Regret-Based Exploration and Latent Space Clustering in SUNRISE Deep Q-Network Ensembles

William Zheng

December 2020

Abstract

Simple ensemble methods have achieved state-of-the-art performance in model-free Reinforcement Learning (RL) through stabilizing learning by training a set of diverse agents and performing efficient exploration. This work seeks to improve aspects of training diverse agents and of performing exploration for ensemble methods by developing two new ensemble methods for general off-policy RL methods, U-SUNRISE and R-SUNRISE. U-SUNRISE improve the training of diverse agents within the ensemble by encouraging each agent to learn samples from clusters obtained through the unsupervised clustering of raw state observations in a low-dimensional latent space. R-SUNRISE improves efficient action exploration through the inclusion of online regret minimization algorithms. After testing on Atari games, we show that these algorithms can outperform previously developed simple ensemble methods.

Introduction

While reinforcement Learning (RL) algorithms have successfully learned a variety of complex tasks including playing Atari games [1], these algorithms remain difficult to train. Many of these algorithms depend on training neural networks which are highly sensitive to random noise, careful hyper-parameter settings, optimal training data distributions, and efficient exploration of unseen state-action pairs [2], [3].

A variety of techniques and algorithms have been developed to improve the stability and efficiency of these RL algorithms. For off-policy algorithms like Deep Q Networks (DQN)s and Soft Actor-Critic (SAC) [4], the training data can be weighted in order to improve learning based on prioritizing important transitions [5] and to overcome pathological data distribution problems that arise from lack of corrective feedback in approximate dynamic programming methods (e.g., DQNs and Actor-Critic algorithms) [2]. For DQNs, a variety of different improvements from Double Q-learning [6], to Dueling Networks [7], to Distributional RL [8], and Noisy Nets [9] have led to scoring and training stability improvements for many tasks. Hessel, Modayil, Hasselt, et al. [10] combine these improvements together to create a DQN-based algorithm, RAINBOW, which has exceptional performance on Atari games.

A new complementary improvement, ensembles, has shown state-of-the-art learning performance. Lee, Laskin, Srinivas, et al. [3] were able to show that a simple unified ensemble scheme, SUNRISE, outperformed existing off-policy RL methods and model-based RL methods. Two key ideas behind SUNRISE were maintaining diversity among the agents within the ensemble through bootstrapping and efficient exploration via upper confidence bound inference. Building on this work, we introduce two methods, U-SUNRISE and R-SUNRISE, that seek to explore new ways to maintain agent diversity and perform efficient exploration within ensembles respectively.

1. **Unsupervised Environment Clustering (U-SUNRISE):** We map our high-dimensional states to a low-dimensional latent space and perform unsupervised clustering. Then, each

agent in the ensemble is assigned a cluster-centroid in the latent space. A state-action pair is added to each agent's buffer with a probability vector defined as a soft-max over the latent space distances between the state observation to each of the cluster-centroids. This encourages each agent in the ensemble to "specialize" in similarly clustered state observations and encourage the training of a diverse ensemble.

2. **Regret Weighted Exploration (R-SUNRISE):** We incorporate a decision-theoretic online learning algorithm that minimizes regret, Normal Hedge [11], in order to give more weight to the predictions of the agents in the ensemble that have accumulated lower regret when suggesting actions.

We benchmark the performance of these ensemble algorithms against SUNRISE [3] in two Atari Environments (limited by compute) Asteroids and Boxing. For the agents in our ensemble, we use Rainbow DQN [10]. In our experiments, we see our modifications, R-SUNRISE and U-SUNRISE, improve upon the performance of SUNRISE.

Methods

Regret-Weighted Exploration

In SUNRISE, Algorithm 1 listed in the appendix, exploration is conducted through an upper confidence bound approach across the agents, which are DQNs in our application, in the ensemble through the following function:

$$a_t = \arg\max_{a_{t,i} \in \mathcal{A}} Q_{\text{mean}}(s_t, a_{t,i}) + \lambda Q_{\text{std}}(s_t, a_{t,i})$$

where λ is a constant and:

$$Q_{\text{mean}}(s_t, a_{t,i}) = 1/n \sum_{i=1}^{n} Q_i(s_t, a_{t,i})$$

$$Q_{\text{std}}(s_t, a_{t,i}) = \sqrt{1/n \sum_{i=1}^{n} (Q_i(s_t, a_{t,i}) - Q_{\text{mean}})^2}$$

The intuition here is that the uncertainty on each action can be quantified by the standard deviation of the Q-values predicted by the ensemble of DQNs. Exploration is then encouraged by providing a bonus for visiting unseen state-action pairs with high uncertainty [3].

While this approach allows for efficient search, there is no mechanism to penalize unreliable DQNs in the ensemble. Therefore, we modify the Q_{std} term by weighting the standard deviation contribution of each DQN by how many times it under-performed quantified by a custom loss function. Our expression for choosing the next action at time t becomes:

$$a_t = \arg\max_{a_{t,i} \in \mathcal{A}} Q_{\text{mean}}(s_t, a_{t,i}) + \lambda Q_{\text{std}}^{\text{Regret}}(s_t, a_{t,i})$$

where $Q_{\text{std}}^{\text{Regret}}(s_t, a_{t,i})$ is calculated by the following for a n-ensemble:

$$Q_{\text{std}}^{\text{Regret}}(s, a) = \sqrt{\sum_{i=1}^{n} w_i \cdot (Q_{\text{mean}}(s, a) - Q_i(s_t, a_{t,i}))^2}$$

where $w \in \Delta^{n-1}$ is determined by regret minimization function, Normal Hedge [11], through the following procedure. Each DQN accumulates loss when an action is taken, a reward is received, and it didn't predict a high enough Q-value for that action. If a DQN predicted a

low Q-value for the action relative to an arbitrary reference function of the DQNs, $Q_{\text{ref}}(s_t, a_t)$, in the ensemble, it incurs a loss. These losses can then be fed to into NormalHedge which attempts to choose weights, $w \in \Delta^{n-1}$, which decrease the standard deviation contribution to $Q_{\text{std}}^{\text{Regret}}(s_t, a_{t,i})$ for DQN i if it has accumulated a significant amount of regret and therefore is unreliable. Overall, this regret-minimization exploration algorithm would be more robust to unreliable DQNs within the ensemble. For this implementation, we select an arbitrary DQN to a fixed DQN in the ensemble. The total algorithm, Algorithm 2, is listed in the appendix.

Unsupervised Environment Clustering

In SUNRISE, the agents are trained to be diverse through bootstrapping [12] which has been shown to help stabilize the learning process and improve performance. The diversity from bootstrapping comes from the initial random parameter values and the different random training samples generated for each agent. Binary masks, $m_{t,i}$ are drawn from a Bernoulli distribution with parameter $\beta \in (0, 1]^n$ at each time-step t. These masks determine if a sample transition at time t is added to the buffer an agent i. While this is an effective way to train agents independently, we would like to choose the binary masks in such a way that allows each agent to further specialize in specific state-action pairs and further enforce diversity.

As inspiration, Seo, Lee, Clavera, et al. [13] recently developed a method that learns a multi-headed dynamics model and specializes each head to similar environments through clustering. An adaptive planning method then selects the most applicable head based on a latent context vector dependent on past experiences.

To treat each agent as a head specialized to each environment, we first attempt to derive a low-dimensional representation of the states and then perform unsupervised clustering, as seen in Fig. 5 in the appendix. RL algorithms that leverage latent space state representations have been shown to improve overall performance and sample efficiency [14]. Additionally, it can help reduce computational complexity. In this work, we use Principal Component Analysis (PCA) in order to map from our state observations to the low-dimensional latent space. We chose PCA because it's a dimensionality reduction technique whose components are easy to analyze for correctness, Fig. 3 in appendix, which is helpful when developing a new algorithm.

For unsupervised clustering, we perform K-means. By using K-means, we are allowed to choose $k = n$ in order to match the number of agents in our ensemble. After clustering with K-means, we can extract the centroid of each cluster, $c_1, \ldots, c_n$, and assign each agent i in the ensemble to a cluster-centroid c_i. Given a state s_t, our PCA-based encoder as $\phi(x) : \mathbb{R}^x \to \mathbb{R}^y, : x \geq y$, dist representing the euclidean norm, and $c_1, \ldots, c_n \in \mathbb{R}^k$ as the cluster centroids for n agents in the ensemble. The Bernoulli parameter for agent i at time t is determined by:

$$\rho_{t,i} = \beta \cdot \frac{\exp(\text{dist}(\phi(s_t), c_i))}{\sum_{l=1}^{n} \exp(\text{dist}(\phi(s_t), c_l))}$$

By applying a soft-max to the latent space distances between s_t and our centroids, we've observed experimentally that ϕ_t doesn't concentrate all the probability mass on one agent. This helps prevent training an ensemble where each agent is over-trained in only a specific region of the latent space but left without any support for other states. Additionally, β will still control the number of transitions included in the buffer.

In order to train the encoder and clustering algorithm, we first use a random policy to collect training states. The total algorithm, Algorithm 3, is listed in the appendix.

Results

Scores on Atari Games

To benchmark our modified algorithms, U-SUNRISE and R-SUNRISE, we evaluate performance on two Atari games, Boxing and Asteroids. Each score is the average performance on 10 evaluation episodes after 100K training steps averaged over 3 random trials with set random seeds. Additionally, we run SUNRISE and SUNRISE without UCB exploration (SUNRISE no UCB) as baselines for performance comparison.

Game	**SUNRISE no UCB**	**SUNRISE**	**U-SUNRISE**	**R-SUNRISE**
Boxing	1.2	4.067	**5.167**	3.8
Asteroids	699.67	679.33	750.66	**812.67**

Table 1: The scores are the performance of each algorithm after 100K training steps averaged over 3 trials. For Boxing, we see that U-SUNRISE outperforms both baselines and R-SUNRISE outperforms SUNRISE no UCB. For Asteroids, U-SUNRISE and R-SUNRISE outperform both baselines.

Asteroids Training Performance

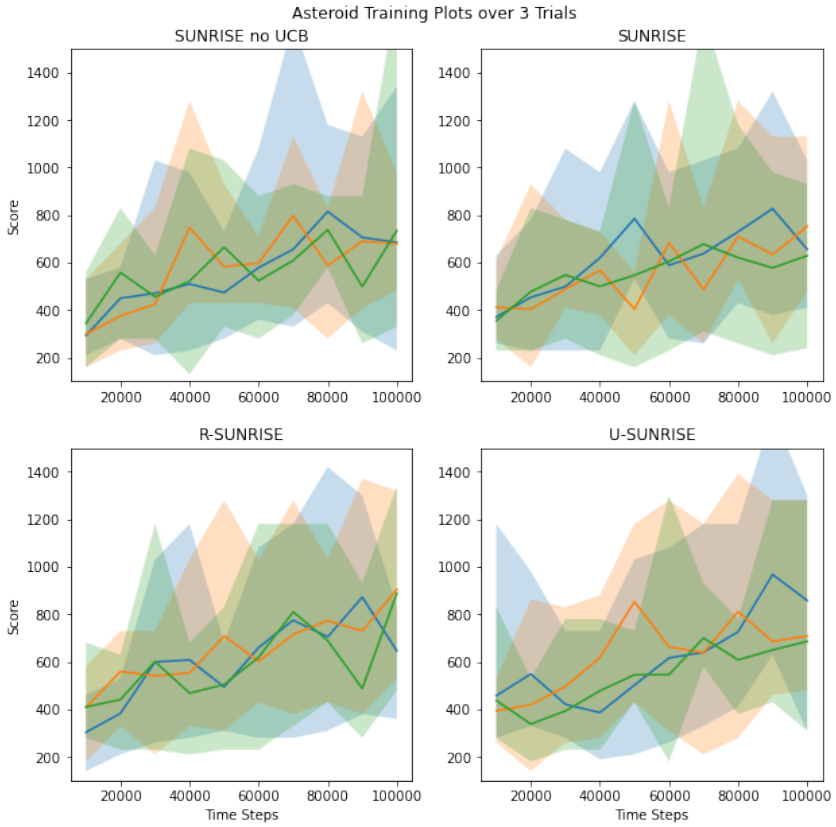

Figure 1: Above are the Asteroids scores of each algorithm as a function of time. Each color denotes a different random seed used in training. The lighter region for each run is bounded by the max score observed and the min score observed over 10 evaluation episodes.

Boxing

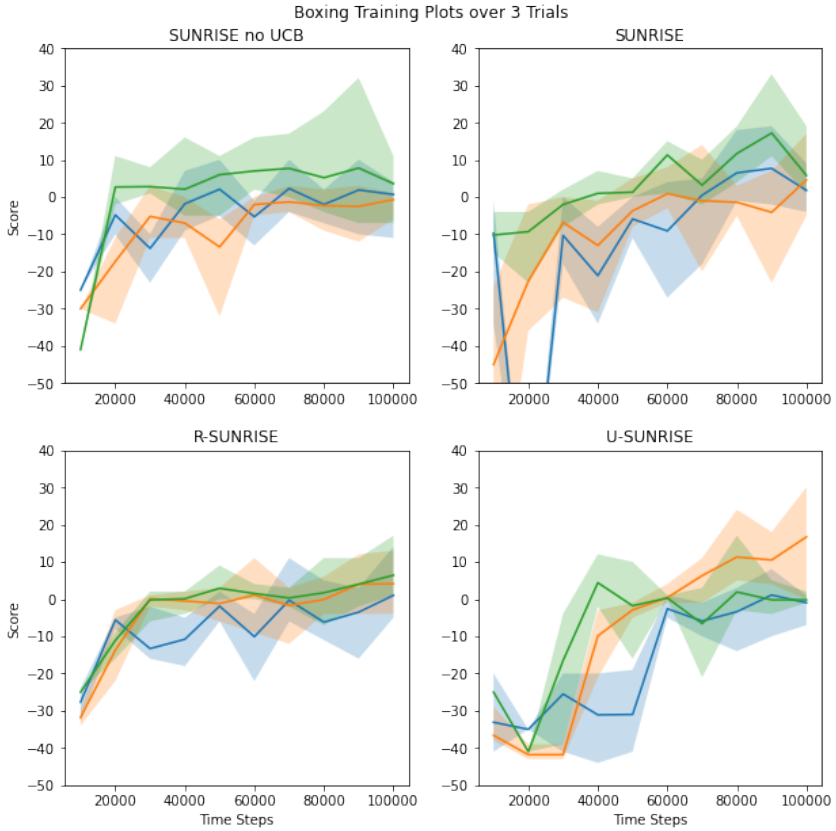

Figure 2: Above are the Boxing scores of each algorithm as a function of time. Each color denotes a different random seed used in training. The lighter region for each run is bounded by the max score observed and the min score observed over 10 evaluation episodes.

Discussion

Algorithm and Training Performance

For the Boxing experiments, U-SUNRISE out-performed all other methods. This, however, could be attributed to the extremely high average evaluation performance on a single trial. When compared to similar random seed trials, U-SUNRISE under-performed SUNRISE and R-SUNRISE 2 out of 3 times. On Boxing, R-SUNRISE seemed to have performed similarly to SUNRISE and out-performed SUNRISE no UCB. However, we notice that in Fig. 2, R-SUNRISE has much more stable training which could be attributed to the regret minimization modification to its exploration.

For Asteroids, R-SUNRISE out-performed all other methods and U-SUNRISE out-performed both our baselines. Surprisingly, SUNRISE no UCB out-performed SUNRISE. In general, the training performance for all algorithms, as observed in Fig. 1, was fairly similar.

Possible Drawbacks and Improvements

For U-SUNRISE, there are two possible improvements. Firstly, the clustering procedure we use can result in imbalanced clusters leading to unequal training samples for each agent in the ensemble. A clustering procedure that can balance each cluster more equally would solve this problem. Secondly, the clustering procedure is dependent on observations obtained by a random policy and therefore the clusters learned over the sampled data may not be representative of

the true state clusters. Instead of using a random policy, we can use the state-action pairs generated through training an agent to obtain data distribution that would better resemble what the ensemble would actually encounter. Additionally, we can take an online approach and update the clusters as the ensemble trains on new state-action pairs.

For R-SUNRISE, an improvement could address an underlying assumption where the reliability of each algorithm is non-stationary (i.e., it doesn't change with time). This assumption could be flawed as the performance of the agents is constantly fluctuating through training. Much decision-theoretic online learning (DTOL) methods do not handle situations where the reliability of the experts changes over time. However, there are a variety of DTOL methods that can handle non-stationary agents competitively [15], [16].

Conclusion

In this work, we were able to develop two algorithms U-SUNRISE and R-SUNRISE from the simple unified ensemble method SUNRISE [3]. These two algorithms improved ensemble diversity and efficient exploration respectively and are designed to be compatible with general off-policy RL algorithms as agents in the ensemble, like SUNRISE. With Rainbow DQNs as agents, we benchmarked U-SUNRISE and R-SUNRISE against SUNRISE on two Atari games, Boxing and Asteroids. We found that U-SUNRISE and R-SUNRISE outperformed SUNRISE in a variety of instances and R-SUNRISE showed better overall training stability.

Future Work

For the choice of latent space encoders, many methods leverage Variational Auto-Encoders [17]–[19] and newer state-of-the-art approaches have sought to leverage encoders that preserve bisimulation metrics [20]. While we used PCA as the encoding method for projecting our states into a lower-dimensional latent space, future work should leverage state-of-the-art bisimulation-based encoders.

As mentioned in the Possible Drawbacks section, using a DTOL algorithm that can better handle non-stationary agent reliability could improve the performance of R-SUNRISE.

We are also hopeful that ensemble methods with diverse agents could be applied competitively to tasks in Meta-Learning and Transfer Learning.

References

[1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, et al. "Playing Atari with Deep Reinforcement Learning". In: *CoRR* abs/1312.5602 (2013). arXiv: 1312.5602. URL: http://arxiv.org/abs/1312.5602.

[2] Aviral Kumar, Abhishek Gupta, and Sergey Levine. *DisCor: Corrective Feedback in Reinforcement Learning via Distribution Correction*. 2020. arXiv: 2003.07305 [cs.LG].

[3] Kimin Lee, Michael Laskin, Aravind Srinivas, et al. *SUNRISE: A Simple Unified Framework for Ensemble Learning in Deep Reinforcement Learning*. 2020. arXiv: 2007.04938 [cs.LG].

[4] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, et al. "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor". In: *CoRR* abs/1801.01290 (2018). arXiv: 1801.01290. URL: http://arxiv.org/abs/1801.01290.

[5] Tom Schaul, John Quan, Ioannis Antonoglou, et al. *Prioritized Experience Replay*. 2016. arXiv: 1511.05952 [cs.LG].

[6] Hado Hasselt. "Double Q-learning". In: *Advances in Neural Information Processing Systems*. Ed. by J. Lafferty, C. Williams, J. Shawe-Taylor, et al. Vol. 23. Curran Associates, Inc., 2010, pp. 2613–2621. URL: https://proceedings.neurips.cc/paper/2010/file/091d584fced301b442654dd8c23b3fc9-Paper.pdf.

[7] Ziyu Wang, Nando de Freitas, and Marc Lanctot. "Dueling Network Architectures for Deep Reinforcement Learning". In: *CoRR* abs/1511.06581 (2015). arXiv: 1511.06581. URL: http://arxiv.org/abs/1511.06581.

[8] Marc G. Bellemare, Will Dabney, and Rémi Munos. "A Distributional Perspective on Reinforcement Learning". In: *CoRR* abs/1707.06887 (2017). arXiv: 1707.06887. URL: http://arxiv.org/abs/1707.06887.

[9] Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, et al. "Noisy Networks for Exploration". In: *CoRR* abs/1706.10295 (2017). arXiv: 1706.10295. URL: http://arxiv.org/abs/1706.10295.

[10] Matteo Hessel, Joseph Modayil, Hado van Hasselt, et al. "Rainbow: Combining Improvements in Deep Reinforcement Learning". In: *CoRR* abs/1710.02298 (2017). arXiv: 1710.02298. URL: http://arxiv.org/abs/1710.02298.

[11] Kamalika Chaudhuri, Yoav Freund, and Daniel J. Hsu. "A parameter-free hedging algorithm". In: *CoRR* abs/0903.2851 (2009). arXiv: 0903.2851. URL: http://arxiv.org/abs/0903.2851.

[12] Ian Osband, Charles Blundell, Alexander Pritzel, et al. "Deep Exploration via Bootstrapped DQN". In: *CoRR* abs/1602.04621 (2016). arXiv: 1602.04621. URL: http://arxiv.org/abs/1602.04621.

[13] Younggyo Seo, Kimin Lee, Ignasi Clavera, et al. *Trajectory-wise Multiple Choice Learning for Dynamics Generalization in Reinforcement Learning*. 2020. arXiv: 2010.13303 [cs.LG].

[14] Alex X. Lee, Anusha Nagabandi, Pieter Abbeel, et al. "Stochastic Latent Actor-Critic: Deep Reinforcement Learning with a Latent Variable Model". In: *CoRR* abs/1907.00953 (2019). arXiv: 1907.00953. URL: http://arxiv.org/abs/1907.00953.

[15] Dmitry Adamskiy, Wouter M. Koolen, Alexey Chernov, et al. "A Closer Look at Adaptive Regret". In: *Proceedings of the 23rd International Conference on Algorithmic Learning Theory*. ALT'12. Lyon, France: Springer-Verlag, 2012, pp. 290–304. ISBN: 9783642341052. DOI: 10.1007/978-3-642-34106-9_24. URL: https://doi.org/10.1007/978-3-642-34106-9_24.

[16] Haipeng Luo and Robert E. Schapire. "Achieving All with No Parameters: AdaNormalHedge". In: *Proceedings of The 28th Conference on Learning Theory*. Ed. by Peter Grünwald, Elad Hazan, and Satyen Kale. Vol. 40. Proceedings of Machine Learning Research. Paris, France: PMLR, Mar. 2015, pp. 1286–1304. URL: http://proceedings.mlr.press/v40/Luo15.html.

[17] David Ha and Jürgen Schmidhuber. "World Models". In: *CoRR* abs/1803.10122 (2018). arXiv: 1803.10122. URL: http://arxiv.org/abs/1803.10122.

[18] Danijar Hafner, Timothy P. Lillicrap, Ian Fischer, et al. "Learning Latent Dynamics for Planning from Pixels". In: *CoRR* abs/1811.04551 (2018). arXiv: 1811.04551. URL: http://arxiv.org/abs/1811.04551.

[19] Danijar Hafner, Timothy P. Lillicrap, Jimmy Ba, et al. "Dream to Control: Learning Behaviors by Latent Imagination". In: *CoRR* abs/1912.01603 (2019). arXiv: 1912.01603. URL: http://arxiv.org/abs/1912.01603.

[20] Amy Zhang, Rowan McAllister, Roberto Calandra, et al. *Learning Invariant Representations for Reinforcement Learning without Reconstruction*. 2020. arXiv: 2006.10742 [cs.LG].

Appendix

The original SUNRISE algorithm for the Rainbow DQN along with R-SUNRISE and U-SUNRISE are listed here for reference.

Algorithm 1: SUNRISE: Rainbow Version

for *each iteration* **do**
  for *each timestep* t **do**
    \\UCB Exploration
    Choose the action that maximizes $a_t = \arg\max_{a_{t,i} \in \mathcal{A}} Q_{\text{mean}}(s_t, a_{t,i}) + \lambda Q_{\text{std}}(s_t, a_{t,i})$
    Collect state s_{t+1} and reward r_t from the environment by taking action a_t
    Sample bootstrap masks $M_t = \{m_{t,i} \sim \text{Bernoulli}(\beta) \mid i \in \{1, ..., N\}\}$
    Store transitions $\tau_t = (s_t, a_t, s_{t+1}, r_t)$ and masks in replay buffer $\mathcal{B} \leftarrow \mathcal{B}(\tau_t, M_t)$
  \\Update Q-Functions via Bootstrap and Weighted Bellman Backup
  for *each gradient step* **do**
    Sample random minibatch $\{(\tau_j, M_j)\}_{j=1}^{B} \sim \mathcal{B}$
    for *each agent* i **do**
      Update the Q-function by minimizing $\frac{1}{B} \sum_{j=1}^{B} m_{j,i} \mathcal{L}_{\text{WQ}}^{\text{DQN}}(\tau_j, \theta_i)$

Algorithm 2: R-SUNRISE: Regret-Weighted Exploration

Initialize **NormalHedge**
for *each iteration* **do**
  for *each timestep* t **do**
    \\UCB Exploration
    Choose the action that maximizes $a_t = \arg\max_{a_{t,i} \in \mathcal{A}} Q_{\text{mean}}(s_t, a_{t,i}) + \lambda Q_{\text{std}}^{\text{Regret}}(s_t, a_{t,i})$
    Collect state s_{t+1} and reward r_t from the environment by taking action a_t
    Sample bootstrap masks $M_t = \{m_{t,i} \sim \text{Bernoulli}(\beta) \mid i \in \{1, ..., N\}\}$
    Store transitions $\tau_t = (s_t, a_t, s_{t+1}, r_t)$ and masks in replay buffer $\mathcal{B} \leftarrow \mathcal{B}(\tau_t, M_t)$
    if $r_t > 0$ **then**
      Q_i accumulates loss following $L_i^{\text{Regret}} = \mathbb{1}(Q_i(s_t, a_t) < Q_{\text{ref}}(s_t, a_t))$
      Update regret minimization algorithm, **NormalHedge**, with loss vector L^{Regret}
  \\Update Q-Functions via Bootstrap and Weighted Bellman Backup
  for *each gradient step* **do**
    Sample random minibatch $\{(\tau_j, M_j)\}_{j=1}^{B} \sim \mathcal{B}$
    for *each agent* i **do**
      Update the Q-function by minimizing $\frac{1}{B} \sum_{j=1}^{B} m_{j,i} \mathcal{L}_{\text{WQ}}^{\text{DQN}}(\tau_j, \theta_i)$

Algorithm 3: SUNRISE: Unsupervised Environment Clustering

for *each iteration* **do**
  \\Pre-Sampling
  Randomly choose an action a_t and store the state s_t in a buffer, $\mathcal{B}_{\text{pre-sample}}$
\\Projecting
Apply Principal Component Analysis (PCA) to the buffer $\mathcal{B}_{\text{pre-sample}}$ to learn the top 100 principal components for mapping to a latent space.
\\Clustering
Apply K-means to choose K clusters in our latent space and derive the K centroids (K should match the number of ensembles).
for *each iteration* **do**
  for *each timestep* t **do**
    \\UCB Exploration
    Choose the action that maximizes $a_t = \arg\max_{a_{t,i} \in \mathcal{A}} Q_{\text{mean}}(s_t, a_{t,i}) + \lambda Q_{\text{std}}(s_t, a_{t,i})$
    Collect state s_{t+1} and reward r_t from the environment by taking action a_t
    Sample bootstrap masks $M_t = \{m_{t,i} \sim \text{Bernoulli}(\rho_{t,i}) \mid i \in \{1, ..., N\}\}$
    Store transitions $\tau_t = (s_t, a_t, s_{t+1}, r_t)$ and masks in replay buffer $\mathcal{B} \leftarrow \mathcal{B}(\tau_t, M_t)$
  \\Update Q-Functions via Bootstrap and Weighted Bellman Backup
  for *each gradient step* **do**
    Sample random minibatch $\{(\tau_j, M_j)\}_{j=1}^{B} \sim \mathcal{B}$
    for *each agent* i **do**
      Update the Q-function by minimizing $\frac{1}{B} \sum_{j=1}^{B} m_{j,i} \mathcal{L}_{\text{WQ}}^{\text{DQN}}(\tau_j, \theta_i)$

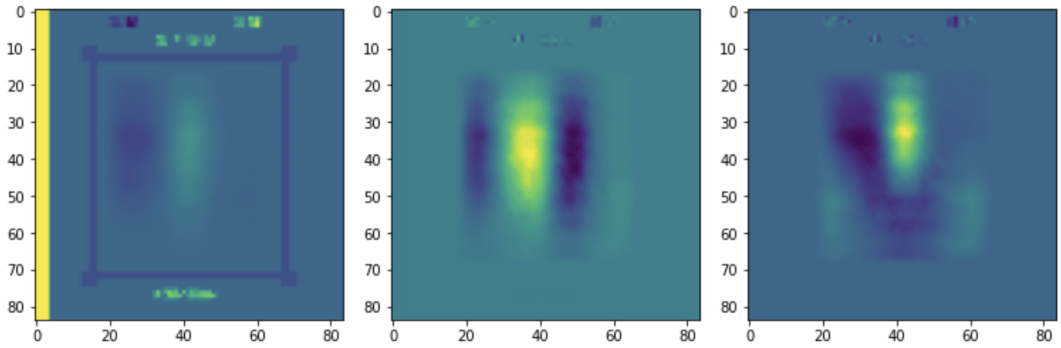

Figure 3: Illustration of three of the orthogonal components from the PCA method on the Boxing game. These components are easily interpretable as the game window as well as the position of the players can be inferred.

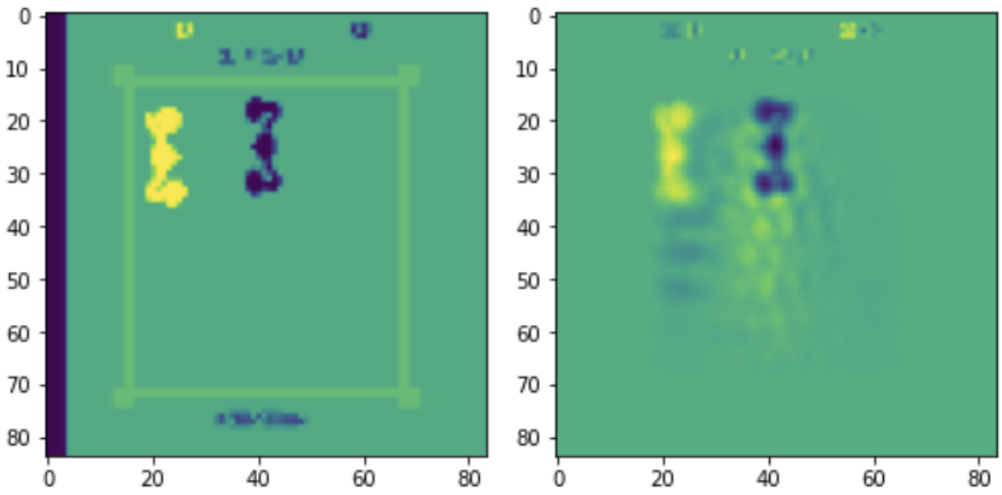

Figure 4: The actual image vs the reconstructed PCA image from the Boxing game. We can see that the method can somewhat recover the original game image.

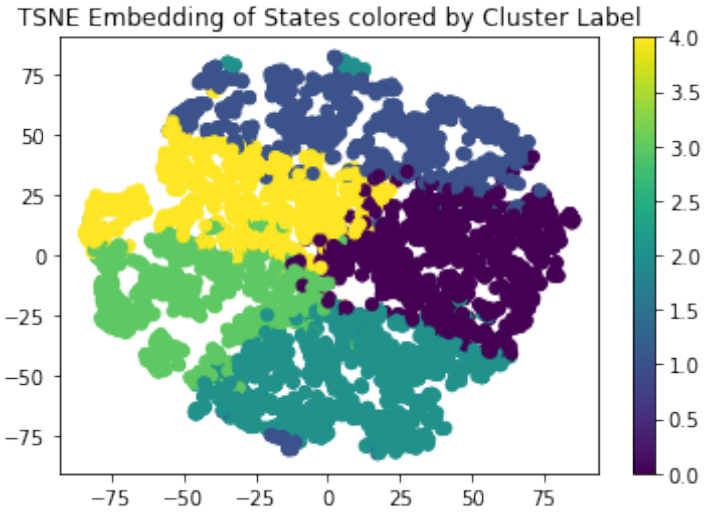

Figure 5: T-SNE visualization of how the clusters of states determined by the PCA+K-means technique look in a 2D space.

The code will be in the following repository: https://github.com/magittan/sunrise