Mila

McGill

Reinforcement Learning for Robotics

NCRN NSERC Canadian Robotics Network

RCRC Réseau canadien de robotique du CRSNG

David Meger

Nov 19, 2019

COMP 417 – Intro to Robotics

McGill CENTRE FOR Intelligent Machines

Where are we so far?

- Optimal control gives us a good language for optimizing behavior:
  - LQR with nice guarantees under strict assumptions
  - Extensions in the bonus slides to extend this idea by linearizing
  - More ideas under the umbrella trajectory optimization
- All of the above solve the optimal control constrained optimization assuming knowledge of both dynamics, f, and cost, g.

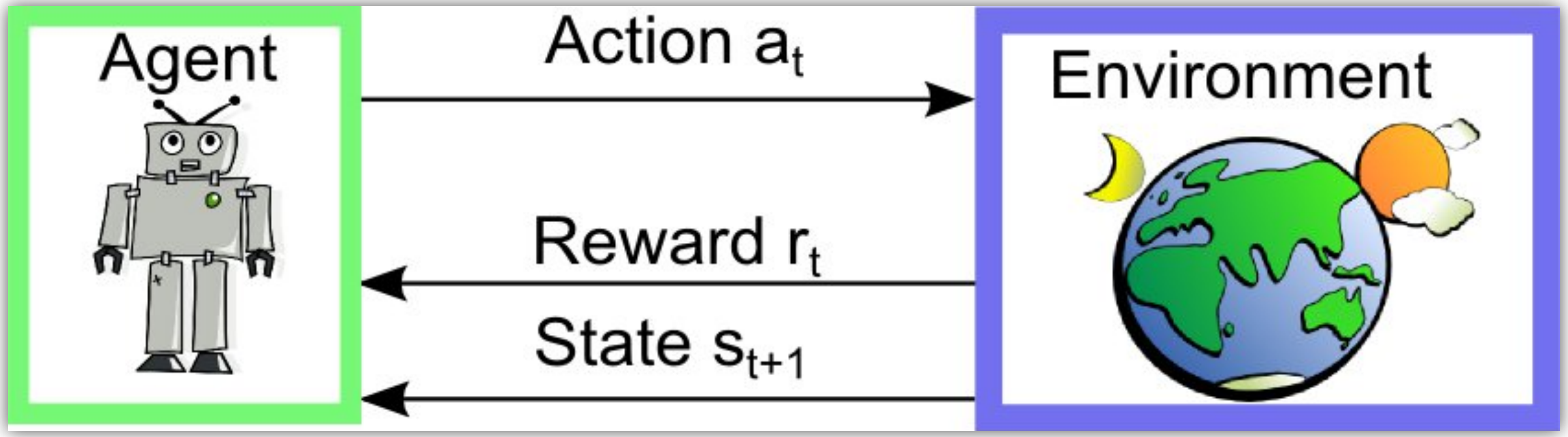
$$\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}, \mathbf{u})$$

$$g(\mathbf{x}_t, \mathbf{u}_t) = \mathbf{x}_t^T Q \mathbf{x}_t + \mathbf{u}_t^T R \mathbf{u}_t$$

- What if neither is known?

$$\underset{u_0,\ldots,u_N}{\operatorname{argmin}} \sum_{t=0}^{t=N} g(\mathbf{x}_t, \mathbf{u}_t)$$

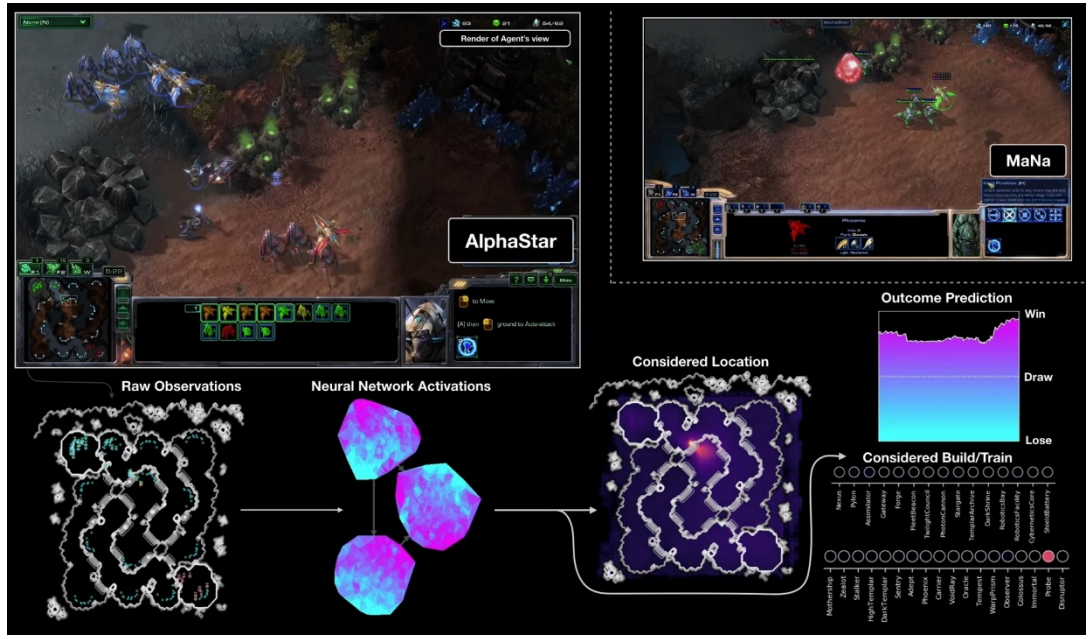
Reinforcement Learning



Reinforcement Learning

- RL means learning by trial and error
- Instead of typing in equations for f, A, B, Q, R, g etc, we assume the environment can give us these:
  - What is the f for a robot swimming with flippers?
    - We can guess it or compute crazy fluid dynamics, but we can more easily just observe its motions over time and use Machine Learning to derive a model
  - What is the g for winning ping pong?
    - This is difficult to write by math, but we can easily check the scoreboard during the game
- We will switch from guaranteed, math-based reasoning to data-driven methods. Model on human development learning. Can we get “AGI”?

RL agents outperform humans!



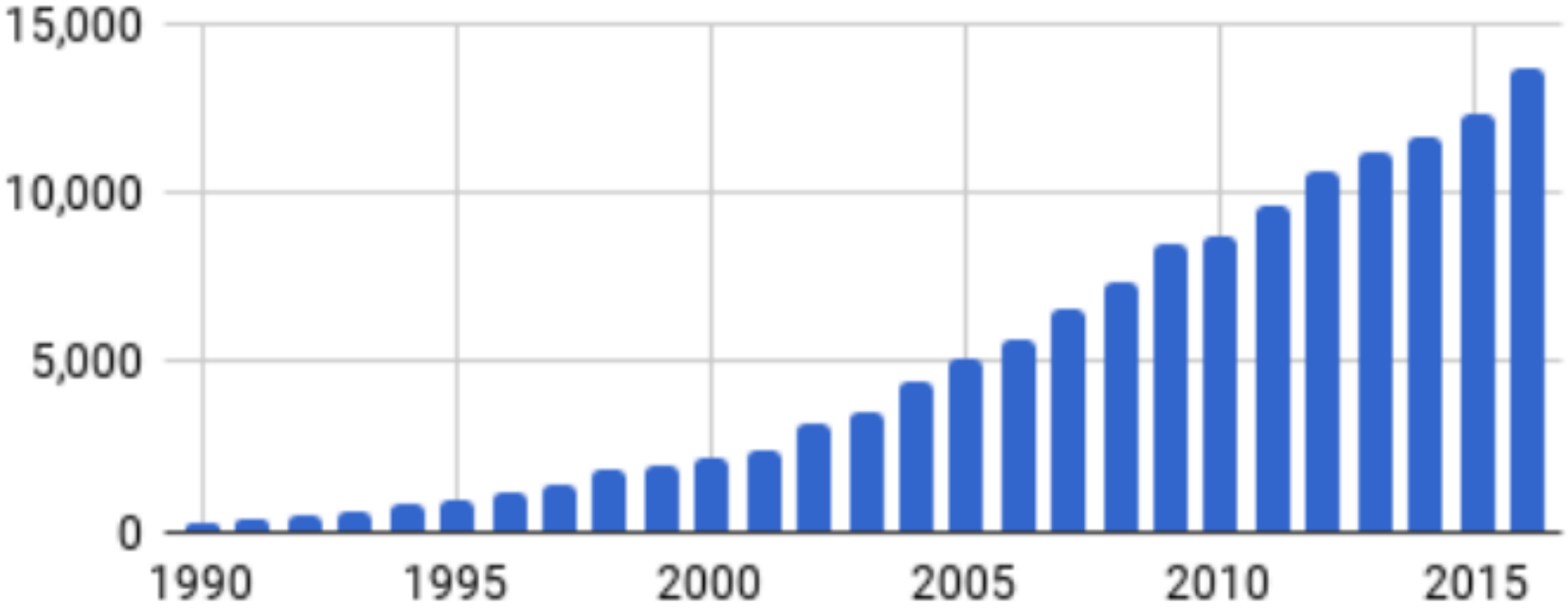


Figure 1: Growth of published reinforcement learning papers. Shown are the number of RL-related publications (y-axis) per year (x-axis) scraped from Google Scholar searches.

Value Iteration

- Recall: Value means the expected discounted reward from here on
- We saw that VI is dynamic programming that uses ***neighboring value estimate*** to capture “everything after the next step”
- Value estimates guaranteed to converge as each gets sequentially closer to truth

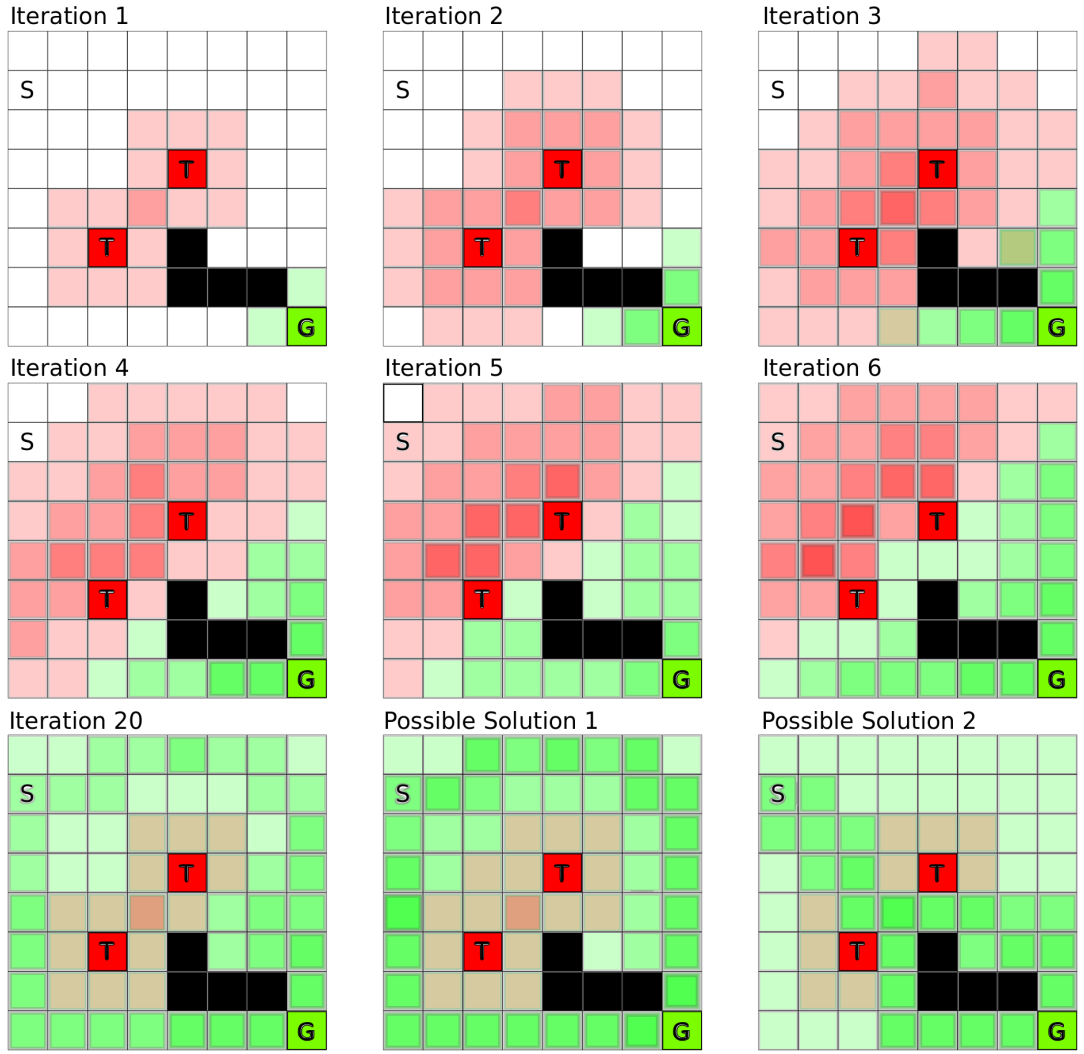


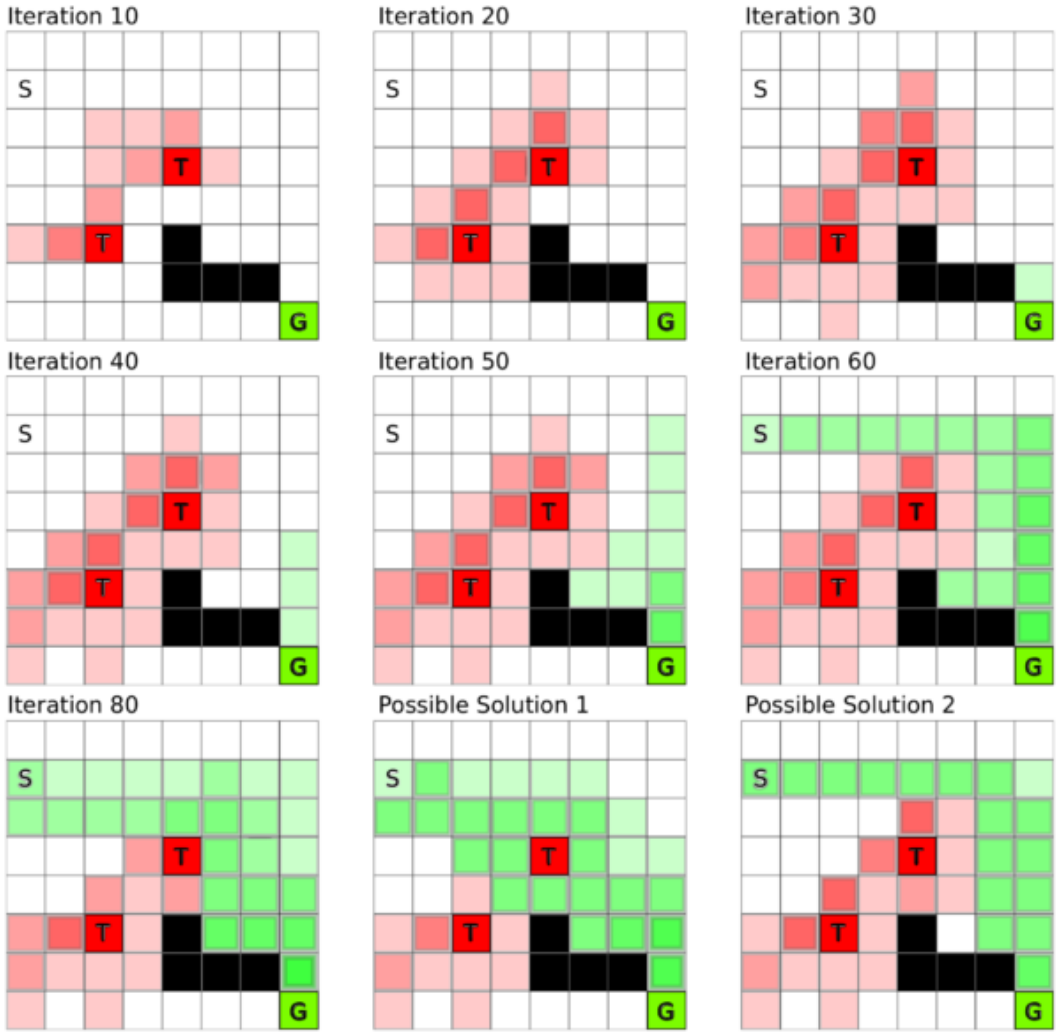
Image credit: https://devblogs.nvidia.com/deep-learning-nutshell-reinforcement-learning/

Q-Learning: Online

- Now we must experience the world through exploration:
  - s,a,r,s': being somewhere, taking an action, and observing where you end up plus reward
- Use s,a,r,s' to update Q values

$$Q(s,a) \leftarrow r + \gamma \max_{a'} Q(s',a')$$

- [Watkins 1989] proves convergence with infinite data



Credit: https://devblogs.nvidia.com/deep-learning-nutshell-reinforcement-learning/

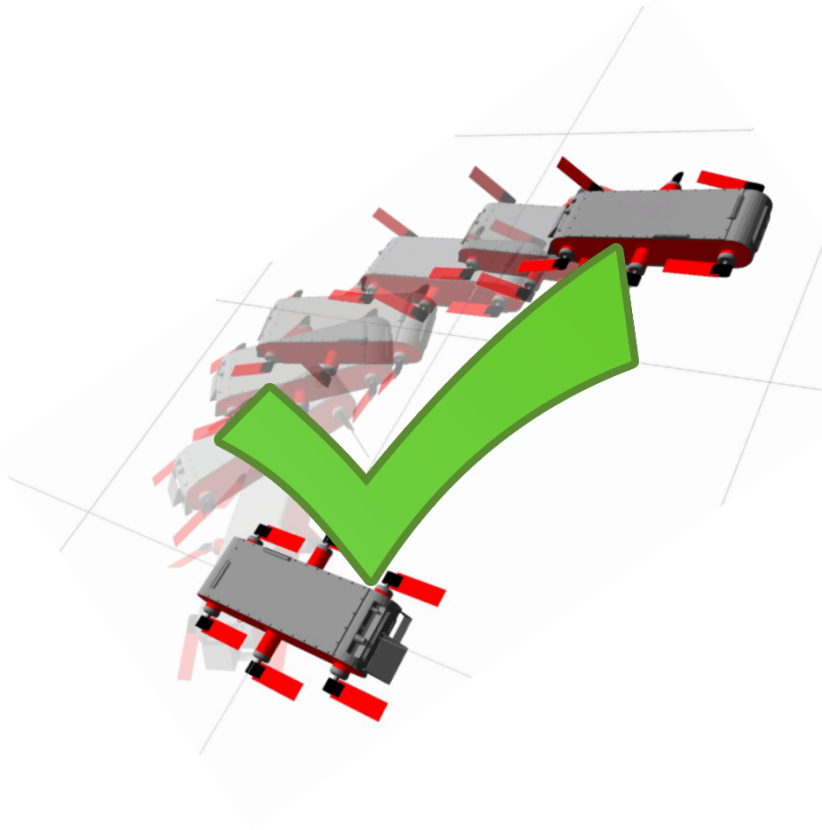
Deep Q Learning: The Engine behind Go, SC2, etc.

- We learn a deep neural network to predict the action-value function:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\Big[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)\Big]$$

- This is a recursive equation, so similar to our DP table, for every data point observed, we update estimates based on neighbors. Over time, the estimate (hopefully) converges to accurately capture ”how good it is to be at state S and take action A”.
- We can then apply control to the robot by taking the max over actions for the current state read by the sensors.

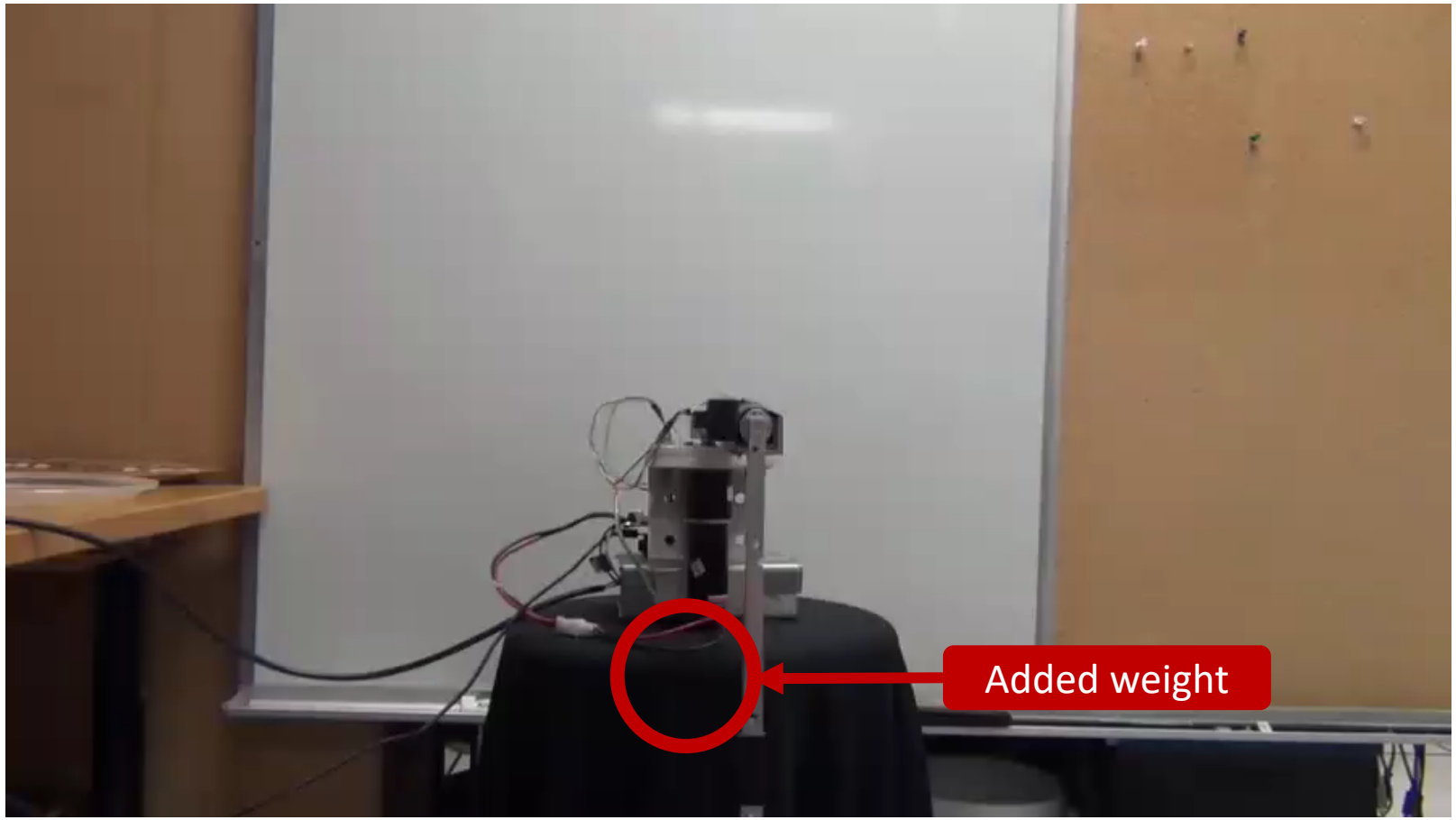
Unless RL works on robots, it's not going anywhere – Tomaso Poggio < 2017



Gait learning on underwater robot

David Meger, Juan Camilo Gamboa Higuera, Anqi Xu, Philippe Giguere and Gregory Dudek. Learning Legged Swimming Gaits from Experience. ICRA 2015.

Transfer between cart-pole of varied mass



Juan Camilo Gamboa Higuera, David Meger and Gregory Dudek. Adapting Robot Behaviours Through Policy Adjustments. ICRA 2017.

12

Discrete vs Continuous Value Functions

- For discrete problems, there may be many actions, but at least we can track the outcome of each

- In continuous control problems, there are truly infinite outcomes at least stage, and the slightest differences may matter!

Continuous State/Action: Actor Critic

- We replace the max operator with a policy π:

$$Q(s,a) \leftarrow r + \gamma Q'(s',a'), \qquad a' \sim \pi'(s')$$

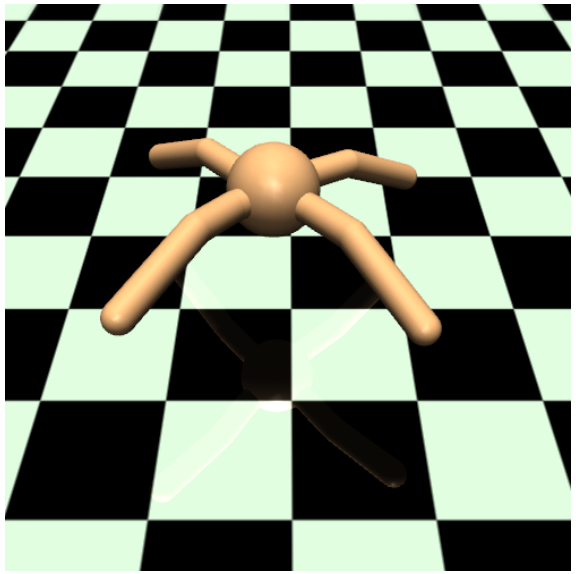
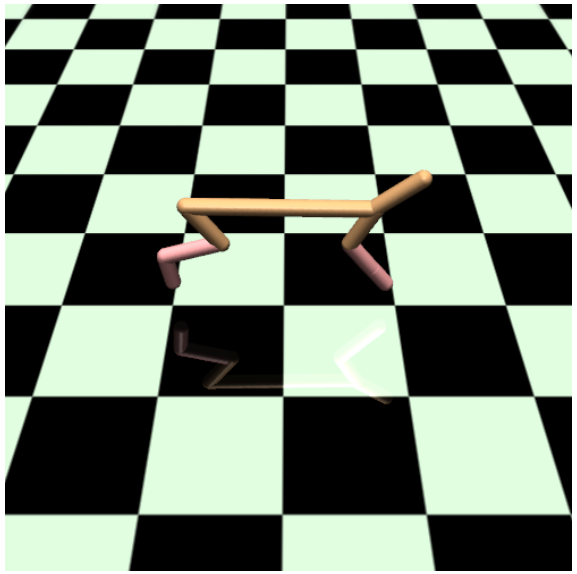
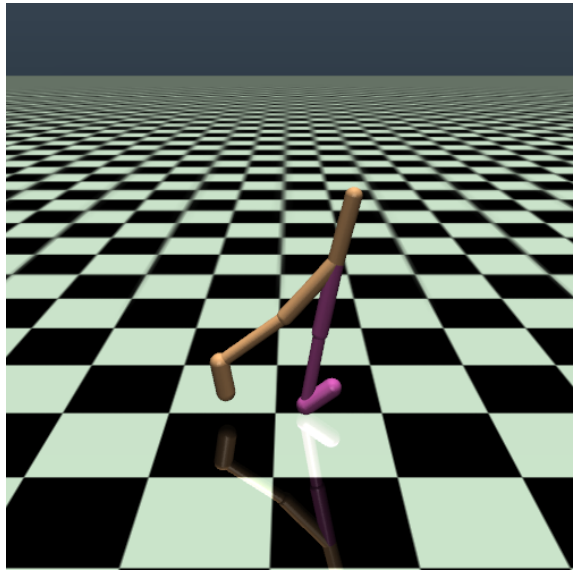
- The policy can be updated following the ***Policy Gradient***:

$$\nabla J(\pi) = \nabla Q(s,a)\Big|_{\mathrm{a}=\pi(s)} \nabla \pi(s)$$

$$\pi = \pi + \alpha \nabla J(\pi)$$

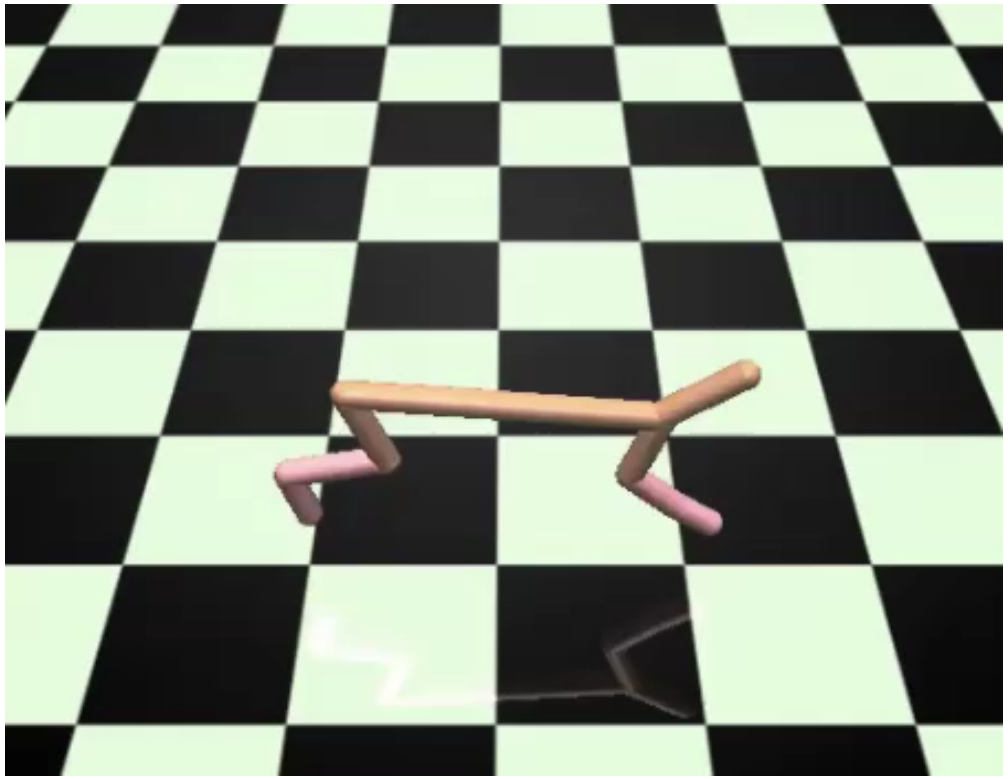
How to measure Deep RL for robots?

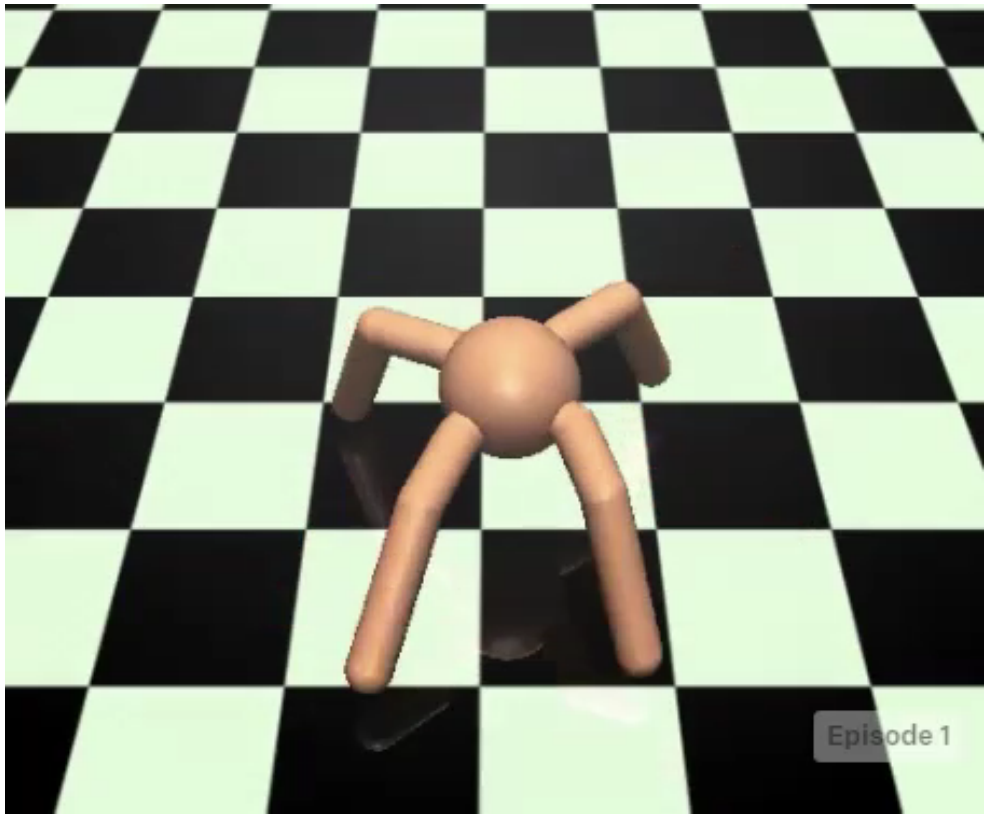
- Standard “robot-like” evaluation is OpenAI Gym

- Simple interface for a suite of benchmark tasks to measure progress

Learning Progress





Peter Henderson's story

- Began working on ***Method A***, the most promising in Sept 2016
- Same authors published ***Method B*** in Spring 2017
  - Paper results showed order of magnitude greater performance. "***Method A*** is unable to solve the task, which our new idea tackles."

Peter Henderson's story

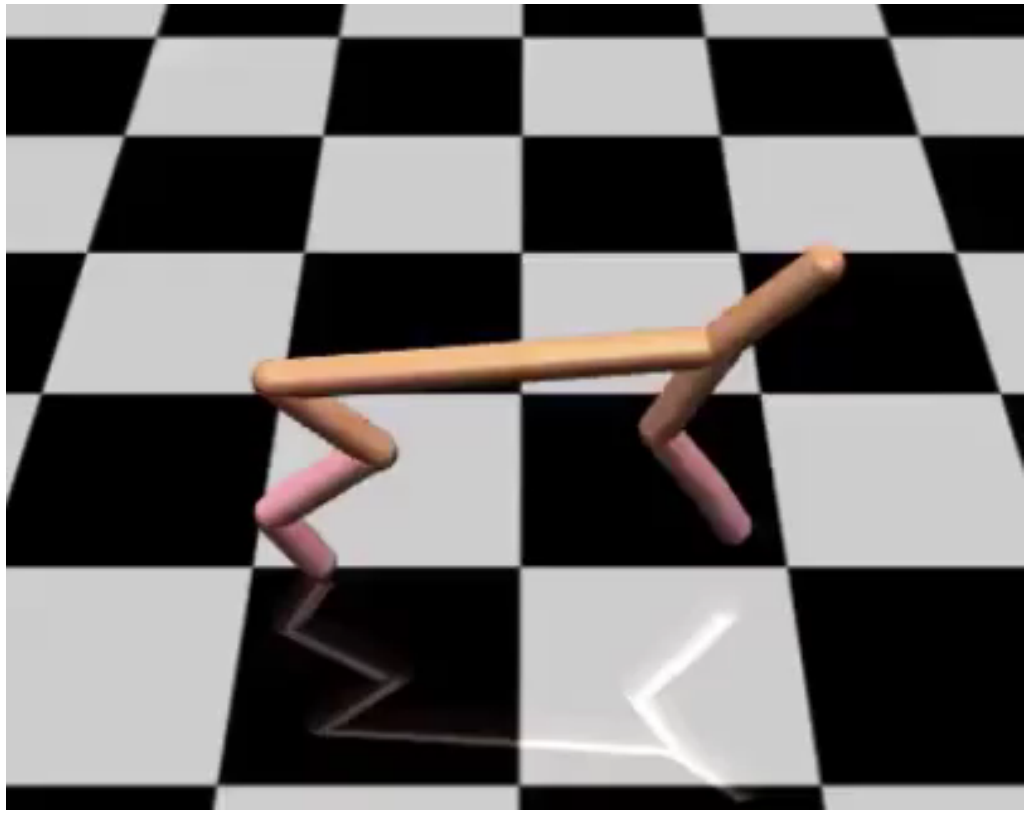
- Began working on ***Method A***, the most promising in Sept 2016

- Same authors published ***Method B*** in Spring 2017:
  - Paper results showed order of magnitude greater performance. "***Method A*** is unable to solve the task, which our new idea tackles."
  - News articles announce "RL is approaching solved"

Peter Henderson's story

- Began working on ***Method A***, the most promising in Sept 2016
- Same authors published ***Method B*** in Spring 2017
- Peter's own code showed ***Method A*** and ***Method B*** performing equally
  - Better than the paper in both cases, through only trivial code fixes…

This was the performance at the time




© Yi Liu/Caters News

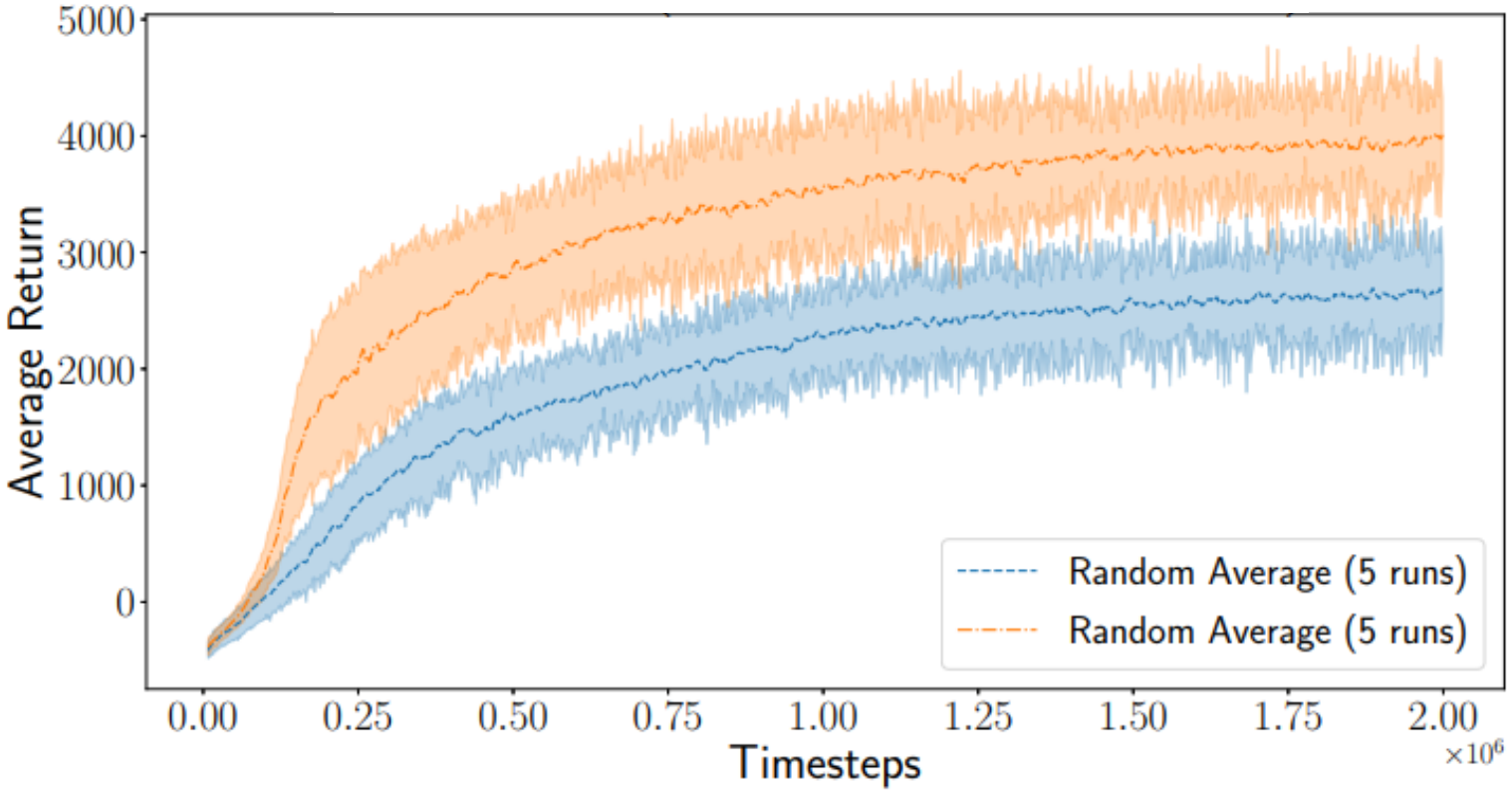
Wagstaff, “ML that Matters”, ICML 2012

“Much of current machine learning (ML) research has lost its connection to problems of import to the larger world of science and society. From this perspective, there exist glaring limitations in the data sets we investigate, the metrics we employ for evaluation, and the degree to which results are communicated back to their originating domains. What changes are needed to how we conduct research to increase the impact that ML has?”

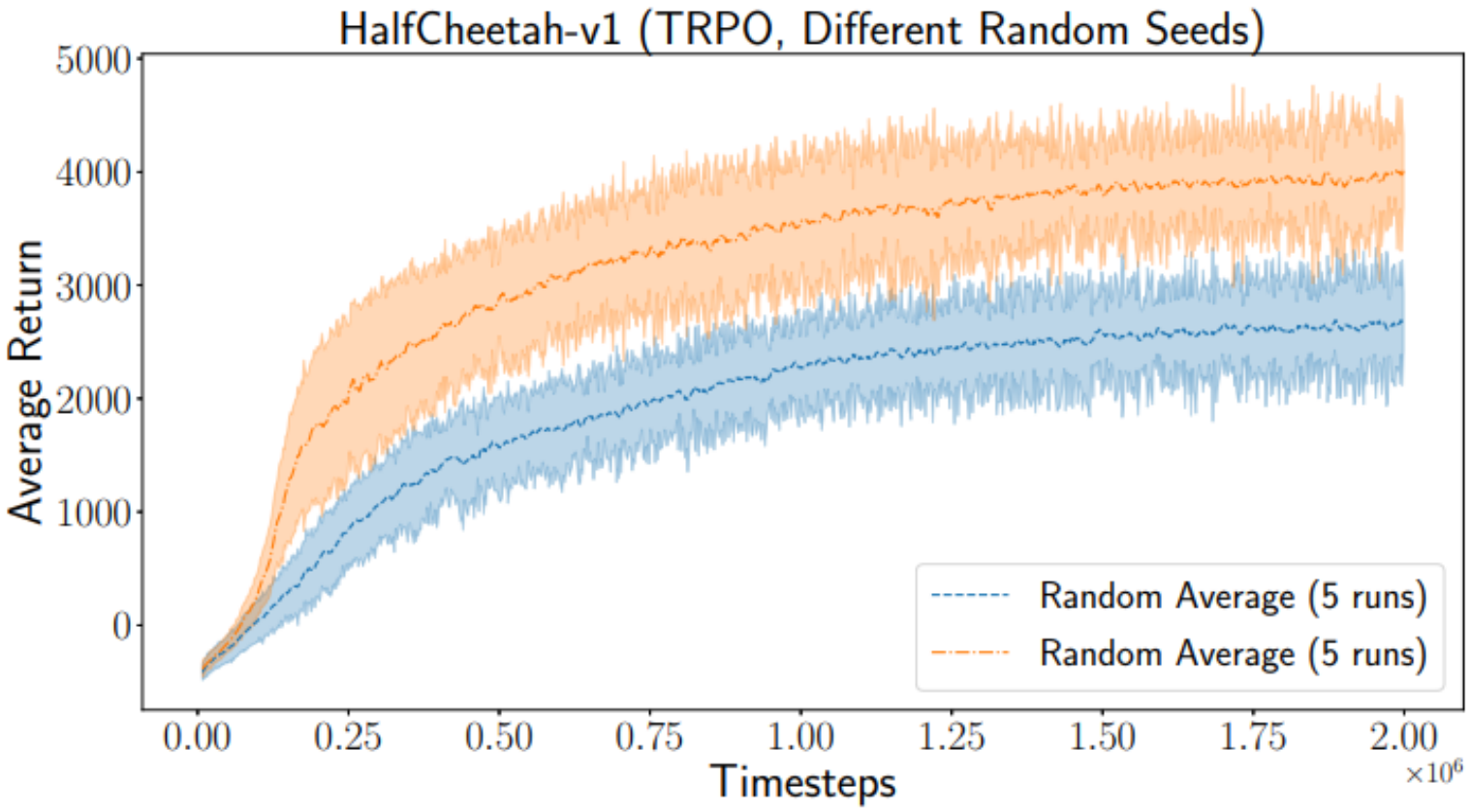
Henderson et al, “Deep RL that Matters”, AAAI 2018

- What factors lead to lack of reproducibility in Deep RL approaches?
- Which of the existing method comparisons are meaningful? Develop actual statistical testing for A vs B RL comparisons.
- Are there “right” ways to do experiments in this domain?

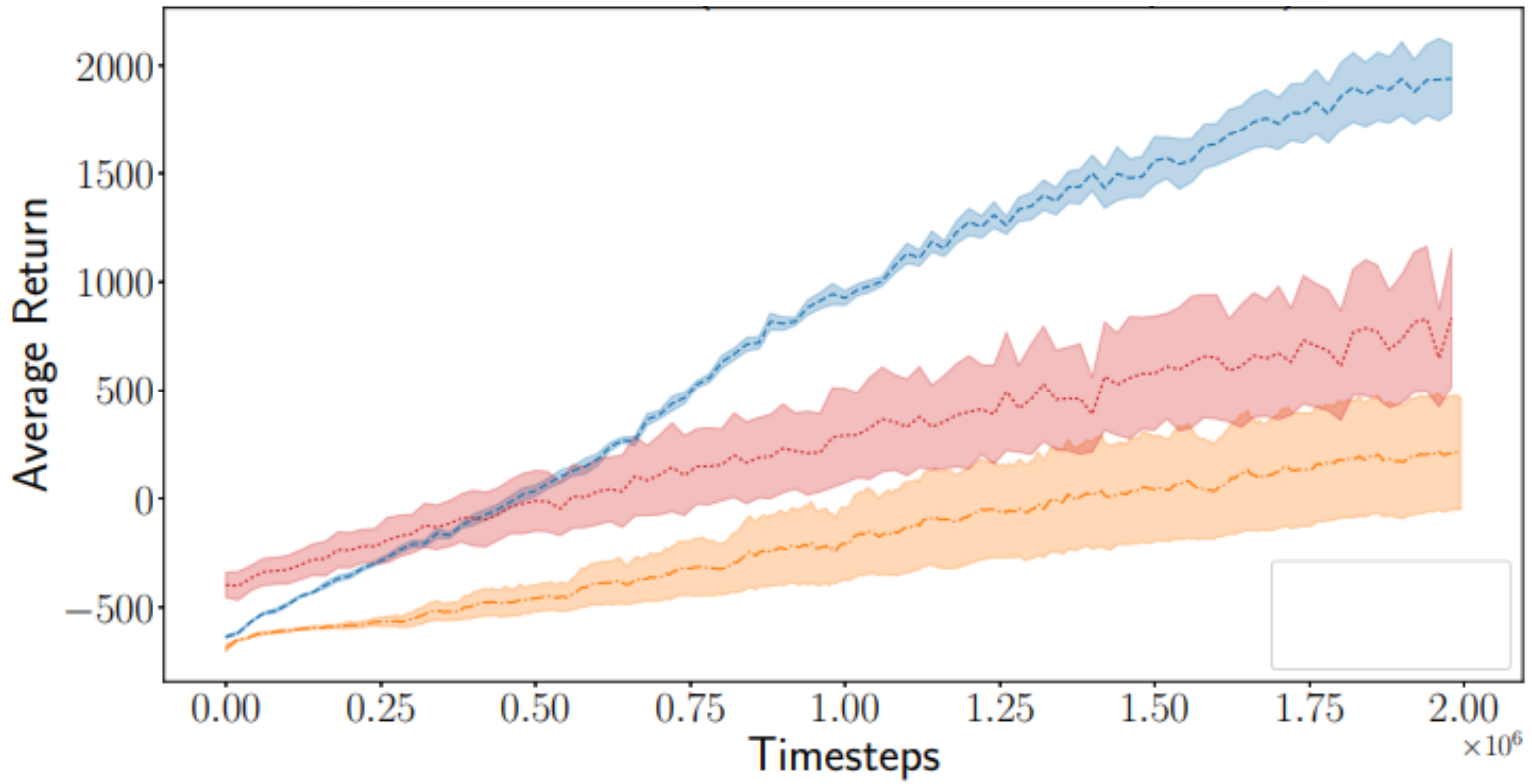
A significant improvement!



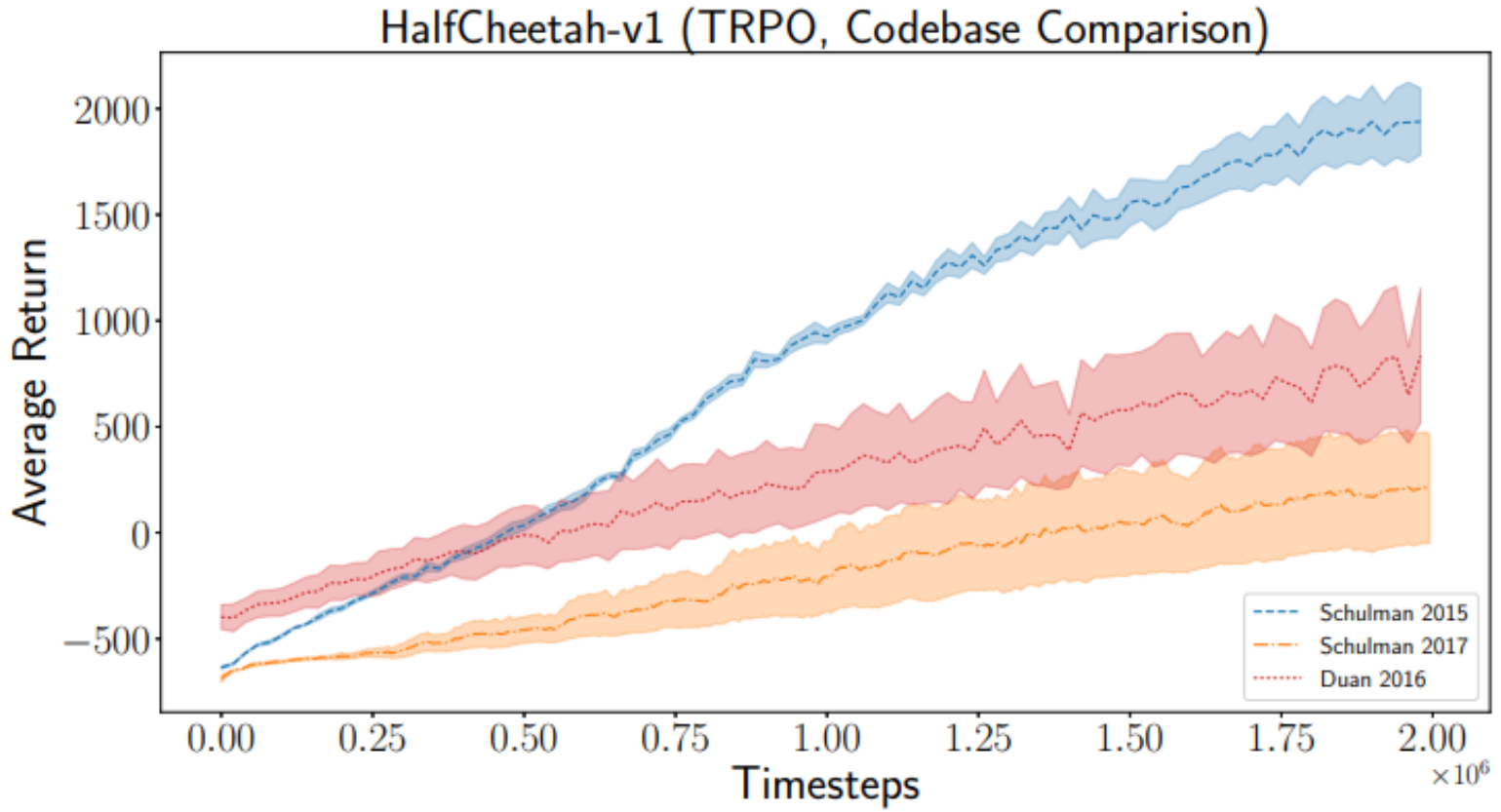
This is the same method run twice, only changing the random seeds…



Another winner!



These are implementations of the same paper, mostly by the same coders…



Method choice rarely outweighs “nuisance” factors

- Random seeds:
  - Initial policy weights
  - Exploration actions
- Hyper parameters:
  - Batch size
  - Learning rate
  - Reward scaling
- Network architecture, activations, normalization

We know these shouldn't matter!

- Reporting the ***max over many trials*** while hiding the mean and variance
- Allowing an optimizer to select the ***best random seed***
- Reporting the score of only a ***single random trial*** or a very small set of repetitions

Outcomes

- We recommend experimental practices that have started to change how methods are evaluated.
  - More transparency about how repeated trials and seeds are handled
  - Must show tables with hyper parameter values
  - Must not optimize hyper parameters for each individual task. This is the RL equivalent of training on the test set
  - It is not OK to hide variance – this must be addressed in the techniques
- ML conferences now request “Reproducibility Checklist” as standard publication practice. Motivated by our findings and similar.

Problem: Many papers state “Our experiments follow the recommendations of [Henderson2018].”

Fewer papers actually follow the recommendations, and it takes significant reviewer effort to verify.

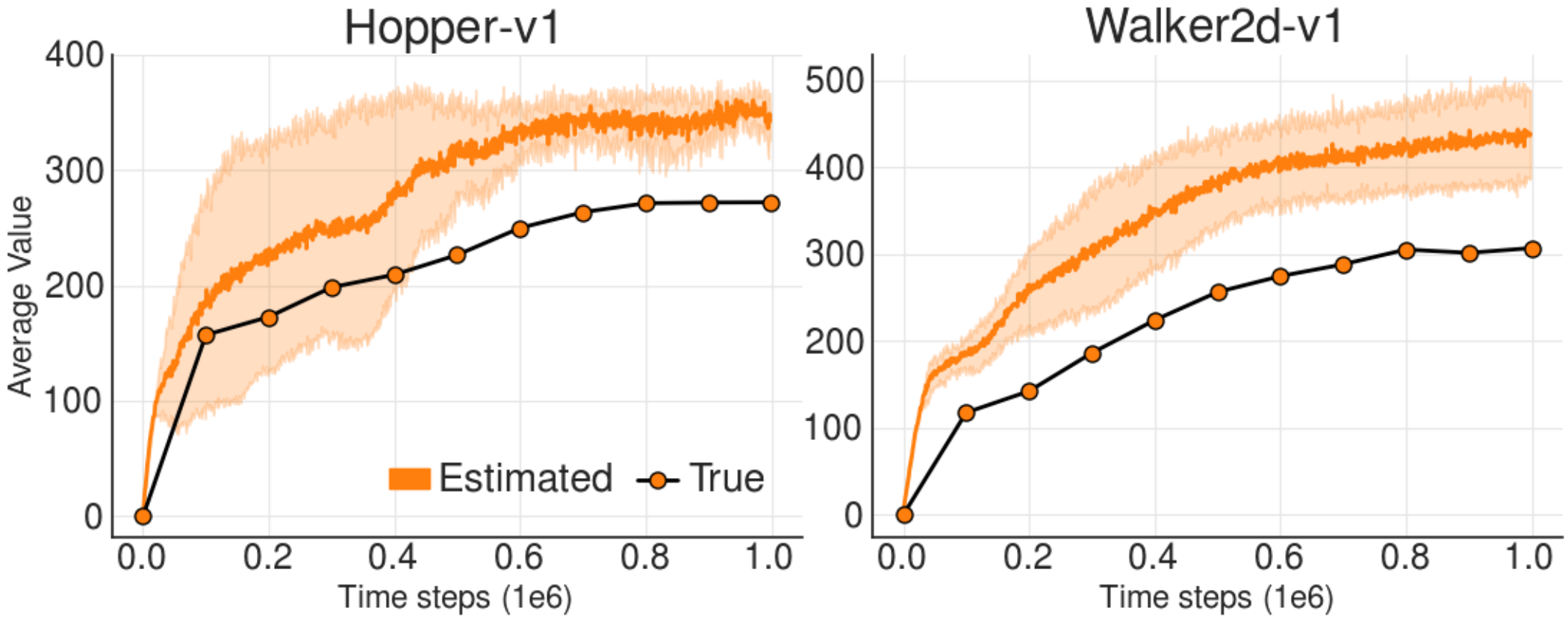
Peter is carrying out a follow-on study using to confirm this.

How many of these same practices and findings apply to robotics experiments?


Position 2
real time
autonomous execution

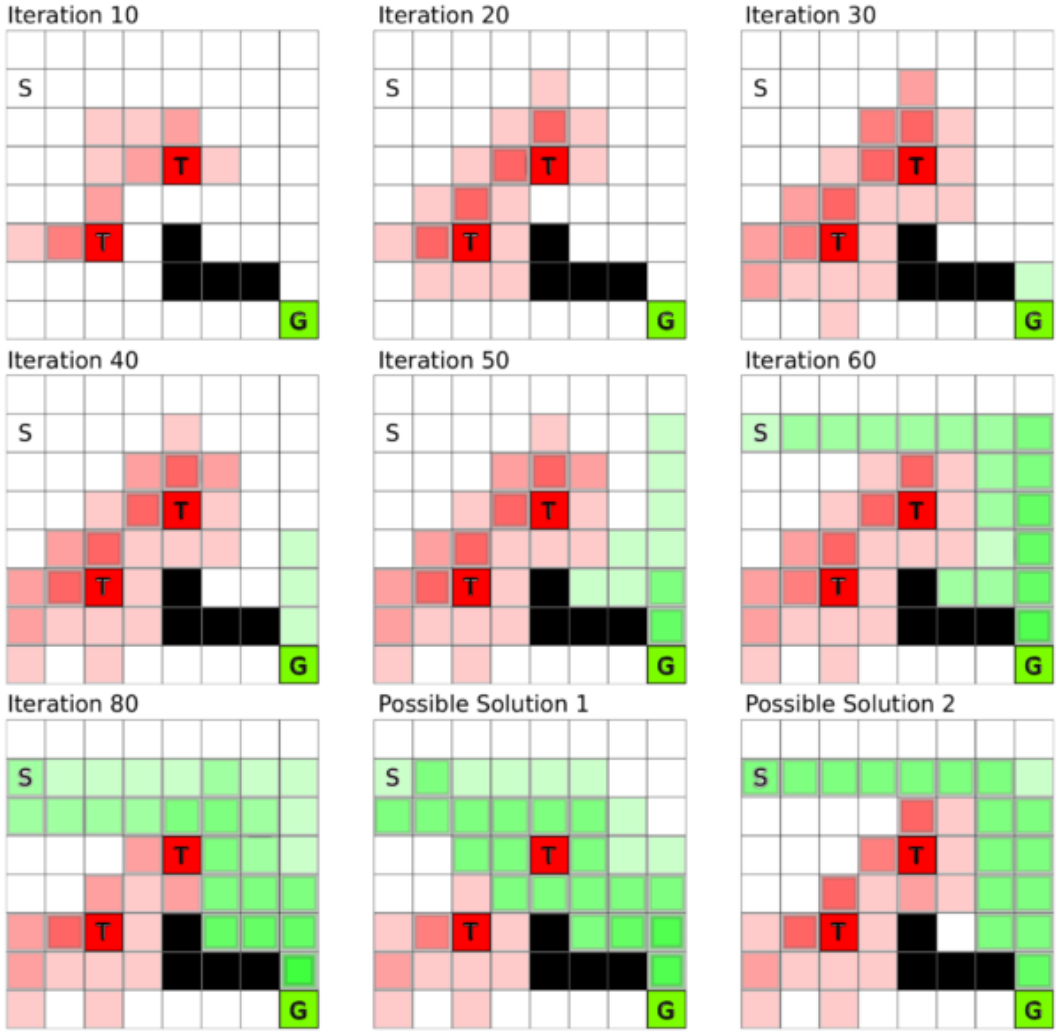
Part 2: Improving DRL stability by diving deep on value learning

Empirical truth: values are overestimated



Why Overestimated when we had proofs?

- [Watkins 1989] proves convergence of Q-Learning with infinite data… however…
- This did not account for the types of errors neural networks make.
- It only worked in the discrete action case or for ”linear function approximation”



Credit: https://devblogs.nvidia.com/deep-learning-nutshell-reinforcement-learning/

Continuous State/Action: Actor Critic

- We replace the max operator with a policy π:

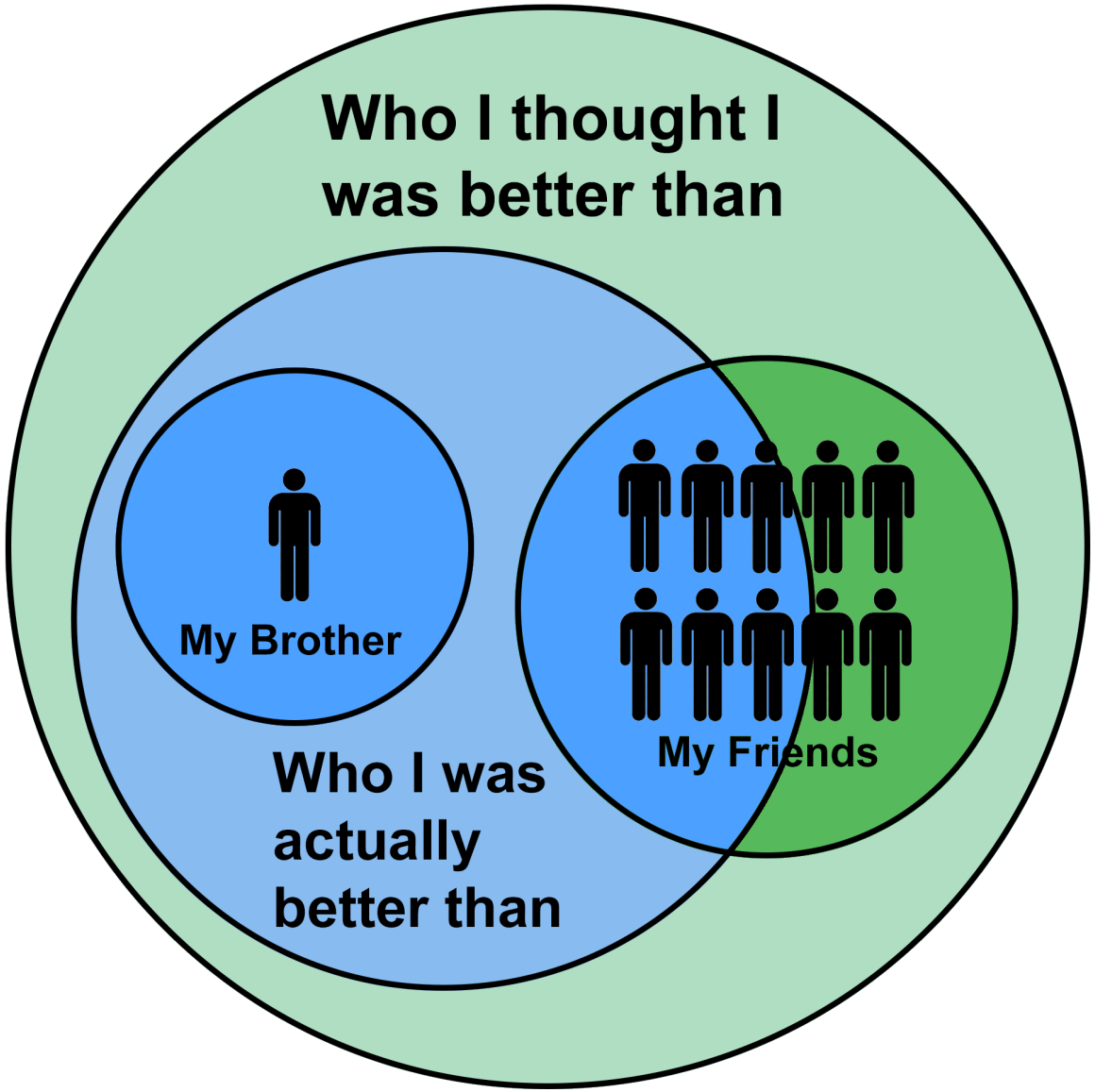
$$Q(s,a) \leftarrow r + \gamma Q'(s',a'), \qquad a' \sim \pi'(s')$$

- The policy can be updated following the ***Policy Gradient***:

$$\nabla J(\pi) = \nabla Q(s,a)\Big|_{\mathrm{a}=\pi(s)} \nabla \pi(s)$$

$$\pi = \pi + \alpha \nabla J(\pi)$$

Scott’s story


Who I thought I was better than
My Brother
My Friends
Who I was actually better than

Goodheart's Law

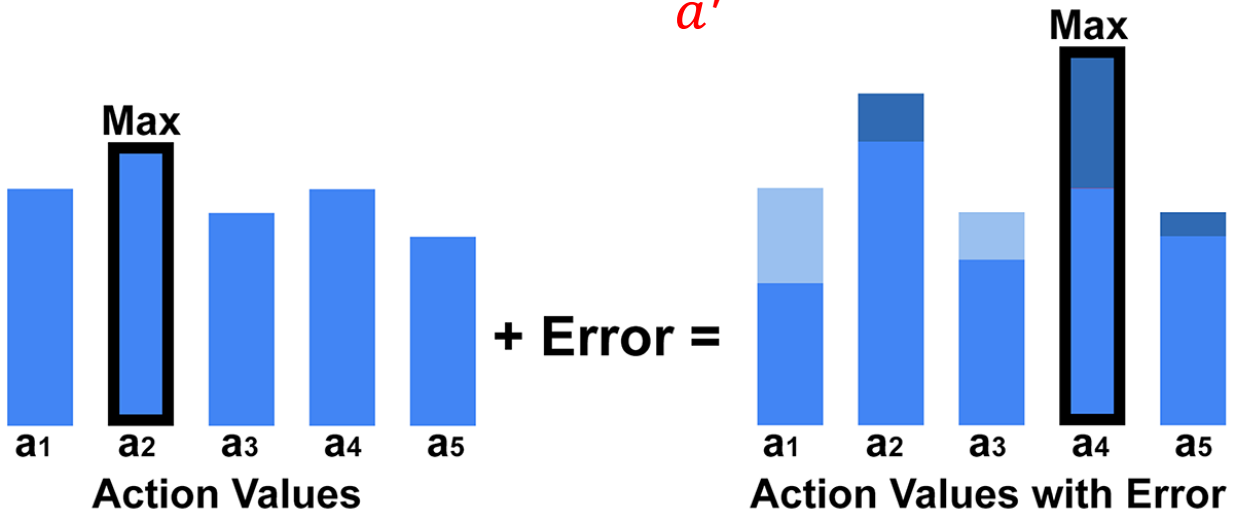
When a measure becomes a target, it ceases to be a good measure.

High interest rate means growing economy?
Most lines of code means best programmer?
Highest RL value estimate means best action?

→ Only if we do not allow "gaming the system"

- Stare at the update rule of Q-learning with discrete actions. Remember Q is a neural network that starts with random weights:

$$Q(s,a) \leftarrow r + \gamma \max_{a'} Q(s',a')$$



- With errors ϵ overestimation is likely to occur (Thrun & Schwartz, 1993) (even if error is initially unbiased!)

$$\mathrm{E}\left[\max_{a'}(Q(s',a') + \epsilon)\right] > \mathrm{E}\left[\max_{a'} Q(s',a')\right]$$

Actor Critic is also expected to over-estimate

- Recall the Policy Gradient learning updates:

$$Q(s,a) \leftarrow r + \gamma Q'(s',a'), \qquad a' \sim \pi'(s')$$

$$\nabla J(\pi) = \nabla Q(s,a)\Big|_{\mathrm{a}=\pi(s)} \nabla \pi(s)$$

$$\pi = \pi + \alpha \nabla J(\pi)$$

- We are the first to show the gradient is expected to choose the an action that over-estimates when Q has noise

Double Q-Learning: Known for discrete MDP

- A strategy for reducing overestimation bias is to use two independent estimates of the value function (van Hasselt, 2010):

$$Q_1 \leftarrow r + \gamma Q_2'(s', \pi_1') \qquad \text{(Double Q-learning)}$$
$$Q_2 \leftarrow r + \gamma Q_1'(s', \pi_2')$$

- In Double DQN, the target network Q_1' is used as one of the estimates (van Hasselt et al., 2015):

$$Q_1 \leftarrow r + \gamma Q_1'(s', \pi_1) \qquad \text{(Double DQN)}$$

Clipped Double Q-learning

- We propose a novel learning target, Clipped Double Q-learning:

$$Q_1 \leftarrow r + \gamma \min(Q_1'(s', \pi_1'), Q_2'(s', \pi_1'))$$

- If Q_1 generally overestimates, we can upper-bound the less biased Q_2 by the biased Q_1

- This eliminates the situation where $Q_2 > Q_1 > \mathrm{Q}_{\mathrm{true}}$

Clipped Double Q-learning

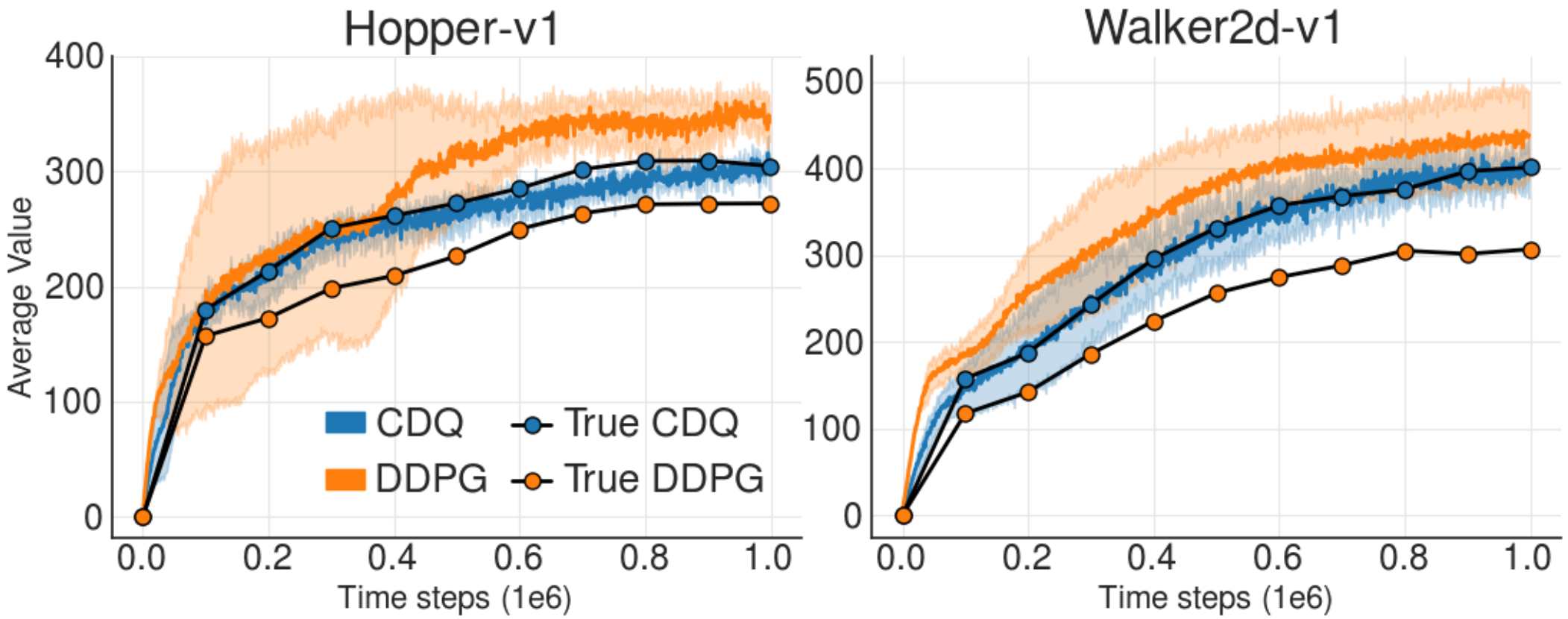
- We propose a novel learning target, Clipped Double Q-learning:

$$Q_1 \leftarrow r + \gamma \min(\underbrace{Q_1'(s', \pi_1')}_{\text{Standard}}, \underbrace{Q_2'(s', \pi_1')}_{\text{Double}})$$

- If Q_1 generally overestimates, we can upper-bound the less biased Q_2 by the biased Q_1

- This eliminates the situation where $Q_2 > Q_1 > \mathrm{Q}_{\mathrm{true}}$

Clipped Double Q-learning



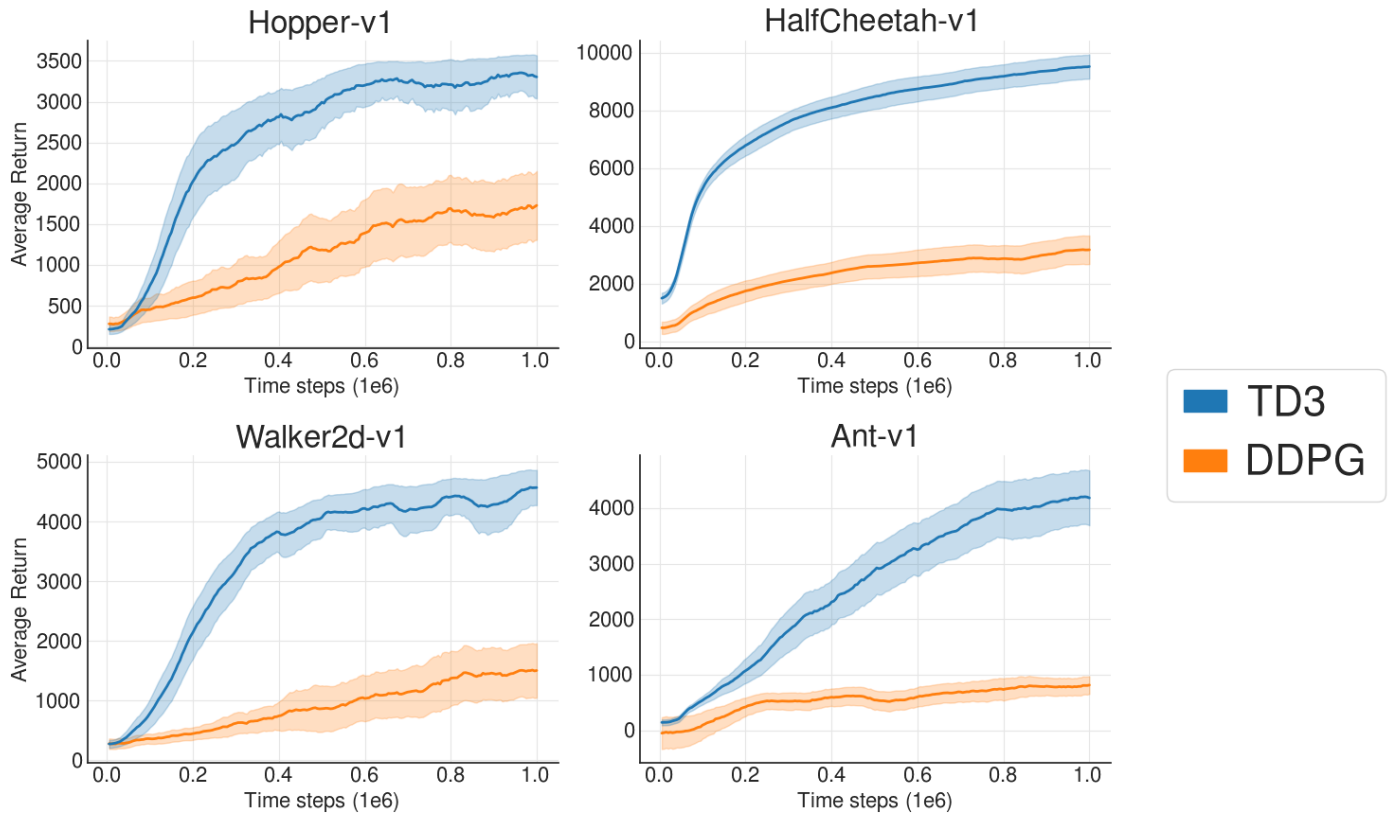
Twin Delayed DDPG (TD3)

- Clipped update and several other modifications are added to DDPG to form Twin Delayed Deep Deterministic policy gradient (TD3).

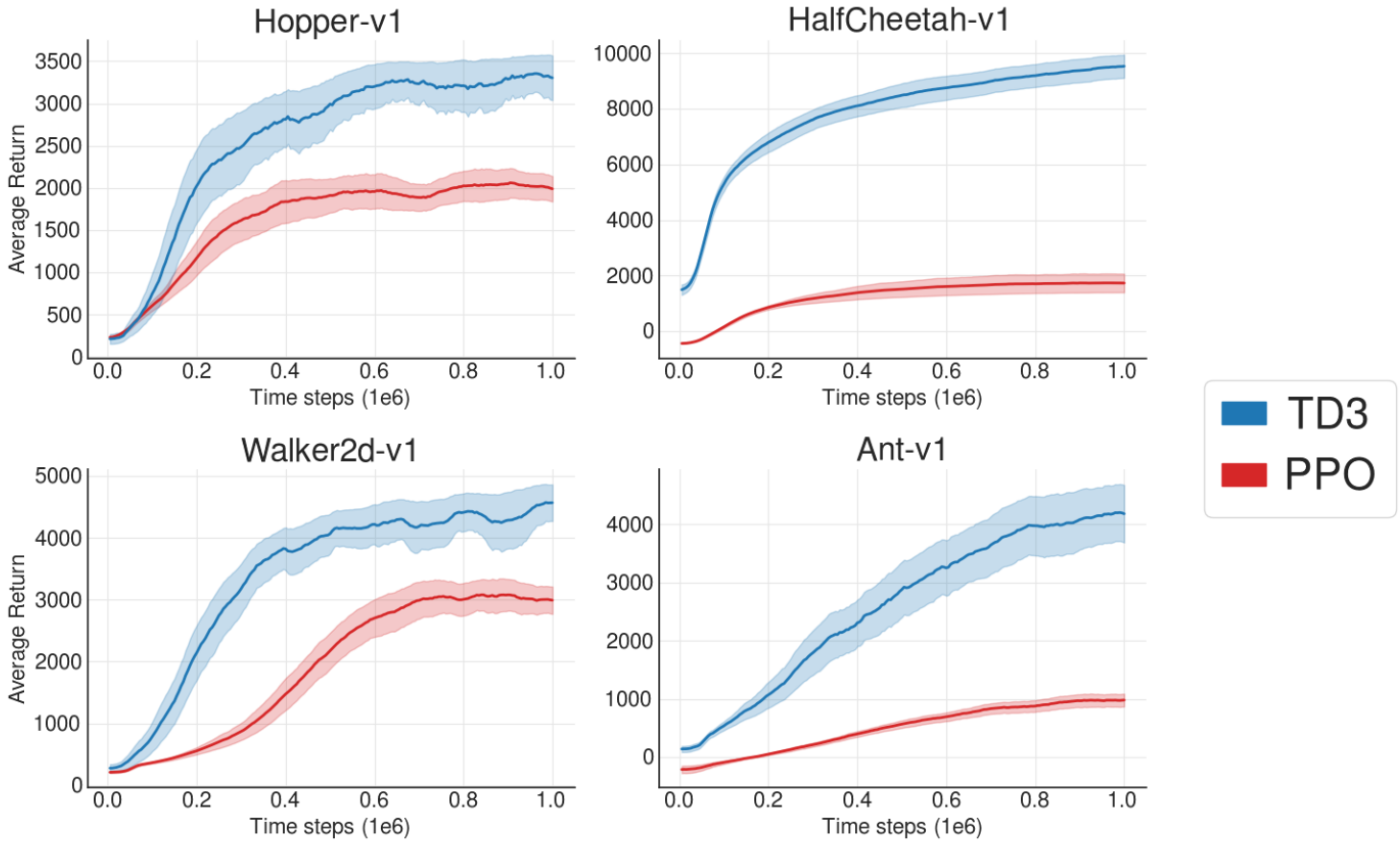
$$Q_1, Q_2 \leftarrow r + \gamma \min\left(Q_1(s', a'), Q_2(s', a')\right),$$
$$a' \sim \text{clip}(\pi(s') + \epsilon, -c, c)$$

- The actor and both target networks are updated every d time steps.

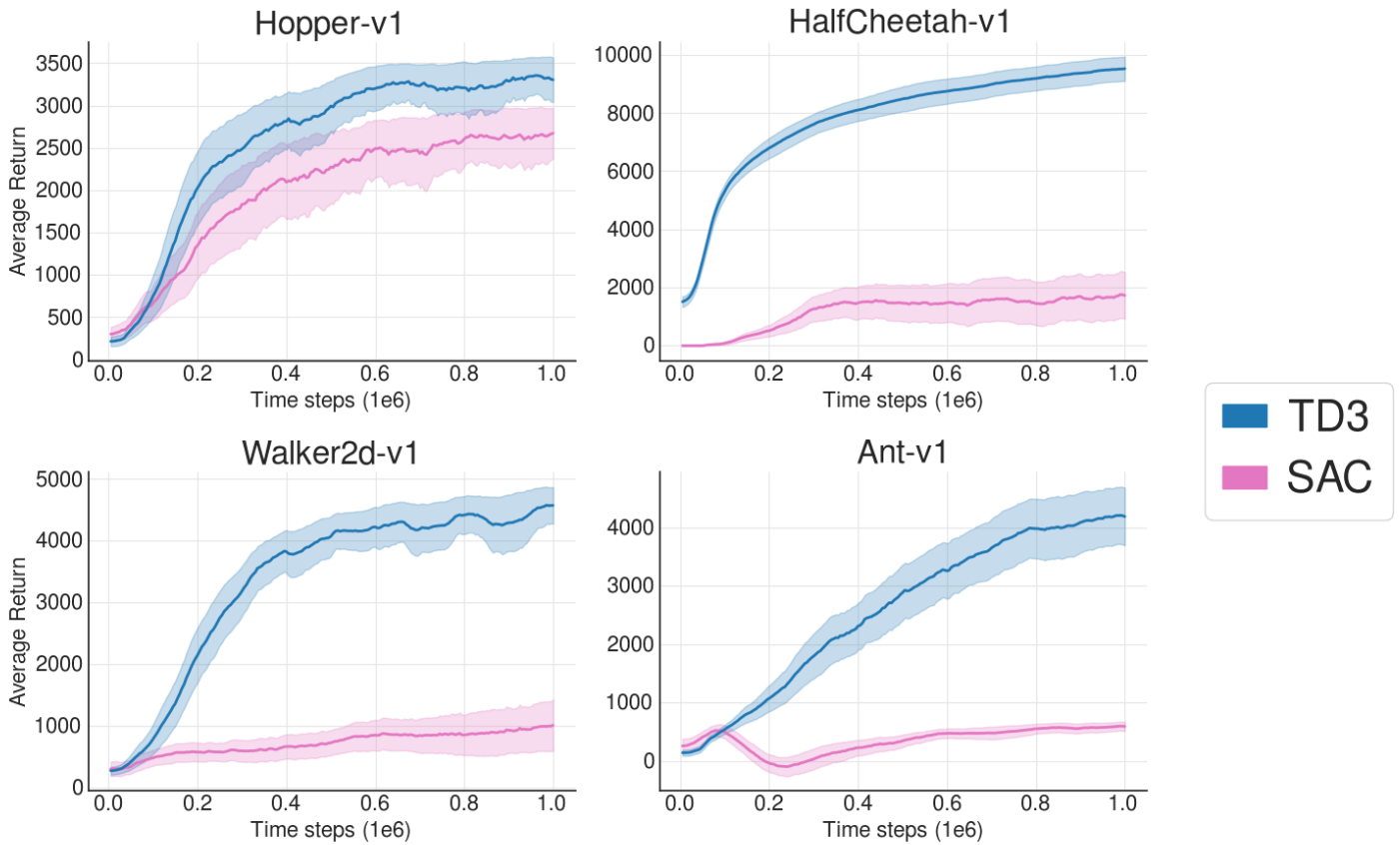
TD3 vs DDPG


Hopper-v1
HalfCheetah-v1
Walker2d-v1
Ant-v1
Average Return
Time steps (1e6)
TD3
DDPG

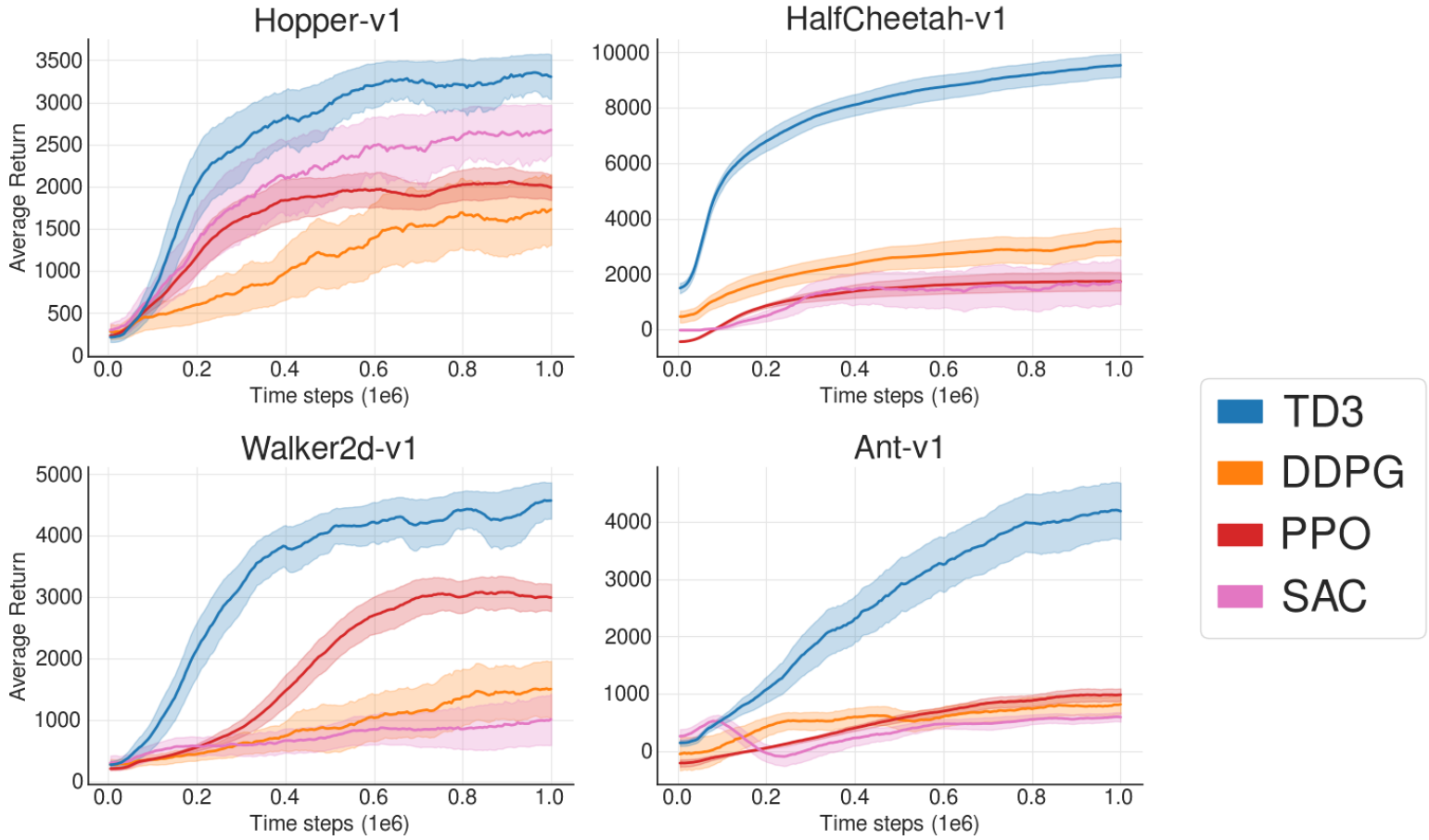
TD3 vs PPO


Hopper-v1
HalfCheetah-v1
Walker2d-v1
Ant-v1
Average Return
Time steps (1e6)
TD3
PPO

TD3 vs SAC* (*prior to TD3-SAC)



Results


Hopper-v1
HalfCheetah-v1
Walker2d-v1
Ant-v1
Average Return
Time steps (1e6)
TD3
DDPG
PPO
SAC

Take-home

1. You will overfit to the metric you optimize with respect to.
   - Overfitting occurs in actor-critic methods.
   - Pessimism in the value function can improve performance.

2. A new perspective on the importance of target networks:
   - Target networks reduce the buildup of error, which is critical when combined with a recursive maximization operation.

3. Updating the policy less can actually improve performance.

- Use TD3! https://github.com/sfujim/TD3
  - Under 200 lines of PyTorch code with comments.

Scott Fujimoto

Herke van Hoof

Scott Fujimoto, Herke van Hoof and David Meger. Addressing Function Approximation Error in Actor-Critic Methods. ICML 2018.

Conclusions

Improving continuous value estimates

The Batch RL problem

Batch-Constrained deep Q-learning (BCQ)

- Value-based methods give the strongest theoretical guarantees and show such power in discrete tasks, they should be our hammer
- Big improvements possible by acknowledging the practical considerations of real RL use, especially for continuous tasks
- Quite promising to combine these ideas with transfer, Sim2Real, etc!
- All honor to these collaborators, especially ***Scott Fujimoto***, a current PhD at MRL and MILA

Scott Fujimoto

Herke van Hoof

Doina Precup