girafe
ai

Lecture 07: Advantage Actor Critic

Radoslav Neychev

References

These slides are almost the exact copy of Practical RL course week 6 slides.
Special thanks to YSDA team for making them publicly available.

Original slides link: week06_policy_based

REINFORCE algorithm

- Initialize NN weights $\theta_0 \leftarrow random$

- Loop:
  - Sample N sessions **z** under current $\pi_\theta(a|s)$
  - Evaluate policy gradient

$$\nabla J \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{s,a \in z_i} \nabla \log \pi_\theta(a|s) \cdot Q(s,a)$$

  - Ascend $\theta_{i+1} \leftarrow \theta_i + \alpha \cdot \nabla J$

REINFORCE algorithm

- Initialize NN weights $\theta_0 \leftarrow random$

Q: is it off- or on-policy?

- Loop:
  - Sample N sessions **z** under current $\pi_\theta(a|s)$
  - Evaluate policy gradient

$$\nabla J \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{s,a \in z_i} \nabla \log \pi_\theta(a|s) \cdot Q(s,a)$$

  - Ascend $\theta_{i+1} \leftarrow \theta_i + \alpha \cdot \nabla J$

REINFORCE algorithm

- Initialize NN weights $\theta_0 \leftarrow random$

- Loop:

  actions under current policy = on-policy

  - Sample N sessions **z** under current $\pi_\theta(a|s)$
  - Evaluate policy gradient

$$\nabla J \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{s,a \in z_i} \nabla \log \pi_\theta(a|s) \cdot Q(s,a)$$

  - Ascend $\theta_{i+1} \leftarrow \theta_i + \alpha \cdot \nabla J$

value-based Vs policy-based

Value-based

- Q-learning, SARSA, MCTS value-iteration

- Solves harder problem
- Artificial exploration
- Learns from partial experience (temporal difference)
- Evaluates strategy for free :)

Policy-based

- REINFORCE, CEM

- Solves easier problem
- Innate exploration
- Innate stochasticity
- Support continuous action space
- Learns from full session only?

value-based Vs policy-based

Value-based

- Q-learning, SARSA, MCTS value-iteration

- Solves harder problem
- Artificial exploration
- Learns from partial experience (temporal difference)
- Evaluates strategy for free :)

Policy-based

- REINFORCE, CEM

We'll learn much more soon!

- Solves easier problem
- Innate exploration
- Innate stochasticity
- Support continuous action space
- ~~Learns from full session only~~

REINFORCE baselines

- Initialize NN weights $\theta_0 \leftarrow random$

- Loop:
  - Sample N sessions **z** under current $\pi_\theta(a|s)$
  - Evaluate policy gradient

$$\nabla J \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{s,a \in z_i} \nabla \log \pi_\theta(a|s) \cdot Q(s,a)$$

What is better for learning:
random action in good state
or
great action in bad state?

REINFORCE baselines

We can subtract arbitrary baseline b(s)

$$\nabla J = \mathop{E}_{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}} \nabla \log \pi_\theta(a|s)\left(Q(s,a) - b(s)\right) = \ldots$$

$$\ldots = \mathop{E}_{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}} \nabla \log \pi_\theta(a|s) Q(s,a) - \mathop{E}_{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}} \nabla \log \pi_\theta(a|s) b(s) = \ldots$$

REINFORCE baselines

We can subtract arbitrary baseline b(s)

$$\nabla J = \mathop{E}_{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}} \nabla \log \pi_\theta(a|s)\left(Q(s,a) - b(s)\right) = \ldots$$

$$\ldots = \mathop{E}_{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}} \nabla \log \pi_\theta(a|s) Q(s,a) - \mathop{E}_{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}} \nabla \log \pi_\theta(a|s) b(s) = \ldots$$

Q: Can you simplify the second term?

Note that b(s) does not depend on a

REINFORCE baselines

We can subtract arbitrary baseline b(s)

$$\nabla J = \mathop{E}_{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}} \nabla \log \pi_\theta(a|s)\big(Q(s,a) - b(s)\big) = \ldots$$

$$\ldots = \mathop{E}_{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}} \nabla \log \pi_\theta(a|s) Q(s,a) - \mathop{E}_{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}} \nabla \log \pi_\theta(a|s) b(s) = \ldots$$

$$\mathop{E}_{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}} \nabla \log \pi_\theta(a|s) b(s) = b(s) \cdot \mathop{E}_{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}} \nabla \log \pi_\theta(a|s) = 0$$

REINFORCE baselines

We can subtract arbitrary baseline b(s)

$$\nabla J = \mathop{E}_{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}} \nabla \log \pi_\theta(a|s)\left(Q(s,a) - b(s)\right) = \ldots$$

$$\ldots = \mathop{E}_{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}} \nabla \log \pi_\theta(a|s) Q(s,a) - \mathop{E}_{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}} \nabla \log \pi_\theta(a|s) b(s) = \ldots$$

Gradient direction doesn't change!

$$\ldots = \mathop{E}_{\substack{s\sim p(s)\\ a\sim\pi_\theta(a|s)}} \nabla \log \pi_\theta(a|s) Q(s,a)$$

REINFORCE baselines

- Gradient direction ∇J stays the same
- Variance may change

Gradient variance: $Var[Q(s,a)-b(s)]$

as a random variable over (s, a)

$$Var[Q(s,a)]-2\cdot Cov[Q(s,a),b(s)]+Var[b(s)]$$

REINFORCE baselines

- Gradient direction ∇J stays the same
- Variance may change

Gradient variance: $Var[Q(s,a)-b(s)]$

as a random variable over (s, a)

$$Var[Q(s,a)]-2\cdot Cov[Q(s,a),b(s)]+Var[b(s)]$$

If b(s) correlates with Q(s,a), variance decreases

REINFORCE baselines

- Gradient direction ∇J stays the same
- Variance may change

Gradient variance: $Var[Q(s,a)-b(s)]$

as a random variable over (s, a)

$$Var[Q(s,a)]-2\cdot Cov[Q(s,a),b(s)]+Var[b(s)]$$

Q: can you suggest any such b(s)?

REINFORCE baselines

- Gradient direction ∇J stays the same
- Variance may change

Gradient variance: $Var[Q(s,a)-b(s)]$

as a random variable over (s, a)

$$Var[Q(s,a)]-2\cdot Cov[Q(s,a),b(s)]+Var[b(s)]$$

Naive baseline: b = moving average Q *over all (s, a),* *Var[b(s)] = 0,* *Cov[Q, b] > 0*

REINFORCE baselines

Better baseline: b(s) = V(s)

$$\nabla J \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{s,a \in z_i} \nabla \log \pi_\theta(a|s) \cdot (Q(s,a) - V(s))$$

Q: but how do we predict V(s)?

Actor-critic

- Learn both V(s) and $\pi_\theta(a|s)$
- Hope for the best of both worlds :)

Advantage actor-critic

Idea: learn both $\pi_\theta(a|s)$ and $V_\theta(s)$

Use $V_\theta(s)$ to learn $\pi_\theta(a|s)$ faster!

Q: how can we estimate **A(s,a)** from (s,a,r,s') and V-function?

Advantage actor-critic

Idea: learn both $\pi_\theta(a|s)$ and $V_\theta(s)$

Use $V_\theta(s)$ to learn $\pi_\theta(a|s)$ faster!

$$A(s,a)=Q(s,a)-V(s)$$

$$Q(s,a)=r+\gamma\cdot V(s')$$

$$A(s,a)=r+\gamma\cdot V(s')-V(s)$$

Advantage actor-critic

Idea: learn both $\pi_\theta(a|s)$ and $V_\theta(s)$

Use $V_\theta(s)$ to learn $\pi_\theta(a|s)$ faster!

$$A(s,a)=Q(s,a)-V(s)$$

$$Q(s,a)=r+\gamma\cdot V(s')$$

$$A(s,a)=r+\gamma\cdot V(s')-V(s)$$

Also: n-step version

Advantage actor-critic

Idea: learn both $\pi_\theta(a|s)$ and $V_\theta(s)$

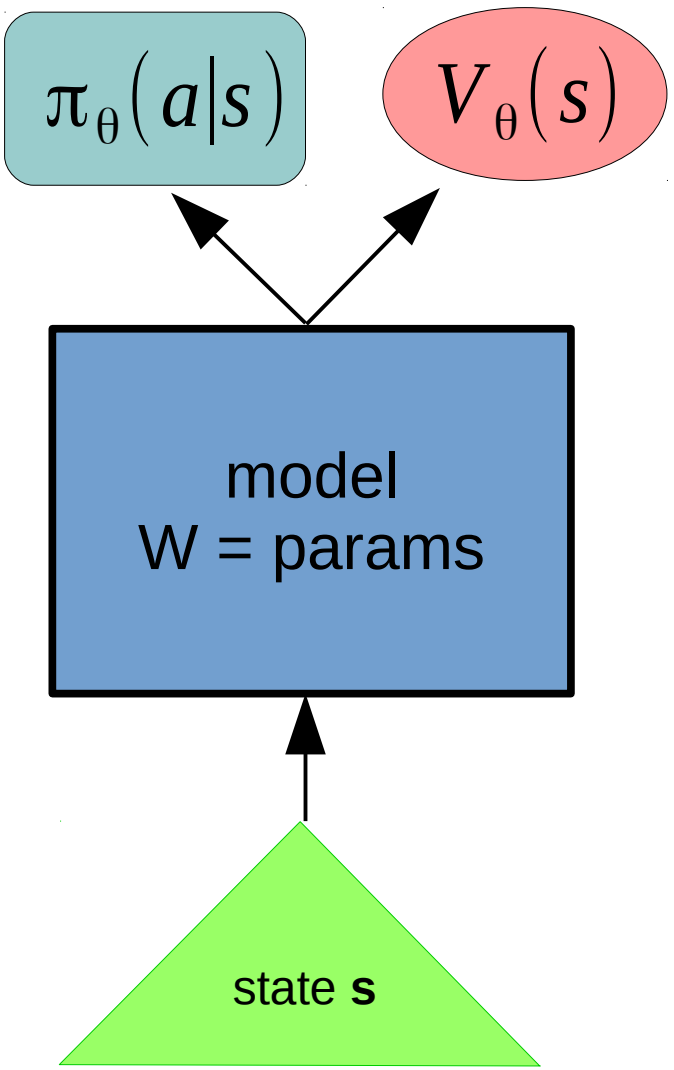
Use $V_\theta(s)$ to learn $\pi_\theta(a|s)$ faster!

$$A(s,a) = r + \gamma \cdot V(s') - V(s)$$

$$\nabla J_{actor} \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{s,a \in z_i} \nabla \log \pi_\theta(a|s) \cdot \underline{A(s,a)}$$

consider const

Advantage actor-critic



Improve policy:

$$\nabla J_{actor} \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{s,a \in z_i} \nabla \log \pi_\theta(a|s) \cdot A(s,a)$$

Improve value:

$$L_{critic} \approx \frac{1}{N} \sum_{i=0}^{N} \sum_{s,a \in z_i} \left(V_\theta(s) - [r + \gamma \cdot V(s')]\right)^2$$

Continuous action spaces

What if there's continuously many actions?

- Robot control: control motor voltage
- Trading: assign money to equity

How does the algorithm change?

Continuous action spaces

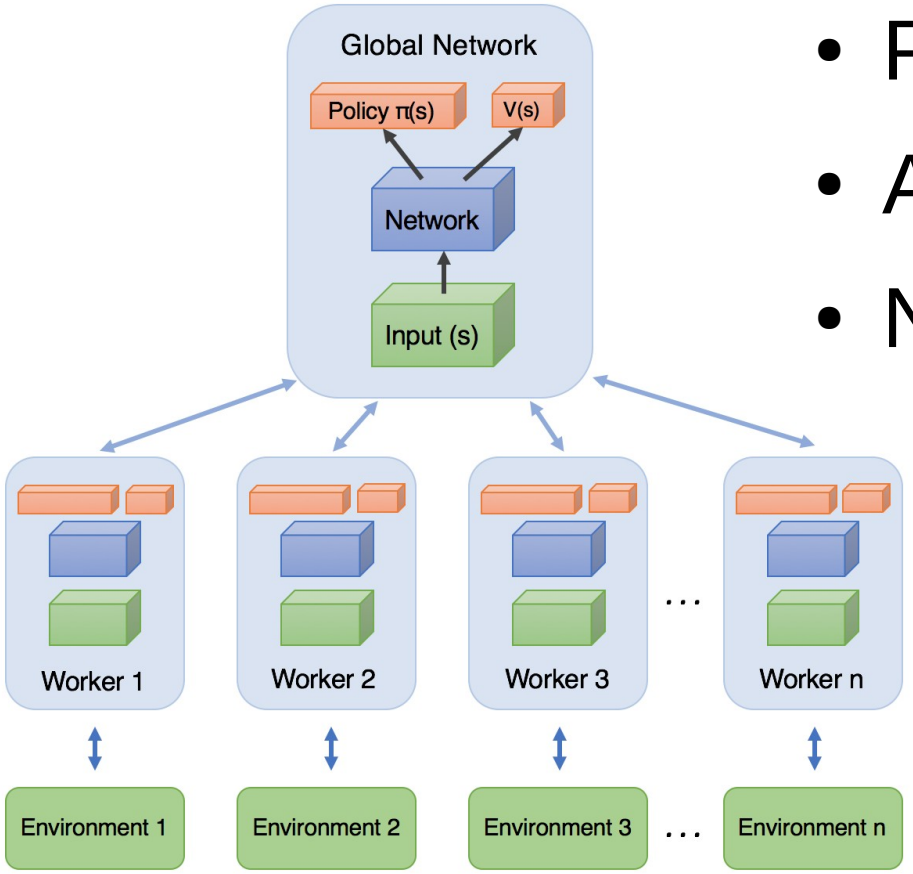
What if there's continuously many actions?

- Robot control: control motor voltage
- Trading: assign money to equity

How does the algorithm change?

it doesn't :)
Just plug in a different formula for pi(a|s), e.g. normal distribution

Asynchronous advantage actor-critic



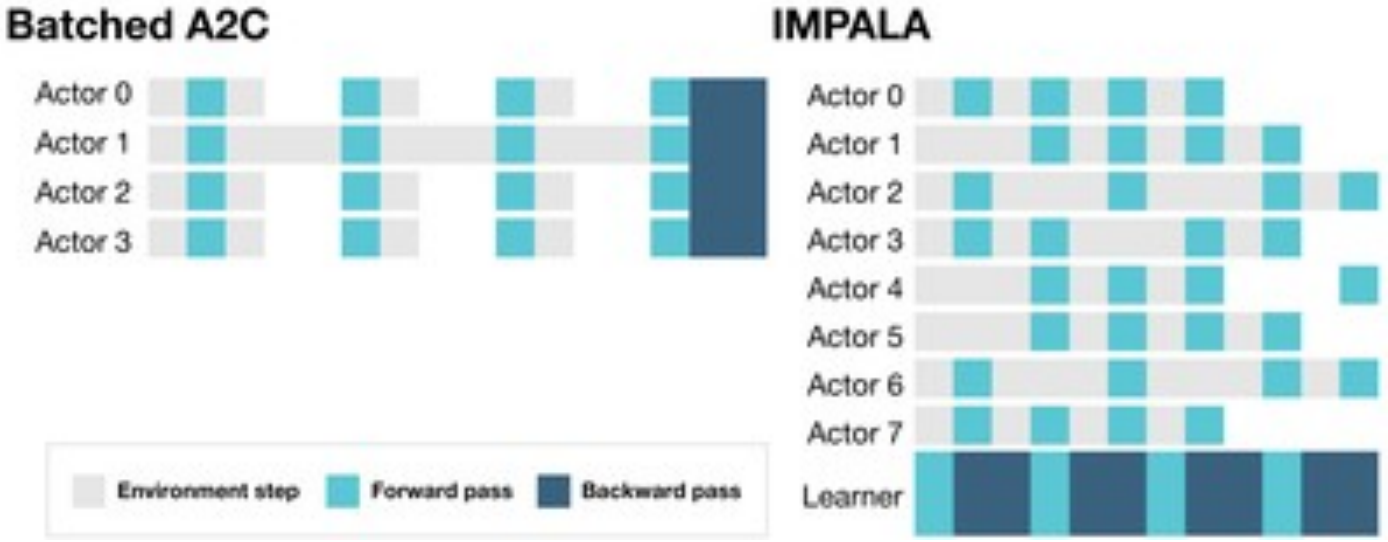
- Parallel game sessions
- Async multi-CPU training
- No experience replay

- LSTM policy
- N-step advantage
- No experience replay

Read more: https://arxiv.org/abs/1602.01783

IMPALA

- Massively parallel
- Separate actor / learner processes
- Small experience replay w/ importance sampling

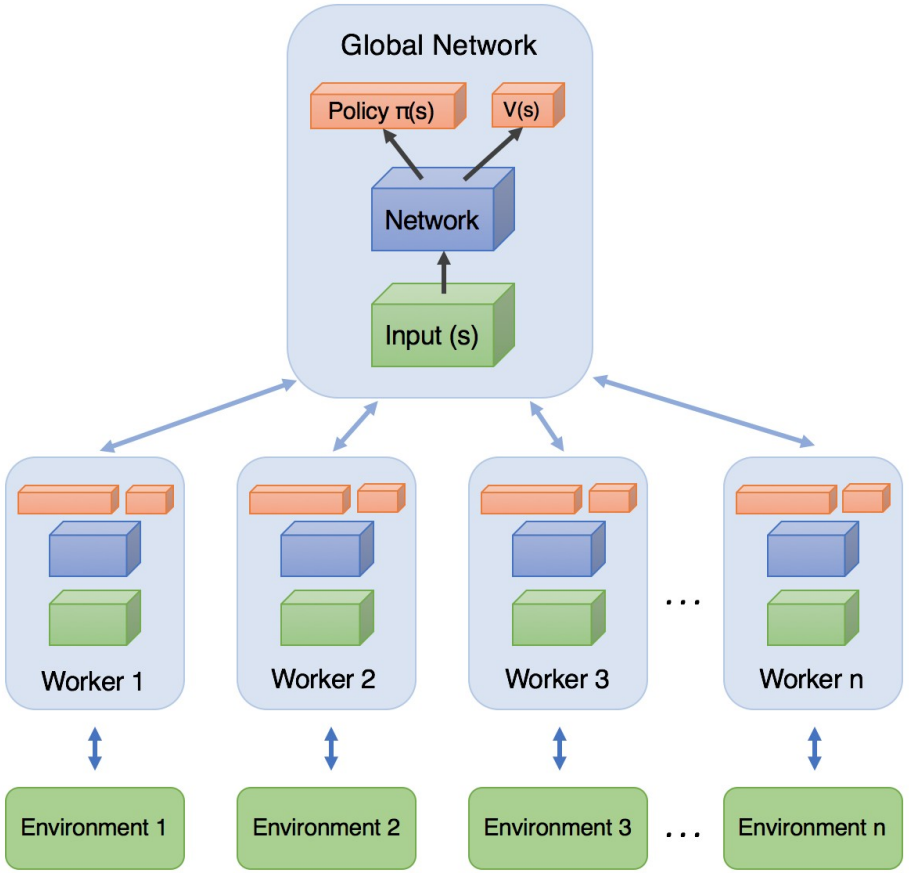


Read more: https://arxiv.org/abs/1802.01561

Duct tape zone

- V(s) errors less important than in Q-learning
  - actor still learns even if critic is random, just slower
- Regularize with entropy
  - to prevent premature convergence
- Learn on parallel sessions
  - Or super-small experience replay
- Use logsoftmax for numerical stability

Asynchronous advantage actor-critic



Outro

- Most recent methods are using best from both Policy-based and Value-based worlds
- Remember the **Expected Grad-Log-Prob Lemma**
- See this paper (ICLR 2019) for the proof of the Variance reduction