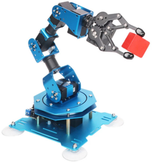
Task-Driven Detection of Distribution Shifts with Statistical Guarantees for Robot Learning

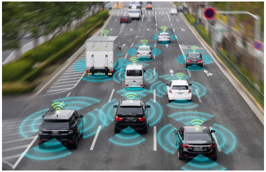
Authors: A. Farid, S. Veer, D. Pachisia, A. Majumdar

Charis Stamouli

April 19th, 2022

Motivation





Result overview

Objectives:

- OOD detection with guaranteed confidence bounds

Result overview

Objectives:

- OOD detection with guaranteed confidence bounds
- Task-driven approach (sensitive only to changes that affect the robot's performance)

Result overview

Objectives:

- OOD detection with guaranteed confidence bounds
- Task-driven approach (sensitive only to changes that affect the robot's performance)

Tools:

- Probably Approximately Correct (PAC)-Bayes theory

Result overview

Objectives:

- OOD detection with guaranteed confidence bounds
- Task-driven approach (sensitive only to changes that affect the robot's performance)

Tools:

- Probably Approximately Correct (PAC)-Bayes theory
- Statistical techniques based on p-values and concentration inequalities

Contributions

- Train control policies with a guaranteed performance bound on the training distribution

Contributions

- Train control policies with a guaranteed performance bound on the training distribution
- Develop two different task-driven OOD detectors

Contributions

- Train control policies with a guaranteed performance bound on the training distribution
- Develop two different task-driven OOD detectors
- Perform OOD and WD detection with guaranteed confidence bounds

Contributions

- Train control policies with a guaranteed performance bound on the training distribution
- Develop two different task-driven OOD detectors
- Perform OOD and WD detection with guaranteed confidence bounds
  $\implies$ Statistical guarantees on both the false positive and false negative rates of the detectors

Contributions

- Train control policies with a guaranteed performance bound on the training distribution
- Develop two different task-driven OOD detectors
- Perform OOD and WD detection with guaranteed confidence bounds
  $\implies$ Statistical guarantees on both the false positive and false negative rates of the detectors
- Demonstrate the benefits of the approach in simulation and hardware

Problem formulation

- Robot dynamics:

$$s_{t+1} = f_E(s_t, a_t)$$

Problem formulation

- Robot dynamics:

$$s_{t+1} = f_E(s_t, a_t)$$

- Cost under policy π:

$$C_E(\pi)$$

Problem formulation

- Robot dynamics:

$$s_{t+1} = f_E(s_t, a_t)$$

- Cost under policy π:

$$C_E(\pi)$$

- Training set:

$$S = \{E_1, \ldots, E_m\} \sim \mathcal{D}^m$$

Problem formulation

- Robot dynamics:
$$s_{t+1} = f_E(s_t, a_t)$$
- Cost under policy π:
$$C_E(\pi)$$
- Training set:
$$S = \{E_1, \dots, E_m\} \sim \mathcal{D}^m$$
- Test set:
$$S' = \{E'_1, \dots, E'_n\} \sim \mathcal{D}'^n$$

Problem formulation

- Robot dynamics:
$$s_{t+1} = f_E(s_t, a_t)$$

- Cost under policy π:
$$C_E(\pi)$$

- Training set:
$$S = \{E_1, \ldots, E_m\} \sim \mathcal{D}^m$$

- Test set:
$$S' = \{E'_1, \ldots, E'_n\} \sim \mathcal{D}'^n$$

After deploying the robot in a few environments in S', can we detect if $\mathcal{D}'$ is different from $\mathcal{D}$?

Problem formulation

- OOD detection if:

$$C_{\mathcal{D}'}(\pi) := \mathbb{E}_{E' \sim \mathcal{D}'} C_{E'}(\pi) > C_{\mathcal{D}}(\pi) := \mathbb{E}_{E \sim \mathcal{D}} C_E(\pi)$$

Problem formulation

- OOD detection if:

$$C_{\mathcal{D}'}(\pi) := \mathbb{E}_{E' \sim \mathcal{D}'} C_{E'}(\pi) > C_{\mathcal{D}}(\pi) := \mathbb{E}_{E \sim \mathcal{D}} C_E(\pi)$$

- WD detection if:

$$C_{\mathcal{D}'}(\pi) \leq C_{\mathcal{D}}(\pi)$$

Problem formulation

- OOD detection if:

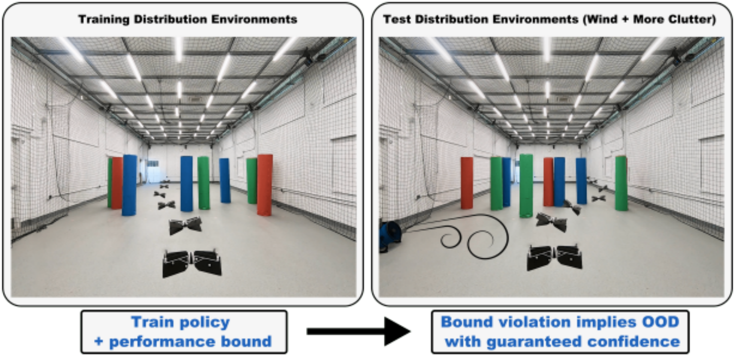
$$C_{\mathcal{D}'}(\pi) := \mathbb{E}_{E' \sim \mathcal{D}'} C_{E'}(\pi) > C_{\mathcal{D}}(\pi) := \mathbb{E}_{E \sim \mathcal{D}} C_E(\pi)$$

- WD detection if:

$$C_{\mathcal{D}'}(\pi) \leq C_{\mathcal{D}}(\pi)$$

- Detection: insensitive to changes in the environment distribution that do not adversely impact the robot's performance

Approach


Training Distribution Environments
Test Distribution Environments (Wind + More Clutter)
Train policy
+ performance bound
Bound violation implies OOD
with guaranteed confidence

Policy training via PAC-Bayes bounds

- Goal: Learn a policy π with a guaranteed bound on the expected cost $C_{\mathcal{D}}(\pi)$

Policy training via PAC-Bayes bounds

- Goal: Learn a policy π with a guaranteed bound on the expected cost $C_{\mathcal{D}}(\pi)$
- PAC-Bayes:
  - Before observing any data choose a **prior distribution** P_0 over the policy space Π

Policy training via PAC-Bayes bounds

- Goal: Learn a policy π with a guaranteed bound on the expected cost $C_{\mathcal{D}}(\pi)$
- PAC-Bayes:
  - Before observing any data choose a **prior distribution** P_0 over the policy space Π
  - Obtain a training dataset S and choose a **posterior distribution** P over the policy space Π

Policy training via PAC-Bayes bounds

- Goal: Learn a policy π with a guaranteed bound on the expected cost $C_{\mathcal{D}}(\pi)$
- PAC-Bayes:
  - Before observing any data choose a **prior distribution** P_0 over the policy space Π
  - Obtain a training dataset S and choose a **posterior distribution** P over the policy space Π
- Training cost:

$$C_S(\pi) = \frac{1}{m} \sum_{E \in S} C_E(\pi)$$

Policy training via PAC-Bayes bounds

Theorem 1. *For any distribution* $\mathcal{D}$*, prior distribution* P_0*,* $\delta \in (0,1)$*, cost bounded in* $[0,1]$*,* $m \geq 8$*, and deterministic algorithm* A *which outputs the posterior distribution* P*, we have the following:*

$$\mathbb{P}_{(S,\pi)\sim(\mathcal{D}^m \times P)}\left[C_{\mathcal{D}}(\pi) \leq \bar{C}_\delta(\pi, S)\right] \geq 1-\delta,$$

where $\bar{C}_\delta(\pi, S) := C_S(\pi) + \sqrt{R}$*,* $R := \left(D_2(P||P_0) + \ln\left(\frac{2\sqrt{m}}{(\delta/2)^3}\right)\right)/(2m)$*,*

and $D_2(P||P_0) := \ln\left(\mathbb{E}_{\pi\sim P_0}\left[\left(\frac{P(\pi)}{P_0(\pi)}\right)^2\right]\right)$ *is the Rényi divergence.*

Policy training via PAC-Bayes bounds

Theorem 1. *For any distribution $\mathcal{D}$, prior distribution P_0, $\delta \in (0,1)$, cost bounded in $[0,1]$, $m \geq 8$, and deterministic algorithm A which outputs the posterior distribution P, we have the following:*

$$\mathbb{P}_{(S,\pi)\sim(\mathcal{D}^m \times P)} \left[C_{\mathcal{D}}(\pi) \leq \bar{C}_\delta(\pi, S) \right] \geq 1 - \delta,$$

where $\bar{C}_\delta(\pi, S) := C_S(\pi) + \sqrt{R}$, $R := \left(D_2(P||P_0) + \ln\left(\frac{2\sqrt{m}}{(\delta/2)^3}\right)\right)/(2m)$,
and $D_2(P||P_0) := \ln\left(\mathbb{E}_{\pi\sim P_0}\left[\left(\frac{P(\pi)}{P_0(\pi)}\right)^2\right]\right)$ *is the Rényi divergence.*

Corollary 1. *Let the assumptions of Theorem 1 hold. Then:*

$$\mathbb{P}_{(S,\pi)\sim(\mathcal{D}^m \times P)} \left[C_{\mathcal{D}}(\pi) \geq \underline{C}_\delta(\pi, S) \right] \geq 1 - \delta,$$

where $\underline{C}_\delta(\pi, S) := C_S(\pi) - \sqrt{R}$.

Policy training via PAC-Bayes bounds

- PAC-Bayes bounds: Strong bounds for policy learning

Policy training via PAC-Bayes bounds

- PAC-Bayes bounds: Strong bounds for policy learning
- Policy training: Search for a posterior P to minimize the upper bound $\bar{C}_\delta(\pi, S)$

Policy training via PAC-Bayes bounds

- PAC-Bayes bounds: Strong bounds for policy learning
- Policy training: Search for a posterior P to minimize the upper bound $\bar{C}_\delta(\pi, S)$
- Training methods:
  - Backpropagation
  - Blackbox optimization (evolutionary strategies)

Task-driven OOD detection with statistical guarantees

- Key idea:

$$\text{Violation of the upper bound } \bar{C}_\delta(\pi, S)$$
$$\Downarrow$$
$$\text{Test environment is OOD}$$

Task-driven OOD detection with statistical guarantees

- Key idea:

$$\text{Violation of the upper bound } \bar{C}_\delta(\pi, S)$$
$$\Downarrow$$
$$\text{Test environment is OOD}$$

- Methods:
  1. Hypothesis testing via p-value

Task-driven OOD detection with statistical guarantees

- Key idea:

$$\text{Violation of the upper bound } \bar{C}_\delta(\pi, S)$$
$$\Downarrow$$
$$\text{Test environment is OOD}$$

- Methods:
  1. Hypothesis testing via p-value
  2. Confidence interval on the difference in expected train and test costs

Method 1: Hypothesis testing

- Goal: Detect if the test dataset is **OOD**, **WD** or **Unknown**

Method 1: Hypothesis testing

- Goal: Detect if the test dataset is **OOD**, **WD** or **Unknown**
- Define p-values for:
  - OOD detection:
    $p_O(S') := \mathbb{P}_{\hat{S}' \sim \mathcal{D}'^n}\left[C_{\hat{S}'}(\pi) \geq C_{S'}(\pi) | C_{\mathcal{D}'}(\pi) \leq C_{\mathcal{D}}(\pi)\right]$

Method 1: Hypothesis testing

- Goal: Detect if the test dataset is **OOD**, **WD** or **Unknown**
- Define p-values for:
  - OOD detection: $p_O(S') := \mathbb{P}_{\hat{S}' \sim \mathcal{D}'^n} \left[C_{\hat{S}'}(\pi) \geq C_{S'}(\pi) | C_{\mathcal{D}'}(\pi) \leq C_{\mathcal{D}}(\pi) \right]$
  - WD detection: $p_W(S') := \mathbb{P}_{\hat{S}' \sim \mathcal{D}'^n} \left[C_{\hat{S}'}(\pi) \leq C_{S'}(\pi) | C_{\mathcal{D}'}(\pi) > C_{\mathcal{D}}(\pi) \right]$

Method 1: Hypothesis testing

- Goal: Detect if the test dataset is **OOD**, **WD** or **Unknown**
- Define p-values for:
  - OOD detection: $p_O(S') := \mathbb{P}_{\hat{S}' \sim \mathcal{D}'^n} \left[C_{\hat{S}'}(\pi) \geq C_{S'}(\pi) | C_{\mathcal{D}'}(\pi) \leq C_{\mathcal{D}}(\pi) \right]$
  - WD detection: $p_W(S') := \mathbb{P}_{\hat{S}' \sim \mathcal{D}'^n} \left[C_{\hat{S}'}(\pi) \leq C_{S'}(\pi) | C_{\mathcal{D}'}(\pi) > C_{\mathcal{D}}(\pi) \right]$
- Perform two hypothesis tests:
  - H_0: WD and H_1: OOD

Method 1: Hypothesis testing

- Goal: Detect if the test dataset is **OOD**, **WD** or **Unknown**
- Define p-values for:
  - OOD detection: $p_O(S') := \mathbb{P}_{\hat{S}' \sim \mathcal{D}'^n}\left[C_{\hat{S}'}(\pi) \geq C_{S'}(\pi) | C_{\mathcal{D}'}(\pi) \leq C_{\mathcal{D}}(\pi)\right]$
  - WD detection: $p_W(S') := \mathbb{P}_{\hat{S}' \sim \mathcal{D}'^n}\left[C_{\hat{S}'}(\pi) \leq C_{S'}(\pi) | C_{\mathcal{D}'}(\pi) > C_{\mathcal{D}}(\pi)\right]$
- Perform two hypothesis tests:
  - H_0: WD and H_1: OOD
$$p_O(S') < \alpha_O \implies \mathcal{D}'\text{: OOD}$$

Method 1: Hypothesis testing

- Goal: Detect if the test dataset is **OOD**, **WD** or **Unknown**
- Define p-values for:
  - OOD detection:
    $p_O(S') := \mathbb{P}_{\hat{S}' \sim \mathcal{D}'^n}\left[C_{\hat{S}'}(\pi) \geq C_{S'}(\pi) | C_{\mathcal{D}'}(\pi) \leq C_{\mathcal{D}}(\pi)\right]$
  - WD detection:
    $p_W(S') := \mathbb{P}_{\hat{S}' \sim \mathcal{D}'^n}\left[C_{\hat{S}'}(\pi) \leq C_{S'}(\pi) | C_{\mathcal{D}'}(\pi) > C_{\mathcal{D}}(\pi)\right]$
- Perform two hypothesis tests:
  - H_0: WD and H_1: OOD
    $$p_O(S') < \alpha_O \implies \mathcal{D}'\text{: OOD}$$
  - H_0: OOD and H_1: WD
    $$p_W(S') < \alpha_W \implies \mathcal{D}'\text{: WD}$$

Method 1: Hypothesis testing

- Challenge:

$$\mathcal{D},\ \mathcal{D}' \text{ are unknown}$$
$$\Downarrow$$
$$\text{Cannot compute the p-values}$$

Method 1: Hypothesis testing

- Challenge:

$$\mathcal{D},\, \mathcal{D}' \text{ are unknown}$$
$$\Downarrow$$
$$\text{Cannot compute the p-values}$$

- Solution: Compute upper bounds for the p-values:

Method 1: Hypothesis testing

- Challenge:

$$\mathcal{D},\ \mathcal{D}' \text{ are unknown}$$
$$\Downarrow$$
$$\text{Cannot compute the p-values}$$

- Solution: Compute upper bounds for the p-values:

$$\mathbb{P}_{(S,\pi)\sim(\mathcal{D}^m\times P)}\left[p_O(S') \leq \exp(-2n\max\{C_{S'}(\pi) - \bar{C}_{\delta_O}(\pi,S),0\}^2\right] \geq 1-\delta_O$$
$$\mathbb{P}_{(S,\pi)\sim(\mathcal{D}^m\times P)}\left[p_W(S') \leq \exp(-2n\max\{\underline{\mathrm{C}}_{\delta_W}(\pi,S) - C_{S'}(\pi),0\}^2\right] \geq 1-\delta_W$$

Method 1: Hypothesis testing

- Challenge:

$$\mathcal{D},\ \mathcal{D}' \text{ are unknown}$$
$$\Downarrow$$
$$\text{Cannot compute the p-values}$$

- Solution: Compute upper bounds for the p-values:

$$\mathbb{P}_{(S,\pi)\sim(\mathcal{D}^m\times P)}\left[p_O(S') \leq \exp(-2n\max\{C_{S'}(\pi) - \bar{C}_{\delta_O}(\pi,S),0\}^2\right] \geq 1-\delta_O$$
$$\mathbb{P}_{(S,\pi)\sim(\mathcal{D}^m\times P)}\left[p_W(S') \leq \exp(-2n\max\{\underline{\mathrm{C}}_{\delta_W}(\pi,S) - C_{S'}(\pi),0\}^2\right] \geq 1-\delta_W$$

- Compare the upper bounds with the significance levels

Method 1: Hypothesis testing

Algorithm 1 OOD/WD Detection using Hypothesis Testing

Input: $\delta_O, \delta_W, \alpha_O, \alpha_W \in (0,1)$.
Input: PAC-Bayes Bounds: $\overline{C}_{\delta_O}(\pi, S)$, $\underline{C}_{\delta_W}(\pi, S)$.
Input: Test dataset $S' \sim \mathcal{D}'^n$ and policy $\pi \sim P$.
Output: OOD, WD, and Unknown
$C_{S'}(\pi) \leftarrow \frac{1}{n}\sum_{E \in S'} C_E(\pi)$
$\overline{\tau} \leftarrow \max\{C_{S'}(\pi) - \overline{C}_{\delta_O}(\pi, S), 0\}$
$\underline{\tau} \leftarrow \max\{\underline{C}_{\delta_W}(\pi, S) - C_{S'}(\pi), 0\}$
if $\exp(-2n\overline{\tau}^2) \leq \alpha_O$ **then**
    OOD $\leftarrow$ True
end if
if $\exp(-2n\underline{\tau}^2) \leq \alpha_W$ **then**
    WD $\leftarrow$ True
end if
if $\exp(-2n\overline{\tau}^2) > \alpha_W$ and $\exp(-2n\underline{\tau}^2) > \alpha_O$ **then**
    Unknown $\leftarrow$ True
end if

Method 1: Hypothesis testing

Algorithm 1 OOD/WD Detection using Hypothesis Testing

Input: $\delta_O, \delta_W, \alpha_O, \alpha_W \in (0,1)$.
Input: PAC-Bayes Bounds: $\overline{C}_{\delta_O}(\pi, S)$, $\underline{C}_{\delta_W}(\pi, S)$.
Input: Test dataset $S' \sim \mathcal{D}'^n$ and policy $\pi \sim P$.
Output: OOD, WD, and Unknown
$C_{S'}(\pi) \leftarrow \frac{1}{n}\sum_{E \in S'} C_E(\pi)$
$\overline{\tau} \leftarrow \max\{C_{S'}(\pi) - \overline{C}_{\delta_O}(\pi, S), 0\}$
$\underline{\tau} \leftarrow \max\{\underline{C}_{\delta_W}(\pi, S) - C_{S'}(\pi), 0\}$
if $\exp(-2n\overline{\tau}^2) \leq \alpha_O$ **then**
  OOD $\leftarrow$ True
end if
if $\exp(-2n\underline{\tau}^2) \leq \alpha_W$ **then**
  WD $\leftarrow$ True
end if
if $\exp(-2n\overline{\tau}^2) > \alpha_W$ and $\exp(-2n\underline{\tau}^2) > \alpha_O$ **then**
  Unknown $\leftarrow$ True
end if

- Consistency: Algorithm 1 returns **only one of the three possible outcomes** (OOD, WD or Unknown)

Method 2: Confidence interval overlap

- Goal: Detect if the test dataset is **OOD**, **WD** or **Unknown**

Method 2: Confidence interval overlap

- Goal: Detect if the test dataset is **OOD**, **WD** or **Unknown**

Theorem 3. *Let $\mathcal{D}$ be the training distribution, $\mathcal{D}'$ be the test distribution, and P be the posterior distribution on the policy space obtained through training. Let $\delta_O, \delta'_O \in (0,1)$ such that $\delta_O + \delta'_O < 1, \gamma_O := \sqrt{\frac{\ln(1/\delta'_O)}{2n}}$, and $\Delta C_O := C_{S'}(\pi) - \gamma_O - \bar{C}_{\delta_O}(\pi, S)$. Similarly, let $\delta_W, \delta'_W \in (0,1)$ such that $\delta_W + \delta'_W < 1, \gamma_W := \sqrt{\frac{\ln(1/\delta'_W)}{2n}}$, and $\Delta C_W := \underline{C}_{\delta_W}(\pi, S) - C_{S'}(\pi) - \gamma_W$. Then,*

$$\mathbb{P}_{(S,\pi,S')\sim(\mathcal{D}^m \times P \times \mathcal{D}'^n)}\left[C_{\mathcal{D}'}(\pi) - C_{\mathcal{D}}(\pi) \geq \Delta C_O\right] \geq 1 - \delta_O - \delta'_O,$$
$$\mathbb{P}_{(S,\pi,S')\sim(\mathcal{D}^m \times P \times \mathcal{D}'^n)}\left[C_{\mathcal{D}}(\pi) - C_{\mathcal{D}'}(\pi) \geq \Delta C_W\right] \geq 1 - \delta_W - \delta'_W.$$

Method 2: Confidence interval overlap

- ΔC_O: high-confidence lower bound for $C_{\mathcal{D}'}(\pi) - C_{\mathcal{D}}(\pi)$

Method 2: Confidence interval overlap

- ΔC_O: high-confidence lower bound for $C_{\mathcal{D}'}(\pi) - C_{\mathcal{D}}(\pi)$

$$\Delta C_O > 0 \implies \mathcal{D}' : \text{OOD}$$

Method 2: Confidence interval overlap

- ΔC_O: high-confidence lower bound for $C_{\mathcal{D}'}(\pi) - C_{\mathcal{D}}(\pi)$

$$\Delta C_O > 0 \implies \mathcal{D}' : \text{OOD}$$

- ΔC_W: high-confidence lower bound for $C_{\mathcal{D}}(\pi) - C_{\mathcal{D}'}(\pi)$

Method 2: Confidence interval overlap

- ΔC_O: high-confidence lower bound for $C_{\mathcal{D}'}(\pi) - C_{\mathcal{D}}(\pi)$

$$\Delta C_O > 0 \implies \mathcal{D}' : \text{OOD}$$

- ΔC_W: high-confidence lower bound for $C_{\mathcal{D}}(\pi) - C_{\mathcal{D}'}(\pi)$

$$\Delta C_W \geq 0 \implies \mathcal{D}' : \text{WD}$$

Method 2: Confidence interval overlap

- ΔC_O: high-confidence lower bound for $C_{\mathcal{D}'}(\pi) - C_{\mathcal{D}}(\pi)$

$$\Delta C_O > 0 \implies \mathcal{D}' : \text{OOD}$$

- ΔC_W: high-confidence lower bound for $C_{\mathcal{D}}(\pi) - C_{\mathcal{D}'}(\pi)$

$$\Delta C_W \geq 0 \implies \mathcal{D}' : \text{WD}$$

- $\Delta C_O \leq 0$ and $\Delta C_W < 0 \implies \mathcal{D}'$: Unknown

Method 2: Confidence interval overlap

Algorithm 2 OOD/WD Detection using Confidence Intervals

Input: $\delta_O, \delta'_O \in (0,1)$ with desired **maximum** false positive rate $\delta_O + \delta'_O < 1$.
Input: $\delta_W, \delta'_W \in (0,1)$ with desired **maximum** false negative rate, $\delta_W + \delta'_W < 1$.
Input: PAC-Bayes Bounds: $\overline{C}_{\delta_O}(\pi, S)$, $\underline{C}_{\delta_W}(\pi, S)$.
Input: Test dataset $S' \sim \mathcal{D}'^n$ and policy $\pi \sim P$.
Output: OOD, WD and Unknown.
$C_{S'}(\pi) \leftarrow \frac{1}{n}\sum_{E' \in S'} C_{E'}(\pi)$
$\gamma_O \leftarrow \sqrt{\frac{\ln(1/\delta'_O)}{2n}}$
$\gamma_W \leftarrow \sqrt{\frac{\ln(1/\delta'_W)}{2n}}$
$\Delta C_O \leftarrow C_{S'}(\pi) - \gamma_O - \overline{C}_{\delta_O}(\pi, S)$
$\Delta C_W \leftarrow \underline{C}_{\delta_W}(\pi, S) - C_{S'}(\pi) - \gamma_W$
if $\Delta C_O > 0$ **then**
  OOD $\leftarrow$ True
end if
if $\Delta C_W \geq 0$ **then**
  WD $\leftarrow$ True
end if
if $\Delta C_O \leq 0$ and $\Delta C_W < 0$ **then**
  Unknown $\leftarrow$ True
end if

Method 2: Confidence interval overlap

Algorithm 2 OOD/WD Detection using Confidence Intervals

Input: $\delta_O, \delta'_O \in (0,1)$ with desired **maximum** false positive rate $\delta_O + \delta'_O < 1$.
Input: $\delta_W, \delta'_W \in (0,1)$ with desired **maximum** false negative rate, $\delta_W + \delta'_W < 1$.
Input: PAC-Bayes Bounds: $\overline{C}_{\delta_O}(\pi, S)$, $\underline{C}_{\delta_W}(\pi, S)$.
Input: Test dataset $S' \sim \mathcal{D}'^n$ and policy $\pi \sim P$.
Output: OOD, WD and Unknown.
$C_{S'}(\pi) \leftarrow \frac{1}{n}\sum_{E' \in S'} C_{E'}(\pi)$
$\gamma_O \leftarrow \sqrt{\frac{\ln(1/\delta'_O)}{2n}}$
$\gamma_W \leftarrow \sqrt{\frac{\ln(1/\delta'_W)}{2n}}$
$\Delta C_O \leftarrow C_{S'}(\pi) - \gamma_O - \overline{C}_{\delta_O}(\pi, S)$
$\Delta C_W \leftarrow \underline{C}_{\delta_W}(\pi, S) - C_{S'}(\pi) - \gamma_W$
if $\Delta C_O > 0$ **then**
  OOD $\leftarrow$ True
end if
if $\Delta C_W \geq 0$ **then**
  WD $\leftarrow$ True
end if
if $\Delta C_O \leq 0$ and $\Delta C_W < 0$ **then**
  Unknown $\leftarrow$ True
end if

- Consistency: Algorithm 2 returns **only one of the three possible outcomes** (OOD, WD or Unknown)

Robotic grasping

- Franka Panda arm with an overhead camera

Robotic grasping

- Franka Panda arm with an overhead camera
- Train the manipulator to grasp mugs placed in poses drawn from a specific distribution

Robotic grasping

- Franka Panda arm with an overhead camera
- Train the manipulator to grasp mugs placed in poses drawn from a specific distribution
- Control policy: DNN which inputs a depth map of the object and a latent state

Robotic grasping

- Franka Panda arm with an overhead camera
- Train the manipulator to grasp mugs placed in poses drawn from a specific distribution
- Control policy: DNN which inputs a depth map of the object and a latent state
- Training:
  - Dataset: Mugs from ShapeNet randomly scaled in all dimensions

Robotic grasping

- Franka Panda arm with an overhead camera
- Train the manipulator to grasp mugs placed in poses drawn from a specific distribution
- Control policy: DNN which inputs a depth map of the object and a latent state
- Training:
  - Dataset: Mugs from ShapeNet randomly scaled in all dimensions
  - Cost: Successful rollout if the robot is able to lift the mug by 10cm and the gripper palm does not contact it

Robotic grasping

- Franka Panda arm with an overhead camera
- Train the manipulator to grasp mugs placed in poses drawn from a specific distribution
- Control policy: DNN which inputs a depth map of the object and a latent state
- Training:
  - Dataset: Mugs from ShapeNet randomly scaled in all dimensions
  - Cost: Successful rollout if the robot is able to lift the mug by 10cm and the gripper palm does not contact it
  - Process: Sample a policy π from the trained posterior

Robotic grasping: Simulation results

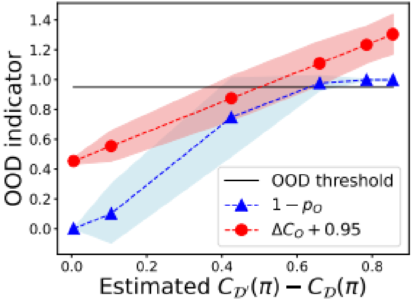
- Gradually make the distribution on the mug poses more challenging

Robotic grasping: Simulation results

- Gradually make the distribution on the mug poses more challenging
- For each distribution:
  - Sample 20 test datasets
  - Compute the OOD indicators

Robotic grasping: Simulation results

- Gradually make the distribution on the mug poses more challenging
- For each distribution:
  - Sample 20 test datasets
  - Compute the OOD indicators

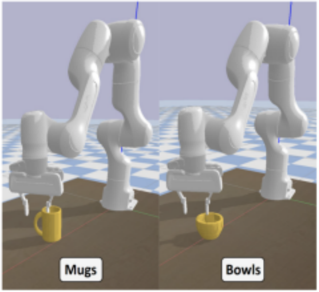


Robotic grasping: Simulation results

- Change the objects from mugs to bowls

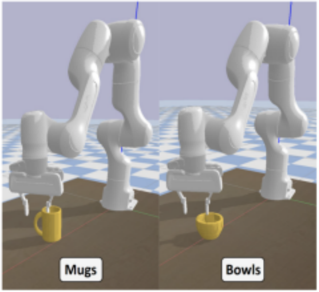
Robotic grasping: Simulation results

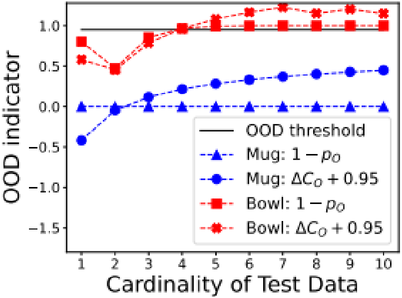
- Change the objects from mugs to bowls



Robotic grasping: Simulation results

- Change the objects from mugs to bowls





Robotic grasping: Hardware results

- Gradually increase the range of location of the mugs

Robotic grasping: Hardware results

- Gradually increase the range of location of the mugs

$$\Downarrow$$

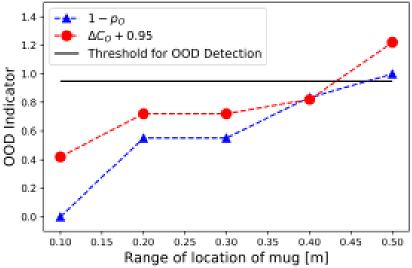
- The proportion of trials that the grasp fails increases

Robotic grasping: Hardware results

- Gradually increase the range of location of the mugs

$$\Downarrow$$

- The proportion of trials that the grasp fails increases

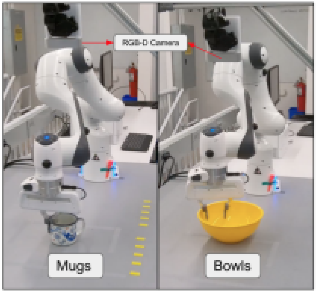


Robotic grasping: Hardware results

- Change the objects from mugs to bowls

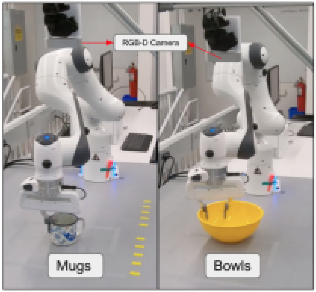
Robotic grasping: Hardware results

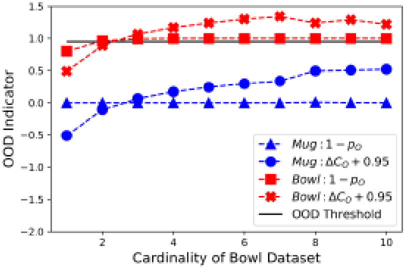
- Change the objects from mugs to bowls



Robotic grasping: Hardware results

- Change the objects from mugs to bowls





Obstacle avoidance with a drone

- Parrot Swing drone

Obstacle avoidance with a drone

- Parrot Swing drone
- Train the drone to avoid obstacles by the largest possible distance

Obstacle avoidance with a drone

- Parrot Swing drone
- Train the drone to avoid obstacles by the largest possible distance
- Control policy: DNN which inputs a depth image and outputs a set of motion primitives

Obstacle avoidance with a drone

- Parrot Swing drone
- Train the drone to avoid obstacles by the largest possible distance
- Control policy: DNN which inputs a depth image and outputs a set of motion primitives
- Training:
  - Dataset: Environments with randomly placed cylindrical obstacles

Obstacle avoidance with a drone

- Parrot Swing drone
- Train the drone to avoid obstacles by the largest possible distance
- Control policy: DNN which inputs a depth image and outputs a set of motion primitives
- Training:
  - Dataset: Environments with randomly placed cylindrical obstacles
  - Cost: $\max\left\{0, 1 - \frac{d_{\min}}{d_{thres}}\right\}$

Obstacle avoidance with a drone

- Parrot Swing drone
- Train the drone to avoid obstacles by the largest possible distance
- Control policy: DNN which inputs a depth image and outputs a set of motion primitives
- Training:
  - Dataset: Environments with randomly placed cylindrical obstacles
  - Cost: $\max\left\{0, 1 - \dfrac{d_{\min}}{d_{thres}}\right\}$
  - Process: Sample a policy π from the trained posterior

Obstacle avoidance with a drone: Simulation results

- Generate test datasets of varying difficulty (by changing the number of obstacles and the maximum/minimum gap-size between obstacles)

Obstacle avoidance with a drone: Simulation results

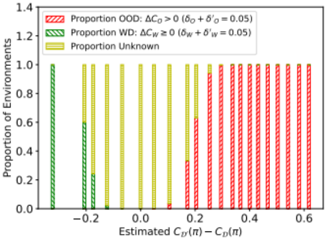
- Generate test datasets of varying difficulty (by changing the number of obstacles and the maximum/minimum gap-size between obstacles)
- For each difficulty setting:
  - Sample 2000 datasets

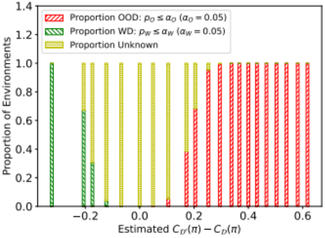
Obstacle avoidance with a drone: Simulation results

- Generate test datasets of varying difficulty (by changing the number of obstacles and the maximum/minimum gap-size between obstacles)
- For each difficulty setting:
  - Sample 2000 datasets
  - Evaluate the proportion of datasets that the detectors declare OOD, WD or Unknown

Obstacle avoidance with a drone: Simulation results

- Generate test datasets of varying difficulty (by changing the number of obstacles and the maximum/minimum gap-size between obstacles)
- For each difficulty setting:
  - Sample 2000 datasets
  - Evaluate the proportion of datasets that the detectors declare OOD, WD or Unknown



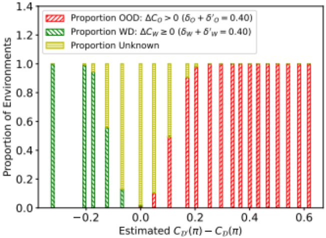


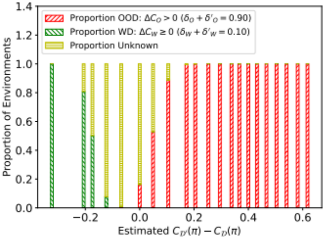
Obstacle avoidance with a drone: Simulation results

- Change the desired maximum false positive and false negative rates

Obstacle avoidance with a drone: Simulation results

- Change the desired maximum false positive and false negative rates





Obstacle avoidance with a drone: Simulation results

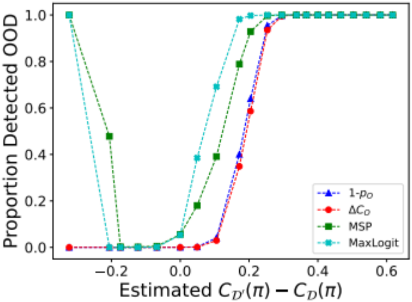
- Compare the detectors with two baselines:

Obstacle avoidance with a drone: Simulation results

- Compare the detectors with two baselines:
  - Maximum softmax probability (MSP)
  - MaxLogit

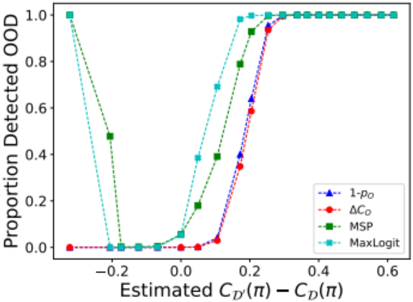
Obstacle avoidance with a drone: Simulation results

- Compare the detectors with two baselines:
  - Maximum softmax probability (MSP)
  - MaxLogit



Obstacle avoidance with a drone: Simulation results

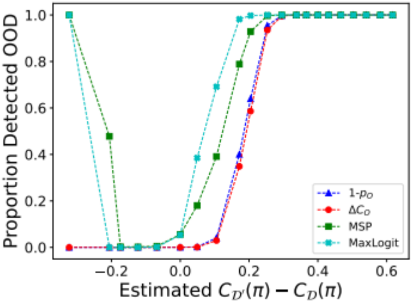
- Compare the detectors with two baselines:
  - Maximum softmax probability (MSP)
  - MaxLogit



- Disadvantages of both baselines:

Obstacle avoidance with a drone: Simulation results

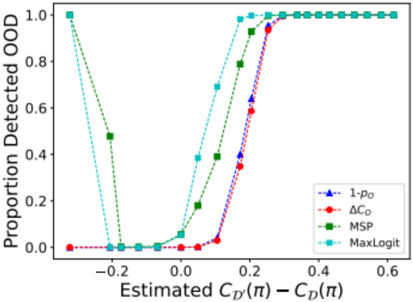
- Compare the detectors with two baselines:
  - Maximum softmax probability (MSP)
  - MaxLogit



- Disadvantages of both baselines:
  - Violation of the maximum false positive rate

Obstacle avoidance with a drone: Simulation results

- Compare the detectors with two baselines:
  - Maximum softmax probability (MSP)
  - MaxLogit



- Disadvantages of both baselines:
  - Violation of the maximum false positive rate
  - Triggered by task-irrelevant shifts

Obstacle avoidance with a drone: Hardware results

Deploy the policy trained in simulation on three kinds of environments:

Obstacle avoidance with a drone: Hardware results

Deploy the policy trained in simulation on three kinds of environments:

- Environments with a smaller number of obstacles (easier environments)

Obstacle avoidance with a drone: Hardware results

Deploy the policy trained in simulation on three kinds of environments:

- Environments with a smaller number of obstacles (easier environments)
- Environments with smaller gaps between obstacles (harder environments)

Obstacle avoidance with a drone: Hardware results

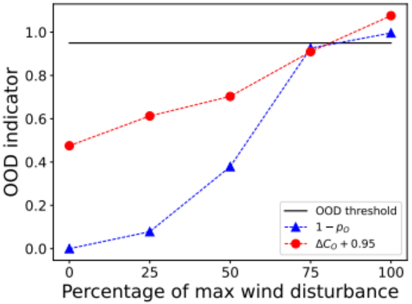
Deploy the policy trained in simulation on three kinds of environments:

- Environments with a smaller number of obstacles (easier environments)
- Environments with smaller gaps between obstacles (harder environments)
- Environments with wind generated using a fan

Obstacle avoidance with a drone: Hardware results

Deploy the policy trained in simulation on three kinds of environments:

- Environments with a smaller number of obstacles (easier environments)
- Environments with smaller gaps between obstacles (harder environments)
- Environments with wind generated using a fan



Experiments

- Videos of the experiments can be found here: https://www.youtube.com/watch?v=jKye3A09le0

Summary

Task-driven OOD and WD detection with statistical guarantees:

Summary

Task-driven OOD and WD detection with statistical guarantees:

- Use PAC-Bayes theory to train a policy with a bound on the expected cost on the training distribution

Summary

Task-driven OOD and WD detection with statistical guarantees:

- Use PAC-Bayes theory to train a policy with a bound on the expected cost on the training distribution
- Perform OOD and WD detection on test environments by checking for violations of the bound using approaches based on both p-values and confidence intervals

Summary

Task-driven OOD and WD detection with statistical guarantees:

- Use PAC-Bayes theory to train a policy with a bound on the expected cost on the training distribution
- Perform OOD and WD detection on test environments by checking for violations of the bound using approaches based on both p-values and confidence intervals
- Provide guarantees on the maximum false positive and false negative rate

Summary

Task-driven OOD and WD detection with statistical guarantees:

- Use PAC-Bayes theory to train a policy with a bound on the expected cost on the training distribution
- Perform OOD and WD detection on test environments by checking for violations of the bound using approaches based on both p-values and confidence intervals
- Provide guarantees on the maximum false positive and false negative rate
- Demonstrate the ability of the approaches to perform OOD detection within a handful of trials

Summary

Task-driven OOD and WD detection with statistical guarantees:

- Use PAC-Bayes theory to train a policy with a bound on the expected cost on the training distribution
- Perform OOD and WD detection on test environments by checking for violations of the bound using approaches based on both p-values and confidence intervals
- Provide guarantees on the maximum false positive and false negative rate
- Demonstrate the ability of the approaches to perform OOD detection within a handful of trials
- Tune the detectors' sensitivity by varying the maximum permissible false positive/negative rate