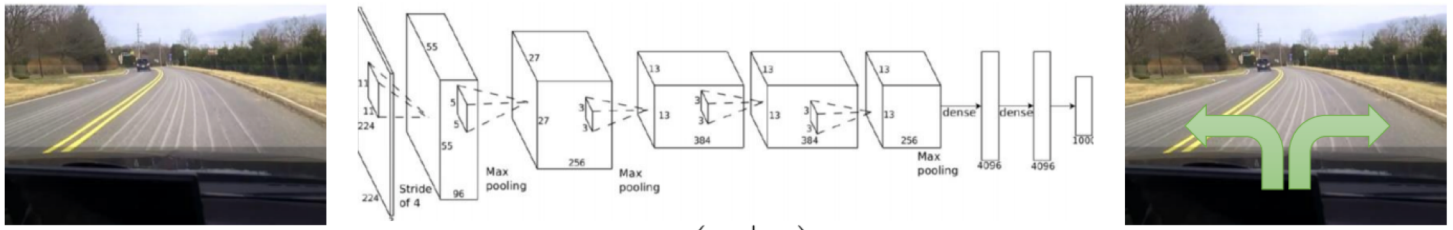
Policy Gradient Methods



Sources: Stanford CS 231n, Berkeley Deep RL course, David Silver's RL course

Image source

Policy Gradient Methods

- Instead of indirectly representing the policy using Q-values, it can be more efficient to parameterize and learn it directly
  - Especially in large or continuous action spaces

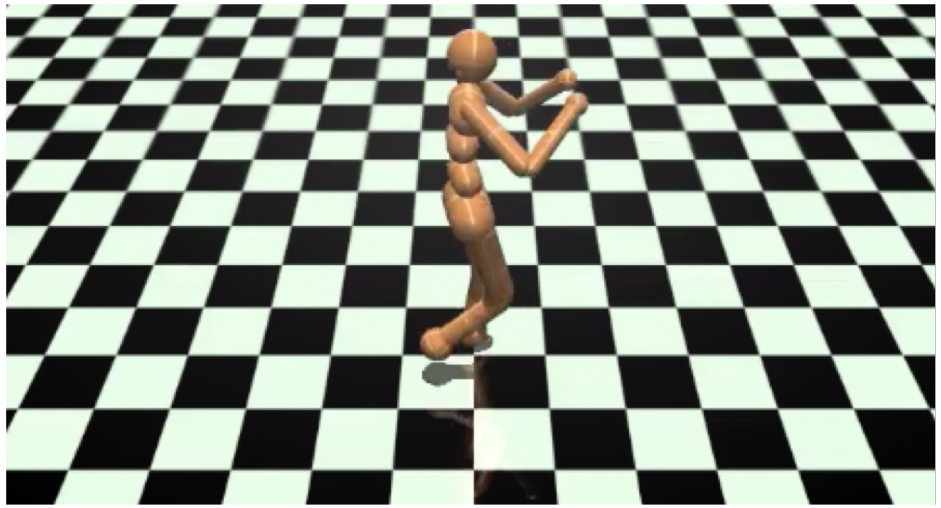


Image source: OpenAI Gym

Outline

- Stochastic policy representation
- Finding the policy gradient
- REINFORCE algorithm
- Reducing variance: Actor-critic algorithms
- Asynchronous advantage actor-critic (A3C)
- Application: policy gradients for image captioning

Stochastic policy representation

- Learn a function giving the probability distribution over actions from the current state:

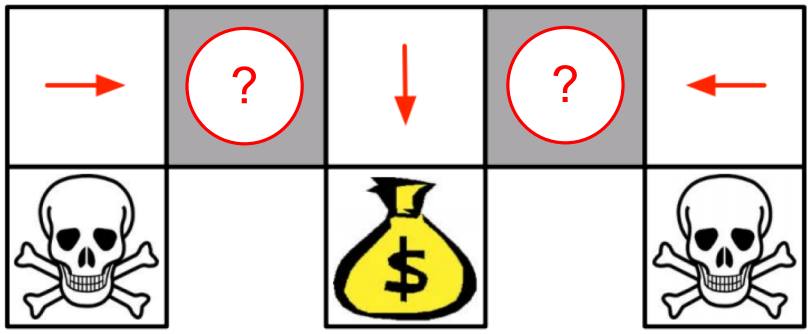
$$\pi_\theta(a|s) \approx P(a|s)$$

Stochastic policy representation

- Learn a function giving the probability distribution over actions from the current state:

$$\pi_\theta(a|s) \approx P(a|s)$$

- Why stochastic policies?
  - There are examples even of grid world scenarios where only a stochastic policy can reach optimality



Source:
D. Silver

The agent can't tell the difference between the gray cells

Stochastic policy representation

- Learn a function giving the probability distribution over actions from the current state:

$$\pi_\theta(a|s) \approx P(a|s)$$

- Why stochastic policies?
  - It's mathematically convenient!
    - Softmax policy:

$$\pi_\theta(a|s) = \frac{\exp(f_\theta(s,a))}{\sum_{a'} \exp(f_\theta(s,a'))}$$

    - Gaussian policy (for continuous action spaces):

$$\pi_\theta(a|s) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(a - f_\theta(s))^2}{2\sigma^2}\right)$$

Expected value of a policy

$$J(\theta) = \mathbb{E}\left[\sum_{t \geq 0} \gamma^t \, r_t \mid \pi_\theta\right]$$

$$= \mathbb{E}_\tau[r(\tau)]$$

Expectation of return over *trajectories* $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \ldots)$

$$= \int_\tau r(\tau) \underbrace{p(\tau; \theta)} d\tau$$

Probability of trajectory τ under policy with parameters θ

Finding the policy gradient

$$J(\theta) = \int_{\tau} r(\tau) p(\tau;\theta) d\tau$$

$$\nabla_\theta J(\theta) = \int_{\tau} r(\tau) \nabla_\theta p(\tau;\theta) d\tau$$

$$= \int_{\tau} r(\tau) p(\tau;\theta) \frac{\nabla_\theta p(\tau;\theta)}{p(\tau;\theta)} d\tau$$

$$= \int_{\tau} r(\tau) p(\tau;\theta) \nabla_\theta \log p(\tau;\theta)\, d\tau$$

$$= \mathbb{E}_\tau [r(\tau) \nabla_\theta \log p(\tau;\theta)]$$

Finding the policy gradient

$$\nabla_\theta J(\theta) = \mathbb{E}_\tau[r(\tau)\nabla_\theta \log p(\tau;\theta)]$$

Probability of trajectory

$\tau = (s_0, a_0, s_1, a_1, \dots)$

$$p(\tau;\theta) = \prod_{t\geq 0} \pi_\theta(a_t|s_t)P(s_{t+1}|s_t,a_t)$$

$$\log p(\tau;\theta) = \sum_{t\geq 0} [\log \pi_\theta(a_t|s_t) + \log P(s_{t+1}|s_t,a_t)]$$

$$\nabla_\theta \log p(\tau;\theta) = \sum_{t\geq 0} \nabla_\theta \log \pi_\theta(a_t|s_t)$$

The *score function*

Score function $\nabla_\theta \log \pi_\theta(a|s)$

- For softmax policy:

$$\pi_\theta(a|s) = \frac{\exp(f_\theta(s,a))}{\sum_{a'} \exp(f_\theta(s,a'))}$$

$$\nabla_\theta \log \pi_\theta(a_t|s_t) = \nabla_\theta f_\theta(s,a) - \sum_{a'} \pi_\theta(a'|s\,)\, \nabla_\theta f_\theta(s,a')$$

- For Gaussian policy:

$$\pi_\theta(a|s) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(a - f_\theta(s))^2}{2\sigma^2}\right)$$

$$\nabla_\theta \log \pi_\theta(a_t|s_t) = \frac{\big(a - f_\theta(s)\big)}{\sigma^2} \nabla_\theta f_\theta(s) - \text{const.}$$

Finding the policy gradient

$$\nabla_\theta J(\theta) = \mathbb{E}_\tau[r(\tau)\nabla_\theta \log p(\tau;\theta)]$$

$$\nabla_\theta \log p(\tau;\theta) = \sum_{t\geq 0} \nabla_\theta \log \pi_\theta(a_t|s_t)$$

$$\nabla_\theta J(\theta) = \mathbb{E}_\tau \left[\underbrace{\left(\sum_{t\geq 0} \gamma^t r_t \right)}_{\text{Return of trajectory } \tau} \underbrace{\left(\sum_{t\geq 0} \nabla_\theta \log \pi_\theta(a_t|s_t) \right)}_{\text{Gradient of log-likelihood of actions under current policy}} \right]$$

- How do we estimate the gradient in practice?

Finding the policy gradient

$$\nabla_\theta J(\theta) = \mathbb{E}_\tau[r(\tau)\nabla_\theta \log p(\tau;\theta)]$$

$$\nabla_\theta \log p(\tau;\theta) = \sum_{t\geq 0} \nabla_\theta \log \pi_\theta(a_t|s_t)$$

$$\nabla_\theta J(\theta) = \mathbb{E}_\tau \left[\left(\sum_{t\geq 0} \gamma^t r_t\right)\left(\sum_{t\geq 0} \nabla_\theta \log \pi_\theta(a_t|s_t)\right)\right]$$

- Stochastic approximation: sample N trajectories $\tau_1, \ldots, \tau_N$

$$\nabla_\theta J(\theta) \approx \frac{1}{N}\sum_{i=1}^{N}\left(\sum_{t=0}^{T_i} \gamma^t r_{i,t}\right)\left(\sum_{t=0}^{T_i} \nabla_\theta \log \pi_\theta(a_{i,t}|s_{i,t})\right)$$

REINFORCE algorithm

1. Sample N trajectories τ_i using current policy π_θ

2. Estimate the policy gradient:

$$\nabla_\theta J(\theta) \approx \frac{1}{N} \sum_{i=1}^{N} r(\tau_i) \left(\sum_{t=0}^{T_i} \nabla_\theta \log \pi_\theta(a_{i,t}|s_{i,t}) \right)$$

3. Update parameters by gradient ascent:

$$\theta \leftarrow \theta + \eta \nabla_\theta J(\theta)$$

Williams et al. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning, 8(3):229-256, 1992

REINFORCE: Single-step version

1. In state s, sample action a using current policy π_θ, observe reward r

2. Estimate the policy gradient:

$$\nabla_\theta J(\theta) \approx r \, \nabla_\theta \log \pi_\theta(a|s)$$

3. Update parameters by gradient ascent:

$$\theta \leftarrow \theta + \eta \nabla_\theta J(\theta)$$

- What effect does this update have?
  - Push up the probability of good actions, push down probability of bad actions

Outline

- Stochastic policy representation
- Finding the policy gradient
- REINFORCE algorithm
- Reducing variance: Actor-critic algorithms

Reducing variance

- Gradient estimate (for a single trajectory):

$$\nabla_\theta J(\theta) \approx \sum_{t \geq 0} r(\tau) \nabla_\theta \log \pi_\theta(a_t | s_t)$$

- **General problem**: rewards of sampled trajectories are too noisy and lead to unreliable policy gradients

Reducing variance

- Gradient estimate (for a single trajectory):

$$\nabla_\theta J(\theta) \approx \sum_{t \geq 0} r(\tau) \nabla_\theta \log \pi_\theta(a_t|s_t)$$

- First observation: it seems bad to weight each action in a trajectory by the return of the entire trajectory. In particular, rewards obtained *before* an action was taken should not be used to weight that action
  - Instead, for each action, consider only the cumulative *future* reward:

$$\nabla_\theta J(\theta) \approx \sum_{t \geq 0} \left(\sum_{t' \geq t} \gamma^{t'-t} r_{t'} \right) \nabla_\theta \log \pi_\theta(a_t|s_t)$$

Reducing variance

$$\nabla_\theta J(\theta) \approx \sum_{t \geq 0} \left(\sum_{t' \geq t} \gamma^{t'-t} r_{t'} \right) \nabla_\theta \log \pi_\theta(a_t | s_t)$$

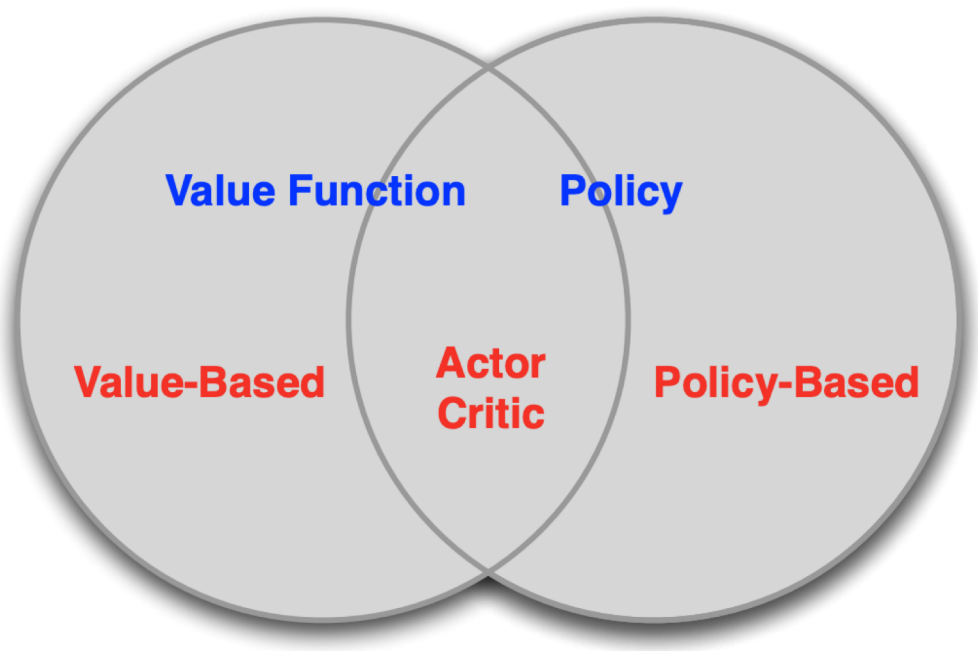
Observed cumulative reward after taking action a_t in state s_t

- But then, why not use *expected* cumulative reward?

$$\nabla_\theta J(\theta) \approx \sum_{t \geq 0} Q^{\pi_\theta}(s_t, a_t) \nabla_\theta \log \pi_\theta(a_t | s_t)$$

Actor-Critic algorithms

- Combine policy gradients and Q-learning by simultaneously training an *actor* (the policy) and a *critic* (the Q-function)



Source: D. Silver

Reducing variance

$$\nabla_\theta J(\theta) \approx \sum_{t \geq 0} Q^{\pi_\theta}(s_t, a_t) \nabla_\theta \log \pi_\theta(a_t|s_t)$$

- Next observation: the raw Q-values are not the most useful. If all Q-values are good, we will try to push up the probabilities of all the actions
- Instead, compare Q-values of actions to some *baseline function* of the state:

$$\nabla_\theta J(\theta) \approx \sum_{t \geq 0} \underbrace{\left(Q^{\pi_\theta}(s_t, a_t) - V^{\pi_\theta}(s_t)\right)}_{\text{Advantage function}} \nabla_\theta \log \pi_\theta(a_t|s_t)$$

Estimating the advantage function

- Advantage function:

$$A^{\pi_\theta}(s,a) = Q^{\pi_\theta}(s,a) - V^{\pi_\theta}(s)$$
$$= \mathbb{E}_{\pi_\theta}[r + \gamma\, V^{\pi_\theta}(s')|s,a] - V^{\pi_\theta}(s)$$
$$\approx r + \gamma\, V^{\pi_\theta}(s') - V^{\pi_\theta}(s)$$

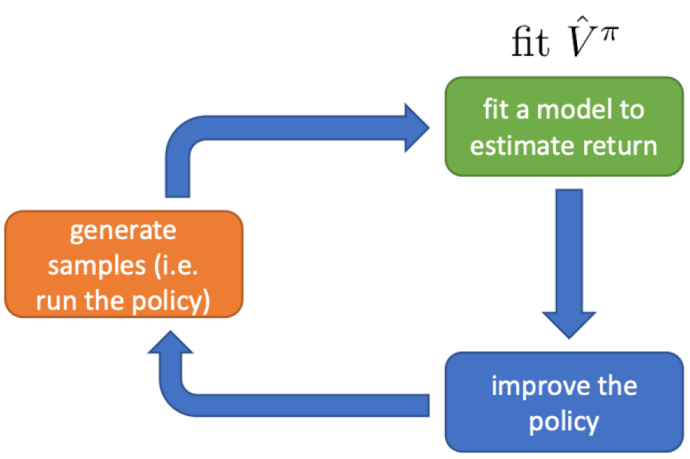
(from a single transition)

- Therefore, it is sufficient to learn the value function:

$$V^{\pi_\theta}(s) \approx V_w(s)$$

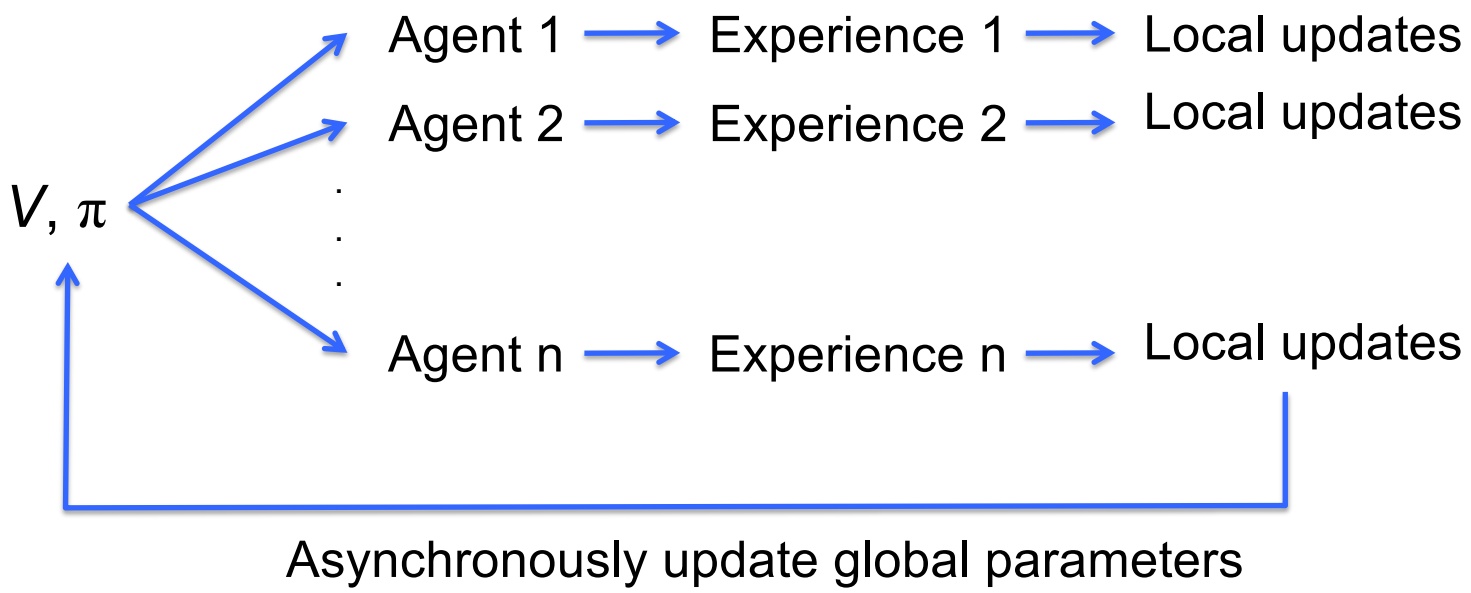
Online actor-critic algorithm

1. Sample action a using current policy, observe reward r, successor state s'
2. Update $V_w(s)$ towards target $r + \gamma V_w(s')$
3. Estimate $A^{\pi_\theta}(s,a) = r + \gamma V_w(s') - V_w(s)$
4. Estimate $\nabla_\theta J(\theta) = A^{\pi_\theta}(s,a) \nabla_\theta \log \pi_\theta(a|s)$
5. Update policy parameters: $\theta \leftarrow \theta + \eta \nabla_\theta J(\theta)$



Source: Berkeley RL course

Asynchronous advantage actor-critic (A3C)



Mnih et al. Asynchronous Methods for Deep Reinforcement Learning. ICML 2016

Asynchronous advantage actor-critic (A3C)

Method	Training Time	Mean	Median
DQN	8 days on GPU	121.9%	47.5%
Gorila	4 days, 100 machines	215.2%	71.3%
D-DQN	8 days on GPU	332.9%	110.9%
Dueling D-DQN	8 days on GPU	343.8%	117.1%
Prioritized DQN	8 days on GPU	463.6%	127.6%
A3C, FF	1 day on CPU	344.1%	68.2%
A3C, FF	4 days on CPU	496.8%	116.6%
A3C, LSTM	4 days on CPU	623.0%	112.6%

Mean and median human-normalized scores over 57 Atari games

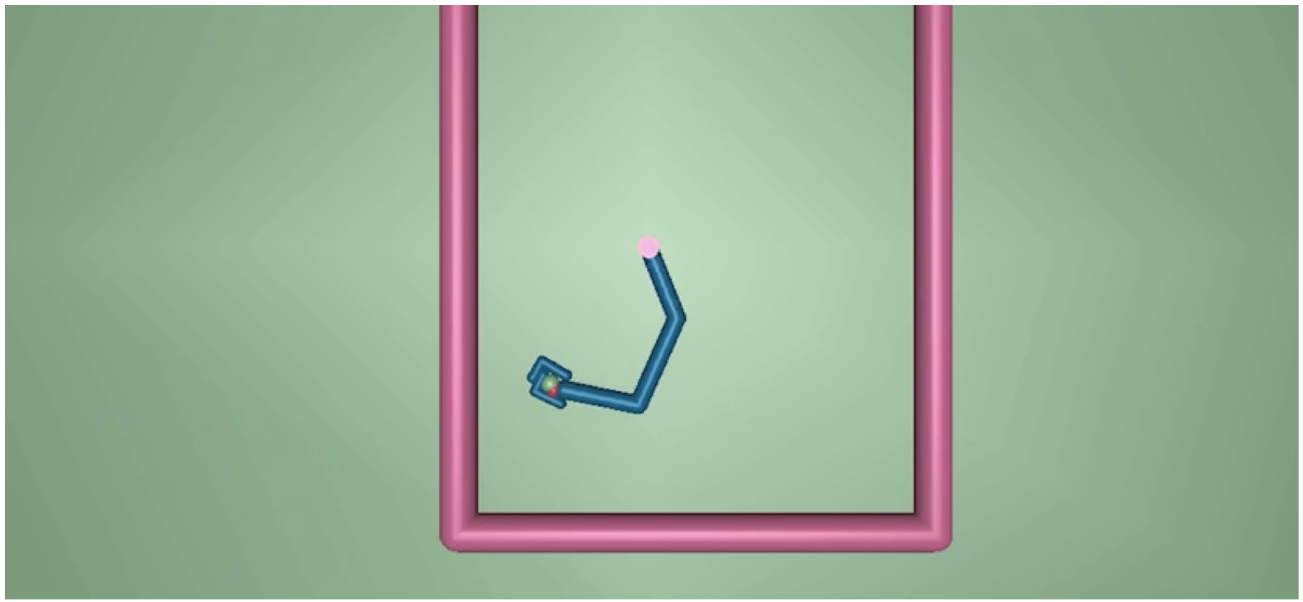
Mnih et al. Asynchronous Methods for Deep Reinforcement Learning. ICML 2016

Asynchronous advantage actor-critic (A3C)

TORCS car racing simulation video

Mnih et al. Asynchronous Methods for Deep Reinforcement Learning. ICML 2016

Asynchronous advantage actor-critic (A3C)

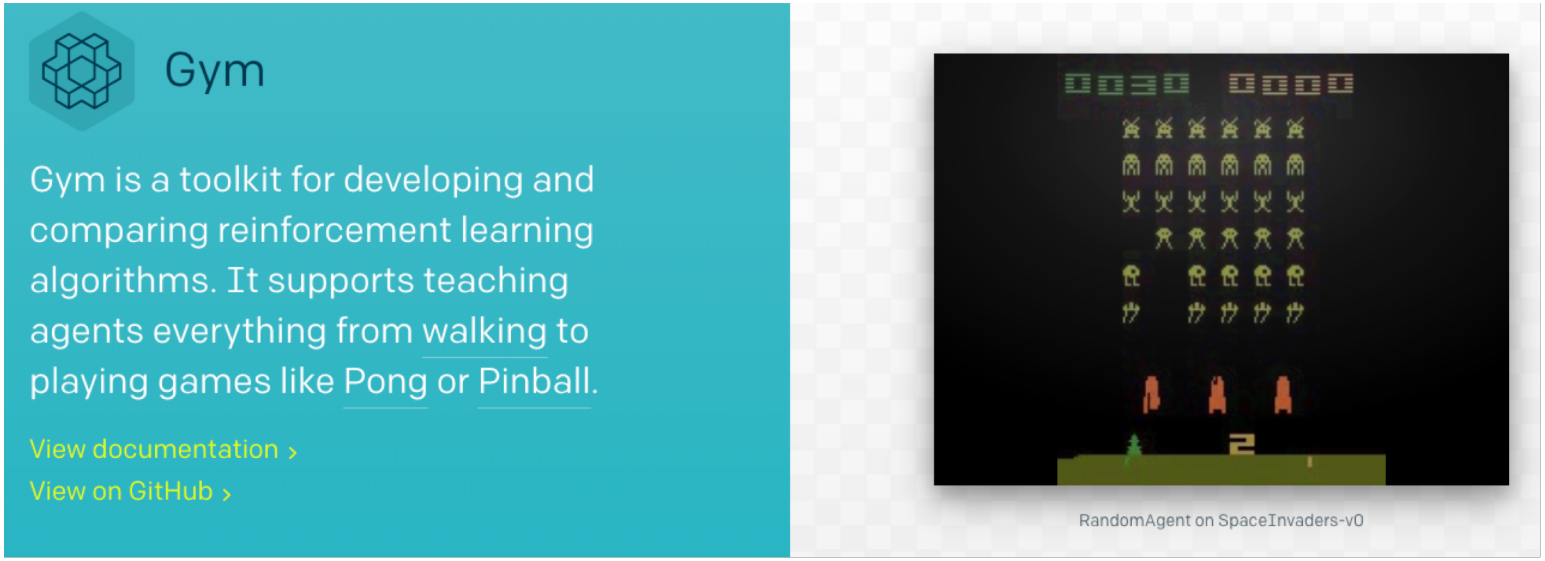


Motor control tasks video

Mnih et al. Asynchronous Methods for Deep Reinforcement Learning. ICML 2016

Benchmarks and environments for Deep RL

OpenAI Gym



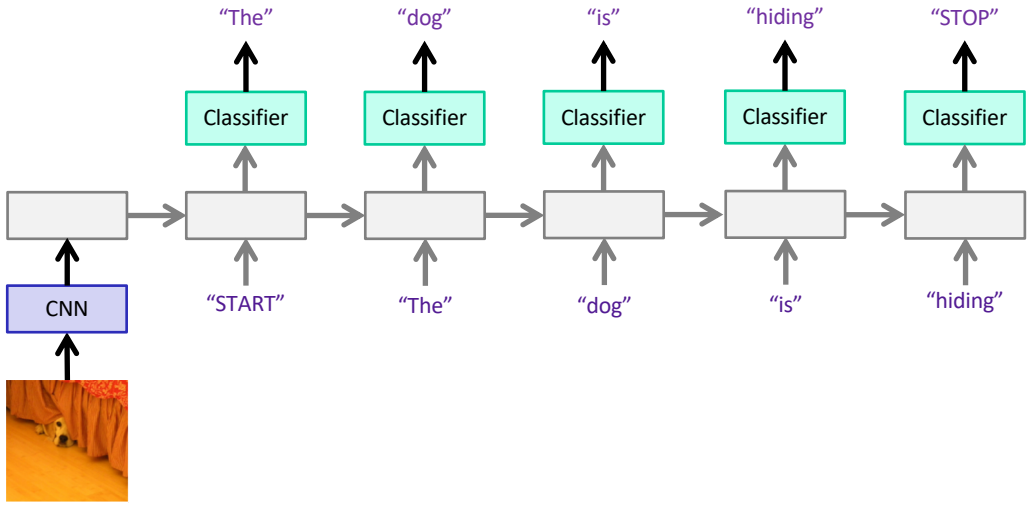
https://gym.openai.com/

Outline

- Stochastic policy representation
- Finding the policy gradient
- REINFORCE algorithm
- Reducing variance: Actor-critic algorithms
- Asynchronous advantage actor-critic (A3C)
- Application: policy gradients for image captioning

Application: Policy gradients for image captioning

- Standard training: maximize log-likelihood of reference sentences given the image
  - Log-likelihood is not related to any specialized caption quality scores (BLEU, CIDER, METEOR, SPICE, etc.)
  - Does not reward high-quality generated sentences that are not identical to the reference ones



Application: Policy gradients for image captioning

- Standard training: maximize log-likelihood of reference sentences given the image
  - Log-likelihood is not related to any specialized caption quality scores (BLEU, CIDER, METEOR, SPICE, etc.)
  - Does not reward high-quality generated sentences that are not identical to the reference ones

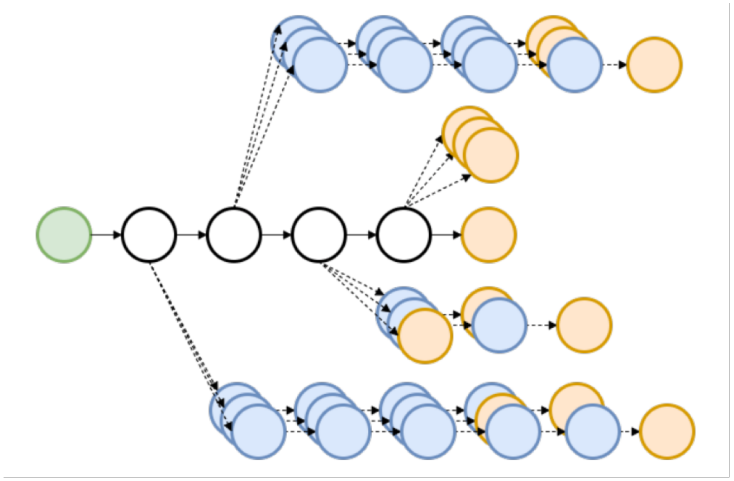
- Solution: train model with a non-differentiable caption quality score as reward

Application: Policy gradients for image captioning

- MDP formulation for captioning
  - **States**
    - Image, sequence generated so far
  - **Actions**
    - Generate word from vocabulary
  - **Transition model**
    - Append generated word to sequence (deterministic)
  - **Reward**
    - BLEU/CIDER/METEOR/SPICE etc. at the end of the sequence
- Challenges
  - Action space is large
  - Reward is sparse

Application: Policy gradients for image captioning

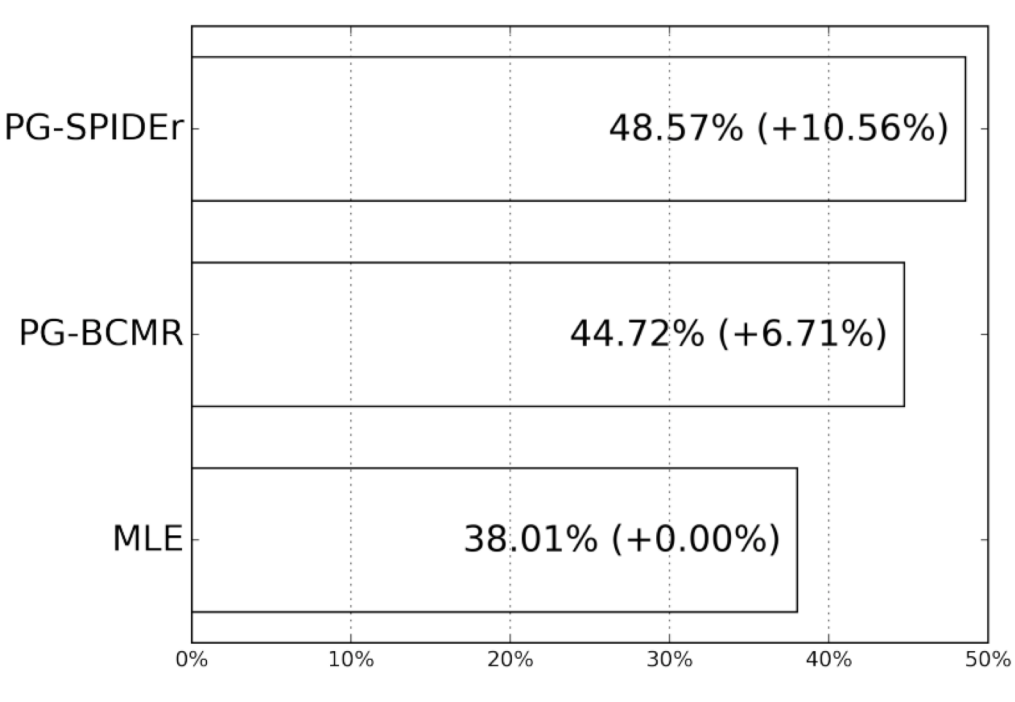
- Initialize policy with ML-trained model
- Estimate expected intermediate returns $Q(s, a)$ using *Monte Carlo rollouts*
  - For each partial sequence, sample *K* continuations until completion and average their reward



S. Liu, Z. Zhu, N. Ye, S. Guadarrama, and K. Murphy, Improved Image Captioning via Policy Gradient optimization of SPIDEr, ICCV 2017

Application: Policy gradients for image captioning

Human study on 492 images: percentage of captions judged to be “not bad”
(87% of human captions are judged to be “not bad”)



Application: Policy gradients for image captioning

Comparison of captions

	1. A woman walking on a city street in a red coat. 2. A group of people that are standing on the side of a street. 3. A woman in a red jacket crossing the street 4. a street light some people and a woman wearing a red jacket 5. A blonde woman in a red coat crosses the street with her friend.	• MLE: a woman walking down a street while holding an umbrella . • PG-SPICE: a group of people walking down a street with a man on a street holding a traffic light and a traffic light on a city street with a city street • MIXER-BCMR: a group of people walking down a street . • MIXER-BCMR-A: a group of people walking down a street . • PG-BCMR: a group of people walking down a city street . • PG-SPIDEr: a group of people walking down a street with a traffic light .
	1. A group of people converse in an office setting. 2. A group of people playing a game with remote controllers. 3. Four young people have crowded into a small office. 4. A group of people standing next to each other in a room. 5. a group of people standing next to each other with some of them holding video game controllers	• MLE: a group of people standing around a living room . • PG-SPICE: a group of people in a room with a man in a chair holding a nintendo wii remote in a living room with a man in a chair holding a • MIXER-BCMR: a group of people standing in a living room . • MIXER-BCMR-A: a group of people standing in a living room playing a video game . • PG-BCMR: a group of people standing in a room . • PG-SPIDEr: a group of people playing a video game in a living room .
	1. A man looking through a book on top of a table. 2. A man sitting on a bed looking at a book 3. a man is flipping through a book on a bed 4. A man sitting on a bed flipping through pages of a book. 5. A man in a black jacket is flipping through a large book.	• MLE: a man sitting in front of a laptop computer . • PG-SPICE: a man sitting in front of a book and a laptop on a table with a laptop computer on top of a table with a laptop computer on top of • MIXER-BCMR: a man sitting in a chair with a book . • MIXER-BCMR-A: a man sitting at a table with a book . • PG-BCMR: a man sitting in front of a book . • PG-SPIDEr: a man sitting at a table with a book .