.2**	**8.08±0.2**	7.41±0.3	3.32±0.2	**8.44±0.2**	**9.43±0.9**	**6.23±0.5**

Table 1: **Final performance comparison**. The proposed dynamic value estimation (DVE) approach results in significant improvements over the previous works in both train and test performance.

worse performance on both train and test levels. Furthermore, we note that while DVE does not focus on generalization specifically (but improves both train and test performance by learning a better value estimate), it still consistently outperforms other generalization methods on all test environments (except the Jumper environment). In particular, averaged over all test environments, we see that DVE leads to a test-time performance improvement of 39.4%, 27.5%, 11.5%, 29.63% and 24.3% over the PPO, BatchNorm, MixReg, ITER and IBAC-SNI methods respectively.

5.2 Learning Implicit State Space Decomposition

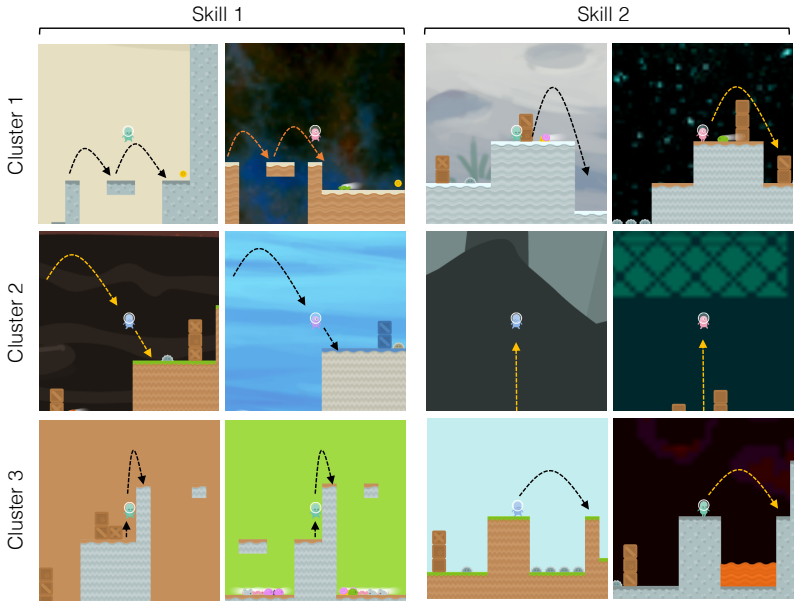

Figure 3: **Visualizing Cluster Features.** Examples of key obstacles types learned by each cluster in the CoinRun Environment. We note that the sparse training divides the overall state space into a distinct sets of game skills.

A key advantage of our method can be seen in its ability to achieve an unsupervised division of the state space into distinct sets of game skills. The state space decomposition is achieved through the sparse cluster assignments, wherein the network learns to assign each state, trajectory pair $\{s_t, \tau^{t-}\}$ to a distinct value cluster. In this section, we use this sparse property of our method to visualize different obstacle types characteristic of each cluster in the CoinRun Environment.

To visualize the distinguishing features for each cluster, we first extract the set of states $\mathcal{S}_i$ for which each cluster is active. The latent representations (output of the LSTM network) for these states are then used to map each $s \in \mathcal{S}_i$ to a two dimensional embedding space using TSNE [26]. This embedding is then manually analysed for clusters to then identify the salient obstacle classes.

Fig. 3 shows some key obstacle types for each cluster. We observe that each cluster is responsible for predicting the value function on a distinct set of obstacles/skills. For instance, *Cluster-1* is responsible for value estimation in cases like double-jump from one side to another (Skill-1) and crossing over moving enemies (Skill-2). On the other hand, *Cluster-2* handles landing after jumps from higher ground (Skill-1) and high jumps with very limited visibility of coming obstacles (Skill-2). Finally, *Cluster-3* takes care of precision climbs (Skill-1) and jumps over wide valleys (Skill-2).

Thus, we see that each disjoint state space set $\mathcal{S}_i,\ i \in [1, N_b]$ represents a distinct curriculum of game skills that must be learned for mastering the overall multi-scene game environment. This division is semantically desirable and is analogous to the human learning paradigm wherein it is quite common to break down a complex task into a set of manageable skills before attempting the complete task.

5.3 Enhanced Navigation Efficiency

Navigation Efficiency (Total reward / Episode length) $[\times 10^{-2}]$							
Method	CoinRun	Caveflyer	Climber	Jumper	Chaser	Bigfish	Plunder
PPO [32]	6.14	3.03	4.21	2.80	2.24	1.47	1.24
MixReg [37]	8.46	3.51	3.98	3.34	2.04	1.88	1.52
DVE (Ours)	**14.08**	**15.42**	**5.99**	**8.52**	**3.09**	**2.01**	**1.73**

Table 2: **Navigation Efficiency Comparison**. Comparing the navigation efficiency between DVE, PPO and MixReg (which is the most consistent baseline beside DVE as per Table 1). We clearly see that DVE leads to significant increase in the overall navigation efficiency *i.e.* it achieves higher rewards while needing much shorter episode lengths (per reward unit).

In addition to reporting results for average episode reward, we also compare model performance based on the agent's efficiency in completing a game level. The navigation efficiency is thus measured by the ratio of the final reward and average episode length. Results are reported in Table 2. We clearly see that DVE leads to better reward scores while on average, using much fewer timesteps per episode[2]. For instance, we see that the DVE leads to an increase of 129.6%, 408.9% & 204.4% (over baseline PPO) in the reported navigation efficiency scores, for the CoinRun, CaveFlyer, and Jumper environments respectively. This massive increase in navigation efficiency results from two reasons,

- The tendency to use fewer time-steps is a direct consequence of optimizing the discounted reward function with $\gamma < 1$ [31]. As a result, the agent is incentivized to minimize the number of steps between the current state and the next reward. Hence, more accurate policy updates (with lower sample variance) should lead to fewer timesteps.
- As explained in Sec. 4.2, the expansion in state space after application of the confusion-contribution loss can lead to potential errors in value function estimation. Thus, the sparse dynamic agent learns to maximize the utilization of the already explored state space.

We next qualitatively analyse how learning a more accurate value function leads to shorter (and efficient) episode trajectories for the sparse DVE agent. Note that the computation of a suboptimal value function at a *critical* environment state (*e.g.* a tricky obstacle) can cause the agent to underestimate the *advantage* of choosing an action which leads to a faster route to the final destination / goal. We next try to identify these *critical* states by comparing episode trajectories for the baseline PPO and DVE agents on the CaveFlyer environment.

[2]Note that the ProcGen environments have no explicit penalty for discouraging longer episode lengths.

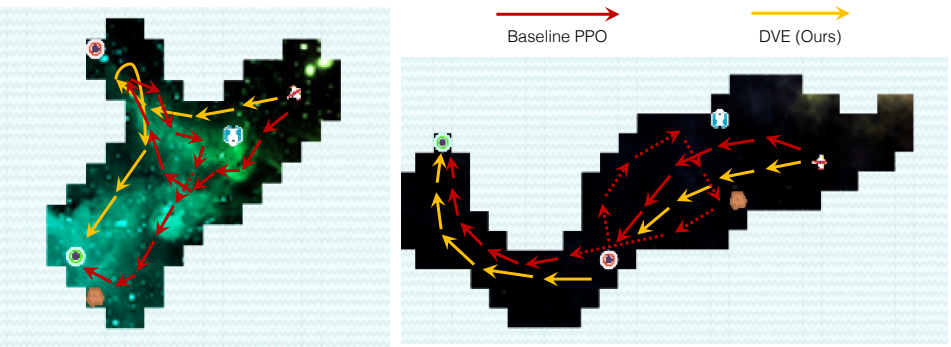

Figure 4: **Comparing Navigation Efficiency.** Demonstrating qualitative difference between successful trajectories for baseline PPO and sparse DVE agents. Our method shows higher efficiency in navigating to the final goals (red & green spheres).

Game Description. The goal of the Caveflyer environment (Fig. 4) is to destroy the red spheres and finally reach the green sphere while avoiding intermediate obstacles. The agent receives a small reward of +3 on destroying a red sphere and an end of episode reward of +10 on successfully reaching the green one. Direct collisions with an obstacle or the red sphere cause immediate episode termination.

Trajectories for both baseline PPO and sparse DVE agents are shown in Fig. 4. We see that the baseline PPO agent after destruction of the red sphere (critical state), effectively restarts its search for the next target, while often revisiting already encountered states. In contrast, the sparse DVE agent with its more accurate value estimates, realizes that the expected value for exploring unseen parts of the cave is much higher than revisiting previous states. Doing so not only helps the DVE agent reach the end goals much faster, but also eliminates the need for evading obstacles that it has already crossed (thereby increasing the episode success rate).

We also note that, the balance between the sparse model's reluctance towards state space expansion and maximization of total reward can be modulated through the scale of coefficients (k_1, k_2) of the confusion-contribution loss. In this regard, the high navigation efficiency of our method provides an effective alternative to designing explicit reward shaping penalties [25, 39, 42] for promoting reduced episode lengths.

6 Related Work

Meta Reinforcement Learning. Duan *et al.* [10] previously proposed the use of recurrent neural networks and episode trajectories as a meta-RL approach for adapting to environment dynamics. While in theory, an LSTM is capable of learning multi-modal distributions, we find that in practice the vanilla-LSTM based conditional value function distribution (for a given state) is usually characterized by a single dominant mode (refer Fig. 1), and thus fails to capture the multi-modal nature of true value function distribution. In contrast, our method explicitly forces multiple dominant modes while estimating the cluster means μ_i and uses episode trajectories to compute the assignment (α_i) of the current state sample to each cluster.

Generalization in Reinforcement Learning. Recently, multi-scene environments have been extensively used to study and address the problem of overfitting in RL. [6] deploy standard regularization techniques from supervised learning like dropout, batch-normalization, L2-regularization to counter overfitting when training on the multi-scene CoinRun environment. [23, 37] improve generalization performance by increasing the diversity of training data observations. [19] reduce overfitting by minimizing non-stationarity while training the RL agent. Noise injection techniques like [18, 35] add noise to the model parameters in order to improve the generalization capability. Our work is different as it does not focus on generalization specifically, but improves both train and test time performance by learning a better value function estimate. Furthermore, we note that our approach is orthogonal to these works and would likely also benefit from the use of different regularization methods.

Recurrent Independent Mechanisms. Goyal *et al.* [14] propose modular structures called Recurrent Independent Mechanisms (RIMs) which specialize to different dynamic processes within an environment, and communicate sparingly through a sparse attention mechanism. While RIMs and our method share the idea of having sparse attention, our work differs significantly as RIMs use separate recurrent models with independent dynamics, whereas we deploy a single actor-critic network with shared dynamics. Another important distinction is that [14] assume a sparse structure from the beginning and hence require environments with clearly independent dynamic processes. In contrast, the sparse assignment in our method is learned progressively and only after sufficient exploration, which allows for a more informed division of the state space (refer Sec. 5.2).

Distributional RL. Recent works like [1, 4, 7] aim to directly learn the value function distribution instead of modelling the expected return. Our work differs in the following aspects. First, distributional RL methods need to discretize the return space using a high number of support locations/nodes (*e.g.* $N = 200$ for [1]) to approximate the overall value distribution. In contrast, we approximate the joint value function through only the modes of the underlying distribution and thus require very few output nodes ($N \in [2, 5]$). Second, the current work on distributional RL is limited to off-policy methods with a shared replay buffer. The use of a large replay buffer implies that the overall sample distribution changes minimally from one update to another. This is in sharp contrast to the high variance seen in on-policy training which is the focus of current work. To the best of our knowledge, our work is the first stable method for approximating value distribution in on-policy RL.

Bootstrapped DQN. Osband *et al.* [28] propose the use of multi-head Q-networks for bootstrapped DQNs. However, they aim to facilitate deep exploration and hence select the Q-network head for a given episode randomly. In contrast, we aim to reduce the prediction error with the true value function distribution and also present a novel approach which progressively learns the most representative cluster for each state $s \in \mathcal{S}$ through the confusion-contribution loss.

7 Conclusion

This paper introduces a novel dynamic value estimation strategy for enhanced variance reduction in multi-scene reinforcement learning environments. The proposed approach consistently outperforms the current generalization methods on both train and test performance for a range of OpenAI ProcGen environments, while exhibiting much higher navigation efficiency to complete a game level. Additionally, we observe that the learned sparse attention parameters divide the overall state space into disjoint subsets. We show that each subset focuses on a distinct set of game-skills, which is semantically desirable and draws a strong parallel with the human learning paradigm.

8 Potential Societal Impacts

Positives. While our work is largely theoretical, we believe that in the long term, it will have major impact in the upcoming area of AI-inspired learning [30]. Recent years have seen the field of deep reinforcement learning demonstrate tremendous success in achieving super-human performance in complex game play. Deepmind's Alphazero [33], Alphastar [36] and OpenAI's Dota-2 [2] are some salient examples. Each such milestone is followed by an increased public interest to analyse and break down the policy of the trained RL agent into a set of simple skills than can be consumed by a human learner [9, 30]. This process is often manual and involves painstaking analysis across hundreds of game runs. As shown in Section 5.2, our method does this automatically by dividing the possible game scenarios (states) into distinct sets of game skills. While each set can be composed of other mini-skills, the broad division achieved by our method promises great potential in the development of semi-automatic, AI-inspired teaching tools for human players.

Potential Negatives. Another societal impact can be envisioned in the field of robotics and social healthcare. An improvement in visual navigation performance (refer supp. material for experiments) within a controlled environment, has the potential to be used in the development of caretaker robots for the elderly [17]. This can be associated with a number of ethical concerns as detailed in [8, 13]. However, as shown by the success of healthcare robots like Moxi [3, 12] in the current COVID pandemic, assistive healthcare robots can work efficiently along-side human nurses by performing time-consuming / repetitive tasks. Thus, overall we believe that the positive benefits of our work by far outweigh such potential concerns.

References

[1] Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, pages 449–458. PMLR, 2017.

[2] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[3] Nupur Chatterji, Courtney Allen, and Sonia Chernova. Effectiveness of robot communication level on likeability, understandability and comfortability. In *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 1–7. IEEE, 2019.

[4] Yunho Choi, Kyungjae Lee, and Songhwai Oh. Distributional deep reinforcement learning with a mixture of gaussians. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9791–9797. IEEE, 2019.

[5] Karl Cobbe, Christopher Hesse, Jacob Hilton, and John Schulman. Leveraging procedural generation to benchmark reinforcement learning. *arXiv preprint arXiv:1912.01588*, 2019.

[6] Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*, pages 1282–1289, 2019.

[7] Will Dabney, Mark Rowland, Marc Bellemare, and Rémi Munos. Distributional reinforcement learning with quantile regression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[8] Thibault De Swarte, Omar Boufous, and Paul Escalle. Artificial intelligence, ethics and human values: the cases of military drones and companion robots. *Artificial Life and Robotics*, 24(3):291–296, 2019.

[9] Deepmind. Alphago teach. `https://alphagoteach.deepmind.com/`, 2017.

[10] Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

[11] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, pages 1407–1416, 2018.

[12] Forbes. Robots have become an essential part of the war against covid-19. 2021.

[13] Susanne Frennert and Britt Östlund. Seven matters of concern of social robots and older people. *International Journal of Social Robotics*, 6(2):299–310, 2014.

[14] Anirudh Goyal, Alex Lamb, Jordan Hoffmann, Shagun Sodhani, Sergey Levine, Yoshua Bengio, and Bernhard Schölkopf. Recurrent independent mechanisms. *arXiv preprint arXiv:1909.10893*, 2019.

[15] Evan Greensmith, Peter L Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5(Nov):1471–1530, 2004.

[16] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[17] Jens Hoefinghoff, Astrid Rosenthal-von Der Pütten, Josef Pauli, and Nicole Krämer. "yes dear, that belongs into the shelf!"-exploratory studies with elderly people who learn to train an adaptive robot companion. In *International Conference on Social Robotics*, pages 235–244. Springer, 2015.

[18] Maximilian Igl, Kamil Ciosek, Yingzhen Li, Sebastian Tschiatschek, Cheng Zhang, Sam Devlin, and Katja Hofmann. Generalization in reinforcement learning with selective noise injection and information bottleneck. In *Advances in Neural Information Processing Systems*, pages 13956–13968, 2019.

[19] Maximilian Igl, Gregory Farquhar, Jelena Luketina, Wendelin Boehmer, and Shimon Whiteson. The impact of non-stationarity on generalisation in deep reinforcement learning. *arXiv preprint arXiv:2006.05826*, 2020.

[20] Arthur Juliani, Ahmed Khalifa, Vincent-Pierre Berges, Jonathan Harper, Ervin Teng, Hunter Henry, Adam Crespi, Julian Togelius, and Danny Lange. Obstacle tower: A generalization challenge in vision, control, and planning. *arXiv preprint arXiv:1902.01378*, 2019.

[21] Niels Justesen, Ruben Rodriguez Torrado, Philip Bontrager, Ahmed Khalifa, Julian Togelius, and Sebastian Risi. Illuminating generalization in deep reinforcement learning through procedural level generation. *arXiv preprint arXiv:1806.10729*, 2018.

[22] Yuji Kanagawa and Tomoyuki Kaneko. Rogue-gym: A new challenge for generalization in reinforcement learning. In *2019 IEEE Conference on Games (CoG)*, pages 1–8. IEEE, 2019.

[23] Michael Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. *arXiv preprint arXiv:2004.14990*, 2020.

[24] Adam Laud and Gerald DeJong. The influence of reward on the speed of reinforcement learning: An analysis of shaping. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 440–447, 2003.

[25] Adam Daniel Laud. Theory and application of reward shaping in reinforcement learning. Technical report, 2004.

[26] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[27] Alex Nichol, Vicki Pfau, Christopher Hesse, Oleg Klimov, and John Schulman. Gotta learn fast: A new benchmark for generalization in rl. *arXiv preprint arXiv:1804.03720*, 2018.

[28] Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn. *arXiv preprint arXiv:1602.04621*, 2016.

[29] Aravind Rajeswaran, Kendall Lowrey, Emanuel V Todorov, and Sham M Kakade. Towards generalization and simplicity in continuous control. In *Advances in Neural Information Processing Systems*, pages 6550–6561, 2017.

[30] M. Sadler and N. Regan. *Game Changer: AlphaZero's Groundbreaking Chess Strategies and the Promise of AI*. New in Chess, 2019.

[31] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

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

[33] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017.

[34] Xingyou Song, Yilun Du, and Jacob Jackson. An empirical study on hyperparameters and their interdependence for rl generalization. *arXiv preprint arXiv:1906.00431*, 2019.

[35] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30. IEEE, 2017.

[36] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[37] Kaixin Wang, Bingyi Kang, Jie Shao, and Jiashi Feng. Improving generalization in reinforcement learning with mixture regularization. *arXiv preprint arXiv:2010.10814*, 2020.

[38] Shimon Whiteson, Brian Tanner, Matthew E Taylor, and Peter Stone. Protecting against evaluation overfitting in empirical reinforcement learning. In *2011 IEEE symposium on adaptive dynamic programming and reinforcement learning (ADPRL)*, pages 120–127. IEEE, 2011.

[39] Mitchell Wortsman, Kiana Ehsani, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. Learning to learn how to learn: Self-adaptive visual navigation using meta-learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6750–6759, 2019.

[40] Amy Zhang, Nicolas Ballas, and Joelle Pineau. A dissection of overfitting and generalization in continuous reinforcement learning. *arXiv preprint arXiv:1806.07937*, 2018.

[41] Chiyuan Zhang, Oriol Vinyals, Remi Munos, and Samy Bengio. A study on overfitting in deep reinforcement learning. *arXiv preprint arXiv:1804.06893*, 2018.

[42] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3357–3364. IEEE, 2017.