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

Alec Farid †, Sushant Veer †, Divyanshu Pachisia ‡, and Anirudha Majumdar †

Abstract—Our goal is to perform out-of-distribution (OOD) detection, i.e., to detect when a robot is operating in environments drawn from a different distribution than the ones used to train the robot. We leverage probably approximately correct-Bayes theory to train a policy with a guaranteed bound on performance on the training distribution. Our idea for OOD detection 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 approach provides guaranteed confidence bounds on OOD detection including bounds on both the false-positive and false-negative rates of the detector and is task-driven and only sensitive to changes that impact the robot's performance. We demonstrate our approach in simulation and hardware for a grasping task using objects with unfamiliar shapes or poses and a drone performing vision-based obstacle avoidance in environments with wind disturbances and varied obstacle densities. Our examples demonstrate that we can perform task-driven OOD detection within just a handful of trials.

Index Terms—Deep learning in robotics and automation, failure detection and recovery, formal methods in robotics and automation, PAC-Bayes.

I. INTRODUCTION

IMAGINE a drone trained to perform vision-based navigation using a dataset of indoor environments and deployed in environments with varying wind conditions or obstacle densities (see 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 particularly pressing in safety-critical settings, where the price of failure is high. In this work, we focus on the problem of autonomously detecting when a robot is operating in environments drawn from a different distribution than the one used to train the robot. This ability to perform OOD detection has the potential to improve the safety of robotic systems operating in OOD environments. For example, a drone operating in a new set of environments could either deploy a highly conservative policy or cease its operations altogether. In addition, OOD detection can also allow the robot to improve its policy by retraining using additional data collected from the new environments.

There are two important desiderata that OOD detection approaches for safety-critical robotic systems should ideally satisfy. First, we would like to develop OOD detection techniques with guaranteed confidence bounds. Second, we would like our OOD detectors to be task-driven and only sensitive to task-relevant changes in the robot’s environment. As an example, consider again the drone navigation setting in Fig. 1 and suppose that the robot’s policy is insensitive to changes in color and lighting. Here, the robot’s OOD detector should not trigger even if the robot is operating in environments with different color/lighting and should only trigger if there are task-relevant variations (e.g., variations in the obstacle density). Unfortunately, current approaches (see Section II) do not typically satisfy both desiderata; they are often based on heuristics and are not task-driven in general.

Statement of contributions: We develop task-driven OOD detection techniques with statistical guarantees on correctness. To this end, we make four specific contributions (see Fig. 1 for an overview).

1) Given a dataset of environments drawn from an (unknown) training distribution, we develop a pipeline based on generalization theory for training control policies with a guaranteed bound on performance (a bound on the expected cost of the policy on the unknown training distribution). Specifically, we leverage recently developed derandomized probably approximately correct (PAC)-Bayes bounds that are well suited to enable OOD detection (see Section IV-A).

2) We develop two OOD detection techniques (see Section IV-B), using $p$-values and concentration inequalities, with differing statistical interpretations. Both detectors are based on the following intuition: if the costs incurred when the robot is deployed in a small number of new environments violate the bound on the policy’s performance, this indicates that the robot is operating OOD. Since our OOD detection scheme leverages the costs incurred in new
In particular, we identify two distinct OOD events: OOD-adverse (OOD_A) and OOD-benign (OOD_B), which correspond to OOD events that result in costs that are higher than the PAC-Bayes generalization bound and lower than the PAC-Bayes generalization bound, respectively. Both OOD_A and OOD_B are OOD events, but the former is detrimental to the robot (requiring an intervention) while the latter is not.

3) Our detection schemes have the ability to perform OOD detection with guaranteed confidence bounds. This allows us to provide statistical guarantees on both the false-positive rate (FPR) (probability that OOD_A is incorrectly detected) and the false-negative rate (FNR) (probability that OOD_B is incorrectly detected) for our detectors; positive detection is one that requires intervention to the robot’s nominal operation whereas a negative detection is one that does not. Developing an effective task-relevant OOD detector hinges on optimizing both the FPR and FNR. A low FNR promotes safety by detecting OOD environments correctly whereas a low FPR promotes performance by ensuring that we do not detect too many within-distribution (WD) environments as OOD, resulting in unnecessary interventions to the robot’s operations. In our article, the bound on OOD_A lets us tune the thresholds for FPR while the bound on OOD_B lets us tune the thresholds for FNR.

4) We demonstrate our approach on two simulated examples (see Section V): 1) a robotic manipulator grasping a new set of objects in varying locations, and 2) a drone navigating a new set of environments. Comparisons with baselines demonstrate the advantages of our approach in terms of providing statistical guarantees and being insensitive to task-irrelevant shifts. We also present a thorough set of hardware experiments for vision-based drone navigation with varying wind conditions and clutter (see Fig. 1) and for grasping with varying objects and poses. Our experiments demonstrate the ability of our approach to perform task-driven OOD detection within just a handful of trials for systems with complex dynamics and rich sensing modalities.

A preliminary version of this work was presented in the Conference of Robot Learning 2021 [1]. In this significantly extended and revised version, we additionally present the following:

1) an extension of our OOD detection methods to also detect OOD-benign (OOD_B) environments (see Section IV-B);
2) formulations of our detectors in terms of algorithms that output the detectors’ predictions (Algorithms 1 and 2);
3) bounds on the FNR on the confidence-interval based OOD detector (Remark 1);
4) hardware results on the OOD detector for the grasping example previously studied in simulation (see Section V-A);
5) expanded simulation results and a study of the effect of the cost function and chosen confidence bounds for the navigation example (see Section V-D).

II. RELATED WORK

A. Anomaly/OOD Detection in Supervised Learning

Anomaly detection in low-dimensional signals has been well-studied in the signal processing literature (see [2] for a review).
Recent work in machine learning has focused on OOD detection for high-dimensional inputs (e.g., images) in supervised learning settings (see [3] for a review). Popular approaches use threshold-based detectors for the output distribution of a given pretrained neural classifier [4], [5], [6]. Other methods use a specific training pipeline in order to improve OOD detection on test samples [7], [8], [9], [10]. However, these methods are often susceptible to adversarial attacks [11]. Thus, approaches for addressing adversarial data have been developed [11], [12], [13]. Some of these approaches are also able to provide theoretical guarantees of performance on adversarial data [11], [14], [15]. Other methods provide PAC-style statistical guarantees [16], [17], [18] or p-values [19]. However, these methods typically focus on supervised learning settings and often require specific network outputs (e.g., softmax) that are incompatible with non-classification tasks. In contrast, we focus on OOD detection for policy learning settings in robotics and do not make assumptions about the specific structure of the policy.

B. Task-Driven OOD Detection

The methods above are aimed at detecting any distributional shift in the data and can be sensitive even to task-irrelevant shifts (i.e., ones that do not impact performance) as we demonstrate in our experiments (see Section V). A recent method determines an estimate of input typicality for pretrained networks and uses it as an OOD detector in supervised learning settings [20]. Another approach performs novelty detection on images from a vision-based robot for collision avoidance [21]. Recent methods have also been developed specifically for reinforcement learning (RL) [22], [23], [24], [25], [26]. In particular, Greenberg and Mannor [26] presented a general task-driven approach for OOD detection on sequential rewards, which is optimal in certain settings. However, neither this method nor others in the RL context provide statistical guarantees on detection. We propose an OOD detection framework, which is both task-driven and provides statistical guarantees by leveraging generalization theory.

C. Generalization Theory

Generalization theory provides a way to learn hypotheses (in supervised learning) with a bound on the true expected loss on the underlying data-generating distribution given only a finite number of training examples. Original frameworks include Vapnik–Chervonenkis (VC) theory [27] and Rademacher complexity [28]. However, these methods often provide vacuous generalization bounds for high-dimensional hypothesis spaces (e.g., neural networks). Bounds based on PAC-Bayes generalization theory [29], [30], [31] have recently been shown to provide strong guarantees in a variety of settings [32], [33], [34], [35], [36], [37], [38], and have been significantly extended and improved [39], [40], [41], [42], [43], [44]. PAC-Bayes has also recently been extended to learn policies for robots with guarantees on generalization to novel environments [45], [46], [47], [48]. In this work, we leverage recently-proposed derandomized PAC-Bayes bounds [49]; this framework allows us to train a single deterministic policy with a guaranteed bound on expected performance on the training distribution (in contrast to [45], [46], [47], [48], which train stochastic neural network policies). This forms the basis for our OOD detection framework: by observing violations of the PAC-Bayes bound on test environments, we are able to perform task-driven OOD detection with statistical guarantees.

III. PROBLEM FORMULATION

A. Dynamics and Environments

Let \( s_{t+1} = f_E(s_t, o_t) \) describe the robot’s dynamics, where \( s_t \in \mathcal{S} \subseteq \mathbb{R}^{n_s} \) is the state of the robot at time-step \( t \), \( o_t \in \mathcal{A} \subseteq \mathbb{R}^{n_o} \) is the action, and \( E \in \mathcal{E} \) is the environment that the robot is operating in. “Environment” here broadly refers to factors that are external to the robot, e.g., a cluttered room that a drone is navigating, disturbances such as wind gusts, or an object that a manipulator is grasping. The dynamics of the robot may be nonlinear/hybrid. We denote the robot’s sensor observations (e.g., RGB-D images) by \( o_t \in \mathcal{O} \subseteq \mathbb{R}^{n_o} \).

B. Cost Functions

The robot’s task is encoded via a cost function and we let \( C_E(\pi) \) denote the cost incurred by a (deterministic) policy \( \pi \) when deployed in environment \( E \) over a finite-time horizon \( T \). The policy \( \pi \in \Pi \) is a mapping from (histories of) sensor observations to actions (e.g., parameterized using a neural network). In the context of obstacle avoidance, the cost could capture how close the drone gets to an obstacle; in the context of grasping, the cost could be 0 if the robot successfully lifts the object and 1 otherwise. We assume that the cost is bounded; without further loss of generality, we assume \( C_E(\pi) \in [0, 1] \). We also assume that the robot has access to the cost \( C_E(\pi) \) after performing a rollout on \( E \) (i.e., at the end of an episode of length \( T \)). This is a relatively benign assumption in robotics contexts since the cost often has physical meaning and can be measured by the robot’s sensors. For example, a drone equipped with a depth sensor can measure the smallest reported depth value during its operation in an environment, and a manipulator equipped with a camera or force-torque sensor can measure if it successfully grasped an object. We make no further assumptions on the cost function (e.g., we do not assume continuity or Lipschitzness).

C. Training and Testing Distribution

We assume that the robot has access to a training dataset \( S = \{E_1, \ldots, E_m\} \) of \( m \) environments drawn independently and identically distributed (i.i.d.) from a training distribution \( \mathcal{D} \), i.e., \( S \sim \mathcal{D}^m \). After training, the robot is deployed on environments in \( S' = \{E_1, \ldots, E_{m'}\} \) drawn from a test distribution \( \mathcal{D}' \); \( S' \sim \mathcal{D}'^m \). Importantly, we do not assume any explicit knowledge of \( \mathcal{D}, \mathcal{D}' \), or the space \( \mathcal{E} \) of environments. We only have indirect access to \( \mathcal{D} \) and \( \mathcal{D}' \) in the form of the finite training datasets \( S \) and \( S' \).

D. Goal: Task-Driven OOD Detection With Statistical Guarantees

After being deployed in (a typically small number of) environments in \( S' \), the robot’s goal is to detect if these environments
were drawn from a different distribution than the training distribution (i.e., if \( \mathcal{D}' \) is different from \( \mathcal{D} \)). Moreover, our goal is to perform task-driven OOD detection. In particular, we consider environments drawn from \( \mathcal{D}' \) as OOD-adverse if \( \mathcal{D}' \) satisfies the following:

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

and OOD-benign if

\[
C_\mathcal{D}'(\pi) \leq C_\mathcal{D}(\pi).
\]

Thus, our OOD-adversary detector should be insensitive to changes in the environment distribution that do not adversely impact the robot’s performance. This is a challenging task since we only assume access to a finite number of environments from \( \mathcal{D} \) and \( \mathcal{D}' \). Moreover, our goal is to develop an OOD detection framework that is broadly applicable in challenging settings involving nonlinear/hybrid dynamics, rich sensing modalities (e.g., RGB-D), and neural network-based policies.

**IV. APPROACH**

Our overall approach is illustrated in Fig. 1. First, we train a policy with an associated guarantee on the expected cost on the training distribution \( \mathcal{D} \) (see Section IV-A). We then apply our OOD detection scheme, which formalizes the following intuition: violation of the bound during deployment implies (with high confidence) that the test distribution \( \mathcal{D}' \) is OOD in a task-relevant manner (see Section IV-B).

**A. Policy Training via Derandomized PAC-Bayes Bounds**

Given a training dataset \( S = \{E_1, \ldots, E_m\} \) of \( m \) environments drawn i.i.d. from the training distribution \( \mathcal{D} \), our goal is to learn a policy \( \pi \) with a guaranteed bound on the expected cost \( C_\mathcal{D}(\pi) := \mathbb{E}_{E \sim \mathcal{D}} C_E(\pi) \). Since our OOD detection scheme will rely on violations of the bound, it is important to obtain bounds that are as tight as possible. In this work, we utilize the probably approximately correct (PAC)-Bayes framework [29], [30], [31] to train policies with strong guarantees. More specifically, we leverage recently developed derandomized PAC-Bayes bounds [49], which are well-suited to the OOD detection setting (as we explain further below).

PAC-Bayes applies to settings where one chooses a distribution over policies (e.g., a distribution over weights of a neural network), and learning algorithms that have the following structure: 1) choose a “prior” \( P_0 \) over the policy space \( \Pi \) before observing any data (this can be used to encode domain/expert knowledge); 2) obtain a training dataset \( S \) and choose a posterior distribution \( P \) over the policy space \( \Pi \). Let \( P \) be the output of an algorithm \( A \), which takes \( P_0 \) and \( S \) as input. Denote the cost incurred by a policy \( \pi \) on the training environments in \( S \) as \( C_S(\pi) := \frac{1}{m} \sum_{E \in S} C_E(\pi) \). The following result is our primary theoretical tool for training policies with bounds on performance.

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

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

where \( C_{\mathcal{D}}(\pi, S) := C_S(\pi) + \sqrt{R} \), \( R := (D_2(P || P_0) + ln \frac{2m}{\delta})^2/(2m) \), and \( D_2 \) is the Rényi Divergence for \( \alpha = 2 \) defined as:

\[
D_2(P || P_0) = \ln (\mathbb{E}_{P \sim P_0}[P(\pi)^2]).
\]

**Proof:** The proof is in Appendix A. We use [49, Th. 2], a general pointwise PAC-Bayes bound. We perform the reduction from supervised learning to policy learning presented in [45].

We can provide a lower bound on \( C_\mathcal{D} \) as an immediate corollary of the abovementioned theorem.

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

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

completing the proof.

These results allow us to obtain policies with guaranteed upper and lower bounds on the expected cost. In particular, we can search for a posterior \( P \) in order to minimize the upper bound \( C_{\mathcal{D}}(\pi, S) \), i.e., in order to minimize the sum of the training cost and the “regularizer” \( \sqrt{R} \). We describe such training methods via backpropagation and closed box optimization in Appendix F and G, respectively. Sampling from the resulting posterior \( P \) provides a policy with a bound on \( C_\mathcal{D}(\pi) \) that holds with high probability (over the sampling of the training dataset \( S \) and the policy \( \pi \)).

Recent work has demonstrated the effectiveness of PAC-Bayes to provide strong bounds for deep neural networks (DNNs) [32], [35], [37] and specifically for policy learning [45], [46], [47], [48]. However, the bounds used by these approaches do not provide a viable approach for performing OOD detection. The approaches are based on traditional PAC-Bayes bounds, where a distribution \( P \) over policies (e.g., a distribution over neural network weights) is chosen; the resulting bound is on \( \mathbb{E}_{P \sim p} C_\mathcal{D}(\pi) \) instead of \( C_\mathcal{D}(\pi) \). Thus, given a test dataset \( S' \) of environments, many policies from the distribution \( P \) must be sampled in order to bound the expected cost on \( S' \). This is not feasible in an OOD detection setting, where there is single execution on the test environments. Our use of the derandomized PAC-Bayes bound in Theorem 1 avoids this issue since we can bound \( C_\mathcal{D}(\pi) \) for a particular policy sampled from \( P \).
We provide approaches for optimizing the bound provided in Theorem 1 using backpropagation (Appendix F) and evolutionary strategies (ES) [50] (Appendix G). Since Theorem 1 requires a deterministic training algorithm, we fix the random seed for stochastic training methods. This makes the algorithm deterministic as the same input will always produce the same output. We choose multivariate Gaussian distributions with diagonal covariance diag(s), i.e., \( P = \mathcal{N}(\mu, \text{diag}(s)) \), for the posterior \( P \) and prior \( P_0 \) distributions. Furthermore, let \( \psi := (\mu, \log s) \); we use the shorthand \( \mathcal{N}_\psi \) for \( \mathcal{N}(\mu, \text{diag}(s)) \). We denote \( \pi_w \) with weights \( w \sim \mathcal{N}_\psi \) as a parameterization of the robot’s policy. In this work, we will use \( w \) to specify weights of a neural network.

As \( w \) is drawn from a multivariate Gaussian distribution \( \mathcal{N}_\psi \), the distribution \( P \) over the policy space is a neural network where each weight is a univariate Gaussian distribution. After training, we sample and fix a \( w \) from the trained posterior for deployment on test environments. We then compute the PAC-Bayes upper bound \( \overline{C}_B(\pi, S) \) and the PAC-Bayes lower bound \( \underline{C}_B(\pi, S) \), each holding with probability \( 1 - \delta \).

### B. Task-Driven OOD Detection With Statistical Guarantees

We now tackle the problem of OOD detection as defined in Section III. The PAC-Bayes training pipeline from Section IV-A produces a policy \( \pi \) with associated bounds \( \overline{C}_B(\pi, S) \) and \( \underline{C}_B(\pi, S) \) on the expected cost \( C_\mathcal{D}(\pi) \) that hold with probability \( 1 - \delta \) over the sampling of the training dataset \( S \sim \mathcal{D}^m \) and the policy \( \pi \sim \mathcal{P} \). Our key idea for OOD detection is that if our PAC bound \( \overline{C}_B(\pi, S) \) is violated by \( \pi \) in the test environments \( S' \) (drawn from the test distribution \( \mathcal{D}' \)), then this indicates that the test environments are OOD-adverse and if \( \underline{C}_B(\pi, S) \) is violated, then this indicates that the test environments are OOD-benign. We present two detectors below that formalize this intuition using two popular frequentist statistical inference tools—hypothesis testing via \( p \)-values and confidence interval overlap.

1) Method 1: Hypothesis testing: The first detector we present leverages hypothesis testing to declare one of the following three outcomes for the test dataset: OOD-adverse (OOD\(_A\)), OOD-benign (OOD\(_B\)), or WD by the detector. We perform this detection by computing upper bounds on the \( p \)-values that hold with high probability. Note that we do not make any normality assumption on the underlying distribution to estimate the \( p \)-values.

To perform hypothesis testing, we first establish a null-hypothesis \( H_0 \) and an alternate hypothesis \( H_1 \), which is the logical negation of \( H_0 \). Statistical inference is then performed by computing the \( p \)-value, which is the likelihood of observing a test dataset \( S \sim \mathcal{D}^m \) with an average cost more extreme\(^1\) than the average cost on the observed test dataset \( S' \sim \mathcal{D}^m \) assuming that \( H_0 \) holds. If the \( p \)-value drops below a significance level \( \alpha \in (0, 1) \), which is chosen before looking at the data, we can conclude that under the null-hypothesis the observed test dataset \( S' \) had a very small probability of being drawn; therefore, the null-hypothesis \( H_0 \) can be rejected.

\(^1\)We will check both, left and right, tails of the distributions.

Our detector performs two hypothesis tests: 1) \( H_0 : \text{OOD}_B \) and \( H_1 : \text{OOD}_A \) and 2) \( H_0 : \text{OOD}_A \) and \( H_1 : \text{OOD}_B \). If the first test returns a \( p \)-value smaller than the significance level \( \alpha_A \), then we declare that the distribution \( \mathcal{D}' \) from which the test dataset \( S' \) is drawn is OOD\(_A\) according to our task-driven notion (1); if the \( p \)-value of the second test is smaller than the significance level \( \alpha_B \), then we declare that environments drawn from the distribution \( \mathcal{D}' \) are OOD\(_B\) according to our notion (2). If both these tests are inconclusive, i.e., \( p \)-values for both are above the significance values, then we cannot declare either OOD\(_A\) or OOD\(_B\) with confidence and therefore declare WD.

A mathematically precise definition of the \( p \)-values for the two tests is given as follows.

**Definition 1 (adapted from [51]):** Let \( \mathcal{D}' \) be the test distribution and \( S' \sim \mathcal{D}^m \) be an observed dataset. Let \( \pi \) be the robot’s control policy. Then, the \( p \)-value for OOD\(_A\) detection is defined as

\[
p_A(S') := \mathbb{P}_{S \sim \mathcal{D}^m} \left[ C_\mathcal{S}(\pi) \geq C_{S'}(\pi) | C_\mathcal{D}(\pi) \leq C_\mathcal{D}(\pi) \right]
\]

and the \( p \)-value for OOD\(_B\) detection is defined as

\[
p_B(S') := \mathbb{P}_{S \sim \mathcal{D}^m} \left[ C_\mathcal{S}(\pi) \leq C_{S'}(\pi) | C_\mathcal{D}(\pi) > C_\mathcal{D}(\pi) \right].
\]

Since we lack an explicit form of the distributions \( \mathcal{D} \) and \( \mathcal{D}' \), direct computation of the \( p \)-values is not feasible. We alleviate this challenge by presenting upper bounds on the \( p \)-values by leveraging the PAC-Bayes generalization bounds (Theorem 1 and Corollary 1). These upper bounds hold with probability \( 1 - \delta \) (over the sampling of \( S \) and \( \pi \)).

**Theorem 2:** Let \( \mathcal{D} \) be the training distribution and \( P \) be the posterior distribution on the space of policies obtained through the training procedure described in Section IV-A. Let \( S' \sim \mathcal{D}^m \) be a test dataset, \( p_A(S') \) and \( p_B(S') \) be the \( p \)-values as defined in Definition 1, \( \delta_A \in (0, 1) \), and \( \delta_B \in (0, 1) \). Then

\[
(i) \quad \mathbb{P}_{(S, \pi) \sim (\mathcal{D}^m \times P)} \left[ p_A(S') \leq \exp(-2n\pi(S))^2 \right] \geq 1 - \delta_A
\]

\[
(ii) \quad \mathbb{P}_{(S, \pi) \sim (\mathcal{D}^m \times P)} \left[ p_B(S') \leq \exp(-2n\pi(S))^2 \right] \geq 1 - \delta_B
\]

where \( \pi(S) := \max\{C_S(\pi) - \overline{C}_\delta_S(\pi, S), 0\} \) and \( \overline{C}(S) := \max\{\underline{C}_\delta_S(\pi, S) - C_{S'}(\pi), 0\} \).

**Proof:** The proof is provided in Appendix B.

Theorem 2 provides an upper bound on the \( p \)-values, which hold with high confidence. If the upper bound is below the respective significance levels \( \alpha_A \) or \( \alpha_B \), then with high confidence we can say that the \( p \)-value is below \( \alpha_A \) or \( \alpha_B \); thereby, Theorem 2 facilitates OOD-adverse/OOD-benign detection through hypothesis testing. The resulting detector is detailed in the Algorithm 1.

A natural question to ask is whether the \( p \)-values for both hypothesis tests can be less than their respective significance levels \( \alpha_A \) and \( \alpha_B \), implying that a dataset is simultaneously OOD\(_A\) and OOD\(_B\) with high probability. In the forthcoming lemma, we show that the abovementioned detector indeed returns mutually exclusive outputs.
Algorithm 1: OOD-adverse/OOD-benign Detection using Hypothesis Testing.

Input: $\delta_A, \delta_B, \alpha, \alpha_B \in (0, 1)$.
Input: PAC-Bayes Bounds: $\overline{C}_{S,\delta}(\pi, S), \overline{C}_{B,\delta}(\pi, S)$.
Input: Test dataset $S' \sim D'^n$ and policy $\pi \sim P$.
Output: OOD$_A$, OOD$_B$, and WD.

$C_S(\pi) \leftarrow \frac{1}{n} \sum_{E \in S'} C_E(\pi)$
$\tau \leftarrow \max\{C_S(\pi) - \overline{C}_{S,\delta}(\pi, S), 0\}$
$\tau' \leftarrow \max\{C_B(\pi, S) - C_S(\pi), 0\}$
if $\exp(-2n\tau^2) \leq \alpha_A$ then
    OOD$_A \leftarrow$ True
end if
if $\exp(-2n\tau'^2) \leq \alpha_B$ then
    OOD$_B \leftarrow$ True
end if
if $\exp(-2n\tau^2) > \alpha_B$ and $\exp(-2n\tau'^2) > \alpha_A$ then
    WD $\leftarrow$ True
end if

Lemma 1: Algorithm 1 returns mutually exclusive outputs, i.e., it returns only one of the three possibilities: OOD$_A$, OOD$_B$, or WD.

Proof: A detailed proof is provided in Appendix C.

2) Method 2: Confidence interval on the difference in expected train and test costs: We now present another method for detecting task-relevant distribution shifts [see Section III, (1) and (2)] by providing bounds on the difference between expected test cost ($C_D(\pi)$) and the expected training cost ($C_T(\pi)$) that hold with high probability. Using a confidence-interval-based method allows us to provide a guaranteed FPR and FNR for our detector, which is important for reliable use in safety-critical environments. We provide two lower bounds: 1) $\Delta C_A$, which lower bounds $C_D(\pi) - C_T(\pi)$ and 2) $\Delta C_B$, which lower bounds $C_T(\pi) - C_D(\pi)$. If $\Delta C_A$ is positive then (with high confidence) $C_D(\pi) > C_T(\pi)$, which corresponds to task-driven OOD-adverse detection. Similarly, if $\Delta C_B$ is non-negative then (with high confidence) $C_T(\pi) \geq C_D(\pi)$, which corresponds to task-driven OOD-benign detection. Finally, if $\Delta C_B$ and $\Delta C_A$ are negative then we cannot declare either OOD-benign (OOD$_B$) or OOD-adverse (OOD$_A$) with confidence and, therefore, declare that environments drawn from the given test dataset $S'$ is WD. We formalize these high-confidence bounds in Theorem 3.

Theorem 3: Let $D$ be the training distribution, $D'$ be the test distribution, and $P$ be the posterior distribution on the space of policies obtained through the training procedure described in Section IV-A. Let $\delta_A, \delta'_A \in (0, 1)$ such that $\delta_A + \delta'_A < 1$, $\gamma_A := \sqrt{\ln(1/\delta'_A) / 2n}$, and $\Delta C_A := C_S(\pi) - \gamma_A - \overline{C}_{S,\delta}(\pi, S)$. Similarly, let $\delta_B, \delta'_B \in (0, 1)$ such that $\delta_B + \delta'_B < 1$, $\gamma_B := \sqrt{\ln(1/\delta'_B) / 2n}$, and $\Delta C_B := \overline{C}_{B,\delta}(\pi, S) - C_S(\pi) - \gamma_B$. Then

\[(i) \quad \mathbb{P}_{(S, \pi, S') \sim (D^m \times P \times D'^n)} [C_D(\pi) - C_T(\pi) \geq \Delta C_A] \geq 1 - \delta_A - \delta'_A \quad (9)\]

\[(ii) \quad \mathbb{P}_{(S, \pi, S') \sim (D^m \times P \times D'^n)} [C_D(\pi) - C_T(\pi) \geq \Delta C_B] \geq 1 - \delta_B - \delta'_B \quad (10)\]

Proof: A detailed proof of this theorem is provided in Appendix D.

The detection scheme based on Theorem 3 is outlined in Algorithm 2. It is important to note that with this detection scheme, the user can pick the desired false positive and false negative OOD detection rates by selecting $\delta_A, \delta'_A, \delta_B,$ and $\delta'_B$ (see Remark 1). This allows us to tune the detector’s sensitivity to distribution shifts according to the situation in which it is deployed. For example, in safety-critical situations one may want to deploy a policy only when we are confident that the robot is operating OOD-benign or WD, hence, we can choose a low maximum permissible FNR. However, when operating in nonsafety-critical settings, a higher FNR can be tolerated, in which case the OOD-benign detector can afford to make declarations less cautiously.

Algorithm 2: OOD-Adverse/OOD-Benign Detection Using Confidence Intervals.

Input: $\delta_A, \delta'_A \in (0, 1)$ with desired maximum false positive rate $\delta_A + \delta'_A < 1$.
Input: $\delta_B, \delta'_B \in (0, 1)$ with desired maximum false negative rate, $\delta_B + \delta'_B < 1$.
Input: PAC-Bayes Bounds: $\overline{C}_{S,\delta}(\pi, S), \overline{C}_{B,\delta}(\pi, S)$.
Input: Test dataset $S' \sim D'^n$ and policy $\pi \sim P$.
Output: OOD$_A$, OOD$_B$, and WD.

$C_S(\pi) \leftarrow \frac{1}{n} \sum_{E \in S'} C_E(\pi)$
$\gamma_A \leftarrow \sqrt{\ln(1/\delta'_A) / 2n}$
$\gamma_B \leftarrow \sqrt{\ln(1/\delta'_B) / 2n}$
$\Delta C_A \leftarrow C_S(\pi) - \gamma_A - \overline{C}_{S,\delta}(\pi, S)$
$\Delta C_B \leftarrow \overline{C}_{B,\delta}(\pi, S) - C_S(\pi) - \gamma_B$
if $\Delta C_A > 0$ then
    OOD$_A \leftarrow$ True
end if
if $\Delta C_B \geq 0$ then
    OOD$_B \leftarrow$ True
end if
if $\Delta C_A \leq 0$ and $\Delta C_B < 0$ then
    WD $\leftarrow$ True
end if

Proof: A detailed proof is provided in Appendix E.

With the confidence-interval-based detector as outlined in Algorithm 2 and Theorem 3, we can guarantee the FPR and the FNR of the detector to be upper bounded by $\delta_A + \delta'_A$ and $\delta_B + \delta'_B$, respectively, as shown in Remark 1.
Remark 1: The detection scheme presented in Algorithm 2 has an FPR upper bounded by $\delta_B + \delta_A'$ and an FNR upper bounded by $\delta_B + \delta_B'$. This is evident from Theorem 3, which states the following.

1) $\Delta C_A > 0 \rightarrow \mathbb{P}[C_D'(\pi) > C_D(\pi)] \geq 1 - \delta_A - \delta_A'$ if and only if $\mathbb{P}[C_D'(\pi) > C_D(\pi)] < \delta_A + \delta_A'$. Therefore, when the detector declares OOD-adverse ($\Delta C_A > 0$), the environments may be OOD-benign or WD with a probability at most $\delta_A + \delta_A'$, i.e., the maximum FPR associated with this detector is upper bounded by $\delta_A + \delta_A'$.

2) $\Delta C_B \geq 0 \rightarrow \mathbb{P}[C_D'(\pi) > C_D(\pi)] \geq 1 - \delta_B - \delta_B'$ if and only if $\mathbb{P}[C_D'(\pi) > C_D(\pi)] < \delta_B + \delta_B'$. Therefore, when the detector declares OOD-benign ($\Delta C_B \geq 0$), the environments may be OOD-adverse or WD with a probability at most $\delta_B + \delta_B'$, i.e., the maximum FNR associated with this detector is upper bounded by $\delta_B + \delta_B'$.

V. EXAMPLES

We demonstrate the ability of our approach to perform task-driven OOD detection with guaranteed confidence bounds on two examples in both simulation and on hardware: a manipulator grasping a new set of objects and a drone navigating a new set of environments. For the navigation task, we compare our methods with popular OOD detection baselines. Our code is available at: https://github.com/irom-princeton/Task_Relevant_OOD_Detection/tree/extensions, and videos of the experiments can be found at https://youtu.be/jKye3A09le0.

A. Robotic Grasping

1) Overview: We use the Franka Panda arm [see Fig. 2(a)] for grasping objects in the PyBullet simulator [52] and build upon the open-source code provided in [47]. The robot employs a vision-based control policy that uses a depth map of the object obtained from an overhead camera and returns an open-loop action $a := (x, y, z, \theta)$, which corresponds to the desired grasp position and yaw orientation of the gripper. We train the manipulator to grasp mugs placed in $\text{SE}(2)$ poses drawn from a uniform distribution; see Appendix H for details. Then, we demonstrate the efficacy of our OOD-adverse detection framework by 1) gradually modifying the distribution on the mug poses and 2) changing the objects from mugs to bowls.

2) Control Policy: The control policy is a DNN, which inputs a $128\times128$ depth map of the object and a latent state $z \in \mathbb{R}^{10}$ sampled from a multivariate Gaussian distribution $N_\psi$ with a diagonal covariance, and outputs an open-loop grasp action $a$; see Fig. 8 in Appendix H for the policy. In [47], the distribution $N_\psi$ on the latent space encodes prior domain/expert knowledge.

3) Training: Mugs from the ShapeNet dataset [53] are randomly scaled in all dimensions to generate a training dataset $S$ of 500 mugs. If the robot is able to lift the mug by 10 cm, then we consider the rollout successful and assign a cost of 0; otherwise the cost is set to 1. In training, we optimize the distribution $N_\psi$ on the latent space to minimize the PAC-Bayes bound provided in Theorem 1 using Algorithm 4, while the weights of the CNN and MLP networks in Fig. 8 in Appendix H remain fixed. The prior $N_\psi$ is chosen as the normal distribution with zero mean and identity covariance. Weights $w$ are sampled from the trained posterior $N_\psi$ to generate a policy $\pi_w$ and the PAC-Bayes bound for this policy is computed as $C_\Delta(\pi, S) = 0.1$ with $\delta = 0.01$.

4) Simulation Results: We perform OOD-adverse and OOD-benign detection using the two methods presented in Theorems 2 and 3. For detection with $p$-value, we choose a significance level $\alpha_A = 95\%$, while, for detection using $\Delta C_A$ (the lower bound on $C_D' - C_D$), we choose a confidence level of 95\%, i.e., $\delta_A + \delta_A' = 0.05$, which ensures that the FPR of our detector is no greater than 5\%. We perform two experiments to demonstrate the efficacy of our approach. First, we make the distribution on the mug’s initial placement progressively more challenging; see Appendix H for the exact distributions. For each distribution, we sample a test dataset of cardinality 10 and compute our OOD indicators: 1) the lower bound on $1 - p_A$ (where $p_A$ is the $p$-value) and 2) $\Delta C_A$ using Theorem 3. Fig. 2(b) plots the mean (dashed line) and a one standard deviation spread (shaded region) for the OOD indicators computed using 20 test datasets as a function of $C_D' - C_D$ (estimated via exhaustive sampling). Note that we plot $\Delta C_A + 0.95$ so that the OOD threshold is
the same (0.95) for both methods. We tested in settings that are more challenging than the training setting and did not detect OOD-benign for any of these settings. Thus, we focus our analysis on the OOD-adverse detectors. However, a potential example of an OOD-benign environment could be grasping a single mug from the training set consistently placed in the same location.

As the cost of the policy deteriorates on test distributions our OOD_A indicators reliably increase, capturing the shift of the test distributions away from the training distribution. In the second experiment, we change the objects that the manipulator must grasp from mugs to bowls. Fig. 2(c) shows that with a small test dataset $S'$ of cardinality 5, both our approaches detect OOD_A when bowls are used (red curves). As expected, our OOD detectors are not triggered for mugs (blue curves), which are drawn from the training distribution.

5) Hardware Results: We perform hardware experiments using the Franka Panda robot arm with input from a downward-facing camera mounted above the manipulator [see Fig. 3(a)]. In these experiments, we use the same policy that was trained on mugs in simulation and then evaluate the performance of this policy on grasping 10 mugs with varied location (progressively encompassing a larger region) and grasping 10 bowls. Note that the detector also captures benign or adverse distribution shifts due to any sim-to-real gap. Our experiments use mugs with shapes similar to those in the ShapeNet dataset [53], to be consistent with the simulation.

Changing location of mugs: For each mug, we vary the location of the mugs according to the first 5 distributions outlined in Appendix H, which correspond to uniform distribution with ranges of 0.1 m, 0.2 m, 0.3 m, 0.4 m, and 0.5 m centered around the middle of camera’s field of view. The sixth distribution is neglected for the hardware trials, as it includes regions that were outside the field of view of the camera. As the range of location gets larger when compared with the training distribution, we find that the proportion of trials that the grasp fails (indicating a task relevant distribution shift) increases. The proportion of failures is used to compute the value of $\Delta C_A + 0.95$ and $1 - p_A$, which monotonically increases as plotted in Fig. 3(b). This is consistent with the simulation results presented in Fig. 2(b).

Note that while we can measure the ground truth $C_{T'R'} - C_{T'R'}$ via exhaustive sampling in simulation, we cannot obtain this ground truth information in hardware experiments. Thus, we use the range of the mug location as a proxy for $C_{T'R'} - C_{T'R'}$ in the hardware results. In hardware, both our detectors declare OOD_A when the range of mug locations is 0.5 m, compared to the 0.4 m and 0.5 m ranges declared to be OOD_A in simulation.

Grasping bowls: The value of $\Delta C_A + 0.95$ and $1 - p_A$ after each attempted grasp of a bowl is plotted in Fig. 3(c) (the raw experimental data is included in Appendix H). In hardware, we detect OOD_A with just 3 test environments of bowls, which is similar to the 4 test environments needed in simulation. Overall, the hardware experiments are consistent with the OOD_A detections in simulation where we detect OOD_A when the range of mug locations increases and for a low-cardinality dataset of bowls.

B. Vision-Based Obstacle Avoidance With a Drone

1) Overview: In both the simulation and hardware portions of this example, we aim to avoid an obstacle field with the Parrot Swing drone; this is an agile quadrotor/fixed-wing hybrid drone shown in Fig. 1. We train a DNN control policy in a simulation setup based on the hardware system shown in Fig. 1. The policy takes in a 50 × 50 depth image and outputs a softmax corresponding to a set of precomputed motion primitives with the goal of avoiding obstacles by the largest distance. Since we designed the simulation portion of this example with application to hardware in mind, we have created motion primitives by capturing (with a Vicon motion tracking system) the trajectories of open-loop control inputs. This results in different maneuvers; the two images in Fig. 1 represent two of these trajectories. The use of motion primitives allows us to perform accurate sim-to-real transfer (as the motion primitives are recorded from the hardware system).

2) Training: Environments consist of a set of randomly placed cylindrical obstacles. We record the minimum distance $d_{min}$ from the obstacles (as recorded by the robot’s 120° field of view)
environments [see \( C = 100 \) from a given distribution that acts as a threshold or \( \Delta = 500 \).] To demonstrate \( \Delta \) and training environments.

To estimate the test cost, we sample a policy \( \pi \) from the trained posterior and compute the PAC-Bayes bounds \( \overline{C}_A(\pi, S) \) and \( \underline{C}_A(\pi, S) \) with \( \delta = 0.01 \) for each of the datasets.

3) Simulation Results: To evaluate our OOD detection methods on vision-based obstacle avoidance in simulation, we randomly generate test datasets of varied environment difficulty by changing the number of obstacles and the maximum or minimum gap-size between obstacles. We perform OOD detection using 10 test environments for each difficulty setting (and present results averaged over 2000 such datasets). With these datasets, we estimate the expected test cost that the policy \( \pi \) would incur on any given difficulty setting; this estimate is used to evaluate which environments our detection schemes should declare as OOD-adverse/OOD-benign (i.e., which difficulty settings are OODA/OODB in our task-relevant sense). To estimate the test cost, we use \( d_{\text{thresh}} = 500 \) mm for the simulation results presented in this section. To evaluate our detectors in simulation, we first verify that the guarantees presented in Section IV-B do indeed hold. We then compare our detectors with two OOD detection baselines: 1) maximum softmax probability (MSP) [4] (an effective and popular baseline for OOD detection), and 2) MaxLogit [6] (a recent state-of-the-art OOD detection baseline). We find that our detection schemes perform similarly to these baselines for task-relevant shifts. However, the baseline detectors are triggered by task irrelevant shifts, while our methods are only triggered by task-relevant shifts. In addition, our detectors provide guarantees on the FPR and FNR and can declare WD environments and also differentiate between OOD-adverse and OOD-benign environments, while the baselines are limited to detecting OOD generically and provide no guarantees.

**OOD-adverse and OOD-benign detection:** To demonstrate our detection schemes (Algorithm 1 and 2), we evaluate the proportion of datasets \( S' \sim D^{\text{tt}} \) from a given distribution that our detectors declare OOD-adverse, OOD-benign, or WD (see Fig. 4). The \( x \)-axis of this plot is the estimated cost (with \( d_{\text{thresh}} = 500 \) mm) of the environment as compared to the training environment; positive \( \Delta C_A \) indicate (task-relevant) OODA environments [see (1)] and nonnegative \( \Delta C_B \) indicate (task-irrelevant) OODB environments [see (2)]. Note that at 0 on the \( x \)-axis of this, we draw datasets from the training distribution. We observe that our confidence-interval based detector [see Algorithm 2, Fig. 4(a)] maintains both its FPR and FNR guarantee, and does not declare any OODA environments as OODB or vice versa (see Fig. 9 in Appendix H for numerical validation of the guaranteed FPR). In addition, the \( p \)-value based method [see Algorithm 1, Fig. 4(b)] performs very similarly and also does not incorrectly classify OOD-adverse as OOD-benign or vice versa. The accuracy of these declarations, however, comes at the cost of conservatism where both detectors output WD when \( C_A(\pi) - C_S(\pi) \) is small, i.e., environments that are on the border of OOD-adverse/OOD-benign. One can counteract this by changing \( \delta_A \) and \( \delta_B \), and, therefore, the desired guarantees.

**Choosing desired maximum FPR and FNR:** The confidence-interval-based detector (see Algorithm 2), as shown in Remark 1, allows for the desired maximum permissible FPR \( (\delta_A + \delta_A') \) and FNR \( (\delta_B + \delta_B') \) to be picked a priori. Higher admissible FPRs/FNRs would allow the detector to declare OOD-adverse/OOD-benign instead of WD for test environments with similar costs to training environments, i.e., for values of \( C_D(\pi) - C_D(\pi) \) close to zero. On the contrary, lower admissible FPRs/FNRs would result in OOD-adverse/OOD-benign declarations only for those environments with test costs very
different from training costs. This property could be especially useful in practice, where the threshold difference in test and train costs for which OOD-adverse/OOD-benign is declared could be altered depending on how safety critical the situation is. For exposition, we pick $\delta_A = \delta_B = 0.01$ and use only $\delta'_A$ and $\delta'_B$ to vary the maximum permissible FPR/FNR. In Fig. 5(a), we show that increasing our permissible FPRs and FNRs to 40% ($\delta_A' + \delta_B = \delta_B' + \delta_B' = 0.40$) results in WD declarations being replaced by OOD-benign or OOD-adverse declarations as compared to the 5% FPR/FNR in Fig. 4(a). It is also possible to set $\delta'_A$ and $\delta'_B$ to be different values, as is displayed in Fig. 5(b), where the permissible FNR is chosen to be 10% ($\delta_B' + \delta_B' = 0.10$), while the permissible FPR is 90% ($\delta_A' + \delta_A' = 0.90$). This is particularly suitable for safety-critical contexts where it can be dangerous to declare OOD-adverse environments as OOD-benign, i.e., a low FNR is desirable. In Fig. 5(b), we see that we declare OOD-adverse (red bars) for more environments, including some of those with similar test and train costs and only declare OOD-benign when the test cost is significantly below the training cost. Note that the true FPR/FNR is below the maximum permissible rates, consistent with the guarantee presented in Remark 1.

Comparison with Baselines for OOD detection: We compare our task-driven OOD-adverse detection approach with two baselines: 1) MSP [4], and 2) MaxLogit [6]; see Appendix H for more details on the baseline approaches. We note that these baselines are specifically designed for networks, which output categorical distributions, and thus, provide strong benchmarks. It is important to note that these baselines do not provide a means to detect OOD-benign environments and so to compare our detection methods to these baselines, we only consider our OOD-adverse declarations. The results are plotted in Fig. 6, using a $p$-value of 0.05 and a guaranteed FPR and FNR of 5% for the confidence interval method. We see that these guarantees do indeed hold, and declare a OOD-benign environments as OOD-adverse less than 5% of the time. In contrast, the baselines do not provide any guarantees; they violate the FPR even on new environments drawn from the training distribution ($C_{DP}(\pi) > C_{DP}(\pi)$). In addition, the baselines do not differentiate between OOD-adverse and OOD-benign environments; they declare environments with $C_{DP}(\pi) > C_{DP}(\pi)$ as OOD, when in practice the policy would not result in failure on these (easier) environments. With regard to OOD-adverse environments, our detectors (which perform similarly) only detect test environments with...
higher expected costs as OOD-adverse, demonstrating that it is task-relevant.

Task-irrelevant shift: As seen in Fig. 6, the baselines are triggered even for task-irrelevant shifts in the environment distribution. For example, the baselines are triggered on environments with lower expected costs than the training environment ($C_D(\pi) - C_D(\pi) = -0.3$). To illustrate this, we compare the baselines with our methods on a distribution where environments consist of 4 (uniformly) randomly located obstacles, and we evaluate costs using a threshold distance, $d_{\text{thresh}} = 500$ mm. In this setting, the control policy achieves a near-identical expected cost $C_D(\pi)$ (as estimated by exhaustive sampling of environments) to the expected training cost $C_D(\pi)$ (in particular, $C_D(\pi) - C_D(\pi) = -0.02$). Therefore, if we view this setting in the context of the performance of the policy, the difference in distribution is task-irrelevant. However, for this setting, MSP [4] classified 100% of test datasets as OOD and MaxLogit [6] classified 98.6% as OOD. Thus, the baselines are triggered by a task-irrelevant shift in the distribution. In contrast, our OOD-adverse detection method had a detection rate of only 0.1% in this setting.

4) Hardware Results: We use a Parrot Swing drone for the hardware experiments (see Fig. 1). We simulate a depth sensor for the drone (as if the sensor was mounted on the drone) by generating a synthetic depth image using the positions of objects from the Vicon motion capture system. We do not provide any other information to the policy, such as the position of obstacles or the environmental wind conditions. We generate each environment the same way as in simulation, and then place the real-world obstacles in the generated locations.

Varied environment difficulty and wind disturbances: We deploy the policy trained in simulation on the following three kinds of OOD environments in hardware:

1) environments with a smaller number of obstacles (i.e., “easier” environments);
2) environments with smaller gaps between obstacles (i.e., “harder” environments);
3) environments with wind generated using a fan (see Fig. 1 right) with the same obstacle distribution as training.

For each setting, we run 10 trials on the hardware and use this for OOD-adverse detection, where we evaluate costs using the obstacle threshold radius, $d_{\text{thresh}} = 300$ mm. As expected, our OOD-adverse detectors are not triggered by the easier environments. For the harder environments, we compute $1 - p_A \geq 0.81$ and $\Delta C_A = -0.11$. Results from the windy environments are shown in Fig. 7 for increasing values of wind (up to about 5 m/s).

We note that the sim-to-real distribution shift (corresponding to the zero wind case) is not viewed as being OOD in a task-relevant manner by our approaches. Both our approaches assign OOD-adverse with increasingly high confidence as the wind speed is increased. We note that this detection is despite the fact that disturbances such as wind cannot be detected via the depth image given to the robot’s policy. Thus, any OOD detection technique that relies solely on the output of the policy, such as MSP [4] and MaxLogit [6], would be unable to detect these environments as OOD.

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
to grasp may be low. However, in a safety-critical system, rollouts with increased cost may not be tolerable. It would be of practical interest to extend our approach to settings where the robot encounters environments in an online manner (instead of the batch setting we consider here). Another particularly exciting direction is to develop versions of our approach that are more proactive; instead of having to incur costs on the test environments, one could potentially perform OOD-adverse detection based on predicted costs (thus avoiding the need to potentially fail on the test environments) such as in [54]. Finally, another potential direction for future research is to leverage the OOD-adverse/OOD-benign detection schemes presented here to learn policies that are distributionally robust; in particular, one could envision learning policies such that environments from a broad range of test distributions are detected as OOD-benign in the task-relevant sense employed in this article.

APPENDIX

A. Proof of Theorem 1

We begin with an introduction to the PAC-Bayes framework, and then provide a complete proof of Theorem 1. PAC-Bayes provides an upper bound on the expected cost of deploying a policy distribution $P$ on environments $E$ drawn from an unknown distribution $D$, i.e., $E_{E \sim D} \mathbb{E}_{\pi \sim P} C_E(\pi)$. This upper bound only depends on the cost of deploying $P$ in a finite set of training environments $S \sim D^m$, i.e., the training cost $\mathbb{E}_{\pi \sim P} C_E(\pi)$, and a regularizer that depends on the KL-divergence between $P$ and a prior $P_0$ that is chosen before observing $S$; note that $P_0$ need not be a Bayesian prior. The following is the PAC-Bayes bound that was presented in [30] and tightened in [55].

Theorem 4 (PAC-Bayes Bound [30]): For any distribution over environments $D$, data-independent prior distribution $P_0$, cost $C$ bounded in $[0,1]$, $m \geq 8$, and $\delta \in (0,1)$, with probability at least $1-\delta$ over a sampling of $S \sim D^m$, the following holds for all posterior distributions $P$:

$$
\mathbb{E}_{E \sim D} \mathbb{E}_{\pi \sim P} C_E(\pi) \leq \sum_{i=1}^{m} \mathbb{E}_{\pi \sim P} C_E(\pi) + D_{KL}(P\|P_0) + \ln \frac{2\sqrt{m}}{\delta}
$$

(11)

where $D_{KL}$ is the KL-divergence.

The abovementioned theorem and the forthcoming PAC-Bayes theorems in this section are presented for policy learning instead of supervised learning using the reduction provided in [45]. Note that this bound provides a guarantee for a distribution over policies rather than a specific policy. This allows for a regularizer dependent on the KL-divergence between the prior and posterior distributions rather than one which is a direct expression of the complexity of the policy space (such as the VC-dimension). However, this creates a challenge for calculating the upper bound, which requires computing an expectation over $\pi \sim P$, or using potentially loosening sample convergence bounds. Thus, we make use of the recent work, which provides a framework for derandomized PAC-Bayes bounds (i.e., bounds which hold for a sampling of policy $\pi$ rather than an expectation over $\pi \sim P$) [49]. The following is a general theorem for formulating the derandomized PAC-Bayes bounds.

Theorem 5 (Pointwise PAC-Bayes Bound [49]): For any positive function $\phi$, distribution $D$, prior distribution $P_0$, and $\delta \in (0,1)$, with probability $1-\delta$ over a sampling of $S \sim D^m$ and $\pi \sim P$, the following holds for any posterior distribution $P$:

$$
\frac{\alpha}{\alpha-1} \ln(\phi(\pi, S)) \leq D_{\alpha}(P\|P_0) + \ln \left(\frac{1}{(\delta/2)^2} \mathbb{E}_{S^m \sim P_0} \mathbb{E}_{\pi \sim P} \phi(\pi', S') \right) + \ln \left(\frac{1}{(\delta/2)^2} \mathbb{E}_{S^m \sim P_0} \mathbb{E}_{\pi \sim P} \phi(\pi', S') \right)
$$

(12)

where $P$ is the output of algorithm $A$ on the training data $S$, i.e., $P := A(P_0, S)$ and $D_{\alpha}$ is the Rényi divergence.

Now we can proceed with the statement and proof.

Theorem 6: 1 For any distribution $D$, prior distribution $P_0$, $\delta \in (0,1)$, cost bounded in $[0,1]$, and deterministic algorithm $A$, which outputs the posterior distribution $P$ we have the following:

$$
\mathbb{P}_{(S,\pi) \sim (D^m \times P)} \left[ C_D(\pi) \leq \mathbb{E}_{\pi \sim P} C_S(\pi) + \sqrt{\frac{D_{KL}(S\|P_0) + \ln \frac{2\sqrt{m}}{\delta}}{2m}} \right] \leq 1 - \delta
$$

(13)

where $D_\alpha$ is the Rényi Divergence for $\alpha = 2$.

Proof: We begin with the statement in Theorem 5, which is proved in [49]. Let $\alpha = 2$ and $\phi(\pi, S) = \exp[\frac{\alpha}{\alpha-1} m D_{KL}(C_S(\pi)\|C_P(\pi))]$. Thus, we have the following with at least probability $1-\delta$ over the random choice $S \sim D^m$ and $\pi \sim P$:

$$
D_{KL}(S\|P) \leq \frac{1}{m} D_{KL}(C_P(\pi)) + \ln \left(\frac{1}{(\delta/2)^2} \mathbb{E}_{S^m \sim P_0} \mathbb{E}_{\pi \sim P} e^{m D_{KL}(C_S(\pi')\|C_P(\pi'))} \right)
$$

(14)

From [55], we can upper bound $\mathbb{E}_{S^m \sim P_0} \mathbb{E}_{\pi \sim P} e^{m D_{KL}(C_S(\pi')\|C_P(\pi'))}$ by $2\sqrt{m}$ when $m \geq 8$. This gives us the following bound:

$$
D_{KL}(S\|P) \leq \frac{1}{m} D_{KL}(P\|P_0) + \ln \frac{2\sqrt{m}}{\delta/2^4}
$$

(15)

We then apply the Pinkser’s inequality, i.e., $D_{KL}(P\|q) \leq c \to q \leq p + \sqrt{c/2}$, which results in Inequality (3). Note that we could also use a quadratic version of the upper bound for the KL divergence between two distributions and produce an upper bound analogous to the one presented in [42].

B. Proof of Theorem 2

For the readers’ convenience, we restate Theorem 2 here and provide a detailed proof.

Theorem 7: Let $D$ be the training distribution and $P$ be the posterior distribution on the space of policies obtained through the training procedure described in Section IV-A. Let $S' \sim D^m$ be a test dataset, $P_A(S')$ and $P_B(S')$ be the $p$-values as defined
in Definition 1, $\delta_A \in (0, 1)$, and $\delta_B \in (0, 1)$. Then

(i) $\Pr_{(S,\pi)\sim(D^m \times P)}[p_A(S') \leq \exp(-2n\tau(S)^2)] \geq 1 - \delta_A$ \hspace{1cm} (16)

(ii) $\Pr_{(S,\pi)\sim(D^m \times P)}[p_B(S') \leq \exp(-2n\tau(S)^2)] \geq 1 - \delta_B$ \hspace{1cm} (17)

where $\tau(S) := \max\{C_S(\pi) - \overline{C}_\delta(\pi, S), 0\}$ and $\tau(S) := \max\{C_{\hat{\delta}}(\pi, S) - C_S(\pi), 0\}$.

To prove Theorem 2, we establish the following lemmas.

Lemma 3: Let the assumptions of Theorem 2 hold. For notational simplicity and without loss of generality, we let $\delta_A = \delta$.

Then

$\Pr_{(S,\pi)\sim(D^m \times P)}[p_A(S') \leq \exp(-2n\tau(S)^2)] \geq 1 - \delta$ \hspace{1cm} (18)

where $\tau(S) := \max\{C_S(\pi) - \overline{C}_\delta(\pi, S), 0\}$.

Proof: We prove this lemma by considering two cases: when the PAC-Bayes cost inequality in Theorem 1 holds, i.e., $C_D(\pi) \leq \overline{C}_\delta(\pi, S)$, and when it does not, i.e., $C_D(\pi) > \overline{C}_\delta(\pi, S)$; the two cases are considered in (19) and (20). In the latter case, we cannot say anything about the $p$-value, while in the former case, which holds with probability at least $1 - \delta$, we show in (21)–(22) that $p_A(S') \exp(-2n\tau(S)^2)$

Let us begin the proof by conditioning

$\Pr_{(S,\pi)\sim(D^m \times P)}[p_A(S') \leq \exp(-2n\tau(S)^2)]$ as follows:

$\Pr_{(S,\pi)\sim(D^m \times P)}[p_A(S') \leq \exp(-2n\tau(S)^2)]$\hspace{1cm} (23)

$\geq \Pr_{(S,\pi)\sim(D^m \times P)}[C_D(\pi) \leq \overline{C}_\delta(\pi, S)]$\hspace{1cm} (19)

$\geq 0$\hspace{1cm} (20)

which on using in (20) completes the proof of this lemma. The rest of this proof is dedicated to establishing the claim in (21).

We are given

$C_D(\pi) \leq \overline{C}_\delta(\pi, S)$.

From Definition 1, we have

$p_A(S') = \Pr_{S \sim D^n}[C_S(\pi) \geq C_S'(\pi)|C_D(\pi) \leq C_D(\pi)]$ \hspace{1cm} (23)

From (22) and the assumption that the null hypothesis holds in (25), it follows that $C_D(\pi) \leq \overline{C}_\delta(\pi, S)$, which ensures that the following implication holds for $\tau$ defined in the statement of the lemma:

$C_S(\pi) - \overline{C}_\delta(\pi, S) \geq \tau \implies C_S(\pi) - C_D(\pi) \geq \tau$ \hspace{1cm} (26)

Therefore, if $\tau > 0$ we have that

$\Pr_{S \sim D^n}[C_S(\pi) - \overline{C}_\delta(\pi, S) \geq \tau|C_D(\pi) \leq C_D(\pi)]$ \hspace{1cm} (27)

Combining the two cases for $\tau > 0$ in (27) and $\tau = 0$ in (28) gives us the following implication:

$C_D(\pi) \leq \overline{C}_\delta(\pi, S) \implies p(S') \leq \exp(-2n\tau(S)^2)$.

Now, we expand the left-hand side of (21) using the definition of conditional probability

$\Pr_{(S,\pi)\sim(D^m \times P)}[p_A(S') \leq \exp(-2n\tau(S)^2)]$\hspace{1cm} (29)

$\geq 0$\hspace{1cm} (30)

which on using in (20) completes the proof of the claim in (21).

Lemma 4: Let the assumptions of Theorem 2 hold. For notational simplicity and without loss of generality, we let $\delta_W = \delta$. 

$$\Pr_{S \sim D^n}[C_S(\pi) - \overline{C}_\delta(\pi, S) \geq C_S'(\pi) - \overline{C}_\delta(\pi, S)]$$

$$\geq \Pr_{S \sim D^n}[C_D(\pi) \leq C_D(\pi)]$$

$$\Pr_{S \sim D^n}[C_S(\pi) - \overline{C}_\delta(\pi, S) \geq \tau|C_D(\pi) \leq C_D(\pi)].$$
Then,
\[ P_{S,\pi}^{S,n} \left[ p_B(S') \leq \exp(-2n\tau(S)^2) \right] \geq 1 - \delta \]  
(33)
where \( \tau(S) := \max\{C_\delta(\pi, S) - C_{\delta'}(\pi, 0)\} \).

Proof: We prove this lemma by considering two cases: when the PAC-Bayes cost inequality in Corollary 1 holds, i.e., \( C_P(\pi) \geq C_\delta(\pi, S) \), and when it does not, i.e., \( C_P(\pi) < C_\delta(\pi, S) \); the two cases are considered in (19) and (20). In the latter case, we cannot say anything about the \( p \)-value, while in the former case, which holds with probability at least \( 1 - \delta \), we show in (21)–(32) that \( p_B(S') \leq \exp(-2n\tau(S)^2) \).

Let us begin the proof by conditioning on \( (S,\pi) \sim (D^n \times P) \):
\[ P_{S,\pi}^{S,n} \left[ p_B(S') \leq \exp(-2n\tau(S)^2) \right] \]
\[ = P_{S,\pi}^{S,n} \left[ p_B(S') \leq \exp(-2n\tau(S)^2) | C_P(\pi) \geq C_\delta(\pi, S) \right] \]
\[ + P_{S,\pi}^{S,n} \left[ p_B(S') \leq \exp(-2n\tau(S)^2) | C_P(\pi) < C_\delta(\pi, S) \right] \]
\[ \geq 0 \]
(34)
\[ P_{S,\pi}^{S,n} \left[ C_P(\pi) \geq C_\delta(\pi, S) \right] = 1 \]
(35)
which on using in (35) completes the proof of this lemma. The rest of this proof is dedicated to establishing the claim in (36).

We are given
\[ C_P(\pi) \geq C_\delta(\pi, S), \]
(37)
From Definition 1, we have
\[ p_B(S') = P_{S \sim D^n} \left[ C_S(\pi) \leq C_S(\pi)(C_P(\pi) > C_P(\pi)) \right] \]
(38)
\[ + P_{S \sim D^n} \left[ C_\delta(\pi, S) - C_S(\pi) \geq C_\delta(\pi, S) - C_S(\pi) \right] \]
(39)
\[ = P_{S \sim D^n} \left[ C_\delta(\pi, S) - C_\delta(\pi) \geq C_\delta(\pi, S) - C_\delta(\pi) \right] \]
(40)
\[ = P_{S,\pi}^{S,n} \left[ p_B(S') \leq \exp(-2n\tau(S)^2) \right] \leq C_\delta(\pi, S) \]
(41)
From (37) and the assumption that the null hypothesis holds in (40), it follows that \( C_P(\pi) \geq C_\delta(\pi, S) \), which ensures that the following implication holds for \( \tau \) as defined in the statement of Theorem 2:
\[ C_\delta(\pi, S) - C_{\delta'}(\pi, S) \geq \tau \rightarrow C_P(\pi) - C_{\delta'}(\pi, S) \geq \tau. \]
(42)
Therefore, if \( \tau > 0 \) we have that
\[ P_{S \sim D^n} \left[ C_\delta(\pi, S) - C_{\delta'}(\pi, S) \geq \tau \left| C_P(\pi) > C_P(\pi) \right| \right] \]
\[ \leq P_{S \sim D^n} \left[ C_P(\pi) - C_{\delta'}(\pi, S) > \tau \right] \leq \exp(-2n\tau^2) \]
where the last upper bound follows from Hoeffding’s inequality. Hence, for \( \tau > 0 \), using the above in (40) gives
\[ p_B(S') \leq \exp(-2n\tau^2). \]
(43)
Combining the two cases for \( \tau > 0 \) in (42) and \( \tau = 0 \) in (43) gives us the following implication:
\[ C_P(\pi) \geq C_\delta(\pi, S) \rightarrow p_B(S') \leq \exp(-2n\tau^2). \]
(44)
Now, we expand the left-hand side of (36) using the definition of conditional probability
\[ P_{S,\pi}^{S,n} \left[ p_B(S') \leq \exp(-2n\tau(S)^2) \right] \]
(45)
Equation (46) shown at the bottom of this page.

From (44), we know that \( \{S, \pi)(C_P(\pi) \geq C_\delta(\pi, S)\} \subseteq \{S, \pi) p_B(S') \leq \exp(-2n\tau(S)^2)\} \), therefore,
\[ P_{S,\pi}^{S,n} \left[ p_B(S') \leq \exp(-2n\tau(S)^2) \right] \geq \]
\[ P_{S,\pi}^{S,n} \left[ C_P(\pi) \geq C_\delta(\pi, S) \right] \]
(46)
completing the proof of the claim (36) as well as the lemma.

Proof of Theorem 2: The proof of Theorem 2 follows directly from Lemmas 3 and 4.

C. Proof of Lemma 1

For the readers’ convenience, we restate Lemma 1 here and provide a detailed proof.

Lemma 5: Algorithm 1 returns mutually exclusive outputs, i.e., it returns only one of the three possibilities: OODA, OODB, or WD.
Proof: To prove this property of the detector, we first note that for any test dataset $S'$ the detector can encounter only one of the following distinct cases:

1. $\exp(-2n\tau^2) \leq \alpha_A$ and $\exp(-2n\tau^2) > \alpha_B$;
2. $\exp(-2n\tau^2) > \alpha_A$ and $\exp(-2n\tau^2) \leq \alpha_B$;
3. $\exp(-2n\tau^2) > \alpha_A$ and $\exp(-2n\tau^2 > \alpha_B$;
4. $\exp(-2n\tau^2) \leq \alpha_A$ and $\exp(-2n\tau^2 \leq \alpha_B$.

The first case results in $O(O_A)$, the second case results in $O(O_B)$, and the third case results in $O(W_D)$. In the rest of this proof, we show that the fourth case (i.e., upper bounds on both $p$-values being smaller than their respective significance levels) cannot occur.

Using Theorem 1 and Corollary 1, we can write the following:

$$C_{S'}(\pi) - C_{\delta_A}(\pi, S) = C_S(\pi) - C(S(\pi) - \sqrt{R_A} > 0 \quad (48)$$

$$\iff C_S(\pi) - C_{\delta_A}(\pi, S) < -\sqrt{R_A} \quad (49)$$

$$\iff C_S(\pi) - C_{\delta_A}(\pi, S) < -\sqrt{R_B} < -\sqrt{R_A} - \sqrt{R_B} \leq 0 \quad (50)$$

$$\therefore C_S(\pi) - C_{\delta_A}(\pi, S) - \sqrt{R_B} = C_{\delta_B}(\pi, S) - C_{\delta_A}(\pi, S) < 0. \quad (51)$$

Hence, we have

$$C_S(\pi) - C_{\delta_A}(\pi, S) > 0 \iff C_{\delta_B}(\pi, S) - C_S(\pi) < 0. \quad (52)$$

If $\exp(-2n\tau^2) \leq \alpha_A < 1$ then $\tau > 0$, which further implies that $C_{S'}(\pi) - C_{\delta_A}(\pi, S) > 0$. Hence, from (52), it follows that $C_{\delta_B}(\pi, S) - C_S(\pi) < 0$, which implies that $\tau = 0$ ensuring that $\exp(-2n\tau^2) = 1 > \alpha_B$. Thus, if $\exp(-2n\tau^2) \leq \alpha_A < 1$ then $\exp(-2n\tau^2) = 1$ ensuring that case 4 cannot occur.

D. Proof of Theorem 3

For the readers’ convenience, we restate Theorem 3 here and provide a detailed proof.

**Theorem 3:** Let $D$ be the training distribution, $D'$ be the test distribution, and $P$ be the posterior distribution on the space of policies obtained through the training procedure described in Section IV-A. Let $\delta_A, \delta_B \in (0, 1)$ such that $\delta_A + \delta_B < 1$, $\gamma_A := \sqrt{\frac{\ln(1/\delta_A)}{2n}}$, and $\Delta C_A := C_S(\pi) - \gamma_A - C_{\delta_A}(\pi, S)$. Similarly, let $\delta_B, \delta_B' \in (0, 1)$ such that $\delta_B + \delta_B' < 1$, $\gamma_B := \sqrt{\frac{\ln(1/\delta_B')}{2n}}$, and $\Delta C_B := C_{\delta_B}(\pi, S) - C_S(\pi) - \gamma_B$. Then

$$\begin{align*}
(i) & \quad \mathbb{P}_{(S, \pi, S') \sim (D^m \times P \times D^n)} [C_{D'}(\pi) - C_{D'}(\pi) \\
& \geq \Delta C_A] \geq 1 - \delta_A - \delta_B' \quad (53)
\end{align*}$$

$$\begin{align*}
(ii) & \quad \mathbb{P}_{(S, \pi, S') \sim (D^m \times P \times D^n)} [C_{D'}(\pi) - C_{D'}(\pi) \geq \Delta C_B] \\
& \geq 1 - \delta_B - \delta_B' \quad (54)
\end{align*}$$

To prove Theorem 3, we establish and prove Lemmas 6 and 7.

**Lemma 6:** Let the assumptions of Theorem 3 hold. Then

$$\mathbb{P}_{(S, \pi, S') \sim (D^m \times P \times D^n)} [C_{D'}(\pi) - C_{D'}(\pi) \geq \Delta C_A] \quad (55)$$

Proof: To lower bound the difference between $C_{D'}(\pi)$ and $C_{D'}(\pi)$ with high probability we obtain a lower bound on $C_{D'}(\pi)$, which holds with probability at least $1 - \delta'$ using Hoeffding’s inequality in (56)–(59). Then, we use this bound with the PAC-Bayes bound (3), which holds with probability at least $1 - \delta$ to obtain (55) by following the steps in (60)–(65). Let $\gamma$ be defined as in the statement of the theorem, then, using the independence of $D^m$ from $D^m \times P$, we can write

$$\begin{align*}
\mathbb{P}_{(S, \pi, S') \sim (D^m \times P \times D^n)} [C_{D'}(\pi) \geq C_{S'}(\pi) - \gamma] \\
= \int_{(S, \pi)} \mathbb{P}_{S \sim D^m} [C_{D'}(\pi) \geq C_{S'}(\pi) - \gamma | S, \pi] \quad (56)
\end{align*}$$

For any given $(S, \pi)$, we can apply Hoeffding’s inequality to get

$$\mathbb{P}_{S \sim D^m} [C_{D'}(\pi) \geq C_{S'}(\pi) - \gamma] \geq 1 - \exp(-2n\gamma^2) = 1 - \delta'. \quad (57)$$

Using (57) in (56), we get

$$\mathbb{P}_{(S, \pi, S') \sim (D^m \times P \times D^n)} [C_{D'}(\pi) \geq C_{S'}(\pi) - \gamma] \geq \int_{(S, \pi)} (1 - \delta') d(D^m \times P)(S, \pi) \quad (58)$$

$$= (1 - \delta') \int_{(S, \pi)} d(D^m \times P)(S, \pi) = 1 - \delta'. \quad (59)$$

Now, observe that

$$C_{D'}(\pi) \leq C_{\delta}(\pi, S) \land C_{D'}(\pi) \geq C_{S'}(\pi) - \gamma \rightarrow C_{D'}(\pi) - C_{D'}(\pi) \geq C_{S'}(\pi) - \gamma - C_{\delta}(\pi, S). \quad (60)$$

From the implication (60), it follows that

$$\mathbb{P}_{(S, \pi, S') \sim (D^m \times P \times D^n)} [C_{D'}(\pi) \geq C_{D'}(\pi)]$$

$$\geq \mathbb{P}_{(S, \pi, S') \sim (D^m \times P \times D^n)} [C_{D'}(\pi) \geq C_{S'}(\pi) - \gamma] \land \quad (61)$$

$$\geq \mathbb{P}_{(S, \pi, S') \sim (D^m \times P \times D^n)} [C_{D'}(\pi) \geq C_{S'}(\pi) - \gamma]. \quad (62)$$

Now using the Fréchet inequality $\mathbb{P}[E_1 \land E_2] \geq \mathbb{P}[E_1] + \mathbb{P}[E_2] - 1$ (where $E_1$ and $E_2$ are arbitrary random events) on (62) we obtain

$$\mathbb{P}_{(S, \pi, S') \sim (D^m \times P \times D^n)} [C_{D'}(\pi) \leq C_{\delta}(\pi, S)] \land \quad (63)$$

$$\geq (S, \pi) \sim (D^m \times P) [C_{D'}(\pi) \leq C_{\delta}(\pi, S)]$$

Note that $C_{D'}(\pi)$ and $C_{D'}(\pi)$ implicitly depend on $S$ because the posterior distribution $P$, from which $\pi$ is sampled, is trained on $S$. 

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
where the last inequality follows by using (3) and (59) in (64). Finally, using (65) in (62) completes the proof.

**Lemma 7:** Let the assumptions of Theorem 3 hold. Then
\[\Pr_{(S,\pi') \sim (D^m \times P \times D^m)} [C_{D}(\pi') \geq C_S(\pi') - \gamma] - 1 \geq 1 - \delta - \delta'\] \hspace{1cm} (63)
where \(\delta' = (\delta + \delta_0)\).

**Proof:** Let \(\gamma\) be defined as in the statement of the theorem, then, using the independence of \(D^m\) from \(D^m \times P\), we can write
\[\Pr_{(S,\pi') \sim (D^m \times P \times D^m)} [C_{D}(\pi') \leq C_S(\pi') + \gamma] = \int_{(S,\pi)} \Pr_{S' \sim D^m} [C_{D}(\pi') \leq C_S(\pi') + \gamma | S, \pi] d(D^m \times P)(S, \pi).\] \hspace{1cm} (67)

For any given \((S, \pi)\), we can apply Hoeffding’s inequality to get
\[\Pr_{S' \sim D^m} [C_{D}(\pi') - C_S(\pi) \leq \gamma] \geq 1 - \exp(-2\gamma^2) = 1 - \delta'.\] \hspace{1cm} (68)

Using (68) in (67), we get that
\[\Pr_{(S,\pi') \sim (D^m \times P \times D^m)} [C_{D}(\pi') \leq C_S(\pi') + \gamma] \geq \int_{(S,\pi)} (1 - \delta') d(D^m \times P)(S, \pi)\] \hspace{1cm} (69)
\[= (1 - \delta') \int_{(S,\pi)} d(D^m \times P)(S, \pi) = 1 - \delta'.\] \hspace{1cm} (70)

Now, observe that
\[C_{D}(\pi) \geq C_{\delta}(\pi, S) \land C_{D}(\pi) \leq C_S(\pi') + \gamma\]
\[\therefore C_{D}(\pi') - C_D(\pi) \geq C_{\delta}(\pi, S) - C_S(\pi') - \gamma.\] \hspace{1cm} (71)

Hence
\[\Pr_{(S,\pi') \sim (D^m \times P \times D^m)} [C_{D}(\pi') - C_D(\pi) \geq C_{\delta}(\pi, S) - C_S(\pi') - \gamma] \geq C_{\delta}(\pi, S) - C_S(\pi') - \gamma\] \hspace{1cm} (72)
\[\geq C_{\delta}(\pi, S) - C_S(\pi') - \gamma\] \hspace{1cm} (73)
\[\geq C_{\delta}(\pi, S) - C_S(\pi) + \gamma\] \hspace{1cm} (74)
\[\geq 1 - \delta - \delta'\] \hspace{1cm} (75)

where the first inequality follows from (71), the second inequality follows from Fréchet inequalities, and the third inequality follows from Theorem 1 and (70).

**Proof of Theorem 3:** The proof of Theorem 3 follows directly from Lemmas 6 and 7.

**E. Proof of Lemma 2**

For the readers’ convenience, we restate Lemma 2 here and provide a detailed proof.

**Lemma 8:** Algorithm 2 returns mutually exclusive outputs, i.e., it returns one of three possibilities: OOD\textsuperscript{A}, OOD\textsuperscript{B}, or WD.

**Proof:** To prove this property, we note that the detector can encounter only one of the following four distinct cases involving \(\Delta C_A\) and \(\Delta C_B\):
1) \(\Delta C_A > 0\) and \(\Delta C_B < 0\);
2) \(\Delta C_A \leq 0\) and \(\Delta C_B \geq 0\);
3) \(\Delta C_A \leq 0\) and \(\Delta C_B < 0\);
4) \(\Delta C_A > 0\) and \(\Delta C_B \geq 0\).

The first case results in OOD-adverse, the second case results in OOD-benign, and the third case results in WD. In the rest of this proof we show that the fourth case cannot occur. From the definition of \(\Delta C_A\) we have that
\[\Delta C_A > 0 \iff C_S(\pi) - \gamma_A - C_{\delta_A}(\pi, S) > 0\] \hspace{1cm} (76)
\[\iff C_S(\pi) - C_{\delta_A}(\pi, S) > \sqrt{R_A} > 0\] \hspace{1cm} (77)
\[\iff C_S(\pi) - C_S(\pi') > -\gamma_A - \sqrt{R_A}\] \hspace{1cm} (78)
\[\iff C_S(\pi) - C_S(\pi') > -\gamma_B - \sqrt{R_B}\] \hspace{1cm} (79)
\[\iff C_{\delta_B}(\pi, S) - C_S(\pi') > -\gamma_B - \sqrt{R_B}\] \hspace{1cm} (80)
\[\iff \Delta C_B < 0.\] \hspace{1cm} (81)

We have shown that \(\Delta C_A > 0 \to \Delta C_B < 0\). Thus, case 4 is not possible and the proof is complete.

**F. Training With Backpropogation**

In this section, we describe a method to minimize the upper bound in Theorem 1 using backpropogation. We make use of multivariate Gaussian distributions \(N_\psi\) with diagonal covariance \(\Sigma_{\psi} := \text{diag}(s)\) where \(\psi := (\mu, \log s)\). When training the posterior distribution \(P\), we would like to take gradient steps directly with respect to \(\psi\). However, this would require backpropagation through \(E_{w \sim N_\psi} C_E(\pi_w)\). We follow a similar procedure as in [32] and achieve the desired result of minimizing the upper bound in Inequality (3) using an unbiased estimate of \(E_{w \sim N_\psi} C_E(\pi_w)\)
\[\frac{1}{k} \sum_{i=1}^{k} C_E(\pi_{w_i}), \quad w_i \sim N_\psi \forall i \in \{1, 2, \ldots, k\}.\] \hspace{1cm} (82)

The resulting approach is presented in Algorithm 3. Note that the algorithm must be deterministic in order to maintain the assumptions of Theorem 1. We achieve this by training with a
Algorithm 3: PAC-Bayes Bound Minimization via Backpropagation.

Input: Fixed prior distribution $\mathcal{N}_{\psi_0}$ over policies, fixed seed for random number generation
Input: Training dataset $\mathcal{S}$, learning rate $\gamma$
Output: Optimized $\psi^*$

while not converged do
    Sample $w_i \sim \mathcal{N}_\psi \forall i \in \{1, 2, \ldots, k\}$
    $B \leftarrow \frac{1}{mk} \sum_{E \in \mathcal{S}} \sum_{j=1}^{k} C_E(\pi_{w_j}) + \sqrt{R}$
    $\psi \leftarrow \psi - \gamma \nabla_{\psi} B$
end while

fixed seed for generating random numbers. In addition, note that the backpropagation requires a gradient taken through $D_2(\mathcal{N}_\psi \| \mathcal{N}_{\psi_0})$. We make use of the analytical form for the Réény divergence between two multivariate Gaussian distributions, presented in [56], in order to tractably compute the gradients

$$D_2(\mathcal{N}_\psi \| \mathcal{N}_{\psi_0}) = D_2(\mathcal{N}(\mu, \Sigma) \| \mathcal{N}(\mu_0, \Sigma_0))$$

$$= (\mu - \mu_0)^T \Sigma_2 (\mu - \mu_0) - \frac{1}{2} \ln \frac{\left| \Sigma_2 \right|}{\left| \Sigma_0 \right|}$$

where $\Sigma_2 = 2\Sigma_{s_0} - \Sigma_s$. We also note that there is a restriction on how far the posterior’s variance can drift from the prior. The following expression must be satisfied for $D_2(\mathcal{N}_\psi \| \mathcal{N}_{\psi_0})$ to be finite [56]

$$2\Sigma_2^{-1} - \Sigma_0^{-1} \succ 0.$$ 

(84)

In practice, we project any problematic variances into the range of allowable variances.

After training, since we have used the pointwise PAC-Bayes bound in Theorem 1, we compute the upper bound with a single $w \sim \mathcal{N}_\psi$ in contrast to traditional PAC-Bayes bounds. Thus, the resulting policy $\pi_w$ is deterministic and applicable in a broad range of settings, including ones which require a pretrained network. The resulting policy carries a PAC-Bayes guarantee.

G. Training With ES

To train robot control policies in settings where backpropagation is not feasible (e.g., presence of a “closed box” in the form of a simulator or robot hardware in the forward pass), we use ES that is a class of closed box optimizers [50]. ES addresses this challenge by estimating the gradient via a Monte Carlo estimator

$$\nabla_{\psi} C_S(\mathcal{N}_{\psi}) := \frac{1}{m} \sum_{E \in \mathcal{S}} \nabla_{\psi} \mathbb{E}_{w \sim \mathcal{N}_\psi} \left[ C_E(\pi_w) \right]$$

$$= \frac{1}{m} \sum_{E \in \mathcal{S}} \mathbb{E}_{w \sim \mathcal{N}_\psi} \left[ C_E(\pi_w) \nabla_{\psi} \ln \mathcal{N}_\psi(w) \right].$$

(85)

Although we can compute the gradient of the regularizer analytically (as mentioned in Appendix F), using different methods to estimate the gradient of the empirical cost (ES) and the gradient of the regularizer (analytically) results in poor convergence. To alleviate this, we estimate the regularizer’s gradient using ES as well by leveraging the expectation form of Réény divergence in Theorem 1. This takes the following form:

$$\nabla_{\psi} (C_S(\mathcal{N}_\psi) + \sqrt{R}) = \frac{1}{m} \sum_{E \in \mathcal{S}} \mathbb{E}_{w \sim \mathcal{N}_\psi} \left[ C_E(\pi_w) \right]$$

$$\left[ \frac{e^{\nabla_{\psi} \ln \mathcal{N}_\psi(w)}}{C_E(\pi_w)} - D_2(\mathcal{N}_\psi \| \mathcal{N}_{\psi_0}) \right] \nabla_{\psi} \ln \mathcal{N}_\psi(w).$$

(86)

1) Derivation of (86): To derive (86), note that

$$\nabla_{\psi} (C_S(\mathcal{N}_\psi) + \sqrt{R}) = \nabla_{\psi} C_S(\mathcal{N}_\psi) + \nabla_{\psi} \sqrt{R}$$

$$= \nabla_{\psi} C_S(\mathcal{N}_\psi) + \frac{1}{2\sqrt{R}} \nabla_{\psi} R.$$  

(87)

From (85), we know the gradient for $\nabla_{\psi} C_S(\mathcal{N}_\psi)$. In the rest of this derivation, therefore, we will focus on the computing the gradient of the second term.

Note that the Réény divergence for multivariate Gaussian distributions can be written as

$$D_2(\mathcal{N}_\psi \| \mathcal{N}_{\psi_0}) = \ln \left( \mathbb{E}_{w \sim \mathcal{N}_\psi} \left[ \left( \frac{\mathcal{N}_\psi(w)}{\mathcal{N}_{\psi_0}(w)} \right)^2 \right] \right).$$

(88)

Let

$$\eta := \mathbb{E}_{w \sim \mathcal{N}_\psi} \left[ \left( \frac{\mathcal{N}_\psi(w)}{\mathcal{N}_{\psi_0}(w)} \right)^2 \right].$$

(89)

then, using (88), we have that

$$D_2(\mathcal{N}_\psi \| \mathcal{N}_{\psi_0}) = \ln \eta$$

(90)

which allows us to express $R$ as

$$R = \frac{\ln \eta + \ln(2\sqrt{m}/(\delta/2)^3)}{2m}.$$  

(91)

Hence

$$\nabla_{\psi} R = \frac{1}{2m} \nabla_{\psi} \eta = \frac{e^{-D_2(\mathcal{N}_\psi \| \mathcal{N}_{\psi_0})}}{2m} \nabla_{\psi} \eta$$

(92)

where the last equality follows from (90).

For computing the gradient using ES, we require the cost to be an expectation over the posterior, however, $\eta$ is an expectation on the prior. To address this we perform a change of measure, which gives us the following:

$$\eta = \mathbb{E}_{w \sim \mathcal{N}_\psi} \left[ \left( \frac{\mathcal{N}_\psi(w)}{\mathcal{N}_{\psi_0}(w)} \right)^2 \right] = \mathbb{E}_{w \sim \mathcal{N}_\psi} \left[ \frac{\mathcal{N}_\psi(w)}{\mathcal{N}_{\psi_0}(w)} \nabla_{\psi} \ln \mathcal{N}_\psi(w) \right].$$

(93)

Using (93) in (92) gives us

$$\nabla_{\psi} R = \frac{e^{-D_2(\mathcal{N}_\psi \| \mathcal{N}_{\psi_0})}}{2m} \nabla_{\psi} \mathbb{E}_{w \sim \mathcal{N}_\psi} \left[ \frac{\mathcal{N}_\psi(w)}{\mathcal{N}_{\psi_0}(w)} \nabla_{\psi} \ln \mathcal{N}_\psi(w) \right].$$

(94)
Algorithm 4: PAC-Bayes Bound Minimization via ES.

Input: Fixed prior distribution $\mathcal{N}_{\psi_0}$ over policies, fixed seed for random number generation

Input: Training dataset $S$, learning rate $\gamma$

Output: Optimized $\psi^*$

while not converged do

Sample $w_i \sim \mathcal{N}_\psi \forall i \in \{1, 2, \ldots, k\}$

$\text{grad} \leftarrow \frac{1}{mk} \sum_{E \in S} \sum_{i=1}^k \hat{C}_E(w_i)$

$\psi \leftarrow \psi - \gamma \cdot \text{grad}$

end while

Using (94) and (85) in (87) and combining the expectation terms gives

$$\nabla_{\psi}(C_S(\mathcal{N}_\psi) + \sqrt{R}) = \frac{1}{m} \sum_{E \in S} \mathbb{E}_{w \sim \mathcal{N}_\psi} \left[ C_E(\pi_w) + \frac{e^{-D_2(\mathcal{N}_\psi(w) / \mathcal{N}_{\psi_0}(w))}}{4mR} \mathcal{N}_\psi(w) \nabla_{\psi} \ln \mathcal{N}_\psi(w) \right].$$

(95)

Finally, we note that the dimensionality $d$ of $w$ can be large, in which case the term $\mathcal{N}_\psi(w) / \mathcal{N}_{\psi_0}(w)$ is numerically unstable because it involves the product of $d$ terms. Hence, we express $\mathcal{N}_\psi(w) / \mathcal{N}_{\psi_0}(w)$ as $e^{\ln(\mathcal{N}_\psi(w) / \mathcal{N}_{\psi_0}(w))}$, which gives us (86) as the final form of the gradient.

2) Training Algorithm: The gradient of the PAC-Bayes upper bound is estimated from (86). Since Theorem 1 requires the training algorithm to be deterministic, we train with a fixed seed. The pseudocode for our training is provided in Algorithm 4. After training, a single $w$ is drawn from $\mathcal{N}_\psi$, which corresponds to a policy $\pi_w$, and the derandomized PAC-Bayes bound is computed for this policy.

H. Additional Experimental Details and Results

1) Robotic Grasping: Training platform: Training was performed on a Lambda Blade server with 2x Intel Xeon Gold 5220R (96 CPU threads) and 768 GB RAM.

Control policy architecture: The control policy architecture is shown in Fig. 8. The weights of the DNN are from [47] and the training is warm-started with the posterior in [47, Appendix A5.1].

Distributions on the initial position of the mugs: For all datasets, mugs are placed upright on the table with random yaw orientations sampled from the uniform distribution $U([-\pi\text{rad}, \pi\text{rad}])$. The distributions listed on the mug’s placement were used to generate the plot in Figs. 2(b) and 3(b). Note that distribution (6) enumerated below was not used for the hardware experiments as it was outside the visual field of the camera mounted [for the hardware setup see Fig. 3(a)].

1) $U([0.45m, 0.55m] \times [-0.05m, 0.05m])$ (training distribution).
2) $U([0.40m, 0.60m] \times [-0.10m, 0.10m])$.
3) $U([0.35m, 0.65m] \times [-0.15m, 0.15m])$.
4) $U([0.30m, 0.70m] \times [-0.20m, 0.20m])$.
5) $U([0.25m, 0.75m] \times [-0.25m, 0.25m])$.
6) $U([0.20m, 0.80m] \times [-0.30m, 0.30m])$.

Raw data from hardware experiments: The hardware experiments included a sample of 10 mugs. We report which mugs we failed to grasp (incurring a cost of 1) in Table I, across the distributions of their initial locations. This was then used to compute $\Delta C_A$ and $p_A$ plotted in Fig. 3(b). For the experiment with bowls, we failed to grasp all bowls except for bowl 8 and 10, and successfully grasped all mugs except for mug 8. This was used to plot Fig. 3(c).

2) Vision-Based Obstacle Avoidance With a Drone: The approximate $C_D(\pi)$ (estimated with 50 000 held-out environments) is 0.149; PAC-Bayes, thus, provides a strong bound.

Environment generation: Training environments have 9 obstacles and have at least one gap, which is wide enough to navigate through. We generate environments by randomly placing a set of cylindrical obstacles whose locations are sampled from the uniform distribution $U([4.5m, 7m] \times [-3.5m, 3.5m])$ relative to the drone’s starting point.

Training the prior: Training takes place completely in simulation. To allow for accurate sim-to-real transfer, the motion primitives are recorded trajectories of open-loop control inputs for the Parrot Swing hardware platform. We record multiple rollouts of each open-loop control policy. In simulation, when the policy selects a motion primitive, we randomly select one of the corresponding recorded trajectories to run. We train the prior $\mathcal{N}_{\psi_0}$ over policies by transforming the problem into a supervised learning setting. For each of 10000 training environment in $S$ the policy receives a depth map. Leveraging the simulation,
we simulate each primitive (sampled uniformly from the set of recorded trajectories for that primitive) through each environment. We generate a label for each depth map by recording the minimum distance to an obstacle achieved by each of the primitives and passing the vector of distances through a softmax transformation. Note that even in simulation, we do not assume knowledge of the exact location of obstacles and record the closest distance as viewed by the robot’s 120° field of view depth sensor. These depth maps and softmax labels can then be used for training the prior over policies in a supervised learning setting. We use the cross-entropy loss to train. The result is a policy trained to assign larger values to motion primitives, which achieve a larger distance from obstacles.

Training platform: Training was performed on a desktop computer with an Intel i7–8700k CPU (12 CPU threads) and an NVIDIA Titan Xp GPU with 32GB RAM.

Baselines for OOD detection: Let neural network policy \( \pi_w \) output vector \( \hat{y} = [\hat{y}_1, \ldots, \hat{y}_k] \) of length \( k \). The MSP [4] is given by

\[
\max_{i \in [1, \ldots, k]} \sum_{j=1}^{k} e^{\hat{y}_i} - e^{\hat{y}_j} \tag{96}
\]

and MaxLogit [6] is given by

\[
\max_{i \in [1, \ldots, k]} \log \left( \frac{\hat{y}_i}{1 - \hat{y}_i} \right). \tag{97}
\]

Note that MaxLogit does not require \( \hat{y} \) to be a probability distribution.

**Numerical validation of Theorem 3**: We numerically validate our confidence bound in Fig. 9. We plot:

1) the difference \( C_D^P(\pi) - C_D^F(\pi) \) (estimated via exhaustive sampling of environments);
2) the maximum computed lower bound on \( C_D^P(\pi) - C_D^F(\pi) \) (computed using a confidence level of 0.9) over 500000 datasets \( S' \);
3) the 90th percentile value of the bound over the 100000 datasets.

As guaranteed by Theorem 3, the bound is valid greater than 90% of the time.

**References**

[1] A. Farid, S. Veer, and A. Majumdar, “Task-driven out-of-distribution detection with statistical guarantees for robot learning,” in *Proc. Conf. Robot. Learn.*, 2021, pp. 970–980.

[2] M. Basseville, “Detecting changes in signals and systems—A survey,” *Automatica*, vol. 24, pp. 309–326, 1988.

[3] L. Ruff et al., “A unifying review of deep and shallow anomaly detection,” *Proc. IEEE*, vol. 109, no. 5, pp. 756–795, May 2021.

[4] D. Hendrycks and K. Gimpel, “A baseline for detecting misclassified and out-of-distribution examples in neural networks,” in *Proc. Int. Conf. Learn. Representations*, 2017.

[5] S. Liang, Y. Li, and R. Srikant, “Enhancing the reliability of out-of-distribution image detection in neural networks,” in *Proc. Int. Conf. Learn. Representations*, 2018.

[6] D. Hendrycks, S. Basart, M. Mazeika, M. Motalabjahi, J. Steinhardt, and D. Song, “Scaling out-of-distribution detection for real-world settings,” in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 8759–8773.

[7] T. DeVries and G. W. Taylor, “Learning confidence for out-of-distribution detection in neural networks,” 2018, arXiv:1802.04865.

[8] K. Lee, H. Lee, K. Lee, and J. Shin, “Training confidence-calibrated classifiers for detecting out-of-distribution samples,” in *Proc. Int. Conf. Learn. Representations*, 2018.

[9] D. Hendrycks, M. Mazeika, and T. G. Dietterich, “Deep anomaly detection with outlier exposure,” in *Proc. Int. Conf. Learn. Representations*, 2019.

[10] E. Zisselman and A. Tamar, “Deep residual flow for out of distribution detection,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., 2020, pp. 13994–14003.

[11] J. Chen, Y. Li, X. Wu, Y. Liang, and S. Jha, “Robust out-of-distribution detection for neural networks,” 2020, arXiv:2003.09711.

[12] K. Lee, K. Lee, H. Lee, and J. Shin, “A simple unified framework for detecting out-of-distribution samples and adversarial attacks,” in *Proc. Conf. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7167–7177.

[13] J. Bitterwolf, A. Meinke, and M. Hein, “Certifiably adversarially robust detection of out-of-distribution data,” in *Proc. Conf. Adv. Neural Inf. Process. Syst.*, 2020, pp. 16085–16095.

[14] A. Meinke and M. Hein, “Towards neural networks that provably know when they don’t know,” in *Proc. Int. Conf. Learn. Representations*, 2020.

[15] A. Meinke, J. Bitterwolf, and M. Hein, “Provably robust detection of out-of-distribution data (almost) for free,” 2021, arXiv:2106.04260.

[16] M. A. Siddiqui, A. Fern, T. G. Dietterich, and S. Das, “Finite sample complexity of rare pattern anomaly detection,” in *Proc. Conf. Uncertainty Artif. Intell.*, 2016, pp. 686–695.

[17] S. Liu, R. Garrepalli, T. G. Dietterich, A. Fern, and D. Hendrycks, “Open category detection with PAC guarantees,” in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 3169–3178.

[18] Z. Fang, Y. Li, J. Lu, J. Dong, B. Han, and F. Liu, “On the learnability of out-of-distribution detection,” *J. Mach. Learn. Res.*, vol. 25, pp. 1–83, 2024.

[19] S. Bates, E. Candès, L. Lei, Y. Romano, and M. Sesia, “Testing for outliers with conformal p-values,” *Ann. Statist.*, vol. 51, pp. 149–178, Feb. 2023.

[20] A. Sharma, N. Azizan, and M. Pavone, “Sketching curvature for efficient out-of-distribution detection for deep neural networks,” in *Proc. Conf. Uncertainty Artif. Intell.*, 2021, pp. 1958–1967.

[21] C. Richter and N. Roy, “Safe visual navigation via deep learning and novelty detection,” in *Proc. Robot.: Sci. Syst.*, 2017.

[22] A. Seldemeier, T. Gabor, T. Phan, L. Belzner, and C. Linnhoff-Popien, “Uncertainty-based out-of-distribution classification in deep reinforcement learning,” 2019, arXiv:2003.00496.

[23] A. Seldemeier, R. Müller, S. Illium, and C. Linnhoff-Popien, “Policy entropy for out-of-distribution classification,” in *Proc. Conf. Int. Conf. Learn. Representations*, 2020, pp. 420–431.

[24] F. Cai and X. Koutsoukos, “Real-time out-of-distribution detection in learning-enabled cyber-physical systems,” in *Proc. Int. Conf. Cyber-Phys. Syst.*, 2020, pp. 174–183.

[25] T. Wu and J. Ortiz, “RLAD: Time series anomaly detection through reinforcement learning and active learning,” 2021, arXiv:2104.00543.

[26] I. Greenberg and S. Mannor, “Detecting rewards deterioration in episodic reinforcement learning,” in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 3842–3853.

[27] V. N. Vapnik and A. Y. Chervonenkis, “On the uniform convergence of relative frequencies of events to their probabilities,” *Doklady Akademii Nauk USSR*, vol. 181, no. 4, pp. 781–787, 1968.

[28] S. Shalet-Shwartz and S. Ben-David, *Understanding Machine Learning: From Theory to Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2014.
[29] J. Shawe-Taylor and R. C. Williamson, “A PAC analysis of a Bayesian estimator,” in *Proc. Conf. Comput. Learn. Theory*, 1997, pp. 2–9.
[30] D. A. McAllester, “Some PAC-Bayesian theorems,” *Mach. Learn.*, vol. 37, no. 3, pp. 355–363, 1999.
[31] M. Seeger, “PAC-Bayesian generalisation error bounds for Gaussian process classification,” *J. Mach. Learn. Res.*, vol. 3, pp. 233–269, Oct. 2002.
[32] G. K. Dziugaite and D. M. Roy, “Computing nonvacuous generalization bounds for deep (Stochastic) neural networks with many more parameters than training data,” in *Proc. Conf. Uncertainty Artif. Intell.*, 2017.
[33] J. Langford and J. Shawe-Taylor, “PAC-Bayes & margins,” in *Proc. Conf. Adv. Neural Inf. Process. Syst.*, 2003, pp. 439–446.
[34] P. Germain, A. Lacasse, F. Laviolette, and M. Marchand, “PAC-Bayesian learning of linear classifiers,” in *Proc. Int. Conf. Mach. Learn.*, 2009, pp. 353–360.
[35] P. L. Bartlett, D. J. Foster, and M. J. Telgarsky, “Spectrally-normalized margin bounds for neural networks,” in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2017, pp. 6240–6249.
[36] Y. Jiang, B. Neyshabur, H. Mobahi, D. Krishnan, and S. Bengio, “Fantastic generalization measures and where to find them,” in *Proc. Int. Conf. Learn. Representations*, 2020.
[37] M. Pérez-Ortiz, O. Rivasplata, J. Shawe-Taylor, and C. Szepesvári, “Tighter risk certificates for neural networks,” *J. Mach. Learn. Res.*, vol. 22, pp. 1–40, 2021.
[38] S. Lotfi, M. Finzi, S. Kapoor, A. Potapczynski, M. Goldblum, and A. G. Wilson, “PAC-Bayes compression bounds so tight that they can explain generalization,” in *Proc. Int. Conf. Adv. Neural Inf. Process. Syst.*, 2022, vol. 35, pp. 31459–31473.
[39] O. Catoni, *Statistical Learning Theory and Stochastic Optimization* (École D’été De Probabilités De Saint-Flour), Berlin, Germany: Springer, 2004.
[40] O. Catoni, *PAC-Bayesian Supervised Classification: The Thermodynamics of Statistical Learning* (Lecture Notes—Monograph 56). Institute of Mathematical Statistics, 2007.
[41] D. McAllester, “A PAC-Bayesian tutorial with a dropout bound,” 2013, arXiv:1307.2118.
[42] O. Rivasplata, V. M. Tankasali, and C. Szepesvari, “PAC-Bayes with backprop,” 2019, arXiv:1908.07380.
[43] N. Thiemann, C. Igel, O. Wintenberger, and Y. Seldin, “A strongly quasiconvex PAC-Bayesian bound,” *Mach. Learn. Res.*, vol. 76, pp. 1–26, 2017.
[44] G. K. Dziugaite and D. M. Roy, “Data-dependent PAC-Bayes priors via differential privacy,” in *Proc. Conf. Adv. Neural Inf. Process. Syst.*, 2018.
[45] A. Majumdar and M. Goldstein, “PAC-Bayes control: Synthesizing controllers that provably generalize to novel environments,” in *Proc. Conf. Robot Learn.*, 2018, vol. 87, pp. 293–305.
[46] S. Veer and A. Majumdar, “Probably approximately correct vision-based planning using motion primitives,” in *Proc. Conf. Robot Learn.*, 2020, pp. 1001–1014.
[47] A. Z. Ren, S. Veer, and A. Majumdar, “Generalization guarantees for imitation learning,” in *Proc. Conf. Robot Learn.*, 2020, pp. 1426–1442.
[48] A. Majumdar, A. Farid, and A. Sonor, “PAC-Bayes control: Learning policies that provably generalize to novel environments,” *Int. J. Robot. Res.*, vol. 40, pp. 574–593, 2021.
[49] P. Vialard, P. Germain, A. Habrard, and E. Morvant, “A general framework for the derandomization of PAC-Bayesian bounds,” 2021, arXiv:2102.08649.
[50] D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber, “Natural evolution strategies,” *J. Mach. Learn. Res.*, vol. 15, no. 27, pp. 949–980, 2014.
[51] R. V. Hogg, J. McKean, and A. T. Craig, *Introduction to Mathematical Statistics*. London, U.K.: Pearson Education, 2005.
[52] E. Coumans and Y. Bai, “Pybullet, a python module for physics simulation for games, robotics and machine learning,” 2018. [Online]. Available: http://pybullet.org
[53] A. X. Chang et al., “Shapenet: An information-rich 3D model repository,” 2015, arXiv:1512.03012.
[54] A. Farid, D. Snyder, A. Ren, and A. Majumdar, “Failure prediction with statistical guarantees for vision-based robot control,” in *Proc. Conf. Robot.: Sci. Syst.*, 2022.
[55] A. Maurer, “A note on the PAC Bayesian theorem,” 2004, arXiv:cs.LG/0411099.
[56] M. Gil, F. Alajaji, and T. Linder, “Rényi divergence measures for commonly used univariate continuous distributions,” *Inf. Sci.*, vol. 249, pp. 124–131, 2013.