MineRL Study Summay

PPO (Proximal Policy Optimization)

PPO （Proximal Policy Optimization）算法是一种改进过的policy gradient算法。通常的policy gradient算法对于stepsize选择非常敏感，如果stepsize太小，优化会很慢，而如果stepsize太大，信号可能会淹没在噪声中，比较难得到好的结果。

在PPO 算法中引入了K-L 散度作为loss function 中的一项作为约束来控制每次迭代中的policy变化程度。于是loss function的结构为：

$$L^{CLIP}(\theta) = \hat{E}_t\left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t)\right]$$

- θ is the policy parameter
- $\hat{E}_t$ denotes the empirical expectation over timesteps
- r_t is the ratio of the probability under the new and old policies, respectively
- $\hat{A}_t$ is the estimated advantage at time t
- ε is a hyperparameter, usually 0.1 or 0.2

Reference: [1], [2] ,[3]

PPO result

```
INFO     - 2020-08-04 06:46:54,643 - [chainerrl.experiments.train_agent train_agent 60] statistics:[('average_value', 1.1758298345804215), ('average_entropy
', 1.4586308987736702), ('average_value_loss', 0.12506614780053496), ('average_policy_loss', -0.09961950048804283), ('n_updates', 186528), ('explained_varia
nce', 0.2966157596032608)]
INFO     - 2020-08-04 06:49:00,836 - [chainerrl.experiments.train_agent train_agent 59] outdir:results/MineRLTreechop-v0/ppo\20200803T004120.781889 step:199
2490 episode:1011 R:19.0
INFO     - 2020-08-04 06:49:00,837 - [chainerrl.experiments.train_agent train_agent 60] statistics:[('average_value', 0.886482425391674), ('average_entropy'
, 1.4941473297476768), ('average_value_loss', 0.07142692860215902), ('average_policy_loss', 0.06153009800240397), ('n_updates', 186720), ('explained_varianc
e', 0.3203819062158688)]
INFO     - 2020-08-04 06:50:58,970 - [chainerrl.experiments.train_agent train_agent 59] outdir:results/MineRLTreechop-v0/ppo\20200803T004120.781889 step:199
4490 episode:1012 R:12.0
INFO     - 2020-08-04 06:50:58,971 - [chainerrl.experiments.train_agent train_agent 60] statistics:[('average_value', 0.8206746428608894), ('average_entropy
', 1.5127030355930329), ('average_value_loss', 0.062378294244408605), ('average_policy_loss', 0.056764109483920036), ('n_updates', 186912), ('explained_vari
ance', 0.505989884195077)]
INFO     - 2020-08-04 06:52:15,532 - [chainerrl.experiments.train_agent train_agent 59] outdir:results/MineRLTreechop-v0/ppo\20200803T004120.781889 step:199
5650 episode:1013 R:7.0
INFO     - 2020-08-04 06:52:15,532 - [chainerrl.experiments.train_agent train_agent 60] statistics:[('average_value', 1.024835336983204), ('average_entropy'
, 1.473233350932598), ('average_value_loss', 0.03843395164236427), ('average_policy_loss', 0.08508575239218771), ('n_updates', 187008), ('explained_variance
', 0.2700569731334782)]
INFO     - 2020-08-04 06:54:19,802 - [chainerrl.experiments.train_agent train_agent 59] outdir:results/MineRLTreechop-v0/ppo\20200803T004120.781889 step:199
7650 episode:1014 R:24.0
INFO     - 2020-08-04 06:54:19,803 - [chainerrl.experiments.train_agent train_agent 60] statistics:[('average_value', 0.8410502589344978), ('average_entropy
', 1.4958220786452294), ('average_value_loss', 0.11406988769769669), ('average_policy_loss', -0.08385579662397503), ('n_updates', 187200), ('explained_varia
nce', 0.3297621994279883)]
INFO     - 2020-08-04 06:56:22,630 - [chainerrl.experiments.train_agent train_agent 59] outdir:results/MineRLTreechop-v0/ppo\20200803T004120.781889 step:199
9650 episode:1015 R:23.0
INFO     - 2020-08-04 06:56:22,630 - [chainerrl.experiments.train_agent train_agent 60] statistics:[('average_value', 1.165285474061966), ('average_entropy'
, 1.4615517191290854), ('average_value_loss', 0.09490192923694848), ('average_policy_loss', -0.0314130116626620 3), ('n_updates', 187392), ('explained_varian
ce', 0.44006689537809873)]
INFO     - 2020-08-04 06:57:00,689 - [chainerrl.experiments.train_agent train_agent 59] outdir:results/MineRLTreechop-v0/ppo\20200803T004120.781889 step:200
0000 episode:1016 R:7.0
INFO     - 2020-08-04 06:57:00,689 - [chainerrl.experiments.train_agent train_agent 60] statistics:[('average_value', 1.0711385462284089), ('average_entropy
', 1.4789219467043877), ('average_value_loss', 0.10252851694822311), ('average_policy_loss', -0.00615884703118354), ('n_updates', 187488), ('explained_varia
nce', 0.23340480777769868)]
INFO     - 2020-08-04 06:57:00,872 - [chainerrl.experiments.train_agent save_agent 266] Saved the agent to results/MineRLTreechop-v0/ppo\20200803T004120.781
889\2000000_finish
```

DQN (Deep Q Network)

DQN是一种value based 的强化学习方法，是Deep Learning与Reinforcement Learning的结合。DQN使用深度卷积神经网络逼近值函数,并且使用了经验回放(*Experience replay)对学习过程进行训练。