Lecture 10: Embodied AI Safety (*16-886)*

System-Level Anomalies

Instructor: Andrea Bajcsy

Carnegie Mellon University

intent ROBOTICS LAB

1. **Group 1 System:** A fixed-base manipulation vision-language-action model (e.g., OpenVLA) learned via behavior cloning.

2. **Group 2 System:** An autonomous driving pipeline that predicts human behavior and then plans robot actions.

3. **Group 3 System:** An autonomous drone delivery service that uses a perception system, LLM task-planner, and low-level action policy.

4. **Group 4 System:** A latent safety filter (like we saw in class: it uses a RGB input, a world model, and optimization within that model) that shields a behavior-cloned manipulation policy.

System-Level Uncertainties / Anomalies Brainstorm

1) What are the components you would do UQ on? What could be high-uncertainty inputs to these components?

2) Brainstorm an example of a system-level failure induced by each component.

3) How could you identify if a system-level failure has occurred? Think about metrics / methods.

4) How can you improve the closed-loop performance as a whole?

2022

A System-Level View on Out-of-Distribution Data in Robotics

Rohan Sinha, Apoorva Sharma, Somrita Banerjee, Thomas Lew, Rachel Luo,
Spencer M. Richards, Yixiao Sun, Edward Schmerling, Marco Pavone

***Abstract*— When testing conditions differ from those represented in training data, so-called out-of-distribution (OOD) inputs can mar the reliability of learned components in the modern robot autonomy stack. Therefore, coping with OOD data is an important challenge on the path towards trustworthy learning-enabled open-world autonomy. In this paper, we aim to demystify the topic of OOD data and its associated challenges in the context of data-driven robotic systems, drawing connections to emerging paradigms in the ML community that study the effect of OOD data on learned models in isolation. We argue that as roboticists, we should reason about the overall *system-level* competence of a robot as it operates in OOD conditions. We highlight key research questions around this system-level view of OOD problems to guide future research toward safe and reliable learning-enabled autonomy.**

I. Introduction

Machine learning (ML) systems are poised for widespread usage in robot autonomy stacks in the near future, driven by the successes of modern deep learning. For instance, decision-making algorithms in autonomous vehicles rely on ML-based perception and prediction models to estimate and forecast the state of the environment. As we increasingly rely on ML models to contend with the unstructured and unpredictable real world in robotics, it is paramount that we also acknowledge the shortcomings of our models, especially when we hope to deploy robots alongside humans in safety-critical settings.

In particular, ML models may behave unreliably on data that is dissimilar from the training data — inputs commonly termed *out-of-distribution* (OOD). This poses a significant challenge to deploying robots in the open world, e.g., as autonomous vehicles or home assistance robots, as such robots must interact with complex environments in conditions we cannot control or foresee. Coping with OOD inputs remains a key and largely unsolved challenge on the critical path to reliable and safe open-world autonomy. However, there is no generally-agreed-upon precise definition of what makes data OOD; instead, its definition is often left implicit and varies between problem formalisms and application contexts.

In this paper, we concretize the often nebulous notion of the OOD problem in robotics, drawing connections to existing approaches in the ML community. Critically, we advocate for a *system-level* perspective of OOD data in robotics, which considers the impacts of OOD data on downstream decision making and leverages components throughout the full autonomy stack to mitigate negative consequences. To this end, we present robotics research challenges at three timescales crucial to deploying reliable open-world autonomy: (i) real-time decision-making, (ii) episodic interaction with an environment, and (iii) the data lifecycle as learning-enabled robots are deployed, evaluated, and retrained.

We note that this paper represents neither an algorithmic contribution nor a comprehensive survey of existing paradigms and literature on OOD topics in machine learning or robotics; in fact, many of the OOD topics that we discuss, like runtime-monitoring of perception systems [1] or heuristic uncertainty quantification of deep neural networks [2], constitute well-surveyed subfields in their own right. Rather than survey specific styles of analysis or approaches tailored towards particular submodules of the autonomy stack, our goal in this work is to provide an overview of the core considerations and system-wide challenges that we see as essential areas of robotics research activity for the coming years. Our contribution thus is to establish perspective and context to galvanize more research interest in a topic that we view as critical to improving the reliability of autonomous robots.

II. Running Examples

To better describe the challenges that OOD data creates in learning-enabled robotic systems, we use the two future autonomy systems shown in Figure 1 as running examples in this paper. These conceptual examples highlight the plurality of applications and design paradigms used to leverage ML in the design of robotic systems.

Autonomous Drone Delivery Service: Firstly, we consider an autonomous drone delivering packages in a city. As illustrated in Figure 1, this robot uses several learning-enabled components in its autonomy stack. The delivery drone has to make explainable decisions and meet stringent safety requirements by regulatory agencies to be deployed among humans. Crucially, to maintain these reliability requirements in rare and unforeseen circumstances, the drone needs mechanisms to detect and manage OOD inputs.

Robotic Manipulators Assisting in the Home: Secondly, we consider the deployment of robotic manipulators to assist with various tasks in and around the home, as shown in Figure 1. The manipulators' tasks are so diverse and unstructured that we consider a general manipulation policy trained in an end-to-end fashion in a controlled environment, as commonly considered in the reinforcement learning (RL) community. When we deploy these manipulators in people's homes, the environments and contexts that these robots encounter invariably differ from the lab or simulated environments

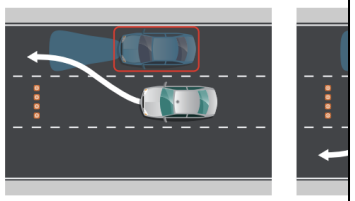
The authors are with the Autonomous Systems Lab at Stanford University, Stanford, CA. {rhnsinha, apoorva, somrita,

This paper has been published in the Proceedings of Robotics: Science and Systems (RSS) 2023.
Please cite the paper as: P. Antonante, S. Veer, K. Leung, X. Weng, L. Carlone and M. Pavone,
"Task-Aware Risk Estimation of Perception Failures for Autonomous Vehicles", *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

Task-Aware Risk Estimation of Perception Failures for Autonomous Vehicles

Pasquale Antonante*, Sushant Veer†, Karen Leung†‡, Xinshuo Weng†, Luca Carlone*, and Marco Pavone†§
* Massachusetts Institute of Technology
† NVIDIA Research
‡ University of Washington
§ Stanford University

***Abstract*—Safety and performance are key enablers for autonomous driving: on the one hand we want our autonomous vehicles (AVs) to be safe, while at the same time their performance (*e.g.,* comfort or progression) is key to adoption. To effectively walk the tight-rope between safety and performance, AVs need to be risk-averse, but not entirely risk-avoidant. To facilitate safe-yet-performant driving, in this paper, we develop a *task-aware* risk estimator that assesses the risk a perception failure poses to the AV's motion plan. If the failure has no bearing on the safety of the AV's motion plan, then regardless of how egregious the perception failure is, our task-aware risk estimator considers the failure to have a low risk; on the other hand, if a seemingly benign perception failure severely impacts the motion plan, then our estimator considers it to have a high risk. In this paper, we propose a task-aware risk estimator to decide whether a safety maneuver needs to be triggered. To estimate the task-aware risk, first, we leverage the perception failure — detected by a perception monitor— to synthesize an alternative plausible model for the vehicle's surroundings. The risk due to**



(a) Task-relevant failure (b)

Fig. 1. **Illustration of task-aware perceptio**
car is the ego vehicle and the blue car is an ext
example, the non-ego vehicle has not been det
of the ego vehicle. Then, Fig. 1(a) depicts a
as the ego vehicle's motion plan will likely c
due to the misdetection. Fig. 1(b) depicts a no
as the ego vehicle's motion plan will not lead
vehicle, regardless of the perception failure.

performant driving: an example hig

Use sensor redundancy + scene generation techniques to hypothesize plausible scenes and identify risk of current perception failure

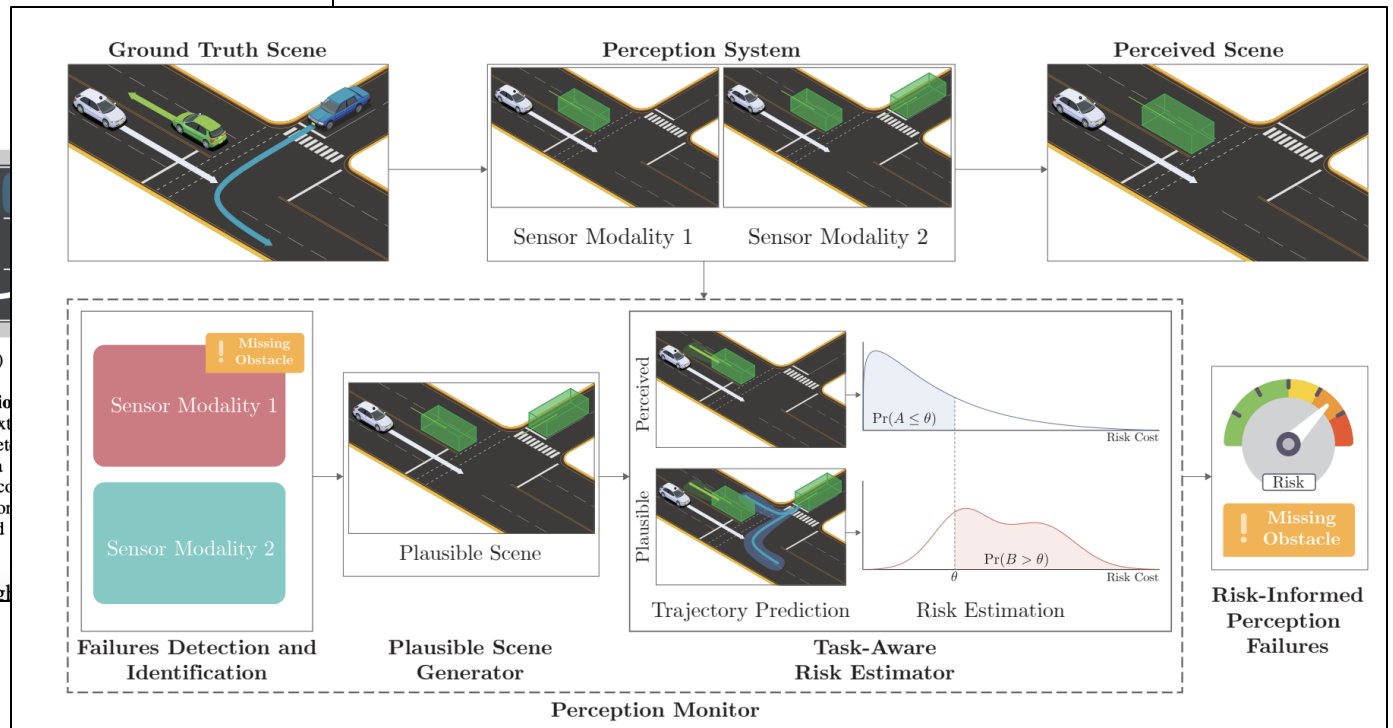


Fig. 2. **Task-aware perception monitor overview.** The scene contains the ego vehicle (white car) and two non-ego agents (green and blue car). The top row shows a scenario in which a perception system fails to detect an obstacle (the blue car): one of the two sensor modalities used by the perception system is not able to detect the obstacle (top-center subfigure), inducing a missing-obstacle failure in the perception output (top-right subfigure). The bottom row depicts the proposed task-aware perception monitor. The failure detection and identification module detects that sensor 1 is failing (for example using spatio-temporal information). The plausible scene generator, uses the information about the active failures, generates a plausible scene from the perceived scene. Finally, the task-aware risk estimator computes the risk associated with the failure. The shaded (green and blue) regions in the bottom-row scenes represent the uncertainty in the trajectories, as computed by the non-ego trajectory prediction module. The possible trajectories induce a distribution of risk costs for each scene, which are used to estimate the risk associated with a perception failure. If the risk in the plausible scene is significantly higher than the risk in the perceived scene, we detect the failure as task relevant. Our detector uses a statistical tool called copula to estimate the tail dependency between the two cost distributions.

Discovering Closed-Loop Failures of Vision-Based Controllers via Reachability Analysis

Kaustav Chakraborty[1], Somil Bansal[1]

***Abstract*—Machine learning driven image-based controllers allow robotic systems to take intelligent actions based on the visual feedback from their environment. Understanding when these controllers might lead to system safety violations is important for their integration in safety-critical applications and engineering corrective safety measures for the system. Existing methods leverage simulation-based testing (or falsification) to find the failures of vision-based controllers, i.e., the visual inputs that lead to closed-loop safety violations. However, these techniques do not scale well to the scenarios involving high-dimensional and complex visual inputs, such as RGB images. In this work, we cast the problem of finding closed-loop vision failures as a Hamilton-Jacobi (HJ) reachability problem. Our approach blends simulation-based analysis with HJ reachability methods to compute an approximation of the backward reachable tube (BRT) of the system, i.e., the set of unsafe states for the system under vision-based controllers. Utilizing the BRT, we can tractably and systematically find the system states and corresponding visual inputs that lead to closed-loop failures. These visual inputs can be subsequently analyzed to find the input characteristics that might have caused the failure. Besides its scalability to high-dimensional visual inputs, an explicit computation of BRT allows the proposed approach to capture non-trivial system failures that are difficult to expose via random simulations. We demonstrate our framework on two case studies involving an RGB image-based neural network controller for (a) autonomous indoor navigation, and (b) autonomous aircraft taxiing.**

I. Introduction

RECENT advances in computer vision and deep learning have enabled autonomous systems to employ vision-result in a system failure. In addition to reasoning about system safety, these failure modes might be useful in engineering corrective measures for the system.

While techniques from adversarial learning and robust optimization have been used to find "adversarial" inputs for vision components, they tend to focus on the component-level safety analysis, i.e., detecting failures or errors only within the vision component, ignoring their effect on the downstream system and the overall robot safety. Indeed, not all vision failures are equal from the robot safety standpoint. For instance, the same prediction error by a visual policy for a high-speed drone can be much more catastrophic near a wall compared to an empty hallway. Thus, it is imperative that we analyze these vision modules in conjunction with the robot dynamics. To that end, formal verification techniques have been used for system-level (or closed-loop) safety analysis of dynamical systems; however, their direct application to vision-based controllers remains impractical due to these controllers' high-dimensional and complicated input spaces and the lack of mathematical models relating the robot state to the visual input at that state. Simulation-based testing has been a promising approach to overcoming these challenges; by treating the system as a black box, one can search for system trajectories that result in a failure under the vision-based controller. However, this process (also called falsification) can be highly time-consuming, and it struggles with exposing long-tail of system failures.

In this work, we cast the problem of finding closed-loop vision failures as a Hamilton-Jacobi (HJ) reachability problem

Use reachability analysis to discover perception errors that lead to closed-loop failures

Task-Driven Out-of-Distribution Detection with Statistical Guarantees for Robot Learning

Alec Farid* **Sushant Veer*** **Anirudha Majumdar**
Department of Mechanical and Aerospace Engineering, Princeton University
{afarid, sveer, ani.majumdar}@princeton.edu

Abstract: Our goal is to perform *out-of-distribution (OOD) detection*, i.e., to detect when a robot is operating in environments that are drawn from a different distribution than the environments used to train the robot. We leverage Probably Approximately Correct (PAC)-Bayes theory in order to train a policy with a *guaranteed bound* on performance on the training distribution. Our key idea for OOD detection then relies on the following intuition: violation of the performance bound on test environments provides evidence that the robot is operating OOD. We formalize this via statistical techniques based on p-values and concentration inequalities. The resulting approach (i) provides guaranteed confidence bounds on OOD detection, and (ii) is *task-driven* and sensitive only to changes that impact the robot's performance. We demonstrate our approach on a simulated example of grasping objects with unfamiliar poses or shapes. We also present both simulation and hardware experiments for a drone performing vision-based obstacle avoidance in unfamiliar environments (including wind disturbances and different obstacle densities). Our examples demonstrate that we can perform task-driven OOD detection within just a handful of trials. Comparisons with baselines also demonstrate the advantages of our approach in terms of providing statistical guarantees and being insensitive to task-irrelevant distribution shifts.

Keywords: Out-of-distribution detection, generalization, PAC-Bayes

1 Introduction

Imagine a drone trained to perform vision-based navigation using a dataset of indoor environments and deployed in environments with varying wind conditions, obstacle densities, or lighting (Fig. 1). Similarly, consider a robot arm manipulating a new set of objects or an autonomous vehicle deployed in a new city. State-of-the-art techniques for learning-based control of robots typically struggle to generalize to such *out-of-distribution* (OOD) environments. This lack of OOD generalization is par

Use statistical generalization theory. Intuition: if the costs incurred in new environments violate the policy bound, then the robot is operating OOD

Unpacking Failure Modes of Generative Policies: Runtime Monitoring of Consistency and Progress

Christopher Agia[1], Rohan Sinha[1], Jingyun Yang[1], Zi-ang Cao[1],
Rika Antonova[1], Marco Pavone[1,2], Jeannette Bohg[1]

[1]Stanford University, [2]NVIDIA Research

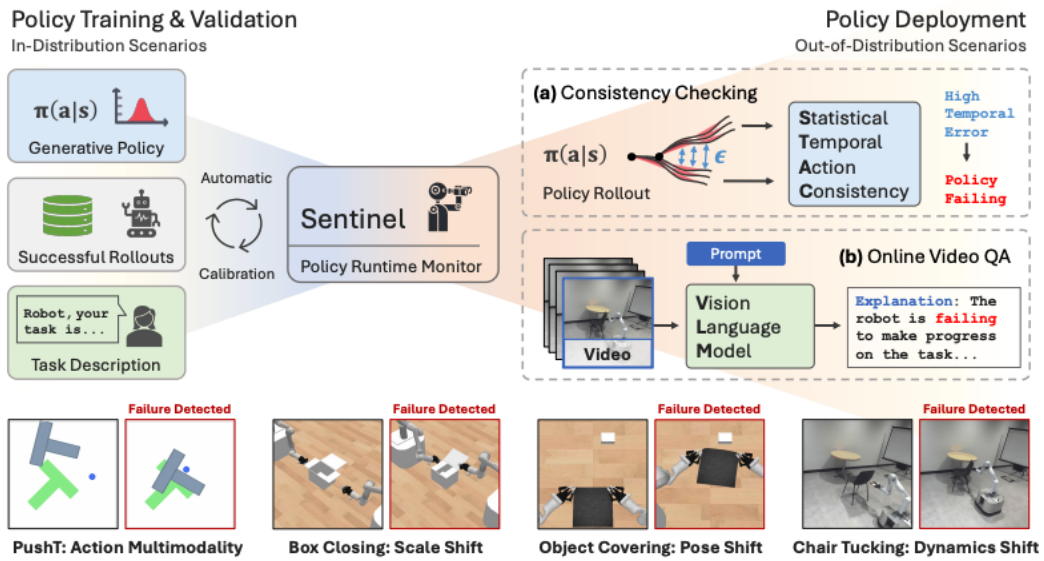


Figure 1: We present **Sentinel**, a runtime monitor that detects unknown failures of generative robot policies at deployment time. Constructing Sentinel requires only a set of successful policy rollouts and a description of the task, from which it detects diverse failures by monitoring **(a)** the temporal consistency of action-chunk distributions generated by the policy and **(b)** the task progress of the robot(s) through video QA with Vision Language Models.

Abstract: Robot behavior policies trained via imitation learning are prone to failure under conditions that deviate from their training data. Thus, algorithms that monitor learned policies at test time and provide early warnings of failure are necessary to facilitate scalable deployment. We propose **Sentinel**, a runtime monitoring framework

1) Use temporal consistency of action generations and 2) VLMs to detect task progression failures

Not All Errors Are Made Equal: A Regret Metric for Detecting System-level Trajectory Prediction Failures

Kensuke Nakamura[1] **Ran Tian**[2] **Andrea Bajcsy**[1]
[1]Carnegie Mellon University [2] UC Berkeley
{kensuken, abajcsy}@andrew.cmu.edu, rantian@berkeley.edu

Abstract: Robot decision-making increasingly relies on data-dri
diction models when operating around people. While these mo
to mispredict in out-of-distribution interactions, only a subset
rors impact downstream robot performance. We propose cha
"system-level" prediction failures via the mathematical notion
regret interactions are precisely those in which mispredictions d
loop robot performance. We further introduce a probabilisti
of regret that calibrates failure detection across disparate deplo
and renders regret compatible with reward-based and reward-fr
ative) planners. In simulated autonomous driving interactions
igation interactions deployed on hardware, we showcase that
failure metric can be used offline to automatically identify cha
loop human-robot interactions that generative human predictors
ners previously struggled with. We further find that the very pr
regret data during predictor fine-tuning is highly predictive of cl
performance improvement. Additionally, fine-tuning with the info
nificantly smaller high-regret data (23% of deployment data) is
fine-tuning on the full deployment dataset, indicating a promising
ciently mitigating system-level human-robot interaction failures.
https://cmu-intentlab.github.io/not-all-err

Keywords: Human-Robot Interaction, Trajectory Prediction, Failure Detection

Now…

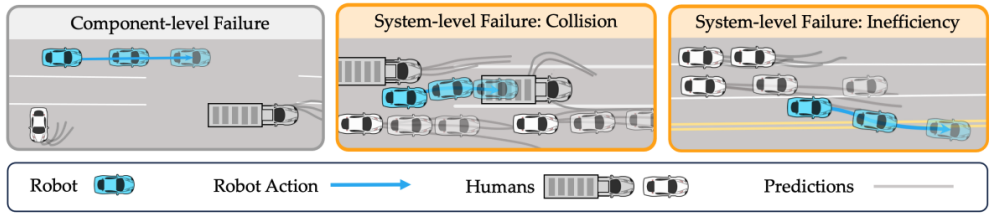


Figure 1: All scenarios have component-level prediction failures: mispredicting that parked cars will turn (left), marooned truck will to move (center), and nearby cars will lane change (right). But, only the center and right scenarios have system-level prediction failures which impact robot performance.

caused undesirable closed-loop robot behavior: collisions in the center and inefficiency in the right. The core challenge we address is the automatic detection of such system-level prediction failures.

Our key insight is that **the mathematical notion of regret is a rigorous way to identify system-level prediction failures.** High-regret interactions are precisely those in which mispredictions caused the robot to make a suboptimal decision in hindsight. In Section 4 we formalize this idea and introduce a probabilistic generalization of regret that no longer depends on explicit reward functions for regret computation, extending its applicability to reward-free planners such as generative models. In Section 6, we extract system-level failures offline from a dataset of collected deployment interactions and compare our approach to alternative failure detection methods with a state-of-the-

Robot Planner

$$\boldsymbol{a}^{\mathrm{R}} = \pi_{\theta}(s, P_{\phi}, C)$$

cost-based *planner:* θ *is learned cost fn*
generative *planner:* θ *is NN weights*

Human Behavior Predictor

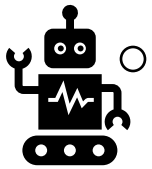
$$P_{\phi}: (\boldsymbol{s}, C) \mapsto \tilde{\boldsymbol{a}}^{\mathrm{H}}$$

state history & context

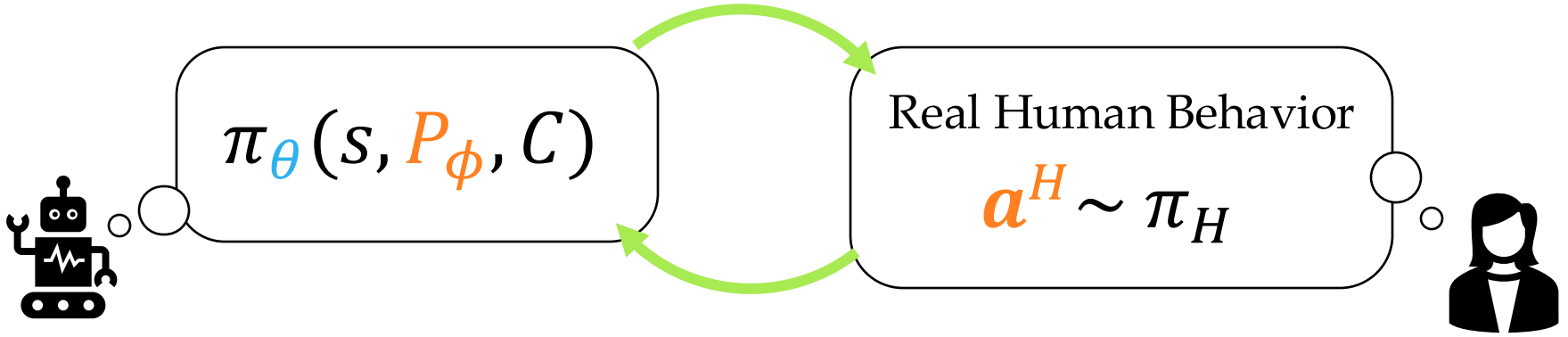
predicted future human actions

Real Human Behavior

$$\boldsymbol{a}^{H} \sim \pi_{H}$$



Which prediction failures *"mattered"* during deployment interactions?
Can we *improve* our interaction models over repeated interactions?



All deployment data

Natural growth over time…

…& human interaction distribution shifts

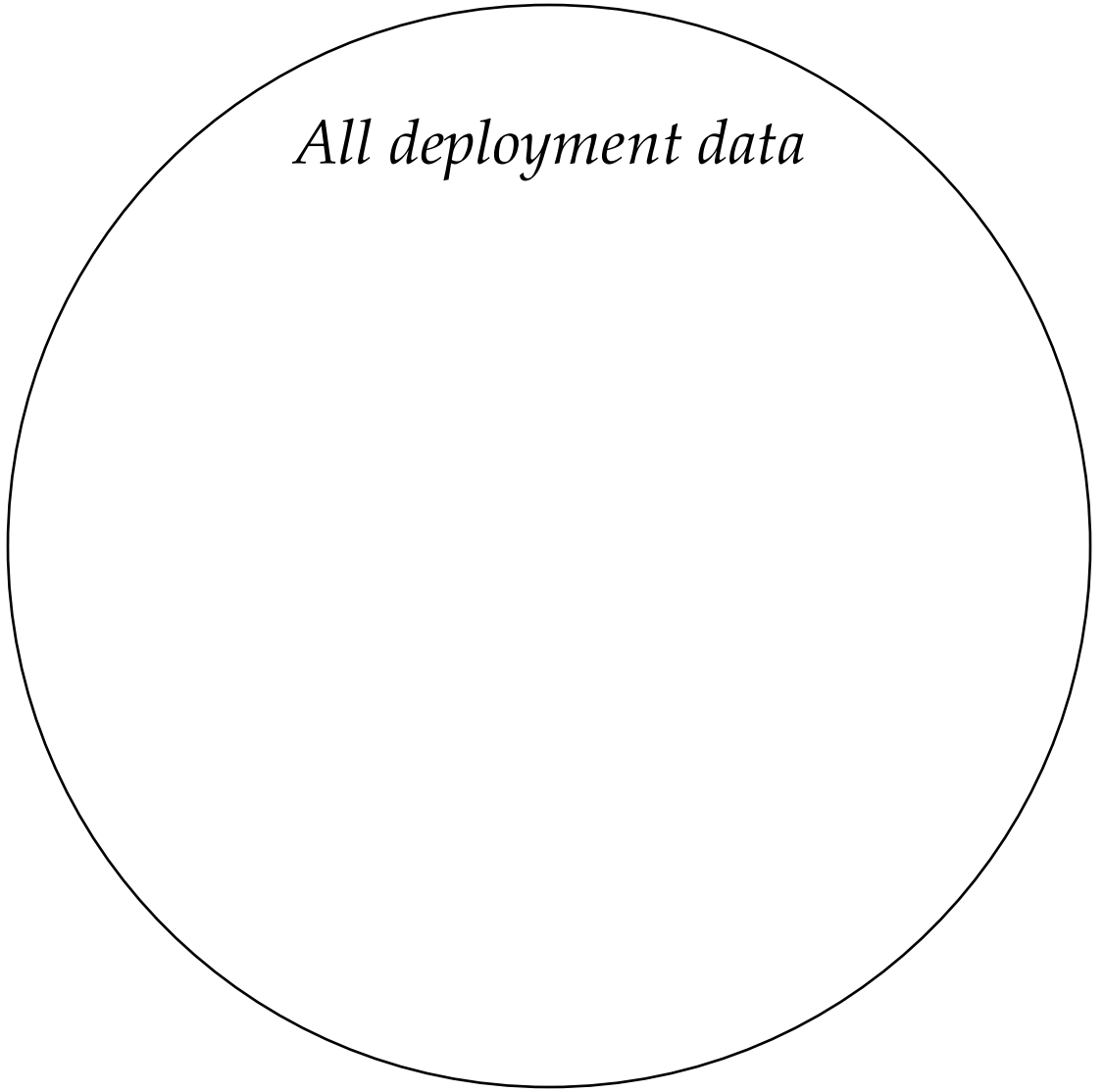
You

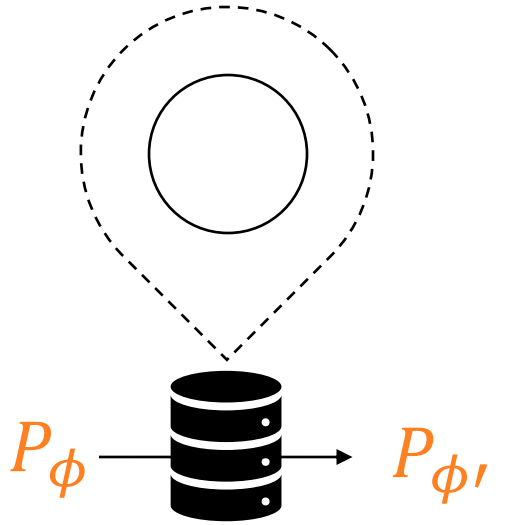
Tell me what happened on Tuesday February 27 2024 in Pittsburgh Pennsylvania, USA

ChatGPT

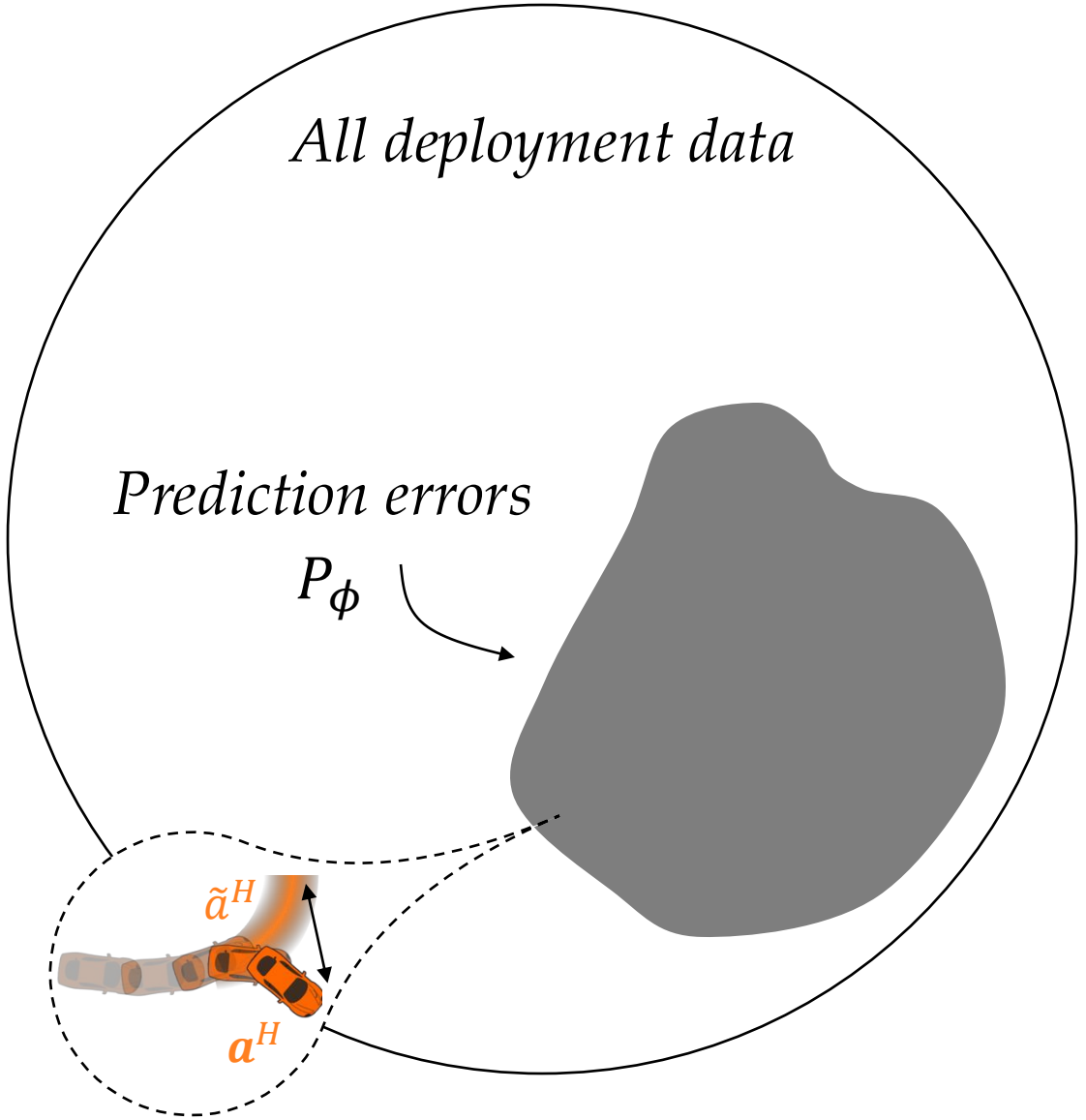
I can't provide real-time or the most recent updates, including events from specific dates such as February 27, 2024, in Pittsburgh, Pennsylvania, or anywhere else, due to my last training data being in April 2023. To find out what happened on that specific date, I recommend checking the latest news from reliable sources or local news websites for Pittsburgh, such as the Pittsburgh Post-Gazette or Tribune-Review. You can also look for any updates on social media platforms or news aggregators for the most current information.

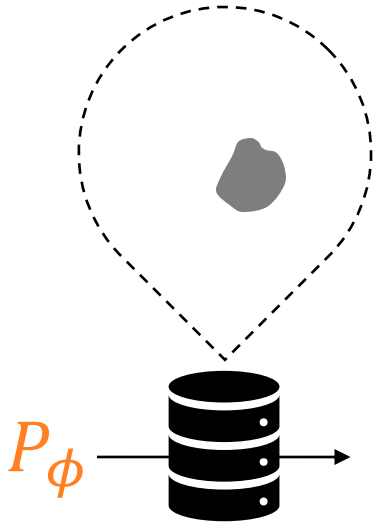
Source: https://twitter.com/djbaskin





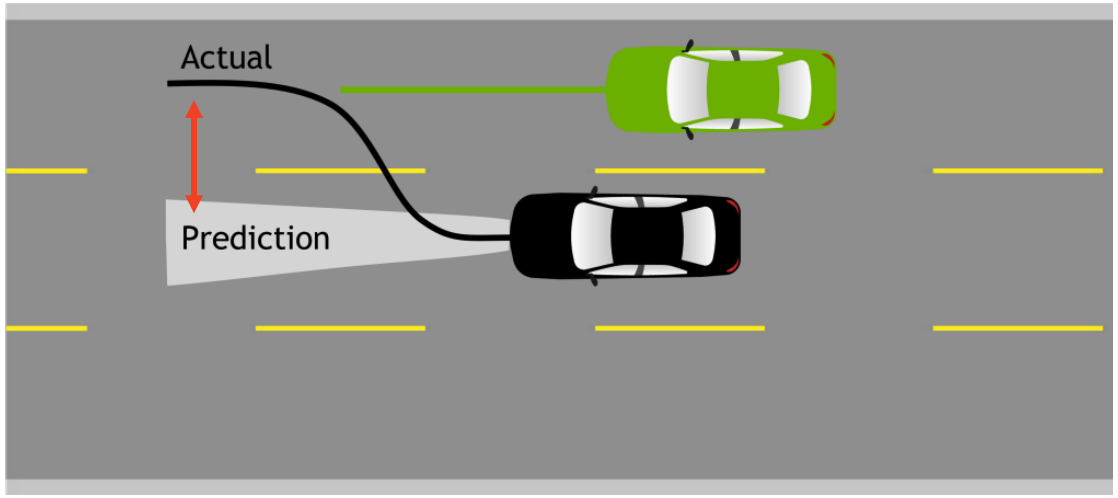
- *limited model capacity (esp. if you want fast inference)*



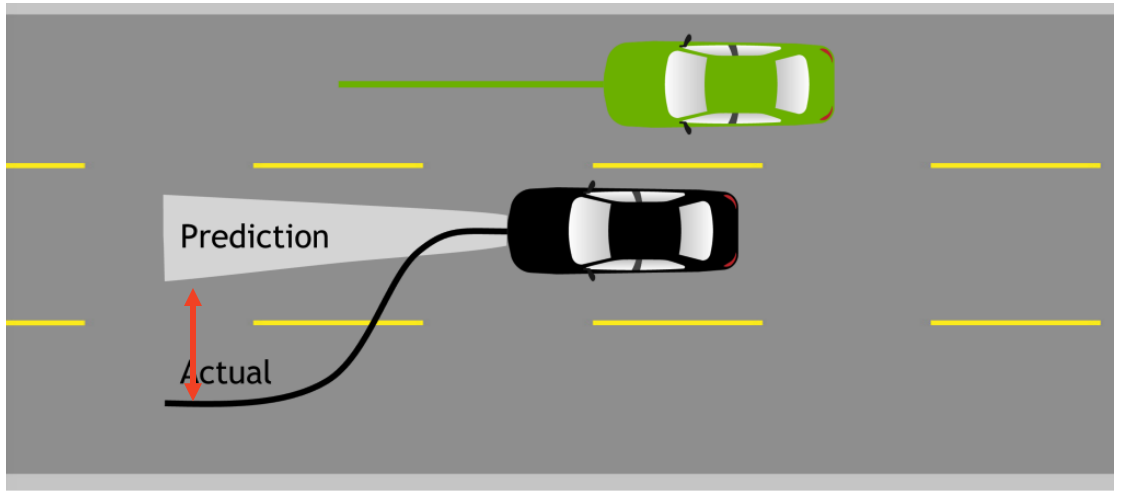


⊖ *may still waste capacity on learning irrelevant behaviors*

Mis-prediction DOES yields system-level failure

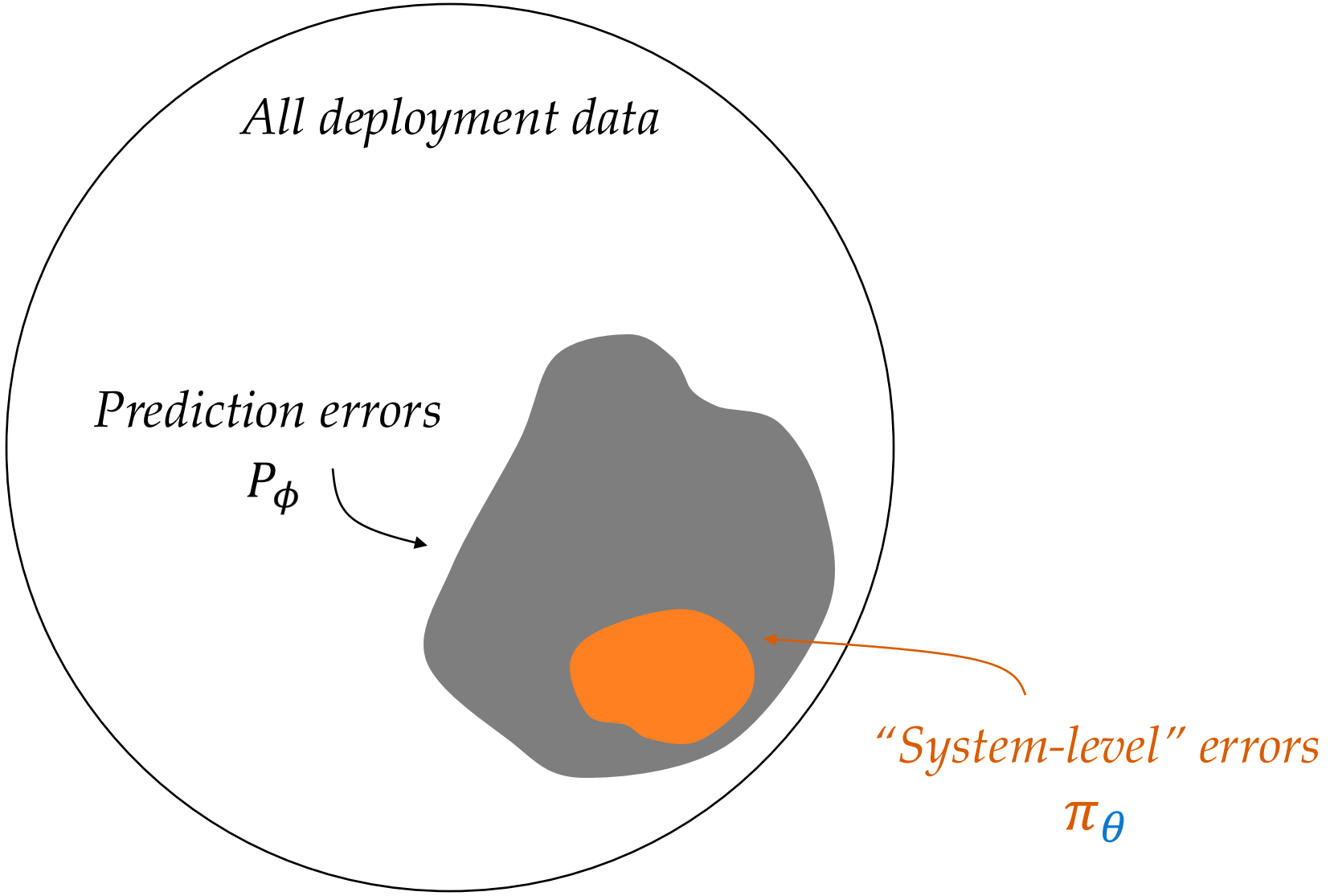


Mis-prediction does NOT yield system-level failure

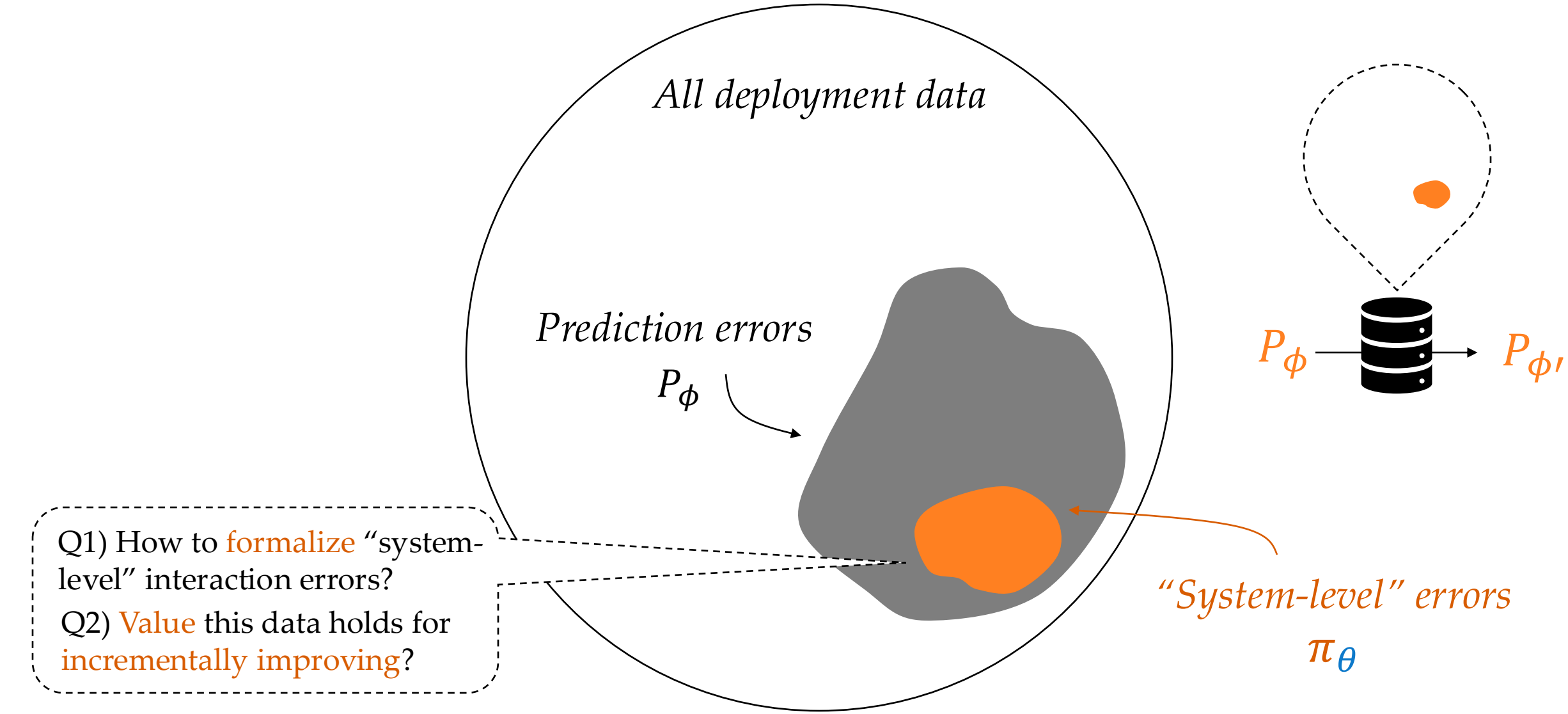


Both mis-predictions have the same L2 prediction error!

Farid, Alec, et al. "Task-relevant failure detection for trajectory predictors in autonomous vehicles." *CORL* 2023.



Sinha, Rohan, et al. "A System-Level View on Out-of Distribution Data in Robotics." *arXiv preprint* (2022).


All deployment data
Prediction errors
P_ϕ
P_ϕ
$P_{\phi'}$
"System-level" errors
π_θ
Q1) How to formalize "system-level" interaction errors?
Q2) Value this data holds for incrementally improving?

Idea 1

Regret precisely characterizes the degree to which prediction errors P_ϕ *degraded* robot performance π_θ

Nakamura, Tian, Bajcsy. "Not All Errors Are Made Equal: A Regret Metric for Detecting System-level Trajectory Prediction Failures". CoRL 2024.

Regret

robot's reward w/ parameters θ

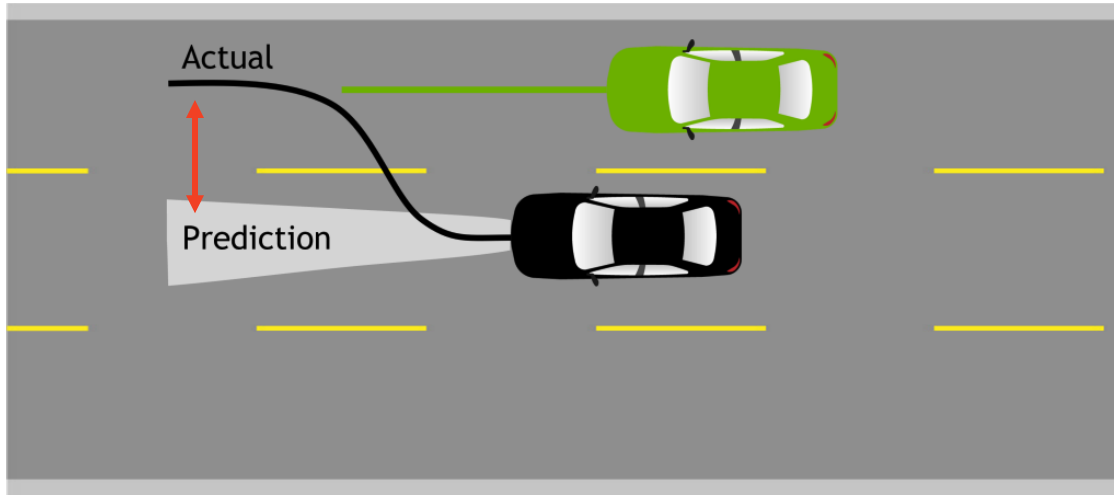
$$\max_{\boldsymbol{a}^{\mathrm{R}}} \left[R_{\theta}^{\mathrm{R}}\left(\boldsymbol{a}^{\mathrm{R}}, \hat{\boldsymbol{a}}^{\mathrm{H}_1:\mathrm{H}_M}, \hat{\boldsymbol{s}}, C\right) \right] - R_{\theta}^{\mathrm{R}}\left(\hat{\boldsymbol{a}}^{\mathrm{R}}, \hat{\boldsymbol{a}}^{\mathrm{H}_1:\mathrm{H}_M}, \hat{\boldsymbol{s}}, C\right)$$

best robot decision in hindsight

executed robot decision

observed human behavior, joint state, scene context

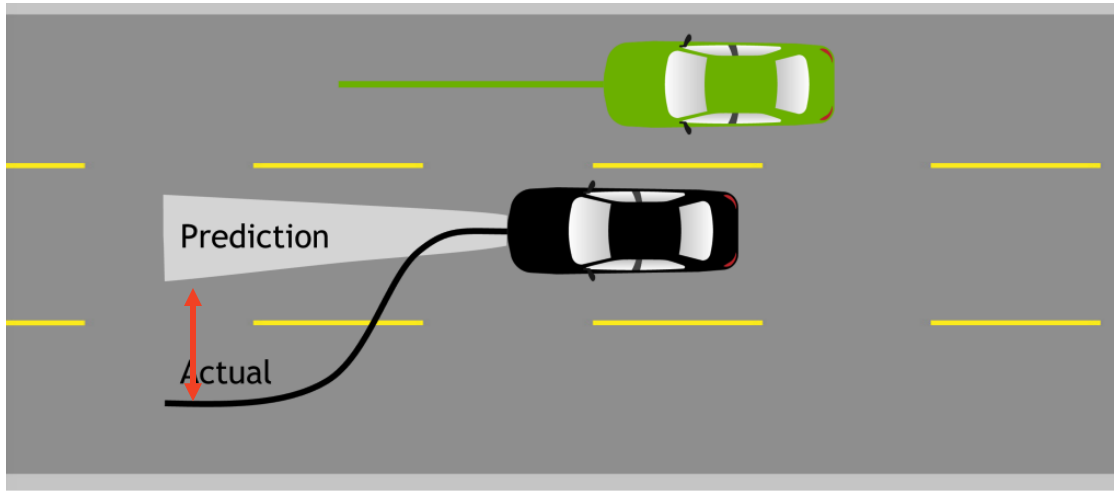
This mis-prediction would cause high regret!



$$\max_{a^R}\left[R_\theta^R(a^R, \text{⌐}\!\text{⌙})\right] - R_\theta^R(\text{—}, \text{⌐}\!\text{⌙})$$

In hindsight, should have slowed down! => ***High*** *regret*

This would not – robot would *still* do the same thing.



$$\max_{a^R}\left[R_\theta^R(a^R, \text{⌐}\!\text{⌙})\right] - R_\theta^R(\text{—}, \text{⌐}\!\text{⌙})$$

In hindsight, would have still done — => ***Low*** *regret*