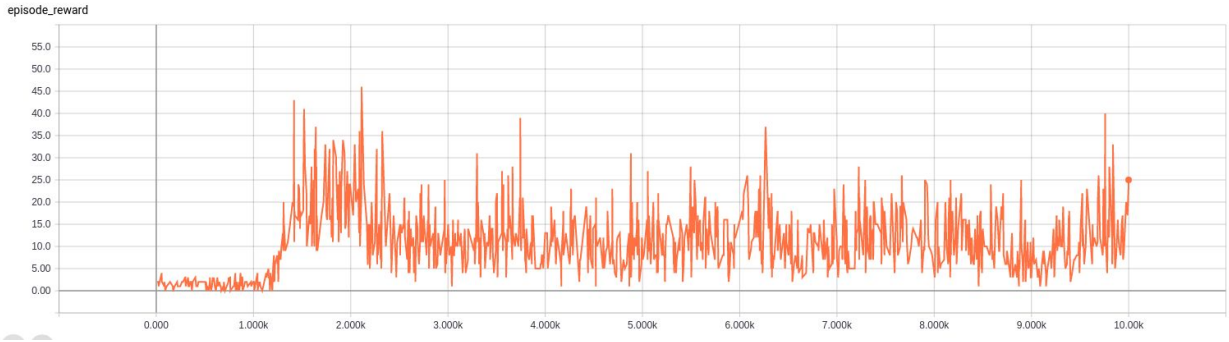
Report

1. **Plot showing epsode rewards of 10000 training episodes.**



2. **Explain deep Q network implementation**
   a. **Network Structure**

Name	Parameters
Conv 1	filters=32, stride=4, kernel_size=8, padding='SAME', activation_fn=relu, weight_init=normal_init(mean=0, stddev=0.01), bias_init=normal_init(mean=0, stddev=0.01)
Conv 2	filters=64, stride=2, kernel_size=4, padding='SAME', activation_fn=relu, weight_init=normal_init(mean=0, stddev=0.01), bias_init=normal_init(mean=0, stddev=0.01)
Conv 3	filters=64, stride=1, kernel_size=3, padding='SAME', activation_fn=relu, weight_init=normal_init(mean=0, stddev=0.01), bias_init=normal_init(mean=0, stddev=0.01)
Fcon 1	output=512, activation_fn=relu, weight_init=normal_init(mean=0, stddev=0.01), bias_init=normal_init(mean=0, stddev=0.01)
Fcon 2	output=4, activation_fn=relu, weight_init=normal_init(mean=0, stddev=0.01), bias_init=normal_init(mean=0, stddev=0.01)

b. **Loss Function**
   MSE (Mean Square Error)
   Use target network get **Label Y** from current reward and next-state Q value; use behavior network get **Logit X** from current-state Q value. Then, calculate the MSE by **Y** and **X**.
   (note that next-state Q value is depend on the action get from behavior network)

3. **Describe the implementation of update_target_network()**
   a. Collecting all variable from tf.global_variables()
   b. Filtering those variables into two groups by the scope name of behavior network and target network.
   c. Assigning variables in behavior network to variables in target network.
   d. tf.Session().run()
4. **Explain the implementation of training process of deep Q learning**
   a. **Populate replay memory**
      Append transition into list of replay_memory every movement. If the size of replay_memory exceed the maximum size, pop out the oldest one.
   b. **Select actions**
      Depend on the epsilon, if the random number smaller than epsilon, select one action randomly, otherwise select one largest Q value from behavior network.
   c. **Update epsilon**
      Decreasing from 1.0 to 0.1 by 500000 times.
   d. **Prepare minibatch for network update**
      Randomly sample the batch from replay_memery every time.
5. **Performance**
   Highest episode reward during training: 46
   (Actually I got highest episode reward as 77. But my code about tf.Summary() had bugs, I forgot to use Writer.flush() every-time to flush the data into disk, which result some of records are missing QQ)