Q-Prop:
Toward Sample-Efficient & Stable Deep RL

Shixiang (Shane) Gu,
Timothy Lillicrap, Zoubin Ghahramani, Richard E. Turner, Sergey Levine
Reinforcement Learning

\[ s_{t+1} \sim p(s_{t+1} | s_t, a_t) \]

Goal: learn optimal policy that maximizes cumulative rewards

\[
\max \mathbb{E}_\pi \left[ \sum_{t=0}^{T} \gamma^t r(s_t, a_t) \right]
\]

Potential applications
- Simulation
- Real-world
- Computation bound
- Data bound
Deep RL in Robotics

Task Generality

Model-based

Model-based + off-/on-policy

Off-policy + on-policy

Data Intensity

100 episodes, 20 min

1000 episodes, 3 hours

>10000 episodes, >1.5 days

Off-policy

On-policy

Deep RL Algorithms

- TRPO [Schulman et. al., 2015]
- A3C [Mnih et. al., 2016]
- ES [Koutnik et. al., 2013, Salimans et. al., 2017]
- Q-Prop [Gu et. al., 2017]
- ACER [Wang et. al., 2017]
- PGQ [O’Donoghue et. al., 2017]
- Dyna-Q [Sutton et. al., 1990]
- AMA-NFQ [Lampe et. al., 2014]
- NAF-IR [Gu et. al., 2016]
- PILQR-GPS [Chebotar et. al., 2017]
- NFQCA [Hafner & Riedmiller, 2011]
- DDPG [Silver et. al., 2014, Lillicrap et. al., 2016]
- NAF [Gu et. al., 2016, 2017]
- GPS [Levine et. al., 2015]
- E2C [Watter et. al., 2015]
- P2T [Wahlstrom et. al., 2015]
On-policy MC policy gradient

\[ \nabla_{\theta} J(\theta) = E_\pi [\nabla_{\theta} \log \pi_{\theta}(a_t|s_t) \hat{Q}(s_t, a_t)] \]

\[ \hat{Q}(s_t, a_t) = \sum_{\tau>t} r(s_\tau, a_\tau) \]

\[ \pi_{\theta}(a|s) = \mathcal{N}(\mu_{\theta}(s_t), \Sigma_{\theta}(s_t)) \]

\[ \pi : \text{on-policy} \]

- Unbiased gradient
- Stable

- High-variance gradient
- Forgets experience
- Sample-intensive
Off-policy RL

**Off-policy Actor-Critic**

\[
\min_w \mathbb{E}_{\beta}[\mathbb{E}_{\beta}[(r(s_t, a_t) + \gamma Q(s_{t+1}, \mu(s_{t+1})) - Q_w(s_t, a_t))^2]
\]

\[
\max_\theta \mathbb{E}_{\beta}[Q_w(s_t, \mu(\theta)(s_t))]
\]

\[
\nabla_\theta J(\theta) \approx \mathbb{E}_\beta[\nabla_a Q_w(s_t, a)|_{a=\mu(\theta)(s_t)} \nabla_\theta \mu(\theta)(s_t)]
\]

\[\beta : \text{off-policy}\]

+ Low-variance gradient
+ Reuse experience
+ Sample-efficient (relatively)

- Biased gradient
- Less Stable

Efficiency
On-policy + Off-policy RL

Q-Prop

\[ \nabla_\theta J(\theta) \approx \mathbb{E}_\pi [\nabla_a Q_w(s_t, a)|_{a=\mu_\theta(s_t)} \nabla_\theta \mu_\theta(s_t)] \]
On-policy + Off-policy RL

### Q-Prop

$$\nabla \theta J(\theta) = \mathbb{E}_\pi \left[ \nabla_a Q_w(s_t, a) | a = \mu_\theta(s_t) \nabla \theta \mu_\theta(s_t) \right]$$

$$+ \mathbb{E}_\pi \left[ \nabla \log \pi_\theta(a_t|s_t) (\hat{Q}(s_t, a_t) - Q_w(s_t, a_t)) \right]$$

$$\bar{Q}_w(s_t, a_t) : \text{first-order Taylor exp. of } Q_w \text{ at } a_t = \mu_\theta(s_t)$$

$$\nabla \theta J(\theta) = \mathbb{E}_\pi \left[ \nabla_a Q(s_t, a) | a = \mu_\theta(s_t) \nabla \theta \mu_\theta(s_t) \right]$$

$$+ \mathbb{E}_\pi \left[ \nabla \log \pi_\theta(a_t|s_t) (\hat{Q}(s_t, a_t) - Q_w(s_t, a_t)) \right] - \nabla \mu \bar{Q}_w(s_t, a_t) | a_t = \mu_\theta(s_t) (a_t - \mu_\theta(s_t))$$

- **Critic** can be fitted off-policy.
- **Policy** is fitted on-policy.
- + unbiased policy gradient
- + low-variance grad from critic
- + sample-efficiency & stability
- + modular
- - more computation
- - higher variance if critic is bad
Analysis

When does Q-Prop help? - When variance is reduced.

Q-Prop is a **control variate** [Ross, 2002]

- use a correlated variable with known expected value to reduce variance of an estimator

\[
\bar{f} = \mathbb{E}[f(x)] = \mathbb{E}[f(x) - \eta g(x) + \eta \bar{g}] = \mathbb{E}[\bar{f}(x)]
\]

\[
\text{Var}(\bar{f}) = \text{Var}(f) + \eta^2 \text{Var}(g) - 2\eta \text{Cov}(f, g)
\]

\[
\eta^* = \frac{\text{Cov}(f, g)}{\text{Var}(g)} \quad \Rightarrow \quad \text{Var}(\bar{f}) = (1 - \rho(f, g)^2)\text{Var}(f)
\]

+ Large variance reduction if \( f \) & \( g \) strongly correlated

+ Guaranteed variance reduction if \( f \) & \( g \) are correlated
Analysis

Adaptive Q-Prop

\[ \nabla_\theta J(\theta) = E_\pi [\nabla_\theta \log \pi_\theta (a_t | s_t) (\hat{Q} - \eta(s_t) \bar{Q}_w)] \\
+ E_\pi [\eta(s_t) \nabla_a \bar{Q}_w | a = \mu_\theta(s_t) \nabla_\theta \mu_\theta(s_t)] \]

“Optimal” adaptation

\[ \eta^*(s_t) = \frac{\text{Cov}(\hat{Q}, \bar{Q}_w)}{\text{Var}(\bar{Q}_w)} \rightarrow \text{Var}(\hat{Q} - \eta^* \bar{Q}_w) = (1 - \rho(\hat{Q}, \bar{Q}_w)^2) \text{Var}(\hat{Q}) \]

+ guaranteed reduction on learning signal variance

Conservative Q-Prop

\[ \eta(s_t) = \begin{cases} \\ 1, & \text{if Cov}(\hat{Q}, \bar{Q}_w) > 0 \\ 0, & \text{otherwise} \end{cases} \]
Q-Prop Diagram

$B$ : use on-policy batch samples

$R$ : use off-policy samples from replay

- Collect samples by rolling out policy
- Compute Monte Carlo critic
- Add samples to replay
- Update off-policy critic
- Compute baseline critic
- Compute Q-Prop mask
- Update policy with Q-Prop gradient

$\pi_\theta$ $B$

$\hat{Q}$

$R$

$Q_w$

$\bar{Q}_w$

$\eta$

$\pi_\theta$

Q-Prop steps

on-policy: state baseline, GAE
[TRPO-GAE, Schulman et. al., 2016]

off-policy policy evaluation:
replay, target network
[DDPG, Lillicrap et. al., 2016]

+allow any on-policy policy gradient & any policy evaluation

on-policy: trust-region
[TRPO-GAE, Schulman et. al., 2016]
Q-Prop + TRPO-GAE vs. TRPO-GAE vs. DDPG

[Gu et. al., 2017]

SOA on-policy
[Schulman et. al., 2016]

SOA off-policy
[Lillicrap et. al., 2016]

- more sample-efficient than TRPO-GAE
- more stable than DDPG
- requires smaller batch size than TRPO-GAE
# Experiments

## Results

Results appear consistent across multiple domains: MuJoCo, OpenAI Gym

<table>
<thead>
<tr>
<th>Domain</th>
<th>Threshold</th>
<th>TR-c-Q-Prop</th>
<th>TRPO</th>
<th>DDPG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>MaxReturn.</td>
<td>Episodes</td>
<td>MaxReturn</td>
</tr>
<tr>
<td>Ant</td>
<td>3500</td>
<td>3534</td>
<td>4975</td>
<td>4239</td>
</tr>
<tr>
<td>HalfCheetah</td>
<td>4700</td>
<td>4811</td>
<td>20785</td>
<td>4734</td>
</tr>
<tr>
<td>Hopper</td>
<td>2000</td>
<td>2957</td>
<td>5945</td>
<td>2486</td>
</tr>
<tr>
<td>Humanoid</td>
<td>2500</td>
<td>&gt;3492</td>
<td>14750</td>
<td>918</td>
</tr>
<tr>
<td>Reacher</td>
<td>-7</td>
<td>-6.0</td>
<td>2060</td>
<td>-6.7</td>
</tr>
<tr>
<td>Swimmer</td>
<td>90</td>
<td>103</td>
<td>2045</td>
<td>110</td>
</tr>
<tr>
<td>Walker</td>
<td>3000</td>
<td>4030</td>
<td>3685</td>
<td>3567</td>
</tr>
</tbody>
</table>

+ results appear consistent across multiple domains
Relation to other work

Q-Prop

\[ \nabla_\theta J(\theta) = \mathbb{E}_\pi [\nabla_\theta \log \pi_\theta(a_t|s_t)(\hat{Q} - Q_w)] + \mathbb{E}_\pi [\nabla_a Q_w|_{a=\mu_\theta(s_t)} \nabla_\theta \mu_\theta(s_t)] \]

Directly mixing on-policy and off-policy

\[ \nabla_\theta J(\theta) \approx \nu \mathbb{E}_\pi [\nabla_\theta \log \pi_\theta(a_t|s_t)\hat{Q}] + (1 - \nu) \mathbb{E}_\beta [\nabla_a Q_w|_{a=\mu_\theta(s_t)} \nabla_\theta \mu_\theta(s_t)] \]

Mixing on-policy and off-policy deep RL
- ACER [Wang et. al., 2017], PGQ [O’Donoghue et. al., 2017]
Take-away Messages

Q-Prop: take off-policy algorithm and correct it with on-policy algorithm on residuals

For RL:
- Toward sample-efficient & stable algorithm
- Toward off-policy policy gradient

For ML:
- An efficient, biased algorithm with a correct algorithm on the residuals
  - Stochastic discrete networks
    - MuProp [Gu et. al., 2016], REBAR [Tucker et. al., 2017]
  - Model-based RL?
    - PILQR [Chebotar et. al., 2017]
- Synthetic gradients?
- GANs?
Thank you!

Acknowledgements:

openai/rllab
OpenAI Gym