DAgger: Taming Covariate Shift with No Regret

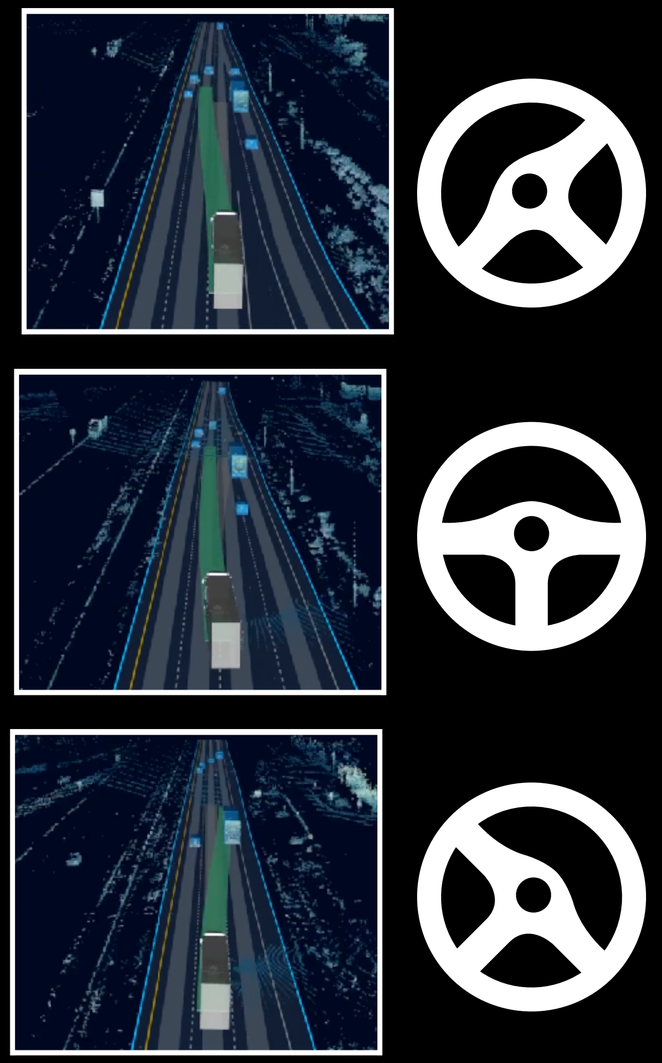
Sanjiban Choudhury

Cornell Bowers CIS
Computer Science

Recap in
60 seconds!

SUPERVISED LEARNING

#1 Get Data



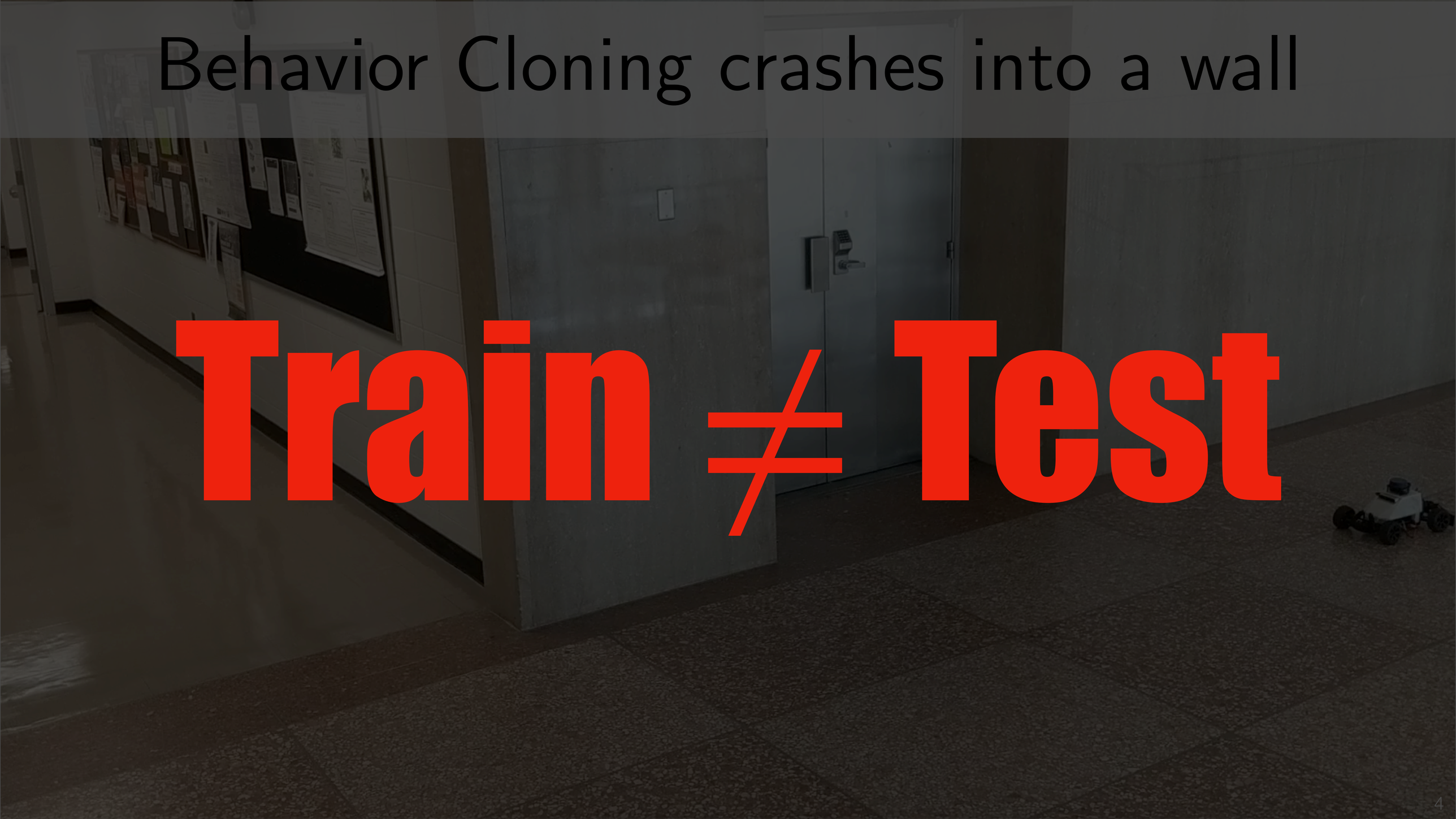
Input (s) Output (a)

#2
Train
Policy

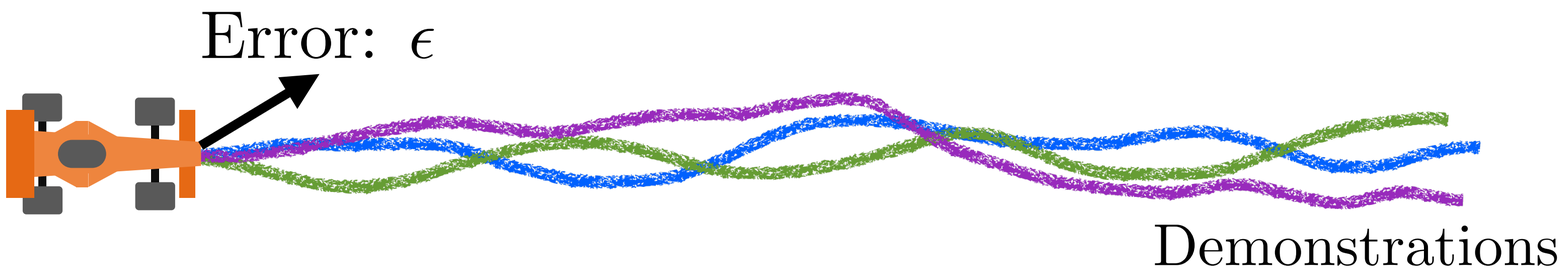
$$\pi : s \rightarrow a$$

#3 Deploy!

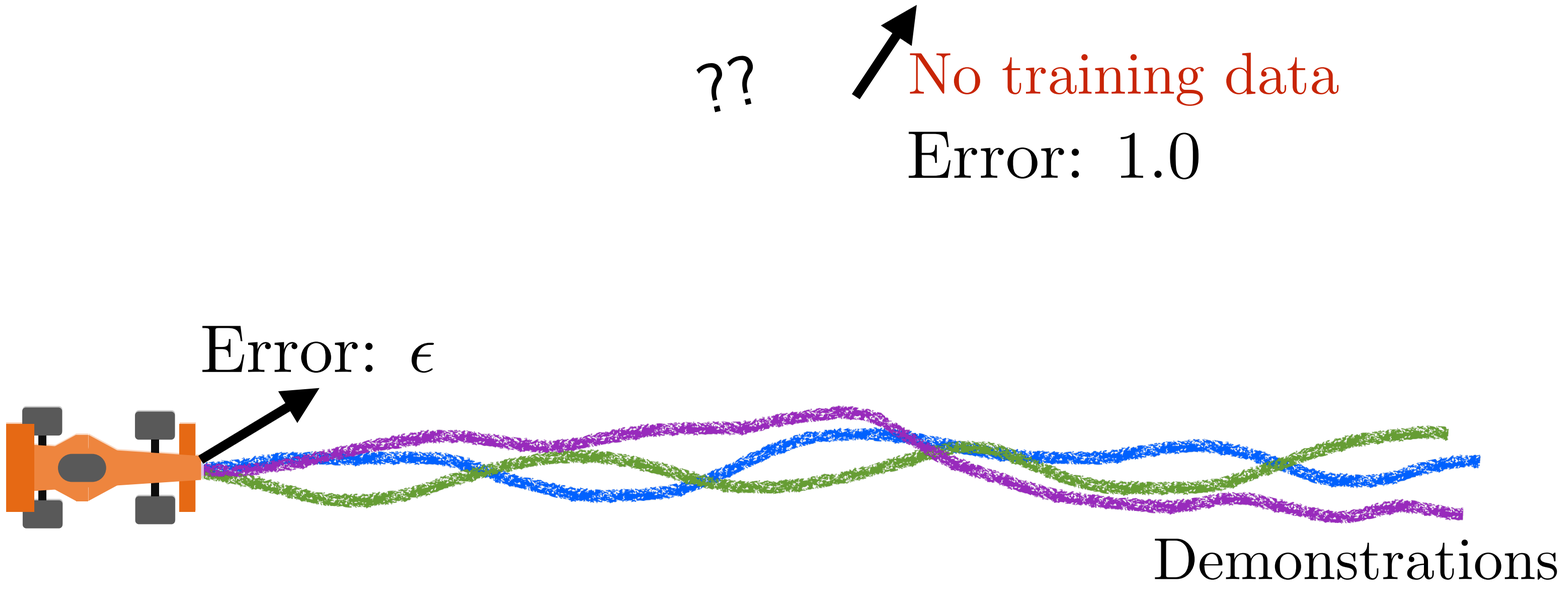
Behavior Cloning crashes into a wall



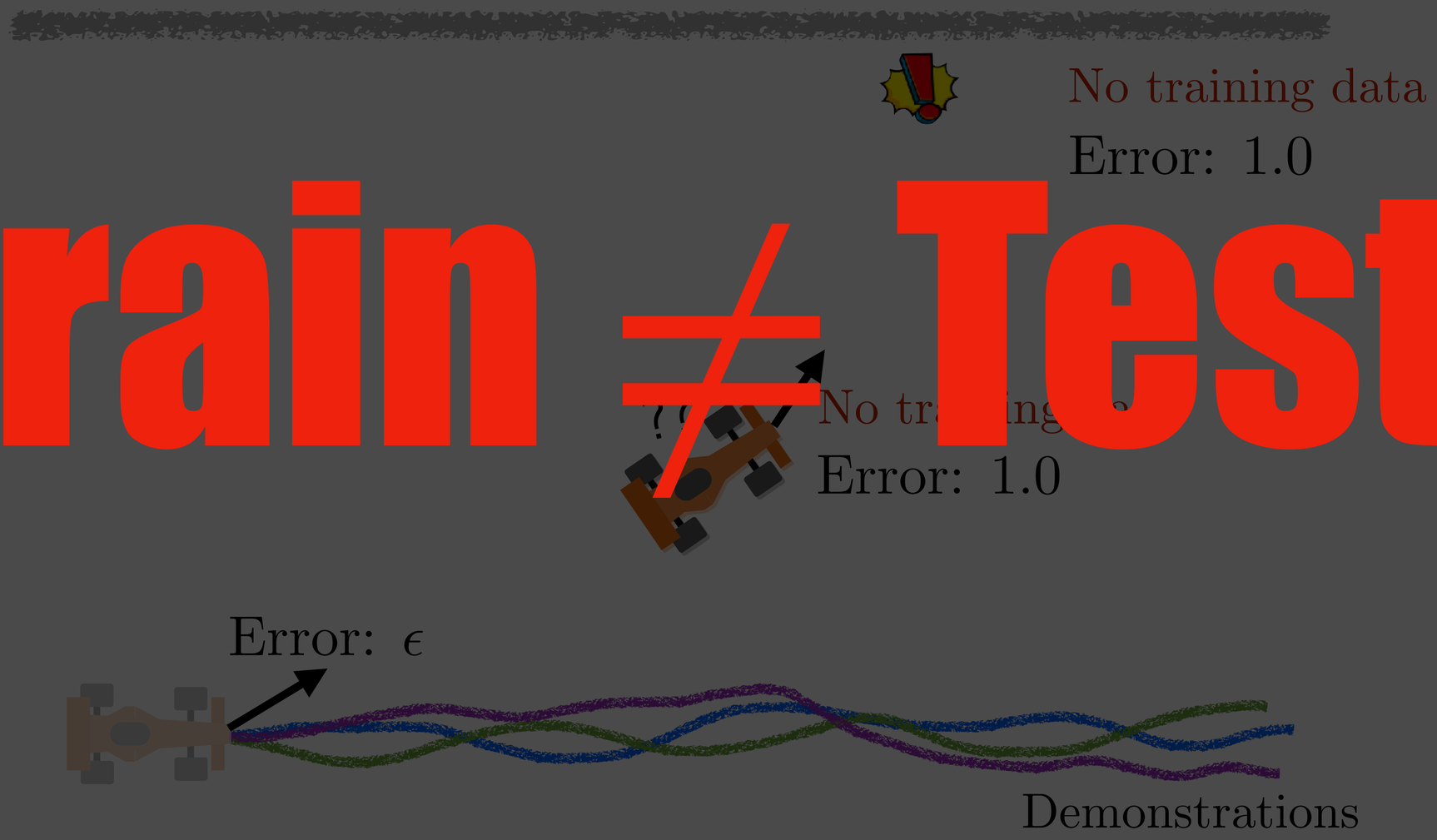
Why did the robot crash?



Why did the robot crash?



Why did the robot crash?



Activity!

Think-Pair-Share!

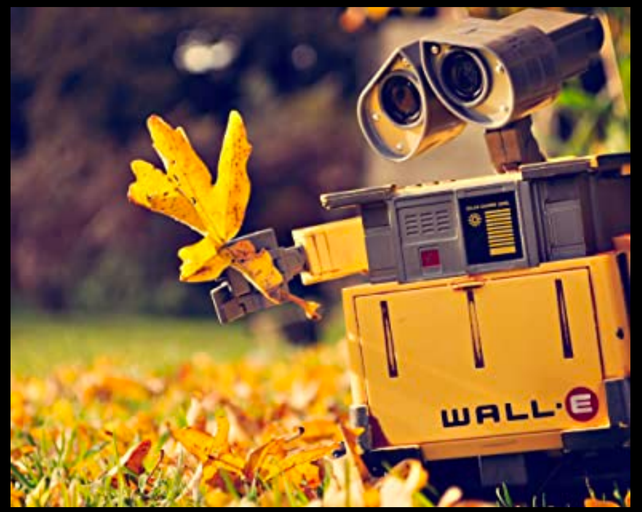
Think (30 sec): What are ALL the different sources of train-test mismatch?

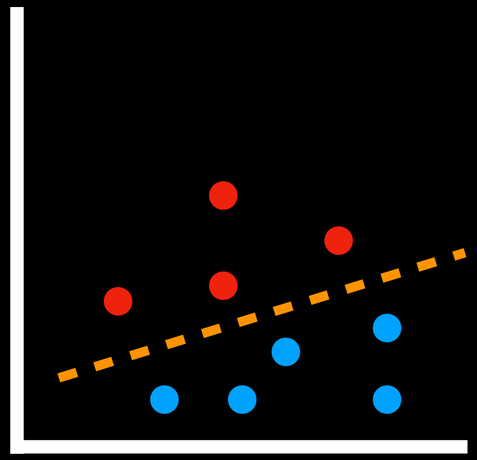
Pair: Find a partner

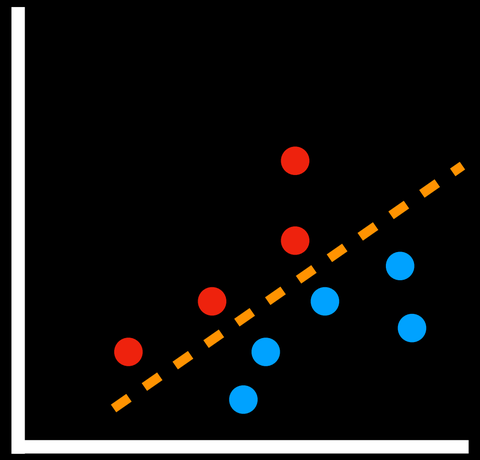
Share (45 sec): Partners exchange ideas

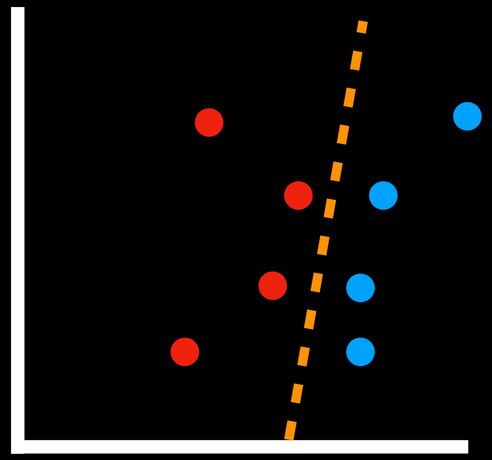
Train $\neq$ Test

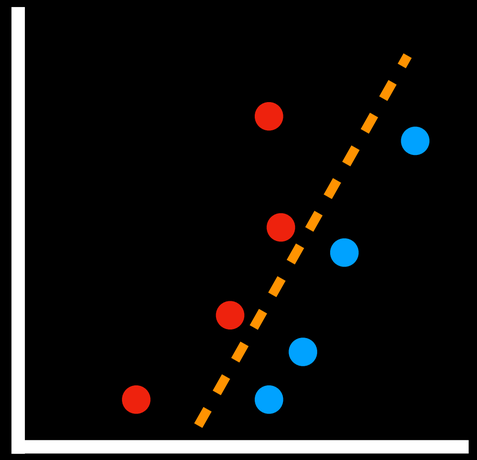
Case 1: Data changes over time



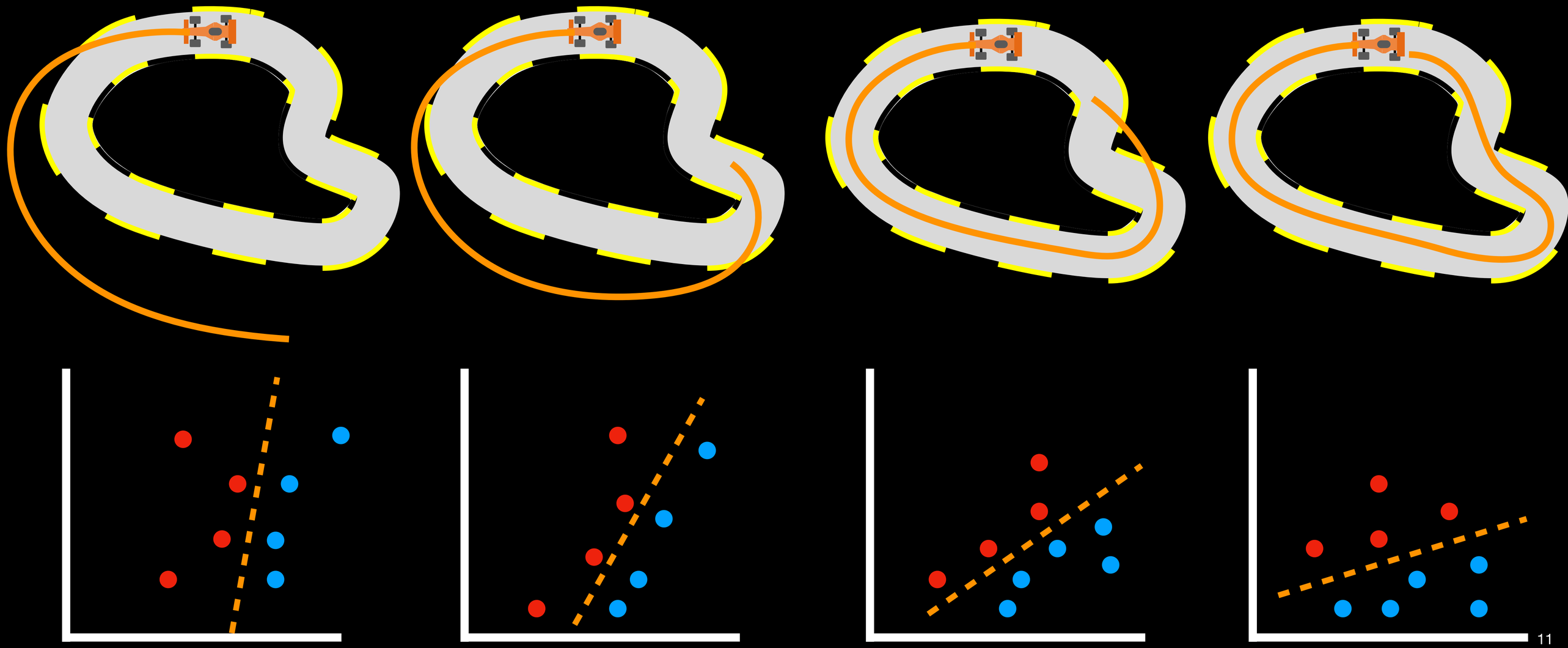




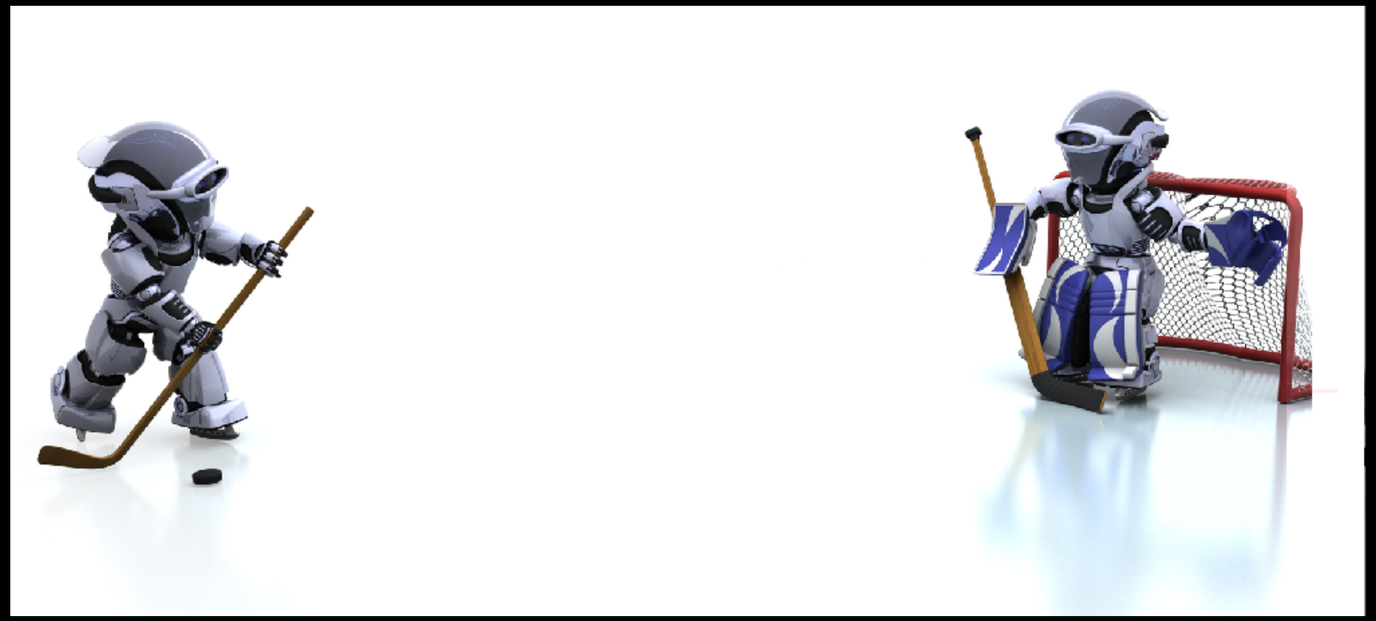


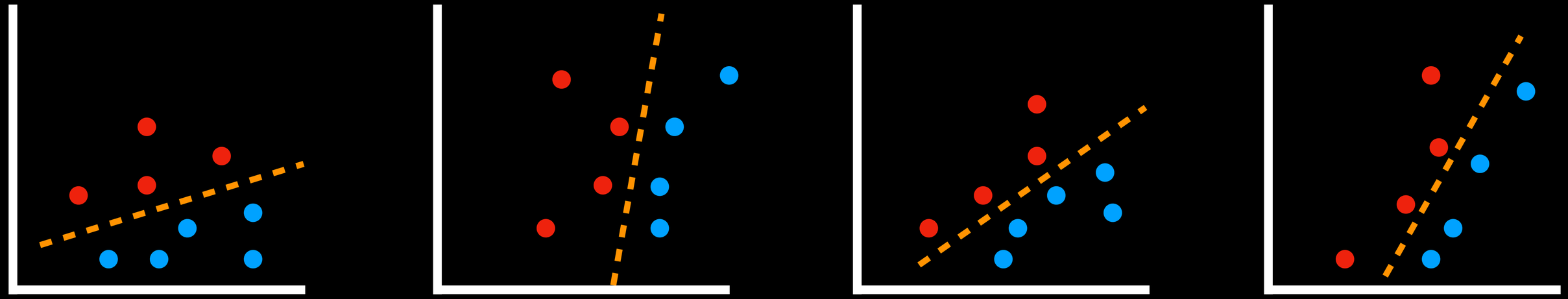


Case 2: Data changes with robot behavior



Case 3: Data changes adversarially (game)



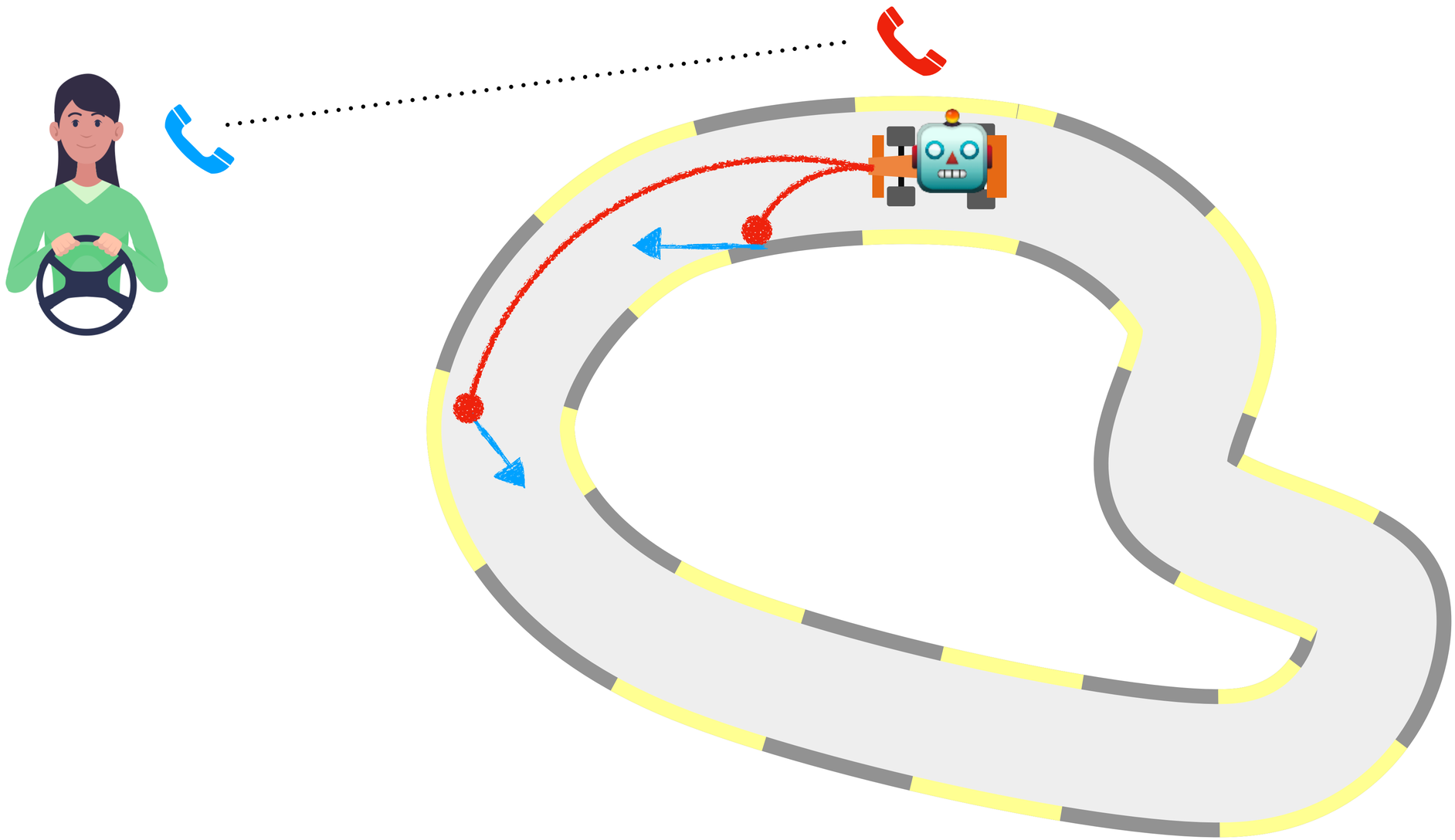


Challenge:

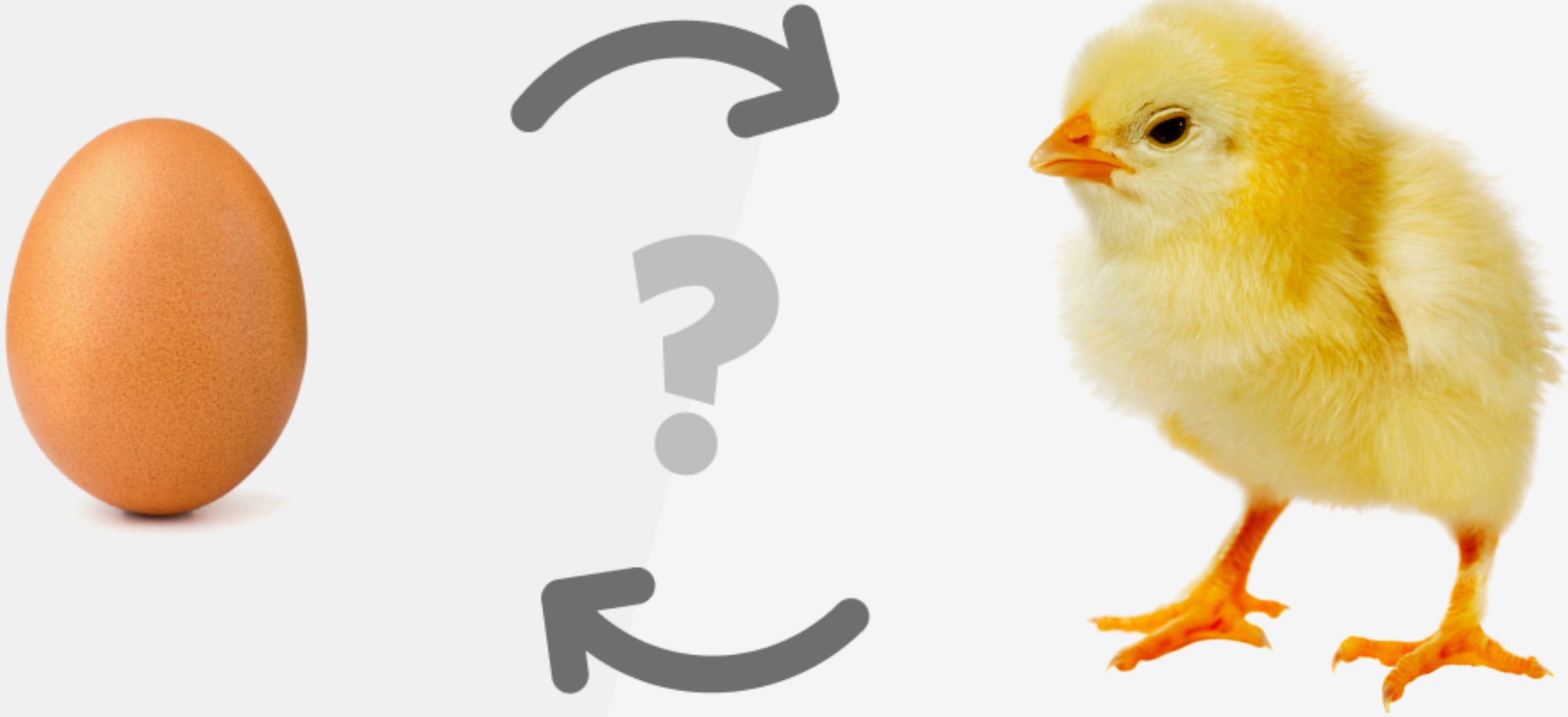
Don't know the test distribution upfront

Learner ← ✗ — Collect Data

Introducing an *interactive* expert!



To know the distribution, you need a learner
To train a learner, you need a distribution



Let's work out a *simple* algorithm

Be stable

Slowly change predictions

DAGGER

Episode IV

A NEW HOPE

DAGGER: A meta-algorithm for imitation learning

A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning

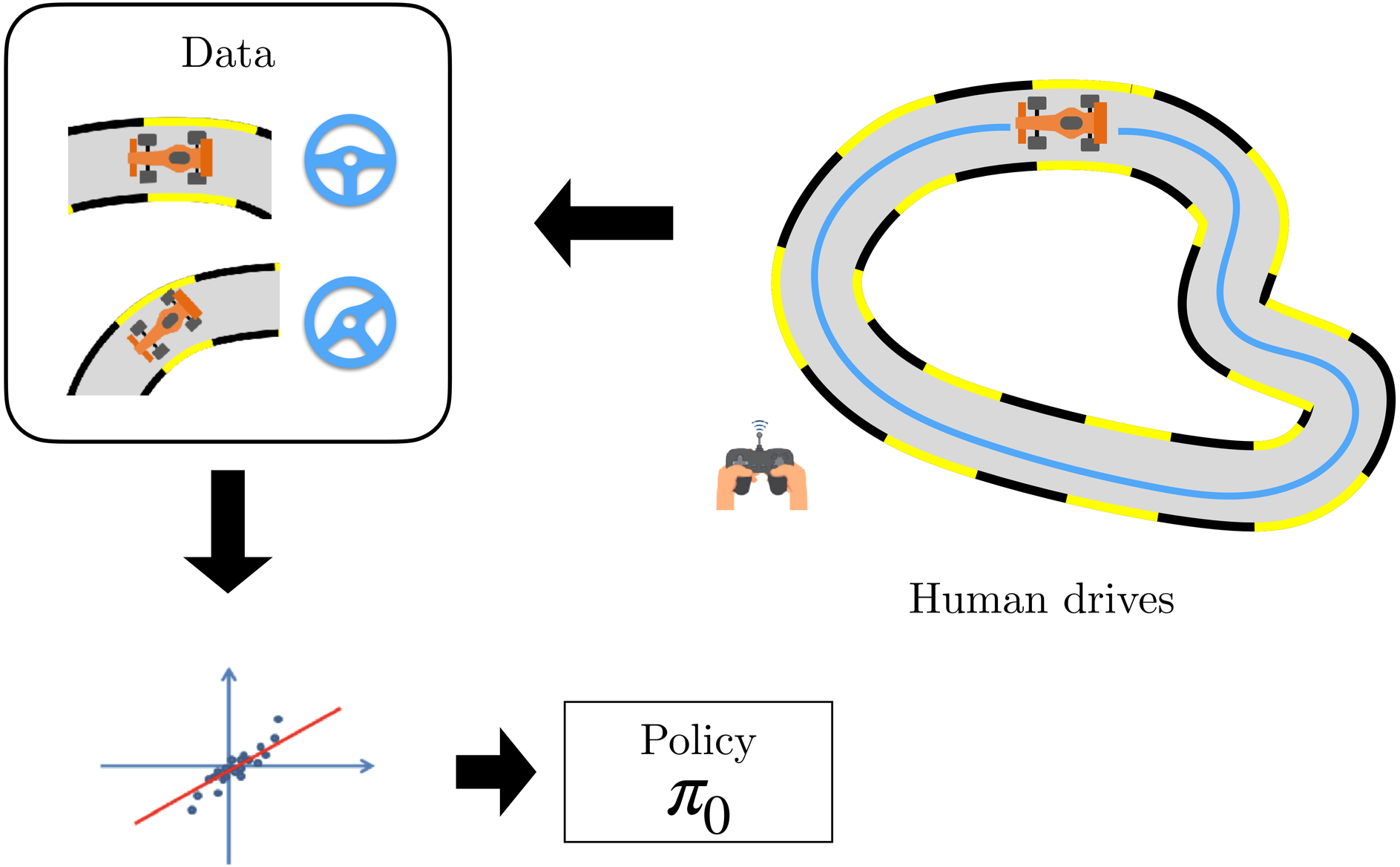
Stéphane Ross
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
stephaneross@cmu.edu

Geoffrey J. Gordon
Machine Learning Department
Carnegie Mellon University
Pittsburgh, PA 15213, USA
ggordon@cs.cmu.edu

J. Andrew Bagnell
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
dbagnell@ri.cmu.edu

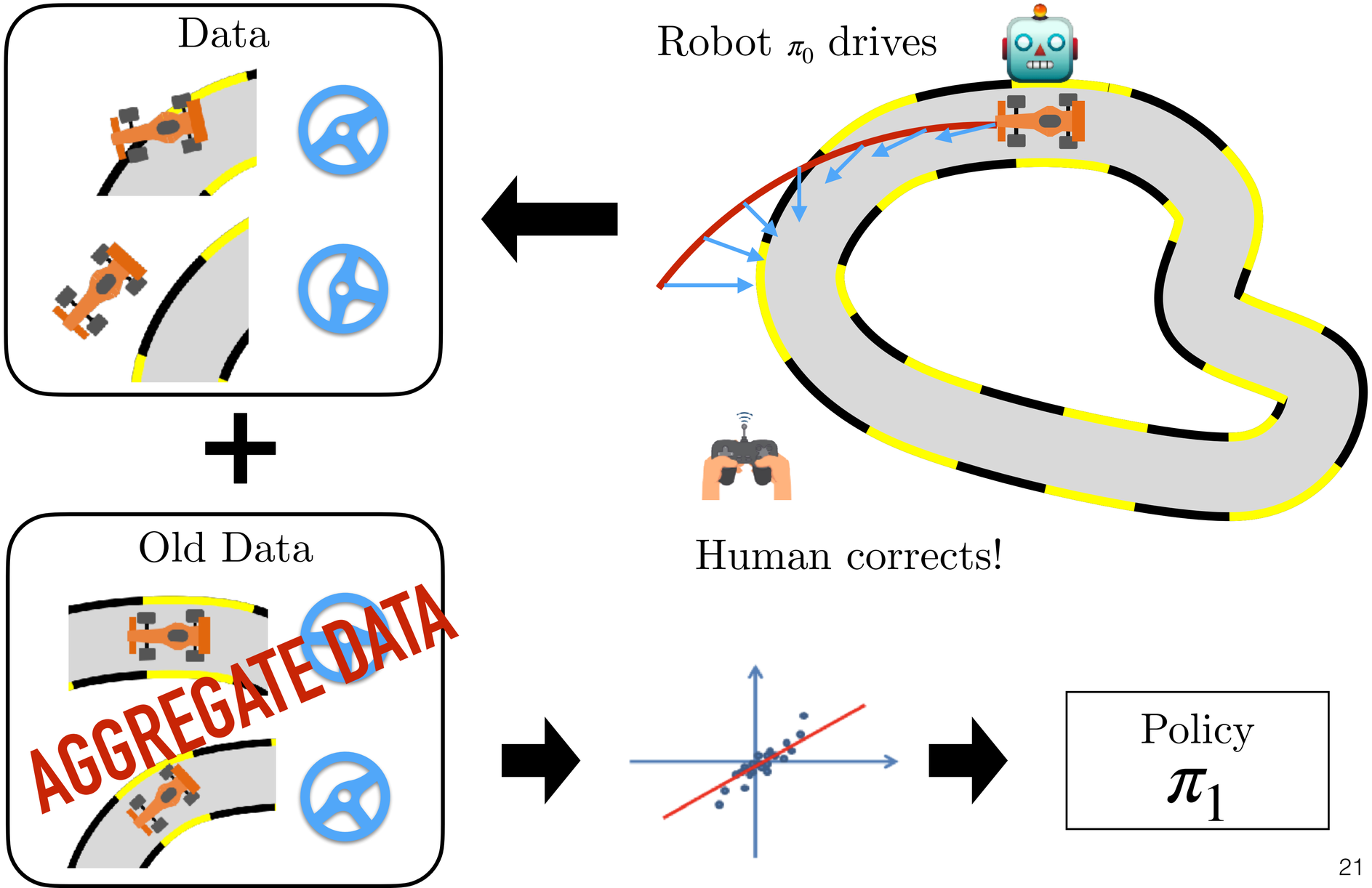
[Ross et al'11]

DAgger: Iteration 0



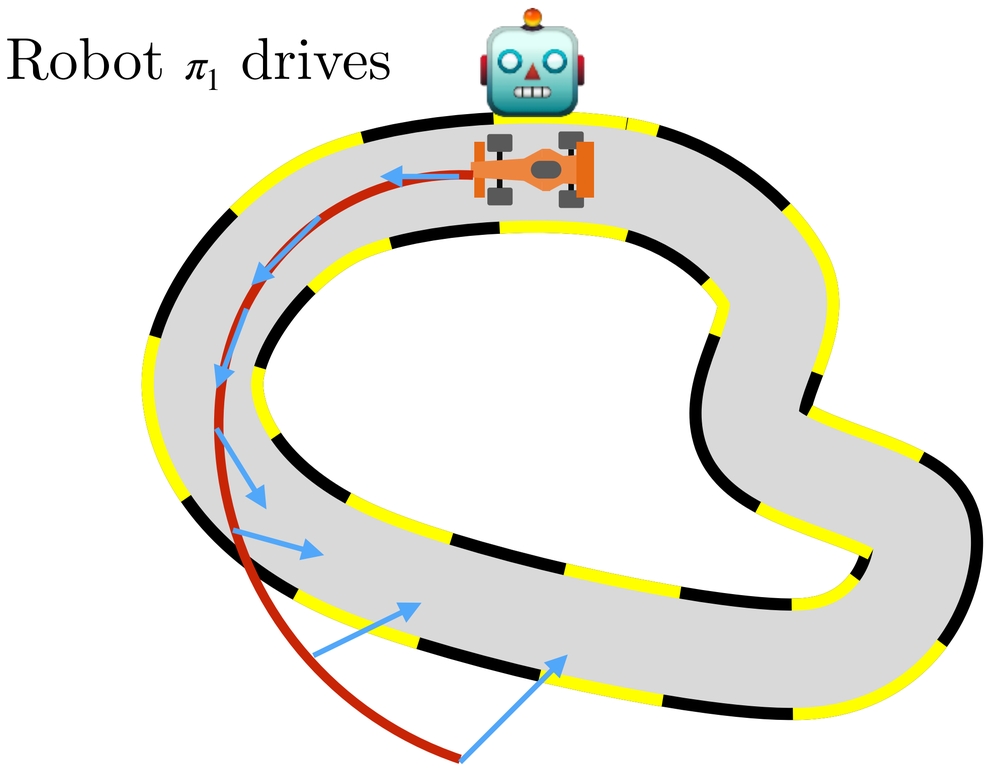
[Ross et al'11]

DAgger: Iteration 1



[Ross et al'11]

DAgger: Iteration 2

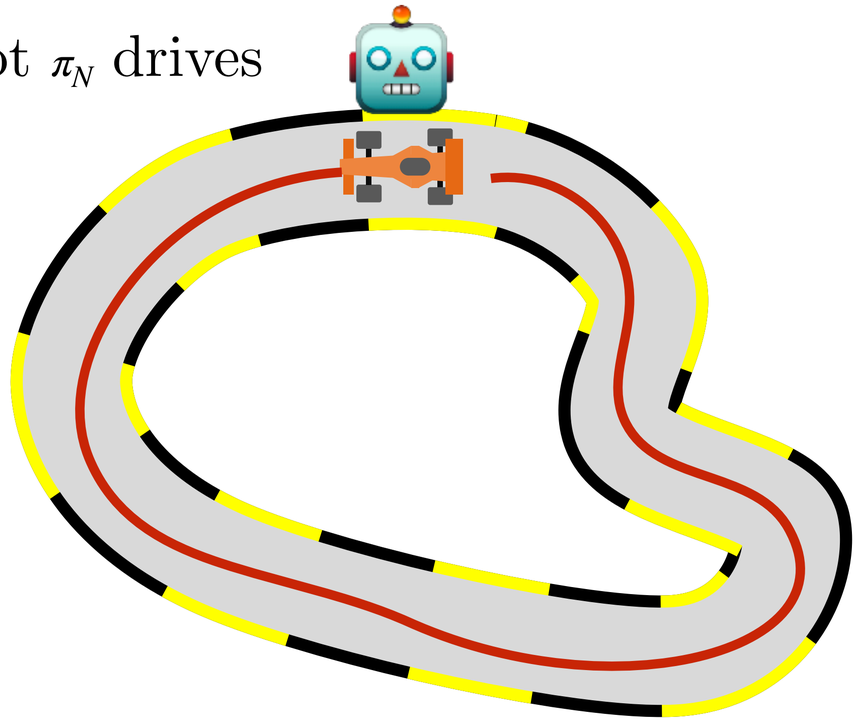


AGGREGATE DATA

[Ross et al'11]

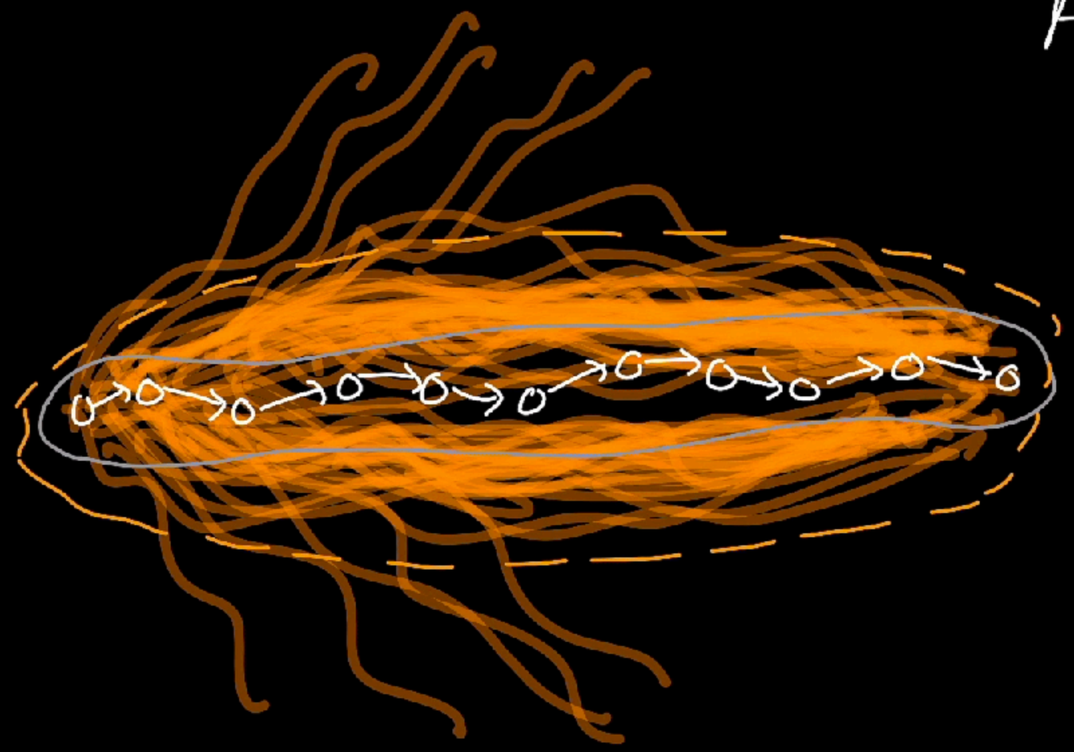
DAgger: Iteration 1

Robot π_N drives



After many iterations ....
we are able to drive like a human!

DAGGER (DATASET AGGREGATION)



AGGREGATED TRAINING DISTRIBUTION

$\approx$

TEST DISTRIBUTION

HUMAN DISTRIBUTION

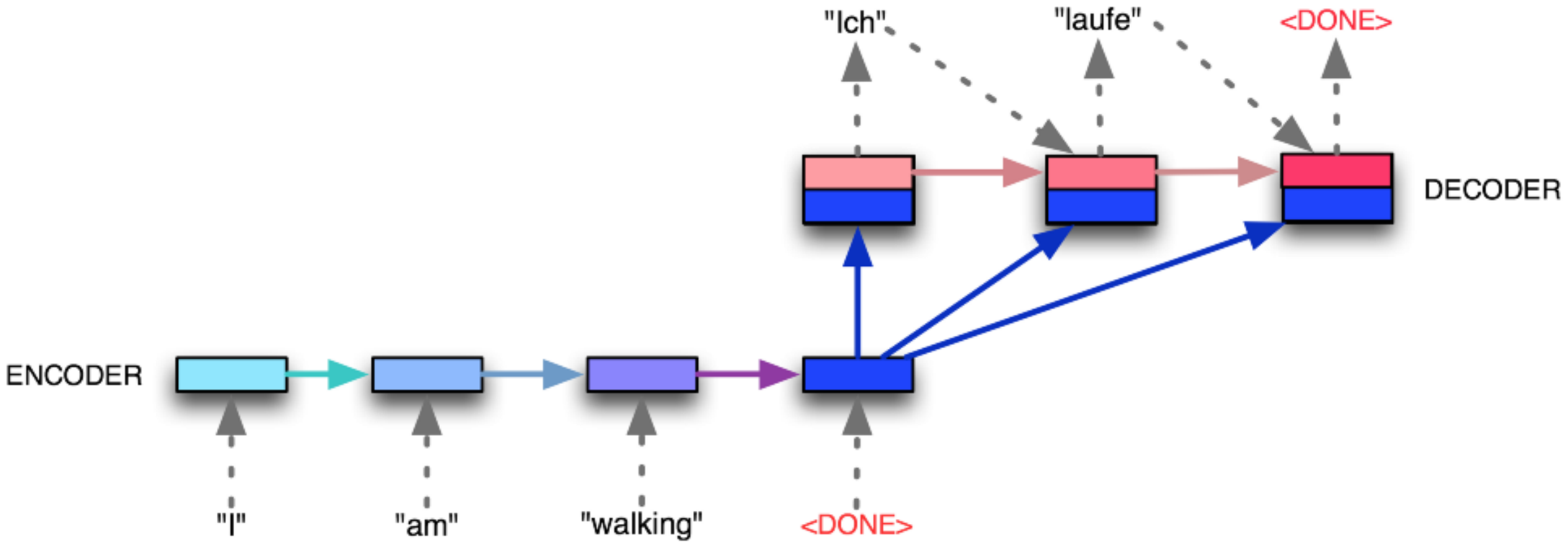
Original results from DAGGER!

Original results from DAGGER!

DAgger here reacts dynamically to an untrained obstacle

A brief history of DAGGER

NLP folks (Hal Daume III in particular) were first to identify feedback effects in sequential prediction tasks



DAGGER is a foundation

Imitation under uncertainty

SAIL

ExPLORE STROLL

Counterfactual Teaching

DPI LOLS

NRPI

Reinforcement Learning

Agnostic SysID DaaD

Model learning

DAEQUIL

AGGREVATE(D)

Imitation learning

EIL

HG-DAGGER

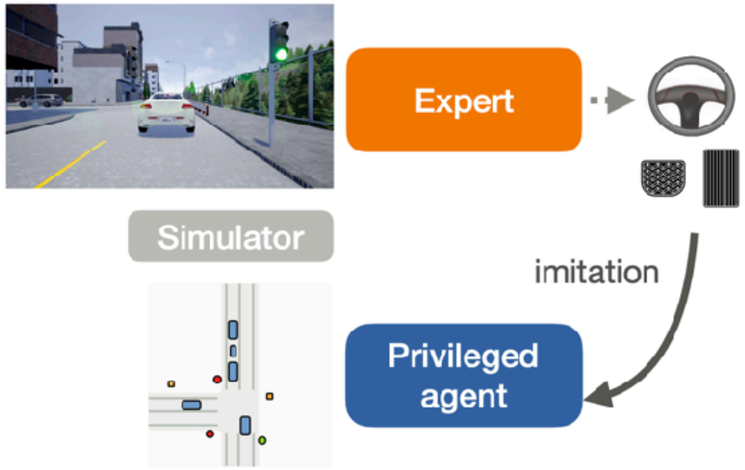
SHIV

Query efficient imitation learning

DAGGER

Many cool applications of DAGGER in robotics

Lee et al, Learning quadrupedal locomotion over challenging terrain (2020)



Chen et al Learning by Cheating(2020)

Choudhury et al, Data Driven Planning via Imitation Learning (2018)

Pan et al Imitation learning for agile autonomous driving (2019)

But why does
aggregating data work?

Think of the human ...

... as an adversary

Interactive Learning

The Imitation Game

Learner

Adversary

Initialize policy

π_1 [policy]

Chooses loss

$l_1(\,.\,)$ [loss]

Update policy

π_2

Chooses loss

$l_2(\,.\,)$

⋮

⋮

Let’s formalize!

How do we design algorithms that are no-regret?

$$\text{Regret} = \sum_{t=1}^{T} l_t(\pi_t) - \min_{\pi^*} \sum_{t=1}^{t} l_t(\pi^*)$$

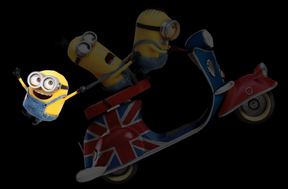
(Learner)

(Best in hindsight)

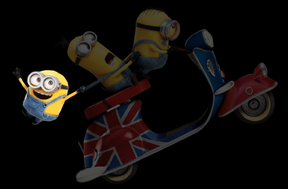
At every round t, choose the best expert in hindsight

$$\pi_t = \arg\min_{\pi} \sum_{i=1}^{t-1} l_i(\pi)$$

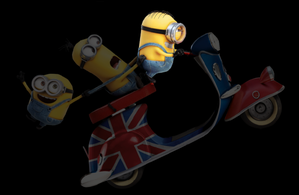
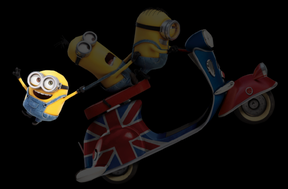
(lowest total loss)

$\sum l_t$		l_1
- -	Expert 1 	1.0
- -	Expert 2 	0.2
- -	Expert 3 	0.5

Avg. Regret: - -

$\sum l_t$		l_1	l_2
1.0	Expert 1 	1.0	0.5
0.2	Expert 2 	0.2	0.5
0.5	Expert 3 	0.5	0.2

Avg. Regret: 0.80

$\sum l_t$		l_1	l_2	l_3
1.5	Expert 1 	$\boxed{1.0}$	0.5	0.5
0.7	Expert 2 	0.2	$\boxed{0.5}$	$\boxed{1.0}$
0.7	Expert 3 	0.5	0.2	0.2

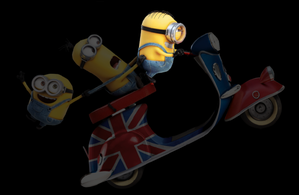
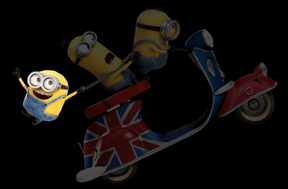
Avg. Regret: 0.40

$\sum l_t$		l_1	l_2	l_3	l_4
2.0	Expert 1 	$\boxed{1.0}$	0.5	0.5	1.0
1.7	Expert 2 	0.2	$\boxed{0.5}$	$\boxed{1.0}$	0.2
0.9	Expert 3 	0.5	0.2	0.2	$\boxed{0.5}$

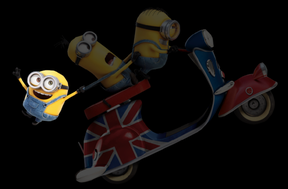
Avg. Regret: 0.53

$\sum l_t$		l_1	l_2	l_3	l_4	l_5
3.0	Expert 1 	$\boxed{1.0}$	0.5	0.5	1.0	0.5
1.9	Expert 2 	0.2	$\boxed{0.5}$	$\boxed{1.0}$	0.2	1.0
1.4	Expert 3 	0.5	0.2	0.2	$\boxed{0.5}$	$\boxed{0.2}$

Avg. Regret: 0.40

$\sum l_t$		l_1	l_2	l_3	l_4	l_5	l_6
3.5	Expert 1 	$\boxed{1.0}$	0.5	0.5	1.0	0.5	1.0
2.9	Expert 2 	0.2	$\boxed{0.5}$	$\boxed{1.0}$	0.2	1.0	0.5
$\boxed{1.6}$	$\boxed{\text{Expert 3}}$ 	0.5	0.2	0.2	$\boxed{0.5}$	$\boxed{0.2}$	$\boxed{0.2}$

Avg. Regret: 0.32

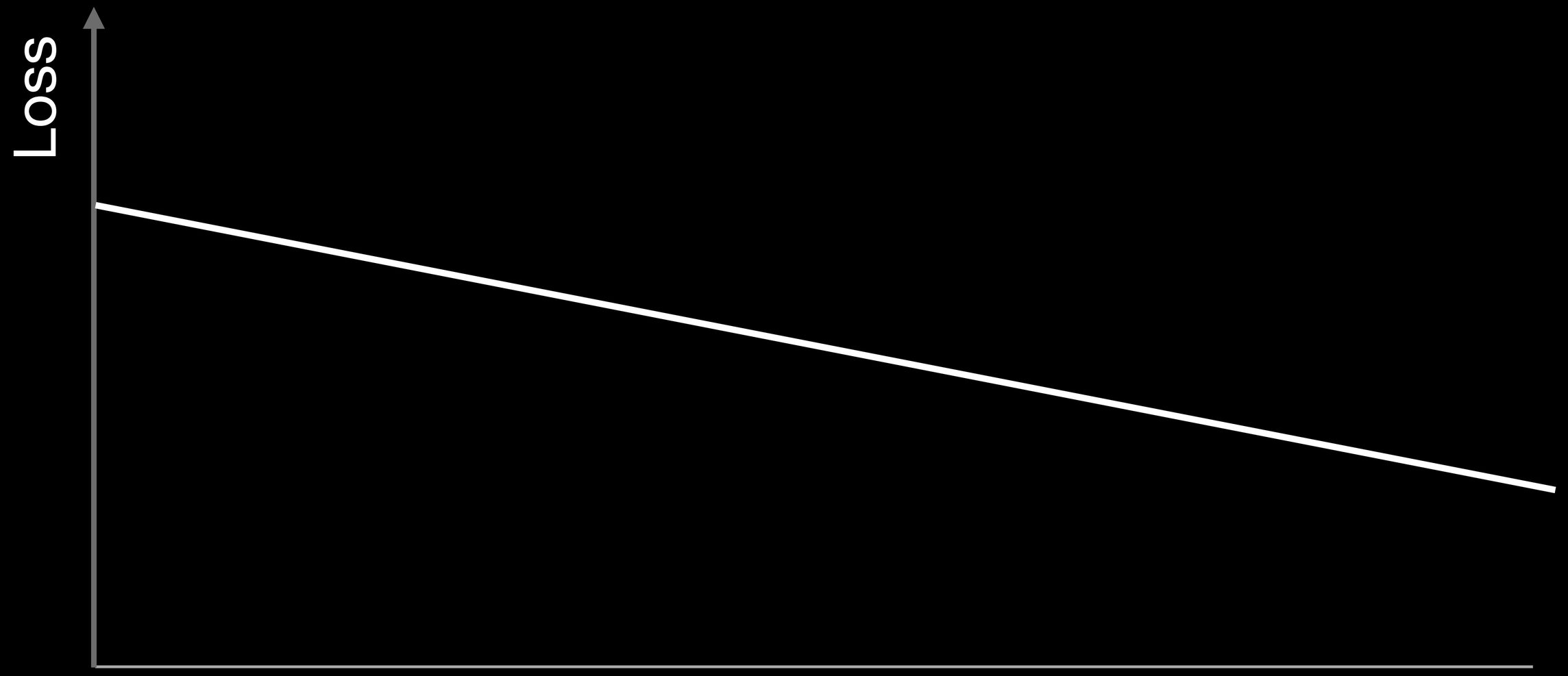
$\sum l_t$		l_1	l_2	l_3	l_4	l_5	l_6
4.5	Expert 1 	$\boxed{1.0}$	0.5	0.5	1.0	0.5	1.0
3.4	Expert 2 	0.2	$\boxed{0.5}$	$\boxed{1.0}$	0.2	1.0	0.5
1.8	Expert 3 	0.5	0.2	0.2	$\boxed{0.5}$	$\boxed{0.2}$	$\boxed{0.2}$

Avg. Regret: 0.26

FTL appears to be no regret ...

Can you think of an example that breaks FTL?

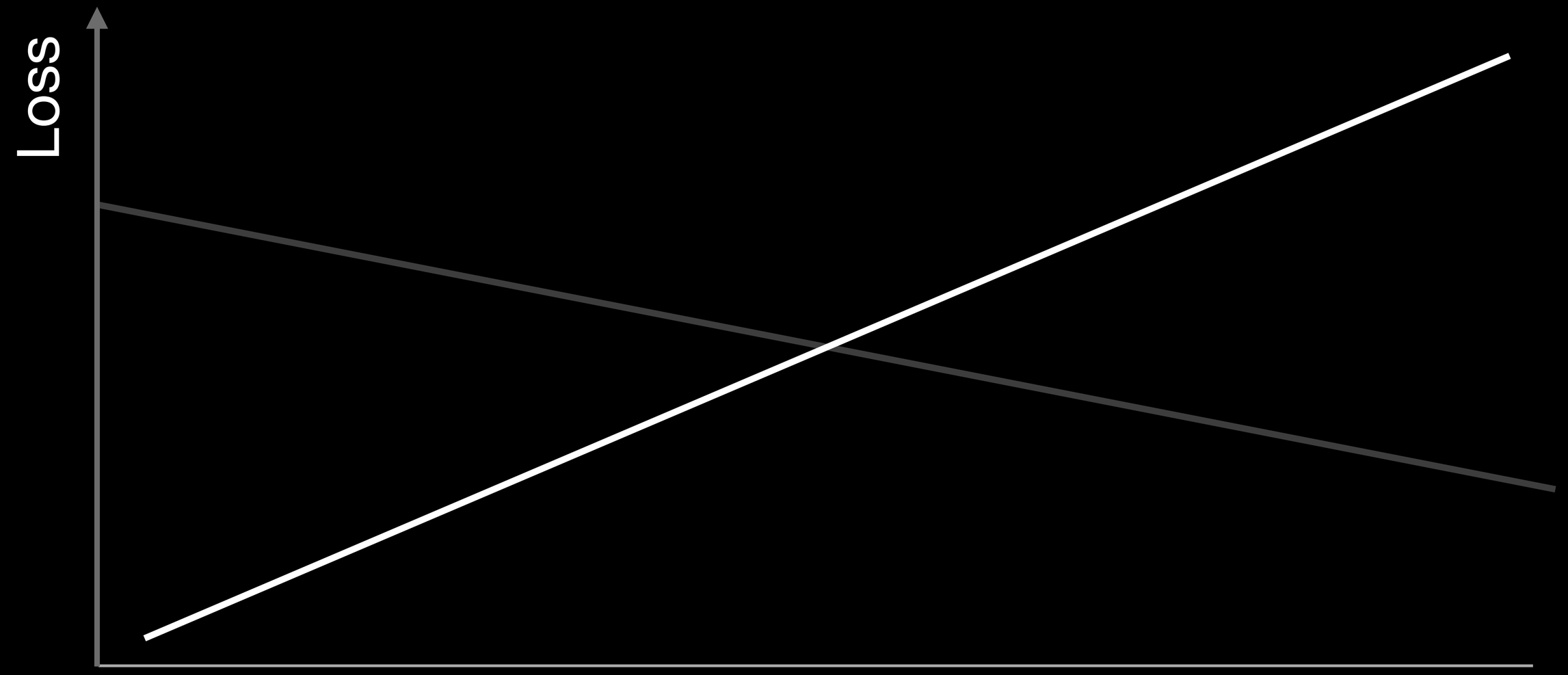
Loss = 0.75 Avg. Regret = 0.5



Choose π^1

Choose π^2

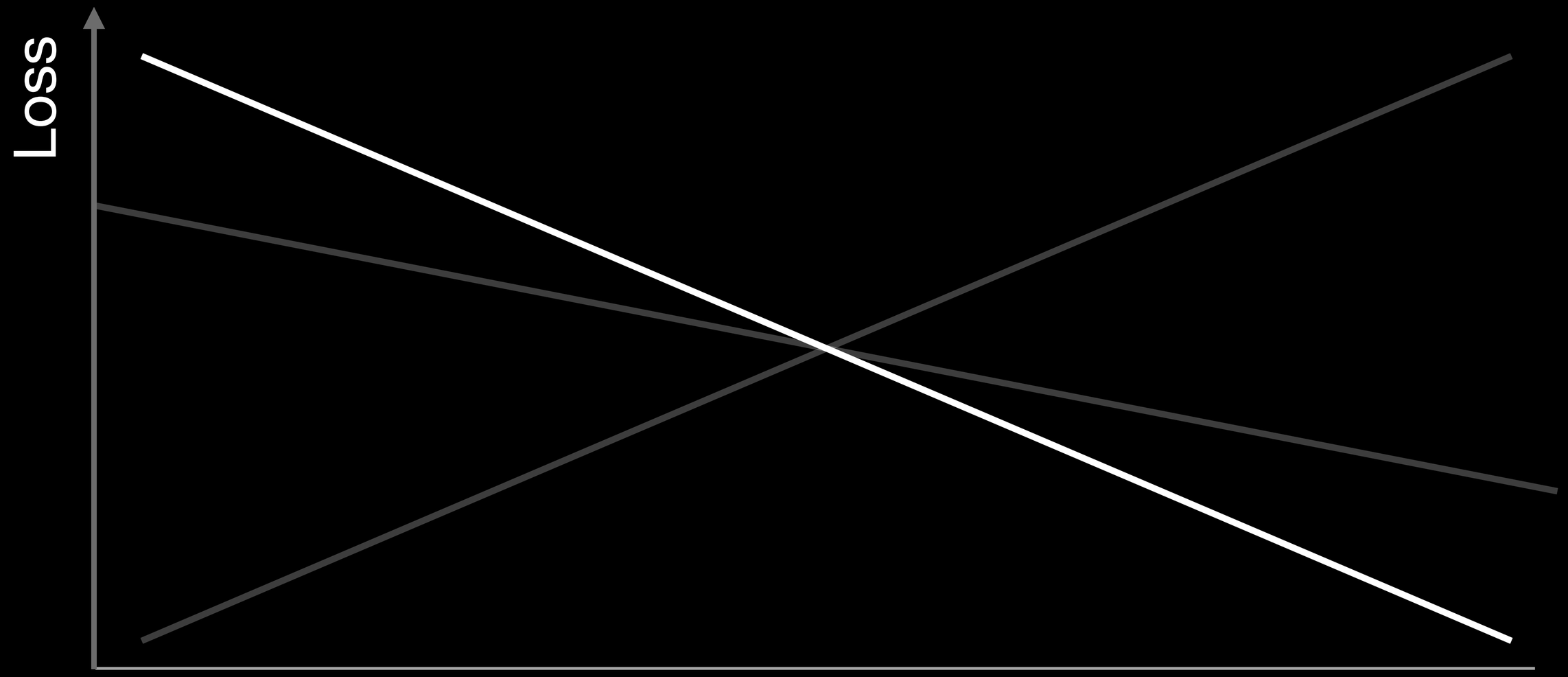
Loss = 1.0 Avg. Regret = 0.5



Choose π^1

Choose π^2

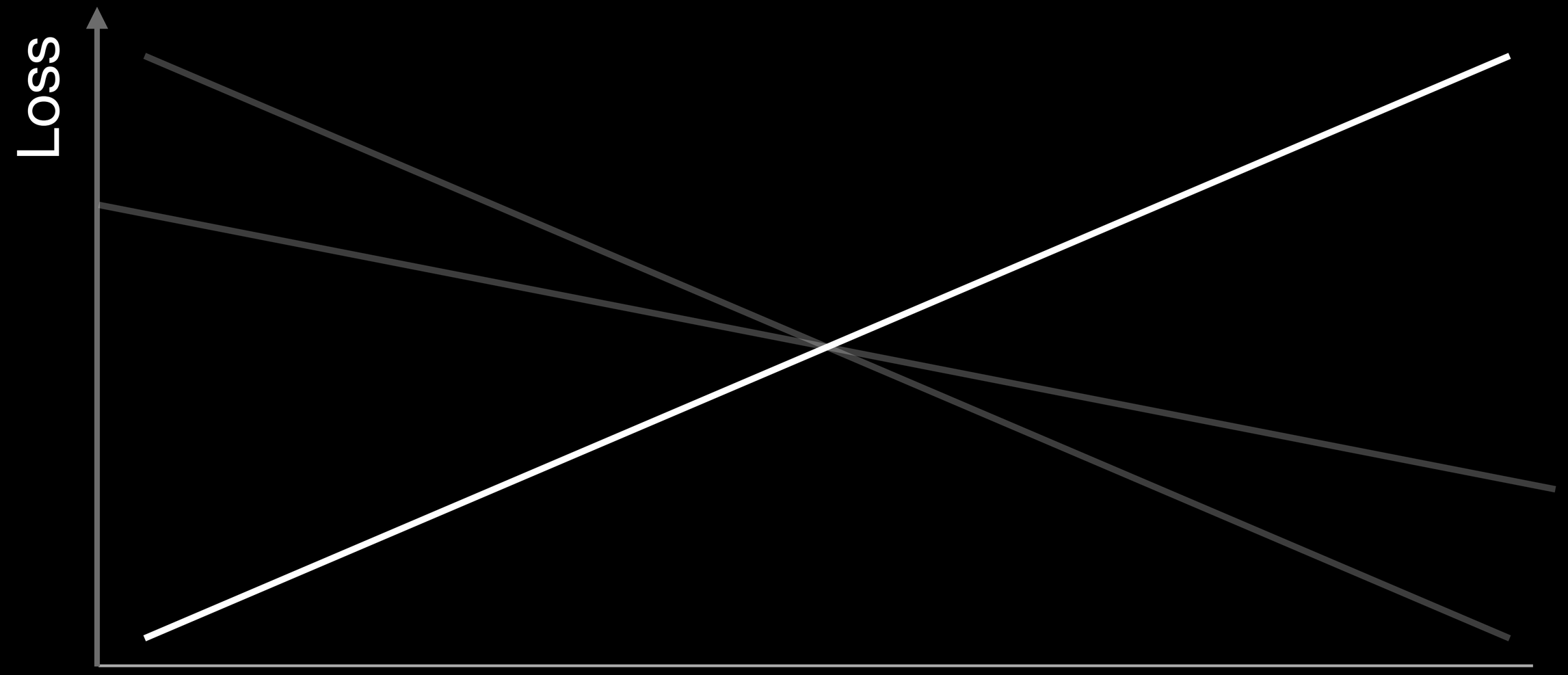
Loss = 1.0 Avg. Regret = 0.5



Choose π^1

Choose π^2

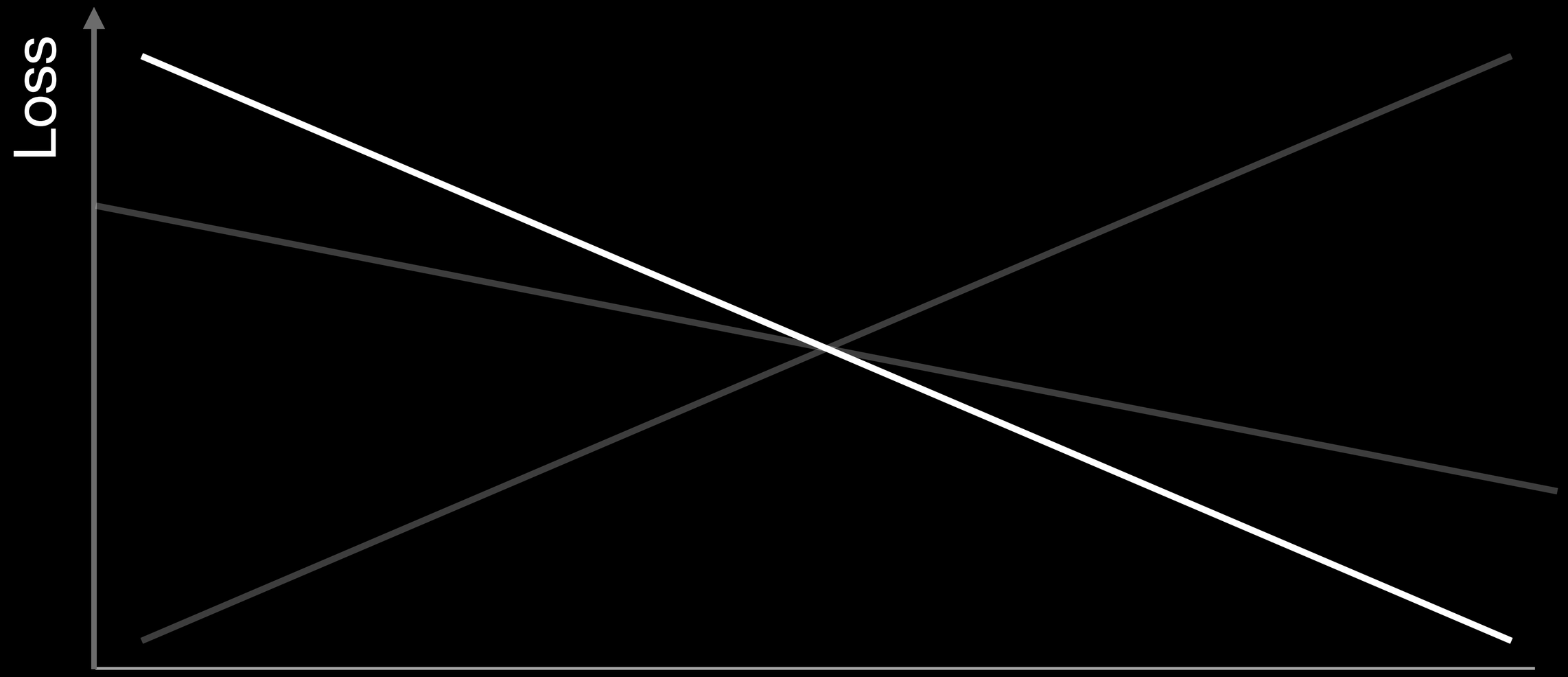
Loss = 1.0 Avg. Regret = 0.5



Choose π^1

Choose π^2

Loss = 1.0 Avg. Regret = 0.5



Choose π^1

Choose π^2

Be stable

Slowly change predictions

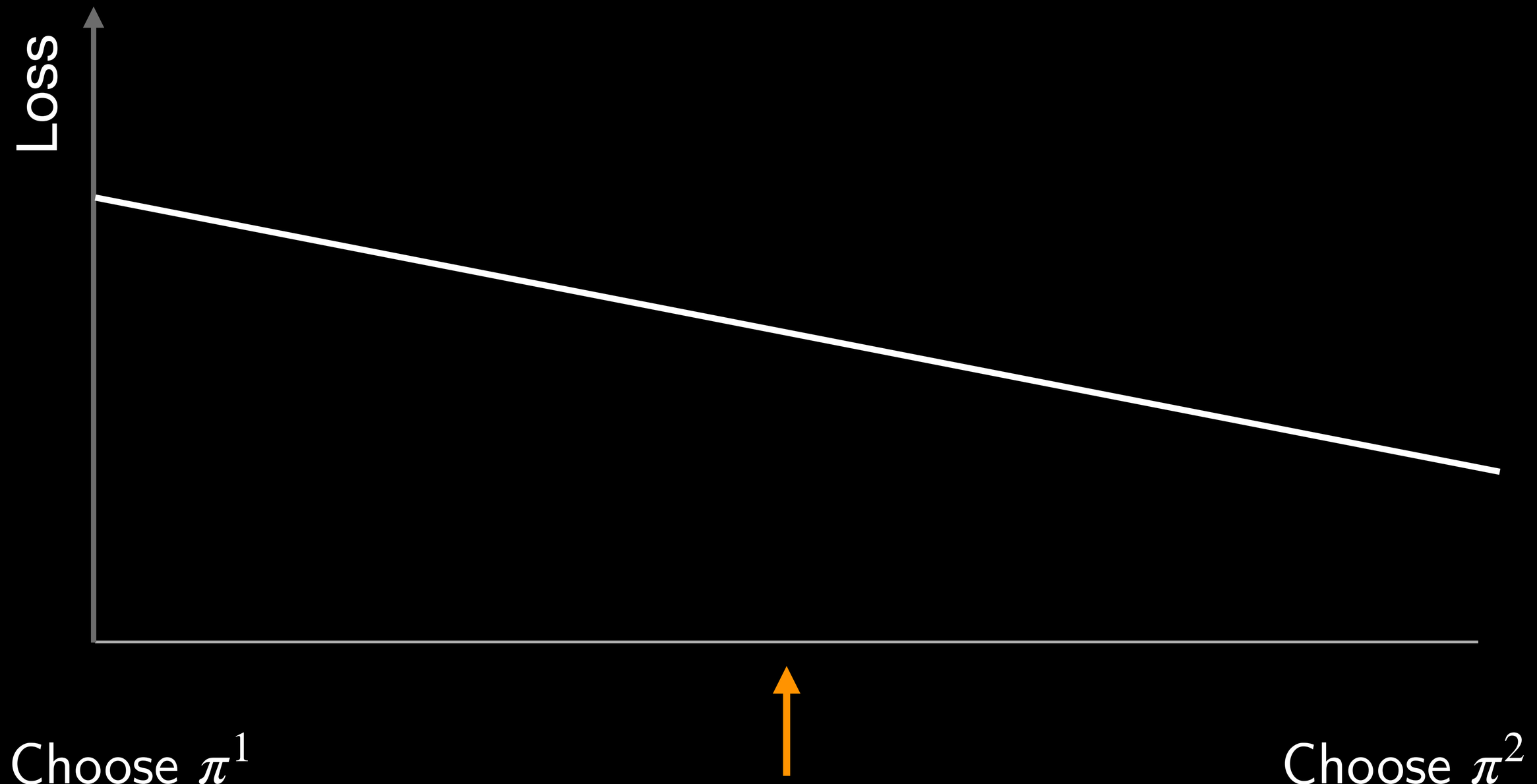
Achieve no-regret

Follow the Regularized Leader

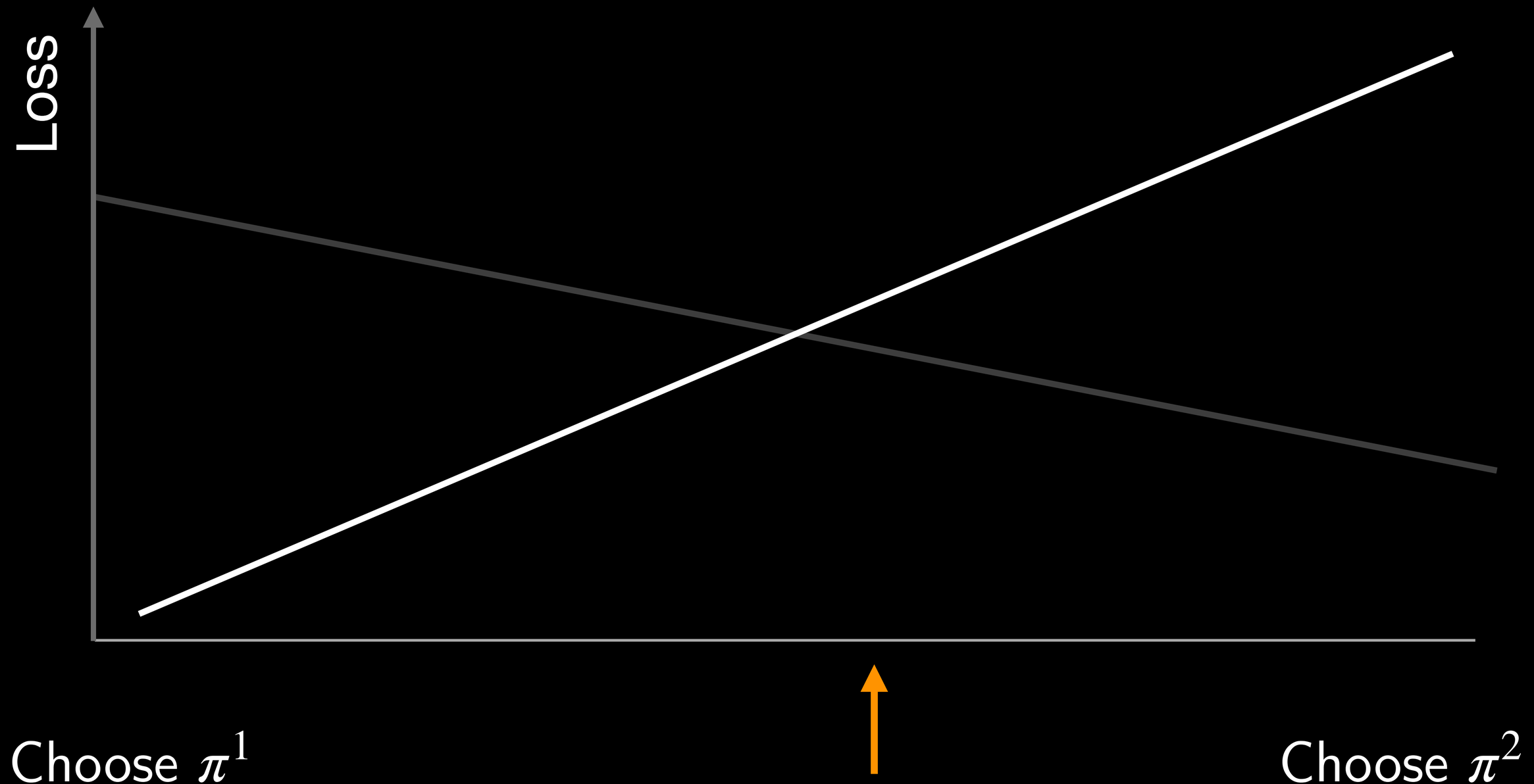
$$\pi_t = \arg\min_{\pi} \sum_{i=1}^{t-1} l_i(\pi) \quad + \eta_t \, R(\pi)$$

Strong regularization!

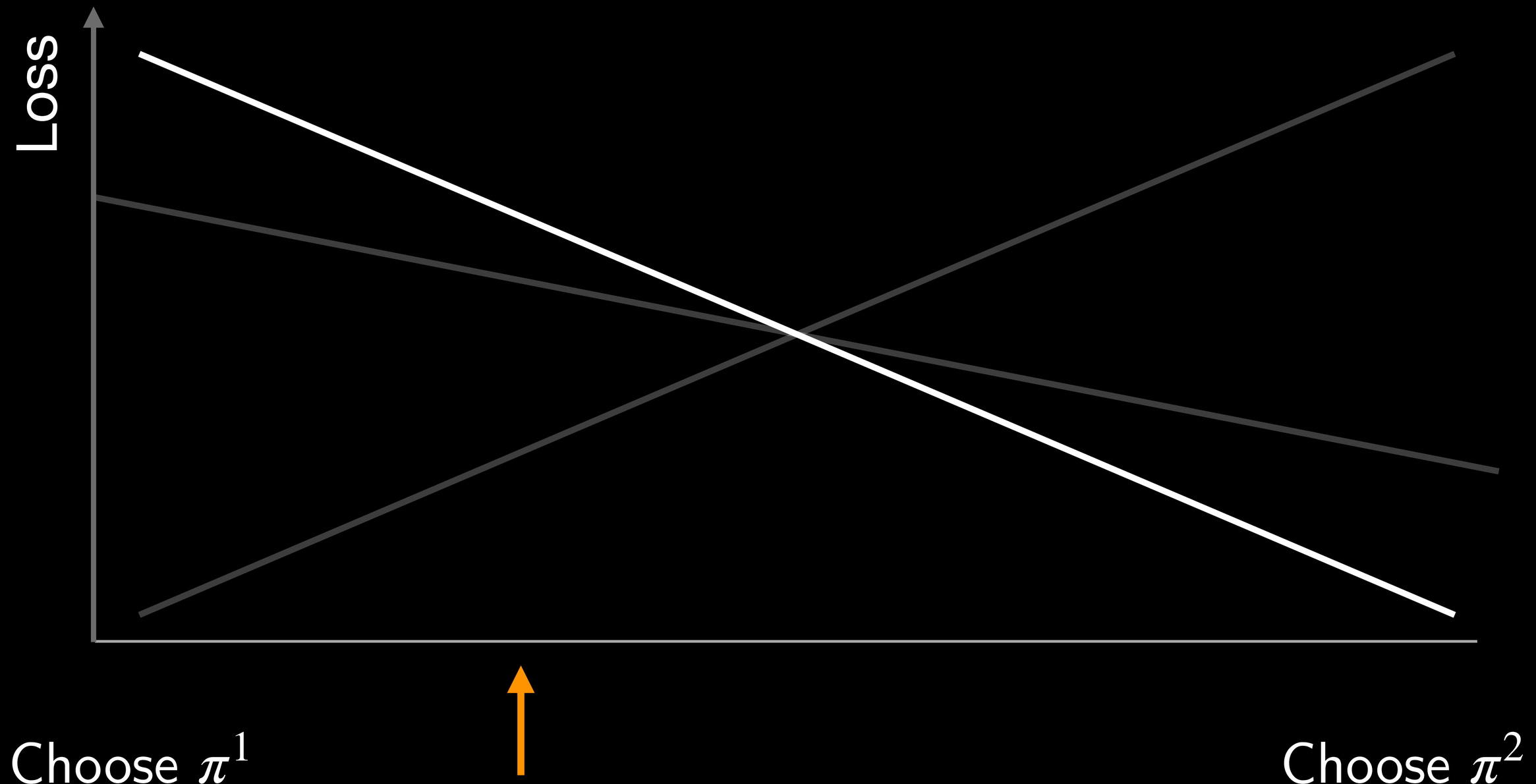
Loss = 0.5 Avg. Regret = 0.25

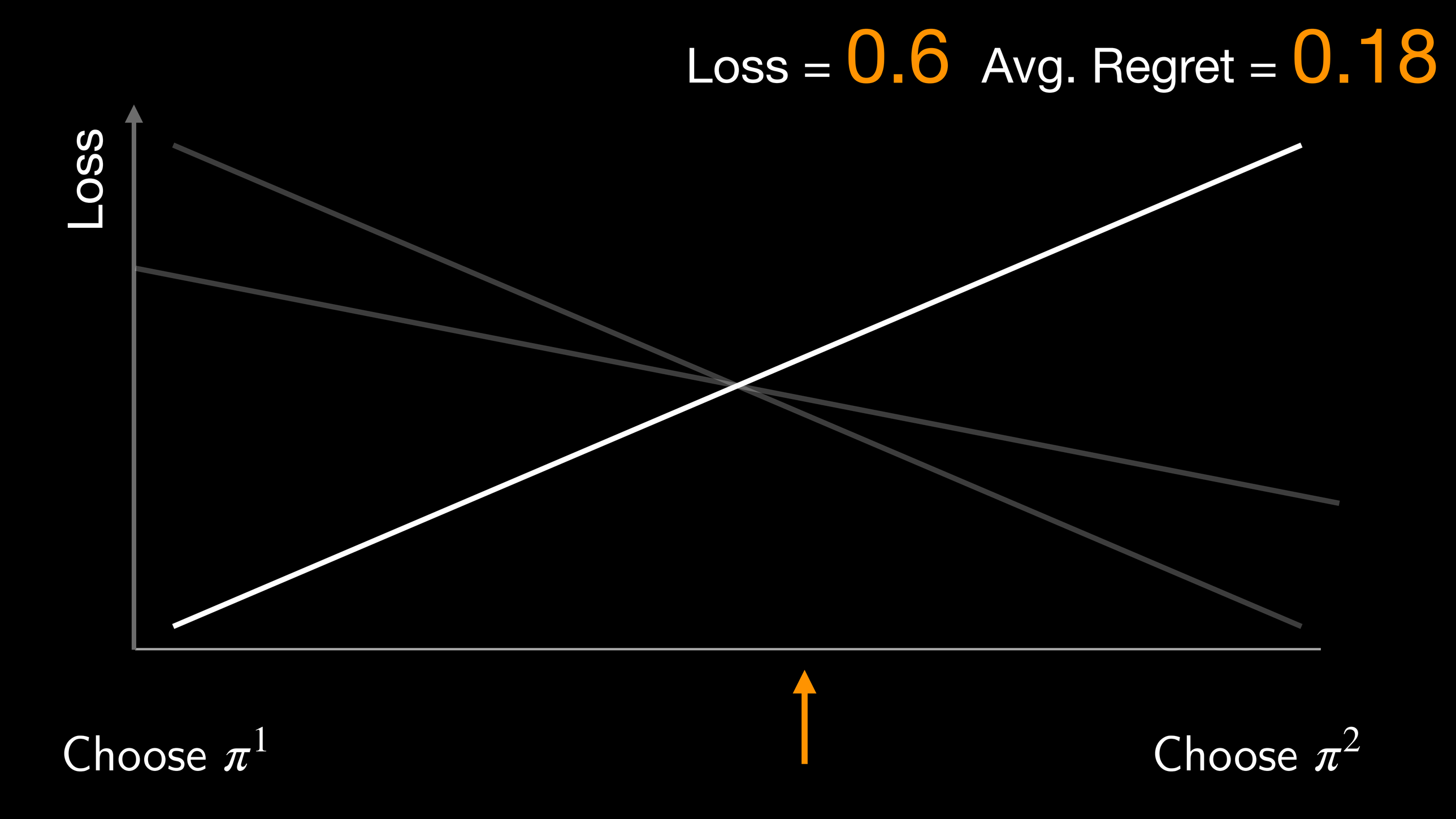


Loss = 0.6 Avg. Regret = 0.17

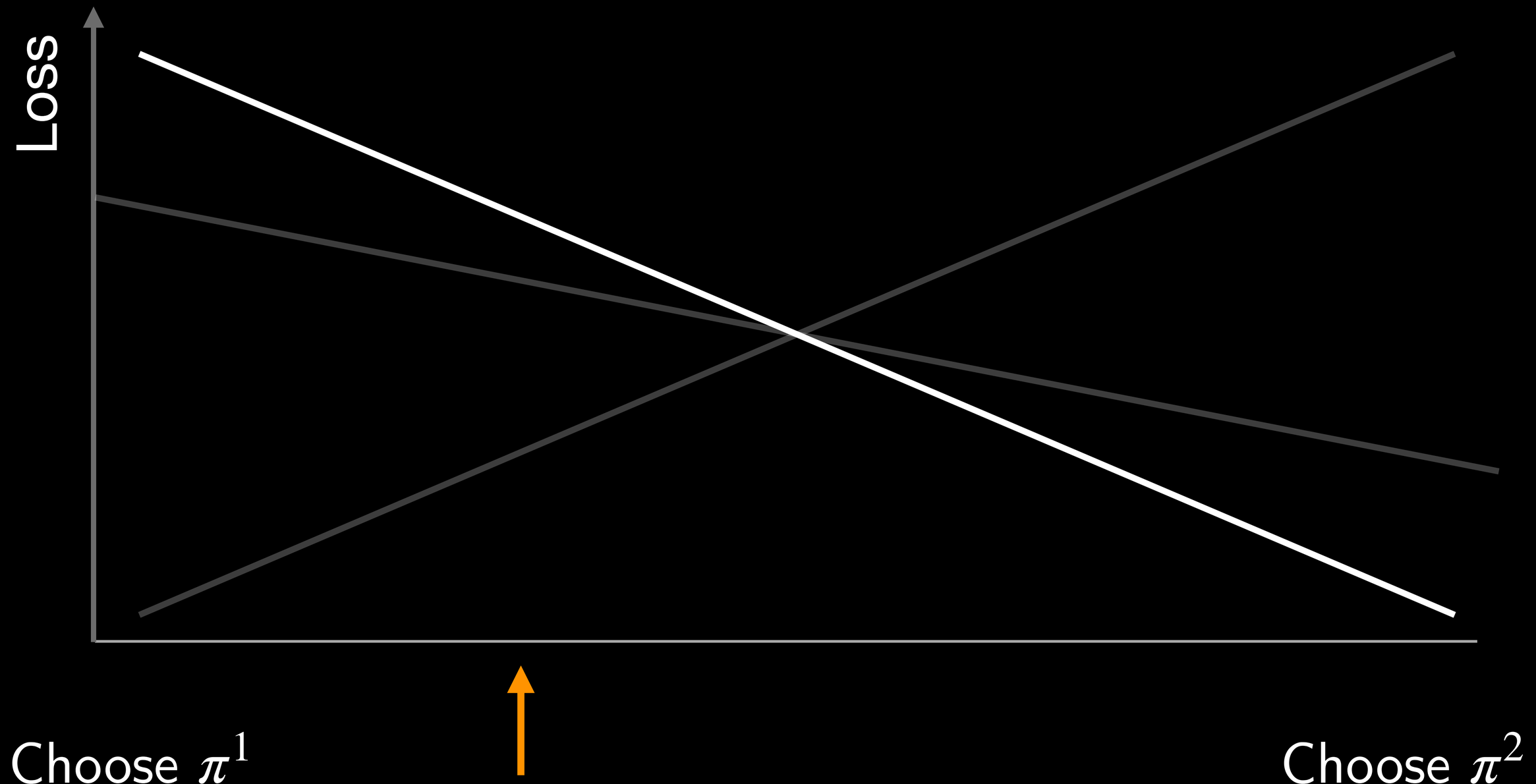


Loss = 0.78 Avg. Regret = 0.21



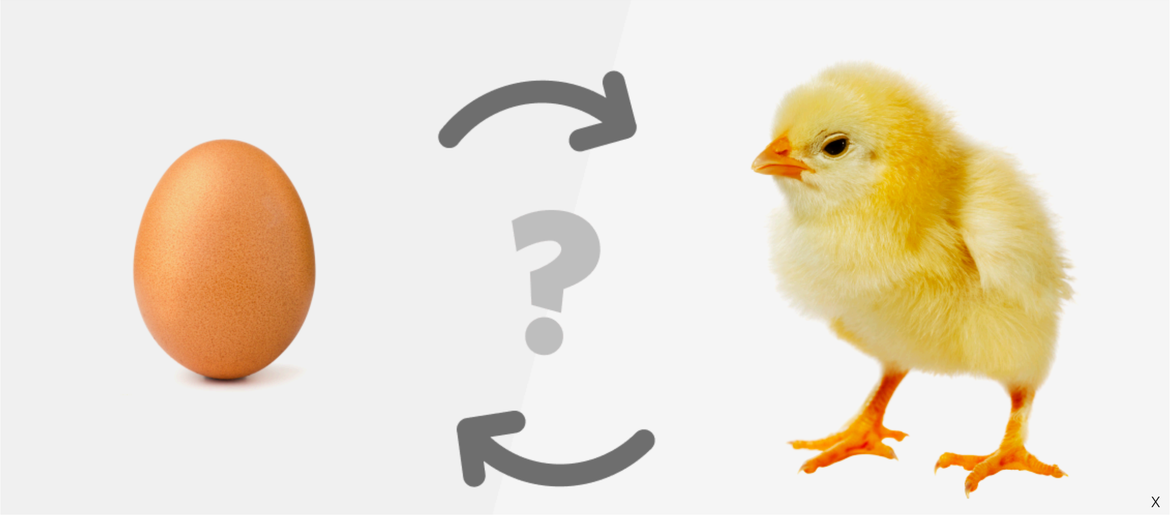

Loss = 0.6 Avg. Regret = 0.18
Loss
Choose π^1
Choose π^2

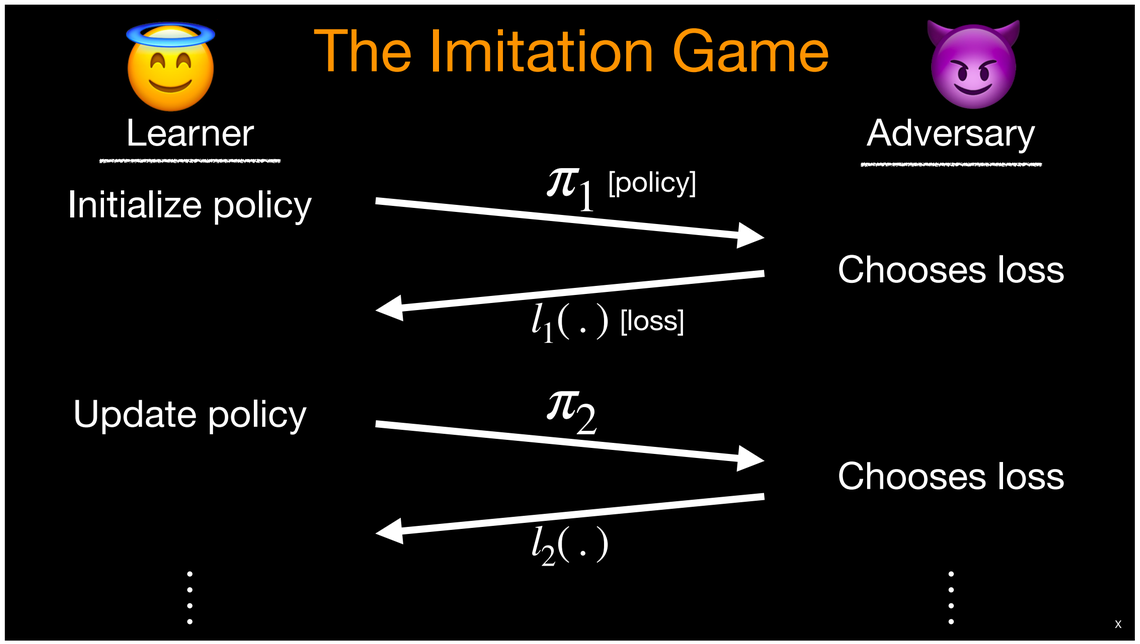
Loss = 0.78 Avg. Regret = 0.2



tl;dr

To know the distribution, you need a learner
To train a learner, you need a distribution





[Ross et al'11]

DAgger: Iteration 1