Certifiably Robust Neural ODE With Learning-Based Barrier Function

Runing Yang, Ruoxi Jia, Xiangyu Zhang, *Senior Member, IEEE*, and Ming Jin

(Special Section on Data-Driven Analysis and Control)

***Abstract*—Neural Ordinary Differential Equations (ODEs) have gained traction in many applications. While recent studies have focused on empirically increasing the robustness of neural ODEs against natural or adversarial attacks, certified robustness is still lacking. In this letter, we propose a framework for training a neural ODE using barrier functions and demonstrate improved robustness for classification problems. We further provide the *first generalization guarantee of robustness against adversarial attacks* using a wait-and-judge scenario approach.**

***Index Terms*—Neural networks, machine learning, data-driven control.**

I. Introduction

NEURAL ordinary differential equations (NODEs) approximate nonlinear mappings by modeling the dynamics of hidden states by an ODE solver [6]. By extending discrete layerwise architectures to continuous-time dynamical systems, NODEs open up a wealth of applications, enjoy many desirable properties, such as parametric efficiency, constant memory requirements, and invertibility [6], [18], [21], and allow researchers to tap into the vast literature on control theory to achieve certain output requirements [18], [21].

This letter aims to enhance the robustness of NODEs. Robustness is the ability of a model to maintain integrity in the face of input perturbations, either natural or adversarial. The vulnerability (lack of robustness) of deep learning to adversarial attacks is well-studied [7], [17], [19], [22]. Recent study [12] indicates that, unlike conventional deep learning, NODEs have a certain level of inherent robustness due to the non-intersecting property of integral curves (see also [13] for another interesting view). Existing work to improve robustness can be classified (with overlaps) below.

1) *Regularization methods* inject random noise into each layer [16], randomly sample the end time of the ODE during training [16], or introduce additional penalty terms on weights [9];
2) *Control-theoretic approaches* design the training loss to promote certain properties of the dynamical system, such as steady-state constraint [12], contraction [21], reachability [11], and Lyapunov stability [15], [18].

While regularization methods are inspired by deep learning experiences (dropout, stochastic depth, and random smoothing in the case of [9], [16]), control-theoretic approaches directly promote certain aspects of the underlying dynamical systems—this is also the approach we take in this letter.

Our key insight is that not all input-perturbation-induced output changes are adverse or need to be penalized; robustness depends only on those points *near the boundary* that are most *susceptible* to adversarial perturbations *along certain directions* (see Fig. 1 for an illustration, detailed in Section V-A). Hence, instead of aiming to make the output for each data point to be a steady-state [12] or within a neighborhood of some Lyapunov-stable equilibrium [15], [18], we focus only on *vulnerable points* and penalize output changes along *adverse directions*. Additionally, we realize that for classification problems with a complex underlying mapping, outputs are *expected* to be different despite a small difference in inputs, which is not the case for robust/contractive systems [21]. To bring these insights to bear, we develop a framework that learns NODE parameters together with a robustness certificate based on control-barrier functions [1]. Our key contributions include.

- Development of a framework for training robust NODEs using learning-based barrier functions;

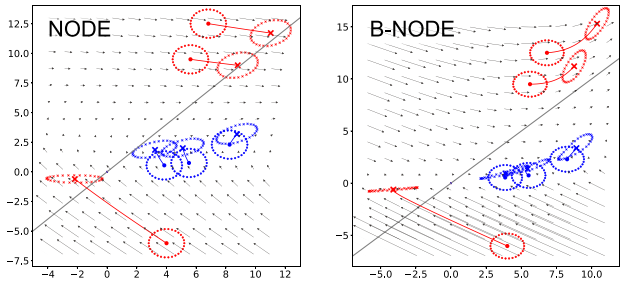

Fig. 1. Comparison between vanilla NODE and B-NODE (ours). The quiver plot shows the flows of NODEs, the propagation of the training data (solid lines), and how perturbed inputs (solid dots) are mapped to perturbed outputs (crosses) for both classes (red and blue). Grey line: decision boundary.

Manuscript received 11 December 2022; revised 20 February 2023; accepted 15 March 2023. Date of publication 7 April 2023; date of current version 24 May 2023. This work was supported in part by the C3.ai Digital Transformation Institute; in part by the U.S. Department of Energy (DOE); and in part by NSF EPCN under Grant 2034137. Recommended by Senior Editor S. Tarbouriech. *(Corresponding author: Ming Jin.)*

Runing Yang, Ruoxi Jia, and Ming Jin are with the Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA 24061 USA (e-mail: jinming@vt.edu).

Xiangyu Zhang is with the Computational Science Center, National Renewable Energy Laboratory, Golden, CO 80401 USA.

Digital Object Identifier 10.1109/LCSYS.2023.3265397

- Establishment of the *first* generalization guarantee of robustness using a wait-and-judge approach;
- Demonstration of robustness against natural and adversarial perturbations for a range of benchmarks.

Contextualization of contributions. In comparison to studies that enhance the empirical robustness of NODEs [9], [12], [13], [15], [16], [18], [21], our study provides a rigorous analysis of the robustness guarantee. Unlike recent works on certified robustness that only analyze a *given* point [11], [14], we extend the certified robustness analysis to an *unseen in-distribution* point (Theorem 2). This difference can be interpreted as an extension from training performance to generalization performance, as defined in Def. 1. In our study, we pursue an approach based on scenario optimization [3], [4], while extending existing works in scenario optimization to the case of agnostic PAC (another contemporary work that also achieves this [5]). We further validate the tightness of our theoretical bounds and demonstrate nontrivial lower bounds.

Next, we discuss preliminaries in Section II. Section III presents the robustness certificate based on barrier functions and the proposed robust training procedure. Theoretical analysis of robustness is performed in Section IV. Numerical results are discussed in Section V and the conclusion is drawn in Section VI. For the proof and additional experiments, please refer to [20].

Notations: We represent $[n] = \{1, \ldots, n\}$, $[k, n] = \{k, \ldots, n\}$, $[a]_+ = \max(0, a)$, and $\mathbb{1}(\cdot)$ as the indicator function (outputs 1 if the argument is true and 0 otherwise).

II. Preliminaries

A. Neural ODEs

As a nonlinear mapping, a NODE specifies the relation between input $z(0) = \phi(x)$ and output $y = \psi(z(T), x)$ by the following differential equation:

$$\dot{z}(t) = f_\theta(z(t), t), \tag{1}$$

where x is the input data, $z(t)$ is the ODE state (hidden layer values) at time t, y is the output, and f_θ is a nonlinear function parameterized by θ. The initial condition $z(0)$ is derived from the input by a feature mapping ϕ (i.e., input layer) and the output y is obtained from $z(T)$ through a function ψ (i.e., output layer). We slightly overload the notation $z(T, x)$ to make explicit the dependence of $z(T)$ on the initial state x. The input layer can be learned from scratch or fine-tuned [12], [15]. The output layer is typically a simple function, such as softmax for classification. We use a time-invariant NODE, as is common practice (e.g., [12]), denoted as $f_\theta(z(t))$ throughout.

Classification: For classification problems, the final state of NODE undergoes an affine transformation to produce an embedding $Wz(T) \in \mathbb{R}^K$, where $W \in \mathbb{R}^{K \times d}$ is the parameters to be learned. The embedding can be passed through either an argmax function $\arg\max_k [Wz(T)]_k$ or a softmax function to obtain a probability vector. The output is denoted as $y = \psi(z(T), x) = g_W(x, f_\theta)$. Unless otherwise specified, we denote θ to also include the parameter W and omit the dependency of g_W on W for notational simplicity.

Forward/backward process: Forward pass can be performed using standard ODE solvers (such as Euler's method or Runge–Kutta method) to obtain the final state $z(T)$. For training, we need to calculate the derivative of the loss function with respect to NODE parameters, which can be obtained by either auto-differentiation or the adjoint sensitivity method [6].

B. Attack Model and Adversarial Training

Attack model: In this letter, we consider evasion attacks, where the attacks occur at the time of inference after the model has already been trained [10].

We consider a range of capabilities, from a random attack scenario, where noise of certain variance is added to the input data, to a whitebox attack, where the attacker can have full access to model parameters to reduce model performance.

Adversarial training: Suppose we have a training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i \in [n]}$. Standard training solves the following empirical risk minimization (ERM) problem:

$$\min_\theta \sum_{(x,y) \in \mathcal{D}} \ell(g(x, f_\theta), y), \tag{2}$$

where ℓ is the loss function, such as the cross-entropy loss. A family of adversarial training techniques can be seen as solving the following min-max optimization:

$$\min_\theta \left\{ \sum_{(x,y) \in \mathcal{D}} \max_{\delta \in \Delta} \ell(g(x + \delta, f_\theta), y) \right\}, \tag{3}$$

where δ is the perturbation, Δ is the set of feasible perturbations (e.g., vectors with bounded ℓ_1, ℓ_2, or ℓ_∞ norm). Typical algorithms to solve inner maximization include projected gradient descent (PGD) [17] and fast sign gradient method (FSGM) [10]. During each iteration of the outer minimization, the inner maximization is approximated by either PGD or FSGM. Let $\mathcal{D}_\theta = \{(x_i', y_i)\}_{i \in [n]}$ denote the adversarial data, where x_i' is obtained by some attack algorithm (e.g., PGD, FSGM) on the data (x, y). Then, the adversarial training procedure can be seen as performing standard training on augmented datasets:

$$\min_\theta \sum_{(x,y) \in \mathcal{D} \cup \mathcal{D}_\theta} \ell(g(x, f_\theta), y), \tag{4}$$

where $\mathcal{D}_\theta$ is a set that depends on both the current model θ and the attack algorithm. In our experiments, we also consider generating $\mathcal{D}_\theta$ by simply adding Gaussian noises to each input to emulate natural perturbations, in which case the dependence on θ is no longer needed.

III. Methodology

A. Robustness Certificate

Robust set: Suppose that the input x takes values in a compact metric space $\mathcal{X}$ and the output $y \in \mathcal{Y} = \{1, \ldots, K\}$. For a given x and a fixed NODE with parameter θ, let $\Xi_\theta(x) = \{z(T) : z(0) = x + \delta, \dot{z}(t) = f_\theta(z(t)), t \in [0, T], \delta \in \Delta\}$ denote the *output perturbation set*, i.e., the set of final states of NODE when the initial state is perturbed by any $\delta \in \Delta$. Let $\mathcal{C}_y = \{z : [Wz]_y \geq [Wz]_{y'}, \forall y' \in \mathcal{Y}\}$ represent the set of embeddings that lead to the selection of class y under the argmax rule applied after an affine transformation. Then, for any data sample (x, y), the *robust set* $\mathcal{S}_\theta(x, y)$ is given by

$$\mathcal{S}_\theta(x, y) := \Xi_\theta(x) \cap \mathcal{C}_y, \tag{5}$$

which represents the set of perturbed final states that still lead to a correct result. Consider the following two cases:

1) $\Xi_\theta(x) \setminus \mathcal{S}_\theta(x,y) \neq \emptyset$ implies that there is a $\delta \in \Delta$ that falsifies the output, i.e., $\arg\max_k [Wz(T)]_k \neq y$;
2) $\Xi_\theta(x) = \mathcal{S}_\theta(x,y)$ implies that NODE θ can provide a correct decision under any perturbation.

We also note that $\mathcal{S}_\theta(x,y) \subseteq \mathcal{C}_y$ always holds.

Robustness certificate: Recall that an extended class $\mathcal{K}_\infty$ function is a function $\alpha : \mathbb{R} \to \mathbb{R}$ that is strictly increasing and with $\alpha(0) = 0$. The following result provides a certificate for a given point (x, y) against arbitrary perturbation $\delta \in \Delta$.

Theorem 1 (Robustness Certificate): Suppose there exists a continuous and almost everywhere differentiable function $h_y : \mathcal{C}_y \to \mathbb{R}$ such that **(1)** $\mathcal{S}_\theta(x,y) \subseteq \mathcal{H}_y \subseteq \mathcal{C}_y$, where $\mathcal{H}_y = \{z : h_y(z) \geq 0\}$, and **(2)** there exists an extended class $\mathcal{K}_\infty$ function α such that for NODE (1):

$$\dot{h}_y(z(t)) \geq -\alpha(h_y(z(t))) \tag{6}$$

for all $z(t) \in \Xi_\theta(x)$ and $t \in [0, T]$, then, for any $\delta \in \Delta$, there exists a finite $T' > 0$ such that $z(t, x+\delta) \in \mathcal{C}_y$ for all $t \geq T'$. We call the function h_y a robustness certificate (for point (x, y) under NODE θ).

Remark 1: A natural candidate for h_y is the family of functions parameterized by W: $h_y(z) = [Wz]_y - \max_{k \neq y} [Wz]_k$, which is continuous, differentiable everywhere, and satisfies condition (1) because in this case $\mathcal{H}_y = \mathcal{C}_y$.

Remark 2: The robustness certificate defined above enforces the invariance of $\mathcal{H}_y$ that includes the robust set $\mathcal{S}_\theta(x,y)$ but excludes the remaining perturbation set $\Xi_\theta(x) \setminus \mathcal{S}_\theta(x,y)$ that corresponds to a corrupted result. In addition, it also ensures that for perturbed final states in $\Xi_\theta(x) \setminus \mathcal{S}_\theta(x,y)$, the NODE will be able to recover to a correct decision asymptotically, that is, $S_\theta(x,y) \subseteq \mathcal{C}_y$ if we run the underlying NODE long enough (with T large enough).

The following corollary extends the certificate to a dataset.

Corollary 1: Given a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i \in [n]}$, suppose there exists a continuous and almost everywhere differentiable function $h_y : \mathcal{C}_y \to \mathbb{R}$ such that satisfies conditions (1) and (2) in Theorem 1. Then, for any $(x, y) \in \mathcal{D}$ and any $\delta \in \Delta$, there exists a finite $T' > 0$ such that $z(t, x+\delta) \in \mathcal{C}_y$ for all $t \geq T'$.

Remark 3: While barrier functions have been widely used in control systems to establish safety, avoidance, or eventuality properties [1], this letter is the first to adapt the method for a classification problem with new notions of robust set and robustness certificate.

B. Training With Robustness Certificate

To integrate the robustness certificate into training, ideally, we can solve the following optimization:

$$\min_{\theta, \{h_y\}_{y \in \mathcal{Y}}} \sum_{(x,y) \in \mathcal{D}} \ell(g(x, f_\theta), y), \tag{7a}$$

$$\text{s.t. } h_y \in C^1(\mathbb{R}^d, \mathbb{R}), \quad \forall y \in \mathcal{Y} \tag{7b}$$

$$\{z : h_y(z) \geq 0\} \subseteq \mathcal{C}_y, \quad \forall y \in \mathcal{Y} \tag{7c}$$

$$\mathcal{S}_\theta(x,y) \subseteq \{z : h_y(z) \geq 0\}, \quad \forall (x,y) \in \mathcal{D} \tag{7d}$$

$$\dot{h}_y(z(t)) \geq -\alpha(h_y(z(t))), \quad \forall z(t) \in \Xi_\theta(x), (x,y) \in \mathcal{D} \tag{7e}$$

where objective (7a) can be the usual cross-entropy loss, (7b) enforces that h_y is continuous and almost everywhere differentiable, (7c), (7d) and (7e) correspond to conditions (1) and (2) in Corollary 1, respectively. For comparison, TisODE [12] and SODEF [15] impose stability or steady-state constraints on NODE dynamics f_θ, whereas robustness is enforced by the existence of a certificate function.

Learning-based certificate: Since optimization over functions $\{h_y\}_{y \in \mathcal{Y}}$ can be intractable, we specify a parametric form of $h_y(z) = [Wz]_y - \max_{k \neq y} [Wz]_k$ for a given W shared among $\{h_y\}_{y \in \mathcal{Y}}$. This immediately satisfies constraints (7b)–(7d) (see Remark 1). We also tie this parameter to the parameter of g_W because their goals are aligned; in this case, the certificate coincides with the last layer transformation. Note that constraint (7e) is specified over a set. While sum-of-squares provides a principled approach [1], computation becomes quickly intractable for higher-dimensional systems. Along the line of learning-based certificates [8], we propose two complementary ways to approximate this constraint with data.

- *Random samples:* For each data point (x, y), we randomly sample a set of perturbations; for each perturbation δ, we forward propagate NODE to obtain $z(T)$. The set of such $z(T)$'s is collected by $\hat{\Xi}_\theta(x)$, which is used to replace $\Xi_\theta(x)$ in (7e).
- *Adversarial examples:* Similar to the above, except that for each data (x, y), we apply an attack algorithm to find a set of δ's. The corresponding $z(T)$'s together with the original $z(T)$ without input perturbation are collected by $\hat{\Xi}_\theta(x)$ used to replace $\Xi_\theta(x)$ in (7e).

In general, we can expect that a sufficient number of random samples can bring $\hat{\Xi}_\theta(x)$ close to $\Xi_\theta(x)$; in the case of adversarial samples, the constraint is more biased towards attack cases, where the robustness certificate is most likely to be violated. We use $\mathcal{D}_\theta(x,y)$ and $\mathcal{D}_\theta = \cup_{(x,y) \in \mathcal{D}} \mathcal{D}_\theta(x,y)$ to represent perturbed input sets for each data (x, y) and the entire dataset, respectively, which can be a mixture of random samples and adversarial examples. Therefore, instead of directly optimizing (7), in our implementation, we optimize the following empirical Lagrangian:

$$\min_\theta \quad \mathcal{L}(\theta) \coloneqq \mathcal{L}_0(\theta) + \lambda_1 \mathcal{L}_1(\theta) + \lambda_2 \mathcal{L}_2(\theta) + \lambda_3 \mathcal{L}_3(\theta), \tag{8}$$

where $\mathcal{L}_0(\theta) = \sum_{(x,y) \in \mathcal{D}} \ell(g(x, f_\theta), y)$ is the usual training loss, $\mathcal{L}_1(\theta) = \sum_{(x,y) \in \mathcal{D}_\theta} [-h_y(z(T,x)) + \epsilon_1]_+$ is the loss that penalizes mistakes due to random/adversarial perturbations, $\mathcal{L}_2(\theta) = \sum_{(x,y) \in \mathcal{D} \cup \mathcal{D}_\theta} [-\dot{h}_y(z(T,x)) - h_y(z(T,x)) + \epsilon_2]_+$ is the regularization term for constraint (7e) with the simple choice of $\alpha(a) = a$, $\mathcal{L}_3(\theta) = \sum_{(x,y) \in \mathcal{D} \cup \mathcal{D}_\theta} [|\dot{h}_y(z(T,x))| - \epsilon_3]_+$ for some constant $\{\epsilon_i > 0\}_{i \in [3]}$, and $\{\lambda_i \geq 0\}_{i \in [3]}$ are regularization coefficients (can be seen as dual variables for the Lagrangian relaxation of (7)).

Remark 4 (Reverse-time Robustness): The term $\mathcal{L}_3(\theta)$ is not needed if we are able to enforce constraint (7e) exactly (i.e., over the entire set $\Xi_\theta(x)$). However, it is necessary in the case of sample-based constraints. Intuitively, consider an input (x, y) that leads to a final state $z(T, x)$ lies within but close to the boundary of $\{z : h_y(z) \geq 0\}$. If the rate $|\dot{h}(z(T,x))|$ is large and if the constraint (7e) is not enforced on all points along the trajectory $z(t) \in \Xi_\theta(x)$, then it is plausible that a small time

shift $t' = t - \delta_t$ can drive the state out of $\mathcal{S}_\theta(x, y)$. Such an attack, called revserse-time attack, has not been discussed in the literature due to the specific nature of NODE; nevertheless, it can be performed easily because perturbation on the input $\delta = z(0, x) - z(-\delta_t, x)$ can be small if δ_t is small (here, $z(-\delta_t, x)$ is the state after running NODE for δ_t in reverse time).

IV. Theoretical Analysis

We present a theoretical framework to analyze to what extent the level of robustness of a trained NODE generalizes to unseen samples from the same distribution.

A. Certified Robustness Bound

Let $\xi_i := (x_i, y_i)$ be a random sample drawn from probability space $(\Omega, \mathcal{F}, \mathbb{P})$, which is endowed with a σ-algebra $\mathcal{F}$ and a probability measure $\mathbb{P}$. A dataset $\mathcal{D}_n := \{\xi_i\}_{i\in[n]} \in \Omega^n$ consists of n observations drawn independently from Ω according to $\mathbb{P}$. Formally, solving (8) for a given dataset can be regarded as a numerical procedure $\mathcal{A}_r : \Omega^r \to \Theta$, indexed by the sample size $r \in [n]$, which returns a NODE parameter θ along with a robustness certificate. We use $\mathcal{A}(S)$ to denote $\mathcal{A}_r(S)$ for any subset $S \subseteq \mathcal{D}_n$, where we omit the subscript $r = |S|$, i.e., the cardinality of S. For every data sample $\xi = (x, y)$, define

$$\Theta_\xi := \Big\{\theta \in \Theta : h_y(z(T, x)) \geq 0, |\dot{h}_y(z(t))| \leq \epsilon, \\ \dot{h}_y(z(t)) \geq -\alpha(h_y(z(t))), \forall z(t) \in \Xi_\theta(x)\Big\}$$

as the set of NODE parameters that renders the data ξ correctly classified (due to $h_y(z(T, x)) \geq 0$) and *certifiably robust* to any perturbation in Δ (due to Theorem 1). Similarly, we define an empirical estimate of the set $\hat{\Theta}_\xi$ by replacing the conditions above with $h_y(z(T, x)) \geq \epsilon_1$, $\dot{h}_y(z(t)) \geq -\alpha(h_y(z(t))) + \epsilon_2$, and $|\dot{h}_y(z(t))| \leq \epsilon_3$ for all $z(t) \in \hat{\Xi}_\theta(x)$, which coincide with the loss terms in $\{\mathcal{L}_i(\theta)\}_{i\in[3]}$ for (x, y).

The goal of the analysis is to study how the robustness of a NODE $\theta_n = \mathcal{A}(\mathcal{D}_n)$ returned by $\mathcal{A}$ on a dataset $\mathcal{D}_n$ generalizes to a yet unseen data $\xi \in \Omega$.

Definition 1: The robustness probability (RP) of a given parameter $\theta \in \Theta$ is defined as $V(\theta) := \mathbb{P}(\xi \in \Omega{:}\theta \in \Theta_\xi)$. The robustness rate (RR) on an evaluation subset $S \subseteq \mathcal{D}_n$ is $\hat{V}(\theta; S) := \frac{1}{n}\sum_{\xi\in\mathcal{D}_n} \mathbb{1}(\theta \in \hat{\Theta}_\xi) - \frac{1}{n}\sum_{\xi\in\mathcal{D}_n\setminus S} \mathbb{1}(\theta \notin \hat{\Theta}_\xi)$.

Note that RP $V(\theta)$ should be interpreted as a *lower bound* on the true robustness probability of a given NODE θ, since $\theta \in \Theta_\xi$ implies that the data ξ is certifiably robust to all $\delta \in \Delta$ but a data ξ' may be robust even if $\theta \notin \Theta_{\xi'}$ (i.e., the condition $\theta \in \Theta_\xi$ is a sufficient but not necessary condition for certified robustness). We use $\hat{V}(\theta)$ for $\hat{V}(\theta; \mathcal{D}_n)$ if the evaluation dataset is $\mathcal{D}_n$ for notational simplicity.

Assumption 1: For any data $\xi \in \mathcal{D}_n$ in the training set, we have that $\hat{\Theta}_\xi \subseteq \Theta_\xi$.

Assumption 2: The algorithm $\mathcal{A} : \Omega^n \to \Theta$ to solve (8) yields a unique output. Suppose the output is $\theta_n = \mathcal{A}(\mathcal{D}_n)$. For any subset $S \subseteq \mathcal{D}_n$ where the membership $\xi \in S$ implies that $\theta_n \in \hat{\Theta}_\xi$, we have that $\theta_n = \mathcal{A}(\mathcal{D}_n \setminus S)$.

Assumption 1 ensures that a point that is robust against adversarial examples is also robust against all perturbations in Δ. This can be achieved by choosing sufficiently large margins $\epsilon_i[i \in [3]$ under the condition that the NODE dynamics $f\theta$ is bounded. To satisfy the uniqueness requirement in Assumption 2, a simple tie-break rule can be used to select the solution with minimum norm in cases where multiple solutions exist. The second requirement is similar to the non-support data concept in support vector machines, indicating that removing non-support data should not alter the solution.

Main result: In the following, we focus the analysis on proving a PAC (probably approximately correct)-type of result: $V(\theta_n) \geq \hat{V}(\theta_n) - \kappa$ for some parameter $\kappa \in (0, 1)$ with a probability at least $1 - \beta$. A probability bound of $1 - \beta$ is necessary as $\theta_n = \mathcal{A}(\mathcal{D}_n)$ is a random variable defined over Ω^n. Before stating our main result, consider a function $\mathcal{I}_n : \Omega^n \to \{1, \ldots, n\}$ that returns $\mathcal{S} = \mathcal{I}_n(\mathcal{D}_n)$ for a given dataset $\mathcal{D}_n$, where $\mathcal{S} \subseteq \{1, \ldots, n\}$ is a subset of indices such that $\mathcal{D}_n(\mathcal{S})$, namely, the subset of data indexed by $\mathcal{S}$, has the property that $\mathcal{A}(\mathcal{D}_n) = \mathcal{A}(\mathcal{D}_n(\mathcal{S})))$ and $\hat{V}(\mathcal{A}(\mathcal{D}_n); \mathcal{D}_n) = \hat{V}(\mathcal{A}(\mathcal{D}_n(\mathcal{S})); \mathcal{D}_n(\mathcal{S}))$.[1]

Theorem 2: For the given $\mathcal{A}$ that solves (8), it holds that

$$\mathbb{P}^n\Big(V(\theta_n) \geq \hat{V}(\theta_n) - \kappa(r_n)\Big) \geq 1 - \beta, \tag{9}$$

where $r_n = |\mathcal{I}_n(\mathcal{D}_n)|$ and $\theta_n = \mathcal{A}(\mathcal{D}_n)$. Here, $\kappa(r) : \{0, \ldots, n\} \to [0, 1]$ is any function such that $\kappa(0) = 1$ and

$$\sum_{r=1}^{n} \binom{n}{r} (1 - \kappa(r))^{n-r} \leq \beta.$$

Remark 5: To compute the bound, a simple choice is to split β evenly among the n terms in the summation:

$$\kappa(r) = \begin{cases} 1 & \text{if } r = n \\ \left[1 - \left(\frac{\beta}{n\binom{n}{r}}\right)^{1/(n-r)}\right]_+ & \text{o.w.,} \end{cases} \tag{10}$$

where $\binom{n}{r}$ is the n-choose-r binomial coefficient.

Remark 6: The practical optimization equation (8) is a computationally tractable relaxation of the ideal optimization equation (7) from constrained optimization to unconstrained optimization with empirical data. Theorem 2 provides a bound for the solution obtained from (8), evaluated against the constraints from (7). If a point ξ satisfies the constraints from (7) (i.e., $\theta \in \Theta_\xi$), it is certifiably robust according to Theorem 1. A point ξ that satisfies the empirical version $\theta \in \hat{\Theta}_\xi$ (which leads to zero loss incorporated in (8)) is also certifiably robust. Theorem 2 establishes a lower bound of the robustness of the solution obtained from (8) using RP, and this lower bound holds according to the constraints from (7).

Remark 7: The theoretical bound in (9) differs from common generalization bounds in learning theory in that the bound is evaluated a-posteriori after the solution is computed. This "wait-and-judge" type of result has been developed in convex optimization [3]. However, existing methods primarily apply to feasibility problems [3], assuming the existence of a NODE parameter $\theta \in \Theta$ such that $\theta \in \Theta_\xi$ for all $\xi \in \mathcal{D}_n$. This is

[1] As an example of such function, we can construct a set S such that $\xi_i \in S$ if and only if $\xi_i \in \mathcal{D}_n$ and $\theta_n \notin \Theta_{\xi_i}$. We collect the original indices of all points in S as $\mathcal{S}$ as the output of $\mathcal{I}_n(\mathcal{D}_n)$. The validity of this function is implied by Assumption 2. However, note that the existence of such a function does not rely on Assumption 2.

TABLE I
ROBUSTNESS AGAINST GAUSSIAN NOISE (ZERO MEAN AND DIFFERENT VARIANCES σ^2). THE AVERAGE AND STANDARD DEVIATION ARE ACROSS 10 RANDOM PERTURBATIONS FOR EACH DATA

	$\sigma^2 = 20$	$\sigma^2 = 50$	$\sigma^2 = 100$	$\sigma^2 = 150$
vanilla NODE	99.8±0.1	99.7±0.1	82.7±0.5	53.7±0.5
B-NODE	99.8±0.1	99.7±0.1	98.4±0.2	89.4±0.6

TABLE II
ROBUSTNESS AGAINST PGD ATTACK ON MNIST

PGD	0.01	0.02	0.03	0.04	0.05
vanilla NODE	86.14±1.33	79.26±1.80	71.04±2.01	60.36±2.34	47.40±2.11
AT-NODE	91.76±1.16	89.34±1.14	85.8±1.94	82.94±1.64	78.38±1.67
ODE-TRADES	91.46±1.07	88.98±1.98	85.08±1.61	81.94±0.94	77.32±1.89
TisODE	**92.67**±1.28	91.06±1.41	87.48±1.42	84.54±2.03	81.28±1.28
B-NODE	92.62±0.87	**91.52±1.22**	**89.16±1.25**	**86.36±1.78**	**83.18±1.75**

TABLE III
ROBUSTNESS AGAINST AUTOPGD ATTACK ON MNIST

AutoPGD	0.01	0.02	0.03	0.04	0.05
vanilla NODE	84.41±1.11	76.66±1.65	68.99±1.71	58.33±2.16	46.31±2.51
AT-NODE	91.34±1.51	89.49±1.89	87.60±2.67	82.70±2.00	78.35±1.87
ODE-TRADES	92.19±1.92	88.30±2.30	85.31±2.41	81.90±1.85	76.89±3.85
TisODE	91.12±1.27	88.05±2.42	87.02±2.21	84.31±2.04	79.48±2.35
B-NODE	**93.45±1.42**	**91.25±1.34**	**88.95±2.09**	**86.30±3.24**	**82.65±2.58**

TABLE IV
ROBUSTNESS AGAINST SQUARE ATTACK ON MNIST

Square attack	0.01	0.02	0.03	0.04	0.05
vanilla NODE	86.55±2.65	83.29±3.66	72.50±5.85	62.49±3.59	51.58±4.05
AT-NODE	92.65±1.72	90.99±2.50	89.50±3.84	85.51±3.84	81.13±5.44
ODE-TRADES	92.06±3.10	91.85±2.73	90.08±2.73	86.24±2.56	82.41±2.96
TisODE	94.10±0.97	91.29±1.28	87.60±1.69	86.87±1.98	83.24± 2.25
B-NODE	**94.49±2.55**	**92.99±0.99**	**91.52±3.49**	**88.51±4.51**	**85.49±2.50**

TABLE V
PERFORMANCE ON FASHIONMNIST

	PGD-$\frac{4}{255}$	PGD-$\frac{6}{255}$	PGD-$\frac{8}{255}$	PGD-$\frac{10}{255}$	PGD-$\frac{12}{255}$
AT-NODE	75.99±2.28	72.15±2.87	70.70±2.81	66.25±2.67	63.85±3.21
ODE-TRADES	77.68±2.25	75.47±1.70	72.53±1.94	68.32±1.94	67.12±2.21
TisODE	77.04±1.66	73.02±1.61	71.44±1.92	65.32±1.97	62.88±1.36
B-NODE	**78.96±2.48**	**76.32±2.67**	**73.62±2.21**	**69.12±2.69**	**67.36±1.99**

unrealistic because we *cannot* expect a NODE that is robust for all data. Our method is broadly applicable to the relaxed setting and can be compared to the extension of standard PAC learning to agnostic PAC theory or the extension of example-consistent frameworks to non-consistent schemes [5].

V. EXPERIMENTS

In this section, we demonstrate the effectiveness of our framework for both non-adversarial and adversarial robustness. For additional experimental details, including ODE architecture and attack specifications, please refer to [20].

A. 2D Binary Classification

For each data shown in Fig. 1, 30 perturbed inputs are uniformly sampled within a radius of 1.2 centered on the original data point (red and blue dots along the circles).

Results: In Fig. 1, both methods correctly classify all original points, but vanilla NODE's output regions intersect the decision boundary, indicating potential misclassifications under perturbation, which is not observed for B-NODE. B-NODE generally has smaller perturbed output regions, with their main axis almost parallel to the decision boundary, indicating successful transformation of anisotropic perturbations on inputs into isotropic perturbations on outputs, where most perturbations are innocuous (i.e., do not change decisions). In contrast, vanilla NODE is more susceptible to output perturbations across decision boundaries (in some cases, even running the integral longer can lead to wrong decisions).

B. Evaluating Robustness on MNIST

First, we test the improvement of robustness against Gaussian noise (Table I). While vanilla NODE is robust under mild perturbations, the performance drops significantly as the variance of Gaussian noise increases.

Robustness against PGD attack [17] is shown in Table II. We compare our method to the vanilla NODE trained with clean data [6], NODE trained with data augmentation (AT-NODE), TisODE [12], and ODE-TRADES [23], using the same architecture for both methods to ensure a fair comparison. As the attack radius increases (ℓ_∞-norm bounds shown in the first row of each table), the performance deteriorates for all methods. However, the drop in performance for B-NODE is much less, indicating enhanced robustness.

We confirmed the tightness of the bound in Theorem 2 by calculating the theoretical lower bound for robust accuracy with 99% confidence. For 2000 training data, we obtained lower bounds ranging from 60.70% to 53.44% (in percentage) as PGD magnitudes increased from 0.01 to 0.05 (see [20] for details). Although there remains a discrepancy between the theoretical lower bound and the actual test error (as shown in the last row of Table II), such numbers are nontrivial considering the size and nonlinearity of the learning system.

According to [13], evaluating against PGD is not sufficient to empirically demonstrate the adversarial robustness of NODEs because NODEs can obfuscate gradients. Therefore, additional experiments against AutoPGD [7] and Square Attack [2] are performed to demonstrate effectiveness of defense, as shown in Table III and IV.

C. Evaluation on FashionMNIST and CIFAR10

For the FashionMNIST data, we tested with convolution Neural ODEs, which uses a 2D matrix to represent hidden states during forward propagation. AT-NODE, ODE-TRADES and B-NODE are trained with clean data augmented with adversarial examples, which are generated by 40 steps L_∞ norm PGD attack, whose magnitude ε is 8/255. The clean data accuracy for vanilla NODE, AT-NODE, B-NODE and ODE-TRADES are 85.36%, 82.10%, 82.68% and 83.24%, respectively.

For the CIFAR10 experiment, we use a pre-trained CNN model for feature extractor, the output of which is provided to NODE as an initial state [15]. The clean data accuracy for vanilla NODE, B-NODE, ODE-TRADES are 89.68%, 89.16%, and 90.48%, respectively.

The results in Table V and VI indicate that B-NODE exhibits the best performance among all methods. Nevertheless, we

TABLE VI
PERFORMANCE ON CIFAR10

	PGD-$\frac{6}{255}$	PGD-$\frac{9}{255}$	PGD-$\frac{12}{255}$	PGD-$\frac{15}{255}$
AT-NODE	58.56±1.65	45.35±2.46	35.71±2.41	25.76±3.12
ODE-TRADES	59.28±2.36	47.60±2.16	39.48±2.93	30.48±2.84
TisODE	61.04±2.25	48.05±2.49	37.67±2.78	27.59±2.36
B-NODE	**61.81±2.35**	**48.52±2.45**	**39.60±3.14**	**30.52±2.27**

TABLE VII
ABLATION ANALYSIS FOR MODELS WITH AND WITHOUT BARRIER FUNCTION LOSS OR DATA AUGMENTATION ON MNIST DATASET

	PGD-0.01	PGD-0.02	PGD-0.03	PGD-0.04	PGD-0.05
vanilla NODE	86.14±1.33	79.26±1.80	71.04±2.01	60.36±2.34	47.40±2.11
B-NODE w/o AT	88.62±1.40	83.82±1.10	79.00±1.39	71.22±2.23	61.46±1.64
AT-NODE	91.76±1.16	89.34±1.14	85.80±1.94	82.94±1.64	78.38±1.67
B-NODE w/ AT	92.62±0.87	91.52±1.217	89.16±1.25	86.36±1.78	83.18±1.75

observed a significant decrease in performance compared to the accuracy on clean data, suggesting that there is room for improvement in future work.

D. Ablation Analysis

We conducted an experiment to analyze the impact of learning-based barrier functions and data augmentation on the robustness of four different models (see details in [20, Appendix]). As shown in Table VII, barrier function without data augmentation can already improve robustness, sometimes by 15% (in the case of PGD-0.05). It is remarkable to see improvement in all cases of attacks, indicating some level of "universal robustness." However, the use of the barrier function in conjunction with data augmentation yields the most robust performance.

VI. CONCLUSION

We have developed an algorithm to train NODEs based on barrier functions with certified robustness. Future directions include examining the necessity of the robustness certificates and exploring the incorporation of (exponential) control barrier functions to problems beyond classifications.

ACKNOWLEDGMENT

This work was authored in part by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for DOE under Contract No. DE-AC36-08GO28308. Funding provided by the U.S. Department of Energy Office of Electricity Advanced Grid Modeling (AGM) program. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.

REFERENCES

[1] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *Proc. 18th Eur. Control Conf. (ECC)*, 2019, pp. 3420–3431.

[2] M. Andriushchenko, F. Croce, N. Flammarion, and M. Hein, "Square attack: A query-efficient black-box adversarial attack via random search," in *Proc. ECCV*, 2020, pp. 484–501.

[3] M. C. Campi and S. Garatti, "Wait-and-judge scenario optimization," *Math. Program.*, vol. 167, no. 1, pp. 155–189, 2018.

[4] M. C. Campi, S. Garatti, and F. A. Ramponi, "A general scenario theory for nonconvex optimization and decision making," *IEEE Trans. Autom. Control*, vol. 63, no. 12, pp. 4067–4078, Dec. 2018.

[5] M. C. Campi and S. Garatti, "Compression, generalization and learning," 2023, *arXiv:2301.12767*.

[6] R. Chen, Y. Rubanova, J. Bettencourt, and D. K. Duvenaud, "Neural ordinary differential equations," in *Proc. NeurIPS*, vol. 31, 2018, pp. 6572–6583.

[7] F. Croce and M. Hein, "Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks," in *Proc. ICML*, 2020, pp. 2206–2216.

[8] C. Dawson, S. Gao, and C. Fan, "Safe control with learned certificates: A survey of neural Lyapunov, barrier, and contraction methods," 2022, *arXiv:2202.11762*.

[9] A. Ghosh, H. S. Behl, E. Dupont, P. H. S. Torr, and V. Namboodiri, "STEER: Simple temporal regularization for neural ODE," in *Proc. NeurIPS*, vol. 33, 2020, pp. 14831–14843.

[10] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. ICLR*, 2015, pp. 1–11.

[11] S. Grunbacher, R. Hasani, M. Lechner, J. Cyranka, S. A. Smolka, and R. Grosu, "On the verification of neural ODEs with stochastic guarantees," in *Proc. AAAI Conf. Artif. Intell.*, vol. 35, 2021, pp. 11525–11535.

[12] H. Yan, J. Du, V. Y. Tan, and J. Feng, "On robustness of neural ordinary differential equations," in *Proc. Int. Conf. Learn. Represent.*, 2020, pp. 1–15.

[13] Y. Huang, Y. Yu, H. Zhang, Y. Ma, and Y. Yao, "Adversarial robustness of stabilized neural ODE might be from obfuscated gradients," in *Proc. Int. Conf. Math. Sci. Mach. Learn.*, 2022, pp. 497–515.

[14] Y. Huang, I. D. J. Rodriguez, H. Zhang, Y. Shi, and Y. Yue, "FI-ODE: Certified and robust forward invariance in neural ODEs," 2022, *arXiv:2210.16940*.

[15] Q. Kang, Y. Song, Q. Ding, and W. P. Tay, "Stable neural ODE with Lyapunov-stable equilibrium points for defending against adversarial attacks," in *Proc. NeurIPS*, vol. 34, 2021, pp. 14925–14937.

[16] X. Liu, T. Xiao, S. Si, Q. Cao, S. Kumar, and C.-J. Hsieh, "How does noise help robustness? explanation and exploration under the neural SDE framework," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 282–290.

[17] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *Proc. ICLR*, 2018, pp. 1–23.

[18] I. Rodriguez, A. Ames, and Y. Yue, "LyaNet: A Lyapunov framework for training neural ODEs," in *Proc. Int. Conf. Mach. Learn.*, 2022, pp. 18687–18703.

[19] V. Tjeng, K. Y. Xiao, and R. Tedrake, "Evaluating robustness of neural networks with mixed integer programming," in *Proc. Int. Conf. Learn. Represent.*, 2018, pp. 1–21.

[20] R. Yang, R. Jia, X. Zhang, and M. Jin. "Certifiably robust neural ODE with learning-based barrier function." 2023. [Online]. Available: http://www.jinming.tech/papers/B-NODE23full.pdf

[21] M. Zakwan, L. Xu, and G. Ferrari-Trecate, "Robust classification using contractive Hamiltonian neural ODEs," 2022, *arXiv:2203.11805*.

[22] Y. Zeng et al., "Towards robustness certification against universal perturbations," in *Proc. 11th Int. Conf. Learn. Represent.*, 2023, pp. 1–18.

[23] H. Zhang, Y. Yu, J. Jiao, E. Xing, L. El Ghaoui, and M. Jordan, "Theoretically principled trade-off between robustness and accuracy," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 7472–7482.