IASD M2 at Paris Dauphine

# Deep Reinforcement Learning

## 24: Challenges and Open Problems

Eric Benhamou David Saltiel

# Acknowledgement

These materials are based on the seminal course of Sergey Levine CS285

# Challenges in Deep Reinforcement Learning

# What’s the problem?

Challenges with **core algorithms**:

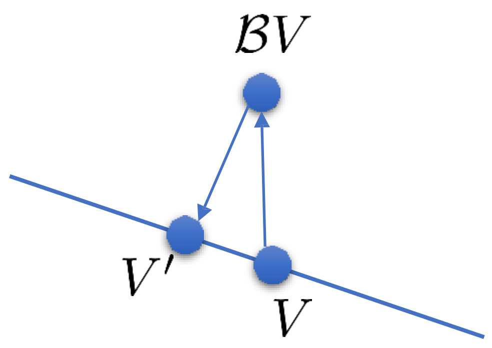
- Stability: does your algorithm converge?
- Efficiency: how long does it take to converge? (how many samples)
- Generalization: after it converges, does it generalize?

Challenges with **assumptions**:

- Is this even the right problem formulation?
- What is the source of *supervision*?

# Stability and hyperparameter tuning

- Devising stable RL algorithms is very hard
- Q-learning/value function estimation
  - Fitted Q/fitted value methods with deep network function estimators are typically not contractions, hence no guarantee of convergence
  - Lots of parameters for stability: target network delay, replay buffer size, clipping, sensitivity to learning rates, etc.
- Policy gradient/likelihood ratio/REINFORCE
  - Very high variance gradient estimator
  - Lots of samples, complex baselines, etc.
  - Parameters: batch size, learning rate, design of baseline
- Model-based RL algorithms
  - Model class and fitting method
  - Optimizing policy w.r.t. model non-trivial due to backpropagation through time
  - More subtle issue: policy tends to *exploit* the model



# The challenge with hyperparameters

- Can’t run hyperparameter sweeps in the real world
  - How representative is your simulator? Usually the answer is “not very”
- Actual sample complexity = time to run algorithm x number of runs to sweep
  - In effect stochastic search + gradient-based optimization
- Can we develop more stable algorithms that are less sensitive to hyperparameters?

# What can we do?

- Algorithms with favorable improvement and convergence properties
  - Trust region policy optimization [Schulman et al. '16]
  - Safe reinforcement learning, High-confidence policy improvement [Thomas '15]
- Algorithms that adaptively adjust parameters
  - Q-Prop [Gu et al. '17]: adaptively adjust strength of control variate/baseline

- More research needed here!
- Not great for beating benchmarks, but absolutely essential to make RL a viable tool for real-world problems

# Sample Complexity

gradient-free methods
(e.g. NES, CMA, etc.)

10x

fully online methods
(e.g. A3C)

10x

policy gradient methods
(e.g. TRPO)

10x

replay buffer value estimation methods
(Q-learning, DDPG, NAF, SAC, etc.)

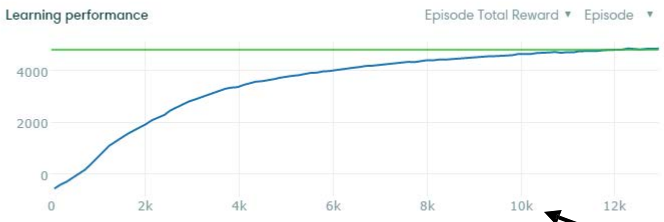
model-based deep RL
(e.g. PETS, guided policy search)

model-based "shallow" RL
(e.g. PILCO)

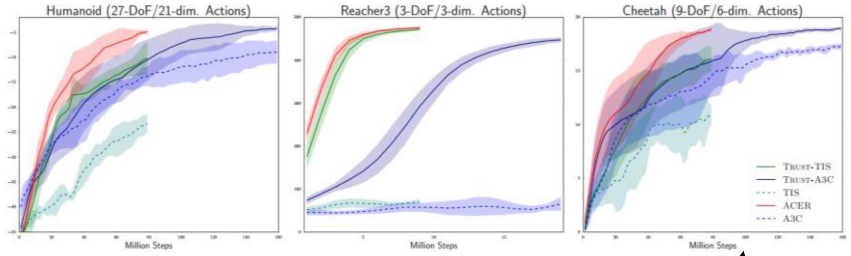
**Evolution Strategies as a Scalable Alternative to Reinforcement Learning**

Tim Salimans[1] Jonathan Ho[1] Xi Chen[1] Ilya Sutskever[1]

half-cheetah (slightly different version)




TRPO+GAE (Schulman et al. '16)

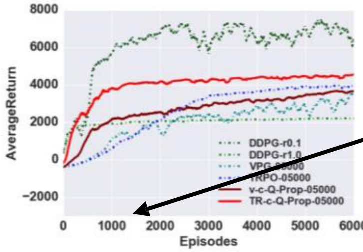



Wang et al. '17

100,000,000 steps
(100,000 episodes)
(~ 15 days real time)

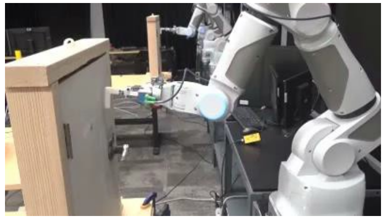
10,000,000 steps
(10,000 episodes)
(~ 1.5 days real time)

half-cheetah

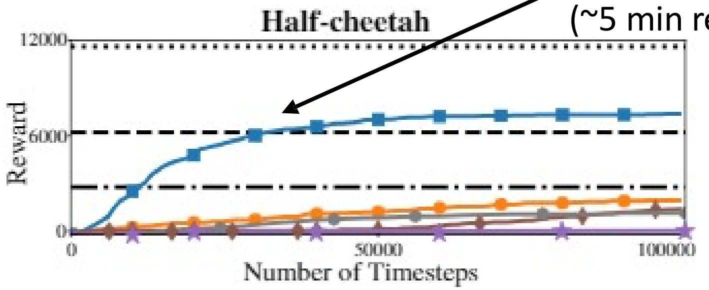



Gu et al. '16

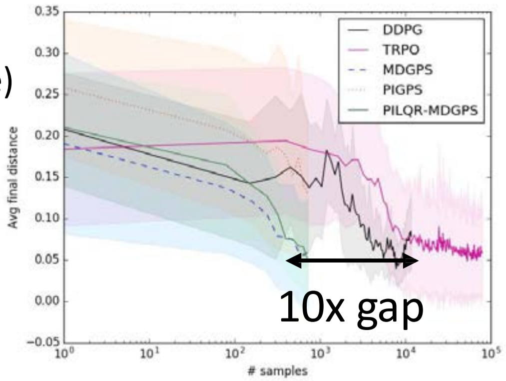
1,000,000 steps
(1,000 episodes)
(~3 hours real time)



30,000 steps
(30 episodes)
(~5 min real time)




Chua et a. '18: Deep Reinforcement Learning in a Handful of Trials




about 20 minutes of experience on a real robot

Chebotar et al. '17 (note log scale)

# The challenge with sample complexity

- Need to wait for a long time for your homework to finish running
- Real-world learning becomes difficult or impractical
- Precludes the use of expensive, high-fidelity simulators
- Limits applicability to real-world problems

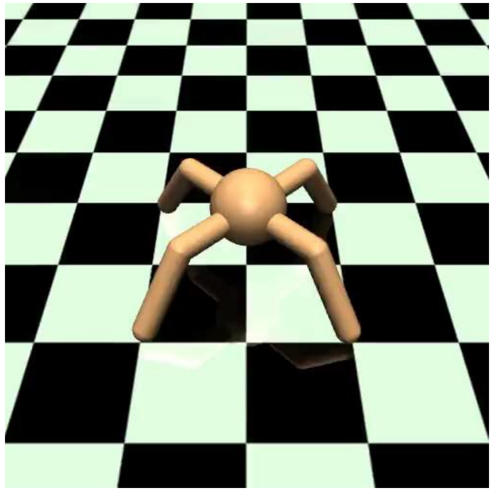

# What can we do?

- Better model-based RL algorithms
- Design faster algorithms
  - Addressing Function Approximation Error in Actor-Critic Algorithms (Fujimoto et al. ‘18): simple and effective tricks to accelerate DDPG-style algorithms
  - Soft Actor-Critic (Haarnoja et al. ‘18): very efficient maximum entropy RL algorithm
- Reuse prior knowledge to accelerate reinforcement learning
  - RL2: Fast reinforcement learning via slow reinforcement learning (Duan et al. ‘17)
  - Learning to reinforcement learning (Wang et al. ‘17)
  - Model-agnostic meta-learning (Finn et al. ‘17)

# Scaling & Generalization

# Scaling up deep RL & generalization

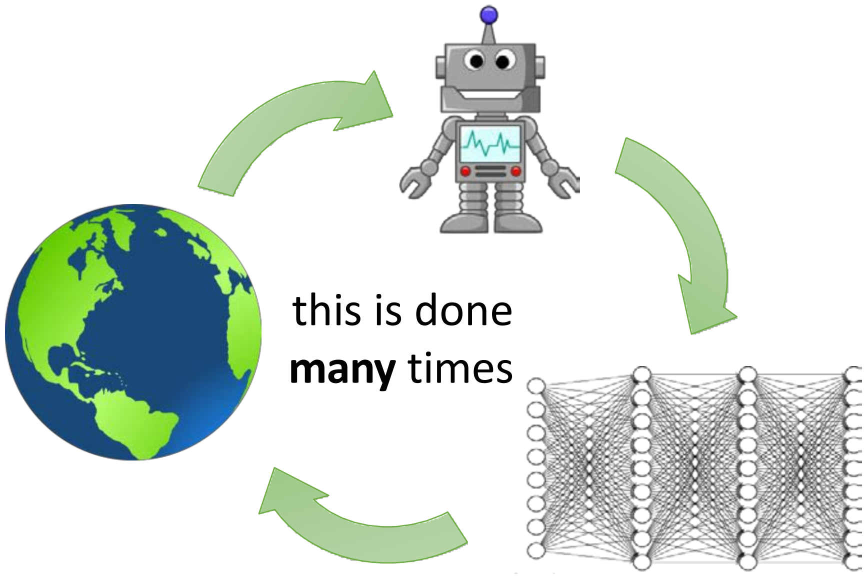

- Large-scale
- Emphasizes diversity
- Evaluated on generalization



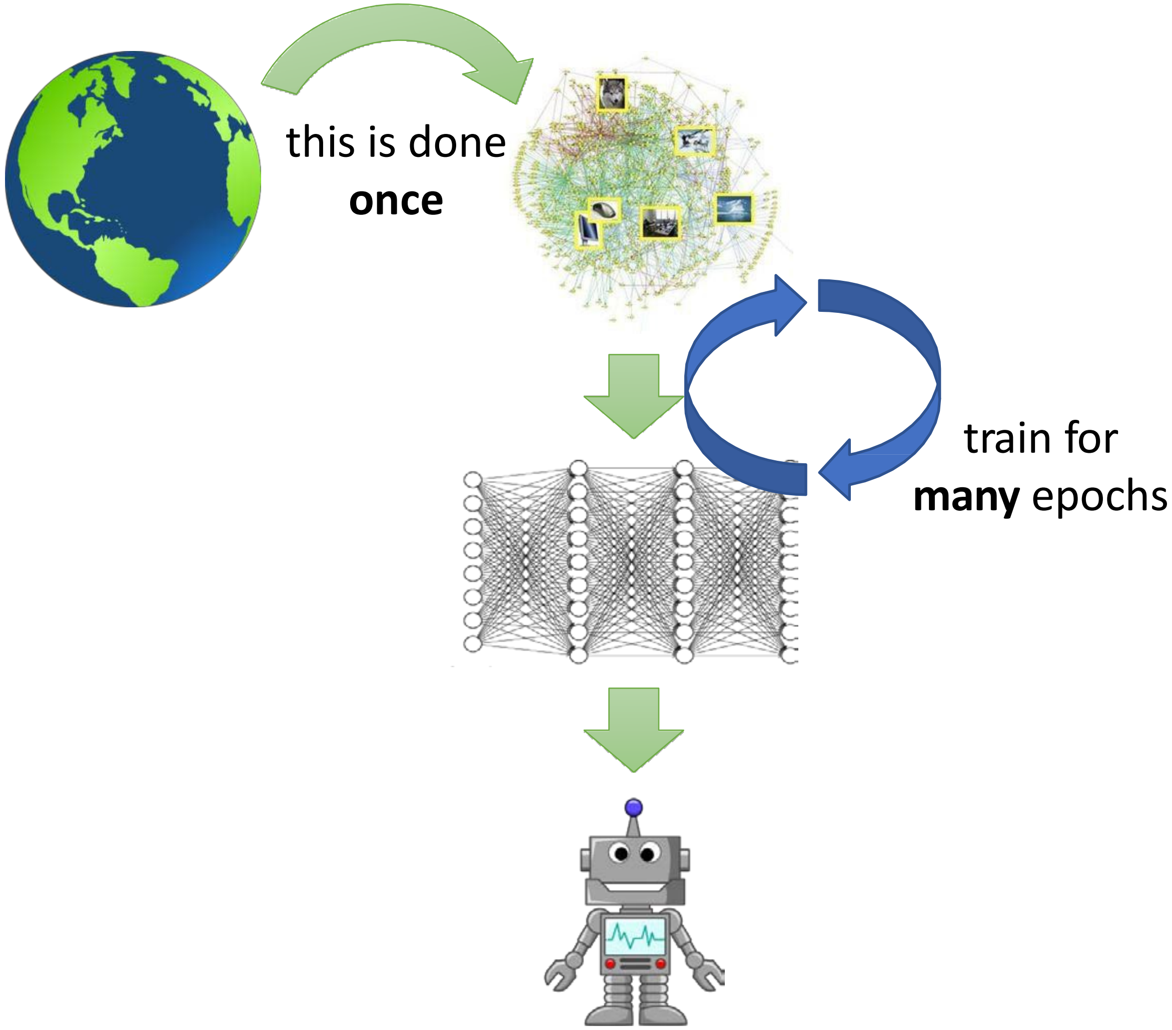
- Small-scale
- Emphasizes mastery
- Evaluated on performance
- Where is the generalization?

# RL has a big problem

## reinforcement learning

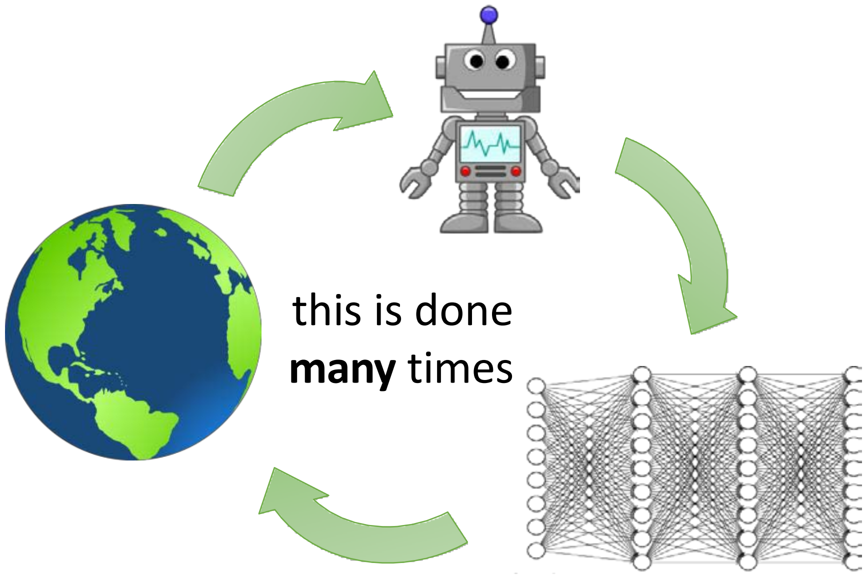



## supervised machine learning

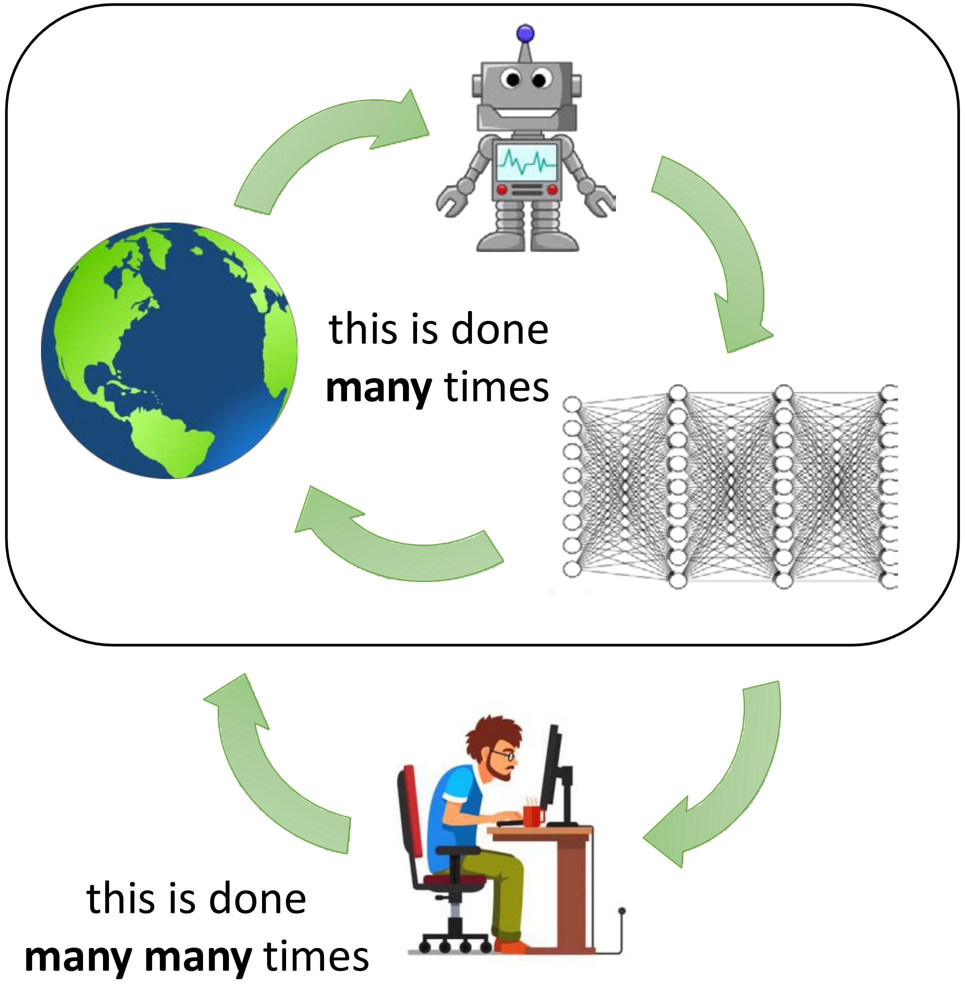


# RL has a big problem

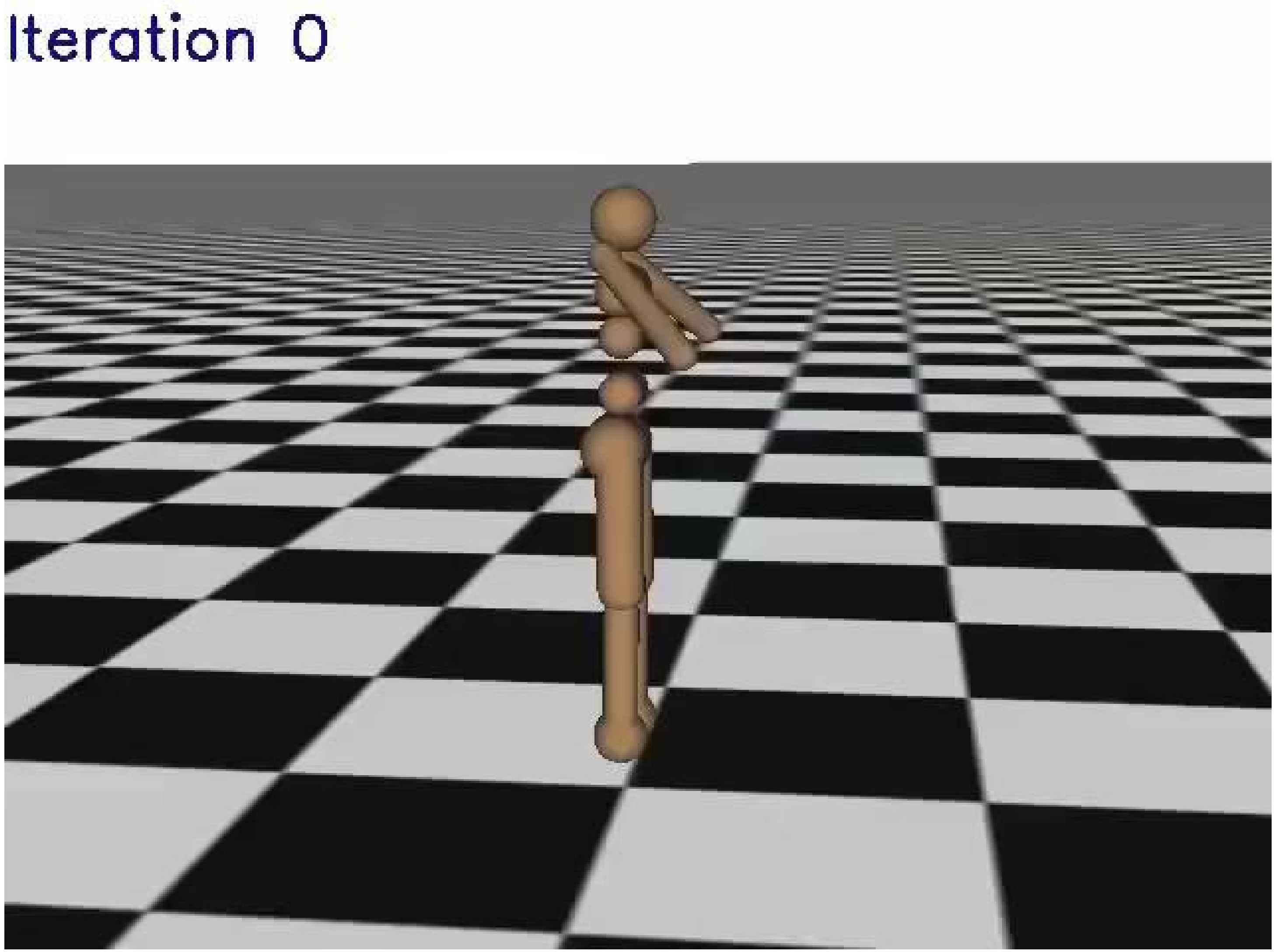
reinforcement learning



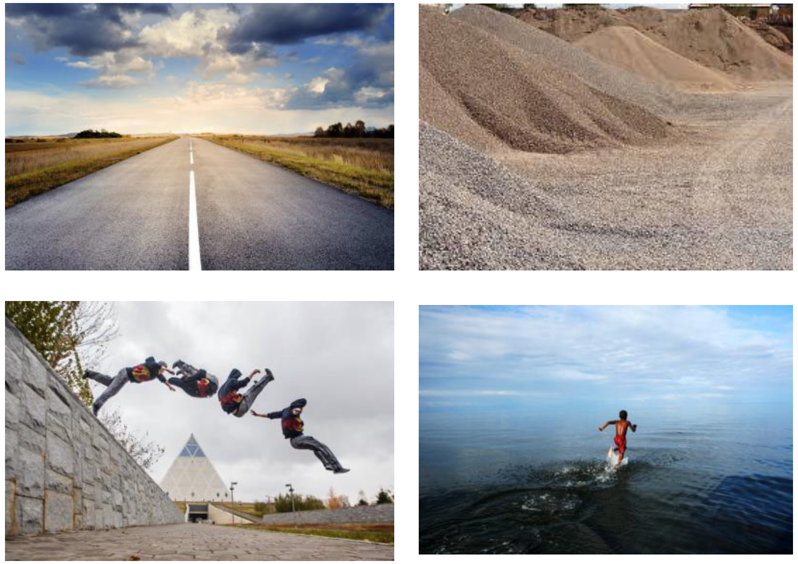

actual reinforcement learning



# How bad is it?




- This is quite cool
- It takes 6 days of real time (if it was real time)
- …to run on an infinite flat plane

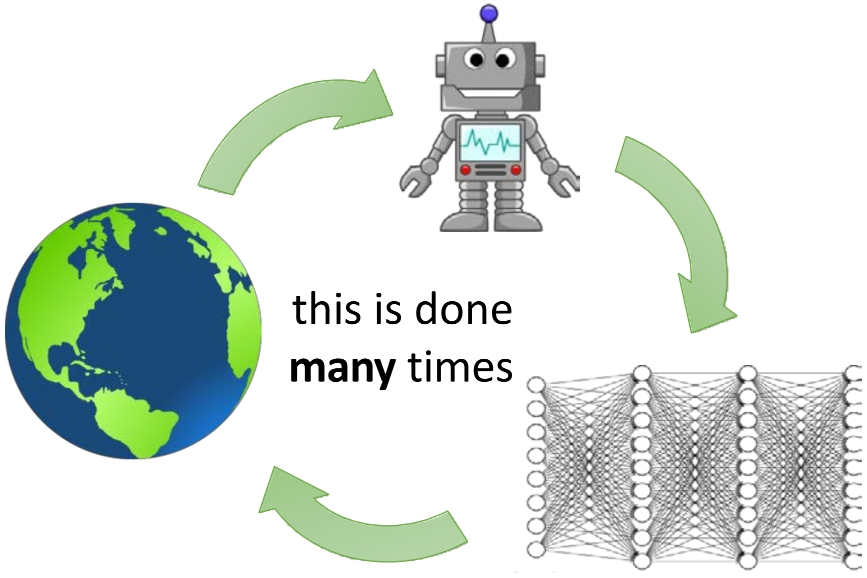


The real world is not so simple!

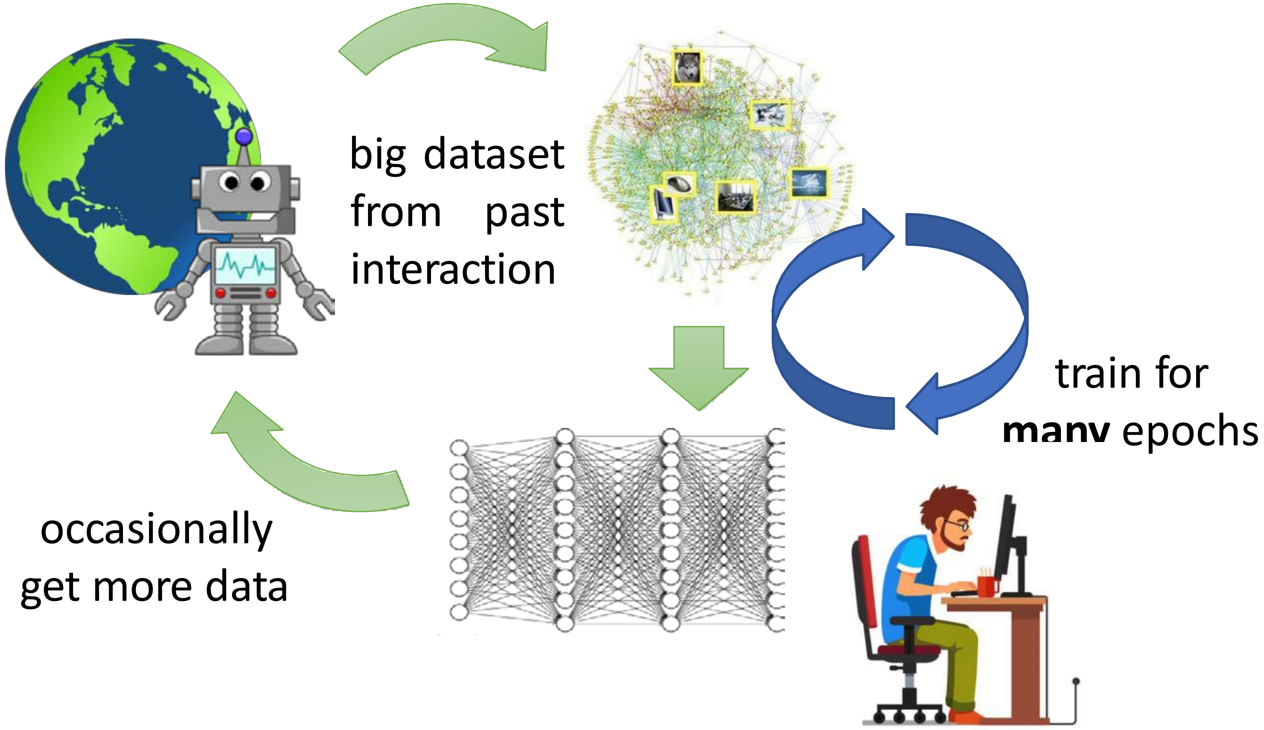
Schulman, Moritz, L., Jordan, Abbeel ’16

# Off-policy RL?

## reinforcement learning




## off-policy reinforcement learning



# Not just robots!

autonomous driving

language & dialogue
(structured prediction)

finance

# What’s the problem?

Challenges with **core algorithms**:

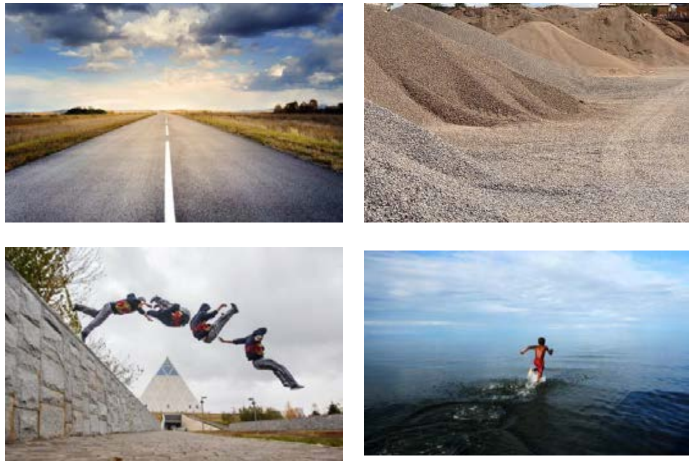
- Stability: does your algorithm converge?
- Efficiency: how long does it take to converge? (how many samples)
- Generalization: after it converges, does it generalize?

Challenges with **assumptions**:

- Is this even the right problem formulation?
- What is the source of *supervision*?

# Problem Formulation

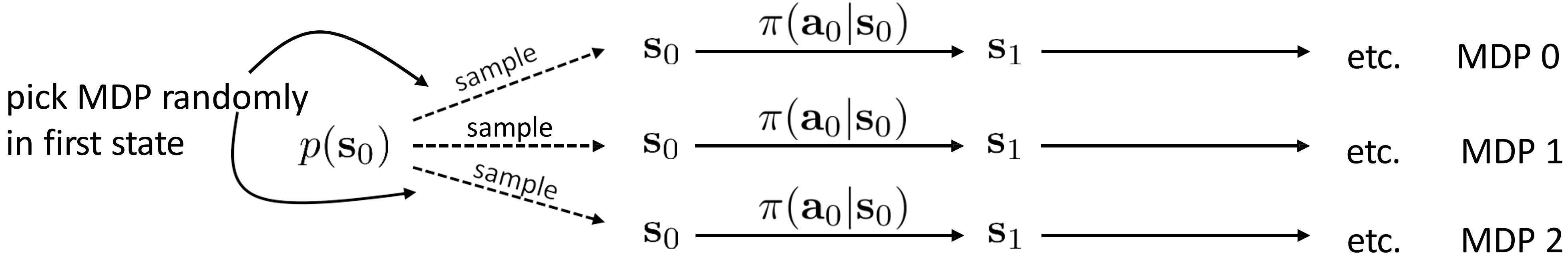
# Single task or multi-task?



The real world is not so simple!

this is where generalization can come from…

maybe doesn't require any new assumption, but might merit additional treatment

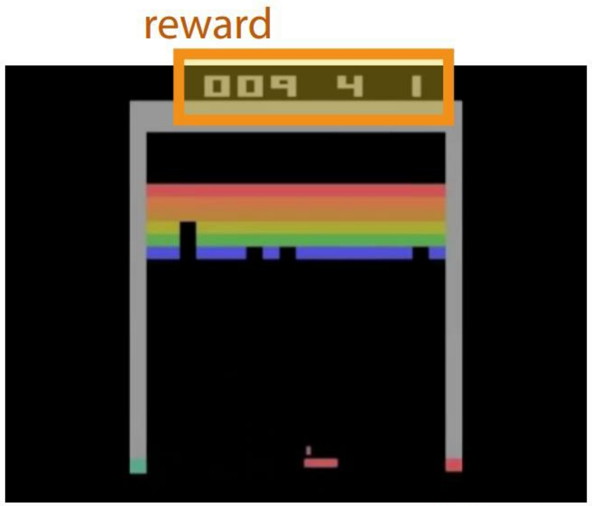


# Generalizing from multi-task learning

- Train on multiple tasks, then try to generalize or finetune
  - Policy distillation (Rusu et al. '15)
  - Actor-mimic (Parisotto et al. '15)
  - Model-agnostic meta-learning (Finn et al. '17)
  - many others…
- Unsupervised or weakly supervised learning of diverse behaviors
  - Stochastic neural networks (Florensa et al. '17)
  - Reinforcement learning with deep energy-based policies (Haarnoja et al. '17)
  - See lecture on unsupervised information-theoretic exploration
  - many others…

# Where does the supervision come from?

- If you want to learn from many different tasks, you need to get those tasks somewhere!
- Learn objectives/rewards from demonstration (inverse reinforcement learning)
- Generate objectives automatically?

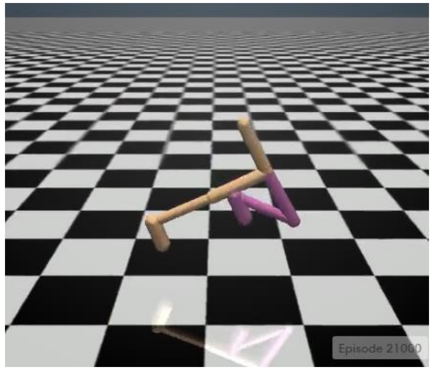



Mnih et al. '15

reinforcement learning agent

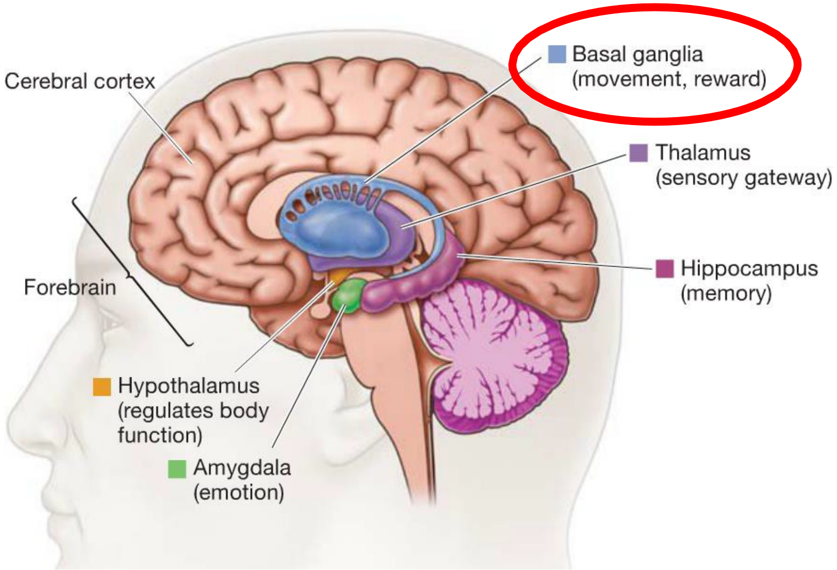
what is the reward?

# What is the role of the reward function?



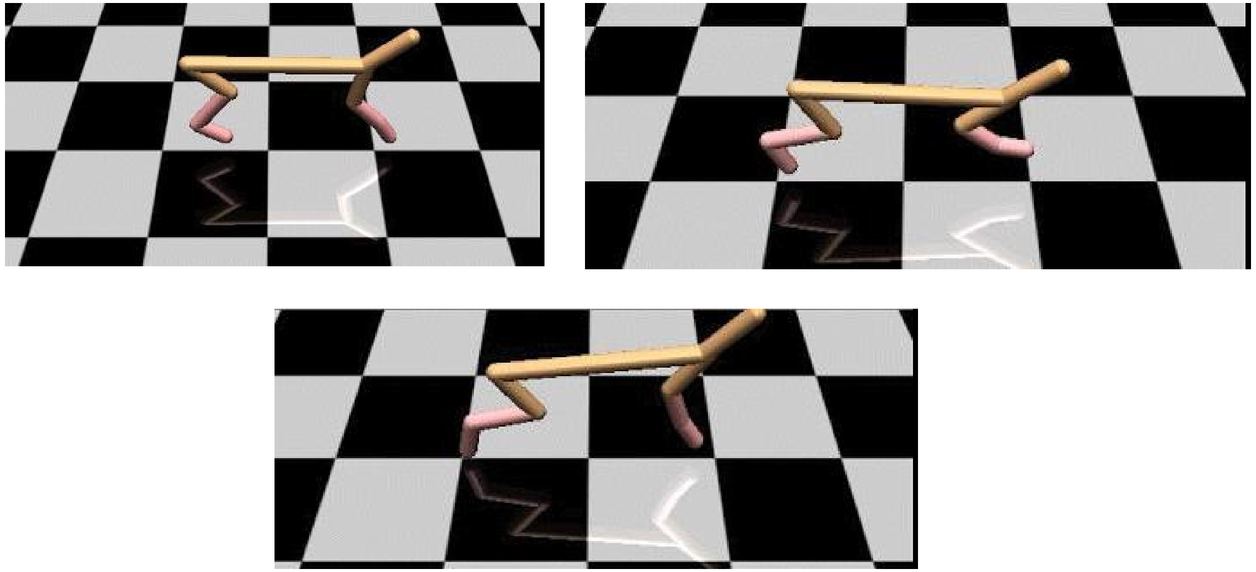

$$r(\mathbf{s}, \mathbf{a}) = \begin{cases} 1 \text{ if walker is running} \\ 0 \text{ otherwise} \end{cases}$$

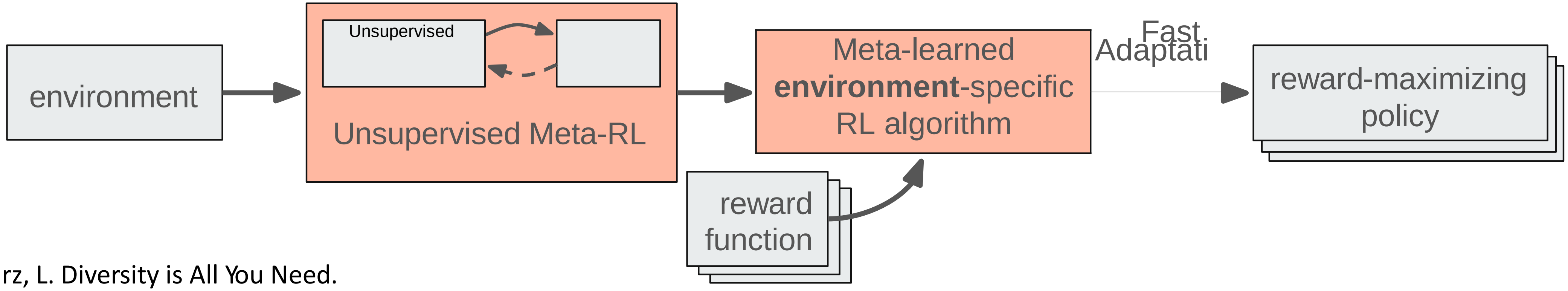
$$\begin{aligned} r(\mathbf{s}, \mathbf{a}) = & w_1 v(\mathbf{s}) + \\ & w_2 \delta(|\theta_{\text{torso}}(\mathbf{s})| < \epsilon) + \\ & w_3 \delta(h_{\text{torso}}(\mathbf{s}) \geq h) \end{aligned}$$




# Unsupervised reinforcement learning?

1. Interact with the world, without a reward function
2. Learn *something* about the world (what?)
3. Use what you learned to quickly solve new tasks






Eysenbach, Gupta, Ibarz, L. Diversity is All You Need.

Gupta, Eysenbach, Finn, L. Unsupervised Meta-Learning for Reinforcement Learning.

# Other sources of supervision

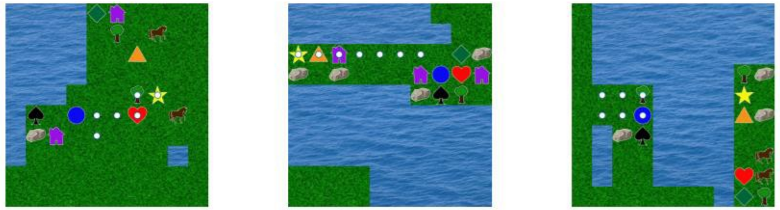
Should supervision tell us **what** to do or **how** to do it?

- Demonstrations
  - Muelling, K et al. (2013). Learning to Select and Generalize Striking Movements in Robot Table Tennis

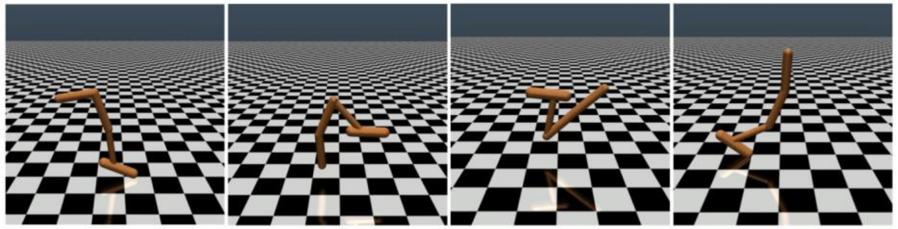
- Language
  - Andreas et al. (2018). Learning with latent language

**Human description:**
`move to the star`

**Inferred description:**
`reach the star cell`



- Human preferences
  - Christiano et al. (2017). Deep reinforcement learning from human preferences

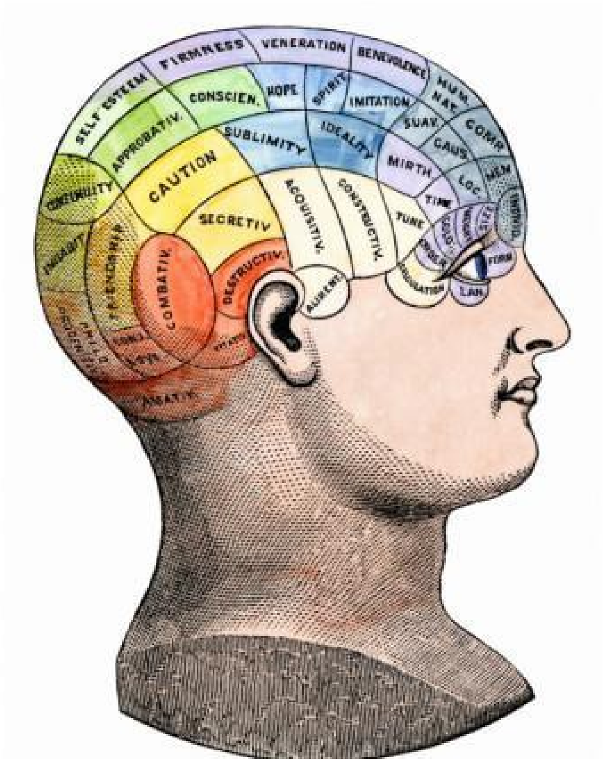


# Rethinking the Problem Formulation

- How should we define a *control* problem?
  - What is the data?
  - What is the goal?
  - What is the supervision?
    - may not be the same as the goal…
- Think about the assumptions that fit your problem setting!
- Don’t assume that the basic RL problem is set in stone

# Back to the Bigger Picture

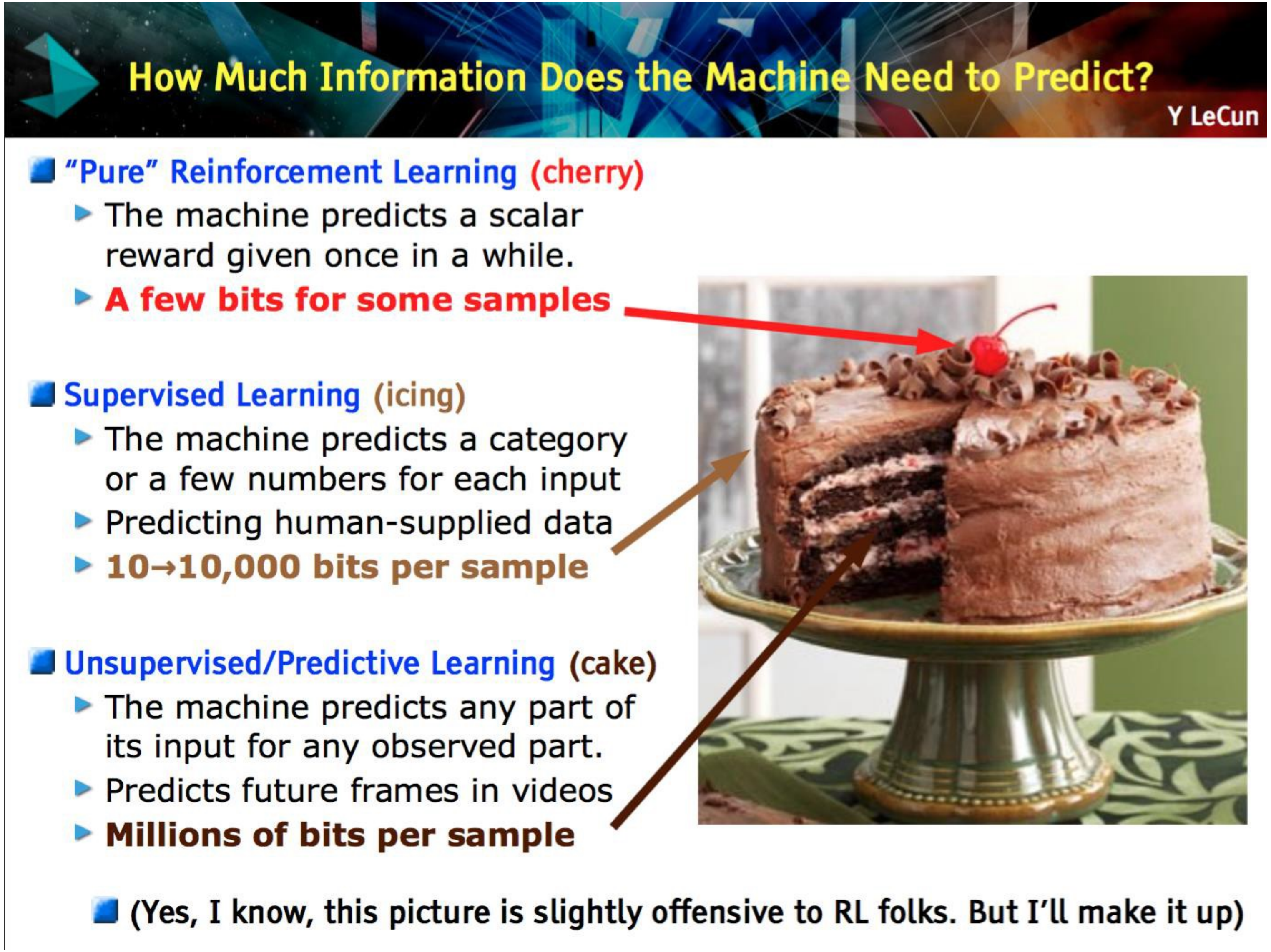
# Learning as the basis of intelligence




- Reinforcement learning = can reason about decision making
- Deep models = allows RL algorithms to learn and represent complex input-output mappings

Deep models are what allow reinforcement learning algorithms to solve complex problems end to end!

# What is missing?



# Where does the *signal* come from?

- Yann LeCun’s cake
  - Unsupervised or self-supervised learning
  - Model learning (predict the future)
  - Generative modeling of the world
  - Lots to do even before you accomplish your goal!
- Imitation & understanding other agents
  - We are social animals, and we have culture – for a reason!
- The giant value backup
  - All it takes is one +1
- All of the above

# How should we answer these questions?

- Pick the right problems!
- Pay attention to generative models, prediction, etc., not just RL algorithms
- Carefully understand the relationship between RL and other ML fields