Dynamic Value Estimation for Single-Task Multi-Scene Reinforcement Learning

Jaskirat Singh, Liang Zheng
College of Engineering and Computer Science
Australian National University
Canberra, Australia
{jaskirat.singh,liang.zheng}@anu.edu.au

Abstract

Training deep reinforcement learning agents on environments with multiple levels / scenes / conditions from the *same task*, has become essential for many applications aiming to achieve generalization and domain transfer from simulation to the real world (Wortsman et al. 2019; Cobbe et al. 2019b). 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 argue 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 $\mathcal{M}$. We further demonstrate that the true joint value function (given a state), 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 an attention mechanism over multiple value function hypothesis / modes. The resulting agent is able to learn a more accurate and scene-specific value function (and hence the advantage function), leading to a lower sample variance. Our proposed approach is simple to accommodate with several existing implementations (like PPO, A3C) and results in consistent improvements for a range of OpenAI ProcGen environments and the AI2-THOR framework based visual navigation benchmark.

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 (Rajeswaran et al. 2017; Zhang, Ballas, and Pineau 2018; Whiteson et al. 2011). 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 (Wortsman et al. 2019; Cobbe et al. 2019b; Zhu et al. 2017; Justesen et al. 2018; Cobbe et al. 2019a; Kanagawa and Kaneko 2019; Juliani et al. 2019). Although incorporating a wider source of data distribution in the training itself has shown promise in bridging the train and test performance,

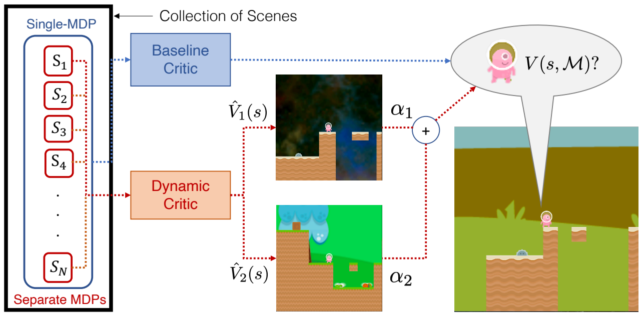

Figure 1: **Overview.** Traditional methods treat the set of training scenes as a single MDP and thus learn a single scene-generic value function estimate. In this paper, we argue that the sample variance is best minimized by treating each scene as a separate MDP and learning a scene-specific value function $V(s, \mathcal{M})$. However, this is challenging since the operation level is not known to the agent during training. To this end, the dynamic model attempts to approximate the joint value function by compressing the information from environment scenes into value estimates $\hat{V}_i(s)$ for a few representative / basis MDPs. These basis estimates are then combined based on the similarities ($\alpha_i \in [0, 1]$) between the current and the basis MDPs, to yield the final value function estimate.

the inclusion of multiple scenes, each defined by a distinct underlying MDP, significantly increases the variance of samples collected for policy gradient computations (Cobbe et al. 2019a; Song, Du, and Jackson 2019).

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, (Zhu et al. 2017; Wortsman et al. 2019) use multiple asynchronous actor critic (A3C) models when training on the AI2-THOR framework based visual navigation task. Similarly, (Cobbe et al. 2019a,b) deploy parallel workers to stabilize policy gradients for multi-level training on procedurally-generated game environments (Justesen et al. 2018). In most of these methods, a significant drop in both stability and final performance is observed on reducing the number of parallel workers.

Most RL generalization benchmarks (Nichol et al. 2018;

Preprint. Under Review.

Zhang et al. 2018; Cobbe et al. 2019b; Igl et al. 2019) *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. However, as we shall discuss in Section 2.2, the theoretical lower bound for sample variance in a multi-scene environment can be achieved 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 $\mathcal{M}$. For the latter treatment, we refer to the corresponding collection of scenes as a *multiple-MDP environment*.

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 this end, we propose a dynamic critic model, based on the multi-modal nature of the true joint value function distribution (refer Fig. 2) across different scenes. The proposed model enables the agent to learn a much better MDP-specific value function (and hence the advantage function), leading to lower sample variance for policy updates. The final proposed model mimics a memory fallback strategy frequently invoked in human learning, where the learning agent falls back on its knowledge about critical but similar experiences (MDPs) to judge the current situation. Figure 1 presents an overview of our method.

To summarize, the main contributions of this paper are:

- Demonstrate the theoretical shortcoming in the traditional value estimation approach of learning a scene-generic value function $V(s)$ for *multiple-MDP environments*.
- Show that the true scene-specific value function distribution is best described using a mixture model with multiple dominant modes, that are not fully captured by the current CNN or LSTM based critic networks.
- Propose an alternative critic model which approximates the true multi-modal value function through an attention mechanism over multiple value function hypothesis / modes.

2 Preliminaries

2.1 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 set $\boldsymbol{\mathcal{M}}$, *i.e.* $\mathbf{E}_{\tau,\mathcal{M}}[\mathcal{R}_{\tau,\mathcal{M}}]$.

2.2 Variance Reduction in Policy Gradient Algorithms

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, (Greensmith, Bartlett, and Baxter 2004; Schulman et al. 2015),

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

where $\psi(s, a) = Q(s, a) - V(s)$ is known as the advantage function. Similarly for a multiple-MDP environment, it can be shown that the optimal formulation for minimizing total sample variance is given by[1],

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

where $\psi(s, a, \mathcal{M}) = Q(s, a, \mathcal{M}) - V(s, \mathcal{M})$. Here $Q(s, a, \mathcal{M})$ & $V(s, \mathcal{M})$ represent the action-value and value function respectively for the particular MDP $\mathcal{M}$. However, since most of the times 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, which is essentially an estimate of the global average over the underlying scene-specific value functions $\{V_{\mathcal{M}_1}(s), V_{\mathcal{M}_2}(s), ..., V_{\mathcal{M}_N}(s)\}$.

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 Our method

3.1 Ambiguity in Value Estimation

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 best described by a multi-modal distribution.

Empirical Demonstration on OpenAI CoinRun[1]. 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 (Cobbe et al. 2019b). 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. 2. We clearly observe that the true value function estimates form multiple clusters that are not captured by traditional CNN or LSTM based critic networks.

3.2 Dynamic Value Function Estimation

The equivalent sample variance (ν) for policy gradients defined by Eq. 2, can be approximated as,

$$\nu \approx \kappa \, . \, \mathbf{E}_{s,a,\mathcal{M}}\left[\psi^2(s, a, \mathcal{M})\right] \qquad (3)$$

$$= \kappa \, . \, \mathbf{E}_{s,a,\mathcal{M}}\left[\left(Q(s, a, \mathcal{M}) - \hat{V}(s, \mathcal{M})\right)^2\right], \qquad (4)$$

where $\kappa = \mathbf{E}_{s,a,\mathcal{M}}\left[(\nabla_\theta \log \pi(a|s))^2\right]$ and $\hat{V}(s, \mathcal{M})$ represents the predicted value function. Now, using the true value function $V(s, \mathcal{M}) = \mathbf{E}_a\left[Q(s, a, \mathcal{M})\right]$, Eq. 4 can be

[1]We refer the readers to the supplementary materials for detailed explanations.

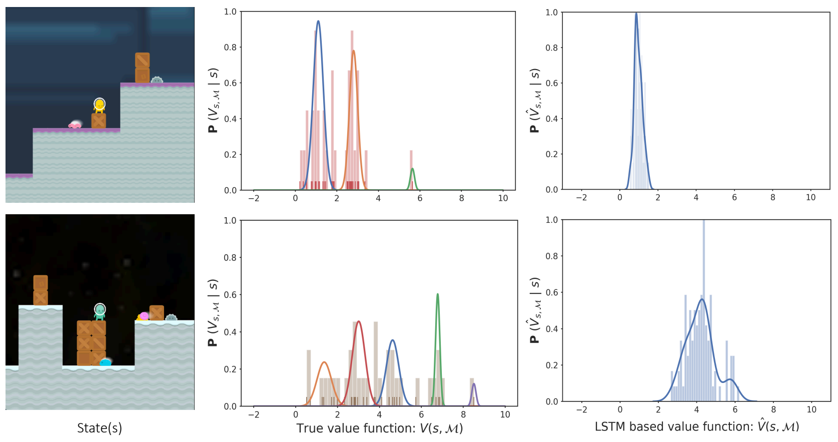

Figure 2: (**Column 1-2**:) Demonstrating multi-modal nature of the true value function distribution, for an intermediate policy π, on a set of randomly selected 50 levels from the CoinRun environment. The true value estimate for a state image shown on the left can be characterized by one of the many 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.

decomposed as,

$$\nu \approx \underbrace{\kappa\,.\,\mathbf{E}_{s,a,\mathcal{M}}\left[(Q(s,a,\mathcal{M})-V(s,\mathcal{M}))^2\right]}_{\text{minimal possible variance}} + \underbrace{\kappa\,.\,\mathbf{E}_{s,\mathcal{M}}\left[\left(V(s,\mathcal{M})-\hat{V}(s,\mathcal{M})\right)^2\right]}_{\text{prediction error}} + \cancelto{0}{(\ldots)}. \quad (5)$$

Thus, we see that the policy gradient sample variance can be minimized by reducing the error between the true value function $V(s,\mathcal{M})$ and the predicted estimate $\hat{V}(s,\mathcal{M})$. While the exact estimation of the true value function is infeasible without knowledge of MDP $\mathcal{M}$, we use the results of Section 3.1, 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.

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

$$P\left(V(s_t,\mathcal{M})|s_t\right) = \sum_{i=1}^{N_b} p_i\,\mathcal{N}(V(s_t,\mathcal{M})|\mu_i,\sigma_i^2), \quad (6)$$

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

$$\hat{V}(s_t,\mathcal{M}) = \sum_{i=1}^{N_b} \alpha_i(s_t,\mathcal{M})\,\mu_i(s_t),$$
$$s.t. \quad \alpha_i > 0, \quad \sum_i^{N_b} \alpha_i = 1. \quad (7)$$

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 the state, episode trajectory pairs $\{s_t,\tau^{t-}\}$ (τ^{t-} : $\{s_0,a_0,....s_{t-1}\}$ is the trajectory till time $t-1$), using a Long Short Term Memory (LSTM) network.

3.3 Interpretation as Learning Basis MDPs

The final dynamic value estimation model resulting from Eq. 7 can also be interpreted as learning a set of basis value functions $\{\mu_1(s),\mu_2(s),...,\mu_{N_b}(s)\}$. These can be further thought of as estimates of the value functions belonging to a set of basis MDPs $\boldsymbol{\mathcal{M}}_b$: $\{\mathcal{M}_{b_1},\mathcal{M}_{b_2},...,\mathcal{M}_{b_{N_b}}\}$, which might or might not be a subset of the original MDP set $\boldsymbol{\mathcal{M}}$. Eq. 7 can thus be written as,

$$\hat{V}(s_t,\mathcal{M}) = \sum_{i=1}^{N_b} P(\mathcal{M}_{b_i}|s_t,\tau^{t-})\,\hat{V}(s_t,\mathcal{M}_{b_i}). \quad (8)$$

Intuitively speaking, given a state of confusion, the agent relies on the value estimates for the basis MDPs, along with its past experience from the current episode, to form an estimate of the value function for the current state. This stems from the fact that not all levels in a game are critical for learning and often have repeated situations. The set of basis MDPs thus reflects a set of levels with varied patterns that are critical for effective game play.

4 Related Work

Meta Reinforcement Learning. (Duan et al. 2016; Wang et al. 2016) 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. 2), and thus fails to capture the multi-modal nature of true value function distribution $V(s,\mathcal{M})$. 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[2].

Uncertainty and Multi-Modal Posterior in Deep Networks. The lack of a measure of confidence in regression based predictions (like the value function in RL) has been a

[2]In this paper, we occasionally use the term cluster while referring to the value function hypothesis, since ideally, each hypothesis represents the mean of a distinct cluster in the true value distribution.

noted problem with deep neural networks. One of the ways to handle uncertainty is to model the underlying distribution parameters as a mixture model. To this end, Mixture Density Networks (MDNs) (Bishop 1994) provide a standard approach for learning the concentration parameters of a mixture model in a neural network setting. For instance, (Bui et al. 2020) use MDNs with a Relaxed-WTA (Rupprecht et al. 2017) approach to learn multiple hypothesis for camera pose parameters in ambiguous scenes. While interesting, we find that optimizing the maximum likelihood objective under changing data distributions (as is common in RL) is highly unstable. We instead directly predict multiple value function hypothesis, and combine them using attention parameters (α_i), which are learned from the state-trajectory tuple $\{s_t, \tau^{t-}\}$.

Generalization in Reinforcement Learning. Recently, multi-scene environments have been extensively used to study and address the problem of overfitting in RL. (Cobbe et al. 2019b) deploy standard regularization techniques from supervised learning like dropout, batch-normalization, L2-regularization to counter overfitting when training on the multi-scene CoinRun environment. (Rajeswaran et al. 2016) learn robust policies over an ensemble of scenes by formulating the overall objective as the expected reward over scenes with the worst performance. Noise injection techniques like (Igl et al. 2019; Tobin et al. 2017) add noise to the model parameters in order to improve the generalization capability. While effective, these works are seen to reduce overfitting at some expense to the training performance. Our work is different as it doesn't focus on generalization specifically, but improves both training and test time performance, by learning a better value function estimate.

In addition to the above, (Liang et al. 2018) propose an attention-based value function (AVF) network for model-based reinforcement learning. However, they consider an attention over past trajectory states, whereas we consider an attention over multiple value function hypotheses.

5 Evaluation on OpenAI Procgen Environments

5.1 Experimental design

In order to do a fair estimate of the benefits of the dynamic model, we adhere to the following three configurations for training and testing:

CNN-LSTM Baseline. The baseline model closely follows[3] the one described in (Cobbe et al. 2019a). Both actor and critic share an IMPALA-CNN network (Espeholt et al. 2018) modeled in the form of an LSTM (Hochreiter and Schmidhuber 1997). The output from the LSTM is then fed to both actor and critic separately, and is followed by a single fully-connected (FC) layer to compute the action scores and value function, respectively. The choice of hyper parameters is kept similar to that described in (Cobbe et al. 2019a) in order to allow for comparison on the baseline values [3], though, local hyper-parameter search was performed for each game to achieve the best average reward score.

Dynamic. The dynamic model is quite similar to the baseline and only requires few changes in the critic network to model the dynamic estimate described by Eq. 8. The probabilities $P(\mathcal{M}_{b_i}|s_t, \tau^{t-})$ are modeled using a fully connected layer followed by a softmax function, while the mean estimates $\hat{V}(s, \mathcal{M}_{b_i})$ are learned using a single fc layer. The number of clusters N_b, is treated as a hyperparameter[4] and is empirically determined using trial and error. For most ProcGen environments, the optimal choice for N_b was found to lie within the range $[2, 10]$.

Control. We design the control network to ensure that the performance gain from the dynamic model is not resulting from either increase in the number of parameters (0.41% increase) or changes in hyper-parameter selection. The increased number of parameters is compensated by adding a hidden layer with size $N_c = 2N_b$ in the baseline network.

All 3 configurations are trained using the Proximal policy optimization (PPO) (Schulman et al. 2017) algorithm. The algorithm is ran with 4 parallel workers 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. Unless otherwise specified, the final reward scores are reported as the average across 4 runs and use 500 levels for training.

5.2 Results

Despite the simplicity of the dynamic model, our method consistently outperforms the LSTM-PPO baseline models over a range of ProcGen environments. The improvements in performance can be seen in terms of both sample efficiency and the final reward score (Fig. 3). For instance, our method results in an 18.2% and 32.3% increase in the average episode reward on the CoinRun and CaveFlyer environments, respectively.

We also report the trend between dynamic model gains and the number of training levels in Fig. 3. On average, the dynamic model leads to significant performance gains till 1k training levels. After this point, generalization effect kicks in, while the variance introduced in the value function estimates due to the increased number of levels saturates [5].

For the sake of completeness, we report the final scores for the baseline LSTM-PPO and the dynamic model on 10 OpenAI ProcGen environments in Table 1.

6 Evaluation on the Visual Navigation Task

Another task which heavily relies on training using multiple-scene environments is visual navigation. Training an agent in different room setup and lighting conditions helps it generalize across the scene intricacies and focus on the bigger picture. Such behavior is desirable especially in experiments

[3] *Cobbe et al.*(Cobbe et al. 2019a) use an IMPALA CNN network as the baseline, while we use an IMPALA CNN-LSTM network.

[4] The Bayesian optimal choice for the number of clusters N_b can be determined by minimizing the Akaike Information Criterion (AIC) (Akaike 1987; Forster and Sober 2011) for the learned value function distribution. We demonstrate this in Section 7.1.

[5] We refer the readers to the supplementary materials for further explanations.

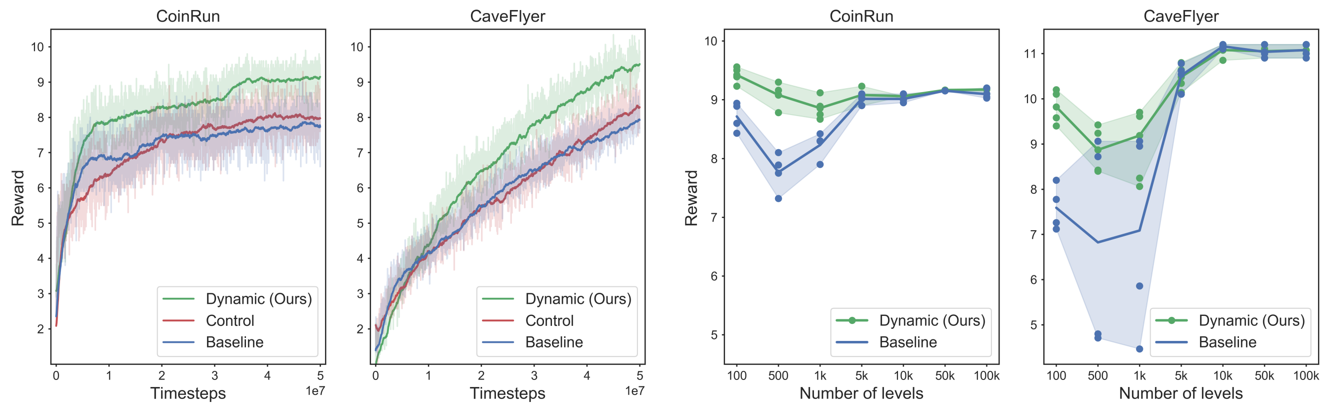

Figure 3: (**Left:**) Performance comparison for baseline, dynamic and control configurations of the critic network, illustrating differences in sample efficiency and total reward. (**Right:**) Performance comparison trend over the number of training levels.

ENVIRONMENT	CNN-LSTM PPO	DYNAMIC PPO
COINRUN	7.75	**9.16**
CAVEFLYER	6.82	**9.02**
DODGEBALL	8.43	**9.68**
PLUNDER	5.88	**7.21**
BIGFISH	15.41	**18.20**
JUMPER	**6.61**	6.52
CLIMBER	7.50	**8.14**
FRUITBOT	4.21	**10.47**
CHASER	7.41	**9.64**
BOSSFIGHT	10.33	**10.75**

Table 1: Final reward comparison between baseline CNN-LSTM PPO and Dynamic PPO on 10 ProcGen environments.

with high domain transfer, for instance, when the agent is trained in simulation but tested in the real world.

6.1 Task Definition

The task of visual navigation consists of a set of scenes $\boldsymbol{\mathcal{S}} = \{\mathcal{S}_1, \mathcal{S}_2, ...\mathcal{S}_n\}$, and a set of possible object classes $\boldsymbol{\mathcal{O}} = \{\mathcal{O}_1, \mathcal{O}_2..\mathcal{O}_m\}$. Note that the set $\boldsymbol{\mathcal{S}}$ is just another annotation for the MDP set $\boldsymbol{\mathcal{M}} = \{\mathcal{M}_1, \mathcal{M}_2, ...\mathcal{M}_n,\}$ as each scene is characterized by a distinct underlying MDP. Each scene $\mathcal{S}_i$ is characterized by a different room setup, distribution of objects and lighting conditions.

A single navigation task $\mathcal{T}$ consists of an agent with action space $\mathcal{A}$ situated in a random position p within the one of the scenes $\mathcal{S}_i$. The goal of the agent is to reach an instance of the target class $\mathcal{O}_k$ (given as a Glove embedding (Pennington, Socher, and Manning 2014)) within a certain number of steps. The agent then continues interacting with the environment using a policy π_θ until it chooses a termination action. The episode is considered a success, if and only if, at the time of termination, the target object is sufficiently close and in the field of view of the agent.

6.2 Experimental design

The performance on the visual navigation task is measured using 3 evaluation metrics: Success weighted by Path Length (SPL), Success rate and the total reward. SPL (Anderson et al. 2018) measures the navigation efficiency of the agent as $\frac{1}{N}\sum_{i=1}^{N} S_i \frac{L_i}{\max(P_i, L_i)}$. Success rate and the total reward are simply the average rate of success: $\frac{1}{N}\sum_{i=1}^{N} S_i$ and the average episode reward: $\frac{1}{N}\sum_{i=1}^{N} \mathcal{R}_i$, respectively.

In above, N is the number of episodes, $S_i \in \{0, 1\}$ indicates the success of an episode, P_i is the path length, L_i is the optimal path length to any instance of the target object class in that scene, and $\mathcal{R}_i$ is the episode reward. The baseline results are reported using the non-adaptive and self-adapting (SAVN) A3C models from (Wortsman et al. 2019) with 12 asynchronous workers. We then modify the critic network as per Eq. 8 to get the dynamic version for both baselines.

The agent is trained using the AI2-THOR environment (Kolve et al. 2017) which consists of 120 distinct scenes. A train/val/test split of 80:20:20 is used for selecting the best model based on highest success rate. Note that the final results are reported after 5M episodes of training which equates to $\approx$ 96 hours of training time on 2 GeForce RTX 2080 Ti GPUs.

6.3 Results

As described in Table 2, the dynamic A3C model results in significant improvements across all 3 performance metrics for both self-adaptive and non-adaptive baselines. Furthermore, the dynamic model provides a huge boost in the sample efficiency (refer Fig. 4). For instance, the dynamic nonadaptive A3C model reaches a test success rate of $\approx$ 30% after only 2M episodes which in contrast with the non-adaptive baseline takes around 4M episodes. Similarly, the dynamic model for self-adaptive A3C achieves a test success rate of $\approx$ 40% after only 1M episodes while the corresponding baseline has a success rate of only 30.8%. Given the huge amounts of training time required in simulation, the dynamic model can be a real asset for applications with stricter time constraints. For instance, the high sample efficiency is useful for quick

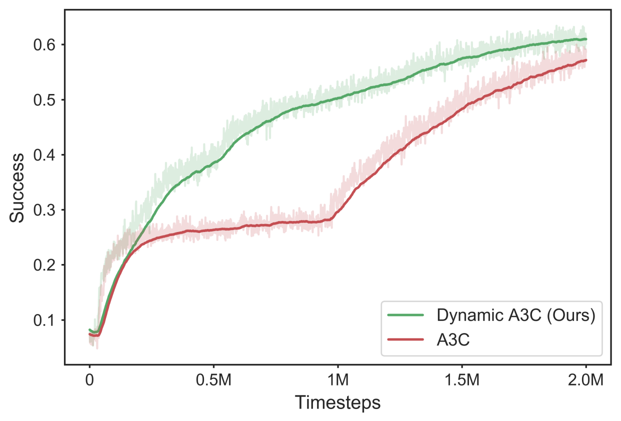

Figure 4: Comparing sample efficiency at training time for A3C and Dynamic A3C model on the visual navigation task.

fine-tuning in mobile navigation robots operating in the real world (Chancán and Milford 2019).

Method	SPL	Success	Total Reward
A3C	14.3	31.8	1.413
Dynamic A3C	**15.4**	**36.5**	**1.638**
SAVN	**15.19**	37.1	1.652
Dynamic SAVN	14.81	**38.7**	**1.824**

Table 2: Comparison on key evaluation metrics for visual navigation. We observe clear improvements when using the dynamic critic network with both the non-adaptive and adaptive (SAVN) baselines.

7 Analysis

7.1 Finding the Optimal Number of Dynamic Hypotheses / Clusters

A critical component in training the dynamic model is the selection of number of hypotheses / clusters N_b. While it is possible to treat it as another hyper-parameter, we present an alternate approach for the same. This approach not only gives us the number of optimal clusters, but also strengthens our belief in the function of the proposed critic network.

Recall that, given our motivation from Section 3.1, we believe that the *true* conditional value function distribution resembles a Gaussian Model Mixture (GMM). Our final model needs to learn a value function with similar distribution, not just across different MDPs belonging to the set $\boldsymbol{\mathcal{M}}$, but also for all states $s \in \mathcal{S}$. Thus, to find out the optimal number of value hypothesis / Gaussian clusters, we begin by approximating the multi-variate distribution $V(s, \mathcal{M})$ using discrete samples $\{s_j, \mathcal{M}_i, V(s_j, \mathcal{M}_i)\}$ for $s_j \in \mathcal{S}$ & $\mathcal{M}_i \in \boldsymbol{\mathcal{M}}$.

For the sample collection, we first obtain an intermediate policy π and the corresponding scene-specific true value estimation networks $\hat{V}_i(s)$ for a random selection of 500 levels from the CoinRun ProcGen environment, using a level-wise critic-finetuning strategy outlined in the supplementary material. Since incorporating the entire state space is infeasible, we try to minimize the error by sampling a large collection of 1000 states from the different levels using the common policy π. Next for each of these states $s_j,\ j \in [1, 1000]$, we obtain the corresponding level-specific value function estimates using $\hat{V}_i(s_j), i \in [1, 500]$.

The generated dataset representing samples from the true multi-variate distribution $V(s, \mathcal{M})$ has shape $[N \times K]$, where N is number of levels and K is the number of states. This dataset is then fitted using a GMM with variable number of components $C \in [1, 10]$. We finally use the Akaike Information Criterion (AIC) from (Akaike 1987), to determine the best selection for the number of components (Fig. 5).

As shown in Fig. 5, the optimum number of clusters (point of minima in the AIC/N curve) increases with the number of training levels. We also note that the AIC/N curve becomes less steep, as the number of training levels increases. This implies that given a sufficient number of training levels, the dynamic model's performance shows low sensitivity (higher robustness) to the selection of the hyper-parameter N_b.

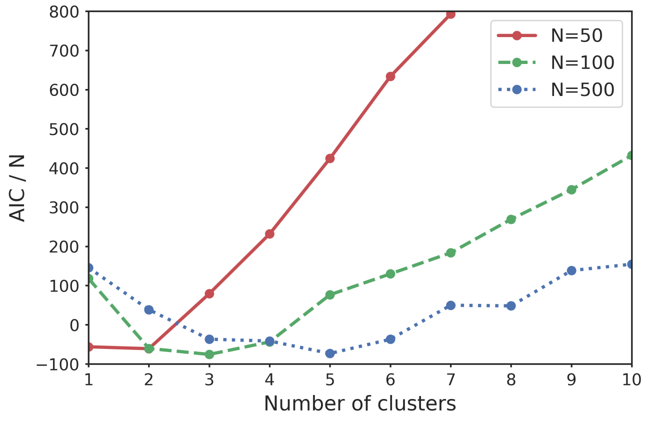

Figure 5: **Selecting number of dynamic hypotheses.** AIC scores for different number of clusters and training levels N. The local minima represents the Bayesian optimal choice (Forster and Sober 2011) for number of dynamic hypotheses / clusters N_b.

7.2 Visualizing Representative MDPs

Recall from Section 3.3, the dynamic model can be interpreted as learning the value function estimates for a set of basis MDPs $\boldsymbol{\mathcal{M}}_b$, which may or may not belong to the original MDP set $\boldsymbol{\mathcal{M}}$. In this section, we present a visualization analysis for what each basis/cluster might represent.

We first begin by defining the term "*confusion*". An agent is said to be in a state of confusion if it is unsure of the cluster / basis MDP to which the current $\{s_t, \tau^{t-}\}$ pair belongs. As per Eq. 7, the probability of an agent (with state pair $\{s_t, \tau^{t-}\}$) choosing the i^{th} hypothesis / cluster is given by $\alpha_i(s_t, \tau^{t-}), i \in [1, N_b]$. Thus, a state of complete confusion would be characterized by a uniform distribution across the

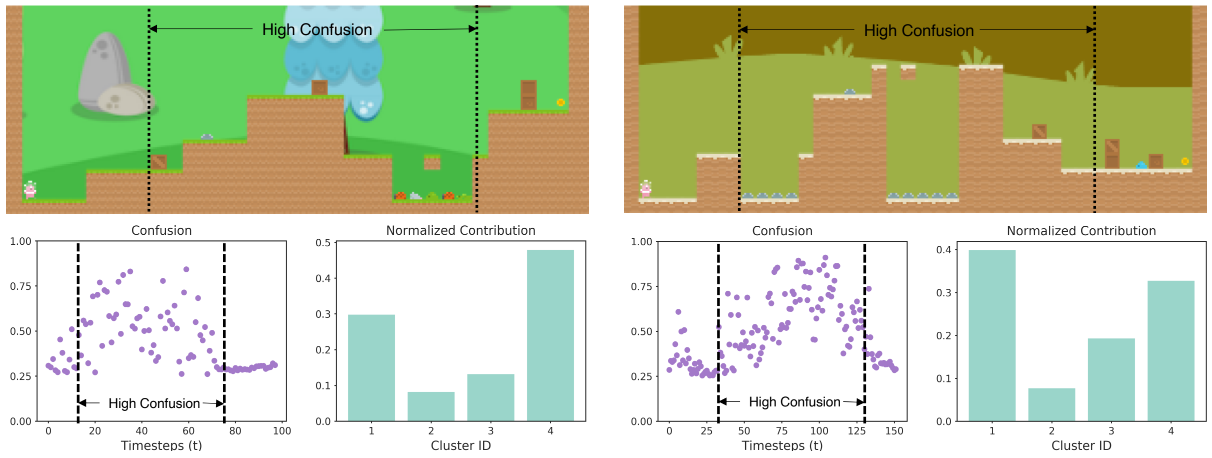

Figure 6: Analysing "confusion" distribution and cluster specific "contribution" scores for different levels from the CoinRun environment. The states with high confusion usually occur in the middle of a level where most of the obstacles are present. We also note that different clusters contribute differently to value estimation depending on the underlying scene. The cluster with highest contribution can be thought of as representing a basis MDP most similar to the current level.

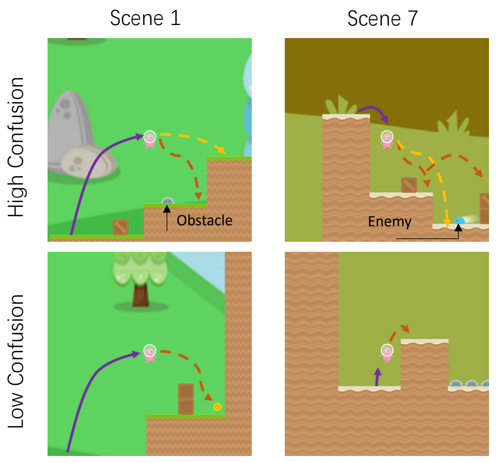

Figure 7: High and low confusion states for different scenes from the CoinRun environment. The red and orange arrows indicate the possible paths that the agent might take (in a possibly different scene/MDP) to avoid the obstacle/enemy. Each path corresponds to a distinct value estimate for a basis MDP and thus, more possible paths indicate higher confusion.

parameters α_i, while a sharp spike in α_i distribution indicates low confusion. Mathematically, confusion $\delta \in [0, 1]$ for the $\{s_t, \tau^{t-}\}$ pair can be defined as,

$$\delta(s_t, \tau^{t-}) = \frac{1}{N_b . \sum_i \alpha_i^2(s_t, \tau^{t-})}. \tag{9}$$

Fig. 7 shows some states of maximum and minimum confusion for two randomly selected levels from the CoinRun environment. States with high confusion usually occur around tricky obstacles or jumps, while low confusion states usually occur in simple scenarios, happening mostly at the beginning or end of a level (refer Fig. 6).

Next we define the "*contribution*" of a cluster / hypothesis to the overall value function estimation while navigating a particular scene / level. Intuitively, a cluster with a higher average α_i across the episode, should have a higher contribution. We would also like to provide more weight-age to cluster contributions for states with high confusion, which as shown in Fig. 6, are usually critical for optimal game play. Hence, given an episode trajectory $\tau : \{s_0, a_0, ..., s_T\}$, we define the weighted contribution ρ_i for the i^{th} cluster as,

$$\rho_i(\tau) = \frac{1}{T} \sum_{t=1}^{T} \delta(s_t, \tau^{t-})\ \alpha_i(s_t, \tau^{t-}). \tag{10}$$

Fig. 6 shows the distribution of confusion and contribution across different game levels. We also observe that, depending on the type of obstacles present, different clusters *contribute* differently to the overall value estimation on each scene. It can be argued that the cluster with the highest contribution represents an underlying basis MDP with scene features most similar to the associated level.

8 Conclusion

This paper proposes a dynamic value estimation strategy for multi-scene reinforcement learning. We demonstrate that despite the lack of knowledge about the operational scene / level, a more accurate scene-specific value estimate can be learned by utilizing the clustering observed in the value function distribution across different scenes. Our dynamic critic network outperforms the current baselines in both sample efficiency and final reward across a range of multi-level ProcGen environments and the visual navigation task. Finally, we provide a mechanism for analysing the proposed model in terms of "confusion" and "contribution". This helps us visualize the representative cluster MDPs and gives deeper insights into what the dynamic agent truly learns.

Broader Impact

Our work is very theoretical in nature, but as a manuscript advancing the bounds of artificial intelligence, it is likely to have significant social impacts in the long term. One of the broader social impacts of our work can be seen in terms of its viability in promoting artificial intelligence. Recent large-scale exhibitions by OpenAI (OpenAI 2018; Berner et al. 2019) and DeepMind (Vinyals et al. 2019a,b) have shown that advances in game-playing attract interest of both technical and non-technical audiences alike. As demonstrated through evaluation on OpenAI ProcGen (Cobbe et al. 2019a) in Section 5, our method promises significant improvements in complex game-play and thus can be used to attract more people to this field.

Another societal impact can be envisioned in the field of robotics and social healthcare. An improvement in visual navigation performance within a controlled environment, as is proposed in this work, has the potential to be used in the development of caretaker robots for the elderly (Hoefinghoff et al. 2015). This can be associated with a number of ethical concerns as detailed in (Frennert and Östlund 2014; De Swarte, Boufous, and Escalle 2019). Nonetheless, we strongly believe that the positive benefits by far outweigh such concerns.

References

Akaike, H. 1987. Factor analysis and AIC. In *Selected papers of hirotugu akaike*, 371–386. Springer.

Anderson, P.; Chang, A.; Chaplot, D. S.; Dosovitskiy, A.; Gupta, S.; Koltun, V.; Kosecka, J.; Malik, J.; Mottaghi, R.; Savva, M.; et al. 2018. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757* .

Berner, C.; Brockman, G.; Chan, B.; Cheung, V.; Dębiak, P.; Dennison, C.; Farhi, D.; Fischer, Q.; Hashme, S.; Hesse, C.; et al. 2019. Dota 2 with Large Scale Deep Reinforcement Learning. *arXiv preprint arXiv:1912.06680* .

Bishop, C. M. 1994. Mixture density networks .

Bui, M.; Birdal, T.; Deng, H.; Albarqouni, S.; Guibas, L.; Ilic, S.; and Navab, N. 2020. 6D Camera Relocalization in Ambiguous Scenes via Continuous Multimodal Inference. *arXiv preprint arXiv:2004.04807* .

Chancán, M.; and Milford, M. 2019. From Visual Place Recognition to Navigation: Learning Sample-Efficient Control Policies across Diverse Real World Environments. *arXiv preprint arXiv:1910.04335* .

Cobbe, K.; Hesse, C.; Hilton, J.; and Schulman, J. 2019a. Leveraging Procedural Generation to Benchmark Reinforcement Learning. *arXiv preprint arXiv:1912.01588* .

Cobbe, K.; Klimov, O.; Hesse, C.; Kim, T.; and Schulman, J. 2019b. Quantifying Generalization in Reinforcement Learning. In *International Conference on Machine Learning*, 1282–1289.

De Swarte, T.; Boufous, O.; and Escalle, P. 2019. Artificial intelligence, ethics and human values: the cases of military drones and companion robots. *Artificial Life and Robotics* 24(3): 291–296.

Duan, Y.; Schulman, J.; Chen, X.; Bartlett, P. L.; Sutskever, I.; and Abbeel, P. 2016. Rl^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779* .

Espeholt, L.; Soyer, H.; Munos, R.; Simonyan, K.; Mnih, V.; Ward, T.; Doron, Y.; Firoiu, V.; Harley, T.; Dunning, I.; et al. 2018. IMPALA: Scalable Distributed Deep-RL with Importance Weighted Actor-Learner Architectures. In *International Conference on Machine Learning*, 1407–1416.

Forster, M.; and Sober, E. 2011. AIC scores as evidence: a Bayesian interpretation. In *Philosophy of Statistics*, 535–549. Elsevier.

Frennert, S.; and Östlund, B. 2014. Seven matters of concern of social robots and older people. *International Journal of Social Robotics* 6(2): 299–310.

Greensmith, E.; Bartlett, P. L.; and Baxter, J. 2004. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research* 5(Nov): 1471–1530.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8): 1735–1780.

Hoefinghoff, J.; Rosenthal-von Der Pütten, A.; Pauli, J.; and Krämer, N. 2015. "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*, 235–244. Springer.

Igl, M.; Ciosek, K.; Li, Y.; Tschiatschek, S.; Zhang, C.; Devlin, S.; and Hofmann, K. 2019. Generalization in reinforcement learning with selective noise injection and information bottleneck. In *Advances in Neural Information Processing Systems*, 13956–13968.

Juliani, A.; Khalifa, A.; Berges, V.-P.; Harper, J.; Teng, E.; Henry, H.; Crespi, A.; Togelius, J.; and Lange, D. 2019. Obstacle tower: A generalization challenge in vision, control, and planning. *arXiv preprint arXiv:1902.01378* .

Justesen, N.; Torrado, R. R.; Bontrager, P.; Khalifa, A.; Togelius, J.; and Risi, S. 2018. Illuminating generalization in deep reinforcement learning through procedural level generation. *arXiv preprint arXiv:1806.10729* .

Kanagawa, Y.; and Kaneko, T. 2019. Rogue-gym: A new challenge for generalization in reinforcement learning. In *2019 IEEE Conference on Games (CoG)*, 1–8. IEEE.

Kolve, E.; Mottaghi, R.; Han, W.; VanderBilt, E.; Weihs, L.; Herrasti, A.; Gordon, D.; Zhu, Y.; Gupta, A.; and Farhadi, A. 2017. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474* .

Liang, X.; Wang, Q.; Feng, Y.; Liu, Z.; and Huang, J. 2018. VMAV-C: A deep attention-based reinforcement learning algorithm for model-based control. *arXiv preprint arXiv:1812.09968* .

Nichol, A.; Pfau, V.; Hesse, C.; Klimov, O.; and Schulman, J. 2018. Gotta learn fast: A new benchmark for generalization in rl. *arXiv preprint arXiv:1804.03720* .

OpenAI. 2018. OpenAI Five. https://blog.openai.com/openai-five/.

Pennington, J.; Socher, R.; and Manning, C. D. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 1532–1543.

Rajeswaran, A.; Ghotra, S.; Ravindran, B.; and Levine, S. 2016. Epopt: Learning robust neural network policies using model ensembles. *arXiv preprint arXiv:1610.01283* .

Rajeswaran, A.; Lowrey, K.; Todorov, E. V.; and Kakade, S. M. 2017. Towards generalization and simplicity in continuous control. In *Advances in Neural Information Processing Systems*, 6550–6561.

Rupprecht, C.; Laina, I.; DiPietro, R.; Baust, M.; Tombari, F.; Navab, N.; and Hager, G. D. 2017. Learning in an uncertain world: Representing ambiguity through multiple hypotheses. In *Proceedings of the IEEE International Conference on Computer Vision*, 3591–3600.

Schulman, J.; Moritz, P.; Levine, S.; Jordan, M.; and Abbeel, P. 2015. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438* .

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

Song, X.; Du, Y.; and Jackson, J. 2019. An Empirical Study on Hyperparameters and their Interdependence for RL Generalization. *arXiv preprint arXiv:1906.00431* .

Tobin, J.; Fong, R.; Ray, A.; Schneider, J.; Zaremba, W.; and Abbeel, P. 2017. 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)*, 23–30. IEEE.

Vinyals, O.; Babuschkin, I.; Chung, J.; Mathieu, M.; Jaderberg, M.; Czarnecki, W.; Dudzik, A.; Huang, A.; Georgiev, P.; Powell, R.; Ewalds, T.; Horgan, D.; Kroiss, M.; Danihelka, I.; Agapiou, J.; Oh, J.; Dalibard, V.; Choi, D.; Sifre, L.; Sulsky, Y.; Vezhnevets, S.; Molloy, J.; Cai, T.; Budden, D.; Paine, T.; Gulcehre, C.; Wang, Z.; Pfaff, T.; Pohlen, T.; Yogatama, D.; Cohen, J.; McKinney, K.; Smith, O.; Schaul, T.; Lillicrap, T.; Apps, C.; Kavukcuoglu, K.; Hassabis, D.; and Silver, D. 2019a. AlphaStar: Mastering the Real-Time Strategy Game StarCraft II. https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/.

Vinyals, O.; Babuschkin, I.; Czarnecki, W. M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D. H.; Powell, R.; Ewalds, T.; Georgiev, P.; et al. 2019b. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* 575(7782): 350–354.

Wang, J. X.; Kurth-Nelson, Z.; Tirumala, D.; Soyer, H.; Leibo, J. Z.; Munos, R.; Blundell, C.; Kumaran, D.; and Botvinick, M. 2016. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763* .

Whiteson, S.; Tanner, B.; Taylor, M. E.; and Stone, P. 2011. Protecting against evaluation overfitting in empirical reinforcement learning. In *2011 IEEE symposium on adaptive dynamic programming and reinforcement learning (ADPRL)*, 120–127. IEEE.

Wortsman, M.; Ehsani, K.; Rastegari, M.; Farhadi, A.; and Mottaghi, R. 2019. 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*, 6750–6759.

Zhang, A.; Ballas, N.; and Pineau, J. 2018. A dissection of overfitting and generalization in continuous reinforcement learning. *arXiv preprint arXiv:1806.07937* .

Zhang, C.; Vinyals, O.; Munos, R.; and Bengio, S. 2018. A study on overfitting in deep reinforcement learning. *arXiv preprint arXiv:1804.06893* .

Zhu, Y.; Mottaghi, R.; Kolve, E.; Lim, J. J.; Gupta, A.; Fei-Fei, L.; and Farhadi, A. 2017. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, 3357–3364. IEEE.