Deep Reinforcement Learning with Vector Quantized Encoding

Liang Zhang, Justin Lieffers, Adarsh Pyarelal
The University of Arizona
Tucson, AZ 85721 USA
\{liangzh, lieffers, adarsh\}@arizona.edu

Abstract

Human decision-making often involves combining similar states into categories and reasoning at the level of the categories rather than the actual states. Guided by this intuition, we propose a novel method for clustering state features in deep reinforcement learning (RL) methods to improve their interpretability. Specifically, we propose a plug-and-play framework termed vector quantized reinforcement learning (VQ-RL) that extends classic RL pipelines with an auxiliary classification task based on vector quantized (VQ) encoding and aligns with policy training. The VQ encoding method categorizes features with similar semantics into clusters and results in tighter clusters with better separation compared to classic deep RL methods, thus enabling neural models to learn similarities and differences between states better. Furthermore, we introduce two regularization methods to help increase the separation between clusters and avoid the risks associated with VQ training. In simulations, we demonstrate that VQ-RL improves interpretability and investigate its impact on robustness and generalization of deep RL.

Introduction

Human thinking reflects clustering characteristics in many modalities such as reasoning and planning. In the CartPole game [Barto, Sutton, and Anderson 1983], as shown in Figure 1, humans cannot estimate the values of the components of the game state with the same precision as a computer - however, they can make general classifications on states and suggest corresponding actions. For example, when the pole tilts to the left, we need to move the cart to the left to keep the pole balanced, and vice versa. In this process, our brain does not classify the various states of the pole to the left with arbitrary precision, but processes similar states as one category. This leads one to ask if we could exploit the clustering properties of feature distributions in neural networks in deep reinforcement learning (RL) problems to (i) explore the relationships between states and (ii) infer relationships between clusters and actions/policies. Doing so will help us understand more about the decision-making process in deep RL and the clustering relationships between states.

In the vein of understanding latent spaces of neural networks better, vector quantized variational autoencoders...
(VQ-VAE) (Oord, Vinyals, and Kavukcuoglu 2017) are a successful family of generative models that combine the variational autoencoder (VAE) framework with discrete latent representations through a novel parameterization of the posterior distribution given observations. VQ-VAEs provide an effective method for giving more access to the latent space of feature encodings. So far, most of the applications of VQ-VAEs have focused on generative models, such as generating images, audio, video, etc. (Oord, Vinyals, and Kavukcuoglu 2017) Gârbacea et al. 2019, Razavi, van den Oord, and Vinyals 2019; Tjandra, Sakti, and Nakamura 2020. Nevertheless, this encoding and training process can also bring us two advantages: First, VQ encoding generates latent feature spaces with improved clustering properties (see Figure 2), i.e., we can more effectively learn the commonality between features within a cluster, and better distinguish features between different clusters. Second, according to the t-distributed Stochastic Neighbor Embedding (t-SNE) (Van der Maaten and Hinton 2008) visualizations in (Mnih et al. 2015; Zahavy, Ben-Zrihem, and Mannor 2016), in the embedding space with dimensionality reduction guided by RL policy training, state features with the same or similar semantics are more likely to be clustered together. These properties provide the potential to implement our proposed clustering intuition in this study.

In order to apply this idea to RL decision-making processes, we introduce a novel learning pipeline, named vector-quantized reinforcement learning or VQ-RL in this work. In our experiments we will study the effectiveness of VQ-RL, analyzing if the improved clustering will lead to increased interpretability, robustness, and generalization. In summary, we make the following contributions in this paper:

- Inspired by the clustering nature of human reasoning, we propose a novel multi-task learning framework based on VQ encoding that improves the clustering properties and interpretability of features.
- We introduce two regularization methods to further improve the separation between clusters in the encoding space and avoid the risk in the codebook training that may be caused by large values.
- We design and perform three sets of experiments on a variety of domains with both continuous and discrete state spaces to demonstrate the effectiveness of our proposed architecture.

**Related Work**

Although deep neural networks have succeeded in many RL problems (Levine et al. 2016; Lee et al. 2019a; Jaderberg et al. 2019; Silver et al. 2016), their black-box characteristics make them difficult to interpret. Besides, it is still challenging to reach human levels of robustness and generalization. Therefore, research on extracting more interpretable, robust and generalized feature representations is actively underway.

**Auxiliary Tasks** Auxiliary tasks are often incorporated into deep RL to improve feature representation and performance. The unsupervised Pixel Control task, which predicts screen changes in discrete control environments, was applied in the UNREAL agent (Jaderberg et al. 2016). Oord, Li, and Vinyals (2018) proposed Contrastive Predictive Coding (CPC) to extract useful representations from high-dimensional data, which predicted the future latent space by using powerful autoregressive models. CPC demonstrated the ability to learn valuable representations while also achieving strong performance on 3D RL problems. Predictions of Bootstrapped Latents (PBL) (Guo et al. 2020) builds on multistep predictive representations of future observations and focuses on capturing structured information about environment dynamics, which improves the performance of state-of-the-art RL methods. To overcome the limitations of reward-driven feature learning in deep RL from images, the Augmented Temporal Contrast (ATC) (Stooke et al. 2021) method was proposed, which matches or outperforms end-to-end RL in most RL testbeds. Diuk, Cohen, and Littman (2008) introduced a representation based on objects and their interactions, which provides a natural way of modeling environments and offers significant generalization opportunities.

**Interpretability** The interpretability of deep RL has been extensively studied in recent years. Programmatically Interpretable Reinforcement Learning (PIRL) (Verma et al. 2018) represents policies using a high-level, domain-specific programming language designed to generate interpretable and verifiable agent policies. To improve the interpretability of the subtasks in hierarchical decision-making, Lyu et al. (2019) introduced symbolic planning into RL and proposed a framework of Symbolic Deep Reinforcement Learning (SDRL) that can handle both high-dimensional sensory inputs and symbolic planning. A soft attention model for reinforcement learning domains was introduced in (Mott et al. 2019) to show that the model learns to query separately about space and content (“where” vs. “what”). Some visualization techniques (Such et al. 2018; Wang et al. 2018; Zahavy, Ben-Zrihem, and Mannor 2016) for feature space have been proposed for better interpretability of RL methods, which showed that features with similar semantics locate closely. While these methods revealed that features in the embedding space have specific clustering properties, they do not explore how these clustering properties can be used to further enhance the performance of RL methods. VQ encoding directly acts on constructing the clustered embedding space, with VQ embeddings as the centers to attract the surrounding features into a smaller space, providing us the potential to cluster features with similar semantics.

**Robustness and generalization** An approach for robustness to action uncertainty was proposed in (Tessler, Efroni, and Mannor 2019), which provided a robust policy learning method and improved performance in the absence of perturbations. To increase robustness to noise and adversaries, Lütjens, Everett, and How (2020) introduced an online certified defense to the Deep Q-Network policy training. Wang, Liu, and Li (2020) studied deep RL models in noisy reward scenarios and developed a robust framework for these scenarios. Furthermore, (Cowbe et al. 2019) introduced Coin-Run, a benchmark for generalization in RL, and studied some factors in neural network training that may affect generalization. (Packer et al. 2018) proposed a benchmark and
experimental protocol and conducted a systematic empirical study. Some specific neural network training techniques have also been proposed in the literature to improve generalization in RL (Lee et al. 2019b, Zhang, Ballas, and Pineau 2018, Lgl et al., 2019).

**VQ-VAE** In recent years, VQ-VAE has been successfully applied to various tasks, such as high-resolution image generation (Razavi, van den Oord, and Vinyals 2019), video generation (Yan et al. 2021), speech coding (G@rbacea et al. 2019), etc. VQ-VAE has also been introduced into model-based deep RL problems to train transition models in some recent works (Robine, Uelwer, and Harmeling 2020). Unlike the above papers where VQ-VAE is utilized for different reconstruction or generation tasks, this paper leverages VQ encoding to cluster features in RL problems and train a policy-guided classification task based on encoded features.

**Method**

In this section, we explain our methodology and architecture, which builds on the VQ-VAE work proposed by Oord, Vinyals, and Kavukcuoglu (2017). Here we provide a brief overview of the relevant VQ-VAE equations and concepts.

The VQ-VAE model starts with an encoder network that maps an input \( x \) to a latent continuous representation \( E(x) \). After that, \( E(x) \) is quantized via the nearest neighbor look-up in the codebook, \( e \in R^K \times D \), where \( K \) is the number of vectors in the codebook and \( D \) is the dimension of the continuous embedding. The \( K \) embedding vectors are represented as an index of the codebook, \( e_i \in R^D, i \in 1 \ldots K \). This nearest neighbor vector is then fed into the decoder network to reconstruct the input \( x \). The following equation summarizes the process of the quantization of VQ encoding:

\[
\text{Quantize}(E(x)) = e_k, \\
k = \arg \min_j ||E(x) - e_j||
\]

The objective of the VQ-VAE training follows:

\[
\mathcal{L}(x, D(e)) = ||x - D(e)||^2 + \|s_g[E(x)] - e\|^2 \\
+ \beta \|s_g[e] - E(x)\|^2
\]

where \( E \) is the encoder and \( D \) is the decoder, \( s_g \) is a stop-gradient operator, and \( \beta \) is the weight preventing the encoder outputs from fluctuating between different code vectors.

**VQ-RL**

Deep RL architectures usually contain two modules: a feature extractor and a prediction network, whose designs vary according to the specific requirements of the problems being studied (Bellemare, Dabney, and Munos 2017, Kaiser et al. 2019, Arulkumaran et al. 2017). To improve the extracted features’ clustering properties in the embedding space, we introduce an auxiliary task with discretized VQ encoding to the classic RL pipeline. This auxiliary task aims to predict the maximum possible output action under the current policy by performing a classification task and will lead to clustering together features with the same maximum possible output, such as the example in Figure [1]. The architecture of our model is shown in Figure [3] and the visualization of vector quantizer is shown in Appendix.

The training process in VQ-VAE causes two effects: 1) The encoded features learn to be as close as possible to the closest embedding vector in the codebook, 2) the embedding vectors in the codebook learn to be as close as possible to all the features in their corresponding cluster. Theoretically, VQ encoding can help achieve clustering properties without performing additional unsupervised tasks. Similar to the encoding process in VQ-VAE, the vector quantizer in VQ-RL returns (i): the embedding in the codebook as the output that is closest to the input feature and (ii): the encoding index identifying the cluster.

Nevertheless, in the reconstruction task of VQ-VAE, features are segmented into multiple parts and VQ encoding is processed for each segment separately. For example, in the image reconstruction task in VQ-VAE (Oord, Vinyals, and Kavukcuoglu 2017), a \( 32 \times 32 \) embedding matrix with \( K = 512 \) for ImageNet is required. In order to cluster features in VQ-RL, we set the embedding vector dimension in the codebook to be the same as the input feature size.

The most straightforward choice for the number of embeddings in the codebook is the number of possible actions, e.g., the original CartPole domain (Barto, Sutton, and Anderson 1983) contains only two performable actions - thus, for this domain, we would simply set the number of embeddings in the codebook to two. However, in practice, the human brain may further divide the state space into more regions than there are possible primitive actions in the domain.

For an instance, in the CartPole domain, although it is necessary to perform the ‘move left’ action both when the pole is slightly to the left and very far to the left, we tend to distinguish them into two types of states since it affects future actions. Similarly, in CoinRun, approaching a moving enemy slightly to the left and very far to the left, we tend to distinguish them into two types of states since it affects future actions. Therefore, we set the number of embeddings to be larger than the num-
ber of performable actions in the experiments to further explore the embedding space’s internal status in greater detail and to better emulate human reasoning.

**Regularization and loss function**

To improve the robustness of the codebook learning, we introduce and modify two regularization forms from François-Lavet et al. (2019) into our training. The first, $L_{d1}$, is defined as follows:

$$L_{d1} = \sum_{i \neq j} \exp \left( -C_d \| e_i - e_j \|_2 \right)$$

where $C_d$ is a constant. In contrast to the form in François-Lavet et al. (2019), which considers random pairs, we calculate the sum loss of all possible pairs in the codebook. This prevents any two candidate embedding vectors from getting too close in the latent space and ensures the difference among codebook embedding vectors.

Similarly to the prior used in the original VAE (Kingma and Welling, 2013), we penalize the values of embedding vectors out of an $L_{\infty}$ ball of radius 1. This prevents the risk that may be caused by large values of embedding vectors and is expressed as:

$$L_{d2} = \max_i \left( \| e_i \|_{\infty}^2 - 1, 0 \right)$$

The total regularization loss is then:

$$L_{\text{reg}} = L_{d1} + L_{d2}$$

We define our VQ encoding loss function as the sum of the encoding loss function from (3) and regularization loss from (4) scaled by a constant factor, $\lambda_{\text{reg}}$:

$$L_{\text{vq-enc}} = \| s g[E(x)] - e \|_2^2 + \beta \| s g[E(x)] - E(x) \|_2^2$$

$$+ \lambda_{\text{reg}} L_{\text{reg}}$$

Our proposed auxiliary task is a typical classification problem, and so we use Softmax cross entropy loss, given as $L_{\text{class}}$, for the total training loss. Which is given as the following equation:

$$L_{\text{class}} = - \sum_i \log \left( \sigma (C(e_i)) \right)$$

where $\sigma$ is the softmax function, $a_i$ is the action, $N_a$ is the number of possible actions, and $C$ is the classification net.

Thus, the total objective of our VQ-RL training can be written as:

$$L_{\text{vq-rl}} = L_{\text{rl}} + \lambda_{\text{vq-enc}} L_{\text{vq-enc}} + \lambda_{\text{class}} L_{\text{class}}$$

where $L_{\text{rl}}$ is the training loss from the corresponding RL algorithm, $L_{\text{class}}$ is the classification loss in the auxiliary task, i.e., Softmax cross entropy loss, $\lambda_{\text{vq-enc}}$ is the weight of the VQ encoding loss, and $\lambda_{\text{class}}$ is the weight of the classification loss. $L_{\text{vq-enc}}$ gathers the features extracted by the feature extractor around the embeddings in the codebook and ensures a nontrivial distance between the embeddings. $L_{\text{class}}$ leads the training of VQ encodings to align with policy learning. Therefore, all these endow VQ-RL with the potential to understand states better and improve the generalizability and robustness of RL models. We will present our results of testing this hypothesis in the next section.

**Simulations**

This section studies three aspects of VQ-RL, interpretability, robustness, and generalization on three domains: CartPole (Barto, Sutton, and Anderson, 1983), Minigrid (Chevalier-Boisvert, Willems, and Pal, 2018), and CoinRun (Cobbe et al., 2020). We choose the PPO approach (Schulman et al., 2017) for our analysis but we believe similar conclusions can be easily generalized to other RL methods. The hyper-parameter settings and neural networks designs can be found in our appendix.

**CartPole**

The first domain we study is CartPole, in which the agent controls a cart which can move to the left or right to maintain the balance of a pole stationed on top of it. Our analysis compared training three different approaches: PPO using our VQ-RL framework (which we call VQ-PPO), VQ-PPO with our two proposed regularization methods (which we call VQ-PPO-Reg), and the original PPO algorithm.

**Interpretability** To investigate interpretability we explore the embedding spaces of the three contrasting methods. Specifically, we randomly select 2,000 states from the legal interval of variables in states of CartPole, and then leverage the three feature extractors of trained models to obtain the corresponding features. After that, we perform principal component analysis (PCA) on those 2,000 features. The visualization of the PCA results are presented in Figure 4 and Figure 5. Since there is no output clustering information in the original PPO algorithm, we choose the corresponding actions for identification.

It can be seen that the features obtained by PPO are scattered in a relatively large space. Generally speaking, the outputs in the left half correspond to action 0, while the outputs in the right half correspond to action 1. However, there is no clear boundary between the two types of features in the middle area. As can be seen from Figure 5, although we select the number of embeddings in the codebook to be eight, only a subset of the embeddings in the codebook are used, all located near the $x$-axis and sharing the same principal components. The unused embeddings are highly separated from the
In the next experiment, we examine the robustness of the distribution of states in clusters. Decision-making, such as the priority of state components, requires insight into the internal information of deep models in patterns of noise. Since VQ-RL clusters states, we can gain further insight into the internal information of deep models in decision-making, such as the priority of state components, the distribution of states in clusters, and the impact of noise on VQ-RL.

**Generalization** To study the generalizability of VQ-RL, we propose a modified version of CartPole and name it Gen-CartPole. In this environment, we consider changing three game-related parameters that affect the game’s transition function between states: the mass of the cart ($m_c$), the mass of the pole ($m_p$), and the length of the pole ($l$). In each episode in training, the parameter values will be randomly selected from the set of 10 parameter values. In each episode in training, the parameter values will be randomly selected from the set of three training sets shown in the upper table of Table 2. It is worth noting that we take a window of the last four consecutive states as input states for models.

**MiniGrid-Empty** The MiniGrid-Empty-Random-6x6-v0 domain is chosen for the model training in the interpretability and robustness experiments in this section. In this domain, an agent moves in a 6x6 grid world with randomly generated initial states. The results for different levels of noise are shown in Table 1. The introduction of VQ encoding and our regularization methods significantly improves the robustness of PPO, with the regularized VQ-PPO performing better than VQ-PPO without regularization.

### Table 1: The results for the noisy CartPole-v1 test.

| $\delta$  | PPO       | VQ-PPO     | VQ-PPO-REG |
|-----------|-----------|------------|------------|
| 120       | $500.0\pm0.0$ | $500.0\pm0.0$ | $500.0\pm0.0$ |
| 110       | 499.4±2.4  | $500.0\pm0.0$ | $500.0\pm0.0$ |
| 100       | 499.5±2.1  | $500.0\pm0.0$ | $500.0\pm0.0$ |
| 90        | 499.3±2.8  | $500.0\pm0.0$ | $500.0\pm0.0$ |
| 80        | 490.7±28.5 | $500.0\pm0.0$ | $500.0\pm0.0$ |
| 70        | 455.8±75.4 | 495.6±19.1  | $500.0\pm0.0$ |
| 60        | 394.6±121.0| 493.4±21.4  | $494.1\pm25.9$|
| 50        | 283.3±159.6| 343.3±167.5 | $439.9\pm128.2$|

### Table 2: This table shows the training set parameters and the results for the Gen-CartPole-v1 test sets along with their parameter sets.

| Training Sets $(m_c, m_p, l)$: | PPO       | VQ-PPO     | VQ-PPO-REG |
|-------------------------------|-----------|------------|------------|
| $(0.5, 0.05, 0.25)$, $(1.0, 0.1, 0.5)$, $(2, 0.2, 1)$ | | | |

| Test Set Results $(m_c, m_p, l)$ | PPO       | VQ-PPO     | VQ-PPO-REG |
|----------------------------------|-----------|------------|------------|
| $(0.75, 0.075, 0.375)$ | $500.0\pm0.0$ | $495.8\pm0.0$ | $500.0\pm0.0$ |
| $(1.5, 0.15, 0.75)$ | 499.4±2.4  | $500.0\pm0.0$ | $500.0\pm0.0$ |
| $(3, 0.3, 1.5)$   | 78.1±21.5  | 474.4±29.8  | $500.0\pm0.0$ |
| $(5, 0.5, 2.5)$   | 45.9±6.5   | 231.6±46.2  | $500.0\pm0.0$ |
| $(7.5, 0.75, 3.75)$| 65.7±11.7  | 190.9±49.4  | $363.2\pm168.7$ |

Figure 5 illustrates the clustering of the 2,000 states of VQ-PPO-Reg. Since VQ-RL clusters states, we can gain further insight into the internal information of deep models in decision-making, such as the priority of state components, the distribution of states in clusters.

**Robustness** In the next experiment, we examine the impact of noise on VQ-RL. This is common in real-world applications of RL. For example, in the driving of unmanned vehicles, rain and snow will produce noise in the visual input. Therefore, we need to demonstrate the method’s robustness to noise to ensure the safety of RL technology. Specifically, we add normally distributed noise $n \sim \mathcal{N}(0, 1)/\delta$ to the input states of CartPole, where $\delta$ is a scaling factor to control the level of noise.

We test the robustness of the three methods on 20 episodes with randomly initial states. The results for different levels of noise are shown in Table 1. The introduction of VQ encoding and our regularization methods significantly improves the robustness of PPO, with the regularized VQ-PPO performing better than VQ-PPO without regularization.
Figure 6: The state clusters for VQ-PPO with our regularization method on the CartPole-v1 domain. The abbreviations CP, CV, PA and PAV represent cart position, cart velocity, pole angle and pole angular velocity, respectively. The upper left plots show the distribution of clusters and the corresponding linear regression lines. The diagonal presents the histogram graphs. The lower right part is the kernel density estimate (KDE) plots of clusters. We see that PA and PAV are the primary state components that distinguish the clusters. In each state component, each cluster presents a continuous distribution in a specific interval.

Figure 7: The clustering result of VQ-PPO-Reg on MiniGrid-Empty-Random-6x6-v0. The model divides the state into three clusters according to the actions that need to be performed. In cluster 1, the agent performs the turning right action. Cluster 2 represents moving forward. Cluster 7 is turning left.

Interpretability Figure 7 shows the clustering results of the state of the trained VQ-PPO-Reg model. The model categorizes the state space into three corresponding clusters according to the actions performed in the policy.

Robustness The robustness of the model is tested by adding random noise to the state (the FoV component) in MiniGrid-Empty-Random-6x6-v0. Specifically, as performing each action, each grid cell in the FoV has a certain probability $p_a$ of being ‘attacked’, which results in an “illusion”, that is, each of the three encoding layers assigns a random value within the valid encoding range for this grid cell. The results are given in Table 3. When $p_a$ is small, the performance of the three models is close. However, as $p_a$ increases, the performance of VQ-PPO decreases rapidly, but VQ-PPO-Reg can still maintain a relative advantage over the other two models.

Generalization To test the generalization of our models, we propose a modified MiniGrid-Empty-Random-6x6-v0 domain that we call Gen-MiniGrid-Empty-Random-v0. During training, each episode randomly produces the goal from four possible locations (the green cells in Figure 8). In the evaluation, we test the performance of models in the rest of the locations (the orange cells in Figure 8). The corresponding results are also presented in Figure 8. We can see that the three methods have very similar performance, except for the three corner cells, for which the PPO algorithm performs decidedly worse than the other two.

CoinRun

In this experiment, we leverage the Procgen version of CoinRun (Cobbe et al. 2020) for testing. In this game, the agent needs to move to the far-right of the screen and touch the gold coin to get rewards. We used 200 generated levels of CoinRun on the easy difficulty setting to train our model.
and the same network and hyperparameters as the one in the original paper [Cobbe et al. 2020], except for adding VQ encoding. Furthermore, each model is trained three times with different random seeds, and the evaluation of each model is based on the average of the three instances.

**Interpretability** The same cluster after training may contain multiple different highest probability actions, although the classification loss function constrains features in the same cluster to have the same highest probability actions as much as possible. This may be due to the choice of hyperparameters (such as relatively small weight for the loss of the classification task, etc.) and the action distribution of states with high entropy (some states may have multiple optimum actions to achieve the goal.). However, we can see meaningful clustering results if the states are sub-classified according to different highest probability actions within each cluster. For example, as shown in Figure 9, the states in each row originate from the same cluster and have the same highest probability action. Therefore, the clustering in VQ-RL can still explore the internal state of the models and help us further understand the decision-making process in deep RL.

**Robustness** Salt and Pepper (SP) is one of the most common image noises [Chan, Ho, and Nikolova 2005]. Therefore, we inject SP noise with different degrees to the RGB states in CoinRun to test the robustness of the three models. The results are shown in Table 4. We see that the VQ-RL framework improves the robustness of PPO.

**Generalization** In the final experiments, we test the generalization of the models for the CoinRun domain. We test the three models trained with the same random parameters on the same 1,000 unseen ‘all distributions’ episodes to ensure fairness. We performed this experiment three times, labeled as Run1-3. The results for this generalization experiment can be seen in Table 5. We see that the VQ-RL framework slightly improves the generalization of PPO.

### Discussion and Conclusion

In this paper, we propose a new framework to enhance the clustering properties of the embedding space of deep reinforcement learning methods. In the experimental part, we test the effectiveness of VQ-RL. Our results on three test domains suggest that 1) VQ-RL improves the interpretability of deep RL from different aspects in the three domains; 2) regarding robustness and generalization, VQ-RL has significant advantages in CartPole with the continuous representation and different degrees of improvement in the other two domains using hand-coding and visual inputs. In general, the baseline is improved in these two aspects; 3) the regularization methods are effective and essential to maintain the robustness and generalization in VQ-RL.
Acknowledgments

Research was sponsored by the Army Research Office and was accomplished under Grant Number W911NF-20-1-0002. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

References

Arulkumaran, K.; Deisenroth, M. P.; Brundage, M.; and Bharath, A. A. 2017. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6): 26–38.

Barto, A. G.; Sutton, R. S.; and Anderson, C. W. 1983. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5): 834–846.

Bellemare, M. G.; Dabney, W.; and Munos, R. 2017. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, 449–458. PMLR.

Chan, R. H.; Ho, C.; and Nikolova, M. 2005. Salt-and-Pepper Noise Removal by Median-Type Noise Detectors and Detail-Preserving Regularization. *IEEE Trans. Image Process.*, 14(10): 1479–1485.

Chevalier-Boisvert, M.; Willems, L.; and Pal, S. 2018. Minimalistic Gridworld Environment for OpenAI Gym. [https://github.com/maximecb/gym-minigrid](https://github.com/maximecb/gym-minigrid).

Cobbe, K.; Hesse, C.; Hilton, J.; and Schulman, J. 2020. Leveraging Procedural Generation to Benchmark Reinforcement Learning. In *Proceedings of the 37th International Conference on Machine Learning*, ICML 2020, 13-18 July 2020, Virtual Event, volume 119 of *Proceedings of Machine Learning Research*, 2048–2056. PMLR.

Cobbe, K.; Klimov, O.; Hesse, C.; Kim, T.; and Schulman, J. 2019. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*, 1282–1289. PMLR.

Diuk, C.; Cohen, A.; and Littman, M. L. 2008. An object-oriented representation for efficient reinforcement learning. In *Proceedings of the 25th international conference on Machine learning*, 240–247.

François-Lavet, V.; Bengio, Y.; Precup, D.; and Pineau, J. 2019. Combined reinforcement learning via abstract representations. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3582–3589.

Gārbacea, C.; van den Oord, A.; Li, Y.; Lim, F. S.; Luebs, A.; Vinyals, O.; and Walters, T. C. 2019. Low bit-rate speech coding with VQ-VAE and a WaveNet decoder. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 735–739. IEEE.

Gārbacea, C.; van den Oord, A.; Li, Y.; Lim, F. S. C.; Luebs, A.; Vinyals, O.; and Walters, T. C. 2019. Low Bit-rate Speech Coding with VQ-VAE and a WaveNet Decoder. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2019, Brighton, United Kingdom, May 12-17, 2019*, 735–739. IEEE.

Guo, Z. D.; Pires, B. A.; Piot, B.; Grill, J.-B.; Altché, F.; Munos, R.; and Azar, M. G. 2020. Bootstrap latent-predictive representations for multi-task reinforcement learning. In *International Conference on Machine Learning*, 3875–3886. PMLR.

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. *arXiv preprint arXiv:1910.12991*.

Jaderberg, M.; Zaremba, W. M.; Dunning, I.; Marris, L.; Lever, G.; Castaneda, A. G.; Beattie, C.; Rabinowitz, N. C.; Morcos, A. S.; Ruderman, A.; et al. 2019. Human-level performance in 3D multiplayer games with population-based reinforcement learning. *Science*, 364(6443): 859–865.

Jaderberg, M.; Mnih, V.; Zaremba, W. M.; Schaul, T.; Leibo, J. Z.; Silver, D.; and Kavukcuoglu, K. 2016. Reinforcement learning with unsupervised auxiliary tasks. *arXiv preprint arXiv:1611.05397*.

Kaiser, L.; Babaeizadeh, M.; Milos, P.; Osinski, B.; Campbell, R. H.; Czechowski, K.; Erhan, D.; Finn, C.; Kozakowski, P.; Levine, S.; et al. 2019. Model-based reinforcement learning for atari. *arXiv preprint arXiv:1903.00374*.

Kingma, D. P.; and Welling, M. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Lee, A. X.; Nagabandi, A.; Abbeel, P.; and Levine, S. 2019a. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. *arXiv preprint arXiv:1907.00953*.

Lee, K.; Lee, K.; Shin, J.; and Lee, H. 2019b. Network randomization: A simple technique for generalization in deep reinforcement learning. *arXiv preprint arXiv:1910.05396*.

Levine, S.; Finn, C.; Darrell, T.; and Abbeel, P. 2016. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1): 1334–1373.

Lütjens, B.; Everett, M.; and How, J. P. 2020. Certified adversarial robustness for deep reinforcement learning. In *Conference on Robot Learning*, 1328–1337. PMLR.

Lyu, D.; Yang, F.; Liu, B.; and Gustafson, S. 2019. SDRL: interpretable and data-efficient deep reinforcement learning leveraging symbolic planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 2970–2977.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; et al. 2015. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533.

Mott, A.; Zoran, D.; Chrzanowski, M.; Wierstra, D.; and Rezende, D. J. 2019. Towards interpretable reinforcement learning using attention augmented agents. *arXiv preprint arXiv:1906.02500*.

Oord, A. v. d.; Li, Y.; and Vinyals, O. 2018. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*. 
Oord, A. v. d.; Vinyals, O.; and Kavukcuoglu, K. 2017. Neural discrete representation learning. arXiv preprint arXiv:1711.00937.

Packer, C.; Gao, K.; Kos, J.; Krähenbühl, P.; Kolotun, V.; and Song, D. 2018. Assessing generalization in deep reinforcement learning. arXiv preprint arXiv:1810.12282.

Razavi, A.; van den Oord, A.; and Vinyals, O. 2019. Generating diverse high-fidelity images with vq-vae-2. In Advances in neural information processing systems, 14866–14876.

Robine, J.; Uelwer, T.; and Harmeling, S. 2020. Discrete Latent Space World Models for Reinforcement Learning. CoRR, abs/2010.05767.

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

Silver, D.; Huang, A.; Maddison, C. J.; Guez, A.; Sifre, L.; Van Den Driessche, G.; Schrittwieser, J.; Antonoglou, I.; Panneershelvam, V.; Lanctot, M.; et al. 2016. Mastering the game of Go with deep neural networks and tree search. nature, 529(7587): 484–489.

Stooke, A.; Lee, K.; Abbeel, P.; and Laskin, M. 2021. Decoupling representation learning from reinforcement learning. In International Conference on Machine Learning, 9870–9879. PMLR.

Such, F. P.; Madhavan, V.; Liu, R.; Wang, R.; Castro, P. S.; Li, Y.; Zhi, J.; Schubert, L.; Bellemare, M. G.; Clune, J.; et al. 2018. An atari model zoo for analyzing, visualizing, and comparing deep reinforcement learning agents. arXiv preprint arXiv:1812.07069.

Tessler, C.; Efroni, Y.; and Mannor, S. 2019. Action robust reinforcement learning and applications in continuous control. In International Conference on Machine Learning, 6215–6224. PMLR.

Tjandra, A.; Sakti, S.; and Nakamura, S. 2020. Transformer vq-vae for unsupervised unit discovery and speech synthesis: Zerospeech 2020 challenge. arXiv preprint arXiv:2005.11676.

Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. Journal of machine learning research, 9(11).

Verma, A.; Murali, V.; Singh, R.; Kohli, P.; and Chaudhuri, S. 2018. Programmatically interpretable reinforcement learning. In International Conference on Machine Learning, 5045–5054. PMLR.

Wang, J.; Gou, L.; Shen, H.-W.; and Yang, H. 2018. Dqnviz: A visual analytics approach to understand deep q-networks. IEEE transactions on visualization and computer graphics, 25(1): 288–298.

Wang, J.; Liu, Y.; and Li, B. 2020. Reinforcement learning with perturbed rewards. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, 6202–6209.

Yan, W.; Zhang, Y.; Abbeel, P.; and Srivivas, A. 2021. VideoGPT: Video Generation using VQ-VAE and Transformers. CoRR, abs/2104.10157.

Zahavy, T.; Ben-Zrihem, N.; and Mannor, S. 2016. Graying the black box: Understanding dqns. In International Conference on Machine Learning, 1899–1908. PMLR.

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