- `def createNetwork():` we can consider the network as Value Function Approximation.

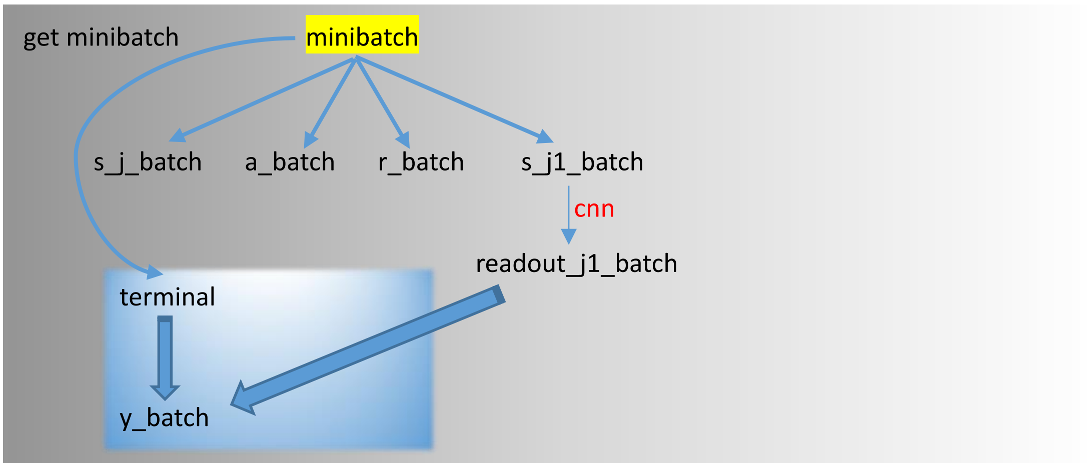
- `def trainNetwork(s, readout, h_fc1, sess):`

case 1: `if t > OBSERVE: false`

```
while "flappy bird" != "angry bird":

                cnn
    readout_t <---- s_t

             ε-greedy

    choose an action a_t by ε-greedy

    [scale down epsilon]

    # run the selected action and observe next state and reward
    x_t1_colored, r_t, terminal = game_state.frame_step(a_t)
  image processing
    s_t1

    [store the transition in deque D]

    if t > OBSERVE: false

    s_t = s_t1

    t += 1
```

case 2: if t > OBSERVE: true

```
while "flappy bird" != "angry bird":

                cnn
    readout_t <---- s_t
           ε -greedy
    choose an action a_t by ε -greedy

    [scale down epsilon]

    # run the selected action and observe next state and reward
    x_t1_colored, r_t, terminal = game_state.frame_step(a_t)
  image processing
    s_t1

    [store the transition in deque D]

    if t > OBSERVE: true
```



```
        # perform gradient step to minimize cost
        train_step.run(feed_dict={y: y_batch, a: a_batch, s: s_j_batch})

    s_t = s_t1

    t += 1
```