Recent Advances in Generative Adversarial Imitation Learning

2020/10/27

朱佳成

Outline

- Generative adversarial imitation learning (GAIL)
- Multi-modal imitation learning
- Imitation learning from incomplete demonstration

Cons of other imitation learning methods

- **Behavioral Cloning:**

  learns a policy as a supervised learning problem over state-action pairs from expert trajectories. (mapping from states to actions)

$$\min_{\theta} \sum_{(s,a)\sim\tau_E} -\log \pi_\theta(a \mid s)$$

- **Inverse Reinforcement Learning:**

  learns a cost function that prioritizes entire trajectories over others.

$$IRL_\psi(\pi_E) = \arg\max_{c\in\mathbb{R}^{S\times A}} -\psi(c) + \left(\min_{\pi\in\Pi} -H(\pi) + \mathbb{E}_\pi[c(s,a)]\right) - \mathbb{E}_{\pi_E}[c(s,a)]$$

Generative adversarial imitation learning (GAIL) [Ho & Ermon, NIPS 2016]

- The GAIL objective:

$$\min_{\theta} \max_{\omega} \mathbb{E}_{\pi_\theta}[\log(D_\omega(s,a))] + \mathbb{E}_{\pi_E}[\log(1-D_\omega(s,a))] - \lambda H(\pi_\theta)$$

Algorithm 1 Generative adversarial imitation learning

1: **Input:** Expert trajectories $\tau_E \sim \pi_E$, initial policy and discriminator parameters θ_0, w_0
2: **for** $i = 0, 1, 2, \ldots$ **do**
3: Sample trajectories $\tau_i \sim \pi_{\theta_i}$
4: Update the discriminator parameters from w_i to w_{i+1} with the gradient

$$\hat{\mathbb{E}}_{\tau_i}[\nabla_w \log(D_w(s,a))] + \hat{\mathbb{E}}_{\tau_E}[\nabla_w \log(1 - D_w(s,a))] \quad (17)$$

5: Take a policy step from θ_i to θ_{i+1}, using the TRPO rule with cost function $\log(D_{w_{i+1}}(s,a))$. Specifically, take a KL-constrained natural gradient step with

$$\begin{aligned} &\hat{\mathbb{E}}_{\tau_i}[\nabla_\theta \log \pi_\theta(a|s) Q(s,a)] - \lambda \nabla_\theta H(\pi_\theta), \\ &\text{where } Q(\bar{s},\bar{a}) = \hat{\mathbb{E}}_{\tau_i}[\log(D_{w_{i+1}}(s,a)) \mid s_0 = \bar{s}, a_0 = \bar{a}] \end{aligned} \quad (18)$$

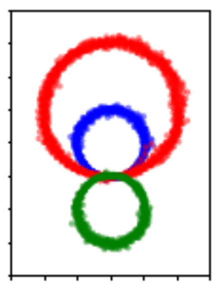
6: **end for**

Outline

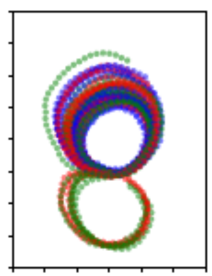
- Generative adversarial imitation learning (GAIL)
- Multi-modal imitation learning
  - InfoGAIL & Triple-GAIL
- Imitation learning from incomplete demonstration

Multi-modal imitation learning

- GAIL can not learns a good policy from multi-modal demonstrations as it assumes all the demonstrations come from a single expert, and can not disentangle the demonstrations.



expert



GAIL

Figure 1. Learned trajectories in the synthetic 2D plane environment

InfoGAIL: Interpretable imitation learning from visual demonstrations [Li, Song & Ermon, NIPS 2017]

- Main challengings:
  1. Discover the latent factors of expert demonstrations
  2. Learned policies that produce trajectories correspond to latent factor

Solution

Introduce a latent variable c into policy function: $\pi(a \mid s, c)$

Guide the generating process.

Discover the salient semantic features of the data distribution.

Maximize the mutual information between c & trajectories

$$I(c:\tau)$$

$$L_I(\pi, Q) = \mathbb{E}_{c\sim p(c), a\sim\pi(\cdot|s,c)}[\log Q(c|\tau)] + H(c)$$
$$\leq I(c;\tau)$$

Lower bound

Approximation of true posterior $P(c \mid \tau)$

InfoGAIL: Interpretable imitation learning from visual demonstrations [Li, Song & Ermon, NIPS 2017]

- The InfoGAIL objective: $\min_{\pi,Q} \max_{D} \mathbb{E}_\pi[\log D(s,a)] + \mathbb{E}_{\pi_E}[\log(1-D(s,a))] - \lambda_1 L_I(\pi, Q) - \lambda_2 H(\pi)$

Algorithm 1 InfoGAIL

Input: Initial parameters of policy, discriminator and posterior approximation $\theta_0, \omega_0, \psi_0$; expert trajectories $\tau_E \sim \pi_E$ containing state-action pairs.
Output: Learned policy π_θ
for $i = 0, 1, 2, ...$ **do**
  Sample a batch of latent codes: $c_i \sim p(c)$
  Sample trajectories: $\tau_i \sim \pi_{\theta_i}(c_i)$, with the latent code fixed during each rollout.
  Sample state-action pairs $\chi_i \sim \tau_i$ and $\chi_E \sim \tau_E$ with same batch size.
  Update ω_i to ω_{i+1} by ascending with gradients

$$\Delta_{\omega_i} = \hat{\mathbb{E}}_{\chi_i}[\nabla_{\omega_i} \log D_{\omega_i}(s,a)] + \hat{\mathbb{E}}_{\chi_E}[\nabla_{\omega_i} \log(1 - D_{\omega_i}(s,a))]$$

  Update ψ_i to ψ_{i+1} by descending with gradients

$$\Delta_{\psi_i} = -\lambda_1 \hat{\mathbb{E}}_{\chi_i}[\nabla_{\psi_i} \log Q_{\psi_i}(c|s,a)]$$

  Take a policy step from θ_i to θ_{i+1}, using the TRPO update rule with the following objective:

$$\hat{\mathbb{E}}_{\chi_i}[\log D_{\omega_{i+1}}(s,a)] - \lambda_1 L_I(\pi_{\theta_i}, Q_{\psi_{i+1}}) - \lambda_2 H(\pi_{\theta_i})$$

end for

Improved InfoGAIL

$$\min_{\theta,\psi} \max_{\omega} \mathbb{E}_{\pi_\theta}[D_\omega(s,a)] - \mathbb{E}_{\pi_E}[D_\omega(s,a)] - \lambda_0 \eta(\pi_\theta) - \lambda_1 L_I(\pi_\theta, Q_\psi) - \lambda_2 H(\pi_\theta)$$

Algorithm 2 InfoGAIL with extensions

Input: Expert trajectories $\tau_E \sim \pi_E$; initial policy, discriminator and posterior parameters $\theta_0, \omega_0, \psi_0$; replay buffer $B = \varnothing$;
Output: Learned policy π_θ
for $i = 0, 1, 2, ...$ **do**
Sample a batch of latent codes: $c_i \sim P(c)$
Sample trajectories: $\tau_i \sim \pi_{\theta_i}(c_i)$, with the latent code fixed during each rollout.
Update the replay buffer: $B \leftarrow B \cup \tau_i$.
Sample $\chi_i \sim B$ and $\chi_E \sim \tau_E$ with same batch size.
Update ω_i to ω_{i+1} by ascending with gradients

$$\Delta_{\omega_i} = \hat{\mathbb{E}}_{\chi_i}[\nabla_{\omega_i} D_{\omega_i}(s,a)] - \hat{\mathbb{E}}_{\chi_E}[\nabla_{\omega_i} D_{\omega_i}(s,a)]$$

Clip the weights of ω_{i+1} to $[-0.01, 0.01]$.
Update ψ_i to ψ_{i+1} by descending with gradients

$$\Delta_{\psi_i} = -\lambda_1 \hat{\mathbb{E}}_{\chi_i}[\nabla_{\psi_i} \log Q_{\psi_i}(c|s,a)]$$

Take a policy step from θ_i to θ_{i+1}, using the TRPO update rule with the following objective (without reward augmentation):

$$\hat{\mathbb{E}}_{\chi_i}[D_{\omega_{i+1}}(s,a)] - \lambda_1 L_I(\pi_{\theta_i}, Q_{\psi_{i+1}}) - \lambda_2 H(\pi_{\theta_i})$$

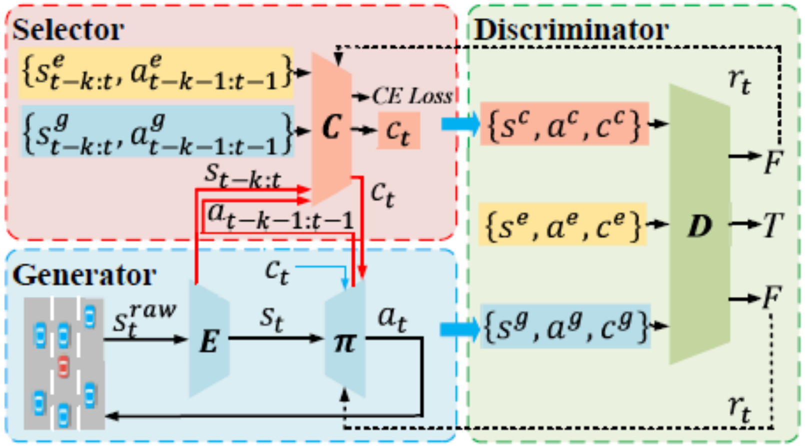
or (with reward augmentation):

$$\hat{\mathbb{E}}_{\chi_i}[D_{\omega_{i+1}}(s,a)] - \lambda_0 \eta(\pi_{\theta_i}) - \lambda_1 L_I(\pi_{\theta_i}, Q_{\psi_{i+1}}) - \lambda_2 H(\pi_{\theta_i})$$

end for

Triple-GAIL: A multi-modal imitation learning framework with generative adversarial nets
[Fei et al., IJCAI 2020]

$$\min_{\alpha,\theta} \max_{\psi} \mathbb{E}_{\pi_E}[\log(1-D_\psi(s,a,c))] + \omega \mathbb{E}_{\pi_\theta}[\log D_\psi(s,a,c)] + (1-\omega)\mathbb{E}_{C_\alpha}[\log D_\psi(s,a,c)] + \lambda_E R_E + \lambda_G R_G - \lambda_H H(\pi_\theta)$$



$$R_E = \mathbb{E}_{\pi_E}[-\log p_{C_\alpha}(c|s,a)] \approx -\frac{1}{N}\sum_{i=0}^{N}\frac{1}{T}\sum_{t=1}^{T} c_{i,t}^{e} \log p_{C_\alpha}\left(c_{i,t}^{c}|s_{i,t}^{e}, a_{i,t-1}^{e}\right)$$

$$R_G = \mathbb{E}_{\pi_\theta}[-\log p_{C_\alpha}(c|s,a)] \approx -\frac{1}{N}\sum_{i=0}^{N}\frac{1}{T}\sum_{t=1}^{T} c_{i,t}^{g} \log p_{C_\alpha}\left(c_{i,t}^{c}|s_{i,t}^{g}, a_{i,t-1}^{g}\right)$$

Triple-GAIL

Algorithm 1 The Training Procedure of Triple-GAIL

Input: The multi-intention trajectories of expert τ_E; **Parameter**: The initial parameters θ_0, α_0 and ψ_0

for $i = 0, 1, 2, \cdots$ **do**
  for $j = 0, 1, 2, \cdots, N$ **do**
    Reset environments by the demonstration episodes with fixed label c_j;
    Run policy $\pi_\theta(\cdot|c_j)$ to sample trajectories: $\tau_{c_j} = (s_0, a_0, s_1, a_1, ...s_{T_j}, a_{T_j}|c_j)$
  end for
  Update the parameters of π_θ via TRPO with rewards: $r_{t_j} = -\log D_\psi(s_{t_j}, a_{t_j}, c_j)$
  Update the parameters of D_ψ by gradient ascending with respect to:

$$\nabla_\psi \frac{1}{N_e}\sum_{n=1}^{N_e}\log(1-D_\psi(s_n^e, a_n^e, c_n^e)) + \frac{1}{N}\sum_{j=1}^{N}\left[\frac{\omega}{T_j}\sum_{t=1}^{T_j}\log D_\psi(s_t^g, a_t^g, c_j^g) + \frac{1-\omega}{T_j}\sum_{t=1}^{T_j}\log D_\psi(s_t^c, a_t^c, c_j^c)\right] \tag{9}$$

  Update the parameters of C_α by gradient descending with respect to:

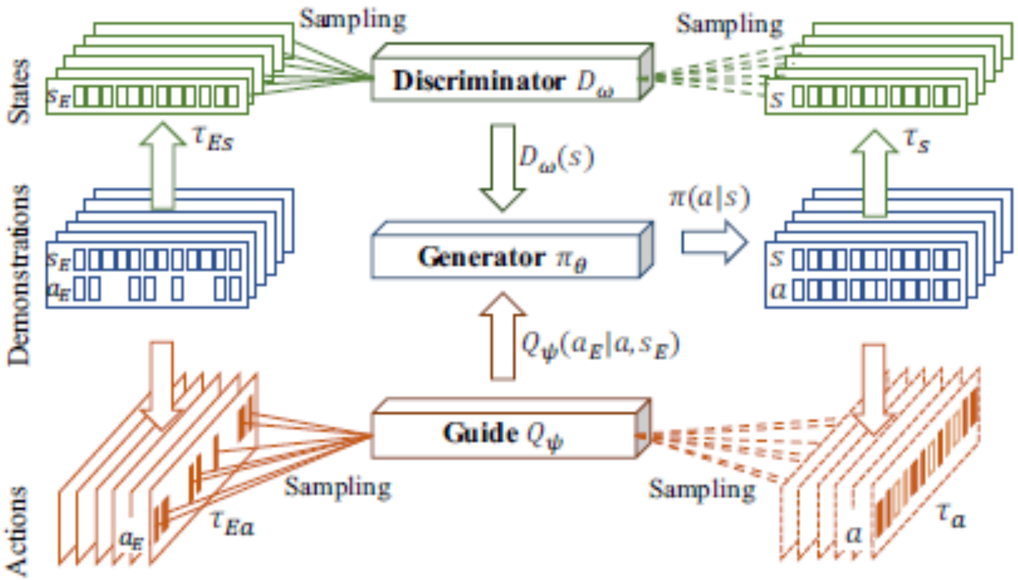
$$\nabla_\alpha \frac{1}{N}\sum_{j=1}^{N}\left[\frac{1-\omega}{T_j}\sum_{t=1}^{T_j}\log D_\psi(s_t^c, a_t^c, c_j^c) - \frac{\lambda_E}{T_j}\sum_{t=1}^{T_j} c_j^e \log p_{C_\alpha}(c_t^c|s_t^e, a_{t-1}^e) - \frac{\lambda_G}{T_j}\sum_{t=1}^{T_j} c_j^e \log p_{C_\alpha}(c_t^c|s_t^g, a_{t-1}^g)\right] \tag{10}$$

end for

Outline

- Generative adversarial imitation learning (GAIL)
- Multi-modal imitation learning
- Imitation learning from incomplete demonstration
  - Adversarial imitation learning from incomplete demonstrations

Adversarial imitation learning from incomplete demonstrations [Sun & Ma, IJCAI 2019]



$$\max_D \mathbb{E}_{s\sim\pi} \log D(s) + \mathbb{E}_{s\sim\pi_E} \log(1 - D(s))$$

$$L_I(\pi, Q) = \mathbb{E}_{a_E\sim\{\tau_{Ea}\}}\left[\log Q(a_E|a, s_E)\right] + H(a_E)$$
$$\leq I\left(a_E; a \sim \pi(s_E)\right)$$

Adversarial imitation learning from incomplete demonstrations [Sun & Ma, IJCAI 2019]

- Objective:

$$\min_{\pi\in\Pi}\left[-\lambda_1 H(\pi_\theta) - \lambda_2 L_I(\pi_\theta, Q_\psi) + \max_D \mathbb{E}_{s\sim\pi_\theta}\log D_\omega + \mathbb{E}_{s\sim\pi_E}\log(1-D_\omega)\right]$$

Algorithm 1 Action-guided adversarial imitation learning

Input: expert trajectories $\tau_E = \{(\tau^i_{Es}, \tau^i_{Ea})\} \sim \pi_E$
Parameter: Policy, discriminator and posterior parameters $\theta_0, \omega_0, \psi_0$; hyperparameters α and β
Output: Learned policy π_θ

for $i = 0, 1, 2, \ldots$ **do**
  Sample trajectories: $\tau^i \sim \pi_{\theta_i}$ during each rollout.
  Sample states $s^i \sim \tau^i_s$, $s^i_E \sim \{\tau^i_{Es}\}$ by same batch size.
  Update ω_i to ω_{i+1} for D_ω based on Equation 4.
  Query $\{a^i_E\}$ and run π_{θ_i} on $\{s^i_E\}$ to collect $\{a^i\}$.
  Update ψ_i to ψ_{i+1} for Q_ψ based on Equation 5.
  Update θ_i to θ_{i+1} via TRPO for Equation 6 with rewards $r(s,a) = \alpha D_{\omega_{i+1}}(s) + \beta Q_{\psi_{i+1}}(a_E|s,a) \quad a_E \sim \tau_{Ea}$
end for