CS7011 : Topics in Reinforcement Learning
Summary of Rainbow DQN : Hessel, Modayil, Hasselt, Schaul

Sahana Ramnath : EE15B109

Abstract

'Rainbow : Combining Improvements in Deep Reinforcement Learning' (Hessel et al., 2017) examines extensions to the DQN algorithm and empirically studies their combination. The many extensions of DQN each address widely different issues regarding its speed and stability. Since all these extensions all built on a shared framework, they can potentially be combined. In some cases, this has already been done; this paper however combines **six** of them in a complementary way. Experiments were conducted on ALE games, resulting in state-of-the-art performance in terms of both final performance and data efficiency.

1. Deep Q-Network

In DQN, a CNN is used to approximate the action values $Q(s,a)$ for a given state s_t(a stack of raw pixel frames). At each step, the agent selects an action ϵ-greedily with respect to $Q(s_t,a)$. The transition $(s_t, a_t, r_{t+1}, \gamma_{t+1}, s_{t+1})$ is added to a replay buffer. The parameters of the neural network are optimized by minimizing the squared error loss $(r_{t+1} + \gamma_{t+1} max_{a'} q_{\theta^-}(s_{t+1}, a') - q_\theta(s_t, a_t))^2$, where time t is picked randomly by sampling the replay buffer. The gradient loss is backpropogated into the parameters θ of the online network whereas θ^- represents the parameters of the target network, a periodic copy of the online network.

2. Extensions to DQN

1. Double Q-Learning : Standard Q-Learning is affected by an overestimation bias due to the maximization step in the parameter update. Double Q-Learning addresses this by decoupling, in the maximization performed for the bootstrap target, the selection of the action from its evaluation : $q_{\theta^-}(s_{t+1}, argmax_{a'} q_\theta(s_{t+1}, a'))$ instead of $q_{\theta^-}(s_{t+1}, a')$
2. Prioritized Replay : DQN samples uniformly from the replay buffer but ideally it should be more frequent for those transitions which offer more to learn. Prioritized experience replay samples transitions with probability relative to the last encountered TD error : $p_t \propto |r_{t+1} + \gamma_{t+1} max_{a'} q_{\theta^-}(s_{t+1}, a') - q_\theta(s_t, a_t)|^w$, where w is a hyperparameter that determines the distribution's shape.
3. Dueling Networks : This is a neural net architecture designed for value-based RL, that features two streams of computation, value and advantage, the two streams sharing a convolutional encoder and merged using a special aggregator.
4. Multi-step Learning : Instead of using a single reward and the next greedy action as the target, this uses a multi-step target : a truncated n-step reward. The loss minimized now $(r_{t+1}^{(n)} + \gamma_t^{(n)} q_{\theta^-}(s_{t+1}, a') - q_\theta(s_t, a_t))^2$.
5. Distributional RL : This learns to approximate the distribution of the return, rather than its expected value; these have to satisfy a variant of Bellman's equation. The distributional variant of Q-learning first constructs a new support for the target distribution, and then minimizes the KL divergence between the current and target distribution.
6. Noisy Nets : Exploring using ϵ-greedy policies have many limitations, the main one being that it constrains exploration to the immediate locality . Noisy Nets proposes a noisy linear layer that combines a deterministic and noisy stream to counter these limitations.

3. The Integrated Agent : Rainbow

The above six components are integrated into a single agent : *Rainbow*.
First, the one step distributional loss is replaced with a multi-step variant. The target distribution is constructed by contracting the value distribution in s_{t+n} according to the cumulative discount, and shifting it by the truncated n-step discounted return.
This loss is combined with Double Q-Learning by using the greedy action at s_{t+n} selected by the online network rather than the target network.
Transitions are prioritized by the KL divergence loss rather than the TD error since this is what the algorithm is minimizing : $p_t \propto$ (KL loss $)^w$.
The network uses a dueling network architecture adapted for use with return distributions.
Finally, all layers are replaced with their noisy equivalents. Factorised gaussian noise is used to reduce the number of independent noise variables.

4. Experiments

Experiments and ablation studies were conducted on all 57 Atari games. Hyperparameter tuning was done in a limited fashion owing to the high combinatorial number of hyperparameters. The best values from each individual paper was used with slight tuning for only the most sensitive of them.
The Rainbow DQN gave state of the art results across all games. Ablation studies proved that the removal of either one of these networks caused a decrease in performance(in most cases).

References

Hessel, Matteo, Modayil, Joseph, Van Hasselt, Hado, Schaul, Tom, Ostrovski, Georg, Dabney, Will, Horgan, Dan, Piot, Bilal, Azar, Mohammad, and Silver, David. Rainbow: Combining improvements in deep reinforcement learning. *arXiv preprint arXiv:1710.02298*, 2017.