Actor-Attention-Critic for Multi-Agent Reinforcement Learning Supplementary Material

Shariq Iqbal [1] **Fei Sha** [1 2]

Algorithm 1 Training Procedure for Attention-Actor-Critic

Initialize E parallel environments with N agents
Initialize replay buffer, D
$T_{\text{update}} \leftarrow 0$
for $i_{\text{ep}} = 1 \ldots$ num episodes **do**
  Reset environments, and get initial o_i^e for each agent, i
  for $t = 1 \ldots$ steps per episode **do**
    Select actions $a_i^e \sim \pi_i(\cdot|o_i^e)$ for each agent, i, in each environment, e
    Send actions to all parallel environments and get o'^e_i, r_i^e for all agents
    Store transitions for all environments in D
    $T_{\text{update}} = T_{\text{update}} + E$
    if $T_{\text{update}} \geq$ min steps per update **then**
      for $j = 1 \ldots$ num critic updates **do**
        Sample minibatch, B
        UPDATECRITIC(B)
      end for
      for $j = 1 \ldots$ num policy updates **do**
        Sample $m \times (o_{1\ldots N}) \sim D$
        UPDATEPOLICIES($o^B_{1\ldots N}$)
      end for
      Update target parameters:

$$\bar{\psi} = \tau\bar{\psi} + (1-\tau)\psi$$

$$\bar{\theta} = \tau\bar{\theta} + (1-\tau)\theta$$

      $T_{\text{update}} \leftarrow 0$
    end if
  end for
end for

Algorithm 2 Update Calls for Critic and Policies

function UPDATECRITIC(B)
  Unpack minibatch $(o^B_{1\ldots N}, a^B_{1\ldots N}, r^B_{1\ldots N}, o'^B_{1\ldots N}) \leftarrow B$
  Calculate $Q_i^\psi(o^B_{1\ldots N}, a^B_{1\ldots N})$ for all i in parallel
  Calculate $a'^B_i \sim \pi_i^{\bar{\theta}}(o'^B_i)$ using target policies
  Calculate $Q_i^{\bar{\psi}}(o'^B_{1\ldots N}, a'^B_{1\ldots N})$ for all i in parallel, using target critic
  Update critic using $\nabla\mathcal{L}_Q(\psi)$ and Adam (Kingma & Ba, 2014)
end function

function UPDATEPOLICIES($o^B_{1\ldots N}$)
  Calculate $a^B_{1\ldots N} \sim \pi_i^{\bar{\theta}}(o'^B_i), i \in 1 \ldots N$
  Calculate $Q_i^\psi(o^B_{1\ldots N}, a^B_{1\ldots N})$ for all i in parallel
  Update policies using $\nabla_{\theta_i} J(\pi_\theta)$ and Adam (Kingma & Ba, 2014)
end function

1. Training Procedure

We train using Soft Actor-Critic (Haarnoja et al., 2018), an off-policy, actor-critic method for maximum entropy reinforcement learning. Our training procedure consists of performing 12 parallel rollouts, and adding a tuple of $(o_t, a_t, r_t, o_{t+1})_{1\ldots N}$ to a replay buffer (with maximum length 1e6) for each timepoint. We reset each environment after every 100 steps (an episode). After 100 steps (across all rollouts), we perform 4 updates for the attention critic and for all policies. For each update we sample minibatches of 1024 timepoints from the replay buffer and then perform gradient descent on the Q-function loss objective, as well as the policy objective, using Adam (Kingma & Ba, 2014) as the optimizer for both with a learning rate of 0.001. These updates can be computed efficiently in parallel (across agents) using a GPU. After the updates are complete, we update the parameters $\bar{\psi}$ of our target critic $Q_{\bar{\psi}}$ to move toward our learned critic's parameters, ψ, as in Lillicrap et al. (2016); Haarnoja et al. (2018): $\bar{\psi} = (1-\tau)\bar{\psi} + \tau\psi$, where τ is the update rate (set to 0.005). Using a target critic has been shown to stabilize the use of experience replay for off-policy reinforcement learning with neural network func-

[1]Department of Computer Science, University of Southern California [2]On leave at Google AI (fsha@google.com). Correspondence to: Shariq Iqbal <shariqiq@usc.edu>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

tion approximators (Mnih et al., 2015; Lillicrap et al., 2016). We update the parameters of the target policies, $\bar{\theta}$ in the same manner. We use a discount factor, γ, of 0.99. All networks (separate policies and those contained within the centralized critics) use a hidden dimension of 128 and Leaky Rectified Linear Units as the nonlinearity. We use 0.01 as our temperature setting for Soft Actor-Critic. Additionally, we use 4 attention heads in our attention critics.

2. Reparametrization of DDPG/MADDPG for Discrete Action Spaces

In order to compare to DDPG and MADDPG in our environments with discrete action spaces, we must make a slight modification to the basic algorithms. This modification is first suggested by Lowe et al. (2017) in order to enable policies that output discrete communication messages. Consider the original DDPG policy gradient which takes advantage of the fact that we can easily calculate the gradient of the output of a deterministic policy with respect to its parameters:

$$\nabla_\theta J = \mathbb{E}_{s\sim\rho}\left[\nabla_a Q(s,a)|_{a=\mu(s)} \nabla_\theta \mu(s|\theta)\right]$$

Rather than using policies that deterministically output an action from within a continuous action space, we use policies that produce differentiable samples through a Gumbel-Softmax distribution (Jang et al., 2017). Using differentiable samples allows us to use the gradient of expected returns to train policies without using the log derivative trick, just as in DDPG:

$$\nabla_\theta J = \mathbb{E}_{s\sim\rho, a\sim\pi(s)}\left[\nabla_a Q(s,a) \nabla_\theta a\right]$$

3. Visualizing Attention

In order to understand how the use of attention evolves over the course of training, we examine the "entropy" of the attention weights for each agent for each of the four attention heads that we use in both tasks (Figures 1 and 2). The black bars indicate the maximum possible entropy (i.e. uniform attention across all agents). Lower entropy indicates that the head is focusing on specific agents, with an entropy of 0 indicating attention focusing on one agent. In Rover-Tower, we plot the attention entropy for each rover. Interestingly, each agent appears to use a different combination of the four heads, but their use is not mutually exclusive, indicating that the inclusion of separate attention heads for each agent is not necessary. This differential use of attention heads is sensible due to the nature of rewards in this environment (i.e. individualized rewards). In the case of Collective Treasure Collection, we find that all agents use the attention heads similarly, which is unsurprising considering that rewards are shared in that environment.

In order to inspect how the attention mechanism is working on a more fine-grained level, we visualize the attention weights for one of the rovers in Rover-Tower (Figure 3), from the head that the agent appears to use the most (determined by looking at Figure 1), while changing the tower that said rover is paired to. In these plots, we ignore the weights over other rovers for simplicity since these are always near zero. We find that the rover learns to strongly attend to the tower that it is paired with, without any explicit supervision signal to do so. The model implicitly learns which agent is most relevant to estimating the rover's expected future returns, and said agent can change dynamically without affecting the performance of the algorithm.

References

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870, Stockholmsmssan, Stockholm Sweden, 10–15 Jul 2018.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2014.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*, 2016.

Lowe, R., Wu, Y., Tamar, A., Harb, J., Abbeel, O. P., and Mordatch, I. Multi-agent actor-critic for mixed cooperative-competitive environments. In *Advances in Neural Information Processing Systems*, pp. 6382–6393, 2017.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540): 529, 2015.

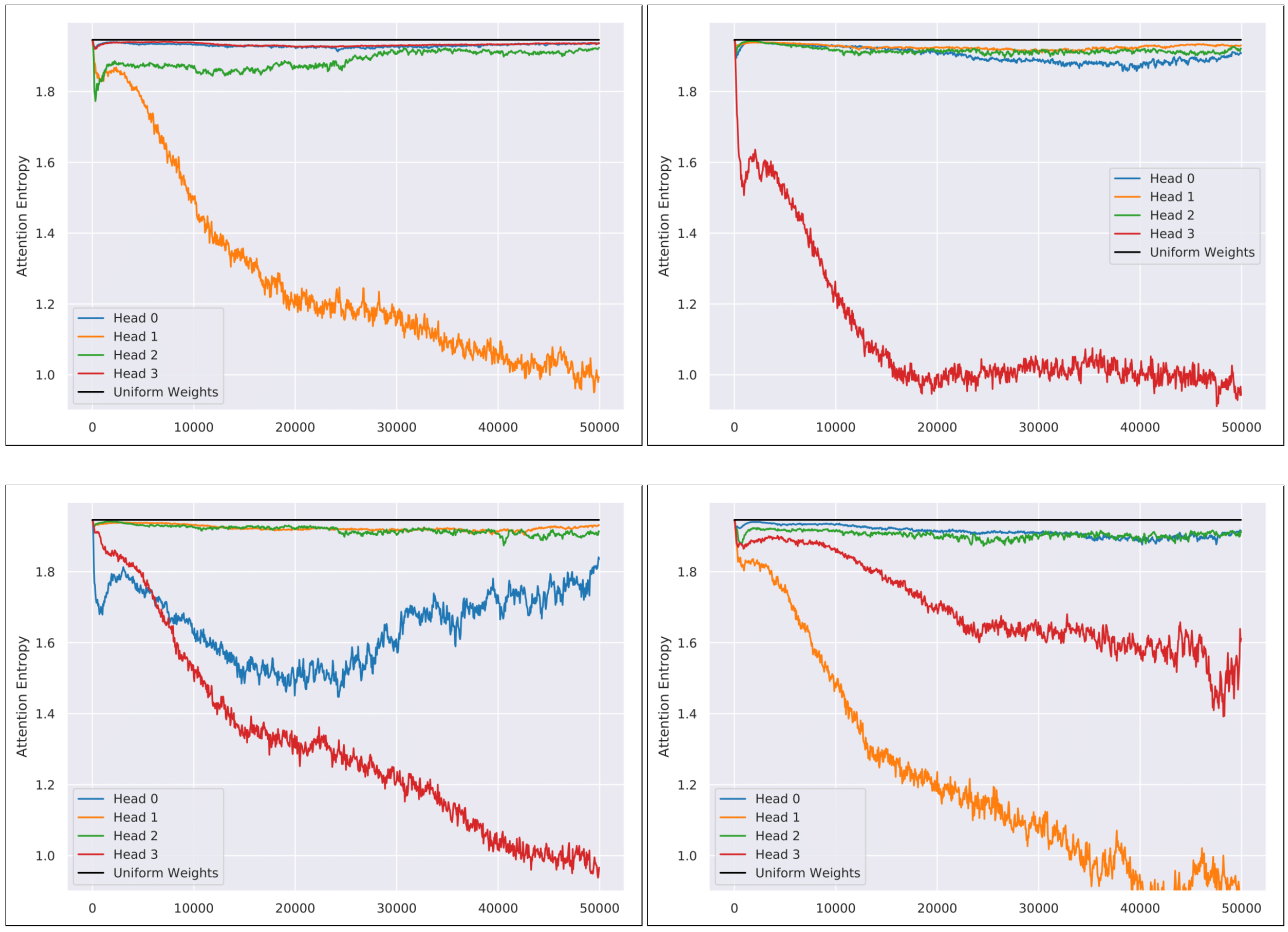

Figure 1. Attention "entropy" for each head over the course of training for the four rovers in the Rover-Tower environment

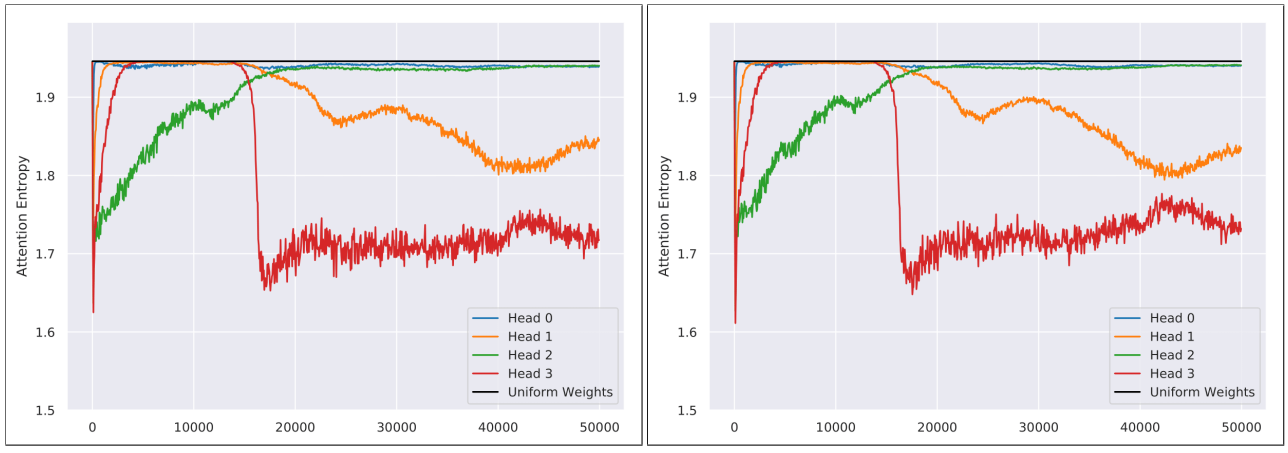

Figure 2. Attention "entropy" for each head over the course of training for two collectors in the Treasure Collection Environment

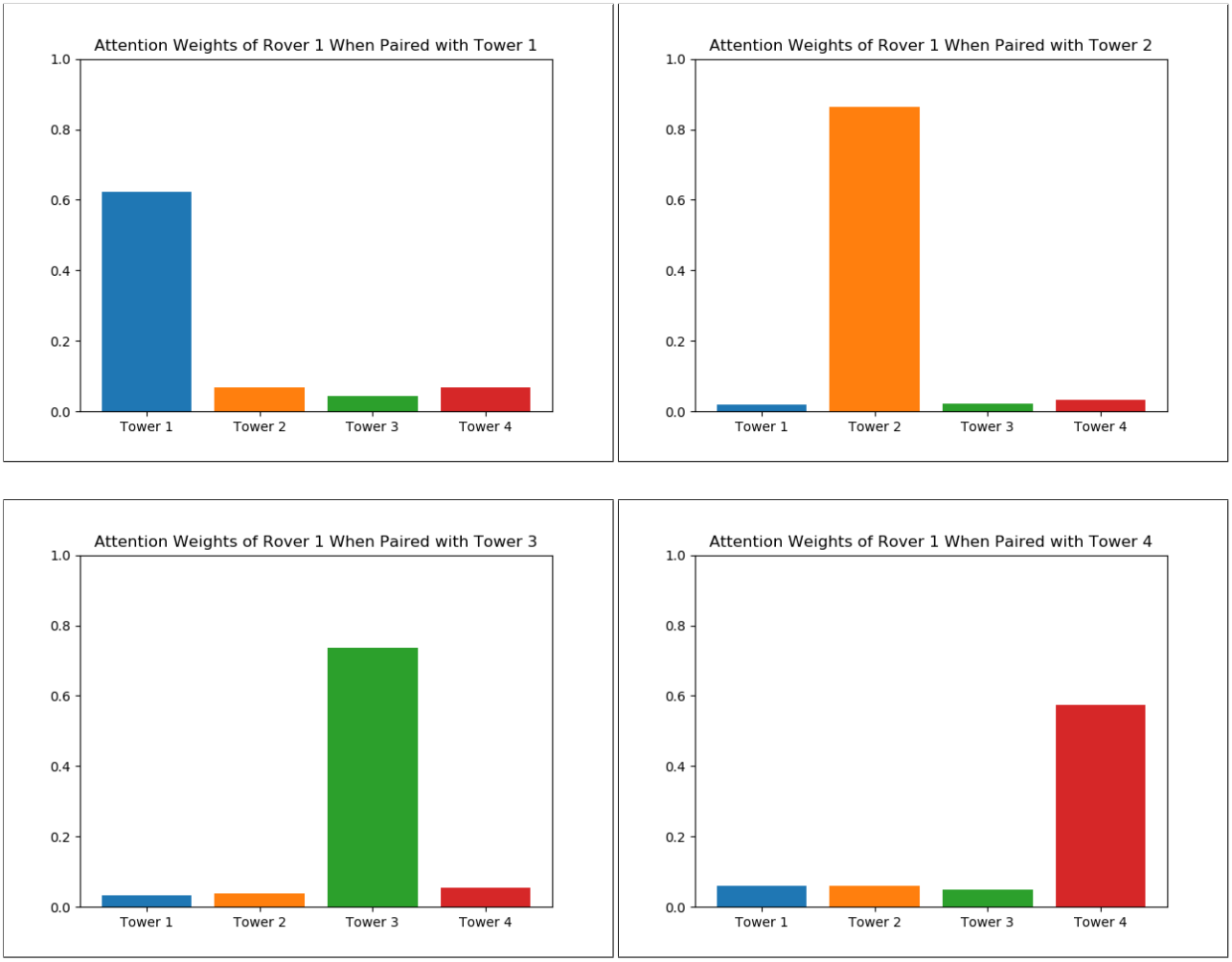

Figure 3. Attention weights when subjected to different Tower pairings for Rover 1 in Rover-Tower environment