imitation learning

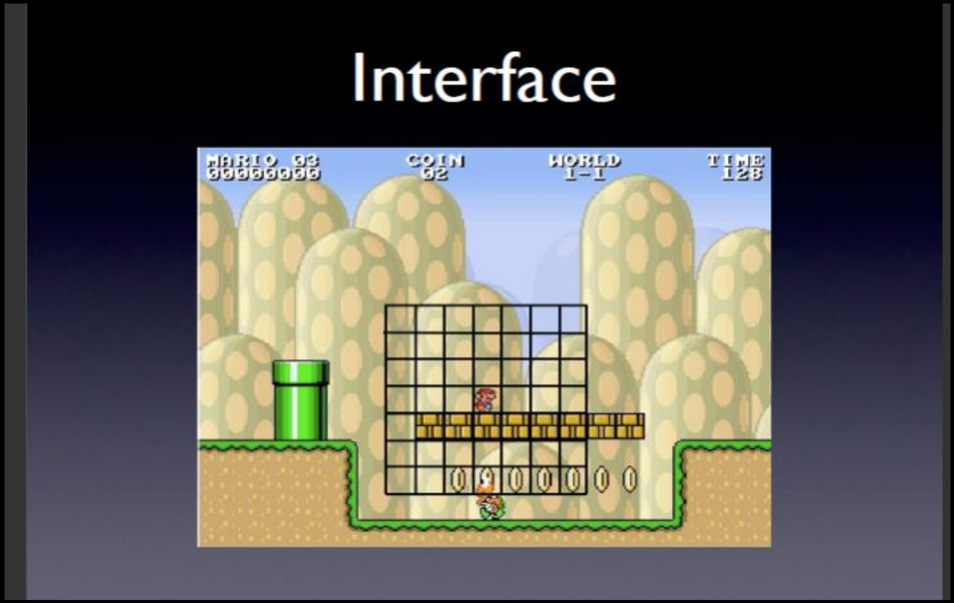
Hal Daumé III | U Maryland / Microsoft Research | me@hal3.name | @haldaume3

1. Learning from reinforcement alone is hard
   1. Exploration is hard
   2. Credit assignment is hard
   3. Designing rewards is hard

2. Yet people are pretty good at many tasks
3. Perhaps we can use them to help

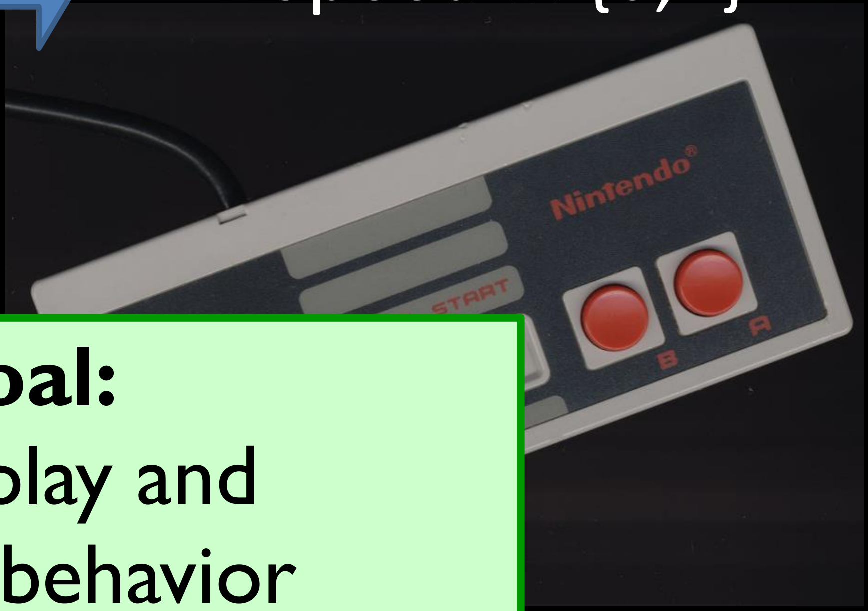
From Mario AI competition 2009

Input:



Output:

Jump in {0,1}
Right in {0,1}
Left in {0,1}
Speed in {0,1}



High level goal:
Watch an expert play and
learn to mimic her behavior

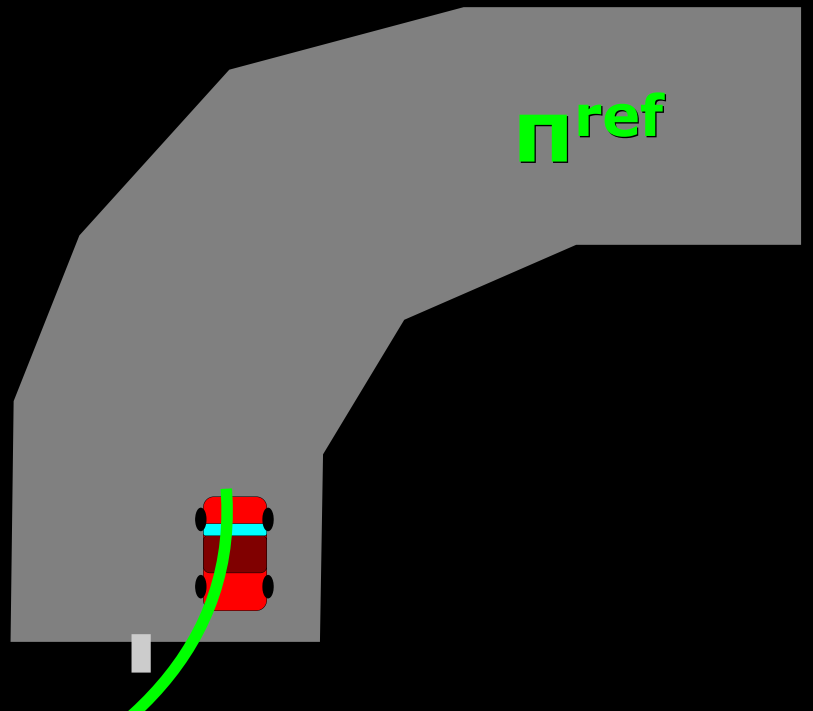
Video credit: Stéphane Ross, Geoff Gordon and Drew Bagnell

$\pi($

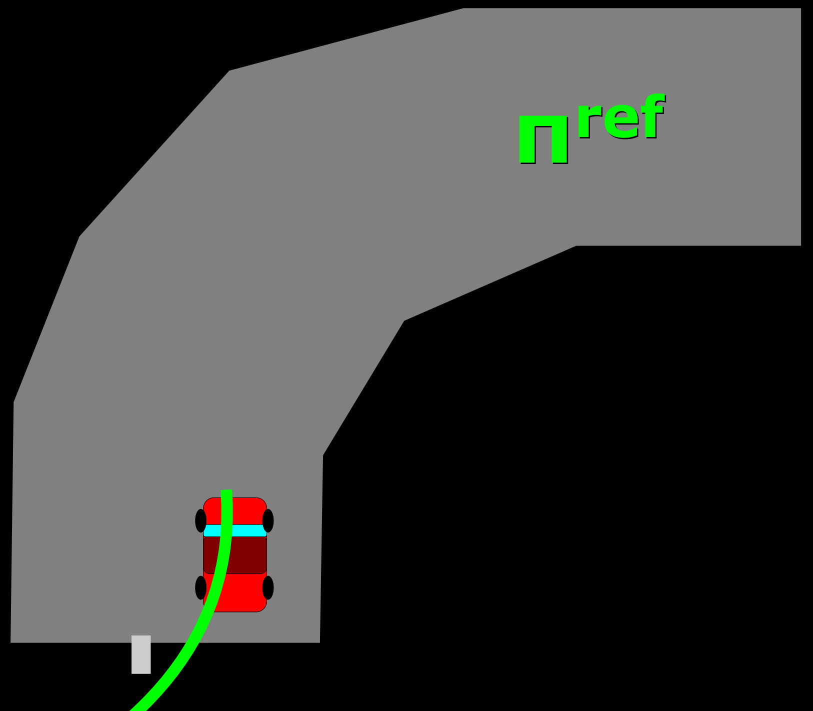
$=a$

1. Collect trajectories from expert π^{ref}
2. Store dataset **D** = $\{ (o, \pi^{ref}(o)) \mid o \sim \pi^{ref} \}$
3. Train classifier π on **D**

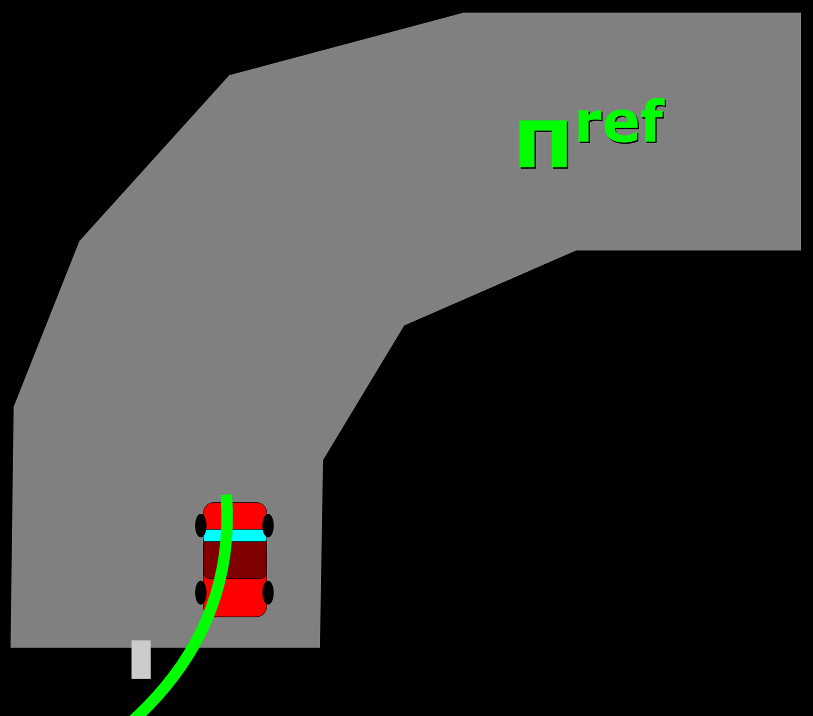
.Let π play the game!



Video credit: Stéphane Ross, Geoff Gordon and Drew Bagnell



π^{ref}


π^{ref}

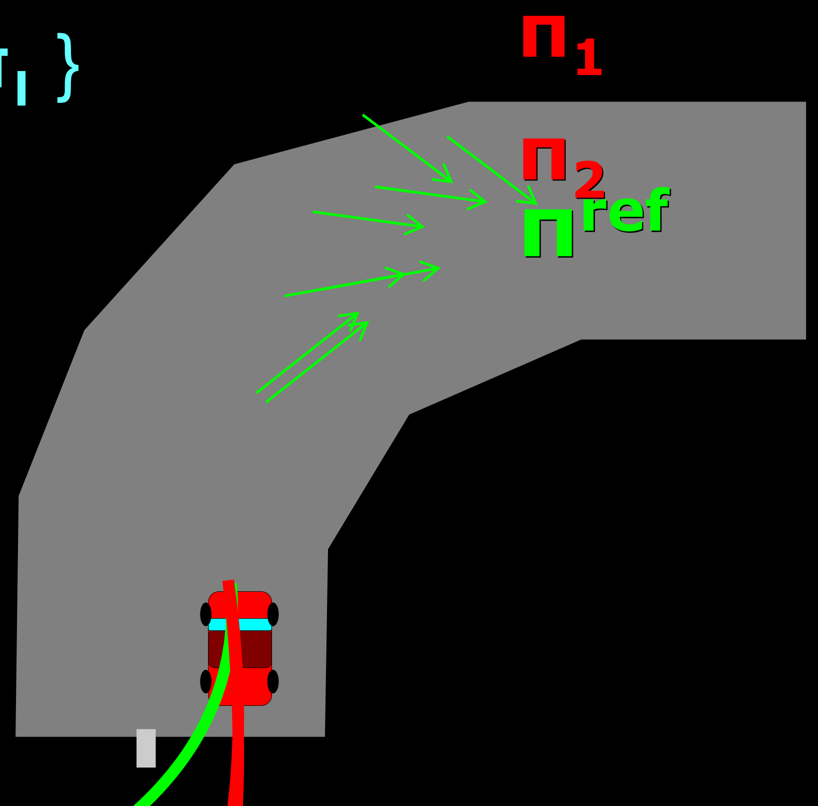
1. Collect trajectories from expert π^{ref}
2. Dataset $D_0 = \{ (o, \pi^{ref}(o,y)) \mid o \sim \pi^{ref} \}$
3. Train π_1 on D_0
4. Collect new trajectories from π_1
   - But let the *expert* steer!
5. Dataset $D_1 = \{ (o, \pi^{ref}(o,y)) \mid o \sim \pi_1 \}$
6. Train π_2 on $D_0 \cup D_1$

- In general:
- $D_n = \{ (o, \pi^{ref}(o,y)) \mid o \sim \pi_n \}$
- Train π_{n+1} on $\cup_{i \le n} D_i$

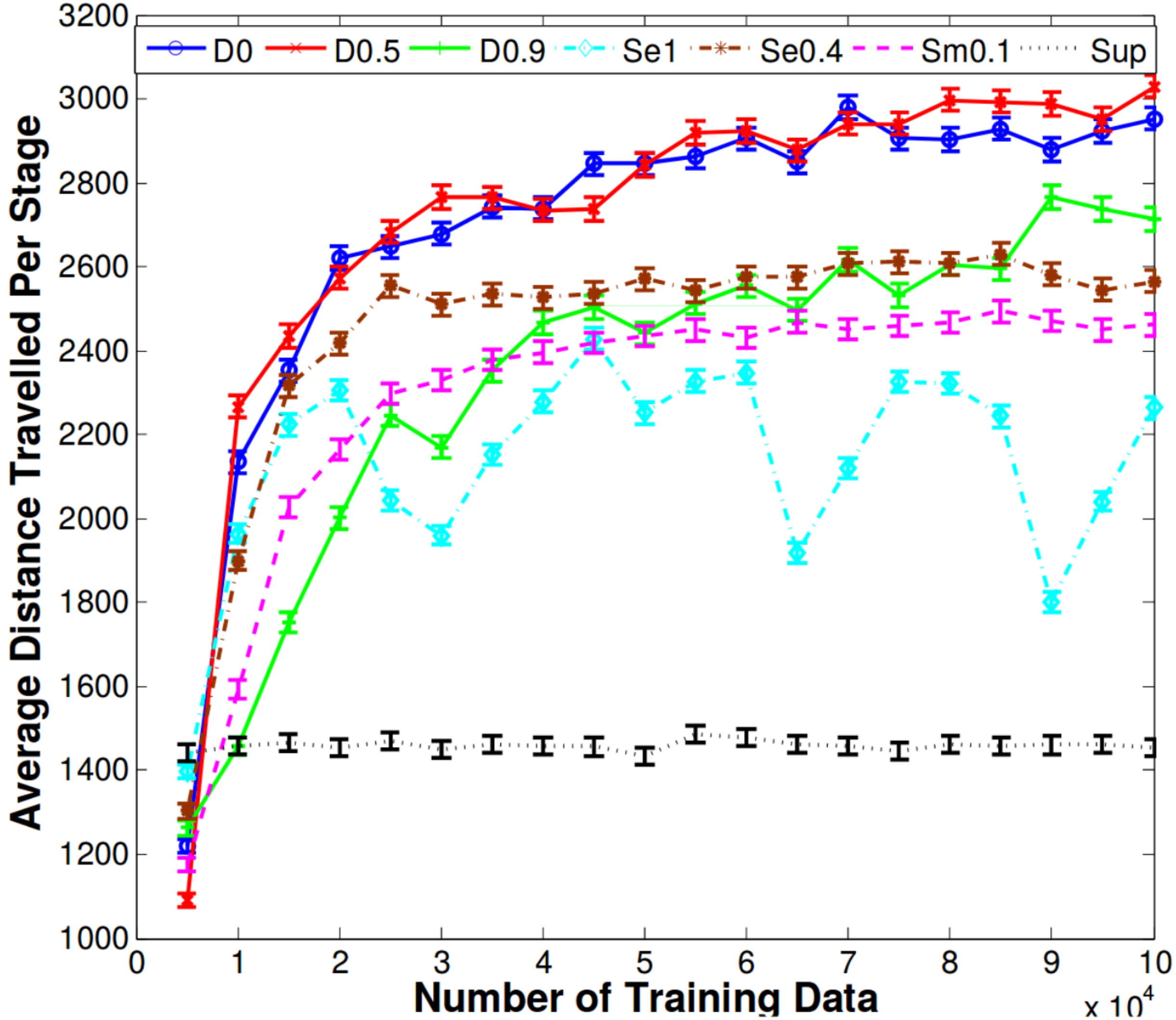
If $N = T \log T$,

$$\mathbf{L}(\pi_n) < T\, \epsilon_N + \mathbf{O}(1)$$

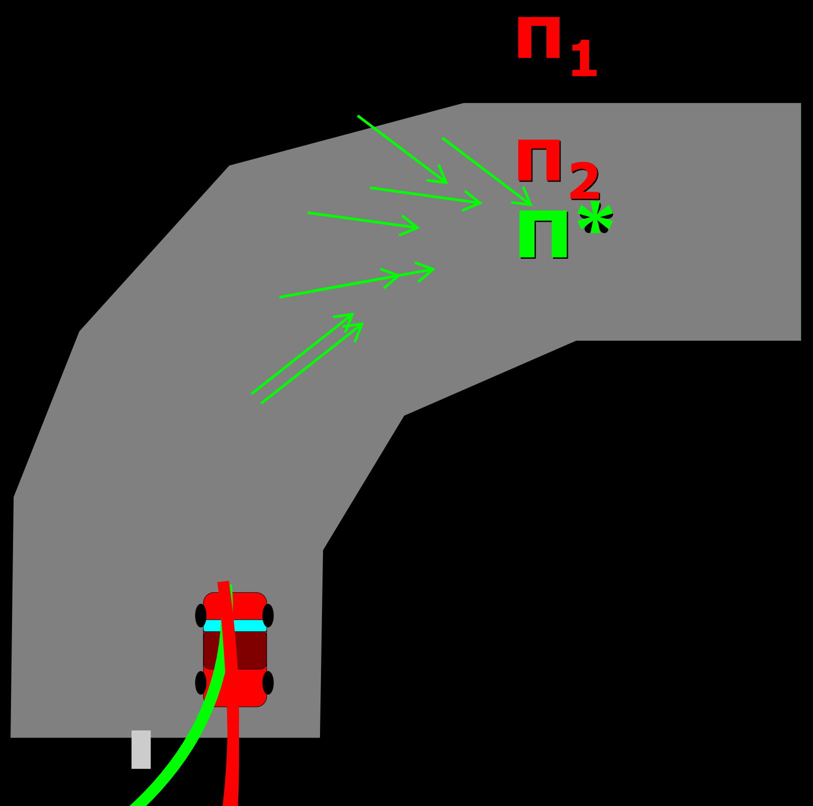
for some n



Video credit: Stéphane Ross, Geoff Gordon and Drew Bagnell



Ross+Bagnell, AIStats’10



π_1
π_2
π^*

Classifier: $h : x \to [K]$

- $(x,y) \in X \times [K]$
- $\min_h \Pr(\, h(x) \neq y \,)$

- $(x,c) \in X \times [0,\infty)^K$
- $\min_h E_{(x,c)} [\, c_{h(x)} \,]$

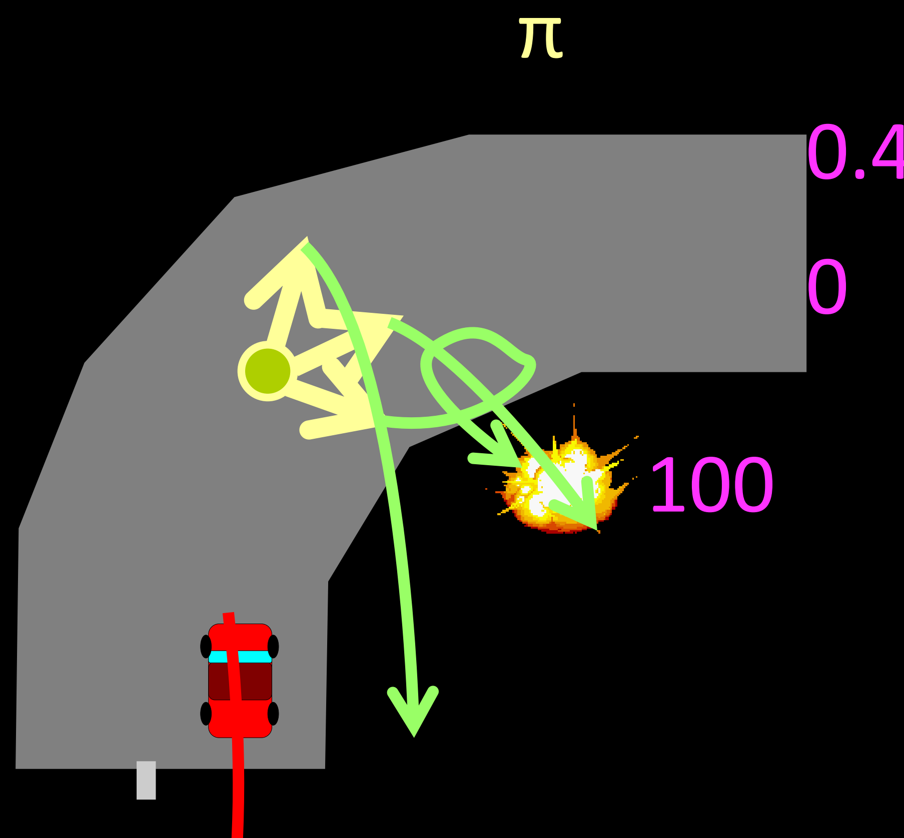
Classifier: $h : x \to [K]$

- $(x,c) \in X \times [0,\infty)^K$
- $\min_h E_{(x,c)} [\ c_{h(x)}\ ]$

Solution learn a K-dimensional regressor on costs; pick minimal cost

1. Let learned policy π drive for t timesteps to obs. o
2. For each possible action a:
   - Take action a, and let expert π^{ref} drive the rest
   - Record the overall loss, c_a
3. Update π based on example:

   $(o, \langle c_1, c_2, \ldots, c_K \rangle)$
4. Goto (1)



- From demonstrations → expert decisions

- From expert decisions → expert costs

observation
optimal(ish) action

$a_1, a_2, \ldots, a_{t-1}$

(expected) minimum achievable loss

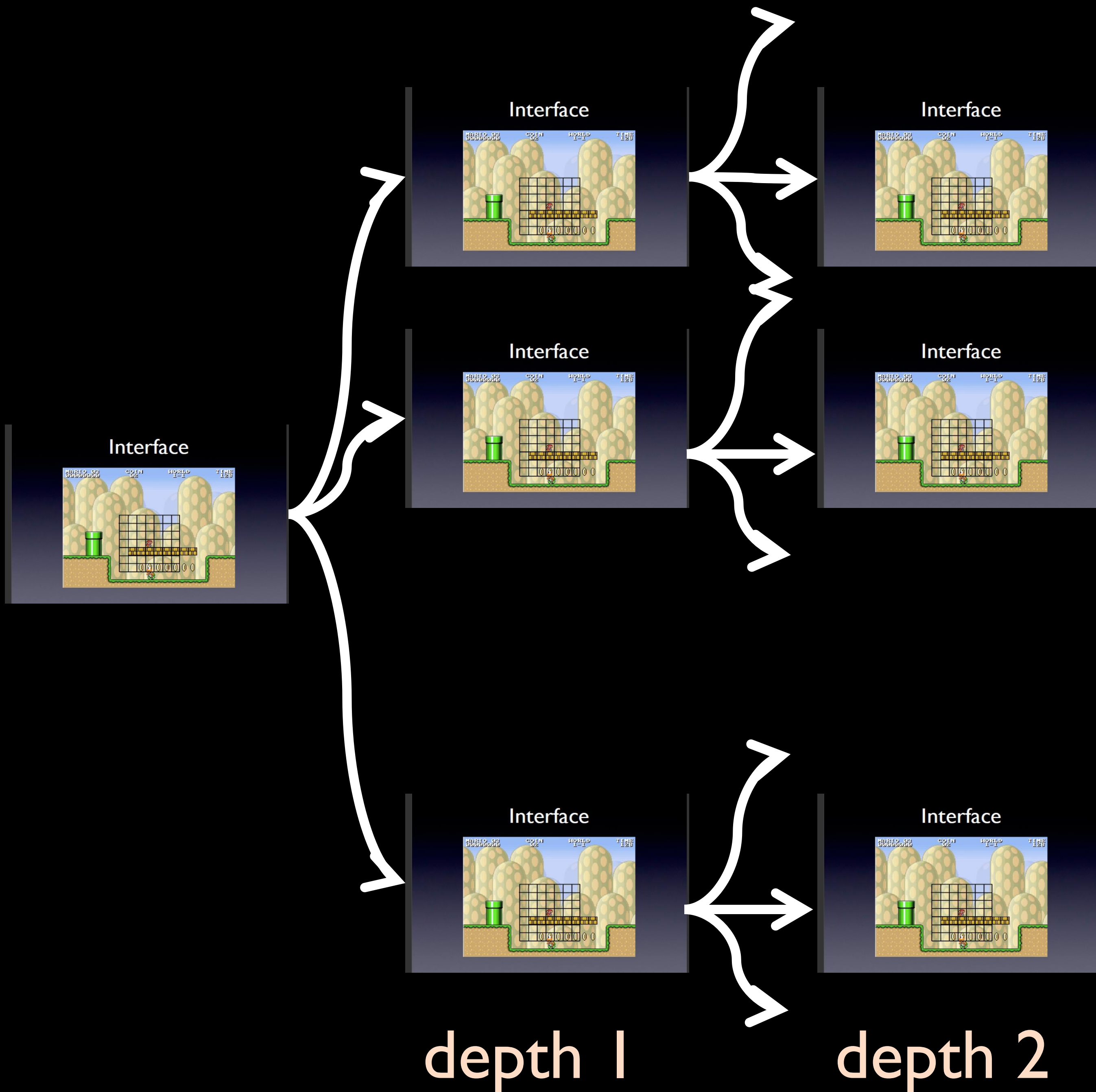
$$\min_{(a_t, a_{t+1}, \ldots)} \mathbf{E}\, \mathrm{loss}(\vec{a})$$

optimal action

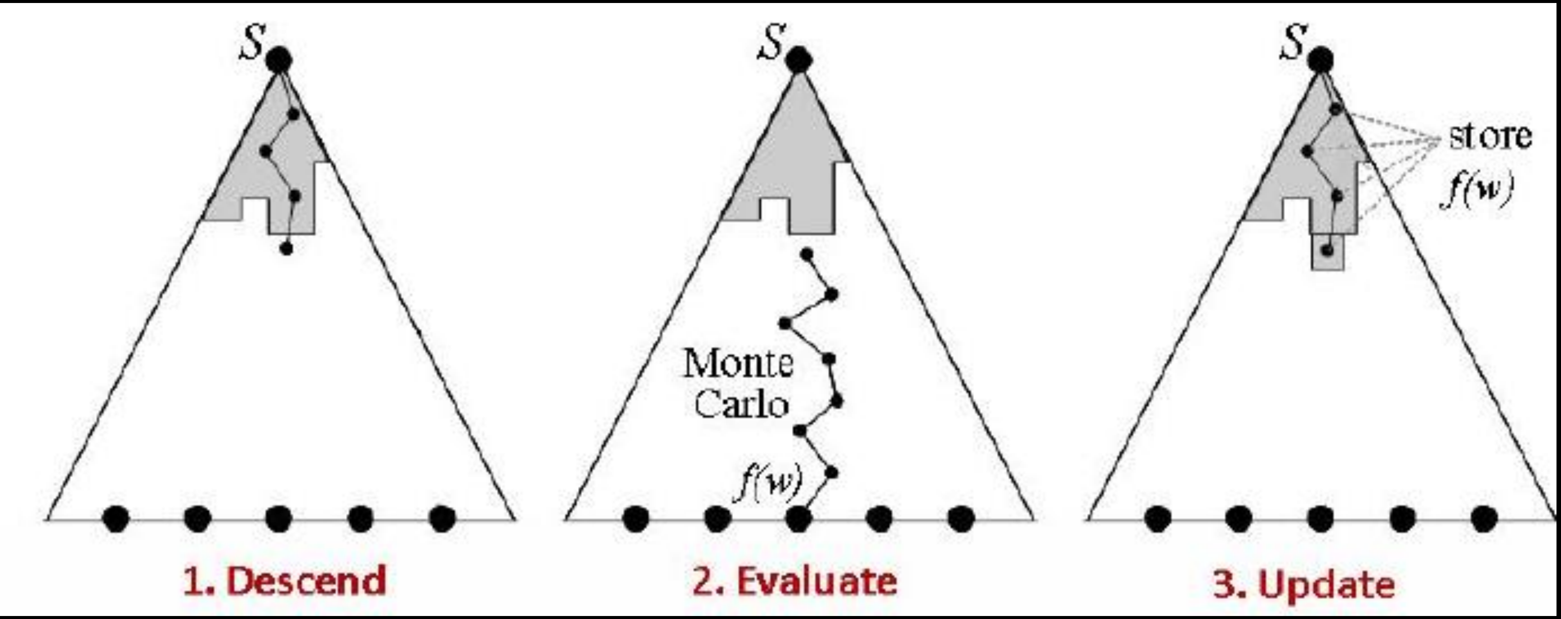
a_t

optimal Q values

a_t



Rollout to some depth
→ Intermediate loss
→ Return best start



e.g., Monte Carlo
Tree Search

Image credit: Michele Sebag and DeepMind

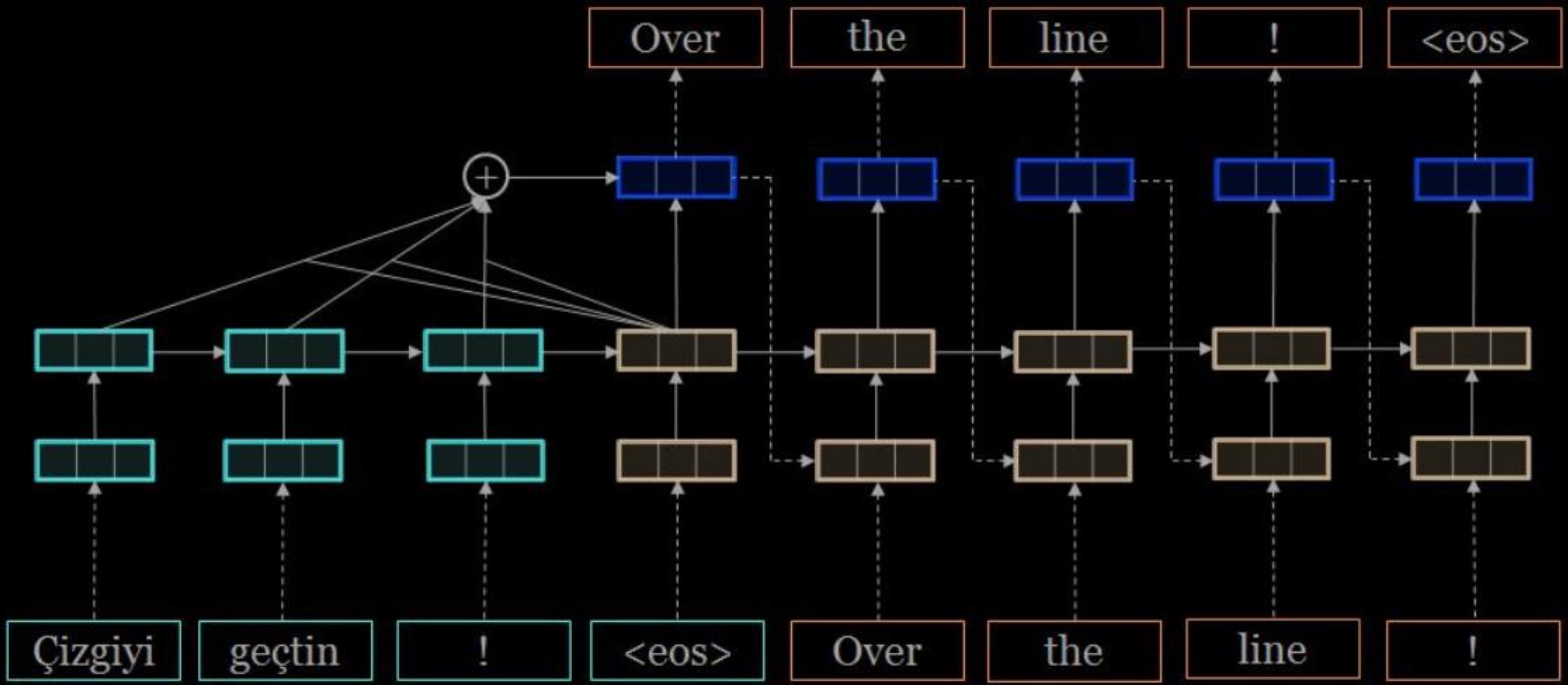
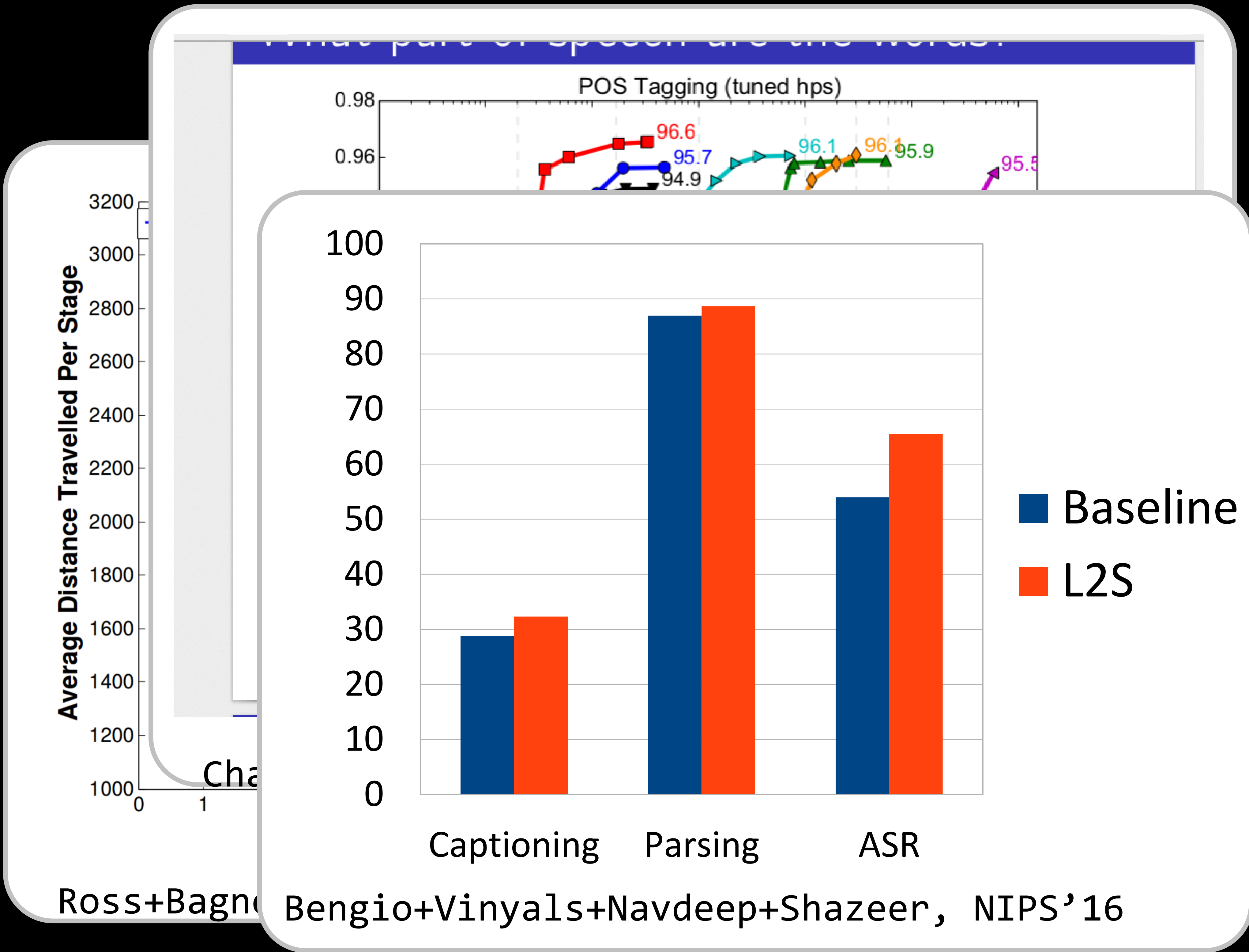


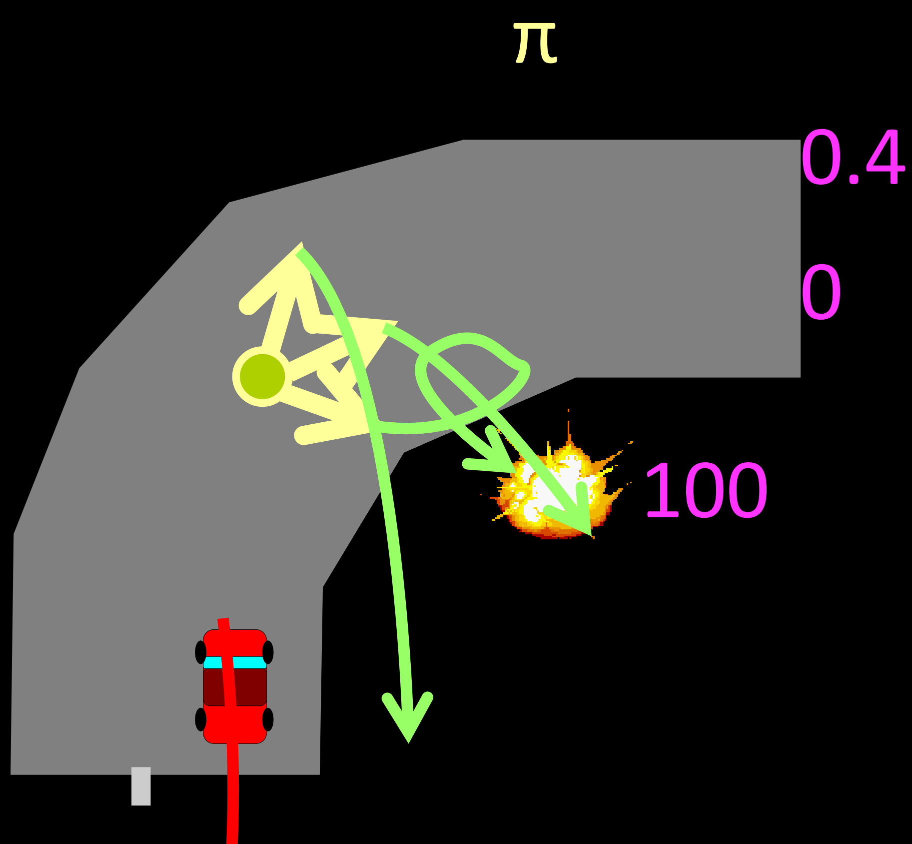
Image credit: Klein et al., 2017


POS Tagging (tuned hps)
0.98
0.96
96.6
95.7
94.9
96.1
96.1
95.9
95.5
Average Distance Travelled Per Stage
3200
3000
2800
2600
2400
2200
2000
1800
1600
1400
1200
1000
0
1
100
90
80
70
60
50
40
30
20
10
0
Captioning
Parsing
ASR
Baseline
L2S
Ross+Bagne
Bengio+Vinyals+Navdeep+Shazeer, NIPS'16

- From demonstrations → expert decisions
- From expert decisions → expert costs
- Whence the expert?

1. Let learned policy π^{in} drive for t timesteps to obs. o
2. For each possible action a:
   - Take action a, and let something π^{out} drive the rest
   - Record the overall loss, c_a
3. Update π based on example:

   $(o, \langle c_1, c_2, \ldots, c_K \rangle)$
4. Goto (1)

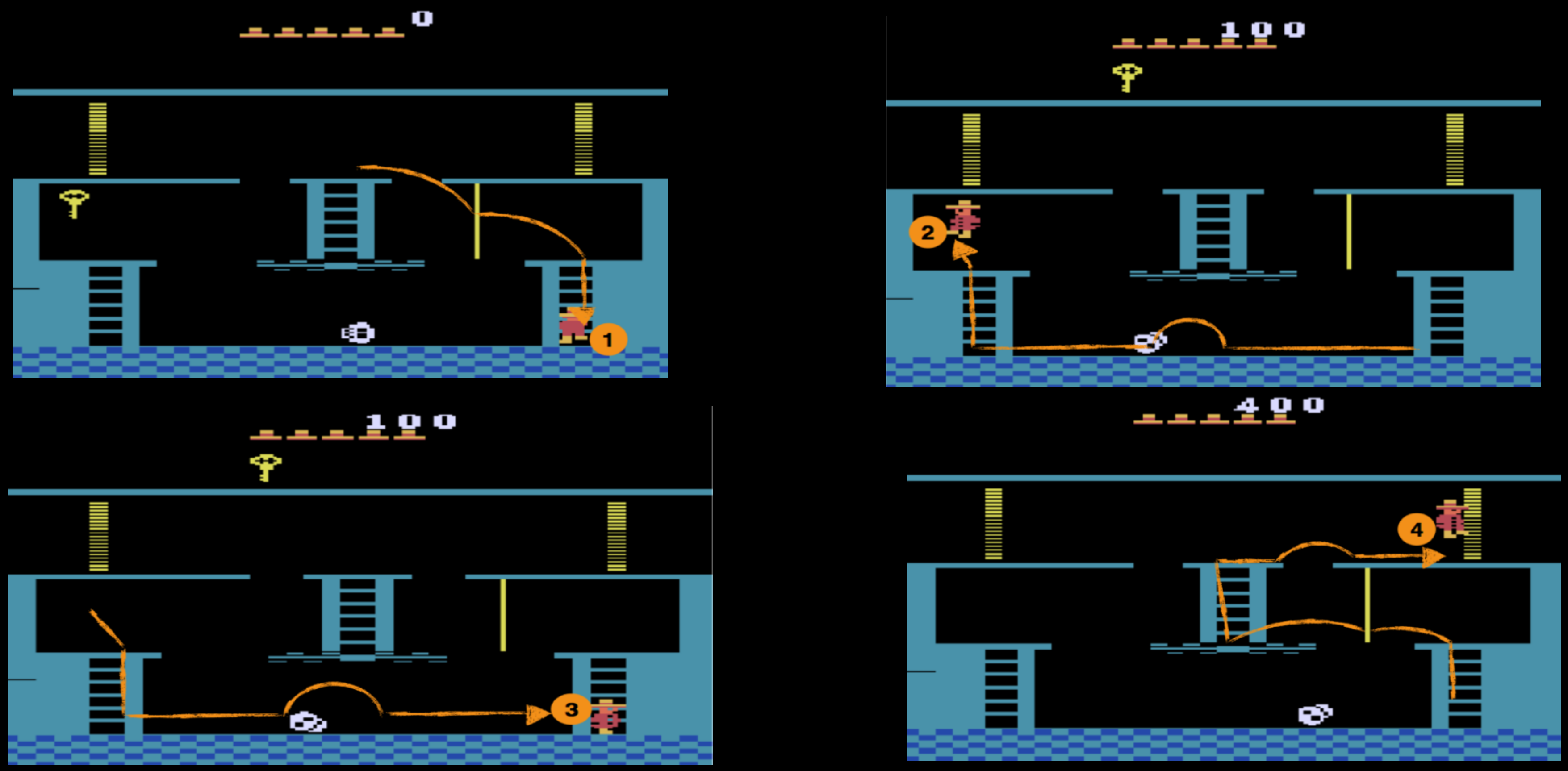


roll-out → ↓ roll-in	**Reference**	**Mixture**	**Learned**
Reference	Inconsistent		
Learned	Not locally opt.	Good	RL

roll-out → / ↓ roll-in	Reference	Mixture	Learned
Reference	Inconsistent		
Learned	Not locally opt.	Good	RL

$$Regret = O\left((KT)^{2/3} \sqrt[3]{\frac{\log(N|\Pi|)}{N}} + T\delta_{class} \right)$$

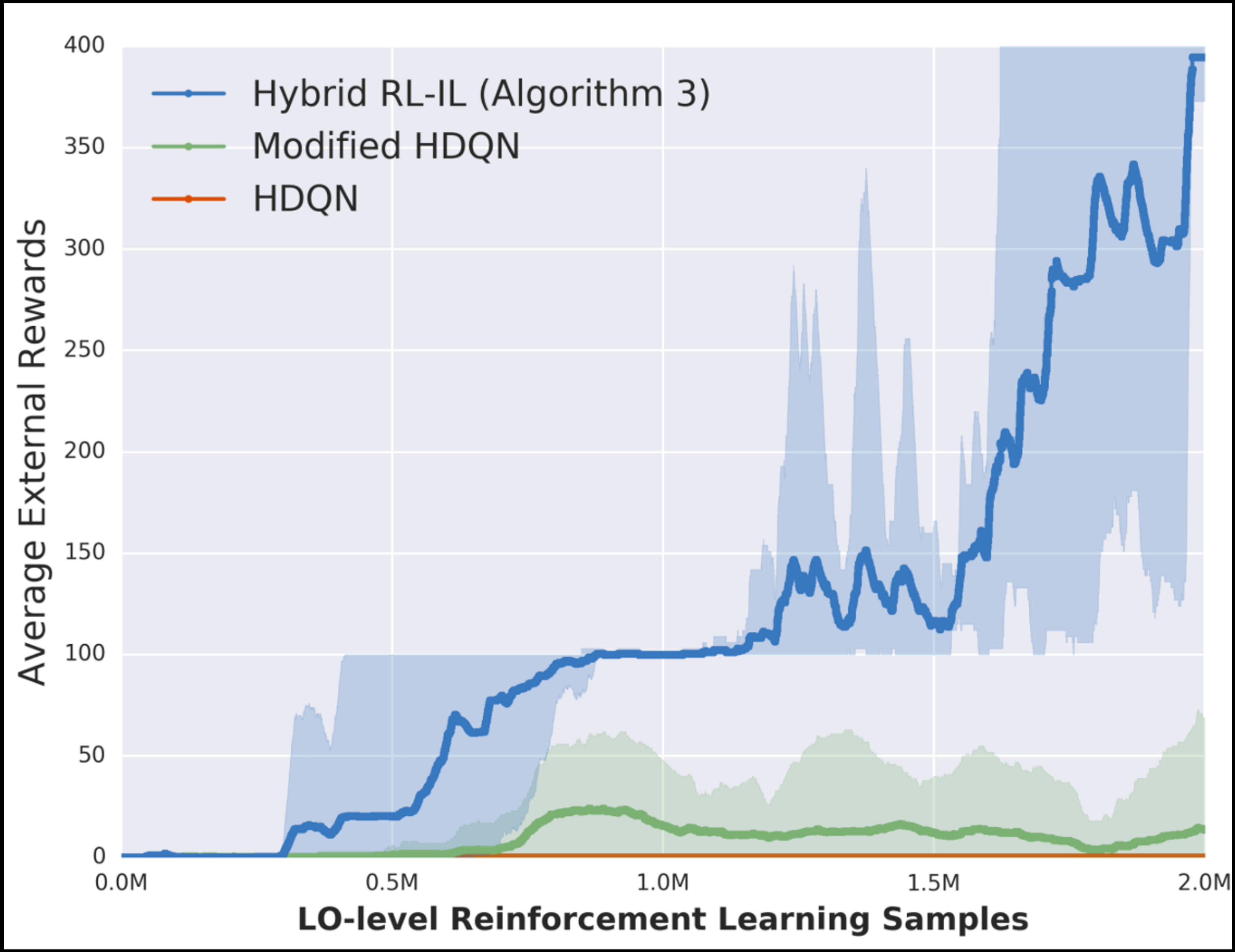

0



Key insight: **verifying if low-level trajectory is successful is cheaper than labeling low-level trajectory**

→ labeling effort = high-level horizon + low-level horizon
only a fraction of the full horizon (as low as sqrt of the full horizon)

→ subpolicies are only learnt in the relevant part of the state space



Hybrid RL-IL (Algorithm 3)
Modified HDQN
HDQN
Average External Rewards
400
350
300
250
200
150
100
50
0
0.0M
0.5M
1.0M
1.5M
2.0M
LO-level Reinforcement Learning Samples

imitation learning summary

successes:

open problems:

Thank you! Queries!