Reinforcement Learning From Imperfect Demonstration

REINFORCEMENT LEARNING FROM IMPERFECT DEMONSTRATIONS

Yang Gao[†,*], Huazhe(Harry) Xu[†,*], Ji Lin[‡], Fisher Yu[†], Sergey Levine[†], Trevor Darrell[†]
[†] UC Berkeley, Department of Electrical Engineering and Computer Science
[‡] Tsinghua University, Department of Electrical Engineering
{yg, huazhe_xu, fy, svlevine, trevor}@eecs.berkeley.edu
{lin-j14}@mails.tsinghua.edu.cn

- **Yunqiu Xu**
- Normalized Actor-Critic (NAC):
  - Similar to DQfD, but can use imperfect (noisy) demonstration
  - **Normalize the Q function, reduce the Q values of actions unseen in the demonstration data**
  - Learn an initial policy network from demonstration, then refine it in real environment (same with DQfD here)
  - Good performance on realistic driving games

---

1. Introduction

- Challenges:
  - DRL requires a large amount of interaction with an environment
  - The weakness of learning from demonstration:
    - Does not leverage reward
    - Imitation: the expert demonstration should be noise-free and near optimal
- Our work: NAC
  - Base on maximum entropy RL
  - Use both offline demonstration and online interaction experience

    - **Do not use supervised loss function**
    - Use a normalized formulation of soft Q-learning gradient (or a variant of policy gradient)
    - **Utilize rewards rather than simply imitating behavior by supervised learning**
    - Do not need optimal demonstration $\rightarrow$ **learn robustly even on corrupted demonstration**

2. Related Work

2.1 Learning from demonstration

- DQfD / DDPGfD / DDPG + HER : Need expert demonstration (near optimal)
- Our difference:
    - Do not require explicit mechanism to determine expert data + Learn to maximize reward using arbitrary data

2.2 Maximum Entropy RL and Soft Value Functions

- Our work is based on
    - Harrnoja et al. 1702.08165 - Reinforcement Learning with deep energy-based policies
    - Schulman et al. 1704.06440 - Equivalence between policy gradients and soft q-learning
- Maximum entropy RL

In the conventional reinforcement learning setting (Sutton & Barto, 1998), the goal of an agent is to learn a policy π_{std}, such that it maximizes the future discounted reward:

$$\pi_{std} = argmax_\pi \sum_t \gamma^t \mathbb{E}_{s_t,a_t\sim\pi(s,a)}[R(s_t,a_t)] \tag{1}$$

Maximum entropy policy learning (Haarnoja et al., 2017) uses an entropy augmented reward, so that the optimal policy will not only optimize for discounted future rewards, but also maximize the discounted future entropy of the action distribution:

$$\pi_{max-ent} = argmax_\pi \sum_t \gamma^t \mathbb{E}_{s_t,a_t\sim\pi(s,a)}[R(s_t,a_t) + \alpha H(\pi(\cdot|s_t))] \tag{2}$$

where α is a weighting term to balance the importance of the entropy. Unlike previous attempts of only adding the entropy term at a single time step, maximum entropy policy learning maximizes the discounted future entropy over the whole trajectory. Maximum entropy reinforcement learning has many benefits, such as better exploration in multi-modal problems, and establishing connections between Q learning and the actor-critic method (Haarnoja et al., 2017; Schulman et al., 2017).

- Soft Value Functions

Since the maximum entropy RL paradigm augments the reward with an entropy term, the definition of the value functions naturally changes to

$$Q_\pi(s,a) = r(s_0,a_0) + \mathbb{E}_{(s_1,a_1,\dots)\sim\pi(s)} \sum_{t=1}^{\infty} \gamma^t (R(s_t,a_t) + \alpha H(\pi(\cdot|s_t))) \tag{3}$$

$$V_\pi(s) = \mathbb{E}_{(s_1,a_1,\dots)\sim\pi(s)} \sum_{t=0}^{\infty} \gamma^t (R(s_t,a_t) + \alpha H(\pi(\cdot|s_t))) \tag{4}$$

- Soft Q-learning and policy gradient

With the entropy augmented reward, one can derive the soft versions of Q learning and policy gradient (Haarnoja et al., 2017). The soft Q learning gradient is given by

$$\nabla_\theta Q_\theta(s,a)(Q_\theta(s,a) - \hat{Q}(s,a)) \tag{7}$$

where $\hat{Q}(s,a)$ is a bootstrapped Q-value estimate obtained by $R(s,a) + \gamma V_Q(s')$. Here, $R(s,a)$ is the reward received from the environment, V_Q is computed from $Q_\theta(s,a)$ with Equation 5. We can also derive a policy gradient variant of this method, by parameterizing as the policy as $\log \pi_\theta(a|s) = Q_\theta(s,a) - V_\theta(s)$, which induces a policy gradient of the form

$$\mathbb{E}\left[\sum_{t=0}^{\infty} \nabla_\theta \log \pi_\theta(a_t|s_t)(\hat{Q}_\pi(s_t,a_t) - b(s_t)) + \alpha \nabla_\theta H(\pi_\theta(\cdot|s_t))\right] \tag{8}$$

where $b(s_t)$ is some arbitrary baseline (Schulman et al., 2017).

- Our difference:

- Focus on learning from demonstration
- Combine PG and Q-learning for RLfD without explicit imitation (supervised) loss

3. Robust Learning from Demonstration and Reward

- Why can't us train only on good demonstration?
  - The agent can not understand why the action is good
  - **The agent can assign actions high Q-value, but can not assign other actions low Q-values**
- NAC
  - Combine soft Q-learning and soft policy gradient
  - New Q-function gradient: **give actions not in demonstration low Q-values**

3.1 Details of NAC

We propose a unified learning from demonstration approach, which applies the normalized actor-critic updates to both off policy demonstrations and in-environment transitions. To better facilitate the comparison with the Q learning method, we re-parametrize the normalized actor-critic with Q function. As shown in the appendix A.1, we have the updates for the actor and the critic being:

$$\nabla_\theta J_{PG} = \mathbb{E}_{s,a\sim\pi_Q}\left[(\nabla_\theta Q(s,a) - \nabla_\theta V_Q(s))(Q(s,a) - \hat{Q}(s,a))\right] \tag{9}$$

$$\nabla_\theta J_V = \mathbb{E}_s\left[\nabla_\theta \frac{1}{2}(V_Q(s) - \hat{V}(s))^2\right] \tag{10}$$

where V_Q and π_Q are deterministic functions of Q: $V_Q(s) = \alpha \log \sum_a \exp(Q(s,a)/\alpha)$; $\pi_Q(a|s) = \exp((Q(s,a) - V_Q(s))/\alpha)$. $\hat{Q}(s,a), \hat{V}(s)$ are obtained by:

$$\hat{Q}(s,a) = R(s,a) + \gamma V_Q(s') \tag{11}$$

$$\hat{V}(s) = \mathbb{E}_{a\sim\pi_Q}\left[R(s,a) + \gamma V_Q(s')\right] + \alpha H(\pi_Q(\cdot|s)) \tag{12}$$

- Different versions of NAC
  - Q-learning variant: omits $\nabla_\theta V(s)$ in Eq.9.
  - PG variant: requires importance sampling to use off-policy data
- Algorithm:

We summarize the proposed method in Algorithm 1. Our method uses samples from the demonstrations and the replay buffer, rather than restricting the samples to be on policy as in standard actor-critic methods. Similar to DQN, we utilize a target network to compute $\hat{Q}(s,a)$ and $\hat{V}(s)$, which stabilizes the training process.

Algorithm 1 The Normalized Actor-Critic Algorithm

θ: parameters for the rapid Q network, θ': parameters for the target Q network, $\mathcal{D}$: demonstrations collected by human or a trained policy network, T: target network update frequency, $\mathcal{M}$: replay buffer, k: number of steps to train on the demonstrations
for step t $\in \{1, 2, ...\}$ **do**
  if t $\leq k$ **then**
    Sample a mini-batch of transitions from $\mathcal{D}$
  else
    Start from s, sample a from π, execute a, observe (s',r) and store (s, a, r, s') in $\mathcal{M}$
    Sample a mini-batch of transitions from $\mathcal{M}$
  end if
  Update θ with gradient: $\nabla_\theta J_{PG} + \nabla_\theta J_V$
  if t mod T = 0 **then**
    $\theta' \leftarrow \theta$
  end if
end for

3.2 Why NAF can reduce Q-value those not observed in demonstration

- Traditional Q-learning:
  - Can not know whether the action itself is good
  - Can not know whether all actions in that states are good
  - So actions in demonstration may not necessary to have higher value than other actions in that state

$$\nabla_\theta Q_\theta(s,a)(Q_\theta(s,a) - \hat{Q}(s,a)) \tag{7}$$

$$\nabla_\theta J_{PG} = \mathbb{E}_{s,a\sim\pi_Q}\left[(\nabla_\theta Q(s,a) - \nabla_\theta V_Q(s))(Q(s,a) - \hat{Q}(s,a))\right] \tag{9}$$

- In NAF, actor's update follows Eq.9.
  - Compared with soft Q-learning update in Eq.7, there is an extra term $-\nabla_\theta V_Q(s)$
  - When Q value $Q(s,a)$ increases, $V_Q(s)$ will decrease because of different sign
  - $V_Q(s) = \alpha log \sum_a exp(Q(s,a)/\alpha)) \rightarrow$ **If $V_Q(s)$ is decreasing, $Q(s,a)$ will**

not increase if actions are not observed in the demonstration

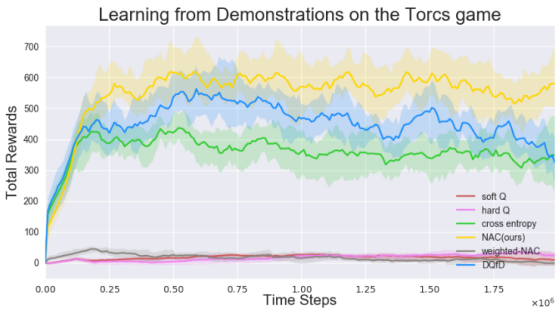
- **感觉这里还是略抽象, 需要进一步理解**

4. Experiment

- Questions:
  - Can NAC benefit from both demonstrations and rewards
  - Can NAC handle imperfect demonstrations
  - Can NAC learn meaningful behaviors with a few demonstrations
- Baselines:
  - DQfD: supervised loss + RL loss
  - Behavioral cloning: supervised imitation
  - DQN: similar to DQfD, DQN is pretrained with demonstration, then fine-tune to environment
  - Soft DQN: DQN + entropy regularized reward
  - NAC with importance sampling weighting:
    - **Why use importance weighting term: correct the action distribution mismatch between the demonstration and current policy**
- Environments: Torcs and GTA

Figure 1: Sample frames from Torcs (left) and GTA (right).

- Results



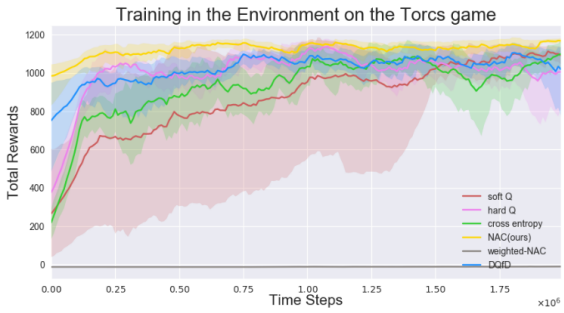
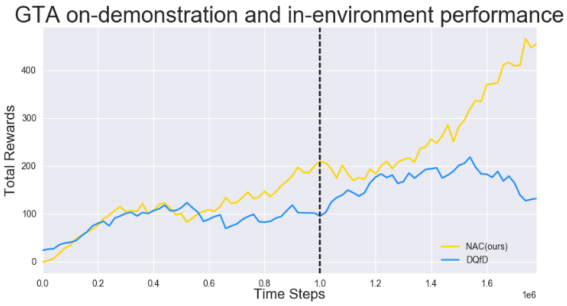
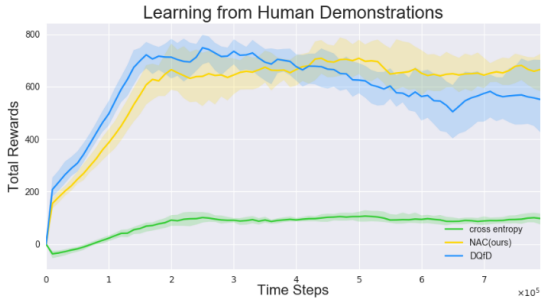


Figure 2: Performances on the Torcs game. The x-axis shows the training iterations. The y-axis shows the average total rewards. Solid lines are average values over 10 random seeds. Shaded regions correspond to one standard deviation. The left figure shows the performance for each agent when they only learn from demonstrations, while the right one shows the performance for each agent when they interact with the environments after learning from demonstrations. Our method consistently outperforms other methods in both cases.

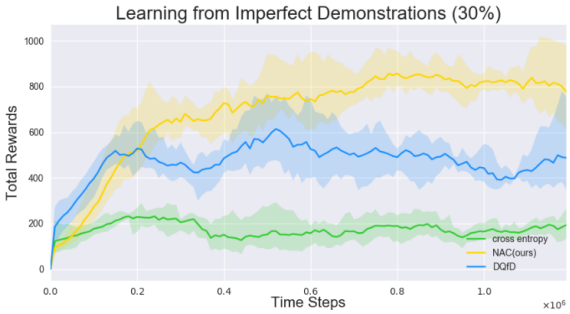


(a) The on-demonstration and in-environment performance of the NAC and DQfD methods on GTA. The vertical line separates the learning from demonstration phase and finetuning in environment phase. Our method consistently outperforms DQfD in both phases.



(b) Performances on the Torcs game with human demonstrations. DQfD performs well in the beginning, but overfits in the end. The behavior cloning method is much worse than NAC and DQfD. Our NAC method performs best at convergence.

Figure 3: Performance on GTA (left) and performance on Torcs with human demonstrations (right)



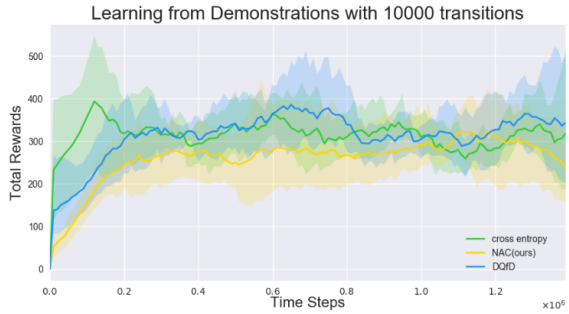


Figure 4: Left: Learning from imperfect data when the imperfectness is 30%. Our NAC method does not clone suboptimal behaviors and thus outperforms DQfD and behavior cloning. Right: Learning from a limit amount of demonstrations. Even with only 30 minutes (10k transitions) of experience, our method could still learn a policy that is comparable with supervise learning method.

5. Summary

- 和之前DQfD等工作类似都是通过demonstration加速强化学习
- 不同的是DQfD要求demonstration必须是很好的, 而NAC允许imperfect demonstration, 在demonstration不那么好时, NAC性能更好 (见Fig.4.)
- NAC是如何达成这一目的的?
  - DQfD等方法的不足在于, 只能判定看到的动作还不错, 但对于没看到的动作没法判定它们不好
  - 结合了soft Q-learning以及soft policy gradient
  - 对于未在demonstration中出现的动作, 降低他们的Q-value
- 还需再理解下原理, 有机会的话尝试实现下, 毕竟我们创建的demonstration不一定是完美的