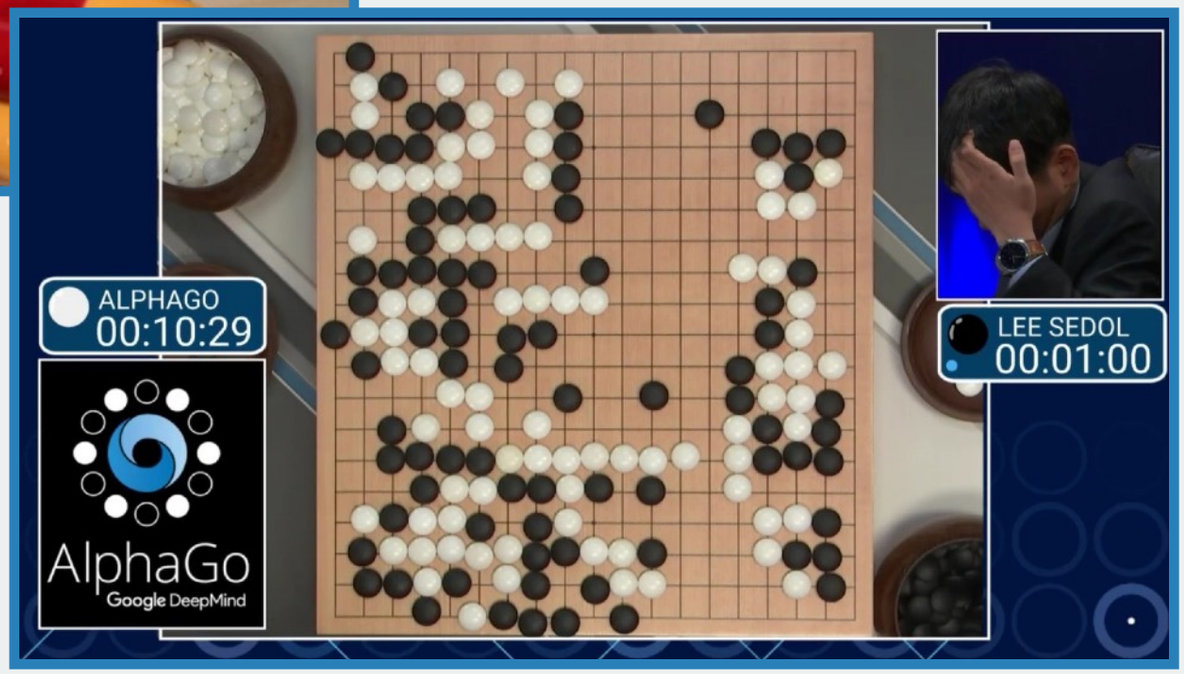
Model-Free Control
(Reinforcement Learning)
and Deep Learning

MARC G. BELLEMARE
Google Brain (Montréal)

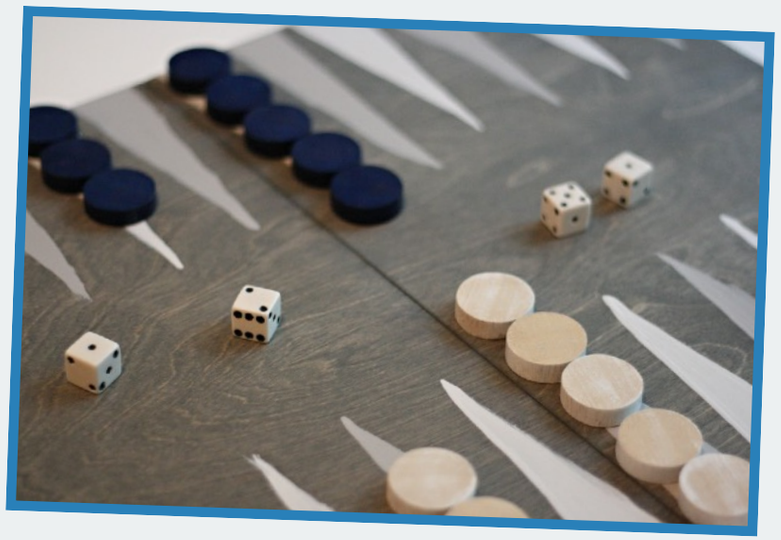

Starters
Onion Bhaji £2.45
Thinly sliced onion in gram flour and herb, deep fried
Samosa (Meat or Vegetable) £2.45
Triangular shaped savoury fried with spicy meat or vegetable
Prawn on Puree £3.25
Prawn cooked in herbs & spices, served on a thin fried bread
King Prawn on Puree £4.25
Large Prawn cooked in herbs & spices, served on a thin fried bread
King Prawn Butterfly £4.25
Buttered king prawns with herbs and spices and deep fried
Tandoori Chicken (Qtr) £3.75
Spring chicken marinated in herbs & spices, cooked over charcoal tandoor
Chicken or Lamb Tikka £3.25
Diced chicken or lamb marinated in yoghurt and spices and then barbecued in the clay oven
Sheek Kebab £3.25
Minced lamb pungently spiced, skewered and barbecued in the tandoor oven
Shami Kebab £3.50
Minced lamb in herbs and spices then fried in ghee
Chicken Chat or Aloo Chat (sour) £2.95
Small juicy pieces of chicken or potatoes spiced with hot and sour sauce
Prawn Cocktail £2.95
Succulent prawns in our own mayonnaise
Dall Soup or Mulligatwany Soup (Lentils) £2.25
Stuffed Tomatoes £3.25
Tomatoes stuffed with vegetables or minced meat
Chicken or Vegetable Pakora £3.25
Chicken tikka or vegetable deep fried in gram flour coating
Plain or Massala Papadum 60p
Mixed Kebab £4.95
Kebab Roll £3.75


NOVO 500


ALPHAGO
00:10:29
AlphaGo
Google DeepMind
LEE SEDOL
00:01:00

1956

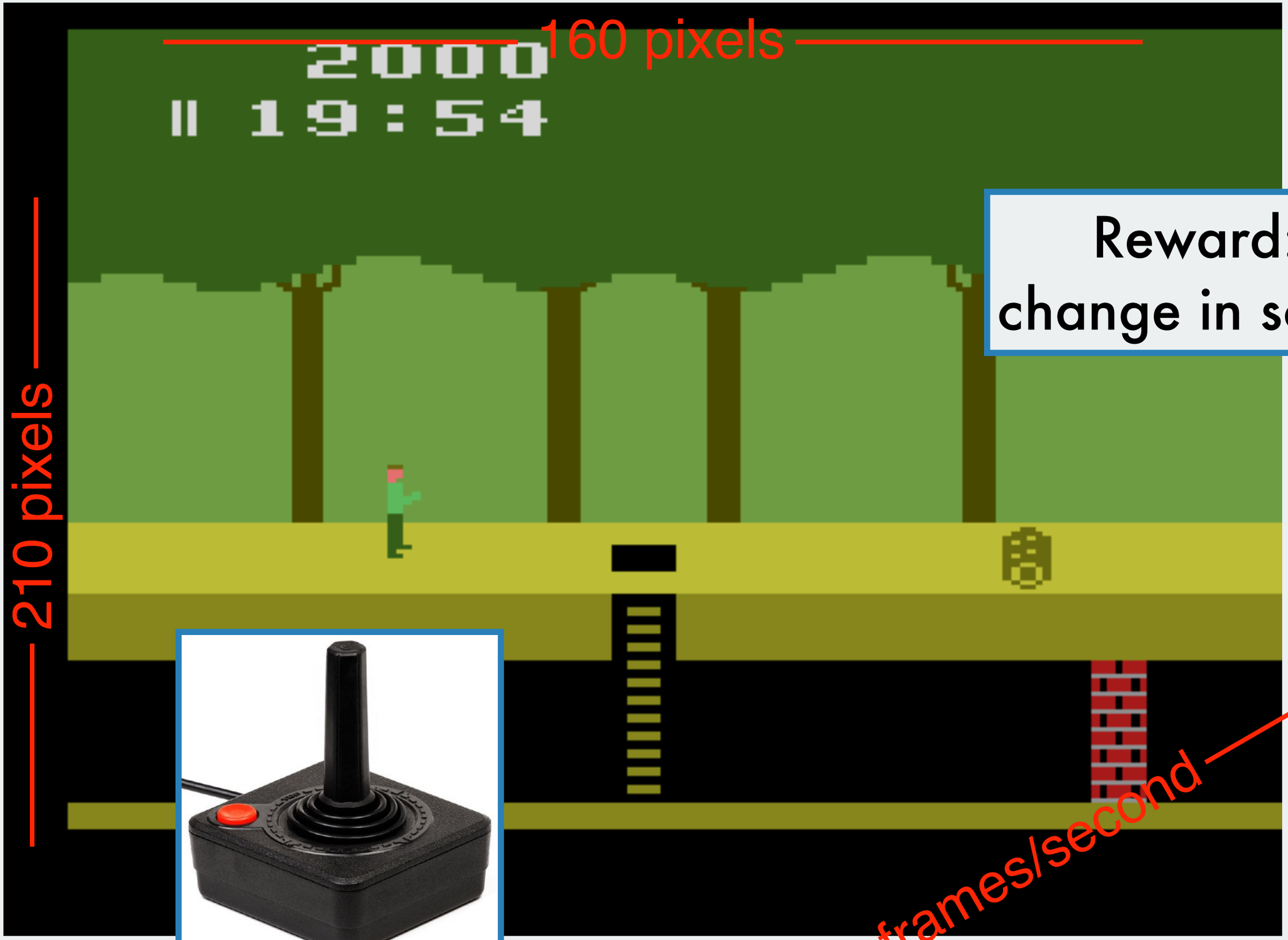
1992



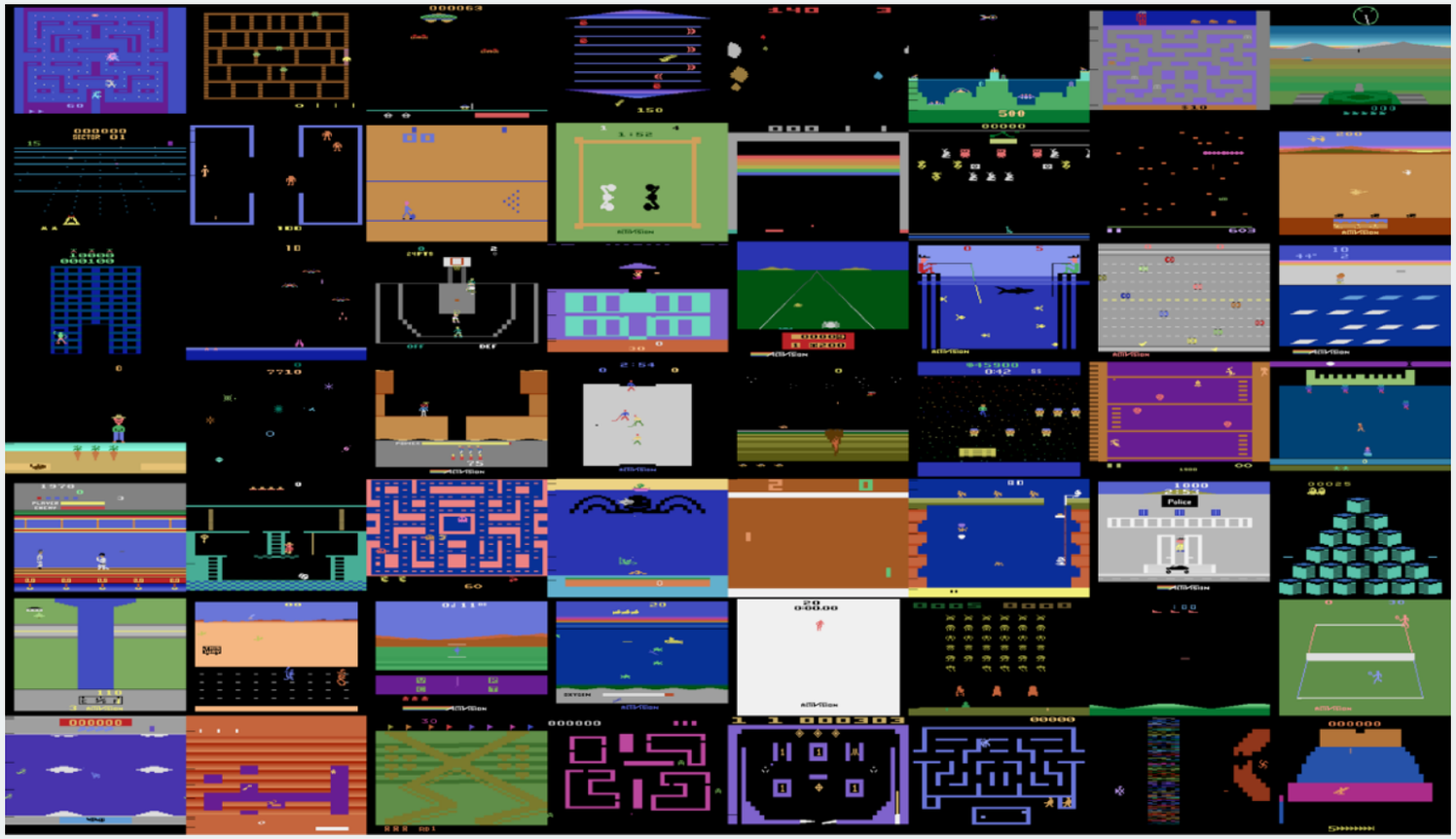
2016

2015

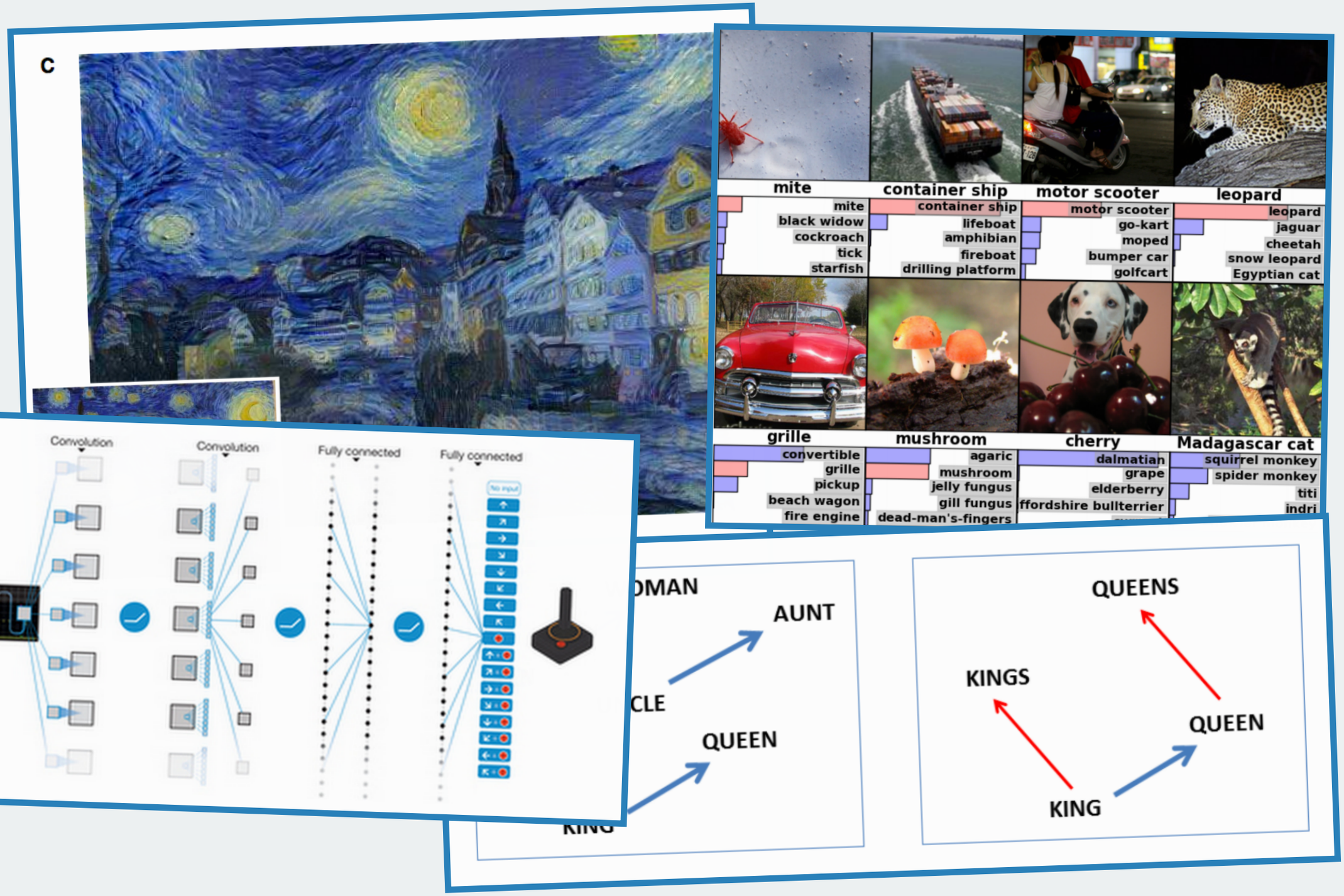
The Arcade Learning Environment (Bellemare et al., 2013)


160 pixels
210 pixels
2000
19:54
Reward:
change in score
18 actions
60 frames/second

- 33,600 (discrete) dimensions
- Up to 108,000 decisions/episode (30 minutes)
- 60+ games: heterogenous dynamical systems



Deep Learning:
An AI Success Story



$$\mathcal{M} := \langle \mathcal{X}, \mathcal{A}, R, P, \gamma \rangle$$

$$Q^*(x,a) = r(x,a) + \gamma \mathop{\mathbf{E}}_{P} \max_{a' \in \mathcal{A}} Q^*(x',a')$$

$$\hat{Q} = \Pi \mathcal{T}^\pi \hat{Q}$$

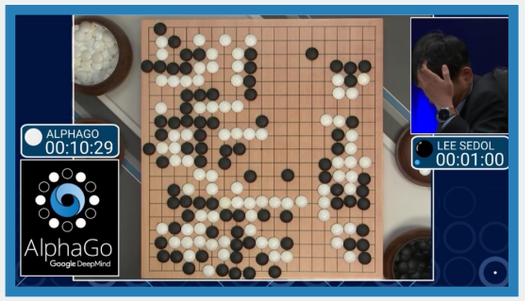
$$\left\| \hat{Q} - Q^\pi \right\|_D \leq \frac{1}{1-\gamma} \left\| \Pi Q^\pi - Q^\pi \right\|_D$$

Theory

Practice

Where has model-free control been so successful?

- Complex dynamical systems
  - Black-box simulators
- High-dimensional state spaces
- Long time horizons
- Opponent / adversarial element



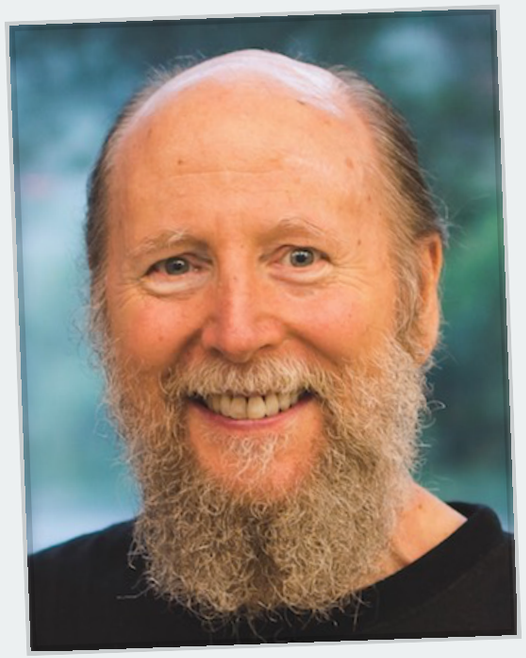
Practical Considerations

- Are simulations reasonably cheap? *model-free*
- Is the notion of "state" complex? *model-free*
- Is there partial observability? *maybe model-free*
- Can the state space be enumerated? *value iteration*
- Is there an explicit model available? *model-based*

Outline of Talk

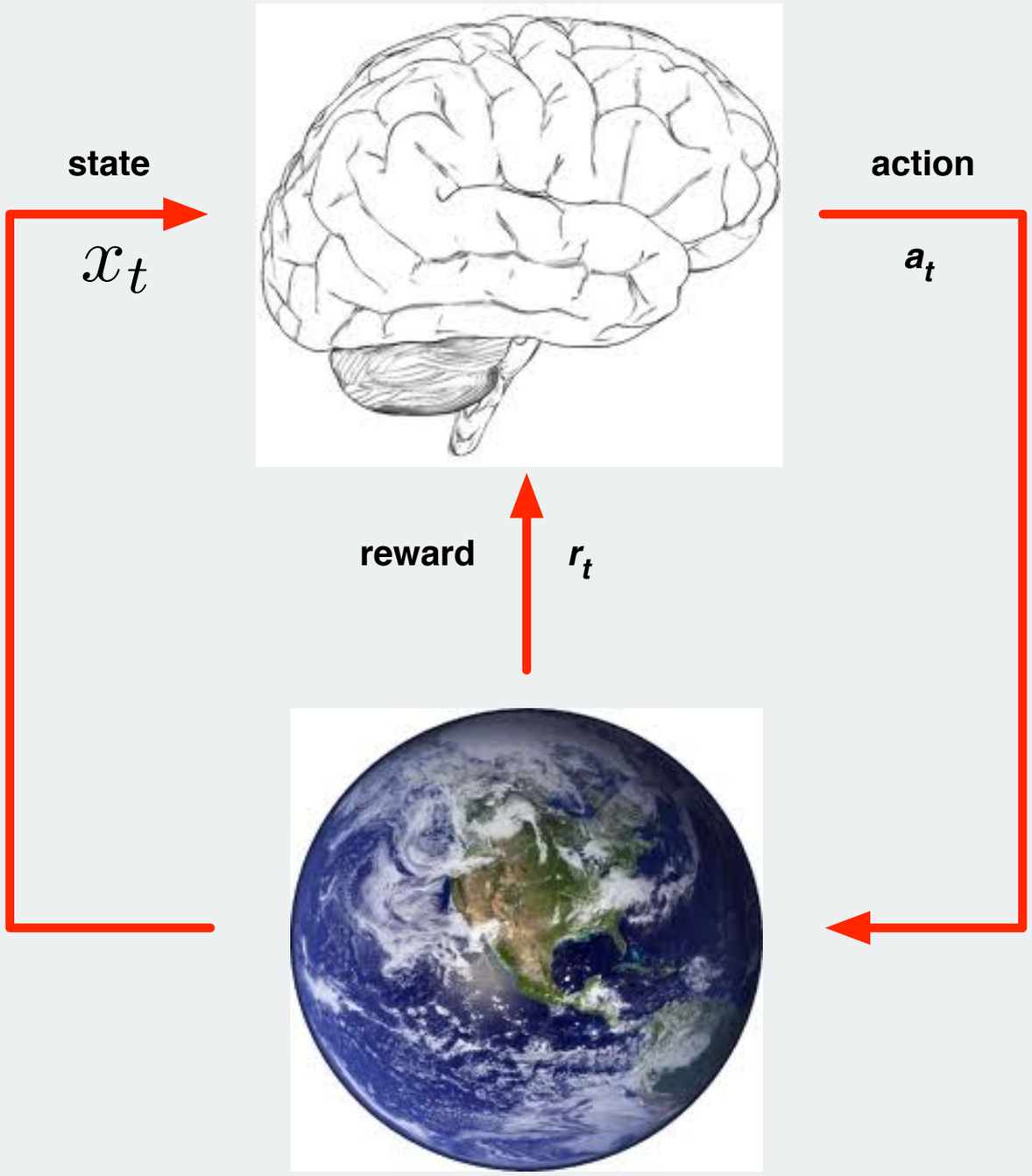
Ideal case → Practical case

What's Reinforcement Learning, Anyway?



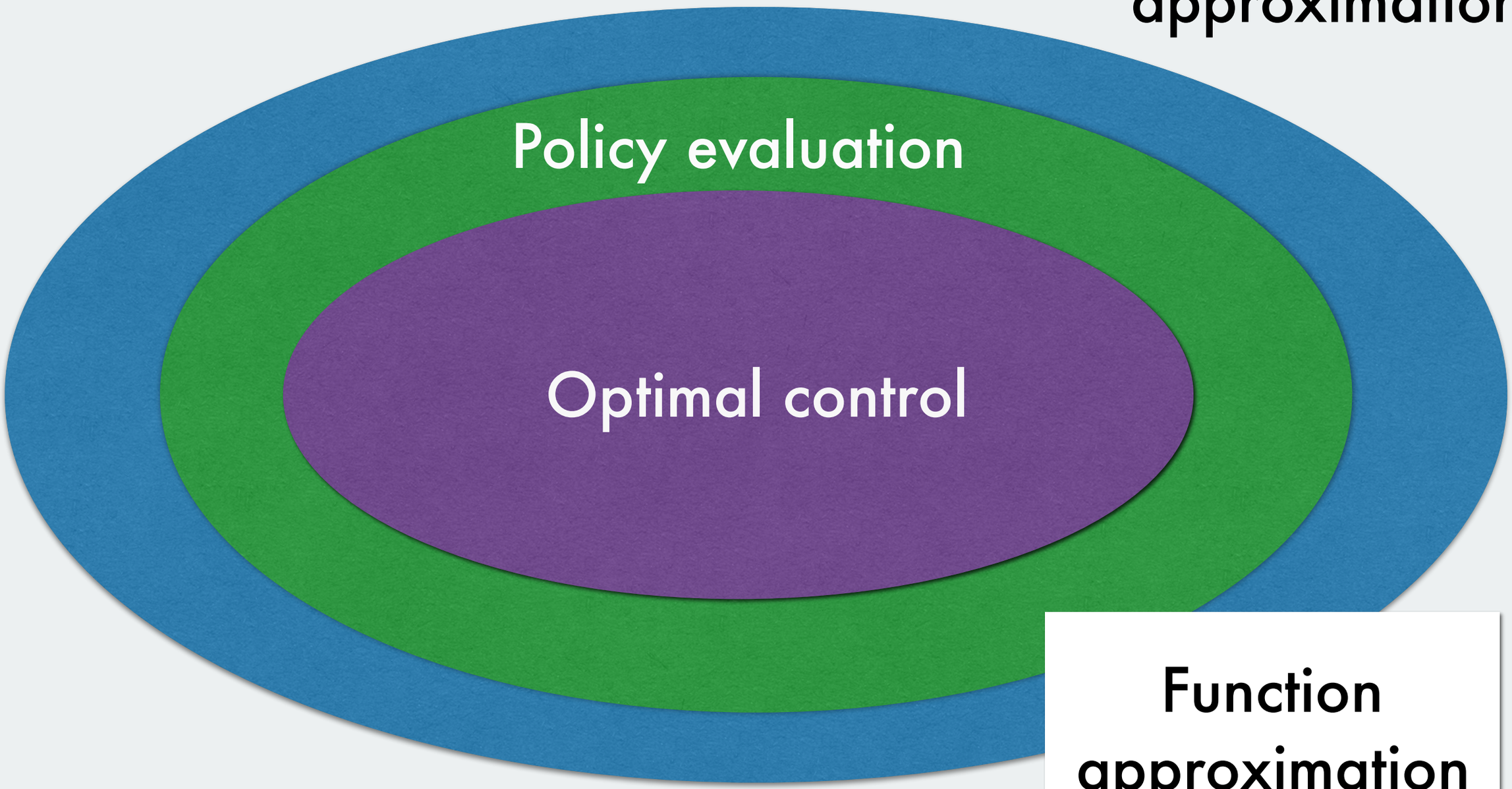
"All goals and purposes … can be thought of as the maximization of some value function"

– Sutton & Barto (2017, in press)



- At each step *t*, the agent
  - Observes a state
  - Takes an action
  - Receives a reward

Three learning problems in one



BACKGROUND

- Formalized as a Markov Decision Process:

$$\mathcal{M} := \langle \mathcal{X}, \mathcal{A}, R, P, \gamma \rangle$$

  - *R*, *P* reward, transition functions
  - γ discount factor

- A trajectory is a sequence of interactions with the environment

$$x_1, a_1, r_1, x_2, a_2, \ldots$$

- Policy π: a probability distribution over actions:

$$a_t \sim \pi(\cdot \,|\, x_t)$$

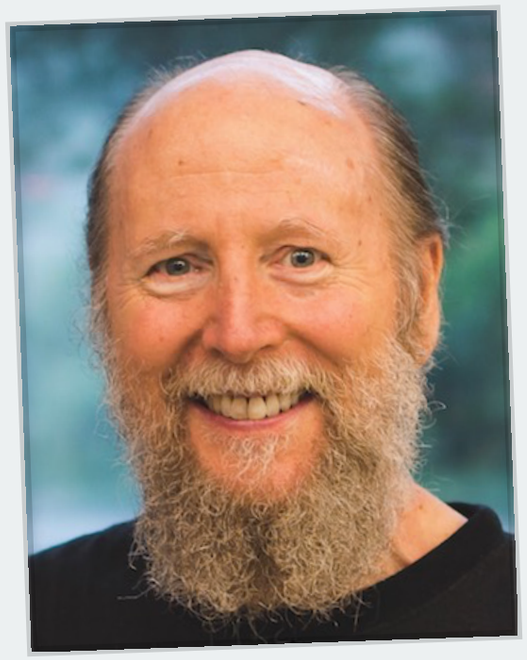
If deterministic: $a_t = \pi(x_t)$

- Transition function: $x_{t+1} \sim P(\cdot \,|\, x_t, a_t)$

- Value function $Q^\pi(x, a)$: total discounted reward

$$Q^\pi(x, a) = \mathop{\mathbf{E}}_{P,\pi}\left[\sum_{t=0}^{\infty} \gamma^t r(x_t, a_t) \,|\, x_0, a_0 = x, a\right]$$

- As a vector in space of value functions: $Q^\pi \in \mathcal{Q}$



- "Maximize value function": find

$$Q^*(x,a) := \max_\pi Q^\pi(x,a)$$

- Bellman's equation:

$$\begin{aligned} Q^\pi(x,a) &= \mathop{\mathbf{E}}_{P,\pi}\left[\sum_{t=0}^{\infty}\gamma^t r(x_t,a_t)\,|\,x_0,a_0=x,a\right] \\ &= r(x,a) + \gamma \mathop{\mathbf{E}}_{P,\pi} Q^\pi(x',a') \end{aligned}$$

- Optimality equation:

$$Q^*(x,a) = r(x,a) + \gamma \mathop{\mathbf{E}}_{P} \max_{a'\in\mathcal{A}} Q^*(x',a')$$

Bellman Operator

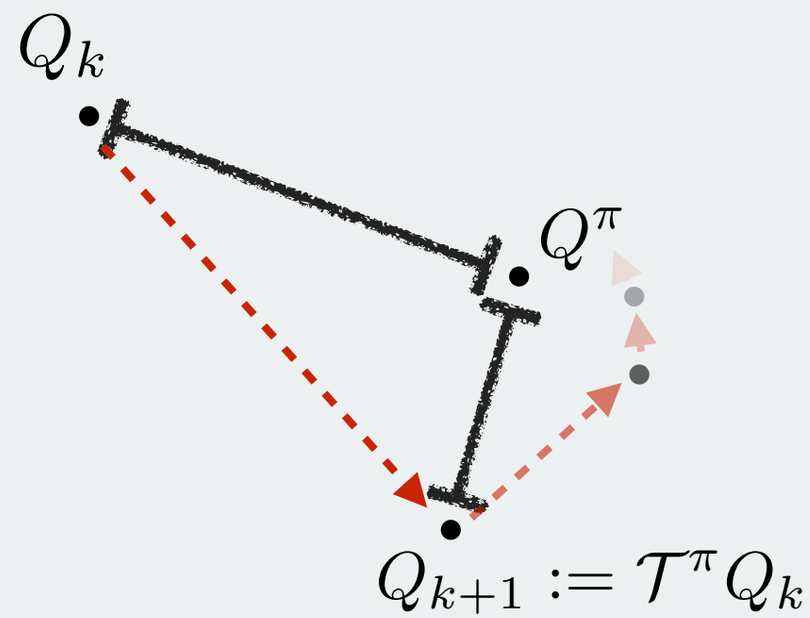
$$\mathcal{T}^\pi Q(x,a) := r(x,a) + \gamma \mathop{\mathbf{E}}_{\substack{x' \sim P \\ a' \sim \pi}} Q(x',a')$$

- The Bellman operator is a γ-contraction:

$$\|\mathcal{T}^\pi Q - Q^\pi\|_\infty \leq \gamma \|Q - Q^\pi\|_\infty$$

- Fixed point:

$$Q^\pi = \mathcal{T}^\pi Q^\pi$$



Bellman Optimality Operator

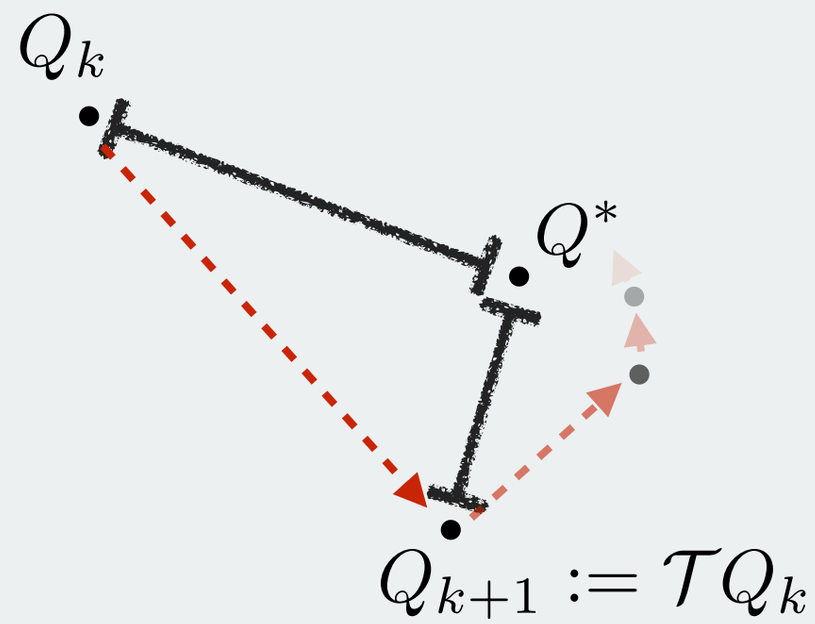
$$\mathcal{T}Q(x,a) := r(x,a) + \gamma \mathop{\mathbf{E}}_{x' \sim P} \max_{a' \in \mathcal{A}} Q(x', a')$$

- Also a γ-contraction (beware! different proof):

$$\|\mathcal{T}Q - Q^*\|_\infty \leq \gamma \|Q - Q^*\|_\infty$$

- Fixed point is optimal v.f.:

$$Q^* = \mathcal{T}Q^* \geq Q^\pi$$



Model-based Algorithms

1.Value iteration:

$$Q_{k+1}(x,a) \leftarrow \mathcal{T}Q_k(x,a) = r(x,a) + \gamma \, \mathbf{E}_P \max_{a' \in \mathcal{A}} Q_k(x',a')$$

2.Policy iteration:

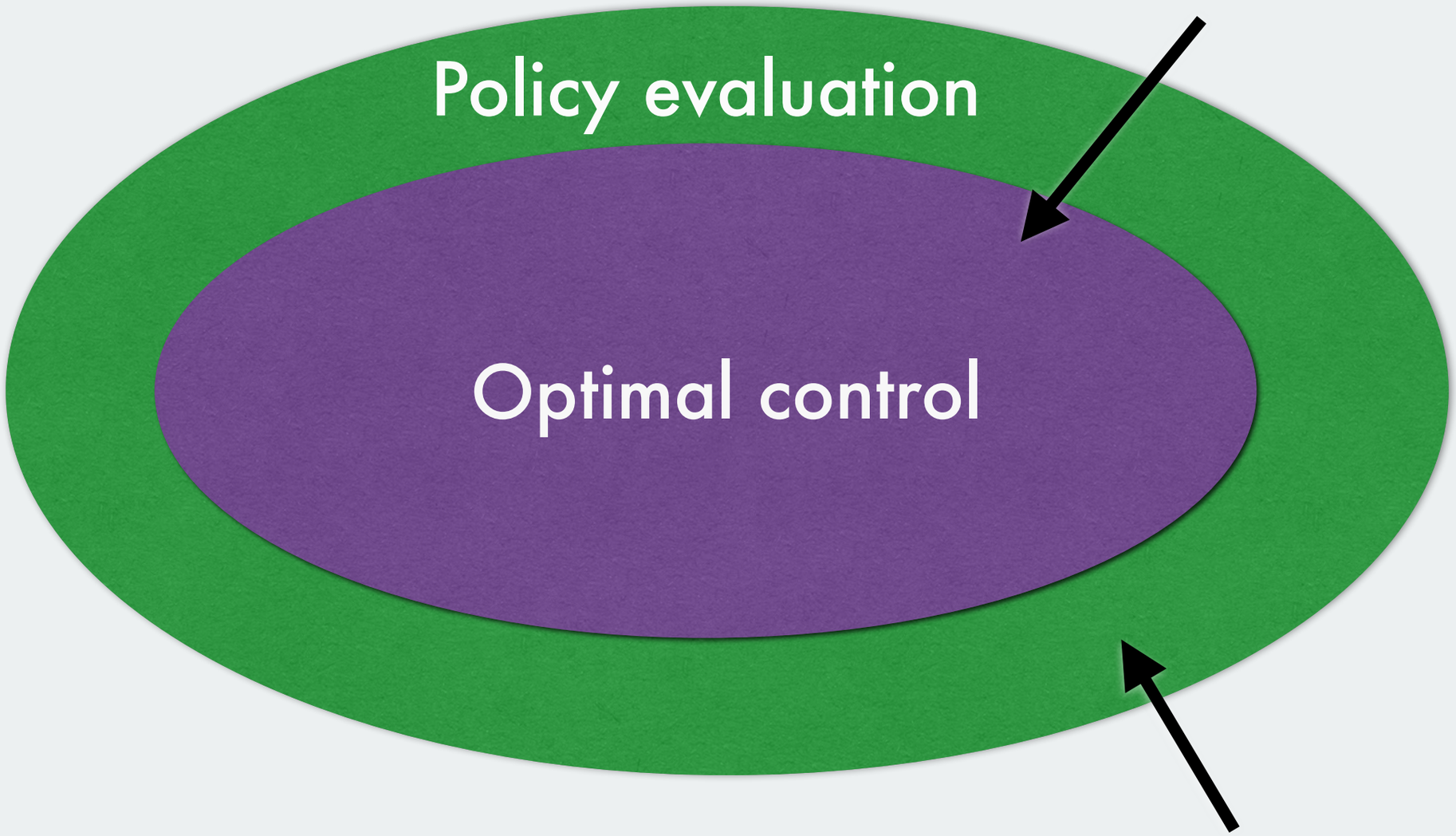
a. $\pi_k = \arg\max_{\pi} \mathcal{T}^{\pi} Q_k(x,a)$ **b.** $Q_{k+1}(x,a) \leftarrow Q^{\pi_k}(x,a)$

3.Optimistic policy iteration:

$$Q_{k+1}(x,a) \leftarrow (\mathcal{T}^{\pi_k})^m Q_k(x,a) = \underbrace{\mathcal{T}^{\pi_k} \cdots \mathcal{T}^{\pi_k}}_{m \text{ times}} Q_k(x,a)$$

Policy Iteration

a. $\pi_k = \arg\max_{\pi} \mathcal{T}^{\pi} Q_k(x, a)$

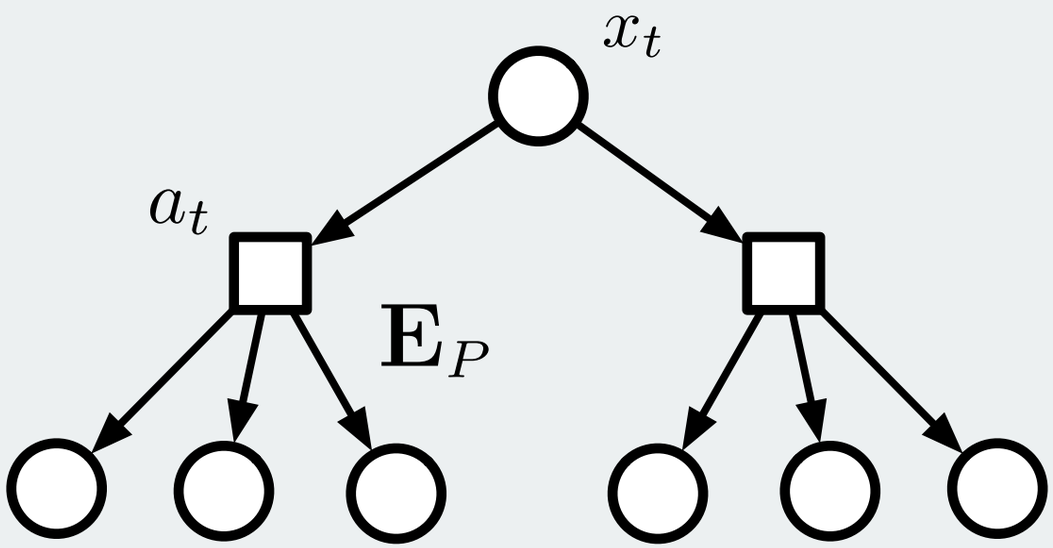


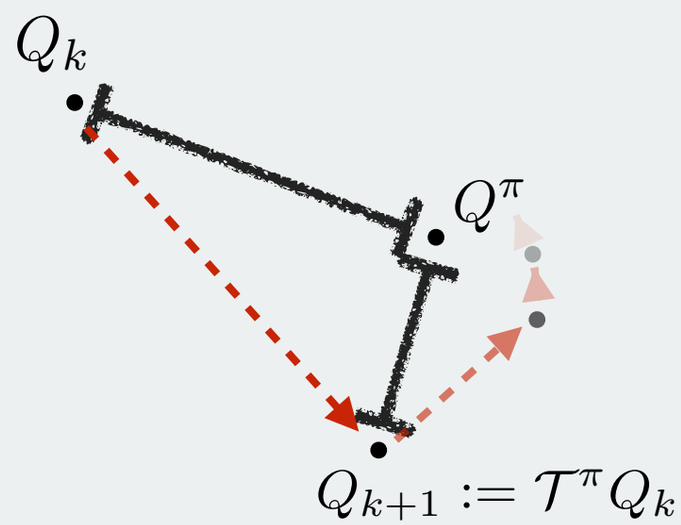
b. $Q_{k+1}(x, a) \leftarrow Q^{\pi_k}(x, a)$

Model-free Reinforcement Learning

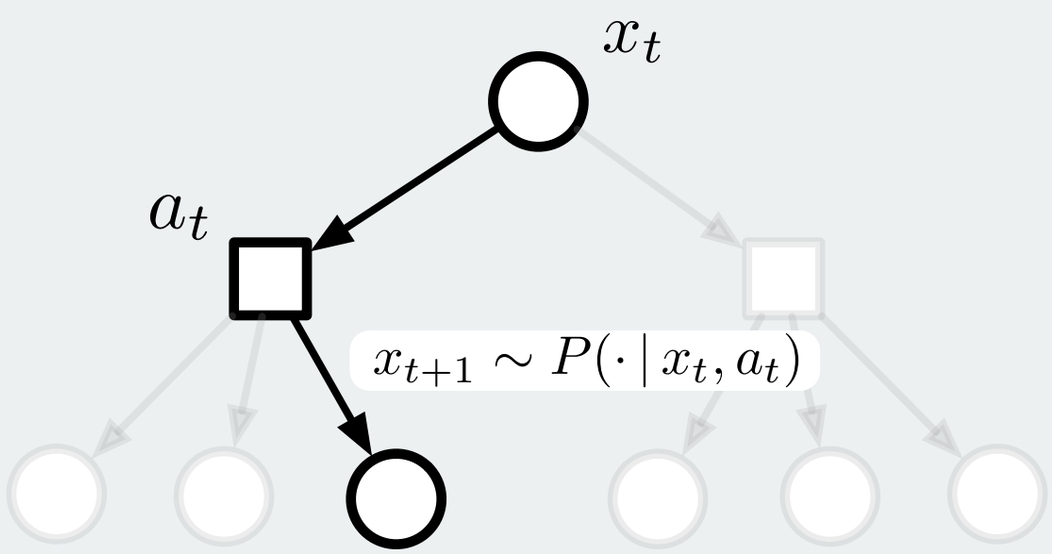
- Typically no access to *P, R*
- Two options:
  - Learn a model (not in this talk)
  - Model-free: learn Q^π or Q^* directly from samples

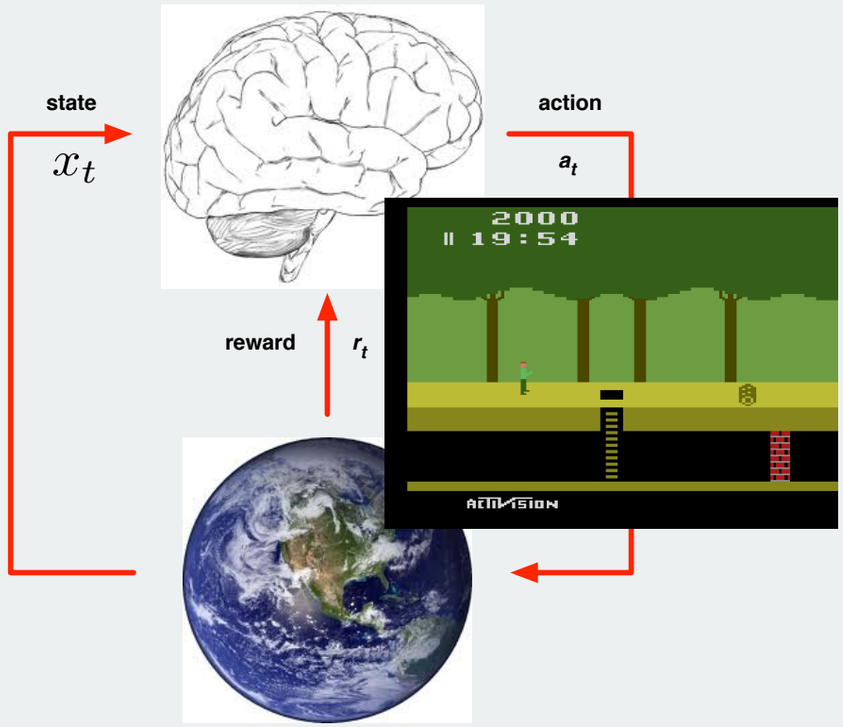
Model-based





Model-free





Model-free RL: Synchronous Updates

- For all *x*, *a*, sample $x' \sim P(\cdot \,|\, x, a), a' \sim \pi(\cdot \,|\, x')$
- The SARSA algorithm:

$$\begin{aligned} Q_{t+1}(x,a) &\leftarrow (1-\alpha_t) Q_t(x,a) + \alpha_t \mathcal{T}_t^\pi Q_t(x,a) \\ &= (1-\alpha_t) Q_t(x,a) + \alpha_t \big(r(x,a) + \gamma Q_t(x',a') \big) \\ &= Q_t(x,a) + \alpha_t \underbrace{\big(r(x,a) + \gamma Q_t(x',a') - Q_t(x,a) \big)}_{\text{TD-error } \delta} \big) \end{aligned}$$

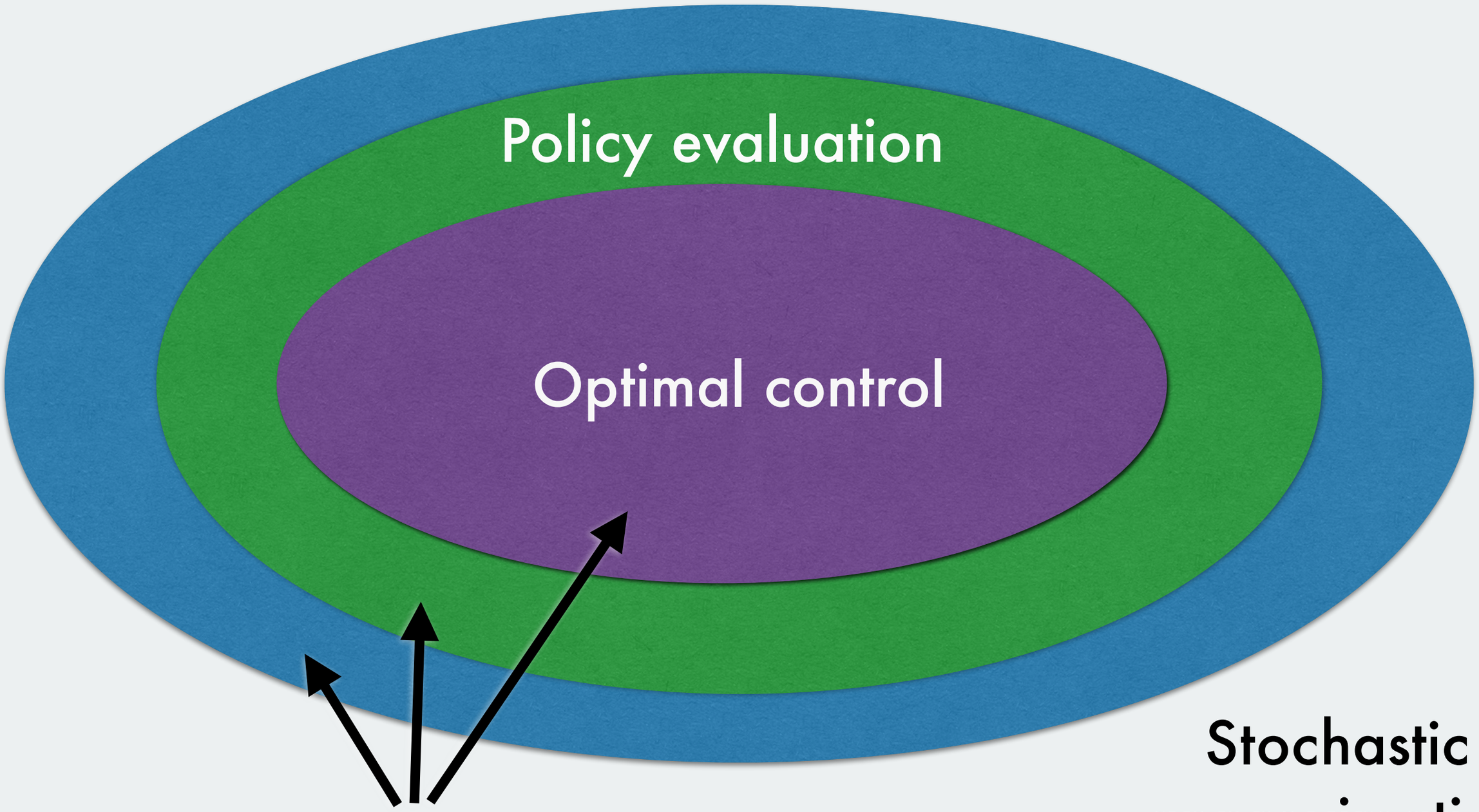
- $\alpha_t \in [0,1)$ is a step-size (sequence)

Model-free RL: Q-Learning

- The Q-Learning algorithm: max. at each iteration

$$Q_{t+1}(x,a) \leftarrow (1-\alpha_t)Q_t(x,a) + \alpha_t\big(r(x,a) + \gamma \max_{a' \in \mathcal{A}} Q_t(x',a') - Q_t(x,a)\big)$$

- Both converge under Robbins-Monro conditions
- Not trivial! Interleaved learning problems



Asynchronous Updates

- The asynchronous case: learn from trajectories

$$x_1, a_1, r_1, x_2, a_2, \cdots \sim \pi, P$$

- Apply update at each step:

$$Q(x_t, a_t) \leftarrow Q_t(x, a) + \alpha_t \big(r_t + \gamma Q(x_{t+1}, a_{t+1}) - Q(x_t, a_t)\big)$$

- This is the setting we usually deal with

- Convergence even more delicate

Open questions/ Areas of active research

- Rates of convergence [1]
- Variance reduction [2]
- Convergence guarantees for multi-step methods [3, 4]
- Off-policy learning: control from fixed behaviour [3, 4]

[1] Konda and Tsitsiklis (2004)
[2] Azar et al., Speedy Q-Learning (2011)
[3] Harutyunyan, Bellemare, Stepleton, Munos (2016)
[4] Munos, Stepleton, Harutyunyan, Bellemare (2016)

Stochastic approximation

Policy evaluation

Optimal control

Function approximation

Stochastic approximation

Policy evaluation

Optimal control

Function approximation

(Value) function Approximation

- Parametrize value function:

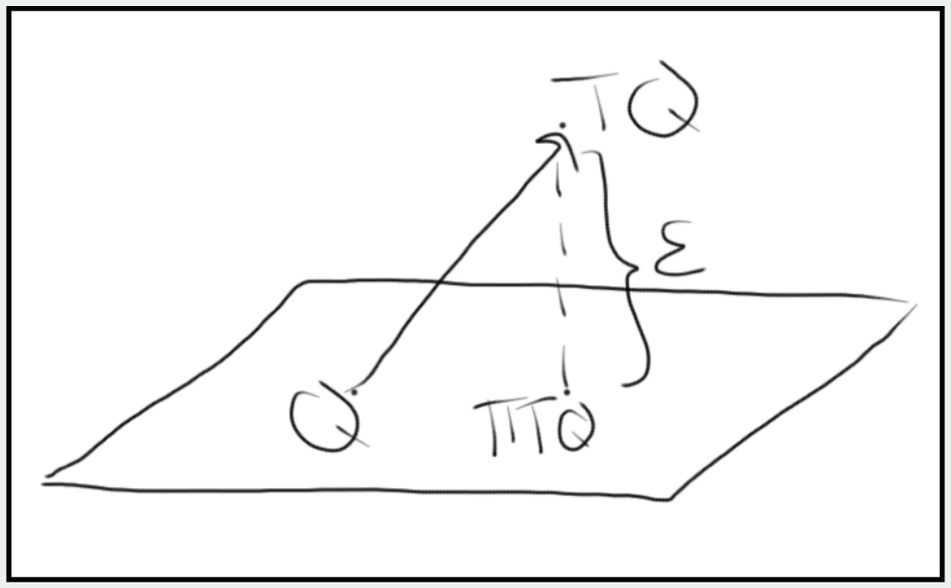
$$Q^{\pi}(x, a) \approx Q(x, a, \theta)$$

- Learning now involves a projection step Π:

$$\Pi \mathcal{T}^{\pi} Q(x, a, \theta_k) : \theta_{k+1} \leftarrow \arg\min_{\theta} \left\| \mathcal{T}^{\pi} Q_k(x, a, \theta_k) - Q(x, a, \theta) \right\|_D$$

- This leads to additional, compounding error

- Can cause divergence



Some classic Results [1]

- Linear approximation:

$$Q^\pi(x,a) \approx \theta^\top \phi(x,a)$$

- SARSA converges to $\hat{Q}$ satisfying

$$\hat{Q} = \Pi \mathcal{T}^\pi \hat{Q} \quad \left\| \hat{Q} - Q^\pi \right\|_D \leq \frac{1}{1-\gamma} \left\| \Pi Q^\pi - Q^\pi \right\|_D$$

- Q-Learning may diverge!

[1] Tsitsiklis and Van Roy (1997)

Open questions/ Areas of active research

- Convergent, linear-time optimal control [1]
- Exploration under function approximation [2]
- Convergence of multi-step extensions [3]

[1] Maei et al. (2009)
[2] Bellemare, Srinivasan, Ostrovski, Schaul, Saxton, Munos (2016)
[3] Touati et al. (2017)

Ideal case

Practical case

Ideal case

Practical case

1956

1992

2016

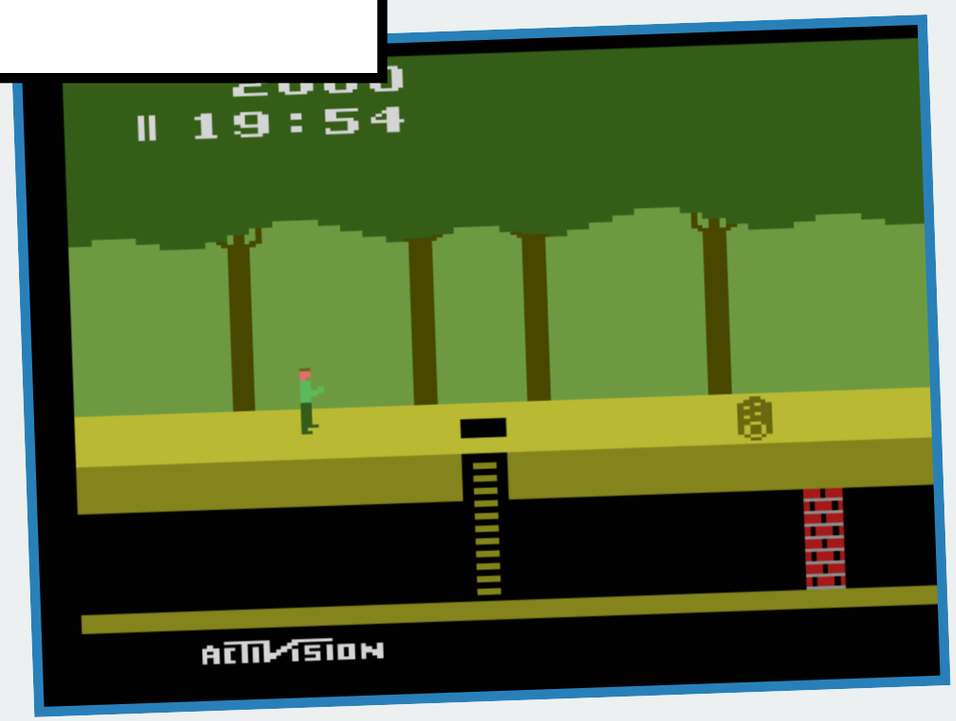
2015

1956

1992

$$Q^{\pi}(x,a) \approx \theta^{\top}\phi(x,a)$$

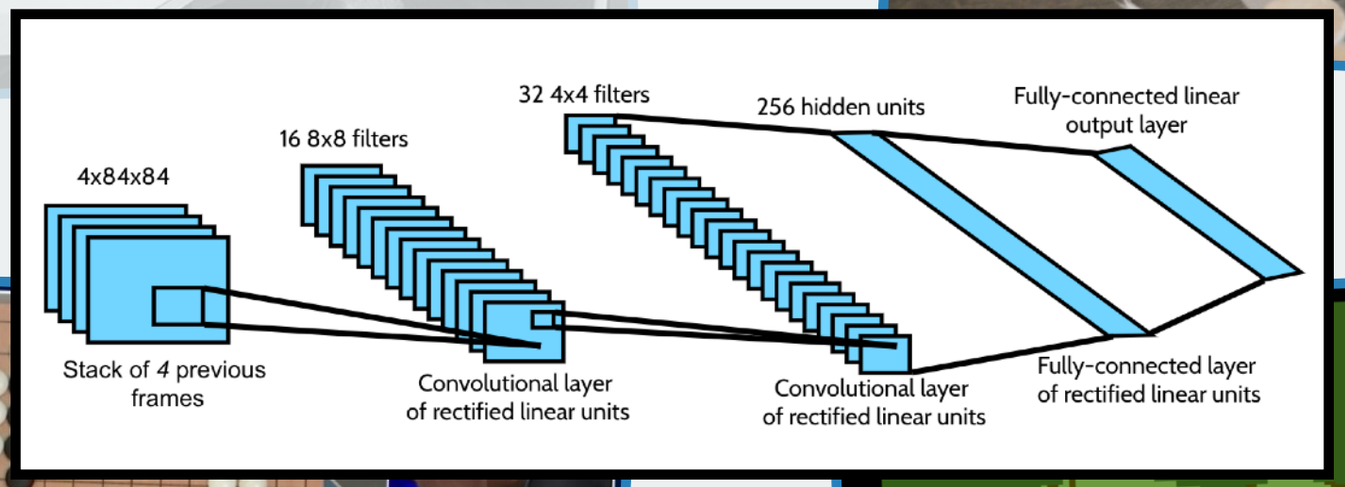
2016

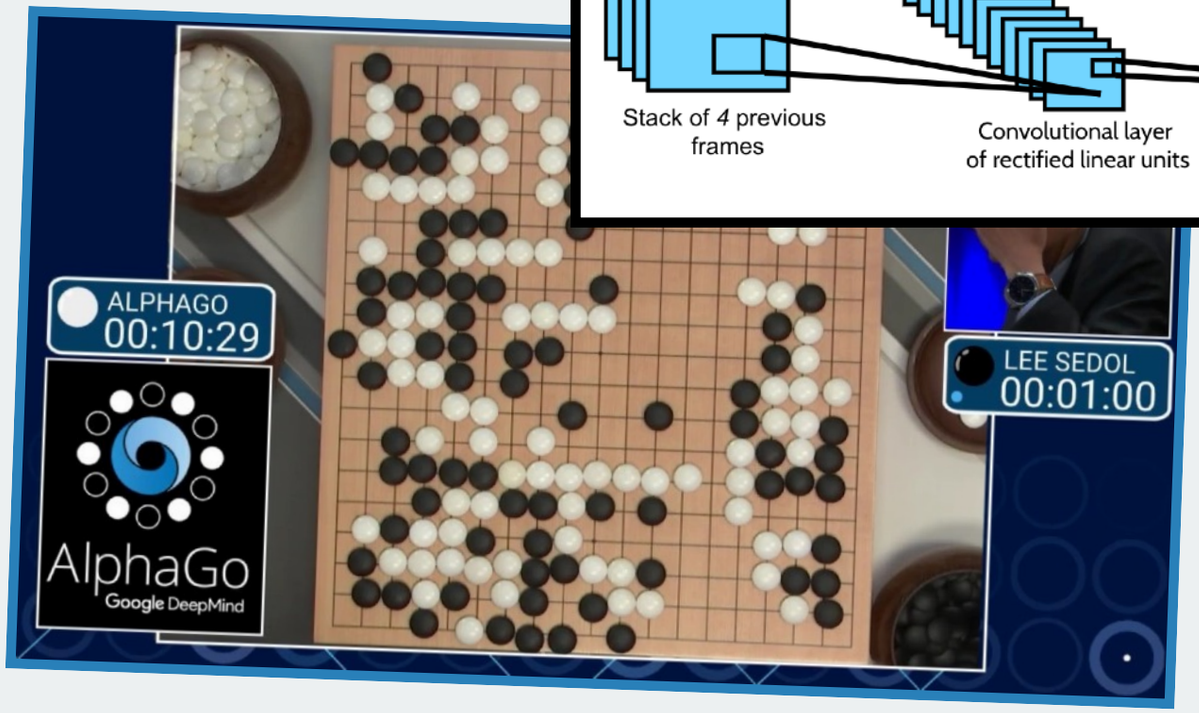


2015

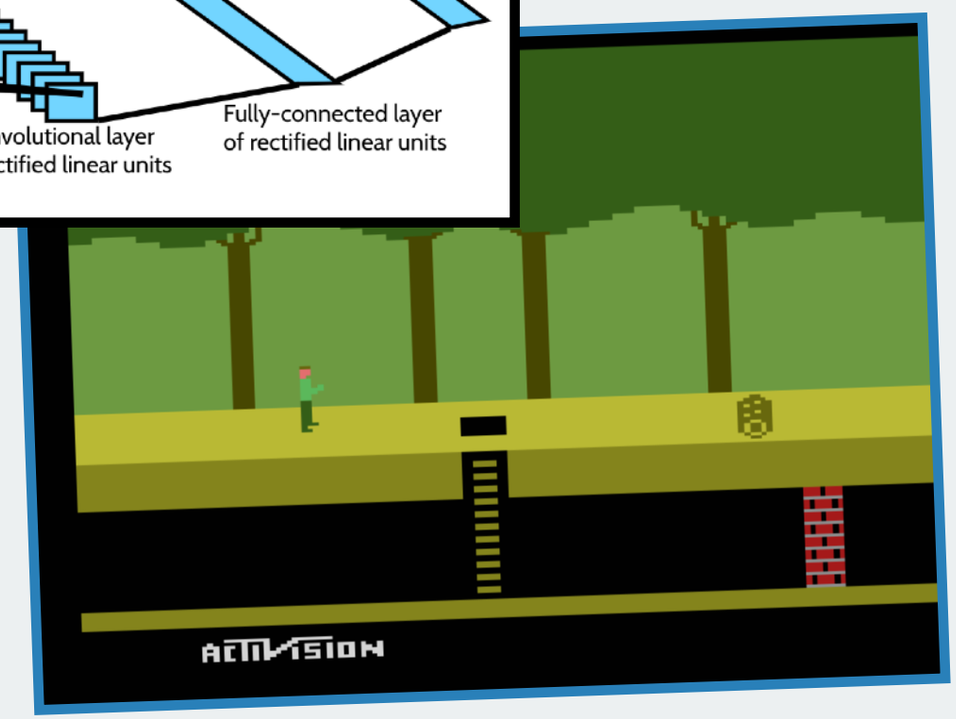
1956

1992



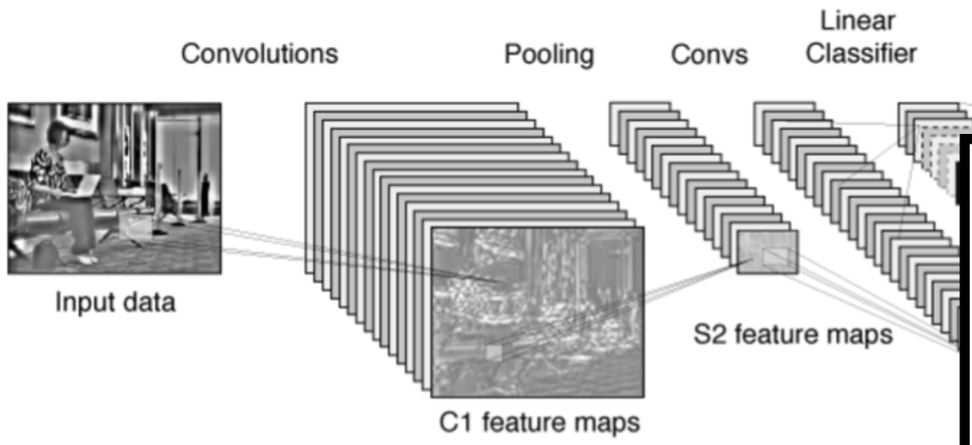


2016

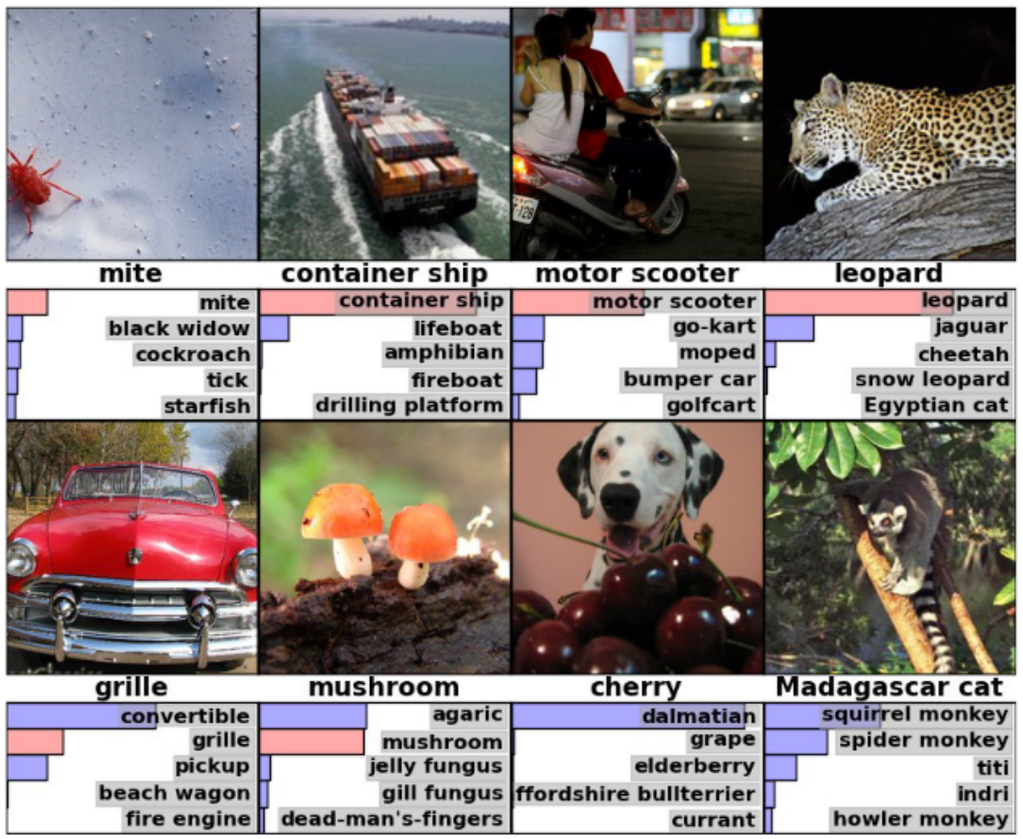


2015

Deep Learning



Torch (2015)

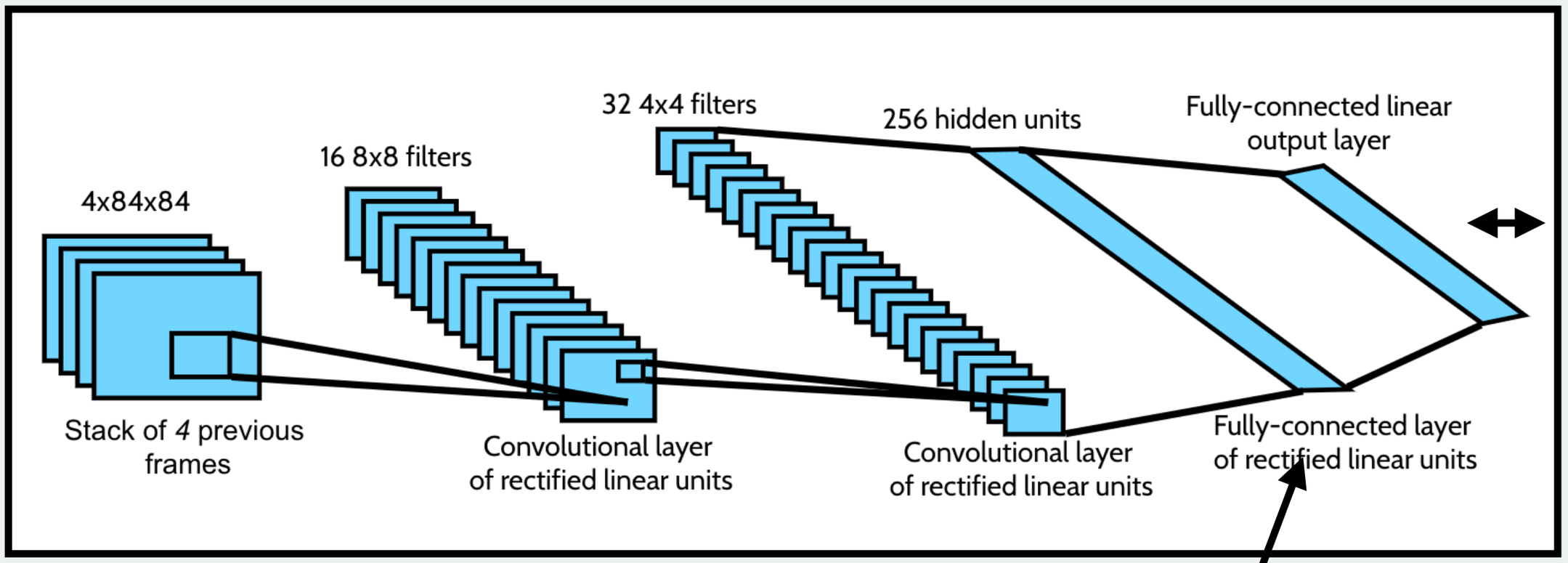


Krizhevsky et al. (2012)

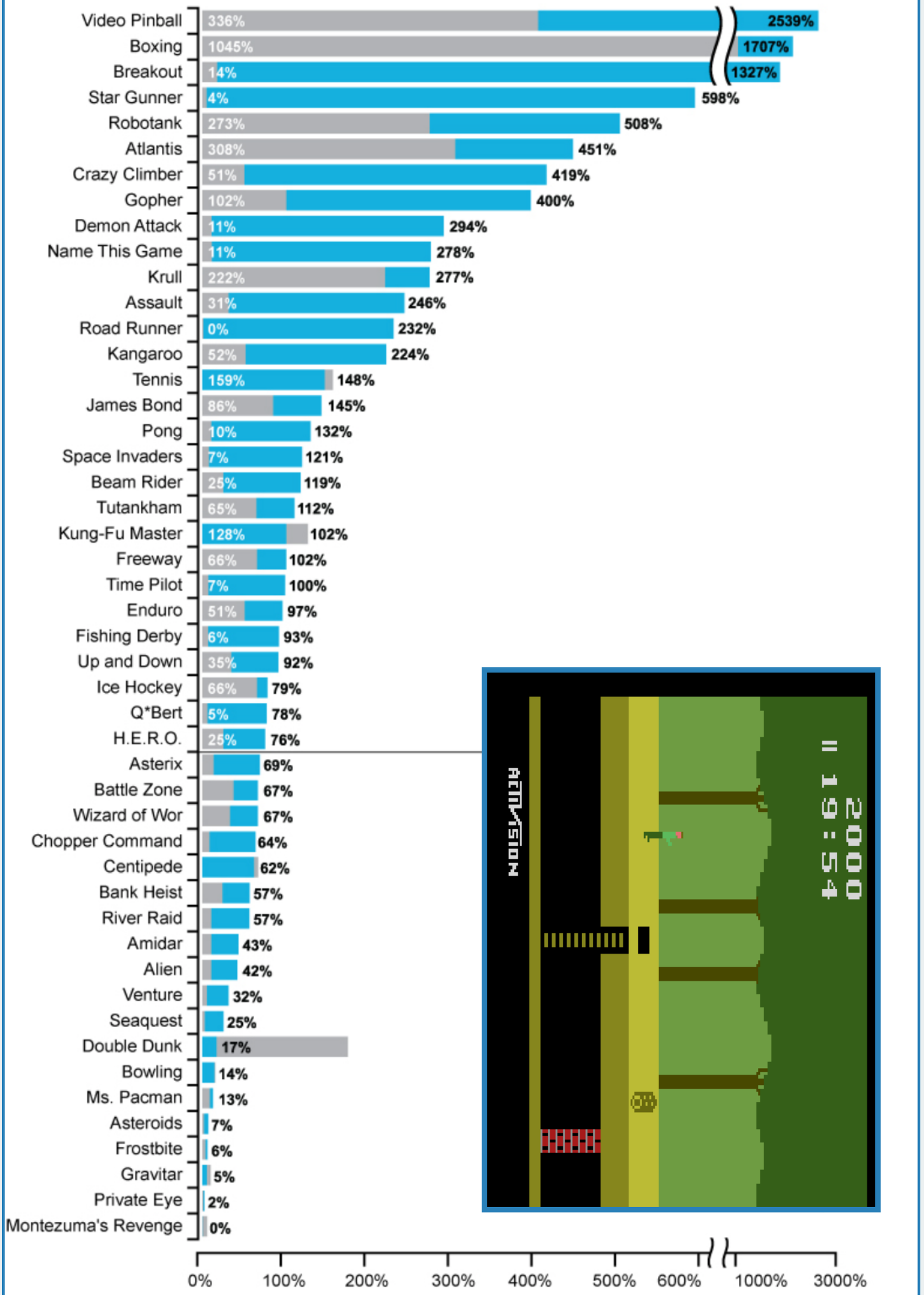
Slide adapted from Ali Eslami

Deep Learning

$$\nabla_\theta \mathcal{L}(\theta)$$



Graphic by Volodymyr Mnih



Mnih et al., 2015

Deep Reinforcement Learning

- Value function as a Q-network $Q(x, a, \theta)$
- Objective function: mean squared error

$$\mathcal{L}(\theta) := \mathbf{E}\left[\Big(\underbrace{r + \gamma \max_{a' \in \mathcal{A}} Q(x', a', \theta)}_{\text{target}} - Q(x, a, \theta)\Big)^2\right]$$

- Q-Learning gradient:

$$\nabla_\theta \mathcal{L}(\theta) = \mathbf{E}\left[\Big(r + \gamma \max_{a' \in \mathcal{A}} Q(x', a', \theta) - Q(x, a, \theta)\Big) \nabla_\theta Q(x, a, \theta)\right]$$

Based on material by David Silver

Stability Issues

- Naive Q-Learning oscillates or diverges

1. Data is sequential
   - ✦ Successive samples are non-iid
2. Policy changes rapidly with Q-values
   - ✦ May oscillate; extreme data distributions
3. Scale of rewards and Q-values is unknown
   - ✦ Naive gradients can be large; unstable backpropagation

Based on material by David Silver

Deep Q-Networks

1. Use experience replay
   - ✦ Break correlations, learn from past policies
2. Target network to keep target values fixed
   - ✦ Avoid oscillations
3. Clip rewards
   - ✦ Provide robust gradients

Based on material by David Silver

Experience Replay

Equivalent to planning with empirical model

- Build dataset from agent's experience
  - Take action according to ε-greedy policy
  - Store *(x, a, r, x′, a′)* in replay memory *D*
  - Sample transitions from *D*, perform asynchronous update:

$$\mathcal{L}(\theta) = \mathop{\mathbf{E}}_{x,a,r,x',a' \sim \mathcal{D}} \left[\left(r + \gamma \max_{a' \in \mathcal{A}} Q(x', a', \theta) - Q(x, a, \theta) \right)^2 \right]$$

  - Effectively avoids correlations within trajectories

Based on material by David Silver

Target Q-Network

- To avoid oscillations, fix parameters of target in loss function
  - Compute targets w.r.t. old parameters

$$r + \gamma \max_{a' \in \mathcal{A}} Q(x', a', \theta^-)$$

- As before, minimize squared loss:

$$\mathcal{L}(\theta) = \mathbf{E}_{\mathcal{D}} \left[\left(r + \gamma \max_{a' \in \mathcal{A}} Q(x', a', \theta^-) - Q(x, a, \theta) \right)^2 \right]$$

- Periodically update target network:

$$\theta^- \leftarrow \theta$$

Similar to policy iteration!

Based on material by David Silver

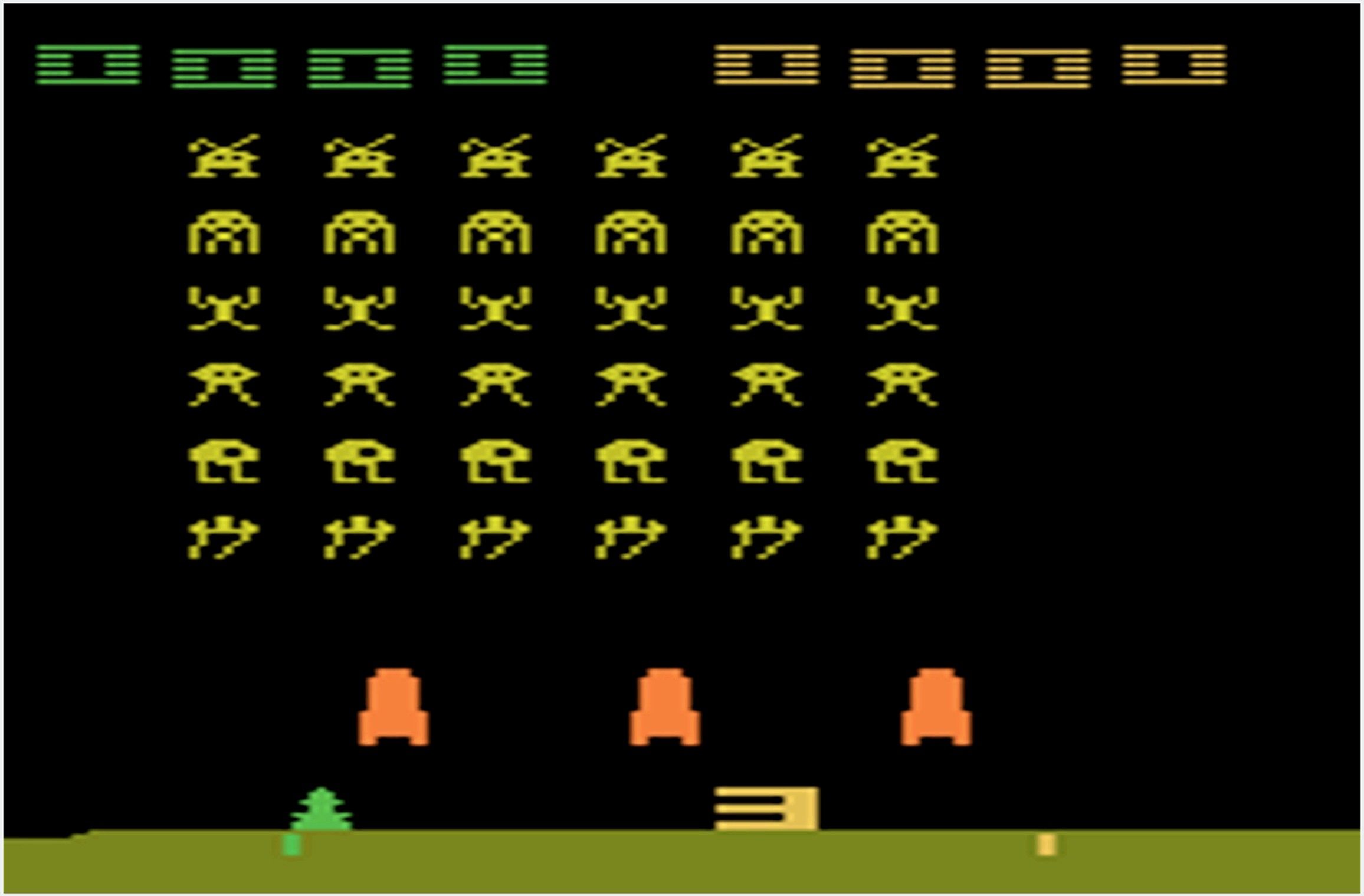
Clipping Rewards

- Clip rewards in range [-1, +1]
- Ensures gradients are well-conditioned
- Also prevents value overestimation
- No longer can tell small, large rewards apart

Based on material by David Silver

	Q-learning	Q-learning + Target Q	Q-learning + Replay	Q-learning + Replay + Target Q
Breakout	3	10	241	**317**
Enduro	29	142	831	**1006**
River Raid	1453	2868	4103	**7447**
Seaquest	276	1003	823	**2894**
Space Invaders	302	373	826	**1089**

Based on material by David Silver



Some Recent Research

Active Research: Off-policy Methods

- Reusing data (e.g. from experience replay) can diverge with approximation:

$$Q(x,a,\theta) \overset{\nabla_\theta}{\leftarrow} r(x,a) + \gamma \underset{\substack{a \sim \mu \\ x' \sim P_a}}{\mathbf{E}} \max_{a' \in \mathcal{A}} Q(x',a',\theta)$$

- Can use importance sampling ratio: $\dfrac{\pi(a \mid s)}{\mu(a \mid s)}$
- But variance is high
- Also safety issues: how to guarantee performance?

Precup, Sutton, and Singh (2000)
Thomas and Brunskill (2016)

Active Research: Multi-step Methods

- Greater accuracy [1] from multi-step returns:

$$\mathcal{T}^{\lambda}Q(x,a) := \sum_{k=0}^{\infty}\lambda^{k}\Big[\underbrace{\sum_{t=0}^{k}\gamma^{t}r(x_t,a_t)+\gamma^{k+1}Q(x_{k+1},a_{k+1})}_{\text{n-step return}}\Big]$$

$$= Q(x,a)+\sum_{t=0}^{\infty}(\lambda\gamma)^{t}\delta(x_t,a_t)$$

- Retrace(λ) [2] both off-policy and multi-step

$$\mathcal{R}Q(x,a) := Q(x,a)+\sum_{t=0}^{\infty}(\lambda\gamma)^{t}\Big(\prod_{s=0}^{t-1}c_s\Big)\delta(x_t,a_t) \quad c_s := \min\Big\{1,\frac{\pi(a_s\,|\,x_s)}{\mu(a_s\,|\,x_s)}\Big\}$$

- Convergence surprisingly nontrivial, even without value approximation

[1] Tsitsiklis and Van Roy (1997)
[2] Munos, Stepleton, Harutyunyan, Bellemare (2016)

Active Research: Gap-Increasing Operators

- Action gap: $\max_{a' \in \mathcal{A}} Q^*(x, a') - Q^*(x, a)$

- New operators that increase action gap, e.g.

$$\tilde{\mathcal{T}} Q(x, a) := \mathcal{T} Q(x, a) - \beta \left[\max_{a' \in \mathcal{A}} Q(x, a') - Q(x, a) \right], \quad \beta \in [0, 1)$$

  - ✦ Not necessarily contraction operators
  - ✦ Suboptimal Q-values may not converge

- Yet: guaranteed convergence:

$$\lim_{k \to \infty} \max_{a \in \mathcal{A}} (\tilde{\mathcal{T}})^k Q(x, a) = \max_{a \in \mathcal{A}} Q^*(x, a)$$

Bellemare, Ostrovski, Guez, Thomas, Munos (2016)

In Conclusion

Stochastic approximation

Policy evaluation

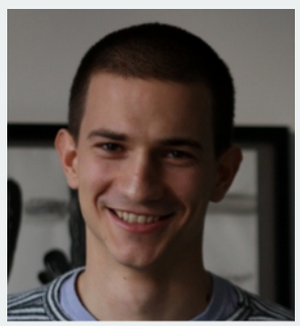
Optimal control

Function approximation

Model-Free Control with Deep Learning

MARC G. BELLEMARE

G. Ostrovski



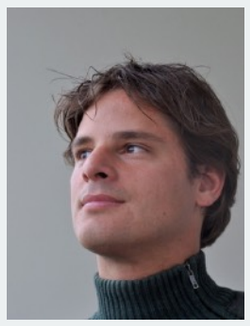
Arthur Guez

D. Saxton

Rémi Munos

T. Stepleton

S. Srinivasan

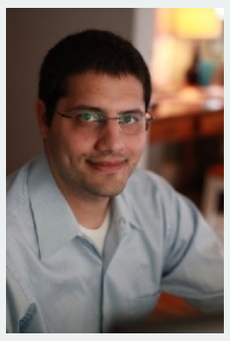


T. Schaul

J. Veness

Y. Naddaf

M. Bowling



Philip Thomas

A. Harutyunyan