Rainbow: Combining Improvements in Deep Reinforcement Learning

Matteo Hessel, Will Dabney, Joseph Modayil, Dan Horgan, Hado van Hasselt, Bilal Piot, Tom Schaul, Mohammad Azar, Georg Ostrovski, David Silver

***Abstract*—The paper combines 6 different DQN variants, namely Double Q-learning, Prioritized replay, Dueling networks, Multi-step learning, Distributional RL and Noisy Nets- together called *Rainbow* to achieve improved data efficiency and higher performance on Atari 2600 benchmark.**

I. Summary of DQN variants

Deep Q-learning : The regular DQN loss for a transition $(S_t, A_t, R_{t+1}, \gamma_{t+1}, S_{t+1})$ is given by $(R_{t+1} + \gamma_{t+1}\max_{a'} q_{\bar{\theta}}(S_{t+1}, a') - q_\theta(S_t, A_t))^2$; where θ denotes the parameters of the *online network* and $\bar{\theta}$ denotes the parameters of the *target network*.

Double Q-learning : Inorder to reduce the overestimation bias in the maximization step of the regular DQN update by decoupling, Double Q-learning was incorporated into DQN giving the loss function $(R_{t+1} + \gamma_{t+1} q_{\bar{\theta}}(S_{t+1}, \max_{a'} q_\theta(S_{t+1}, a')) - q_\theta(S_t, A_t))^2$

Prioritized replay : Instead of sampling uniformly from the replay buffer, transitions containing more information are sampled more often by implementing prioritized experience replay which samples transitions with probability $p_t \propto |R_{t+1} + \gamma_{t+1}\max_{a'} q_{\bar{\theta}}(S_{t+1}, a') - q_\theta(S_t, A_t)|^\omega$. New transitions with maximum priority are added to the replay buffer (recent transitions are provided a bias).

Dueling networks : It is a value based RL architecture with both value (v_η) and advantage (a_ψ) streams (sharing a convolutional encoder f_ζ) merged by a special aggregator. The action value is given by $q_\theta(s,a) = v_\eta(f_\zeta(s)) + a_\psi(f_\zeta(s), a) - \frac{\sum_{a'} a_\psi(f_\zeta(s), a')}{N_{actions}}$; where $\theta = \{\eta, \zeta, \psi\}$

Multi-step learning : Multi-step return is incorporated into the DQN loss, giving the loss $(R_t^{(n)} + \gamma_t^{(n)} \max_{a'} q_{\bar{\theta}}(S_{t+n}, a') - q_\theta(S_t, A_t))^2$. This can lead to faster learning. The *n-step return* is given by $R_t^{(n)} = \sum_{t=0}^{n-1} \gamma_t^{(k)} R_{t+k+1}$.

Distributional RL : The architecture learns to approximate the distribution of returns instead of the expected return. A (discrete) support (a vector with N_{atoms}) is constructed (with $z^i = v_{min} + (i-1)\frac{v_{max} - v_{min}}{N_{atoms} - 1}$ for $i \in \{1, \ldots, N_{atoms}\}$) for the target distribution and the KL divergence between the approximate distribution and target distribution is minimized. Φ_z is the L2-projection of the target distribution onto the fixed support z

Noisy Nets : A noisy linear layer that combines a deterministic and noisy stream ($\boldsymbol{y} = (\boldsymbol{b} + \boldsymbol{W}\boldsymbol{x}) + (\boldsymbol{b}_{noisy} \odot \epsilon^b + (\boldsymbol{W}_{noisy} \odot \epsilon^w)\boldsymbol{x})$) is used to replace the standard linear layer ($\boldsymbol{y} = \boldsymbol{b} + \boldsymbol{W}\boldsymbol{x}$) to enhance exploration (more that ϵ-greedy). This allows for state-conditioned exploration with a form of self-annealing (learn to ignore noise - happens at different rates in different parts of the state-space).

II. Integrated Agent : Rainbow

n-step distributional loss is given by : $D_{KL}(\Phi_z d_t^{(}n) || d_t)$ where the target distribution is $d_t^{(}n) = (R_t^{(}n) + \gamma_t^{(}n) z, p_{\bar{\theta}}(S_{t+n}, a^*_{t+n}))$ and Φ_z is the projection onto z. For including double Q-learning, the greedy action for S_{t+n} is used, selected according to the online network as the bootstrap action a_{t+n}, and evaluating it using the target network. The transitions are prioritized by KL loss ($p_t \propto \left(D_{KL}(\Phi_z d_t^{(}n) || d_t)\right)$) since it might be more robust to noisy stochastic environments as the loss can continue to decrease even when the returns are not deterministic. The network architecture is a dueling network architecture adapted for use with return distributions : $p_\theta^i(s,a) = \frac{exp(v_\eta^i(\phi) + a_\psi^i(\phi,a) - a_\psi^{-i}(s))}{\sum_j exp(v_\eta^j(\phi) + a_\psi^j(\phi,a) - a_\psi^{-j}(s))}$ where $\phi = f_\zeta(s)$ and $a_\psi^{-i}(s) = \frac{1}{N_{actions}} \sum_{a'} a_\psi^i(\phi, a')$; $a_\psi^i(\phi, a)$ is the output corresponding to atom i and action a. All linear layers were replaced with their noisy equivalents (factorised Gaussian noise).

III. Experiments

All the agents were evaluated on 57 Atari 2600 games from the arcade learning environment(ALE). Limited hyperparameter tuning was performed for the Rainbow DQN because the combinatorial space of hyper-parameters were too large for an exhaustive search. Hence the best values reported in the papers of the corresponding components were used and only the most sensitive hyperparameters were manually tuned. Removing Prioritized replay or multi-step learning components caused a large drop in median performance, implying that they are the most crucial components of Rainbow. Distributional Q-learning ranked second to Rainbow. Considering the median performance, the removal of Noisy Nets produced a large drop in performance (ϵ-greedy mechanism is used for exploration: performance was worse in aggregate) for several games and also a small increases in other games. It was hypothesised that clipping the values to a constrained range counteracts the overestimation bias of Q-learning and that the importance of double Q-learning might increase if the support of the distributions is expanded.

References

[1] Matteo Hessel et al. “Rainbow: Combining improvements in deep reinforcement learning”. In: *arXiv preprint arXiv:1710.02298* (2017).