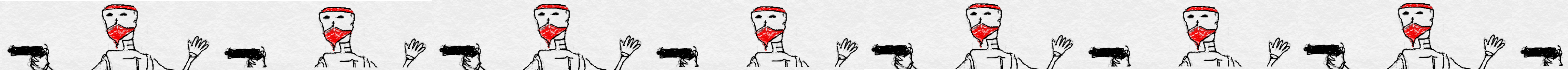

EPYX
PRESENTS
IMPOSSIBLE MISSION
BY DENNIS CASWELL
SPEECH SYNTHESIS BY
ELECTRONIC SPEECH SYSTEMS
©1984 EPYX



RLSS'17

PRESENTS

THEORY OF REINFORCEMENT LEARNING

BY

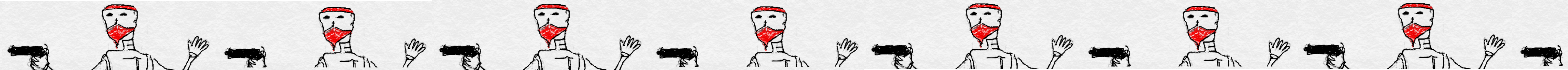
Csaba Szepesvári

UNIVERSITY OF ALBERTA

DeepMind

Contents

- Part 1: Why and what?
- Part 2: Batch learning
- Part 3: When you have a simulator
- Part 4: No simulator; learning ”out there”



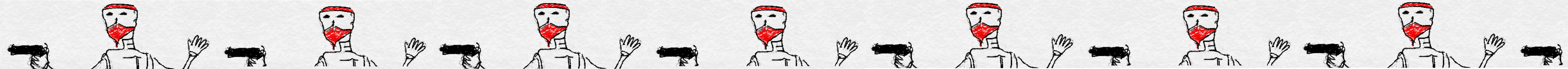
What we will not cover

- 95% of what exist out there
- We will only cover
  - Simplest tasks
  - Principles
  - Illustrate hurdles to overcome

(13 slides)

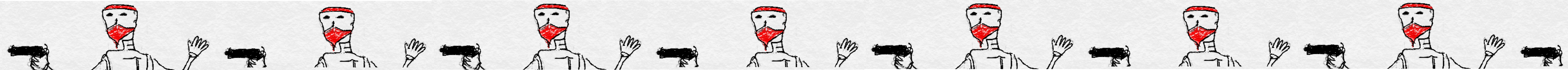
What and why?

..no slinking yet!



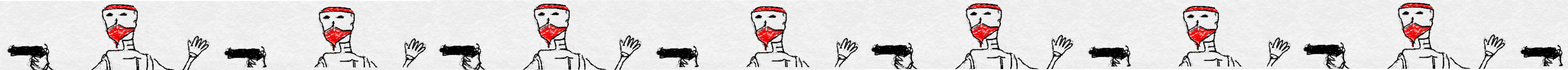
What and why?

- What do you mean by “theory”?
- ~~What do you mean by “RL”~~?
- Who needs theory?
- How does learning theory work?



What is a “theory” (for us)?

- Models
  - Mathematical
- Predictions
  - .. about how things will turn out to be; aka performance “bounds”



Who do you want to be?[1]

$$\nabla \cdot \mathbf{E} = 0$$
$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$
$$\nabla \cdot \mathbf{B} = 0$$
$$\nabla \times \mathbf{B} = \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t}$$

Guglielmo Marconi (1874—1937)

James Clerk Maxwell (1831–1879)

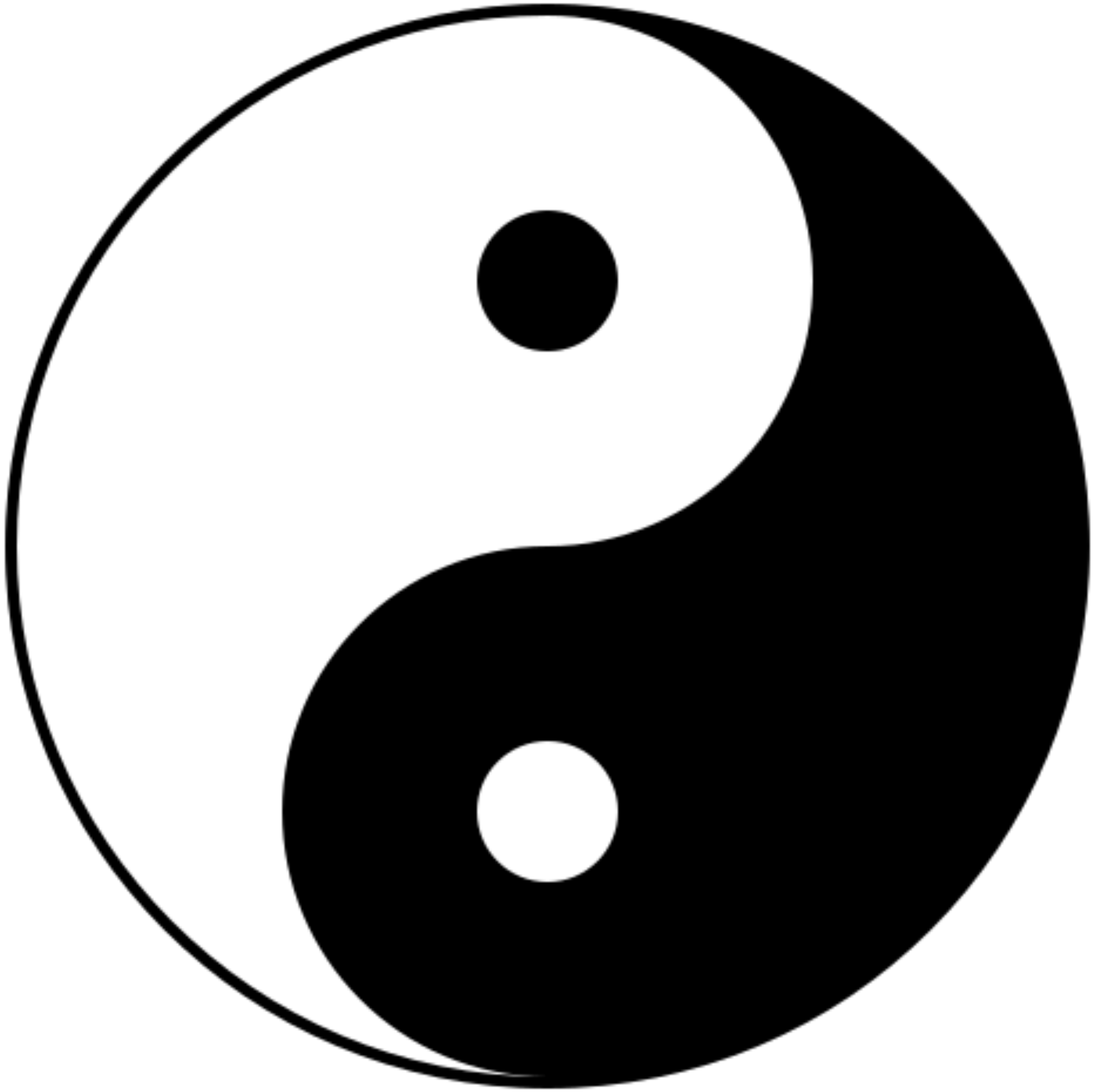
[1]Abraham Flexner: The usefulness of useless knowledge. Harpers, 1939

I won't do theory. Should I care?

- Yes!
- Predictions/theory help you to..
  - Design algorithms
  - Understand their behavior
  - Quantify knowledge/uncertainty
  - Identify new/refine old challenges

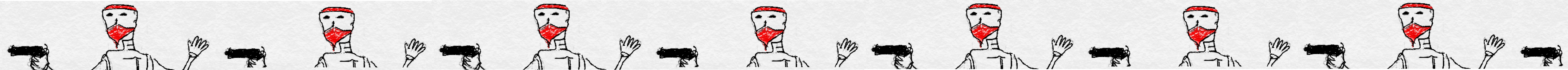
SOFIE LINDBERG

Theory and practice



Statistical learning theory: ingredients

- Distributions $P \in \mathcal{P}$
- i.i.d. samples $S_n \sim P^n$
- Learning algorithms $A : S_n \mapsto h$
- Predictors $\mathcal{H}$
- Loss functions $\ell : \mathcal{H} \to [0, \infty)$

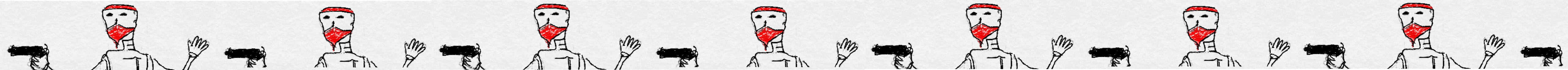


What to predict?

- A priori analysis:
  How well a learning alg. will perform on new data
- A posteriori analysis:
  How well is a learning alg. doing on some data? Quantify uncertainty left

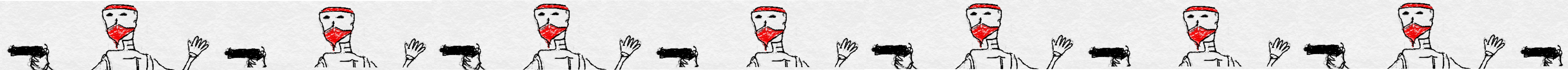
A priori analysis

- Problem #1:
  - Can we compete with best hypothesis from a given set of “hypotheses”?
  - Vapnik’s learning theory
- Problem #2:
  - Can we match the best possible loss assuming the data generating distribution belongs to a known family?
  - [non-]parametric statistics
- Problem #3:
  - Does algorithm X achieve Y?



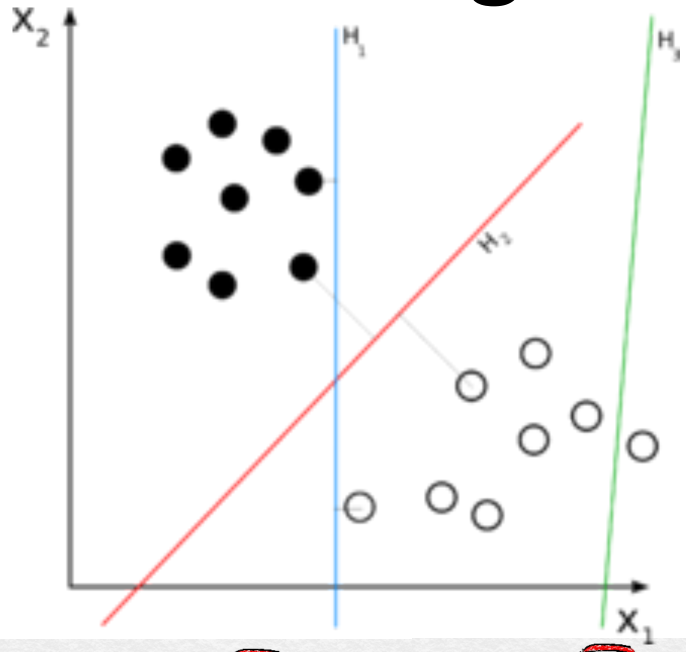
A posteriori analysis

- Quantify uncertainty of prediction loss
- Analyze methods like cross-validation (how big should the error bars be!?)
- Design “self-bounded” algorithms (ala Yoav Freund)



Two fundamental results in SLT

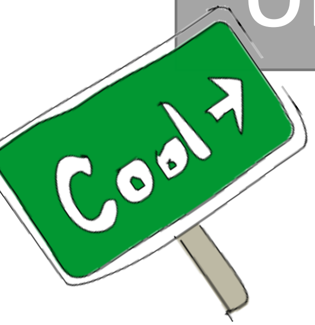
- Fundamental theorem of SLT
- The computational complexity of learning linear classifiers



(Problem #1)

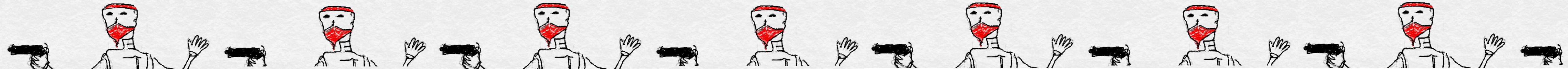
The fundamental theorem of SLT

- Theorem[1]: In binary classification, to match the loss of best hypothesis in class $\mathcal{H}$ up to accuracy ϵ, one needs $\widetilde{\Theta}(\frac{\mathrm{VC}(\mathcal{H})}{\epsilon^2})$ observations.

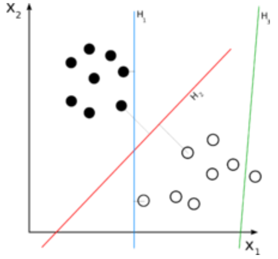


- Pure information theory, "ERM"

[1]http://www.cs.ox.ac.uk/people/varun.kanade/teaching/AML-HT2017/lectures/lecture09.pdf



Computational complexity



- **Theorem**[1]: Unless NP=RP, linear classifiers (hyperplanes!) cannot be learned in polynomial-time.

- What now?
- Hah, we can change the problem!

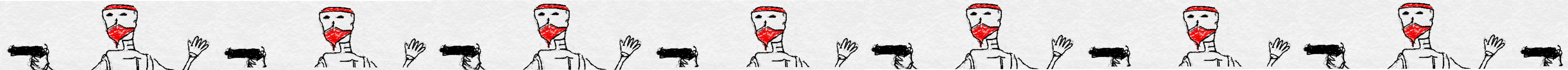
[1] http://www.cs.ox.ac.uk/people/varun.kanade/teaching/AML-HT2017/lectures/lecture09.pdf

Questions?

(6 slides)

Batch learning

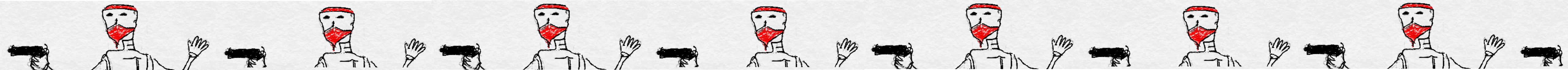
..can we copy supervised learning?



Batch RL: The learning problem

- Data:
  - $(X_t, A_t, Y_t, R_t)_{t=1}^{N}$ iid where

    $$X_t \sim \mu, A_t \sim \pi(\cdot\,|X_t), Y_t \sim P_{A_t}(\cdot|X_t),$$
    $$R_t = r(X_t, A_t, Y_t),$$
  - H: horizon
  - Π: class of policies
- Goal: Find ϵ-optimal policy in Π.



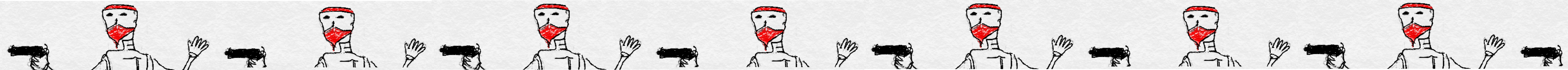
Batch RL and supervised learning

- Recall the value of Markov policy π:

$$V_\pi(x) = \sum_{t=0}^{H} P_\pi^t r_\pi \,.$$

Here, P_π is Markov transition matrix (“kernel”) under π, while r_π is the reward vector (“function”).

- Corollary 1: For $H = 0$, batch RL is “cost sensitive classification” with cost $-r(x, a)$ at input x and “label” a and “hypothesis class” Π.



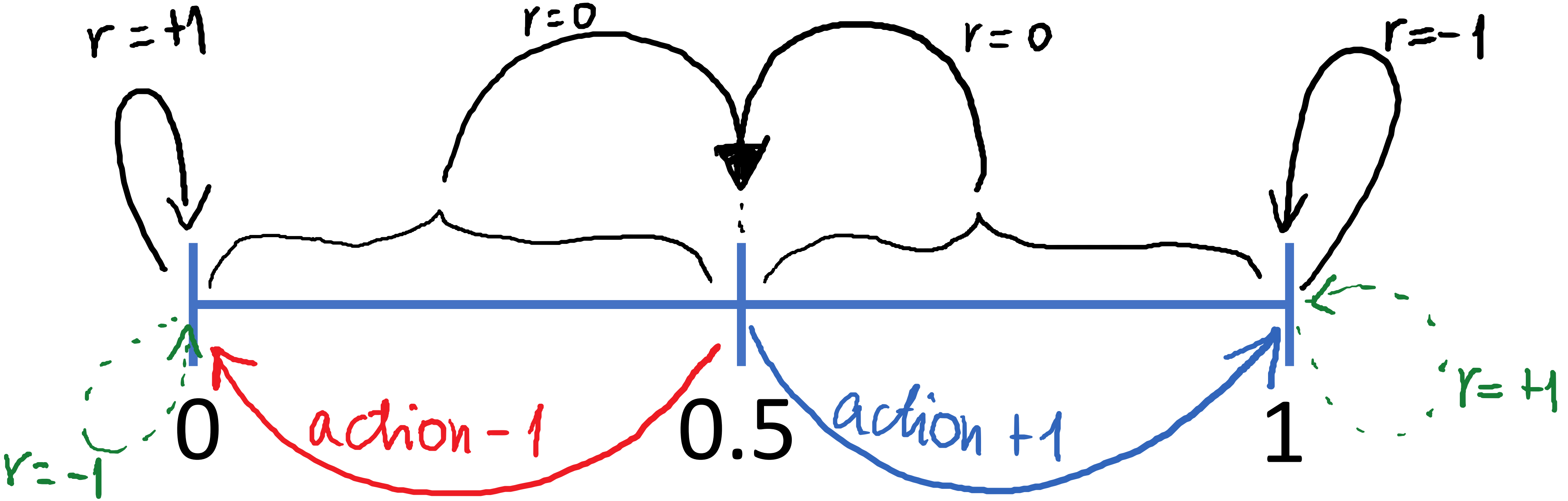
Batch RL and supervised learning

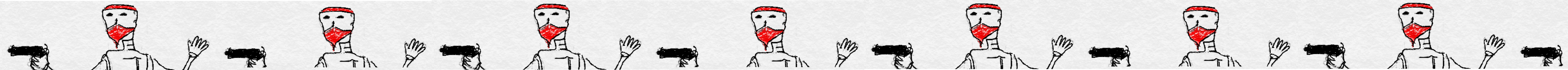
- Corollary 1: For $H = 0$, batch RL is "cost sensitive classification" (CSS) with cost $-r(x, a)$ at input x and "label" a and "hypothesis class" Π.
- Corollary 2: The "Batch RL" learning problem is at least as hard as CSS
- CSS: cost is typically uniform (no dependence on input), and is known.
- CSS with unknown cost function: SLT does not consider this

Batch RL with nontrivial horizons

$\Pi = \{\pi_\theta \mid \pi_\theta(x) = \mathrm{sgn}(x-\theta), \theta \in [0,1]\}$

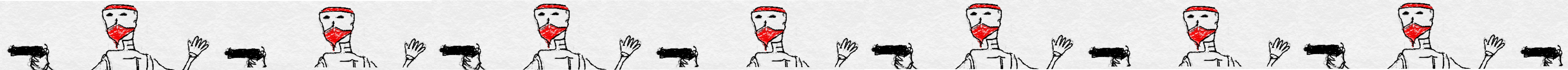
- Theorem: For $H = 2$, the sample complexity of batch RL is "infinite".



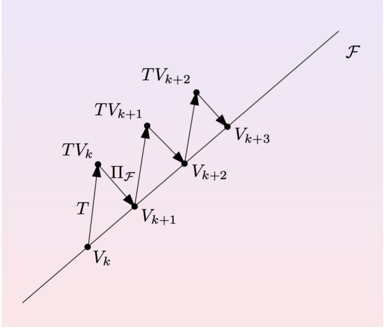


What is the problem?

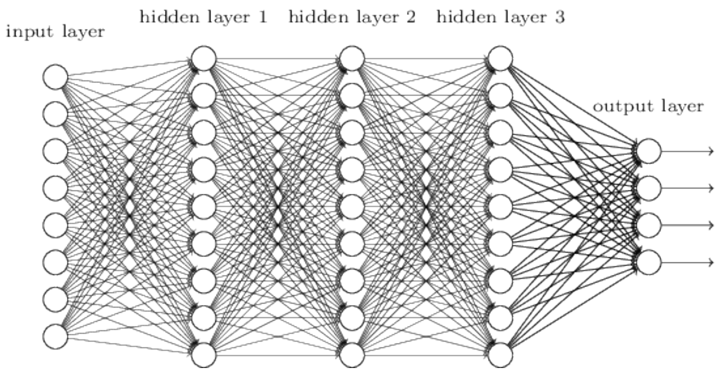
- Critical decision at 0.5, but in the data, 0.5 does not appear!
- What’s next?
  - “Better sampling distributions”; e.g. 0.5 should be in the data!
    - But in fact all “keyhole states” should be in the data!? Too much?



A “generic” recipe for positive result

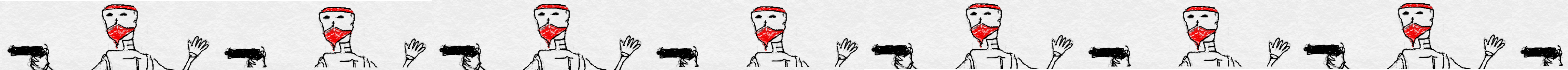


- Write approximate value iteration as
$$Q_{t+1} = TQ_t + \epsilon_t$$
- If all the errors ϵ_t are “small”, then the greedy policy w.r.t. Q_T will not be “too bad”
- How to control errors?
- How many iterations (T=?)?



Questions?

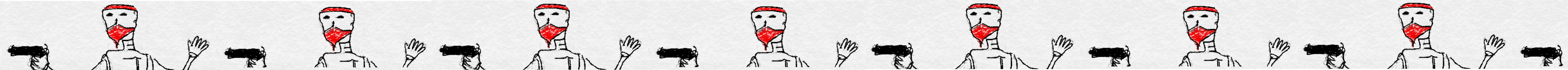
..are you ready for the next run..?



(11 mins)

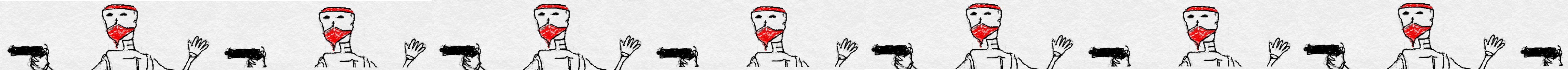
..when you have a simulator

..anyone wants to play Atari games?



Planning problem

- Given a huge MDP, goal is to compute a:
  - Good policy (from Π); or
  - A good action of a good policy from Π at a given state x.
- Which one is easier?
- Computational problem!



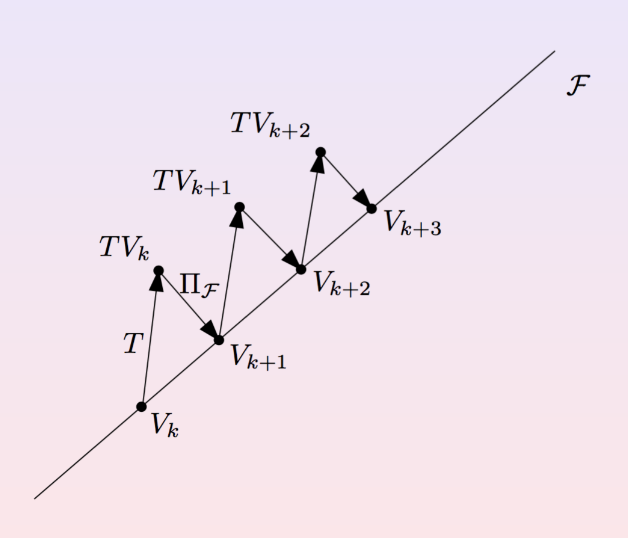
Working with large MDPs

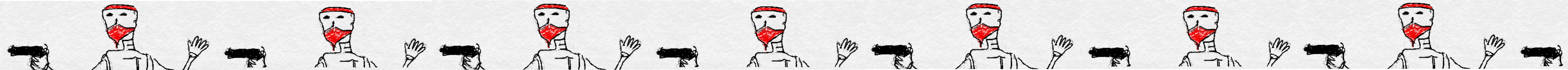
- Deterministic access:
  - Can ask for transition probabilities/densities $p(y|x,a)$, rewards $r(x,a)$ for any (x,a,y).
- Stochastic access/ “generative model”/simulator access:
  - Can ask for simulating transitions/rewards at any (x,a).
  - Can ask to generate states from μ
  - Can ask for simulating transitions/rewards at any (x,a) for x reached earlier.

Fitted Value Iteration

Sampling based fitted value iteration – multi-sample variant

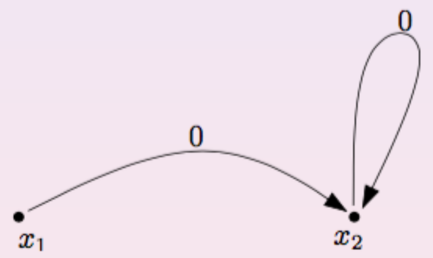
function **SFVI-MULTI**$(N, M, K, \mu, \mathcal{F}, P, S)$
  $V \leftarrow 0$ // approximate value function
  for $k = 1$ to K **do**
    for $i = 1$ to N **do**
      Draw $X_i \sim \mu$, $Y_j^{X_i,a} \sim P(\cdot|X_i, a)$, $R_j^{X_i,a} \sim S(\cdot|X_i, a)$,
      $(j = 1, \ldots, M,\ a \in \mathcal{A})$
    end for
    $\hat{V}_i \leftarrow \max_{a \in \mathcal{A}} \left\{ \frac{1}{M} \sum_{j=1}^{M} \left(R_j^{X_i,a} + \gamma V(Y_j^{X_i,a}) \right) \right\}$
    $V \leftarrow \mathrm{argmin}_{f \in \mathcal{F}} \sum_{i=1}^{N} (f(X_i) - \hat{V}_i)^2$ // fitting
  end for
  return V





New problem: Instability

- Tsitsiklis & Van Roy (1996)
- State space: $\mathcal{X} = \{x_1, x_2\}$
- Dynamics:



- Bellman operator:

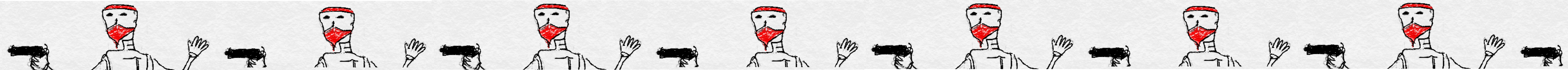
$$\begin{aligned}(TV)(x_1) &= 0 + \gamma V(x_2)\\ (TV)(x_2) &= 0 + \gamma V(x_2).\end{aligned}$$

- Function-space:
$\mathcal{F} = \{\theta\phi \,|\, \theta \in \mathbb{R}\}$,

$$\phi(x_1) = 1,\ \phi(x_2) = 2.$$

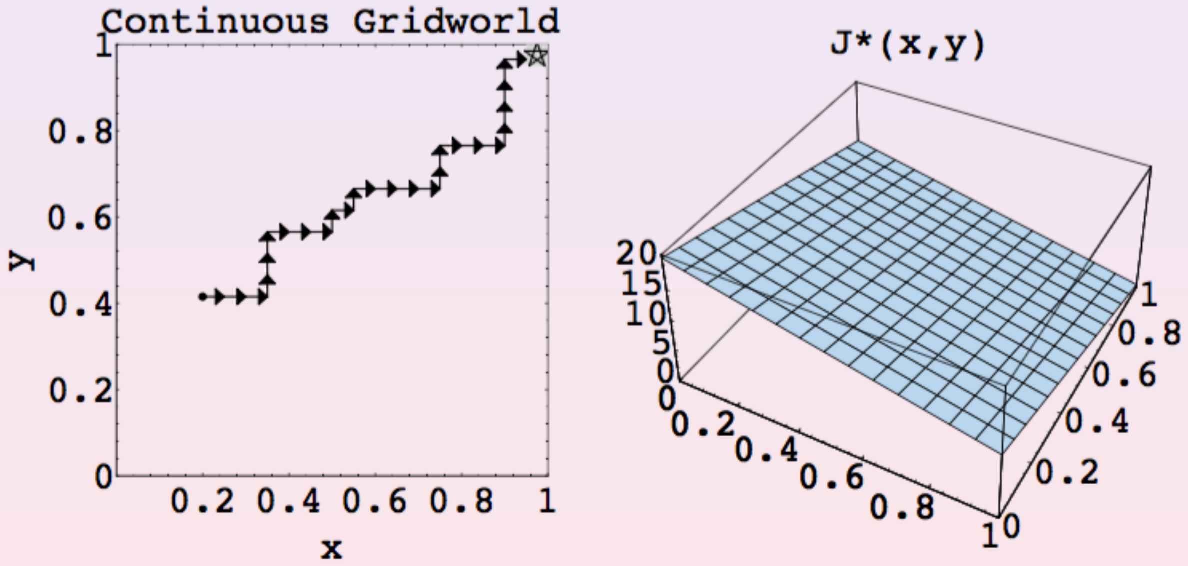
Iteration:

$$\begin{aligned}\theta_{t+1} &= \mathrm{argmin}_\theta \|\theta\phi - T(\theta_t\phi)\|_2\\ &= \mathrm{argmin}_\theta (\theta - \gamma 2\theta_t)^2 + (2\theta - \gamma 2\theta_t)^2 = (6/5\gamma)\theta_t \to +\infty\end{aligned}$$

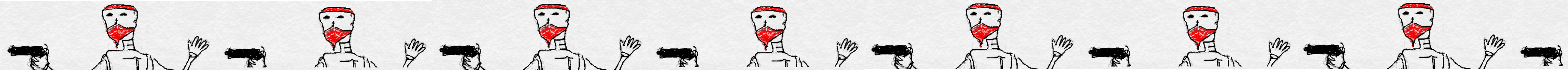


Disaster strikes

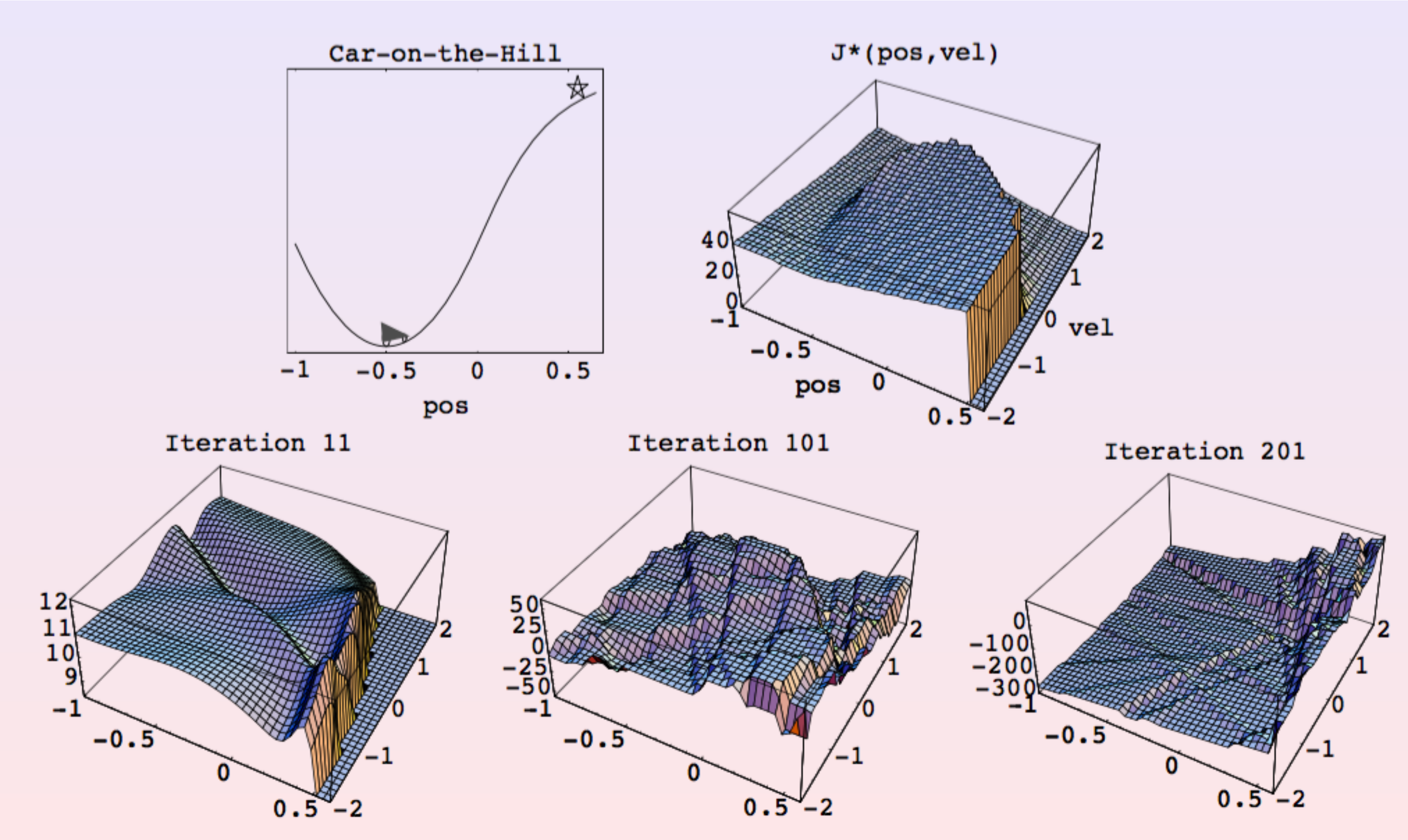
From: Boyan & Moore: “Generalization in Reinforcement Learning: Safely Approximating the Value Function”, *NIPS-7*, 1995.



[1] With thanks to Justin Boyan



..and with neural nets



Conclusions..?

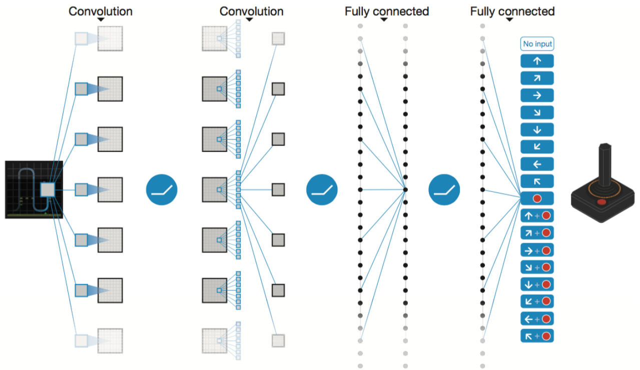
- *"In light of these experiments, we conclude that the straightforward combination of DP and function approximation is not robust."* (Boyan & Moore, NIPS-7, 1995)
- *Unfortunately, many popular functions approximators, such as neural nets and linear regression, do not fall in this*[2] *class (and in fact can diverge).* (G. Gordon, ICML, 1995).

Pushing it harder

[Sz., Munos 2005]

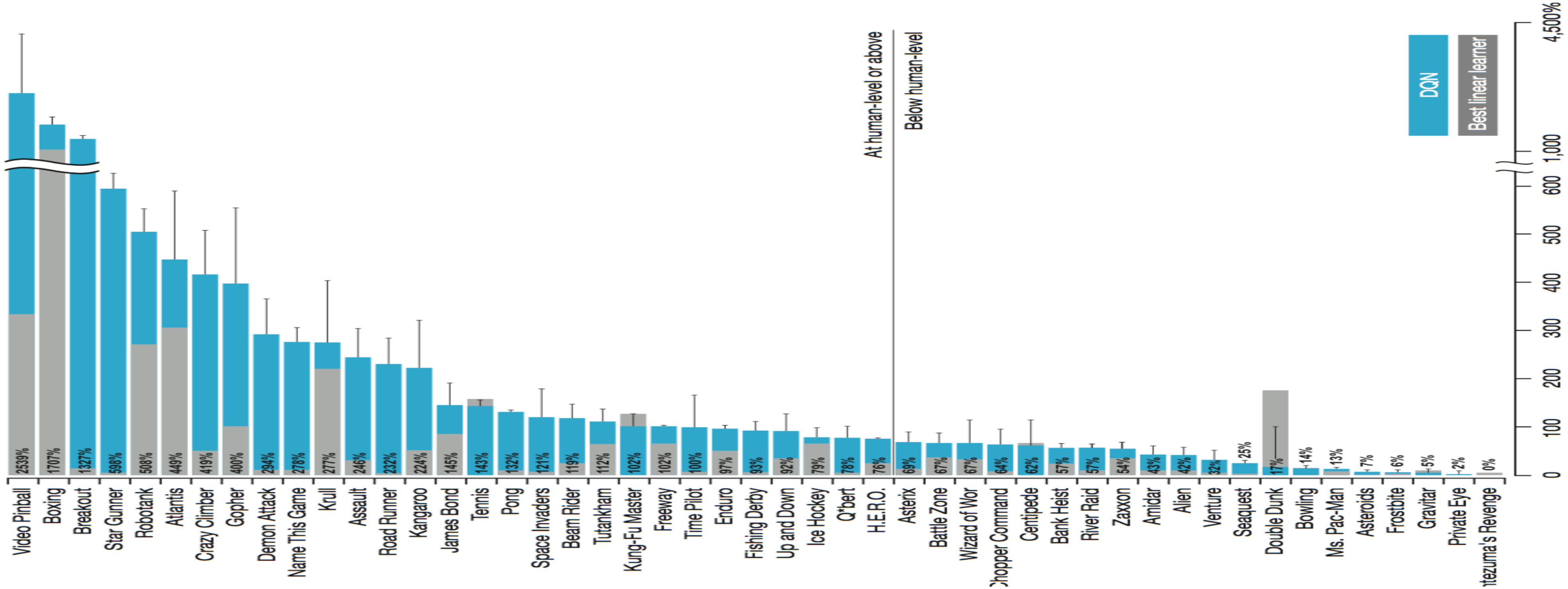
$$
\|V^* - V^{\pi_K}\|_{p,\rho} \le
\frac{2\gamma}{(1-\gamma)^2} \Bigg\{ C(\mu)^{1/p} \Bigg[d(T\mathcal{F},\mathcal{F}) +
c_1 \left(\frac{\mathcal{E}}{N} (\log(N) + \log(K/\delta)) \right)^{1/2p} +
c_2 \left(\frac{1}{M} (\log(N|A|) + \log(K/\delta)) \right)^{1/2} \Bigg] +
c_3 \gamma^K K_{\max} \Bigg\}
$$

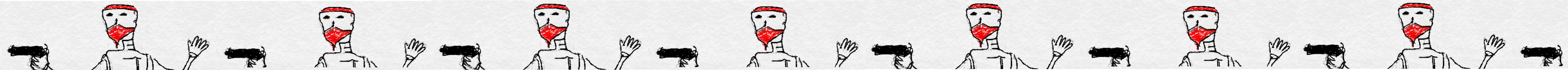
From FVI to DQN



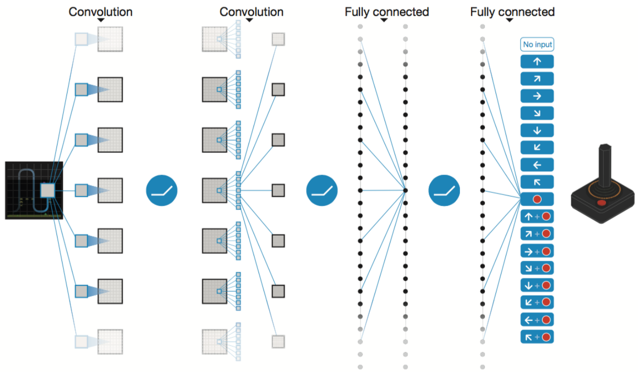
Human-level control through deep reinforcement learning

Volodymyr Mnih[1]*, Koray Kavukcuoglu[1]*, David Silver[1]*, Andrei A. Rusu[1], Joel Veness[1], Marc G. Bellemare[1], Alex Graves[1], Martin Riedmiller[1], Andreas K. Fidjeland[1], Georg Ostrovski[1], Stig Petersen[1], Charles Beattie[1], Amir Sadik[1], Ioannis Antonoglou[1], Helen King[1], Dharshan Kumaran[1], Daan Wierstra[1], Shane Legg[1] & Demis Hassabis[1]

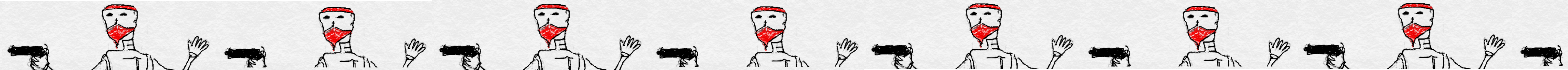




From FVI to DQN

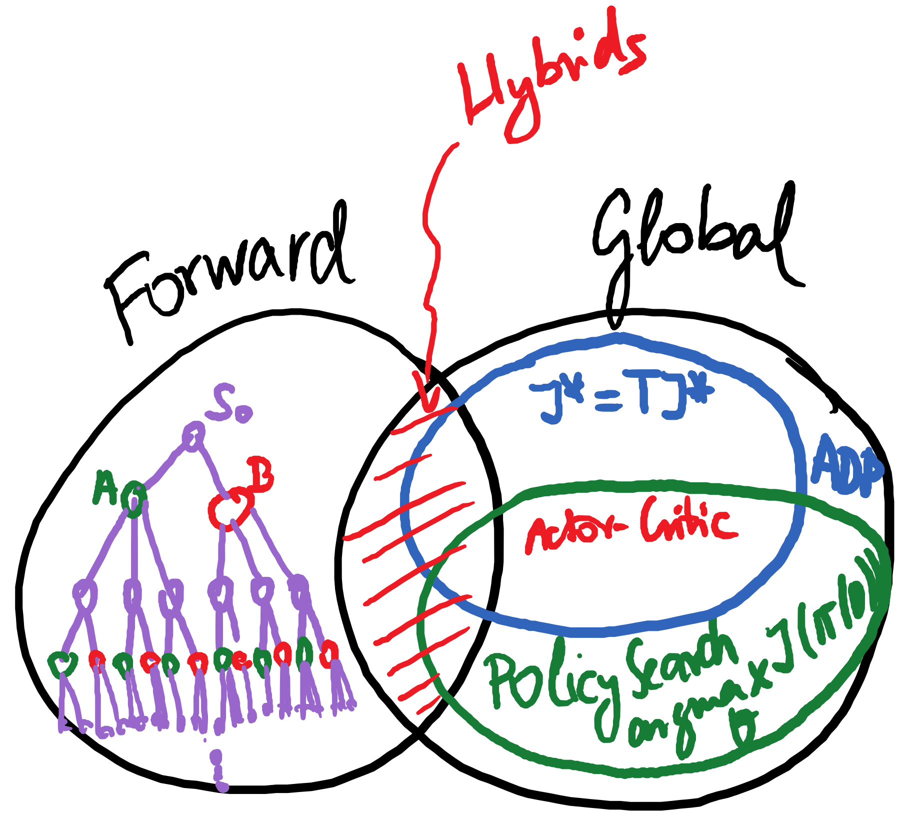


- How did this happen??
  - μ is not fixed, but is slowly(!) changed ("experience replay")
  - "Right" bias through convolutional neural nets
  - Better fit of data and better bias both explained by theory
- ..it'd be good to see some data published on the relative importance of the individual "tricks" used



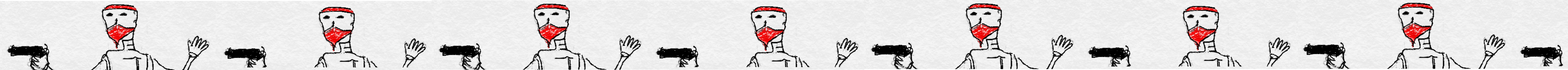
Map of planning methods

- Forward methods:
  - Lookahead tree building
- Global methods:
  - Approximate dynamic programming
  - Policy search
  - Hybrids
- Hybrid forward and global methods



Questions?

..are you ready for the next run..?



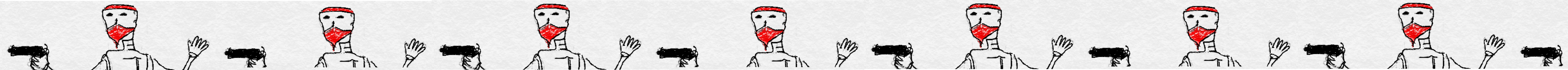
(12 slides)

..no simulator, no pain..? Uh..no..

When things became “real”

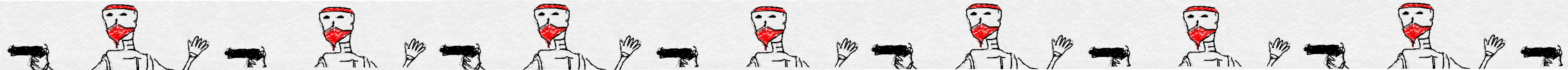
Defining online learning

- Interact with “real” system
- Collect as much reward as possible!
- Performance metric:
  - Total reward collected, or..
  - Regret: Difference to baseline (normalizing)
- PAC-MDP: not covered

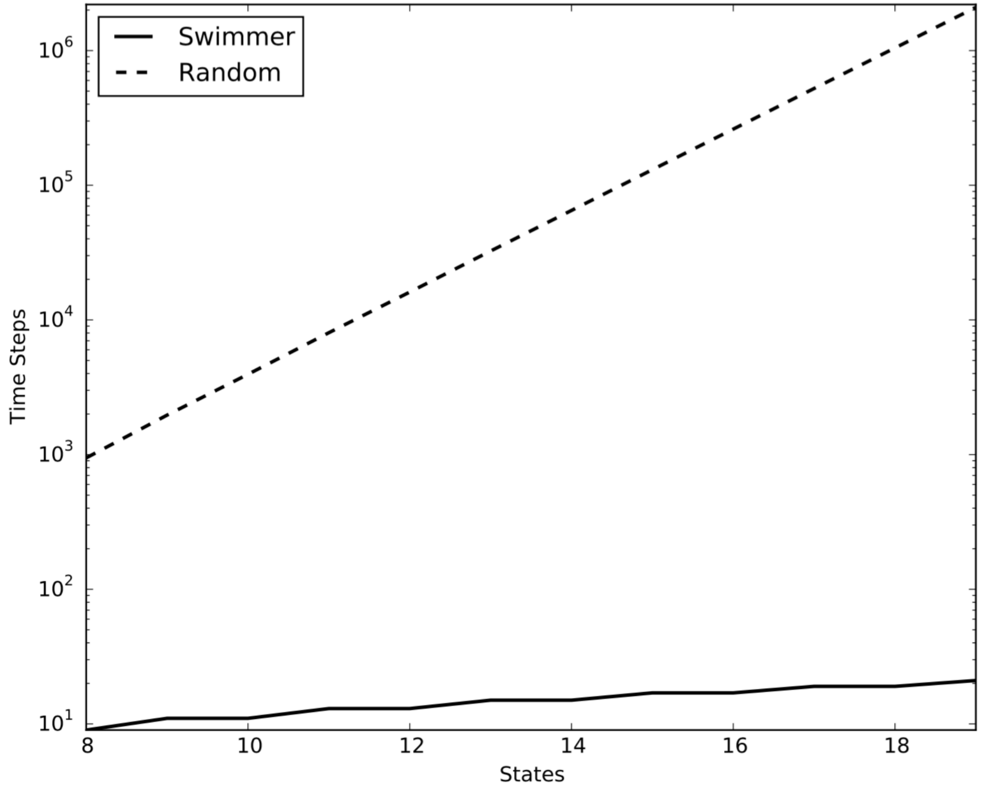


Why should you care?

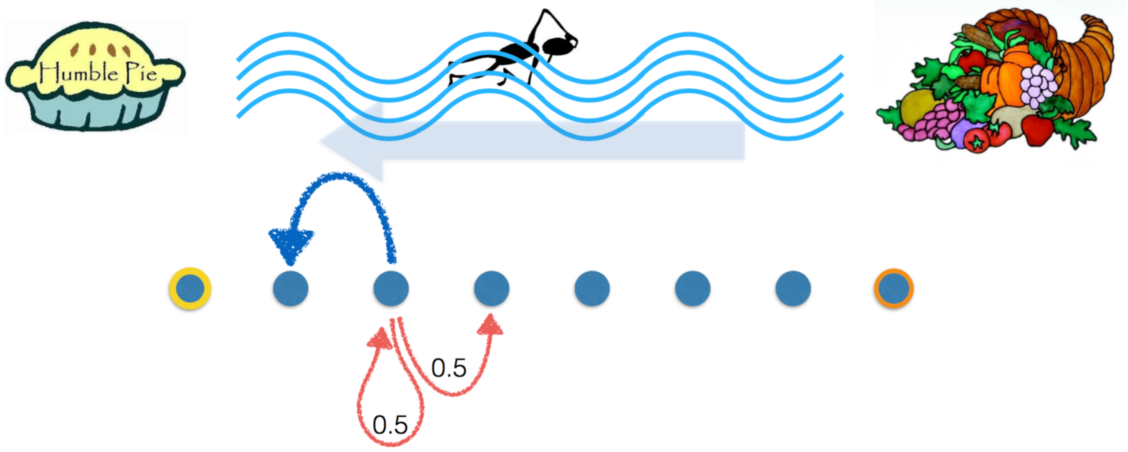
- **Alternative**: Model-based RL
  - Learn a model & use planning (see previous part)
- Problems with model-based RL:
  - Models can be too expensive to build
  - Uncontrolled model inaccuracies may lead to poor behavior
- **Opportunity**: Online learning can be cheaper
  - ..but.. online learning can and often does use model learning..



The challenge



time steps before bounty found using random and “swimmer” policies



- Problem #1:
  Random behavior is often ineffective in exploring the environment
- Problem #2:
  Biasing towards best policy found makes things much worse!

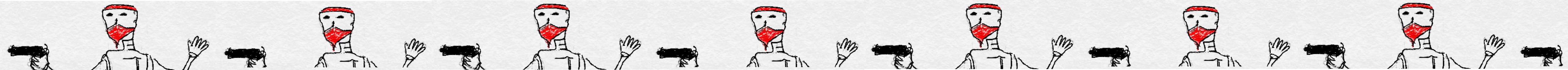
- Need: Principled way of trading off reward and uncertainty

“explore or exploit”?

banditalgs.com

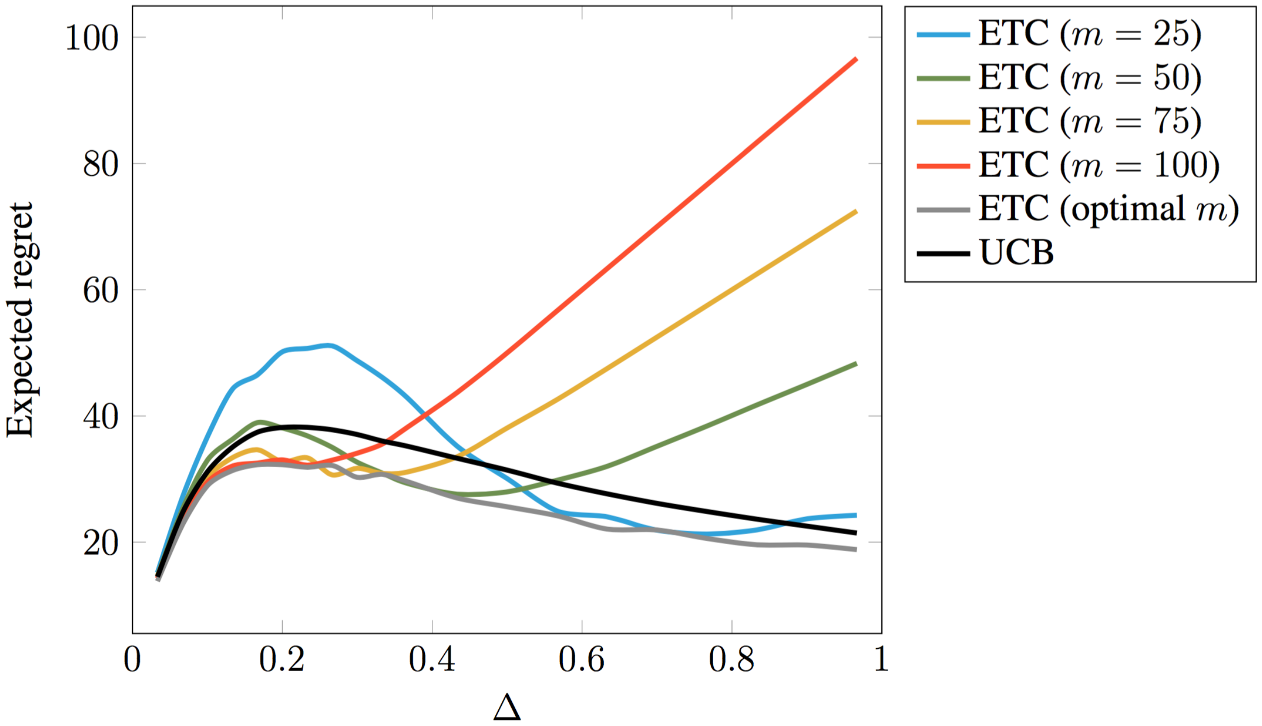
Warmup: Bandits/terminology

- Bandits = RL problem with a single state
- Contextual bandits: RL problem when next state is chosen at random independently of the action chosen
- Linear bandits: (Contextual) bandits when reward is linear in features of state-action pairs

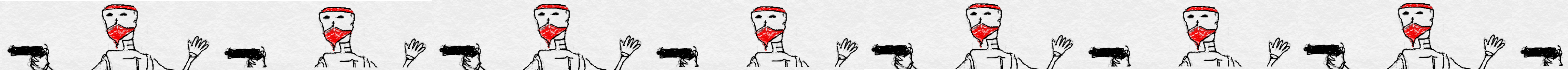


The key result on (stochastic) bandits

- Simple ϵ-greedy, Boltzmann/Gibbs, explore-then-commit (ETC) fail to adapt
- Optimistic algorithms (e.g., UCB) **adapt optimally**



2 arms, unit variance Gaussian rewards with means 0 and –Δ, horizon 1000



Optimism in the face of uncertainty

The optimism in the face of uncertainty principle states that one should choose their actions as if the environment is as nice as **plausibly possible**.

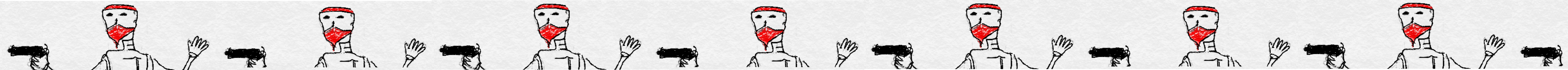
$$\mathrm{UCB}_i(t-1,\delta) \doteq \hat{\mu}_i(t-1) + \sqrt{\frac{2\log(1/\delta)}{T_i(t-1)}}\,.$$

Input K and δ

Choose each action once

For rounds $t > K$ choose action

$$A_t = \mathrm{argmax}_i\, \mathrm{UCB}_i(t-1,\delta)$$



An instance-dependent result

- <u>Theorem</u>: Assume rewards are Gaussian with unit variance or less, and unknown means. Set $\delta = 1/n^2$. Then, the expected regret R_n of UCB satisfies:

$$R_n \leq 3 \sum_{i=1}^{K} \Delta_i + \sum_{i:\Delta_i > 0} \frac{16 \log(n)}{\Delta_i} .$$

An instance-independent result

- **Theorem**: Using $\delta = 1/n^2$ as before, on any Gaussian unit variance environment, the expected regret of UCB satisfies

$$R_n \leq 8\sqrt{nK\log(n)} + 3\sum_{i=1}^{K}\Delta_i\,.$$

Lower bounds

- <u>Theorem</u>: The upper bounds shown above are optimal up to a constant factor.
  Further, by better tuning, UCB can be made strictly optimal in an asymptotic sense.

[Jaksch-Ortner-Auer,'10]

How about MDPs?

S states, ***A*** actions, rewards in [0,1].

<u>Definition:</u> Diameter := maximum of best travel times between pairs of states. River swim: ***D = S***

- **<u>Theorem:</u>** The regret of an OFU learner satisfies

$$R_T = \tilde{O}(DS\sqrt{AT})$$

- **<u>Theorem:</u>** For any algorithm,

$$R_T = \Omega(\sqrt{DSAT})$$

Principled methods for exploration

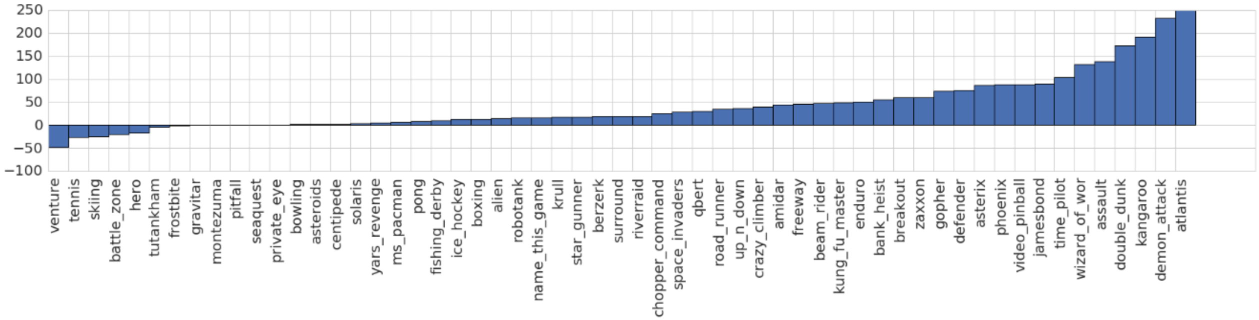
- Optimistic methods
- Posterior sampling
  - Follow-the-perturbed-leader
- Optimal sampling

Frontiers

30 Jun 2017

Noisy Networks for Exploration

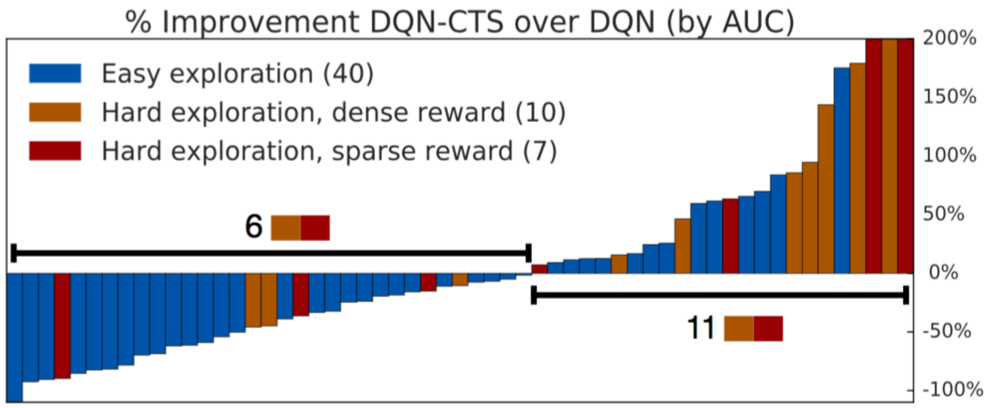
Meire Fortunato* Mohammad Gheshlaghi Azar* Bilal Piot *
Jacob Menick Ian Osband Alex Graves Vlad Mnih
Remi Munos Demis Hassabis Olivier Pietquin Charles Blundell
Shane Legg
DeepMind
{meirefortunato,mazar,piot,
jmenick,iosband,gravesa,vmnih,
munos,dhcontact,pietquin,cblundell,
legg}@google.com

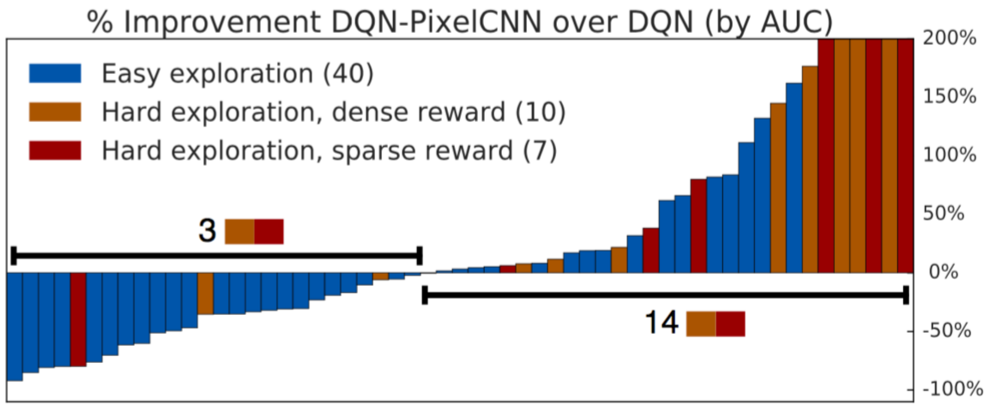


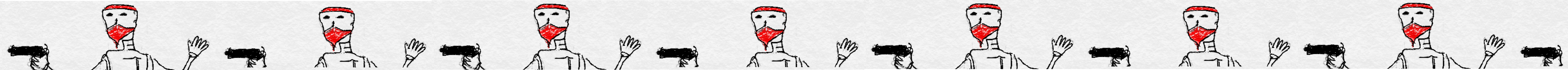
(a) Improvement in percentage of NoisyNet-DQN over DQN [21]

Count-Based Exploration with Neural Density Models

Georg Ostrovski [1] Marc G. Bellemare [1] Aäron van den Oord [1] Rémi Munos [1]

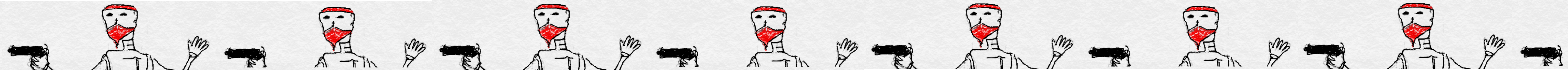






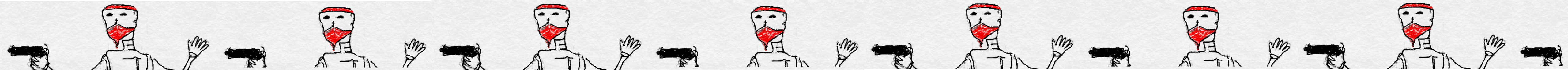
Questions?

..are you ready for the next run..?



Conclusions/summary

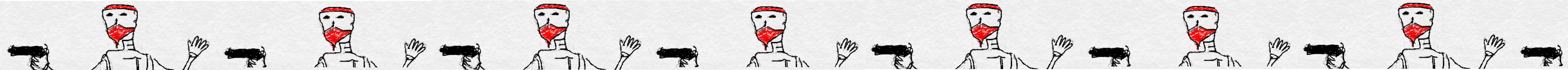
..we deserve that break, don't we?



- Mathematical Model+Predictions = Theory
- Theory can help practice, empirical work inspires/ignites theory work
- RL $\neq$ Supervised Learning
  - Information mismatch
  - Computation
  - Batch, simulation, online

- Not touched: mixing & uncertainty quantification, beyond MDPs, why probabilities and many others

- The unique distinguishing feature of theory:
  - Negative results (aka lower bounds)
- What to do with negative results?
  - Remember them!
  - Twist problem to be solved
- “Bad theory”
  - ~~Incorrect proofs~~
  - Bad modeling assumptions



Questions?