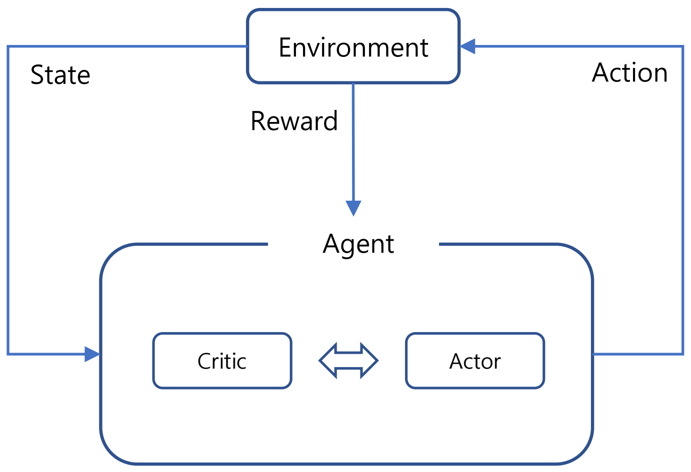
Ensemble Critic

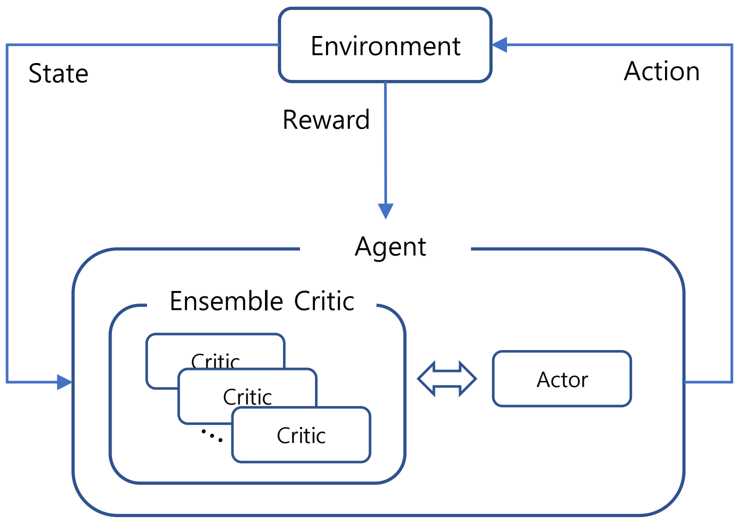
2023/01/26

Ensemble Critic

학습 안정성을 높이기 위해, 2개 이상의 Critic model을 사용



현재



수정

Off-Policy PPO

1. Take action $\boldsymbol{a} \sim \pi(\cdot\,|\boldsymbol{s})$, get $(\boldsymbol{s}, \boldsymbol{a}, r, \boldsymbol{s}', \log\pi(\boldsymbol{a}|\boldsymbol{s}))$ and store in R
2. Sample a batch $\{\boldsymbol{s}_i, \boldsymbol{a}_i, r_i, \boldsymbol{s}_i', \log\pi(\boldsymbol{a}_i|\boldsymbol{s}_i)\}$ from buffer R
3. Update $\hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i)$ using target $y_i = r_i + \gamma \frac{1}{M} \sum_{\boldsymbol{a}_t'} \hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i', \boldsymbol{a}_i')$; $\boldsymbol{a}_i' \sim \pi_{\theta_t}(\cdot\,|\boldsymbol{s}_i')$
4. Evaluate $\hat{A}^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i) = \hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i) - \frac{1}{M} \sum_{\boldsymbol{a}_i^t} \hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i^t)$; $\boldsymbol{a}_i^t \sim \pi_{\theta_t}(\cdot\,|\boldsymbol{s}_i)$
5. Minibatch Learning on $\{\boldsymbol{s}_i, \boldsymbol{a}_i^t, \log\pi_{\theta_t}(\boldsymbol{a}_i^t|\boldsymbol{s}_i)\}$; $\boldsymbol{a}_i^t \sim \pi_{\theta_t}(\cdot\,|\boldsymbol{s}_i)$
   1. $J_t(\theta) = \frac{1}{N} \sum_i \min\left(\frac{\pi_\theta(\boldsymbol{a}_i^t|\boldsymbol{s}_i)}{\pi_{\theta_t}(\boldsymbol{a}_i^t|\boldsymbol{s}_i)} \hat{A}^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i^t), \sim\right)$ and $\theta^1 = \theta_t$
   2. $for\ k = 1, \dots, K\ do\ \theta^{k+1} \leftarrow \theta^k + \alpha \nabla_{\theta^k} J_t(\theta^k)$
   3. $\theta_{t+1} = \theta^K$
6. Repeat.

Line 3&4의 $\hat{Q}_\phi^{\pi_{\theta_t}}$ 계산에 K개 critic의 평균을 사용.

$$\hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}, \boldsymbol{a}) = \frac{1}{K} \sum_{k=1}^{K} \hat{Q}_{\phi_k}^{\pi_{\theta_t}}(\boldsymbol{s}, \boldsymbol{a})$$

기존의 학습 코드/알고리즘을 그대로 유지하면서, Critic의 target value, Actor의 advantage 계산에 K개 critic의 평균을 사용.

Off-Policy PPO

1. Take action $\boldsymbol{a} \sim \pi(\cdot\,|\boldsymbol{s})$, get $(\boldsymbol{s}, \boldsymbol{a}, r, \boldsymbol{s}', \log\pi(\boldsymbol{a}|\boldsymbol{s}))$ and store in R
2. Sample a batch $\{\boldsymbol{s}_i, \boldsymbol{a}_i, r_i, \boldsymbol{s}_i', \log\pi(\boldsymbol{a}_i|\boldsymbol{s}_i)\}$ from buffer R
3. Update $\hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i)$ using target $y_i = r_i + \gamma \frac{1}{M}\sum_{\boldsymbol{a}_t'} \hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i', \boldsymbol{a}_i')$; $\boldsymbol{a}_i' \sim \pi_{\theta_t}(\cdot\,|\boldsymbol{s}_i')$
4. Evaluate $\hat{A}^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i) = \hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i) - \frac{1}{M}\sum_{\boldsymbol{a}_i^t} \hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i^t)$; $\boldsymbol{a}_i^t \sim \pi_{\theta_t}(\cdot\,|\boldsymbol{s}_i)$
5. Minibatch Learning on $\{\boldsymbol{s}_i, \boldsymbol{a}_i^t, \log\pi_{\theta_t}(\boldsymbol{a}_i^t|\boldsymbol{s}_i)\}$; $\boldsymbol{a}_i^t \sim \pi_{\theta_t}(\cdot\,|\boldsymbol{s}_i)$
   1. $J_t(\theta) = \frac{1}{N}\sum_i \min\left(\frac{\pi_\theta(\boldsymbol{a}_i^t|\boldsymbol{s}_i)}{\pi_{\theta_t}(\boldsymbol{a}_i^t|\boldsymbol{s}_i)} \hat{A}^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i^t), \sim\right)$ and $\theta^1 = \theta_t$
   2. $for\ k = 1, \dots, K\ do\ \theta^{k+1} \leftarrow \theta^k + \alpha\nabla_{\theta^k} J_t(\theta^k)$
   3. $\theta_{t+1} = \theta^K$
6. Repeat.

Line 3&4의 $\hat{Q}_\phi^{\pi_{\theta_t}}$ 계산에 K개 critic의 평균을 사용.

$$\hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}, \boldsymbol{a}) = \frac{1}{K}\sum_{k=1}^{K} \hat{Q}_{\phi_k}^{\pi_{\theta_t}}(\boldsymbol{s}, \boldsymbol{a})$$

ppo.py#365

```
def predict(self, state, action):

    values = 0
    for model in self.models:
        values_ = model.predict([state, action], verbose=0)
        values += values_

    values = values / self.num_critics

    return values
```

아래의 코드를 대체

```
def predict(self, state):
    return self.model.predict([state, np.zeros((state.shape[0], 1))])
```

https://github.com/etri-city-traffic-brain/traffic-signal-optimization/blob/8ac45baeedda25f78d5343223ac2edee7aa44ee8/atsc-rl/multiagent_tf2/policy/ppoTF2.py#L150

Off-Policy PPO

1. Take action $\boldsymbol{a} \sim \pi(\cdot\,|\boldsymbol{s})$, get $(\boldsymbol{s}, \boldsymbol{a}, r, \boldsymbol{s}', \log \pi(\boldsymbol{a}|\boldsymbol{s}))$ and store in R
2. Sample a batch $\{\boldsymbol{s}_i, \boldsymbol{a}_i, r_i, \boldsymbol{s}_i', \log \pi(\boldsymbol{a}_i|\boldsymbol{s}_i)\}$ from buffer R
3. Update $\hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i)$ using target $y_i = r_i + \gamma \frac{1}{M} \sum_{\boldsymbol{a}_t'} \hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i', \boldsymbol{a}_i')$; $\boldsymbol{a}_i' \sim \pi_{\theta_t}(\cdot\,|\boldsymbol{s}_i')$
4. Evaluate $\hat{A}^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i) = \hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i) - \frac{1}{M} \sum_{\boldsymbol{a}_i^t} \hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i^t)$; $\boldsymbol{a}_i^t \sim \pi_{\theta_t}(\cdot\,|\boldsymbol{s}_i)$
5. Minibatch Learning on $\{\boldsymbol{s}_i, \boldsymbol{a}_i^t, \log \pi_{\theta_t}(\boldsymbol{a}_i^t|\boldsymbol{s}_i)\}$; $\boldsymbol{a}_i^t \sim \pi_{\theta_t}(\cdot\,|\boldsymbol{s}_i)$
   1. $J_t(\theta) = \frac{1}{N} \sum_i \min\left(\frac{\pi_\theta(\boldsymbol{a}_i^t|\boldsymbol{s}_i)}{\pi_{\theta_t}(\boldsymbol{a}_i^t|\boldsymbol{s}_i)} \hat{A}^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i^t), \sim\right)$ and $\theta^1 = \theta_t$
   2. $for\ k = 1, \ldots, K\ do\ \theta^{k+1} \leftarrow \theta^k + \alpha \nabla_{\theta^k} J_t(\theta^k)$
   3. $\theta_{t+1} = \theta^K$
6. Repeat.

Line 3&4의 $\hat{Q}_\phi^{\pi_{\theta_t}}$ 계산에 K개 critic의 평균을 사용.

$$\boxed{\hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}, \boldsymbol{a}) = \frac{1}{K} \sum_{k=1}^{K} \hat{Q}_{\phi_k}^{\pi_{\theta_t}}(\boldsymbol{s}, \boldsymbol{a})}$$

ppo.py#365

```
def predict(self, state, action):

    values = 0
    for model in self.models:
        values_ = model.predict([state, action], verbose=0)
        values += values_

    values = values / self.num_critics

    return values
```

K개의 critic: $\hat{Q}_\phi^{\pi_{\theta_t}}$

Off-Policy PPO

1. Take action $\boldsymbol{a} \sim \pi(\cdot|\boldsymbol{s})$, get $(\boldsymbol{s}, \boldsymbol{a}, r, \boldsymbol{s}', \log\pi(\boldsymbol{a}|\boldsymbol{s}))$ and store in R
2. Sample a batch $\{\boldsymbol{s}_i, \boldsymbol{a}_i, r_i, \boldsymbol{s}_i', \log\pi(\boldsymbol{a}_i|\boldsymbol{s}_i)\}$ from buffer R
3. Update $\hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i)$ using target $y_i = r_i + \gamma\frac{1}{M}\sum_{\boldsymbol{a}_t'}\hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i', \boldsymbol{a}_i')$; $\boldsymbol{a}_i' \sim \pi_{\theta_t}(\cdot|\boldsymbol{s}_i')$
4. Evaluate $\hat{A}^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i) = \hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i) - \frac{1}{M}\sum_{\boldsymbol{a}_i^t}\hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i^t)$; $\boldsymbol{a}_i^t \sim \pi_{\theta_t}(\cdot|\boldsymbol{s}_i)$
5. Minibatch Learning on $\{\boldsymbol{s}_i, \boldsymbol{a}_i^t, \log\pi_{\theta_t}(\boldsymbol{a}_i^t|\boldsymbol{s}_i)\}$; $\boldsymbol{a}_i^t \sim \pi_{\theta_t}(\cdot|\boldsymbol{s}_i)$
   1. $J_t(\theta) = \frac{1}{N}\sum_i \min\left(\frac{\pi_\theta(\boldsymbol{a}_i^t|\boldsymbol{s}_i)}{\pi_{\theta_t}(\boldsymbol{a}_i^t|\boldsymbol{s}_i)}\hat{A}^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i^t), \sim\right)$ and $\theta^1 = \theta_t$
   2. $for\ k = 1, \dots, K\ do\ \theta^{k+1} \leftarrow \theta^k + \alpha\nabla_{\theta^k}J_t(\theta^k)$
   3. $\theta_{t+1} = \theta^K$
6. Repeat.

Line 3&4의 $\hat{Q}_\phi^{\pi_{\theta_t}}$ 계산에 K개 critic의 평균을 사용.

$$\hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}, \boldsymbol{a}) = \frac{1}{K}\sum_{k=1}^{K}\boxed{\hat{Q}_{\phi_k}^{\pi_{\theta_t}}(\boldsymbol{s}, \boldsymbol{a})}$$

ppo.py#365

```python
def predict(self, state, action):

    values = 0
    for model in self.models:
        values_ = model.predict([state, action], verbose=0)
        values += values_

    values = values / self.num_critics

    return values
```

k번째 critic, i.e, $\hat{Q}_{\phi_k}^{\pi_{\theta_t}}$

Off-Policy PPO

1. Take action $\boldsymbol{a} \sim \pi(\cdot\,|\boldsymbol{s})$, get $(\boldsymbol{s}, \boldsymbol{a}, r, \boldsymbol{s}', \log\pi(\boldsymbol{a}|\boldsymbol{s}))$ and store in R
2. Sample a batch $\{\boldsymbol{s}_i, \boldsymbol{a}_i, r_i, \boldsymbol{s}_i', \log\pi(\boldsymbol{a}_i|\boldsymbol{s}_i)\}$ from buffer R
3. Update $\hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i)$ using target $y_i = r_i + \gamma \frac{1}{M}\sum_{\boldsymbol{a}_t'} \hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i', \boldsymbol{a}_i')$; $\boldsymbol{a}_i' \sim \pi_{\theta_t}(\cdot\,|\boldsymbol{s}_i')$
4. Evaluate $\hat{A}^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i) = \hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i) - \frac{1}{M}\sum_{\boldsymbol{a}_i^t} \hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i^t)$; $\boldsymbol{a}_i^t \sim \pi_{\theta_t}(\cdot\,|\boldsymbol{s}_i)$
5. Minibatch Learning on $\{\boldsymbol{s}_i, \boldsymbol{a}_i^t, \log\pi_{\theta_t}(\boldsymbol{a}_i^t|\boldsymbol{s}_i)\}$; $\boldsymbol{a}_i^t \sim \pi_{\theta_t}(\cdot\,|\boldsymbol{s}_i)$
   1. $J_t(\theta) = \frac{1}{N}\sum_i \min\left(\frac{\pi_\theta(\boldsymbol{a}_i^t|\boldsymbol{s}_i)}{\pi_{\theta_t}(\boldsymbol{a}_i^t|\boldsymbol{s}_i)} \hat{A}^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i^t), \sim\right)$ and $\theta^1 = \theta_t$
   2. $for\ k = 1, \ldots, K\ do\ \theta^{k+1} \leftarrow \theta^k + \alpha \nabla_{\theta^k} J_t(\theta^k)$
   3. $\theta_{t+1} = \theta^K$
6. Repeat.

Line 3&4의 $\hat{Q}_\phi^{\pi_{\theta_t}}$ 계산에 K개 critic의 평균을 사용.

$$\hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}, \boldsymbol{a}) = \frac{1}{K}\sum_{k=1}^{K} \hat{Q}_{\phi_k}^{\pi_{\theta_t}}(\boldsymbol{s}, \boldsymbol{a})$$

ppo.py#365

```
def predict(self, state, action):

    values = 0
    for model in self.models:
        values_ = model.predict([state, action], verbose=0)
        values += values_

    values = values / self.num_critics

    return values
```

k번째 critic에 의해 계산된 state-action value

Off-Policy PPO

1. Take action $\boldsymbol{a} \sim \pi(\cdot\,|\boldsymbol{s})$, get $(\boldsymbol{s}, \boldsymbol{a}, r, \boldsymbol{s}', \log\pi(\boldsymbol{a}|\boldsymbol{s}))$ and store in R
2. Sample a batch $\{\boldsymbol{s}_i, \boldsymbol{a}_i, r_i, \boldsymbol{s}_i', \log\pi(\boldsymbol{a}_i|\boldsymbol{s}_i)\}$ from buffer R
3. Update $\hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i)$ using target $y_i = r_i + \gamma \frac{1}{M}\sum_{\boldsymbol{a}_t'} \hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i', \boldsymbol{a}_i')$; $\boldsymbol{a}_i' \sim \pi_{\theta_t}(\cdot\,|\boldsymbol{s}_i')$
4. Evaluate $\hat{A}^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i) = \hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i) - \frac{1}{M}\sum_{\boldsymbol{a}_i^t} \hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i^t)$; $\boldsymbol{a}_i^t \sim \pi_{\theta_t}(\cdot\,|\boldsymbol{s}_i)$
5. Minibatch Learning on $\{\boldsymbol{s}_i, \boldsymbol{a}_i^t, \log\pi_{\theta_t}(\boldsymbol{a}_i^t|\boldsymbol{s}_i)\}$; $\boldsymbol{a}_i^t \sim \pi_{\theta_t}(\cdot\,|\boldsymbol{s}_i)$
    1. $J_t(\theta) = \frac{1}{N}\sum_i \min\left(\frac{\pi_\theta(\boldsymbol{a}_i^t|\boldsymbol{s}_i)}{\pi_{\theta_t}(\boldsymbol{a}_i^t|\boldsymbol{s}_i)} \hat{A}^{\pi_{\theta_t}}(\boldsymbol{s}_i, \boldsymbol{a}_i^t), \sim\right)$ and $\theta^1 = \theta_t$
    2. $for\ k = 1, \ldots, K\ do\ \theta^{k+1} \leftarrow \theta^k + \alpha\nabla_{\theta^k} J_t(\theta^k)$
    3. $\theta_{t+1} = \theta^K$
6. Repeat.

Line 3&4의 $\hat{Q}_\phi^{\pi_{\theta_t}}$ 계산에 K개 critic의 평균을 사용.

$$\boxed{\hat{Q}_\phi^{\pi_{\theta_t}}(\boldsymbol{s}, \boldsymbol{a}) = \frac{1}{K}\sum_{k=1}^{K} \hat{Q}_{\phi_k}^{\pi_{\theta_t}}(\boldsymbol{s}, \boldsymbol{a})}$$

ppo.py#365

```
def predict(self, state, action):

    values = 0
    for model in self.models:
        values_ = model.predict([state, action], verbose=0)
        values += values_

    values = values / self.num_critics

    return values
```

K개의 critic에 의해 계산된 평균
기존의 코드/알고리즘을 그대로 유지하면서,
Critic의 target value, Actor의 advantage 계산
에 평균을 사용.

Ensemble Critic 모델 생성

ppo.py#264
CriticModel.__init__()

```python
def __init__(self, network_layers, input_shape, action_space, lr, optimizer):

    self.num_critics = 5

    self.models = []
    for i in range(self.num_critics):
        model = self.buildDNN(network_layers, input_shape, action_space, lr, optimizer)
        self.models.append(model)
```

buildNN() 함수를 이용하여 여러 개의 critic model을 생성.

ppo.py#275
CriticModel.buildDNN()

```python
def buildDNN(self, network_layers, input_shape, action_space, lr, optimizer):

    network_layers = (1024, 512, 512, 512, 512, 512, 512, 512, 512, 512, 512)
    #X_input = Input(input_shape)
    #old_values = Input(shape=(1,))
    state_input = Input(input_shape)
    action_input = Input(shape=(action_space, ))

    state_stream = state_input
    action_stream = action_input


    for i, size in enumerate(network_layers):

        if i == 0:
            kernel_regularizer = tf.keras.regularizers.l2(l=0.00005)
            #kernel_regularizer = tf.keras.regularizers.l2(l=0.0005)

            state_stream = Dense(size, activation=tf.nn.tanh, kernel_initializer='
            action_stream = Dense(size, activation=tf.nn.tanh, kernel_initializer=

            #V = Dense(size, activation=tf.nn.tanh, kernel_initializer=tf.random_r
        if i == 1:
            state_stream  = BN_Tanh_Dense(size)(state_stream)
            action_stream = BN_Tanh_Dense(size)(action_stream)

        #if 1 <= i <= 2:
        if 1 <= i <= 5:
            state_stream  = ResBlock(size)(state_stream)
            action_stream  = ResBlock(size)(action_stream)

        #if i == 2:
        if i == 5:
            V = tf.concat([state_stream, action_stream], axis=-1)
            V = BN_Tanh_Dense(size)(V)

        #if i > 2:
        if i > 5:
            V = ResBlock(size)(V)


    value = BN_Tanh_Dense(1, kernel_regularizer=None, use_bias=True)(V)


    model = Model(inputs=[state_input, action_input], outputs=value)
    model.compile(loss=self.critic_PPO2_loss, optimizer=optimizer(lr=lr))

    return model
```

Q & A