PAC CONFIDENCE PREDICTIONS FOR DEEP NEURAL NETWORK CLASSIFIERS

Anonymous authors
Paper under double-blind review

ABSTRACT

A key challenge for deploying deep neural networks (DNNs) in safety critical settings is the need to provide rigorous ways to quantify their uncertainty. In this paper, we propose a novel algorithm for constructing predicted classification confidences for DNNs that comes with provable correctness guarantees. Our approach uses Clopper-Pearson confidence intervals for the Binomial distribution in conjunction with the histogram binning approach to calibrated prediction. In addition, we demonstrate how our predicted confidences can be used to enable downstream guarantees in two settings: (i) fast DNN inference, where we demonstrate how to compose a fast but inaccurate DNN with an accurate but slow DNN in a rigorous way to improve performance without sacrificing accuracy, and (ii) safe planning, where we guarantee safety when using a DNN to predict whether a given action is safe based on visual observations. In our experiments, we demonstrate that our approach can be used to provide guarantees for state-of-the-art DNNs.

1 INTRODUCTION

Due to the recent success of machine learning, there has been increasing interest in using predictive models such as deep neural networks (DNNs) in safety-critical settings, such as robotics (e.g., obstacle detection (Ren et al., 2015) and forecasting (Kitani et al., 2012)) and healthcare (e.g., diagnosis (Gulshan et al., 2016; Esteva et al., 2017) and patient care management (Liao et al., 2020)).

One of the key challenges is the need to provide guarantees on the safety or performance of DNNs used in these settings. The potential for failure is inevitable when using DNNs, since they will inevitably make some mistakes in their predictions. Instead, our goal is to design tools for *quantifying* the uncertainty of these models; then, the overall system can estimate and account for the risk inherent in using the predictions made by these models. For instance, a medical decision-making system may want to fall back on a doctor when its prediction is uncertain whether its diagnosis is correct, or a robot may want to stop moving and ask a human for help if it is uncertain to act safely. Uncertainty estimates can also be useful for human decision-makers—*e.g.,* for a doctor to decide whether to trust their intuition over the predicted diagnosis.

While many DNNs provide confidences in their predictions, especially in the classification setting, these are often overconfident. This phenomenon is most likely because DNNs are designed to overfit the training data (*e.g.,* to avoid local minima (Safran & Shamir, 2018)), which results in the predicted probabilities on the training data being very close to one for the correct prediction. Recent work has demonstrated how to *calibrate* the confidences to significantly reduce overconfidence (Guo et al., 2017). Intuitively, these techniques rescale the confidences on a held-out calibration set. Because they are only fitting a small number of parameters, they do not overfit the data as was the case in the original DNN training. However, these techniques do not provide theoretical guarantees on their correctness, which can be necessary in safety-critical settings to guarantee correctness.

We propose a novel algorithm for calibrated prediction in the classification setting that provides theoretical guarantees on the predicted confidences. We focus on *on-distribution* guarantees—*i.e.,* where the test distribution is the same as the training distribution. In this setting, we can build on ideas from statistical learning theory to provide *probably approximately correctness (PAC)* guarantees (Valiant, 1984). Our approach is based on a calibrated prediction technique called histogram binning (Zadrozny & Elkan, 2001), which rescales the confidences by binning them and then rescal-

ing each bin independently. We use Clopper-Pearson bounds on the tails of the binomial distribution to obtain provably correct upper/lower bounds on the predicted probabilities.

Next, we study how it enables theoretical guarantees in two applications. First, we consider the problem of speeding up DNN inference by composing a fast but inaccurate model with a slow but accurate model—*i.e.,* by using the accurate model only if the confidence of the inaccurate one is underconfident (Teerapittayanon et al., 2016). We use our algorithm to obtain guarantees on accuracy of the composed model. Second, for safe planning, we consider using a DNN to predict whether or not a given action (*e.g.,* move forward) is safe (*e.g.,* do not run into obstacles) given an observation (*e.g.,* a camera image). The robot only continues to act if the predicted confidence is above some threshold. We use our algorithm to ensure safety with high probability. Finally, we evaluate the efficacy of our approach in the context of these applications.

Related work. Calibrated prediction (Murphy, 1972; DeGroot & Fienberg, 1983; Platt, 1999) has recently gained attention as a way to improve DNN confidences (Guo et al., 2017). Histogram binning is a non-parametric approach that sorts the data into finitely many bins and rescales the confidences per bin (Zadrozny & Elkan, 2001; 2002; Naeini et al., 2015). However, traditional approaches do not provide theoretical guarantees on the predicted confidences. There has been work on predicting confidence sets (*i.e.,* predict a set of labels instead of a single label) with theoretical guarantees (Park et al., 2020a), but this approach does not provide the confidence of the most likely prediction, as is often desired. There has also been work providing guarantees on the overall calibration error (Kumar et al., 2019), but this approach does not provide per-prediction guarantees.

There has been work speeding up DNN inference (Hinton et al., 2015). One approach is to allow intermediate layers to be dynamically skipped (Teerapittayanon et al., 2016; Figurnov et al., 2017; Wang et al., 2018), which can be thought of as composing multiple models that share a backbone. Unlike our approach, they do not provide guarantees on the accuracy of the composed model.

There has also been work on safe learning-based control (Akametalu et al., 2014; Fisac et al., 2019; Bastani, 2019; Li & Bastani, 2020; Wabersich & Zeilinger, 2018; Alshiekh et al., 2018); however, these approaches are not applicable to perception-based control. The most closely related work is Dean et al. (2019), which handles perception, but they are restricted to known linear dynamics.

2 PAC Confidence Prediction

In this section, we begin by formalizing the PAC confidence coverage prediction problem; then, we describe our algorithm for solving this problem based on histogram binning.

Calibrated prediction. Let $x \in \mathcal{X}$ be an example and $y \in \mathcal{Y}$ be one of a finite set of labels, and let D be a distribution over $\mathcal{X} \times \mathcal{Y}$. A *confidence predictor* is a model $\hat{f} : \mathcal{X} \to \Delta^{\mathcal{Y}}$, where $\Delta^{\mathcal{Y}}$ is the space of probability distributions over labels. In particular, $\hat{f}(x)_y$ is the predicted confidence that the true label for an example x is y. We let $\hat{y} : \mathcal{X} \to \mathcal{Y}$ be the corresponding *label predictor—i.e.,* $\hat{y}(x) = \arg\max_{y \in \mathcal{Y}} \hat{f}(x)_y$—and let $\hat{p} : \mathcal{X} \to \mathbb{R}_{\geq 0}$ be corresponding *top-label confidence predictor—i.e.,* $\hat{p}(x) = \max_{y \in \mathcal{Y}} \hat{f}(x)_y$. While traditional DNN classifiers are confidence predictors, a naively trained DNN is not reliable; recent work has studied heuristics for improving reliability (Guo et al., 2017). In contrast, our goal is to train a confidence predictor that comes with theoretical guarantees.

We first introduce the definition of calibration (DeGroot & Fienberg, 1983; Zadrozny & Elkan, 2002; Park et al., 2020b)—*i.e.,* what we mean for a predicted confidence to be "correct". In many cases, the main quantity of interest is the confidence of the top prediction. Thus, we focus on ensuring that the top-label predicted confidence $\hat{p}(x)$ is calibrated (Guo et al., 2017); our approach can easily be extended to providing guarantees on all confidences predicted using $\hat{f}$. Then, we say a confidence predictor $\hat{f}$ is *well-calibrated* with respect to distribution D if

$$\mathbb{P}_{(x,y)\sim D}\left[y = \hat{y}(x) \mid \hat{p}(x) = t\right] = t \qquad (\forall t \in [0,1]).$$

That is, among all examples x such that the top prediction $\hat{y}(x)$ has predicted confidence $t = \hat{p}(x)$, y is the correct label for exactly a t fraction of these examples.

Using a change of variables (Park et al., 2020b), $\hat{f}$ being well-calibrated is equivalent to

$$\hat{p}(x) = c_{\hat{f}}(x) \coloneqq \mathbb{P}_{(x',y)\sim D}\left[y = \hat{y}(x') \mid \hat{p}(x') = \hat{p}(x)\right] \qquad (\forall x \in \mathcal{X}), \tag{1}$$

where we refer to $c_{\hat{f}}(x)$ as the *true confidence*. Then, the goal of well-calibration is to make $\hat{p}$ equal to the true confidence. Note that $\hat{f}$ appears on both sides of the equation $\hat{p}(x) = c_{\hat{f}}(x)$—implicitly in $\hat{p}$—which is what makes it challenging to satisfy. Indeed, in general, it is unlikely that (1) holds exactly for all x.

Instead, based on the idea of histogram binning (Zadrozny & Elkan, 2001), we consider a variant where we partition the data into a fixed number of bins and then construct confidence coverages separately for each bin. In particular, consider K bins $B_1, \ldots, B_K \subseteq [0, 1]$, where

$$B_k = \begin{cases} [0, b_1] & \text{if } k = 1 \\ (b_{k-1}, b_k] & \text{otherwise.} \end{cases}$$

Here, K and $0 \leq b_1 \leq ... \leq b_{K-1} \leq b_K = 1$ are hyperparameters. Given $\hat{f}$, let $\kappa_{\hat{f}} : \mathcal{X} \to \{1, ..., K\}$ denote the index of the bin that contains $\hat{p}(x)$—*i.e.,* $\hat{p}(x) \in B_{\kappa(x)}$.

Definition 1 We say $\hat{f}$ is *well-calibrated* for a distribution D and bins $B_1, ..., B_K$ if

$$\hat{p}(x) = c_{\hat{f}}(x) \coloneqq \mathbb{P}_{(x',y)\sim D}\left[y = \hat{y}(x') \mid \hat{p}(x') \in B_{\kappa(x)}\right] \qquad (\forall x \in \mathcal{X}). \tag{2}$$

Intuitively, this definition "coarsens" the calibration problem across the bins—*i.e.,* rather than sorting the inputs x into a continuum of "bins" $\hat{p}(x) = t$ for each $t \in [0, 1]$ as in (1), we sort them into a finite number of bins $\hat{p}(x) \in B_k$. It may not be obvious what downstream guarantees can be obtained based on this definition; we provide examples in Sections 3 & 4.

Problem formulation. We formalize the problem of provably correct calibration. The challenge is quantifying over all x in (2). One approach is to provide guarantees in expectation over $x \sim D$ (Kumar et al., 2019); however, this approach does not provide guarantees for individual predictions.

Instead, our goal is to find a *set* of predicted confidences that includes the true confidence with high probability. To this end, a *confidence coverage predictor* is a model $\hat{C} : \mathcal{X} \to \mathbb{R}_{\geq 0} \times \mathbb{R}_{\geq 0}$, where for a prediction $(c_{\text{low}}, c_{\text{up}}) = \hat{C}(x)$, we want to guarantee $c_{\hat{f}}(x) \in [c_{\text{low}}, c_{\text{up}}]$ with high probability. Implicitly, we use $\hat{C}(x) \subseteq \mathbb{R}_{\geq 0}$ to denote the interval $\hat{C}(x) = [c_{\text{low}}, c_{\text{up}}]$, so we want $c_{\hat{f}}(x) \in \hat{C}(x)$.

We are interested in providing theoretical guarantees for an algorithm used to construct confidence coverage predictor $\hat{C}$ given a held-out calibration dataset $Z \subseteq \mathcal{X} \times \mathcal{Y}$. In addition, we assume the algorithm is given a pretrained confidence predictor $\hat{f}$; we make no assumptions about $\hat{f}$—*e.g.,* it can be heuristically calibrated or even uncalibrated. Thus, we consider $\hat{C}$ as depending on Z and $\hat{f}$, which we denote by $\hat{C}(\cdot; \hat{f}, Z)$. Then, we want $\hat{C}$ to satisfy the following guarantee:

Definition 2 Given $\delta \in \mathbb{R}_{>0}$ and $n \in \mathbb{N}$, $\hat{C}$ is *probably approximately correct (PAC)* if

$$\mathbb{P}_{Z\sim D^n, x\sim D}\left[c_{\hat{f}}(x) \in \hat{C}(x; \hat{f}, Z)\right] \geq 1 - \delta, \tag{3}$$

Unlike PAC generalization bounds, our definition of PAC confidence coverage predictors does not include a value ϵ, intended to capture the idea that a prediction is "approximately" correct. In our setting, this idea is implicit by the fact that $\hat{C}$ outputs a range instead of a single value.

Algorithm. We propose a confidence coverage predictor that satisfies the PAC property. The problem of estimating the confidence interval $\hat{C}(x)$ of the binned true confidence $c_{\hat{f}}(x)$ is closely related to the binomial proportion confidence interval estimation problem. In particular, consider a Bernoulli random variable $b \sim B \coloneqq \text{Bernoulli}(\theta)$, where $b = 1$ denotes a success and $b = 0$ denotes a failure, and where θ is unknown. Given a sequence of observations $\vec{b} = (b_1, ..., b_n) \sim B^n$, the goal is to construct an interval $\hat{\theta}(\vec{b}) = [\theta_{\text{min}}, \theta_{\text{max}}] \subseteq \mathbb{R}$ that includes θ with high probability—*i.e.,*

$$\mathbb{P}_{\vec{b}\sim B^n}\left[\theta \in \hat{\Theta}(\vec{b})\right] \geq 1 - \alpha, \tag{4}$$

where $\alpha \in \mathbb{R}$ is a given confidence level. In particular, the Clopper-Pearson interval

$$\hat{\Theta}_{\text{CP}}(\vec{b}; \alpha) = \left[\inf_{\theta}\left\{\theta \;\middle|\; \mathbb{P}\left[S \geq s\right] > \frac{\alpha}{2}\right\},\ \sup_{\theta}\left\{\theta \;\middle|\; \mathbb{P}\left[S \leq s\right] > \frac{\alpha}{2}\right\}\right],$$

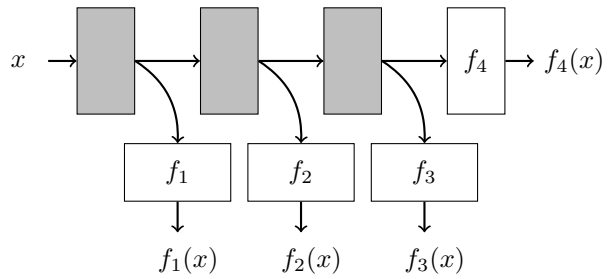


Figure 1: A composed model in a cascading way for $M = 4$.

guarantees (4) (Clopper & Pearson, 1934; Brown et al., 2001), where $s = \sum_{b \in \vec{b}} b$ is the number of observed successes, $n = |\vec{b}|$ is the number of observations, and S is a Binomial random variable $S \sim \text{Binomial}(n, \theta)$. Intuitively, the interval is constructed such that the number of observed success falls in the region with high-probability for any θ in the interval. The following expression is equivalent due to the relationship between the Binomial and Beta distributions (Brown et al., 2001):

$$\hat{\Theta}_{\text{CP}}(\vec{b}; \alpha) = \left[\frac{\alpha}{2} \text{ quantile of Beta}(s, n-s+1),\ 1 - \frac{\alpha}{2} \text{ quantile of Beta}(s+1, n-s)\right].$$

Now, for each of the K bins, we apply $\hat{\Theta}_{\text{CP}}$ with confidence level $\alpha = \delta$ — *i.e.,*

$$\hat{C}(x; \hat{f}, Z, \delta) = \hat{\Theta}_{\text{CP}}\left(W_{\kappa_{\hat{f}}(x)}; \delta\right) \quad \text{where} \quad W_k \coloneqq \left\{ \mathbb{1}(\hat{y}(x) = y) \;\middle|\; (x, y) \in Z \text{ s.t. } \kappa_{\hat{f}}(x) = k \right\}.$$

Here, W_k is the set of observations of successes vs. failures corresponding to the subset of labeled examples $(x, y) \in Z$ such that $\hat{p}(x)$ falls in the bin B_k, where a success is defined to be a correct prediction $\hat{y}(x) = y$. We note that for efficiency, the confidence interval for each of the K bins can be precomputed. Our construction of $\hat{C}$ satisfies the following; see Appendix A for a proof:

Theorem 1 *Our confidence coverage predictor $\hat{C}$ satisfies the PAC property.*

Finally, an important special case is when there is a single bin $B = [0, 1]$—*i.e.,*

$$\hat{C}_0(x; \hat{f}, Z, \delta) = \hat{\Theta}_{\text{CP}}(W; \delta) \qquad \text{where} \qquad W \coloneqq \{\mathbb{1}(\hat{y}(x) = y) \mid (x, y) \in Z\}.$$

Note that $\hat{C}_0$ does not depend on x, so we drop it—*i.e.,* $\hat{C}_0(\hat{f}, Z, \delta) = \hat{\Theta}_{\text{CP}}(W; \delta)$.

3 Application to Fast DNN Inference

A key application of predicted confidences is to perform *model composition* to improve the running time of DNNs without sacrificing accuracy. The idea is to use a fast but inaccurate model when it is confident in its prediction, and switch to an accurate but slow model otherwise (Bolukbasi et al., 2017); we refer to the combination as the *composed model*. To further improve performance, we can have the two models share a backbone—*i.e.,* the fast model shares the first few layers of the slow model (Teerapittayanon et al., 2016). We refer to the decision of whether to skip the slow model as the *exit condition*; then, our goal is to construct confidence thresholds for exit conditions in a way that provides theoretical guarantees on the overall accuracy.

Problem formulation. The early-stopping approach for fast DNN inference can be formalized as a sequence of branching classifiers organized in a cascading way—*i.e.,*

$$\hat{y}_C(x; \gamma_{1:M-1}) = \begin{cases} \hat{y}_i(x) & \text{if } \bigwedge_{i=1}^{m-1} (\hat{p}_i(x) < \gamma_i) \wedge \hat{p}_{m-1}(x) \geq \gamma_{m-1} \ \ (\forall m \in \{1, ..., M-1\}) \\ \hat{y}_M(x) & \text{otherwise,} \end{cases}$$

where M is the number of prediction branches, $\hat{f}_m$ is the confidence predictor, $\hat{y}_m$ and $\hat{p}_m$ are the associated label predictor and top-label confidence predictor, respectively, and d_m is the exit condition of the mth branch (*e.g.,* $d_1(x) \coloneqq \mathbb{1}(\hat{p}_1(x) \geq \gamma_1)$) with threshold $\gamma_m \in [0, 1]$. The $\hat{f}_m$

share a backbone network and are trained in the standard way. Figure 1 illustrates the composed model for $M = 4$; the gray area represents the shared backbone.

Desired error bound. Given $\epsilon \in \mathbb{R}_{>0}$, we use our confidence coverage predictor to choose $\gamma_{1:M-1}$ so the error difference of the composed classifier $\hat{y}_C$ and the full classifier $\hat{y}_M$ is at most ϵ—*i.e.,*

$$p_{\text{err}} := \mathbb{P}_{(x,y)\sim D}\left[\hat{y}_C(x) \neq y\right] - \mathbb{P}_{(x,y)\sim D}\left[\hat{y}_M(x) \neq y\right] \leq \epsilon. \tag{5}$$

To obtain the desired error bound, we exit at the mth branch if $\hat{y}_m$ correctly classifies an example that is also correctly classified by $\hat{y}_M$, allowing for at most ϵ fraction of errors total.

Fast inference. To minimize running time, we prefer to allow higher error rates at the lower branches—*i.e.,* we want to choose γ_m as small as possible at lower branches m.

Algorithm. Our algorithm takes as input the confidence predictors $\hat{f}_m$ (for $m \in \{1, ..., M\}$), the desired relative error $\epsilon \in [0, 1]$, a confidence level $\delta \in [0, 1]$, and a calibration set $Z \subseteq \mathcal{X} \times \mathcal{Y}$, and outputs $\gamma_{1:M-1}$ so that (5) holds with probability at least $1 - \delta$. It iteratively chooses the thresholds from γ_1 to γ_{M-1}; at each step, it chooses γ_m as small as possible subject to $p_{\text{err}} \leq \epsilon$. Note that γ_m implicitly appears in p_{err} in the constraint due to the dependence of $d_m(x)$ on γ_m. The challenge is enforcing the constraint since we cannot evaluate it. To this end, let

$$\begin{aligned} e_m &:= \mathbb{P}_{(x,y)\sim D}\left[\hat{y}_m(x) \neq y \wedge \hat{y}_m(x) \neq \hat{y}_M(x) \wedge d_m(x) = 1\right] \\ e'_m &:= \mathbb{P}_{(x,y)\sim D}\left[\hat{y}_M(x) \neq y \wedge \hat{y}_m(x) \neq \hat{y}_M(x) \wedge d_m(x) = 1\right], \end{aligned}$$

then it is possible to show that $p_{\text{err}} = \sum_{m=1}^{M-1} e_m - e'_m$ (see proof of Theorem 2 in Appendix B). Then, we can compute bounds on e_m and e'_m using the following:

$$\begin{aligned} \mathbb{P}\left[\hat{y}_m(x) = y \mid \hat{y}_m(x) \neq \hat{y}_M(x) \wedge d_m(x) = 1\right] &\in [\underline{c}_m, \bar{c}_m] := \hat{C}_0\left(\hat{f}_m, Z_m, \frac{\delta}{3(M-1)}\right) \\ \mathbb{P}\left[\hat{y}_M(x) = y \mid \hat{y}_m(x) \neq \hat{y}_M(x) \wedge d_m(x) = 1\right] &\in [\underline{c}'_m, \bar{c}'_m] := \hat{C}_0\left(\hat{f}_M, Z_m, \frac{\delta}{3(M-1)}\right) \\ \mathbb{P}\left[\hat{y}_m(x) \neq \hat{y}_M(x) \wedge d_m(x) = 1\right] &\in [\underline{r}_m, \bar{r}_m] := \hat{\Theta}_{\text{CP}}\left(W_m; \frac{\delta}{3(M-1)}\right), \end{aligned}$$

where

$$\begin{aligned} Z_m &:= \{(x, y) \in Z \mid \hat{y}_m(x) \neq \hat{y}_M(x) \wedge d_m(x) = 1\} \\ W_m &:= \{\mathbb{1}(\hat{y}_m(x) \neq \hat{y}_M(x) \wedge d_m(x) = 1) \mid (x, y) \in Z\}. \end{aligned}$$

Thus, we have $e_m \leq \bar{c}_m \bar{r}_m$ and $e'_m \geq \underline{c}'_m \underline{r}_m$, in which case it suffices to solve

$$\gamma_m = \underset{\gamma \in [0,1]}{\arg\min}\ \gamma \qquad \text{subj. to} \qquad \sum_{i=1}^{m} \bar{c}_i \bar{r}_i - \underline{c}'_i \underline{r}_i \leq \epsilon. \tag{6}$$

Here, $\bar{c}_m$, $\bar{r}_m$, $\underline{c}_m$, and $\underline{r}_m$ are implicitly a function of γ, which we can optimize using line search. We have the following guarantee; see Appendix B for a proof:

Theorem 2 *We have $p_{err} \leq \epsilon$ with probability at least $1 - \delta$ over $Z \sim D^N$.*

4 Application to Safe Planning

Robots acting in open world environments must rely on deep learning for tasks such as object recognition—*e.g.,* detect objects in a camera image; providing guarantees on these predictions is critical for safe planning. Safety requires not just that the robot is safe while taking the action, but that it can safely come to a stop afterwards—*e.g.,* that a robot can safely come to a stop before running into a wall. We consider a binary classification DNN trained to predict a probability $\hat{f}(x) \in [0, 1]$ of whether the robot is unsafe in this sense.[1] If $\hat{f}(x) \geq \gamma$ for some threshold $\gamma \in [0, 1]$, then the robot comes to a stop (*e.g.,* to ask a human for help). If the label $\mathbb{1}(\hat{f}(x) \geq \gamma)$ correctly

[1] Since $|\mathcal{Y}| = 2$, $\hat{f}$ can be represented as a map $\hat{f} : \mathcal{X} \to [0, 1]$; the second component is simply $1 - \hat{f}(x)$.

predicts safety, then this policy ensures safety as long as the robot starts from a safe state (Li & Bastani, 2020). We apply our approach to choose γ to ensure safety with high probability.

Problem formulation. Given a performant but potentially unsafe policy $\hat{\pi}$ (*e.g.,* a DNN policy trained to navigate to the goal), our goal is to override $\hat{\pi}$ as needed to ensure safety. Let $x \in \mathcal{X}$ be the states, $\mathcal{X}_{\text{safe}} \subseteq \mathcal{X}$ be the safe states (*i.e.,* our goal is to ensure the robot stays in $\mathcal{X}_{\text{safe}}$), $o \in \mathcal{O}$ be the observations, $u \in \mathcal{U}$ be the actions, $g : \mathcal{X} \times \mathcal{U} \to \mathcal{X}$ be the (deterministic) dynamics, and $h : \mathcal{X} \to \mathcal{O}$ be the observation function. A state x is *recoverable* (denoted $x \in \mathcal{X}_{\text{rec}}$) if the robot can use $\hat{\pi}$ in state x and then safely come to a stop using a *backup policy* π_0 (*e.g.,* braking).

Then, the *shield policy* uses $\hat{\pi}$ if $x \in \mathcal{X}_{\text{rec}}$, and π_0 otherwise (Bastani, 2019). This policy guarantees safety as long as an initial state is recoverable—*i.e.,* $x_0 \in \mathcal{X}_{\text{rec}}$. The challenge is determining whether $x \in \mathcal{X}_{\text{rec}}$. When we observe x, we can use model-based simulation to perform this check. However, in many settings, we only have access to observations—*e.g.,* camera images or LIDAR scans—so this approach does not apply. Instead, we propose to train a DNN to predict recoverability—*i.e.,*

$$\hat{y}(o) = \begin{cases} 1 & \text{if } \hat{f}(o) \geq \gamma \\ 0 & \text{otherwise} \end{cases} \qquad \text{where} \qquad o = h(x),$$

with the goal that $\hat{y}(o) \approx y^*(x) \coloneqq \mathbb{1}(x \notin \mathcal{X}_{\text{rec}})$. We want to choose γ so that π_{shield} ensures safety with high probability—*i.e.,* given $\epsilon \in \mathbb{R}_{>0}$, we have

$$\mathbb{P}_{x_0 \sim D}[x_0, x_1, ... \in \mathcal{X}_{\text{safe}}] \geq 1 - \epsilon,$$

where D is a distribution over initial states $\mathcal{X}_0 \subseteq \mathcal{X}_{\text{rec}}$, $x_{t+1} = g(x_t, \pi_{\text{shield}}(h(x_t)))$, and

$$\pi_{\text{shield}}(o) = \begin{cases} \hat{\pi}(o) & \text{if } \hat{y}(o) = 0 \\ \pi_0(o) & \text{otherwise.} \end{cases}$$

We assume that the rollout terminates either once the robot reaches its goal, or once the robot switches to π_0 and comes to a stop; in particular, the robot never switches from π_0 back to $\hat{\pi}$.

Safety guarantee. Our goal is to ensure that the robot is safe with high probability—*i.e.,*

$$p_{\text{unsafe}} \coloneqq \mathbb{P}_{x_0 \sim D}[\zeta(x_0, \pi_{\text{shield}}) \not\subseteq \mathcal{X}_{\text{safe}}] \leq \epsilon, \tag{7}$$

where $\zeta(x_0, \pi) = (x_0, x_1, ...)$ is the rollout from x_0 generated using π—*i.e.,* $x_{t+1} = g(x_t, \pi(x_t))$.[2]

Success rate. To maximize the success rate (*i.e.,* the rate at which the robot achieves its goal), we need to minimize how often π_{shield} switches to π_0, which corresponds to maximizing γ.

Algorithm. Our algorithm takes as input the confidence predictor $\hat{f}$, desired bound $\epsilon \in \mathbb{R}_{>0}$ on the unsafety probability, confidence level $\delta \in [0, 1]$, calibration set $W \subseteq \mathcal{X}^\infty$ of rollouts $\zeta = (x_0, x_1, ...) \sim D_\zeta$ sampled using $\hat{\pi}$ from $x_0 \sim D$, and calibration set $Z \subseteq \mathcal{O}$ of samples from distribution $\tilde{D}$ described below. We want to maximize γ subject to $p_{\text{unsafe}} \leq \epsilon$, where p_{unsafe} is implicitly a function of γ. However, we cannot evaluate p_{unsafe}, so we need an upper bound.

Now, consider a rollout $\zeta(x_0, \hat{\pi}) = (x_0, x_1, ...)$, where $x_0 \sim D$. Let E_t be the event that the first unsafe state encountered is on step t (*i.e.,* $x_t \notin \mathcal{X}_{\text{safe}}$ but $x_i \in \mathcal{X}_{\text{safe}}$ for all $i < t$), let $p_t = \mathbb{P}_{x_0 \sim D}[E_t]$, and let $\bar{p} = \sum_{t=0}^{\infty} p_t$ be the probability the rollout is unsafe. Consider the distribution $\tilde{D}$ over $\mathcal{O}$ with probability density function $p_{\tilde{D}}(o) = \sum_{t=0}^{\infty} p(o \mid E_t) \cdot p_t / \bar{p}$; in particular, we can draw a sample $o \sim \tilde{D}$ by sampling the first observation $o = h(x)$ in a rollout such that $y^*(x) = 1$ (and rejecting if the entire rollout is safe). Then, we can show (see proof of Theorem 3 in Appendix C):

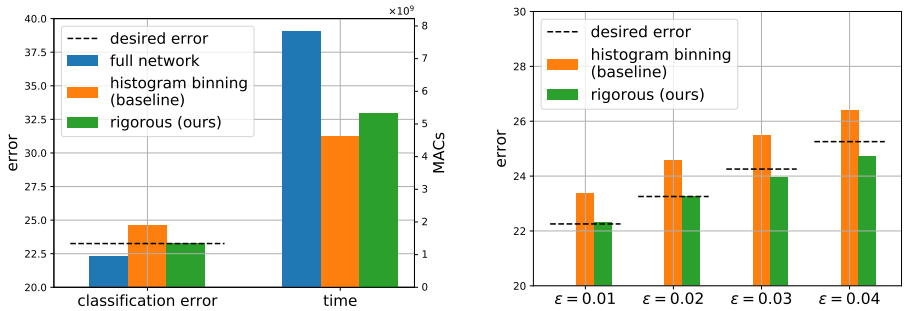
$$p_{\text{unsafe}} \leq \bar{p} \cdot \mathbb{P}_{o \sim \tilde{D}}[\hat{y}(o) = 0] =: \bar{p}_{\text{unsafe}}$$

We use our confidence coverage predictor $\hat{C}_0$ to compute bounds on $\bar{p}_{\text{unsafe}}$—*i.e.,*

$$\mathbb{P}_{o \sim \tilde{D}}[\hat{y}(o) = 0] \in [\underline{r}, \bar{r}] \coloneqq \hat{C}_0\left(\hat{f}, Z', \frac{\delta}{2}\right) \qquad \text{where} \qquad Z' = \{(o, 1) \mid o \in Z\}$$

$$\bar{p} \in [\underline{c}, \bar{c}] \coloneqq \hat{\Theta}_{\text{CP}}\left(W', \frac{\delta}{2}\right) \qquad \text{where} \qquad W' = \{\mathbb{1}(\zeta \not\subseteq \mathcal{X}_{\text{safe}}) \mid \zeta \in W\}.$$

[2] We can handle infinitely long rollouts, but in practice rollouts will be finite (but possibly arbitrarily long).



(a) Comparison to baselines.

(b) Varying desired error rate ϵ.

method	top-1 error (%)	CPU (µs)	GPU (µs)	MACs
original	22.32	135.7	2.04	7.83×10^9
histogram binning (baseline)	24.57	56.0	0.85	4.63×10^9
rigorous (ours)	23.26	63.3	0.97	5.34×10^9

(c) Running time comparison.

Figure 2: Fast DNN inference results; default parameters are $n = 20,000$, $\epsilon = 0.02$, and $\delta = 10^{-2}$.

Then, we have $\bar{p}_{\text{unsafe}} \leq \bar{c} \cdot \bar{r}$, so it suffices to solve the following problem:

$$\gamma := \underset{\gamma' \in [0,1]}{\arg\max}\ \gamma' \qquad \text{subj. to} \qquad \bar{c} \cdot \bar{r} \leq \epsilon \tag{8}$$

Here, $\bar{r}$ is implicitly a function of γ'; thus, we use line search to solve this optimization problem. We have the following safety guarantee, see Appendix C for a proof:

Theorem 3 *We have $p_{unsafe} \leq \epsilon$ with probability over $1 - \delta$ over $W \sim D_\zeta^N$ and $Z \sim \tilde{D}^{N'}$.*

5 Experiments

We demonstrate that how our proposed approach can be used to obtain provable guarantees in our two applications, fast DNN inference and safe planning. Additional results are in Appendix D.

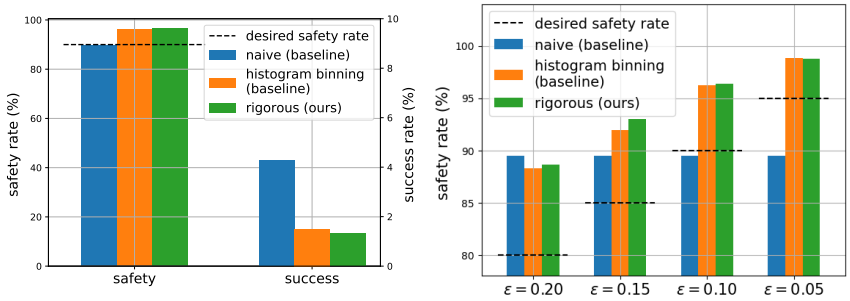
5.1 Fast DNN Inference

Setup. We use the ImageNet dataset (Russakovsky et al., 2015) and ResNet101 (He et al., 2016) for evaluation. We split the ImageNet validation set into $20,000$ calibration images and $10,000$ test images. For the composed model, we use the original ResNet101 as the full network, and add a single exit branch (*i.e.,* $M = 2$) at a quarter of the way from the input layer. We train the newly added branch using the standard procedure for training ResNet101.

Baselines. As a strong baseline, we compare to histogram binning (Zadrozny & Elkan, 2001)—*i.e.,* our approach but using the means of each bin instead of the upper/lower bounds.

Metrics. First, we measure test set top-1 classification error (*i.e.,* $1 -$ accuracy), which we want to guarantee this lower than a desired error (*i.e.,* the error of the full model and desired relative error ϵ). To measure inference time, we consider the average number of multiplication-accumulation operations (MACs) used in inference per example. Note that the MACs are averaged over all examples in the test set since the combined model may use different numbers of MACs for different examples.

Results. The comparison results with the baselines are shown in Figure 2a. The original neural network model is denoted by "full network", our approach (6) by "rigorous", and our baseline by "histogram binning". For each method, we plot the classification error and time in MACs. The desired error upper bound is plotted as a dotted line; the goal is for the classification error to be lower than this line. As can be seen, our method is guaranteed to achieve the desired error, while



(a) Comparison to baselines.

(b) Varying desired safety rate ϵ.

Figure 3: Safe planning results; default parameters are: $N = 20,000$, $\epsilon = 0.1$, $\delta = 10^{-2}$.

improving the inference time by 32% compared to the full model. On the other hand, our baseline improves the inference time but does not satisfy the desired error.

Next, Figure 2b shows the classification error as we vary the desired relative error ϵ. As can be seen, our approach always achieves the desired classification error on the test set, and is often very close (which maximizes speed). However, the baseline often violates the desired error bound.

Finally, the MAC metric is only an approximation of the actual inference time. To complement MACs, we also measure CPU and GPU running time using the PyTorch profiler. In Table 2c, we show the inference times for each method. As can be seen, trends are as before. First, our approach improves running time by 53%, while only reducing classification error by 1%. Second, the baseline is faster than our approach, but does not satisfy the error guarantee.

5.2 Safe Planning

Setup. We evaluate on AI Habitat (Savva et al., 2019), an indoor robot simulator that provides agents with observations $o = h(x)$ that are RGB camera images. The safety constraint is to avoid colliding with obstacles such as furniture in the environment. We use the learned policy $\hat{\pi}$ available with the environment. Then, we train a DNN to predict whether x is recoverable given o, trained using 500 rollouts with a horizon of 100. We calibrate this model on an additional 20,000 rollouts.

Baselines. We compare to two baselines: (i) histogram binning—*i.e.,* our approach but using the means of each bin rather than upper/lower bounds, and (ii) a naive approach of choosing $\gamma = 0.5$.

Metrics. We measure both the safety rate and the success rate; in particular, a rollout is successful if the robot reaches its goal state, and a rollout is safe if it does not reach any unsafe states.

Results. We show results in Figure 3a. The desire safety rate ϵ is shown by the dotted line—*i.e.,* we expect the safety rate to be above this line. As can be seen, our approach achieves the desired safety rate. While it sacrifices success rate, this is because the underlying learned policy $\hat{\pi}$ is frequently unsafe; in particular, it is only safe in about 30% of rollouts. The naive approach fails to satisfy the safety constraint. The baseline performs similarly to our approach. The main benefit of our approach is providing the absolute guarantee on safety, which the baseline does not provide. Thus, in this case, our approach can provide this guarantee while achieving similar performance.

Figure 3b shows the safety rate as we vary the desired safety rate ϵ. Both our approach and the baseline satisfy the desired safety guarantee, whereas the naive approach does not always do so.

6 Conclusion

We have proposed a novel algorithm for calibrated prediction that provides provable correctness guarantees, and demonstrated how our approach can be applied to fast DNN inference and to safe planning. There are many directions for future work—*e.g.,* handling the regression setting and leveraging these techniques in more application domains.

REFERENCES

A. K. Akametalu, J. F. Fisac, J. H. Gillula, S. Kaynama, M. N. Zeilinger, and C. J. Tomlin. Reachability-based safe learning with gaussian processes. In *53rd IEEE Conference on Decision and Control*, pp. 1424–1431, 2014.

Mohammed Alshiekh, R. Bloem, R. Ehlers, Bettina Könighofer, S. Niekum, and U. Topcu. Safe reinforcement learning via shielding. *ArXiv*, abs/1708.08611, 2018.

Osbert Bastani. Safe planning via model predictive shielding, 2019.

Tolga Bolukbasi, Joseph Wang, Ofer Dekel, and Venkatesh Saligrama. Adaptive neural networks for efficient inference. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pp. 527–536. JMLR. org, 2017.

Lawrence D Brown, T Tony Cai, and Anirban DasGupta. Interval estimation for a binomial proportion. *Statistical science*, pp. 101–117, 2001.

Charles J Clopper and Egon S Pearson. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrika*, 26(4):404–413, 1934.

Sarah Dean, Nikolai Matni, Benjamin Recht, and Vickie Ye. Robust guarantees for perception-based control, 2019.

Morris H DeGroot and Stephen E Fienberg. The comparison and evaluation of forecasters. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 32(1-2):12–22, 1983.

Andre Esteva, Brett Kuprel, Roberto A Novoa, Justin Ko, Susan M Swetter, Helen M Blau, and Sebastian Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *nature*, 542(7639):115–118, 2017.

Michael Figurnov, Maxwell D Collins, Yukun Zhu, Li Zhang, Jonathan Huang, Dmitry Vetrov, and Ruslan Salakhutdinov. Spatially adaptive computation time for residual networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1039–1048, 2017.

J. F. Fisac, A. K. Akametalu, M. N. Zeilinger, S. Kaynama, J. Gillula, and C. J. Tomlin. A general safety framework for learning-based control in uncertain robotic systems. *IEEE Transactions on Automatic Control*, 64(7):2737–2752, 2019.

Varun Gulshan, Lily Peng, Marc Coram, Martin C Stumpe, Derek Wu, Arunachalam Narayanaswamy, Subhashini Venugopalan, Kasumi Widner, Tom Madams, Jorge Cuadros, et al. Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *Jama*, 316(22):2402–2410, 2016.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q Weinberger. On calibration of modern neural networks. *arXiv preprint arXiv:1706.04599*, 2017.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Kris M Kitani, Brian D Ziebart, James Andrew Bagnell, and Martial Hebert. Activity forecasting. In *European Conference on Computer Vision*, pp. 201–214. Springer, 2012.

Ananya Kumar, Percy S Liang, and Tengyu Ma. Verified uncertainty calibration. In *Advances in Neural Information Processing Systems*, pp. 3792–3803, 2019.

S. Li and O. Bastani. Robust model predictive shielding for safe reinforcement learning with stochastic dynamics. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7166–7172, 2020.

Shuo Li and Osbert Bastani. Robust model predictive shielding for safe reinforcement learning with stochastic dynamics. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7166–7172. IEEE, 2020.

Peng Liao, Kristjan Greenewald, Predrag Klasnja, and Susan Murphy. Personalized heartsteps: A reinforcement learning algorithm for optimizing physical activity. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, 4(1):1–22, 2020.

Allan H Murphy. Scalar and vector partitions of the probability score: Part i. two-state situation. *Journal of Applied Meteorology*, 11(2):273–282, 1972.

Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using bayesian binning. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

Sangdon Park, Osbert Bastani, Nikolai Matni, and Insup Lee. Pac confidence sets for deep neural networks via calibrated prediction. In *International Conference on Learning Representations*, 2020a. URL `https://openreview.net/forum?id=BJxVI04YvB`.

Sangdon Park, Osbert Bastani, James Weimer, and Insup Lee. Calibrated prediction with covariate shift via unsupervised domain adaptation. In *The 23rd International Conference on Artificial Intelligence and Statistics*, 2020b.

John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 1999.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pp. 91–99, 2015.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252, 2015.

Itay Safran and Ohad Shamir. Spurious local minima are common in two-layer relu neural networks. In *International Conference on Machine Learning*, pp. 4433–4441. PMLR, 2018.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, Devi Parikh, and Dhruv Batra. Habitat: A Platform for Embodied AI Research. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

Surat Teerapittayanon, Bradley McDanel, and Hsiang-Tsung Kung. Branchynet: Fast inference via early exiting from deep neural networks. In *2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 2464–2469. IEEE, 2016.

Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

Kim Wabersich and Melanie Zeilinger. Linear model predictive safety certification for learning-based control. 03 2018. doi: 10.1109/CDC.2018.8619829.

Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. Skipnet: Learning dynamic routing in convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 409–424, 2018.

Bianca Zadrozny and Charles Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *In Proceedings of the Eighteenth International Conference on Machine Learning*. Citeseer, 2001.

Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 694–699. ACM, 2002.

A PROOF OF THEOREM 1

We prove this by exploiting the independence of each bin. Recall that B_k is the kth bin. Then, the following holds:

$$\begin{aligned}
\mathbb{P}\left[c_{\hat{f}}(x) \in \hat{C}\left(x; \hat{f}, Z, \delta\right)\right] &= \mathbb{P}\left[c_{\hat{f}}(x) \in \hat{C}\left(x; \hat{f}, Z, \delta\right) \wedge \bigvee_{k=1}^{K} x \in B_k\right] \\
&= \sum_{k=1}^{K} \mathbb{P}\left[c_{\hat{f}}(x) \in \hat{C}\left(x; \hat{f}, Z, \delta\right) \,\middle|\, x \in B_k\right] \mathbb{P}\left[x \in B_k\right] \\
&\geq \sum_{k=1}^{K} (1-\delta) \mathbb{P}\left[x \in B_k\right] \\
&= 1 - \delta,
\end{aligned}$$

as claimed.

B PROOF OF THEOREM 2

We drop probabilities over $(x, y) \sim D$. First, we decompose $\mathbb{P}\left[\hat{y}_C(x) \neq y\right]$ as follows:

$$\begin{aligned}
\mathbb{P}\left[\hat{y}_C(x) \neq y\right] &= \mathbb{P}\left[\hat{y}_C(x) \neq y \text{ and } (\hat{y}_C(x) = \hat{y}_M(x) \text{ or } \hat{y}_C(x) \neq \hat{y}_M(x))\right] \\
&= \mathbb{P}\left[(\hat{y}_C(x) \neq y \text{ and } \hat{y}_C(x) = \hat{y}_M(x)) \text{ or } (\hat{y}_C(x) \neq y \text{ and } \hat{y}_C(x) \neq \hat{y}_M(x))\right] \\
&= \mathbb{P}\left[\hat{y}_C(x) \neq y \text{ and } \hat{y}_C(x) = \hat{y}_M(x)\right] + \mathbb{P}\left[\hat{y}_C(x) \neq y \text{ and } \hat{y}_C(x) \neq \hat{y}_M(x)\right].
\end{aligned}$$

Similarly, for $\mathbb{P}\left[\hat{y}_M(x) \neq y\right]$, we have:

$$\mathbb{P}\left[\hat{y}_M(x) \neq y\right] = \mathbb{P}\left[\hat{y}_M(x) \neq y \text{ and } \hat{y}_C(x) = \hat{y}_M(x)\right] + \mathbb{P}\left[\hat{y}_M(x) \neq y \text{ and } \hat{y}_C(x) \neq \hat{y}_M(x)\right].$$

Thus, the error difference can be represented as follows:

$$\begin{aligned}
&\mathbb{P}\left[\hat{y}_C(x) \neq y\right] - \mathbb{P}\left[\hat{y}_M(x) \neq y\right] \\
&= \mathbb{P}\left[\hat{y}_C(x) \neq y \text{ and } \hat{y}_C(x) \neq \hat{y}_M(x)\right] - \mathbb{P}\left[\hat{y}_M(x) \neq y \text{ and } \hat{y}_C(x) \neq \hat{y}_M(x)\right].
\end{aligned} \tag{9}$$

To complete the proof, we need to upper bound (9) by ϵ. Define the following events:

$$\begin{aligned}
D_m &\coloneqq \bigwedge_{i=1}^{m-1} (\hat{p}_i(x) < \gamma_i) \wedge \hat{p}_m(x) \geq \gamma_m \qquad (\forall m \in \{1, ..., M-1\}) \\
D_M &\coloneqq \bigwedge_{i=1}^{M-1} (\hat{p}_i(x) < \gamma_i) \\
E_C &\coloneqq \hat{y}_C(x) \neq \hat{y}_M(x) \\
E_m &\coloneqq \hat{y}_m(x) \neq \hat{y}_M(x) \qquad (\forall m \in \{1, ..., M-1\}) \\
F_C &\coloneqq \hat{y}_C(x) \neq y \\
F_m &\coloneqq \hat{y}_C(x) \neq y \qquad (\forall m \in \{1, ..., M-1\}) \\
G &\coloneqq \hat{y}_M(x) \neq y.
\end{aligned}$$

Then, we have:

$$\begin{aligned}
\mathbb{P}\left[\hat{y}_C(x) \neq y \text{ and } \hat{y}_C(x) \neq \hat{y}_M(x)\right] &= \mathbb{P}\left[F_C \wedge E_C\right] \\
&= \mathbb{P}\left[F_C \wedge E_C \wedge \bigvee_{m=1}^{M} D_m\right] \\
&= \mathbb{P}\left[\bigvee_{m=1}^{M} (F_C \wedge E_C \wedge D_m)\right] \\
&= \sum_{m=1}^{M} \mathbb{P}\left[F_C \wedge E_C \wedge D_m\right] \\
&= \sum_{m=1}^{M} \mathbb{P}\left[F_m \wedge E_m \wedge D_m\right] \\
&= \sum_{m=1}^{M} \mathbb{P}\left[F_m \mid E_m \wedge D_m\right] \mathbb{P}\left[E_m \wedge D_m\right].
\end{aligned}$$

Similarly, we have:

$$\mathbb{P}\left[\hat{y}_M(x) \neq y \text{ and } \hat{y}_C(x) \neq \hat{y}_M(x)\right] = \sum_{m=1}^{M} \mathbb{P}\left[G \mid E_m \wedge D_m\right] \cdot \mathbb{P}\left[E_m \wedge D_m\right].$$

Thus, (9) can be rewritten as follows:

$$\begin{aligned}
&\mathbb{P}\left[\hat{y}_C(x) \neq y\right] - \mathbb{P}\left[\hat{y}_M(x) \neq y\right] \\
&= \sum_{m=1}^{M} \left(\mathbb{P}\left[F_m \mid E_m \wedge D_m\right] \cdot \mathbb{P}\left[E_m \wedge D_m\right] - \mathbb{P}\left[G \mid E_m \wedge D_m\right] \cdot \mathbb{P}\left[E_m \wedge D_m\right]\right) \\
&= \sum_{m=1}^{M} (e_m - e'_m) \\
&= \sum_{m=1}^{M-1} (e_m - e'_m) \\
&\leq \epsilon,
\end{aligned}$$

as claimed.

C Proof of Theorem 3

For clarity, we use s to denote "safe" (*i.e.,* $y = 0$) and u to denote "unsafe" (*i.e.,* $y = 1$). Now, note that a rollout $\zeta(x_0, \pi_{\text{shield}}) = (x_0, x_1, ...)$ is only unsafe if (i) at some step t, we have $y^*(x_t) = u$ (*i.e.,* x_t is not recoverable), yet $\hat{y}(o_t) = s$ where $o_t = h(x_t)$ (*i.e.,* $\hat{y}$ predicts x_t is recoverable), and furthermore (ii) for every step $i < t$, $y^*(x_i) = \hat{y}(o_i) = s$—*i.e.,*

$$p_{\text{unsafe}} \leq \mathbb{P}_{x_0 \sim D}\left[\bigwedge_{t=0}^{\infty} \left(\bigwedge_{i=0}^{t-1} y^*(x_i) = s \wedge \hat{y}(o_i) = s\right) \wedge (y^*(x_t) = u \wedge \hat{y}(o_t) = s)\right]. \tag{10}$$

The first parenthetical inside the probability captures (ii), and the captures (i). Condition (i) is captured by the second parenthetical inside the probability; intuitively, it says that $\hat{y}(o_t)$ is a false negative. Condition (ii) is captured by the first parenthetical inside the probability; intuitively, it says that $\hat{y}(o_i)$ is not a false negative for any $i < t$. Next, the event E_t is

$$E_t \coloneqq \left(\bigwedge_{i=0}^{t-1} y^*(x_i) = s\right) \wedge (y^*(x_t) = u).$$

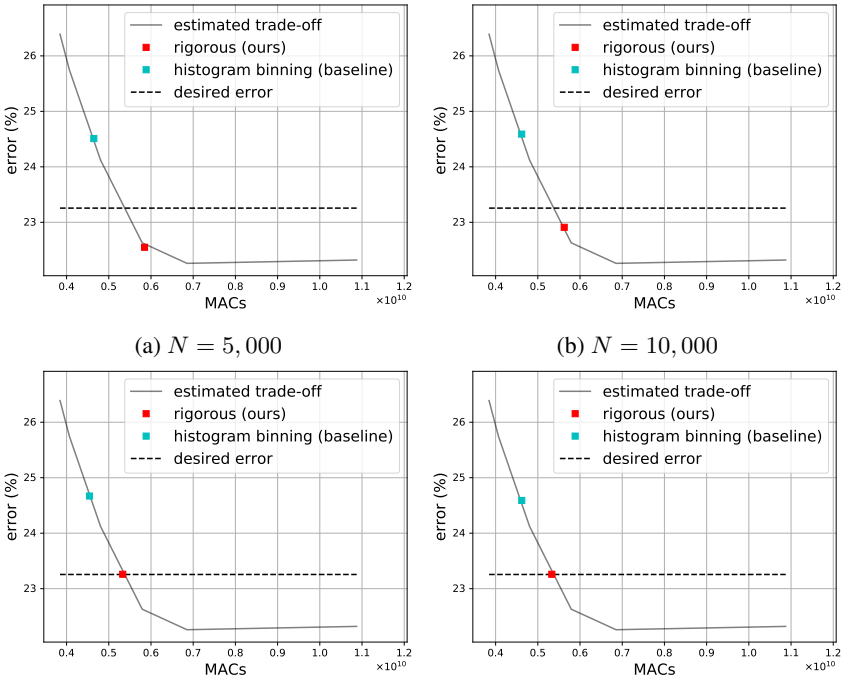
Then, we can upper bound (10) as follows:

$$p_{\text{unsafe}} \leq \mathbb{P}_{x_0 \sim D}\left[\bigwedge_{t=0}^{\infty} E_t \wedge \hat{y}(o_t) = s\right] = \sum_{t=0}^{\infty} \mathbb{P}_{x_0 \sim D}[\hat{y}(o_t) = s \mid E_t] \cdot \mathbb{P}_{x_0 \sim D}[E_t]$$
$$\leq \bar{p} \cdot \mathbb{P}_{o \sim \tilde{D}}[\hat{y}(o) = s].$$

Thus, the constraint in (8) implies $p_{\text{unsafe}} \leq \epsilon$, so the claim follows.

D ADDITIONAL EXPERIMENTS

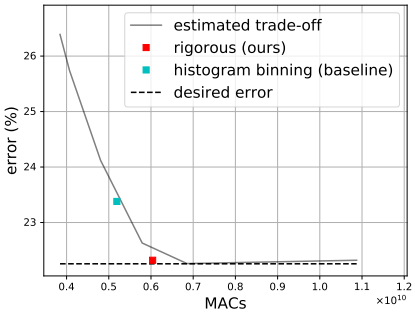
D.1 FAST DNN INFERENCE



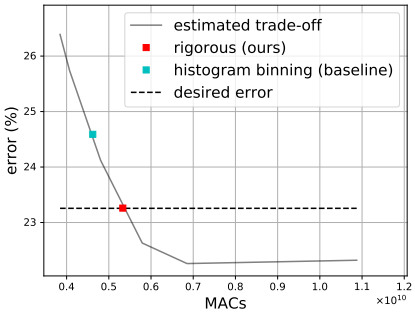
(a) $N = 5,000$ (b) $N = 10,000$

(c) $N = 15,000$ (d) $N = 20,000$

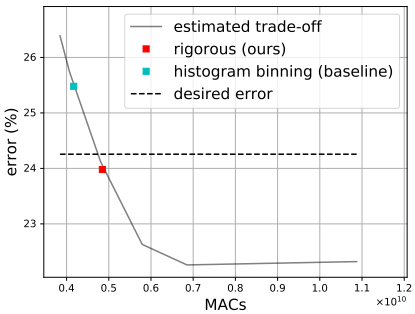
Figure 4: Ablation study for various N ($\epsilon = 0.02$, $\delta = 10^{-2}$)



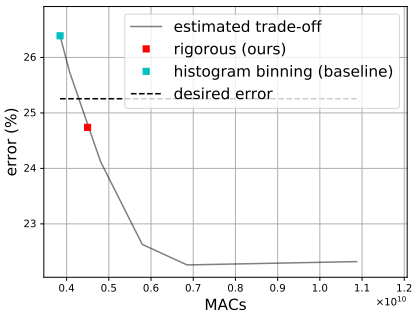
(a) $\epsilon = 0.01$



(b) $\epsilon = 0.02$

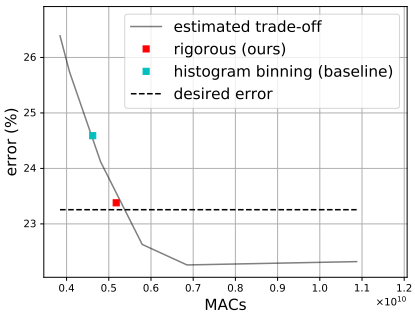


(c) $\epsilon = 0.03$

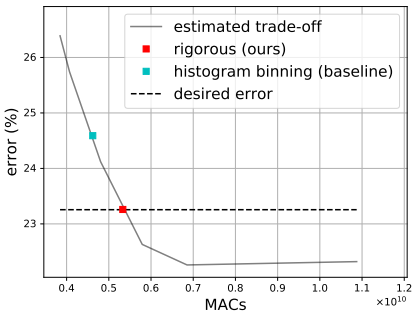


(d) $\epsilon = 0.04$

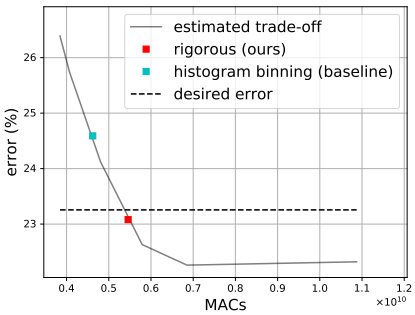
Figure 5: Ablation study for various ϵ ($N = 20,000$, $\delta = 10^{-2}$)



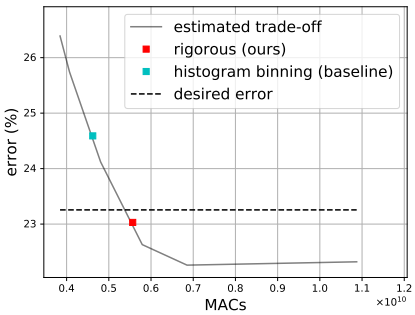
(a) $\delta = 10^{-1}$



(b) $\delta = 10^{-2}$



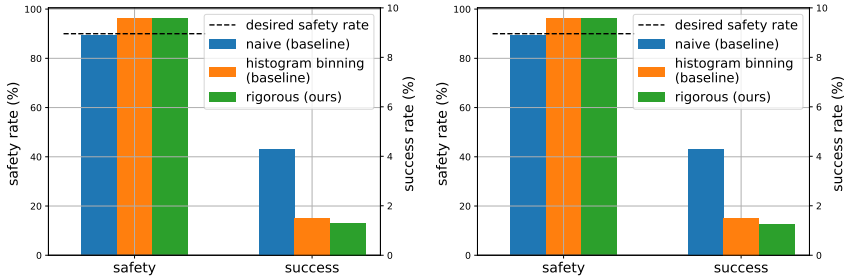
(c) $\delta = 10^{-3}$



(d) $\delta = 10^{-4}$

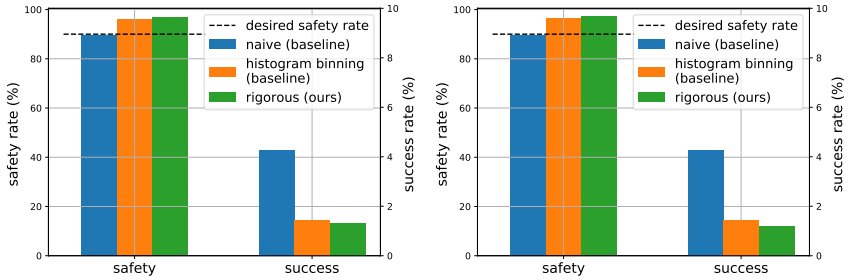
Figure 6: Ablation study for various δ ($N = 20,000$, $\epsilon = 0.02$)

D.2 SAFE PLANNING



(a) $N = 20,000$

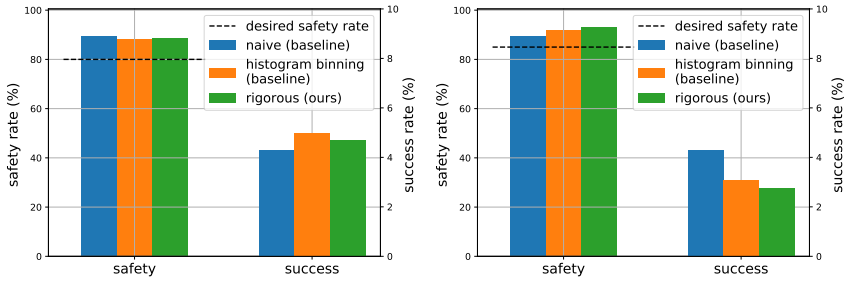
(b) $N = 15,000$



(c) $N = 10,000$

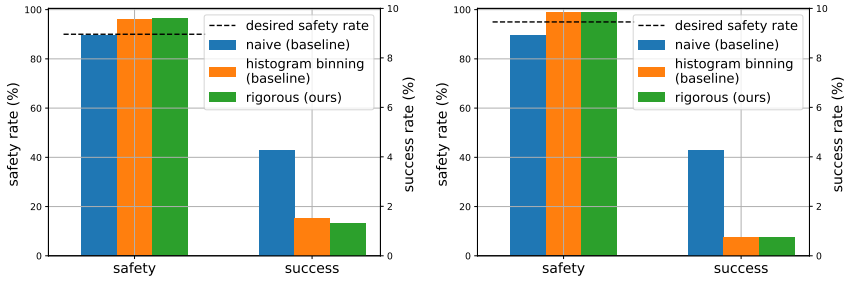
(d) $N = 5,000$

Figure 7: Ablation study for various N ($\epsilon = 0.1$, $\delta = 10^{-2}$)



(a) $\epsilon = 0.2$

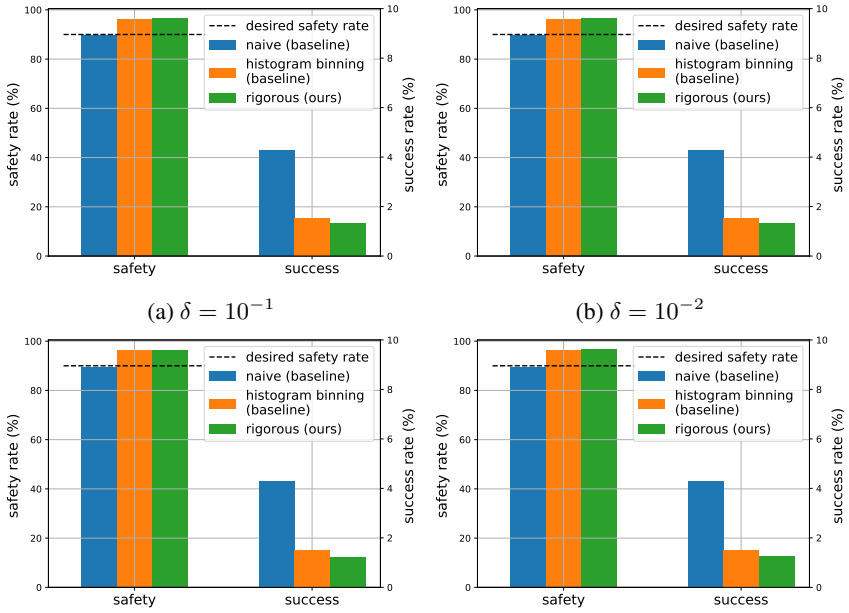
(b) $\epsilon = 0.15$



(c) $\epsilon = 0.1$

(d) $\epsilon = 0.05$

Figure 8: Ablation study for various ϵ ($N = 20,000$, $\delta = 10^{-2}$)



(a) $\delta = 10^{-1}$ (b) $\delta = 10^{-2}$

(c) $\delta = 10^{-3}$ (d) $\delta = 10^{-4}$

Figure 9: Ablation study for various δ ($N = 20,000$, $\epsilon = 0.1$)