Computers and Chemical Engineering 163 (2022) 107860

Contents lists available at ScienceDirect

Computers and Chemical Engineering

journal homepage: www.elsevier.com/locate/compchemeng

# Statistical machine-learning-based predictive control using barrier functions for process operational safety

Scarlett Chen[a], Zhe Wu[c], Panagiotis D. Christofides[a,b,*]

[a] Department of Chemical and Biomolecular Engineering, University of California, Los Angeles, CA 90095-1592, USA
[b] Department of Electrical and Computer Engineering, University of California, Los Angeles, CA 90095-1592, USA
[c] Department of Chemical and Biomolecular Engineering, National University of Singapore, 117585, Singapore

ARTICLE INFO

*Article history:*
Received 4 April 2022
Revised 25 May 2022
Accepted 26 May 2022
Available online 27 May 2022

*Keywords:*
Neural networks
Generalization error
Nonlinear model predictive control
Process operational safety
Barrier functions
Statistical machine learning

ABSTRACT

In this work, we present statistical model predictive control with Control Lyapunov-Barrier Functions (CLBF) built using machine learning approaches, and analyze closed-loop stability and safety properties in probability using statistical machine learning theory. A feedforward neural network (FNN) is used to construct the Control Barrier Function, and a generalization error bound can be obtained for this FNN via the Rademacher complexity method. The FNN Control Barrier Function is incorporated in a CLBF-based model predictive controller (MPC), which is used to control a nonlinear process subject to input constraints. The stability and safety properties of the closed-loop system under the sample-and-hold implementation of FNN-CLBF-MPC are evaluated in a statistical sense. We use a chemical process example to demonstrate the relation between various factors of building an FNN model and the generalization error, as well as the probabilities of closed-loop safety and stability for both bounded and unbounded unsafe sets.



## 1. Introduction

Safety-critical systems are prevalent in many application domains such as aviation, automobiles, energy, chemical processing, medicine, and safety-related requirements must be strictly satisfied on their inputs and states in order to prevent harm on process stability, economic gains, and/or operational safety. There has been extensive research on providing safety verification of a system as well as synthesizing control laws with provable safety properties (Prajna and Jadbabaie, 2004; Ratschan and She, 2007; Althoff et al., 2011; Mitra et al., 2013). Amongst these methods, Control Barrier Functions (CBFs) are proposed as a tool to characterize the safety of dynamical systems by certifying whether a control law achieves forward invariance of a safe set, similar to the utility of Control Lyapunov Function (CLF) in certifying stability properties (Ames et al., 2014, 2016, 2017; Xu et al., 2015; Xu, 2016). CBFs can be incorporated in the design of control laws for multi-objective control of safety-critical systems, e.g., controllers designed based on Control Lyapunov Barrier Functions (CLBF), where a CLF is used to characterize a stability region and ensure stability properties, and a CBF is used to characterize an unsafe region where the state trajectory under the CLBF-based control law will not enter at all times (Romdlony and Jayawardhana, 2016). This approach has been further explored for nonlinear systems subject to constrained inputs in Wu and Christofides (2019), Wu et al. (2019a), where CLBF-based control laws are used as contractive constraints in the design of a model predictive controller (MPC) to provide closed-loop safety and stability guarantees for nonlinear processes with embedded bounded and unbounded unsafe regions.

The development of an explicit CBF expressed in closed form remains a challenging task, especially for nonlinear processes, regardless of whether the process dynamics is well-defined. There has been previous works on characterizing a barrier function using machine learning methods, such as using support vector machines (Srinivasan et al., 2020) and neural networks (Jin et al., 2020; Zhao et al., 2020). Moreover, in Lindemann et al. (2020), Robey et al. (2020), optimization-based approaches are used to learn CBFs from data for nonlinear continuous control affine dynamical systems as well as hybrid systems. In (Yaghoubi et al., 2020), an imitation learning framework is proposed to learn neural network-based feedback controllers with CBF constraints for systems under disturbances. The work in Jin et al. (2020) uses neural networks to jointly learn a Lyapunov-like function and a barrier function and obtains a safe and goal-reaching control policy. Simi-

* Corresponding author at: Department of Chemical and Biomolecular Engineering, University of California, Los Angeles, CA 90095-1592, USA.
*E-mail address:* pdc@seas.ucla.edu (P.D. Christofides).

https://doi.org/10.1016/j.compchemeng.2022.107860
0098-1354/

larly, in Zhao et al. (2020), barrier functions are synthesized using neural networks that use a devised activation function Bent-ReLu and checked against the barrier function criteria as a formal guarantee. Although formal proofs of guaranteed safety and stability have been provided either from *a priori* theoretical development or posterior empirical verification, the question of generalization accuracy of machine learning techniques has not been addressed.

There has been some research into probabilistic safety certification of barrier functions, but the probability analysis is with respect to uncertainties that exist in the process dynamics (Luo et al., 2020; Khojasteh et al., 2020; Liu et al., 2021), and not in the sense of analyzing the generalization error of the modeling method. For example, in Liu et al. (2021), a Gaussian process is used to model the projection of unknown residual dynamics onto a CBF; similarly in Khojasteh et al. (2020), the Gaussian process approach is used to obtain a distribution over the system dynamics, which is then used to ensure safety with high probability by specifying a chance constraint on a CBF. The work in Clark (2019) develops barrier functions for stochastic systems with sufficient conditions for safety with probability.

On the other hand, probably approximately correct (PAC) learning theory provides a framework for analyzing the generalization ability of machine learning models, and provides the conditions under which a learning algorithm is probably able to yield an output that is approximately correct (Valiant, 1984; Mohri et al., 2018). One way to characterize the machine learning model's capability to generalize new unseen data based on learned data is to examine the generalization error in Eryarsoy et al. (2009), a tighter error bound on the performance of classification via Support Vector Machine (SVM) is characterized by exploiting domain knowledge. A bound on the generalization error of feed-forward neural networks has been developed by providing a bound on the Rademacher complexity of the network (Golowich et al., 2018). In Wu et al. (2021), a similar bound is provided for recurrent neural networks, and statistical stability analysis of Lyapunov-based MPC using the recurrent neural network model was introduced. Generalization error in deep learning algorithms has been surveyed in Jakubovitz et al. (2019) with discussions on different measures to assess generalization capabilities of deep neural networks, such as PAC-Bayes theory, algorithm stability, algorithm robustness, and compression-based approach. In this work, we provide statistical analysis on the CBF construction method proposed in our previous work in Chen et al. (2021), and model the CBF using a feed-forward neural network, which will be used to design a CLBF-based model predictive control system. We first develop the generalization error bound on the FNN-CBF, and derive probabilistic safety and stability guarantees for the control law designed using a CLBF with FNN-CBF under sufficient conditions. The sampling, modeling, and verification procedures of the FNN are discussed. Then, we extend the probabilistic stability and safety properties to the FNN-CLBF-MPC, and demonstrate that with high probability, the FNN-CLBF-MPC is able to maintain the closed-loop state of a nonlinear process within a safe set and ultimately keep it bounded within a terminal set around the origin.

The rest of the paper is organized as follows. Preliminaries on the nonlinear system and definitions of Lyapunov Function and Barrier Function are given in Section 2. The construction of barrier functions using neural networks, including assumptions, design, data generation and model verification, are presented in Section 3. Section 4 develops the generalization error bounds on the FNN-CBF and explain their implications. In Section 5, the design of the FNN-CLBF control law and the FNN-CLBF-based MPC are provided, and the probabilistic stability and safety properties of the control system are provided. Lastly, the proposed control method and the associated generalization error and closed-loop performance are shown via a nonlinear chemical process example in Section 6.

**Table 1**
Descriptions of frequently used variables.

| Variable | Description |
|---|---|
| $x$ | State vector of nonlinear system |
| $u$ | Input vector of nonlinear system |
| $B(x)$ | Barrier function |
| $V(x)$ | Lyapunov function |
| $W(x)$ | Control Lyapunov-Barrier function |
| $\mathbf{x}$ | Input of FNN model |
| $\mathbf{y}, \hat{\mathbf{y}}$ | True and predicted output of FNN model |
| $d_x, d_y$ | Dimension of FNN input and output |
| $\sigma_l(\cdot)$ | Activation function of in each layer $l$ of the FNN model |
| $W_l$ | Weight parameter matrix in each layer $l$ of the FNN model |
| $B_X, B_W$ | Upper bound on the FNN inputs and FNN weight matrices |
| $m$ | Number of samples |
| $d$ | FNN depth (number of layers) |
| $L$ | Loss function minimized during FNN training |
| $h(\mathbf{x})$ | Hypothesis function mapping FNN input to FNN output |
| $\delta$ | Confidence associated with generalization error upper bound |
| $\epsilon$ | Rademacher random variable |
| $L_fV$ | Lie derivative of Lyapunov function along $f$ |
| $L_gV$ | Lie derivative of Lyapunov function along $g$ |

## 2. Preliminaries

### 2.1. Notation

The Euclidean norm is denoted by the operator $|\cdot|$. The notation $\|W\|_{1,\infty} = \max_j(\sum_i |W_{i,j}|)$ denotes the infinity norm of the 1-norms of the columns of matrix $W$. We use "\" to represent set subtraction, i.e., $A \backslash B := \{x \in \mathbf{R}^n \mid x \in A, x \notin B\}$. $\mathbf{x}^T$ denotes the transpose of matrix $\mathbf{x}$. $L_fV(\mathbf{x}) := \frac{\partial V(\mathbf{x})}{\partial x} f(\mathbf{x})$ represents the Lie derivative of $V$ with respect to $f$. A function $f$ is class $C^1$ if the first derivative of $f$ exists and is continuous. A function $f : \mathbf{R}^n \to \mathbf{R}^m$ is said to be $L$-Lipschitz continuous, if there exists $L \geq 0$ such that for all $a, b \in \mathbf{R}^n$, $|f(a) - f(b)| \leq L|a - b|$. A continuous function $r : [0, a) \to [0, \infty)$ belongs to a class $\mathcal{K}$ function if $r(0) = 0$, and is strictly increasing. Lastly, $\mathbb{P}(A)$ represents the probability of the occurrence of an event $A$, and $\mathbb{E}[X]$ denotes the expected value of a random variable $X$. More descriptions of frequently used variables can be found in Table 1.

### 2.2. Class of systems

In this study, we consider a general class of continuous-time nonlinear systems, which can be represented by the following state-space model:

$$\dot{x} = F(x, u) := f(x) + g(x)u, \;\; x(t_0) = x_0 \tag{1}$$

where $x \in \mathbf{R}^n$ is the state vector, $u \in \mathbf{R}^k$ denotes the manipulated input vector bounded by $u \in U$, where $U := \{u_{\min} \leq u \leq u_{\max}\} \subset \mathbf{R}^k$. It is assumed that the vector and matrix functions $f(\cdot)$ and $g(\cdot)$ are sufficiently smooth with $f(0) = 0$, and thus the origin is a steady-state of the nonlinear system. Lastly, the initial time is assumed to be at 0, i.e., $t_0 = 0$.

### 2.3. Stabilizability via lyapunov-based control

For the nonlinear system of Eq. (1), it is assumed that a stabilizing feedback control law $u = \Phi(x) \in U$ exists such that there exists a positive definite and proper Control Lyapunov Function (CLF), denoted as $V(x)$, that satisfies the following inequalities as well as the small control property:

$$c_1|x|^2 \leq V(x) \leq c_2|x|^2 \tag{2a}$$

$$\left|\frac{\partial V(x)}{\partial x}\right| \leq r_V(|x|) \tag{2b}$$

$$L_f V(x) < 0, \forall x \in \{z \in \mathbf{R}^n \backslash \{0\} \mid L_g V(z) = 0\} \tag{2c}$$

where $r_V$ is a function that belongs to class $\mathcal{K}$, and $c_1$, $c_2$ are positive constants. $V(x)$ also meets the small control property, which states that, for every $\varepsilon > 0$, $\exists\ \delta > 0$, s.t. $\forall\ x \in \mathcal{B}_\delta(0)$, there exists an input $u$ satisfying $|u| < \varepsilon$ and $L_f V(x) + L_g V(x) \cdot u < 0$ (Sontag, 1989). The existence of such CLF implies that the origin of the nonlinear system of Eq. (1) is rendered asymptotically stable under $u = \Phi(x) \in U$ for all $x$ in a neighborhood around the origin. This region where the time derivative of $V(x)$ can be rendered negative under $u = \Phi(x) \in U$ is defined as $\phi_u = \{x \in \mathbf{R}^n \mid \dot{V}(x) = L_f V(x) + L_g V(x) \cdot u < 0, u = \Phi(x) \in U\} \cup \{0\}$. Furthermore, we define a level set of $V(x)$ within $\phi_u$ as $U_b := \{x \in \phi_u \mid V(x) \leq b,\ b > 0\}$, which is a forward invariant set in a sense that for any initial condition $x_0 \in U_b$, the closed-loop trajectory $x(t),\ t \geq 0$ of the nonlinear system of Eq. (1) remains in $U_b$ under $u = \Phi(x) \in U$.

### 2.4. Control barrier function

Consider that an open set $\mathcal{D}$ exists in state space, forming an unsafe region that should be avoided at all times for reasons such as violation of safety protocols. In contrast, a set of safe states can also be characterized as $\mathcal{X}_0 := \{x \in \mathbf{R}^n \backslash \mathcal{D}\}$ where $\{0\} \in \mathcal{X}_0$ and $\mathcal{X}_0 \cap \mathcal{D} = \emptyset$. The safe set $\mathcal{X}_0$ represents the set of initial conditions that will be considered. In this work, we consider process operational safety as follows:

**Definition 1.** For any initial state $x(t_0) = x_0 \in \mathcal{X}_0$, if there exists a constrained control law $u = \Phi(x) \in U$ that renders the origin of the closed-loop system of Eq. (1) asymptotically stable, and the closed-loop state trajectories do not enter the unsafe set $D$ at all times, i.e., $x(t) \in \mathcal{X}_0$, $x(t) \notin \mathcal{D}$, $\forall\ t \geq 0$, then the control law $u = \Phi(x) \in U$ maintains the closed-loop state within the safe region $\mathcal{X}_0$ for all times.

Subsequently, we present the properties of a Control Barrier Function (CBF) in the following definition: (Wieland and Allgöwer, 2007)

**Definition 2.** Consider $\mathcal{D}$ which is a set of unsafe state values in state space, a $\mathcal{C}^1$ function $B(x) : \mathbf{R}^n \to \mathbf{R}$ is a Control Barrier Function (CBF) if the following conditions are met:

$$B(x) > 0, \quad \forall\ x \in \mathcal{D} \tag{3a}$$

$$L_f B(x) \leq 0,\ \forall\ x \in \{z \in \mathbf{R}^n \backslash \mathcal{D} \mid L_g B(z) = 0\} \tag{3b}$$

$$\mathcal{X}_B := \{x \in \mathbf{R}^n \mid B(x) \leq 0\} \neq \emptyset \tag{3c}$$

## 3. Barrier function construction using feed-forward neural networks

### 3.1. Model structure and training

The control barrier function is developed from operating data in the state space that are labelled based on their safety status. This barrier function will then be synthesized using a feed-forward neural network (FNN), which typically consists of an input layer, some hidden layers, and an output layer. Each layer contains neurons undergoing nonlinear transformations, with activation functions of the weighted sum of neurons in the previous layer plus a bias term. In this study, the inputs to the FNN are the state vector $x \in \mathbf{R}^n$ of the nonlinear system of Eq. (1), and the output of the FNN predicts the barrier function value $\hat{B}(x) \in \mathbf{R}^n$. Training data points are collected from both the unsafe and the safe operating regions, where the target output values of $B(x)$ will satisfy the CBF conditions of Eqs. (3a) and (3c) for the unsafe and the safe regions, respectively. More specifically, safe data points are labeled with a target output value of $B(x) = -1$, and unsafe data points are labeled with a target output value of $B(x) = +1$.

A general FNN model is considered, where $m$ number of data samples are used to develop this model. The data samples are generated independently as per the data distribution over $X \times Y \in \mathbf{R}^{d_x} \times \mathbf{R}^{d_y}$, where $d_x$ and $d_y$ denote the dimension of the FNN input and output vectors respectively; in this application, $d_x = n$, which is the dimension of the state vector of the nonlinear system of Eq. (1), and $d_y = 1$, which is the dimension of the barrier function output $B(x)$. The general structure of FNN model with inputs denoted as $\mathbf{x} \in \mathbf{R}^{d_x}$ and predicted output denoted as $\hat{\mathbf{y}} \in \mathbf{R}^{d_y}$ in terms of scalar or vector-valued functions and weight matrices for $d$ total number of layers can be formulated as follows:

$$\hat{\mathbf{y}} = \sigma_d(W_d \sigma_{d-1}(W_{d-1}\sigma_{d-2}(\ldots \sigma_1(W_1 \mathbf{x})))) \tag{4}$$

where each $W_l$ for $l = 1, \ldots, d$ layers represents the weight parameter matrix, and each $\sigma_l$ represents the activation function in each layer. The number of layers $d$ represents the depth of the network, and the width of the network $h_{\max}$ can be defined as the maximum number of neurons in a hidden layer (maximal column or row dimension of $W_l$), i.e., $h_{\max} = \max_{l=1,\ldots,d}\{h_l\}$, where $h_l$ denotes the number of neurons in the $l$-th layer.

In this study, due to the unique dichotomous nature of $B(x)$, we choose a hyperbolic tangent sigmoid function $\sigma(z) = \tanh(z) = \frac{2}{1+e^{-2z}} - 1$ as the activation function to polarize the output of the network and in turn, improve the prediction accuracy. This is because of the property of the $tanh(z)$ function approaching $+1$ as $z$ approaches $+\infty$, and $-1$ as $z$ approaches $-\infty$, thus polarizing the outputs of each layer and enforces the output of the FNN to approximate constant positive values (+1 for safe points), or constant negative values (-1 for unsafe points). To clarify notations used in this paper, when discussing the general properties of FNN, the input and output of the FNN model are denoted by the bold face $\mathbf{x} \in \mathbf{R}^{d_x}$ and $\mathbf{y} \in \mathbf{R}^{d_y}$ respectively. For this particular application, $\mathbf{x}$ is the state vector of Eq. (1) ($x \in \mathbf{R}^n$), and $\mathbf{y}$ is the barrier function value ($B(x) \in \mathbf{R}^1$).

Before proceeding with developing the generalization error bound, there are some standard assumptions presented as follows:

**Assumption 1.** The FNN inputs are bounded, i.e., $|\mathbf{x}_i| \leq B_X$, for all $i = 1, \ldots, m$ samples.

**Assumption 2.** The maximal 1-norm ($l_1/l_\infty$) of the rows of weight matrices in the output and in the hidden layers are bounded as follows:

$$\|W\|_{1,\infty} \leq B_W \tag{5}$$

**Assumption 3.** All the datasets (i.e., training and testing) are drawn from the same underlying distribution.

**Assumption 4.** $\sigma_l$ (where $l$ denotes any hidden layers) is a 1-Lipschitz continuous activation function, and satisfies $\sigma_l(0) = 0$.

**Remark 1.** Assumption 1 specifies the upper bound on the FNN inputs, which is consistent with the way we sample the FNN inputs (i.e., the state vector) as we only consider a bounded set around the steady-state of the nonlinear system of Eq. (1). Assumption 2 assumes the boundedness of the FNN weight matrices; this can be ensured during FNN training, as only a finite class of hypothesis functions are searched to find the optimal set of FNN parameters. Assumption 3 is required as the model trained from the training dataset will be evaluated on the testing dataset, and training and testing model accuracy metrics are compared against each other.

The evaluation of the model on the testing data (including closed-loop simulations) as well as the comparison of accuracy metrics are only valid if the two datasets have the same underlying target distribution. Assumption 4 is an assumption on the activation functions of the FNN, which is satisfied by many common activation functions, and can be used to derive the upper bound for the Rademacher complexity of the FNN hypothesis class. An example of a 1-Lipschitz continuous activation function is $tanh(\cdot)$.

We sample points from the operating region of the system (i.e., $x \in \mathcal{X} \subset \mathbf{R}^n$ where $\mathcal{X}$ is a compact set) to use as training and testing data for the FNN. Since the conditions of Eq. (3) imposed on the resulting $\hat{B}(x)$ must be satisfied in a continuous sense, the regions from which discrete data points are sampled from must be compact and connected. This is done by first characterizing a compact and connected set $\mathcal{H}$, which is a superset of the open set $\mathcal{D}$ (as indicated in Eq. (32) in Section 5), then designing a larger compact and connected set $\mathcal{H}'$, which is a superset of $\mathcal{H}$ and encloses $\mathcal{H}$ with sufficient margin. This region $\mathcal{H}'$ is used to generate unsafe data points from, such that the unsafe set the FNN model predicts will remain as a superset of $\mathcal{H}$, given bounded modeling and numerical error of the FNN model. This means that the FNN model may classify safe points as unsafe, but will not classify unsafe points as safe; the latter is not tolerated and should be avoided. Readers who are interested may refer to Chen et al. (2021) for more details on how to characterize the unsafe region for data collection purposes when building a FNN-CLBF-MPC that uses both first-principles and RNN models. We collect samples from the safe region $\mathcal{X}\backslash\mathcal{H}'$ and the unsafe region $\mathcal{H}'$ by discretizing the regions by a grid size of $(\delta x)_{\mathcal{H}'}$ and $(\delta x)_{\mathcal{X}\backslash\mathcal{H}'}$ respectively. The datasets consisting of finite samples are denoted as $S_{\mathcal{I}}$ and $S_{\mathcal{H}'}$ for safe and unsafe regions, respectively. Together, $S_{\mathcal{I}}$ and $S_{\mathcal{H}'}$ form the overall sample set $S_s$.

The FNN parameters (weights and biases) are optimized by minimizing the loss function shown in Eq. (6) using the *Adam* solver as a part of the Tensorflow Keras software package. Specifically, the loss function consists of two parts. The first part $L_1$ uses mean squared error to calculate the difference between the target $B(x)$ and the prediction $\hat{B}(x)$, and in minimizing this error, aims to satisfy the conditions of Eqs. 3a and (3c). The second part $L_2$ penalizes sample points that do not comply with the conditions of Eq. (3b) by using the $ReLu(\cdot)$ function and adding a small positive constant $\tau_l$ as seen in Eq. (6c).

$$L\left(\hat{B}, B\right) = \alpha L_1 + \beta L_2 \tag{6a}$$

$$L_1 = \frac{1}{m}\sum_{k=1}^{m}\left(\hat{B}(x_k) - B_k\right)^2 \tag{6b}$$

$$L_2 = \frac{1}{N_{l_f}}\sum_{j=1}^{N_{l_f}} ReLu\left(L_f\hat{B}(x_j) + \tau_l\right) \tag{6c}$$

where $L_1$ tracks the mean squared error (MSE) between the target $B$ and the predicted barrier function $\hat{B}$ for all discretized data points $x_k, k = 1, \ldots, m$, in the entire operating region that we sample from, and $L_2$ is the loss function term that aims to satisfy $L_f\hat{B} \leq 0$ for all $x \in \{S_{\mathcal{I}} | L_g\hat{B}(x) = 0\}$, where $N_{l_f}$ is the number of discretized data points that satisfies this condition in the safe region. Since *ReLu* takes the maximum between its argument and 0, i.e., $ReLu(z) = \max\{0, z\}$, $L_2$ penalizes any samples that produce $L_f\hat{B}_j + \tau_l > 0$, therefore forcing $L_f\hat{B}_j \leq 0$ to hold for the applicable points in the safe region. $\alpha > 0$ and $\beta > 0$ are hyperparameters that adjust the weighting of $L_1$ and $L_2$ in the cost function. When $L_2$ has reached 0 during training, then the weights and biases have been optimized in a way that the predicted barrier function $\hat{B}(x)$ satisfies the condition Eq. (3b). To make sure that all conditions of Eq. (3) are satisfied at the end of training, $L_1$ and $L_2$ are evaluated and monitored separately during training, and both $L_1$ and $L_2$ are required to be below a respective threshold value such that the modeling error for $\hat{B}(x)$ is bounded and the negative semi-definiteness of $L_f\hat{B}(x)$ for all $x$ in the safe region with $L_g\hat{B}(x) = 0$ can be shown.

### 3.2. Verification of FNN-based CBF

Upon arriving at an FNN-CBF from the discretized data samples, it is important to demonstrate that the conditions of Eq. (3) in the Definition of CBF are satisfied and that FNN-CBF can be used to design control laws for the continuous nonlinear system of Eq. (1).

#### 3.2.1. Continuity and differentiability

The CBF is continuously differentiable (i.e., a $C^1$ function) by Definition 2, mandating that $\hat{B}(x)$ and $\dot{\hat{B}}(x)$ must be proven to be continuous. As per the universal approximation theorem (Sontag, 1992), with sufficient model complexity, FNNs are capable of modeling any continuous nonlinear functions on a compact set of the state space. In addition, $\hat{B}(x)$ is the output of an FNN that consists of a chain of nonlinear activation functions, i.e., $tanh(\cdot)$, which is a Lipschitz continuous and continuously differentiable function in the compact subset we sample from. Thus, $\hat{B}(x)$ is also Lipschitz continuous and continuously differentiable on the sampled compact subset. In terms of FNN notations, we have shown that the overall hypothesis function class $h(\mathbf{x})$ that maps the FNN inputs $\mathbf{x}$ to the FNN output $\mathbf{y}$ in the form of barrier function value is also a $C^1$ function. It is assumed that the barrier function satisfies the following inequality:

$$\left|\frac{\partial B}{\partial x}\right| \leq r_B(|x|) \tag{7}$$

where $r_B$ is a class $\mathcal{K}$ function similar to $r_V$ in Eq. (2b).

#### 3.2.2. Verification

Training an FNN that minimizes the loss function of Eq. (6) aims to meet the conditions of Eq. (3) in Definition 2 for all discretized points sampled from the compact subsets that we consider, but does not guarantee that the conditions are met for all points in the respective compact subsets. Therefore, the conditions must be verified to hold over the compact subsets in a continuous sense. Similar to the approaches implemented in Bobiti and Lazar (2016), Richards et al. (2018), Jin et al. (2020), we use a Lipschitz method to verify that the decrease condition holds for a candidate function on a finite sample of a bounded set. The following theorem presents the necessary criteria to use this verification technique:

**Theorem 1.** *Consider a compact set $S \subset \mathbf{R}^n$ and let $S_s$ be a finite set sampled from $S$ s.t. $\forall\ x \in S$, there exists at least a pair $(x_s, \delta_{x_s}) \in S_s \times \mathbf{R}_+$ such that $|x - x_s| \leq \delta x_s$. If $F(x_s) \leq -L_F \cdot \delta x_s$ (or respectively $F(x_s) < -L_F \cdot \delta x_s$) holds for all $x_s \in S_s$ , where the Lipschitz constant for the function $F$ is denoted by $L_F > 0$, then $F(x) \leq 0$ (respectively $F(x) < 0$) holds for all $x \in S$ (Bobiti and Lazar, 2016).*

Therefore, by checking the tightened inequality $L_f\hat{B}(x) \leq -L' \cdot \delta x_{\mathcal{X}\backslash\mathcal{H}'},\ \forall\ x \in S_{\mathcal{I}}$, it will be verified that $L_f\hat{B}(x) \leq 0,\ \forall\ x \in \mathcal{X}\backslash\mathcal{H}'$, where $L' > 0$ is the Lipschitz constant for $L_f\hat{B}(x)$, the finite set $S_{\mathcal{I}}$ is sampled from the compact set $\mathcal{X}\backslash\mathcal{H}'$, and $\delta x_{\mathcal{X}\backslash\mathcal{H}'} > 0$ is the discretization grid size (distance between two discretized $x$ points) of the safe set $\mathcal{X}\backslash\mathcal{H}'$. On a similar note, $\hat{B}(x) \leq 0,\ \forall\ x \in \mathcal{X}\backslash\mathcal{H}'$ can be shown to hold by verifying that $\hat{B}(x) \leq -L'' \cdot \delta x_{\mathcal{X}\backslash\mathcal{H}'}\ \forall\ x \in S_{\mathcal{I}}$, where the Lipschitz constant for $\hat{B}$ is denoted by $L''$. Lastly, we show that Eq. (3a) is satisfied by checking $-\hat{B}(x) < -L'' \cdot \delta x_{\mathcal{H}'},\ \forall\ x \in S_{\mathcal{H}'}$,

which is sufficient to verify that $-\hat{B}(x) < 0 \ \forall \ x \in \mathcal{H}'$, thus equivalent to $\hat{B}(x) > 0 \ \forall \ x \in \mathcal{H}'$. These conditions will be checked for all sample points in the respective discretized sets after an FNN model is obtained. More details on the sampling, design, training, and verification of the FNN-CBF can be found in our previous work in Chen et al. (2021).

## 4. FNN generalization error

When we train an FNN model, the model is obtained by minimizing the loss function calculated based on training data samples only. Therefore, there is no information given on the error or performance of the model on new testing data. The generalization error measures the model's ability of making an accurate prediction for new data from the same underlying distribution that has not been seen or studied by the neural network. Using statistical theory commonly used in machine learning, we present an upper bound for the generalization error of the FNN model in predicting the value of the barrier function output.

We first introduce some important preliminary concepts that will be referenced in the development of FNN generalization error bound. Without loss of generality, we let $\mathcal{H}_h$ be the hypothesis class of FNN functions $h(\cdot)$ that map a $d_x$-dimensional input $\mathbf{x} \in \mathbf{R}^{d_x}$ to a $d_y$-dimensional output $\hat{\mathbf{y}} \in \mathbf{R}^{d_y}$. We use $\hat{\mathbf{y}} = h(\mathbf{x})$ to denote the predicted output of the FNN model and $L(\hat{\mathbf{y}}, \mathbf{y})$ to represent the loss function. Here, the loss function can be of many forms; for example, in our case of constructing a barrier function FNN, the loss function is the sum of two loss functions as shown in Eq. (6), where one loss function ($L_1$) assesses the mean squared error between the predicted and the true barrier function output values, and the other loss function ($L_2$) ensures that the Lie derivative properties of the resulting FNN barrier function are met. Nevertheless, in supervised learning where the true output values are known and used during training, the loss function will involve calculating the difference between $\hat{\mathbf{y}}$ and $\mathbf{y}$. The following error definitions are presented for FNN model training.

**Definition 3.** Mohri et al. (2018) Given a function $h$ that predicts $\mathbf{y}$ (output) using $\mathbf{x}$ (input), the generalization error or expected loss / error over an underlying data distribution is $D_d$ is

$$L_{D_d}(h) \triangleq \mathbb{E}[L(h(\mathbf{x}), \mathbf{y})] = \int_{X \times Y} L(h(\mathbf{x}), \mathbf{y}) \rho(\mathbf{x}, \mathbf{y}) d\mathbf{x} d\mathbf{y} \tag{8}$$

where $\rho(\mathbf{x}, \mathbf{y})$ is the joint probability distribution for $\mathbf{x}$ and $\mathbf{y}$, $X$ and $Y$ respectively denote the vector space for all possible inputs and outputs.

In most cases, the joint probability distribution $\rho$ is not known. Therefore, we approximate the expected error by using the empirical error presented as follows:

**Definition 4.** Mohri et al. (2018) Consider a dataset $S_s = (s_1, \ldots, s_m)$, $s_i = (\mathbf{x}_i, \mathbf{y}_i)$, with $m$ number of data samples collected from the underlying data distribution $D_d$, the **empirical risk** or **error** is

$$\hat{\mathbb{E}}_{S_s}[L(h(\mathbf{x}), \mathbf{y})] = \frac{1}{m} \sum_{i=1}^{m} L(h(\mathbf{x}_i), \mathbf{y}_i) \tag{9}$$

In addition, we also need to demonstrate the loss function $L(\hat{\mathbf{y}}, \mathbf{y})$ is locally Lipschitz continuous. In this particular study, the true FNN output is the true barrier function value $B \in \mathbf{R}^1$ that takes the values of either $-1$ or $+1$, thus $|\mathbf{y}| \leq 1$. Since the FNN uses hyperbolic tangent sigmoid $\sigma(z) = \tanh(z) = \frac{2}{1+e^{-2z}} - 1$ as the activation function, the predicted FNN output $\hat{B}$ is also bounded by $|\hat{\mathbf{y}}| \leq 1$. Furthermore, the training of FNN is designed such that it will only stop after $L_2$ in Eq. (6) reaches below a threshold (i.e., $L_f \hat{B}(x) \leq 0 \ \forall \ x \in \{S_{\mathcal{I}} | L_g \hat{B}(x) = 0\}$ is satisfied only when $L_2 \leq \tau_l$, where $\tau_l > 0$ is a small positive constant). Therefore, $L_2$ is also upper bounded. With these considerations, both $L_1$ and $L_2$ loss functions are locally Lipschitz continuous, and the overall loss function $L$ is also locally Lipschitz continuous with the following inequality satisfied for any two predictions:

$$|L(\mathbf{y}, \hat{\mathbf{y}}_2) - L(\mathbf{y}, \hat{\mathbf{y}}_1)| \leq L_r |\hat{\mathbf{y}}_2 - \hat{\mathbf{y}}_1| \tag{10}$$

where $L_r$ denotes the local Lipschitz constant for the loss function $L$.

### 4.1. Rademacher complexity

We use empirical Rademacher complexity to bound the generalization error as it is commonly used in machine learning theory to quantify the richness of a class of functions. The Rademacher complexity is defined as follows:

**Definition 5.** Mohri et al. (2018) Given a dataset of $m$ samples $S_s = \{s_1, \ldots, s_m\}$, and a hypothesis class $\mathcal{F}$ of scalar-valued functions, the empirical Rademacher complexity of $\mathcal{F}$ is defined as:

$$\mathcal{R}_{S_s}(\mathcal{F}) = \mathbb{E}_{\boldsymbol{\epsilon}} \left[ \sup_{f \in \mathcal{F}} \frac{1}{m} \sum_{i=1}^{m} \epsilon_i f(s_i) \right] \tag{11}$$

where $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_m)^T$ contains Rademacher random variables $\epsilon_i$ that are independent and identically distributed (i.i.d.) and satisfy $\mathbb{P}(\epsilon_i = -1) = \mathbb{P}(\epsilon_i = 1) = 0.5$.

For the hypothesis class $\mathcal{H}_h$ of vector-valued functions $h \in \mathbf{R}^{d_y}$, it also satisfies the inequality shown in the following lemma:

**Lemma 1** (c.f. Corollary 4 in (Maurer, 2016))**.** *Given a hypothesis class $\mathcal{H}_h$ of vector-valued functions $h \in \mathbf{R}^{d_y}$, and a dataset of $m$ samples $S_s = \{s_1, \ldots, s_m\}$. Consider the loss function $L(\cdot)$ which is a $L_r$-Lipschitz function mapping $h \in \mathbf{R}^{d_y}$ to $\mathbf{R}$, then we have*

$$\mathbb{E}_{\boldsymbol{\epsilon}} \left[ \sup_{h \in \mathcal{H}_h} \sum_{i=1}^{m} \epsilon_i L(h(\mathbf{x}_i), \mathbf{y}_i) \right] \leq \sqrt{2} L_r \mathbb{E}_{\boldsymbol{\epsilon}} \left[ \sup_{h \in \mathcal{H}_h} \sum_{i=1}^{m} \sum_{k=1}^{d_y} \epsilon_{ik} h_k(\mathbf{x}_i) \right] \tag{12}$$

*where $\epsilon_{ik}$ is a $m \times d_y$ matrix consisting of independent Rademacher variables, and $h_k(\cdot)$ denotes the $k$-th component of the vector-valued function $h(\cdot)$. For simplicity, the subscript $\boldsymbol{\epsilon}$ on the expectation will be omitted for the remainder of the manuscript.*

The following bound (Maurer, 2016) can be derived to simplify the bound in terms of vector-valued functions to one in terms of scalar-value functions:

$$\mathbb{E} \left[ \sup_{h \in \mathcal{H}_h} \sum_{i=1}^{m} \sum_{k=1}^{d_y} \epsilon_{ik} h_k(\mathbf{x}_i) \right] \leq \sum_{k=1}^{d_y} \mathbb{E} \left[ \sup_{h \in \mathcal{H}_{h,k}} \sum_{i=1}^{m} \epsilon_i h(\mathbf{x}_i) \right] \tag{13}$$

where $\mathcal{H}_{h,k}$, $k = 1, \ldots, d_y$ represent scalar-valued function classes for the components of the vector-valued function class $\mathcal{H}_h$ for a network of $d$ layers. We derive the bound for empirical Rademacher complexity in terms of scalar-valued function class first, then use Eq. (13) to develop the bound for vector-valued functions.

### 4.2. Generalization error bound of FNN

Consider the class of loss functions associated with the function class $\mathcal{H}_h$:

$$\mathcal{G} = \{g_L : (\mathbf{x}, \mathbf{y}) \to L(h(\mathbf{x}), \mathbf{y}), h \in \mathcal{H}_h\} \tag{14}$$

where $\mathbf{y}$ is the true FNN output vector, $\mathbf{x}$ is the FNN input vector, and $h(\mathbf{x})$ represents the predicted FNN output vector. We have the

following lemma to upper bound the generalization error using the Rademacher complexity of the family of loss functions $\mathcal{R}_{S_s}(\mathcal{G})$.

**Lemma 2** (c.f. Theorem 3.3 in Mohri et al. (2018)). *Given a data set of m number of i.i.d samples, the following inequality holds for all $g_L \in \mathcal{G}$ over the sample space $S_s = (s_i)$, $s_i = (\mathbf{x}_i, \mathbf{y}_i)$ with probability of at least $1-\delta$:*

$$\mathbb{E}[g_L(\mathbf{x},\mathbf{y})] \leq \frac{1}{m}\sum_{i=1}^{m} g_L(\mathbf{x}_i,\mathbf{y}_i) + 2\mathcal{R}_{S_s}(\mathcal{G}) + 3\sqrt{\frac{\log(\frac{2}{\delta})}{2m}} \tag{15}$$

Interested readers may refer to Wu et al. (2021) and Mohri et al. (2018) for the full proof of this lemma. The RHS of this inequality includes three terms, the sum of which specifies the upper bound for the FNN generalization error. These three terms represent the empirical loss based on the sample dataset $S_s$, the Rademacher complexity, and an error term that depends on the sample size and confidence $\delta$. We further bound the Rademacher complexity such that the upper bound of the generalization error can be quantified by known specific values such as the sample size $m$, confidence $\delta$, neural network depth $d$, input dimension $d_x$, and upper bounds on the input vector $B_X$ and on the weight matrices $B_W$.

We first consider the hypothesis class $\mathcal{H}_{h,k}$ of scalar-valued functions, where $k$ represents components of the vector-valued function class $\mathcal{H}_h$. For the scalar-valued function class $\mathcal{H}_{h,k}$, the following lemma is presented to upper-bound the scaled empirical Rademacher complexity. We will later use this lemma to derive the upper bound for the empirical Rademacher complexity for the vector-valued hypothesis function class $\mathcal{H}_h$.

**Lemma 3** (c.f. Lemma 4 in Wu et al. (2021)). *With $\lambda > 0$, the scaled empirical Rademacher complexity $m\mathcal{R}_{S_s}(\mathcal{H}_{h,k}) = \mathbb{E}[\sup_{h\in\mathcal{H}_{h,k}} \sum_{i=1}^{m} \epsilon_i h(\mathbf{x}_i)]$ satisfies the following inequality:*

$$\begin{aligned} m\mathcal{R}_{S_s}(\mathcal{H}_{h,k}) &= \mathbb{E}\left[\sup_{h\in\mathcal{H}_{h,k}} \sum_{i=1}^{m} \epsilon_i h(\mathbf{x}_i)\right] \\ &= \frac{1}{\lambda}\log\exp\left(\lambda\mathbb{E}\left[\sup_{h\in\mathcal{H}_{h,k}} \sum_{i=1}^{m} \epsilon_i h(\mathbf{x}_i)\right]\right) \\ &\leq \frac{1}{\lambda}\log\left(\mathbb{E}\left[\sup_{h\in\mathcal{H}_{h,k}} \exp(\lambda\sum_{i=1}^{m} \epsilon_i h(\mathbf{x}_i))\right]\right) \end{aligned} \tag{16}$$

We further specify the upper bound of the Rademacher complexity by breaking down the function $h(\mathbf{x}_i)$; this is done through a "peeling" approach to "peel" off the weights and activation functions of the FNN model layer by layer. Here, due to the unique application of the FNN model we construct, all the activation functions are $tanh(\cdot)$ in order to polarize the results to $+1$ and $-1$ values. We present the following lemma, which is modified from Lemma 2 in (Golowich et al., 2018), to demonstrate this peeling step inside a convex, monotonically increasing function (such as $exp(\cdot)$) for a 1-Lipschitz activation function $\sigma(\cdot)$ that satisfies $\sigma(0) = 0$ (such as $tanh(\cdot)$).

**Lemma 4** (c.f. Lemma 2 in Golowich et al. (2018)). *Given any vector-valued function class $\mathcal{N}$ with a 1-Lipschitz continuous activation function $\sigma(\cdot)$ that satisfies $\sigma(0) = 0$ applied element-wise, and a convex and monotonically increasing function $p : \mathbf{R} \to \mathbf{R}_+$, the following inequality holds:*

$$\mathbb{E}\left[\sup_{||W||_{1,\infty}\leq B_W, \nu\in\mathcal{N}} p\left(||\sum_{i=1}^{m} \epsilon_i \sigma(W\nu(\mathbf{x}_i))||_\infty\right)\right]$$

$$\leq 2\mathbb{E}\left[\sup_{\nu\in\mathcal{N}} p\left(B_W||\sum_{i=1}^{m} \epsilon_i \nu(\mathbf{x}_i)||_\infty\right)\right] \tag{17}$$

Lemma 4 holds for the vector-valued function class $\nu \in \mathcal{N}$ (or equivalently $h \in \mathcal{H}_h$), and therefore also holds for the scalar-valued function class $\nu \in \mathcal{N}_k$ (or equivalently $h \in \mathcal{H}_{h,k}$), where $k$ represents the $k$-th component of the vector-valued function class. Following Lemma 4, we now reference Theorem 2 in Golowich et al. (2018) to derive a bound on the Rademacher complexity for the scalar-valued FNN function class $\mathcal{H}_{h,k}$, as presented in Lemma 5. The full proof of Lemma 5 can be found in Golowich et al. (2018). First, Eq. (16) is used as a starting point to provide an inequality involving the scaled Rademacher complexity for the scalar-valued function class $\mathcal{H}_{h,k}$ and the scalar-valued hypothesis function $h(\mathbf{x}) \in \mathcal{H}_{h,k}$, which provides the predicted output in the output layer. Since the function $exp(\cdot)$ in Eq. (16) qualifies as a convex, monotonically increasing function, we can apply Lemma 4 repetitively to Eq. (16) by "peeling" off the neural network layer by layer, starting from $h(\mathbf{x})$ in the output layer. The function $p(\cdot)$ in Eq. (17) refers to $exp(\cdot)$, and the scalar-valued functions $\nu \in \mathcal{N}_k$ refer to subnetworks of the FNN from the input layer up to the layer being "peeled". The resulting upper bound on the Rademacher complexity for the scalar-valued function class $\mathcal{H}_{h,k}$ is presented in Lemma 5 and can be represented in terms of FNN input bound, weight matrix bounds, FNN depth, sample size, and FNN input dimension.

**Lemma 5** (c.f. Theorem 2 in Golowich et al. (2018)). *Given neural networks with depth d and a class of scalar-valued functions $\mathcal{H}_{h,k}$ where $||W_l||_{1,\infty} \leq B_W$ for all $l = 1,\ldots,d$, and Assumptions 1 - 4 satisfied, the following inequality holds:*

$$\mathcal{R}_{S_s}(\mathcal{H}_{h,k}) \leq \frac{2B_X(B_W)^d\sqrt{d+1+log(d_x)}}{\sqrt{m}} \tag{18}$$

*Interested readers may refer to Section 7 of Golowich et al. (2018) for the proof of this theorem.*

The above lemma presents the Rademacher complexity upper bound for the scalar-valued functions $\mathcal{H}_{h,k}$, $k = 1,\ldots,d_y$, for the $k$-th component of the vector-valued function class $\mathcal{H}_h$. Now we will derive the generalization error bound for the loss function class associated with the vector-valued hypothesis FNN function class $\mathcal{H}_h$. We use Eqs. (12),(13) to derive the following theorem:

**Theorem 2** (c.f. Theorem 1 in Wu et al. (2021)). *Consider the dataset $S_s$ consisting of m i.i.d. data samples and the class of loss functions associated with the vector-valued FNN hypothesis class $\mathcal{H}_h$ satisfying Assumptions 1 - 4. With probability of at least $1-\delta$, we have the following inequality:*

$$\mathbb{E}[g_L(\mathbf{x},\mathbf{y})] \leq \mathcal{O}\left(L_r d_y \frac{B_X(B_W)^d\sqrt{d+1+log(d_x)}}{\sqrt{m}}\right)$$

$$+3\sqrt{\frac{\log(\frac{2}{\delta})}{2m}} + \frac{1}{m}\sum_{i=1}^{m} g_L(\mathbf{x}_i,\mathbf{y}_i) \tag{19}$$

*where $B_X$ is the upper bound on FNN inputs defined in Eq. (1), $B_W$ is the upper bound on FNN weight matrices as stated in Eq. (2), $L_r$ is the local Lipschitz constant for the loss function $L(\cdot)$ as defined in Eq. (10), $d_x$ is the FNN input dimension, $d_y$ is the FNN output dimension.*

**Proof.** Using Eqs. (12),(13), we can derive the following upper bound for the loss function $L(h(\mathbf{x}_i),\mathbf{y}_i)$ with $h(\mathbf{x}_i)$ being vector-valued functions:

$$\mathcal{R}_{S_s}(\mathcal{G}) = \mathbb{E}\left[\sup_{h\in\mathcal{H}_h} \frac{1}{m}\sum_{i=1}^{m} \epsilon_i L(h(\mathbf{x}_i),\mathbf{y}_i)\right]$$

$$\leq \sqrt{2}L_r\mathbb{E}\left[\sup_{h\in\mathcal{H}_h}\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{d_y}\epsilon_{ik}h_k(\mathbf{x}_i)\right]$$
$$\leq \sqrt{2}L_r\frac{1}{m}\sum_{k=1}^{d_y}\mathbb{E}\left[\sup_{h\in\mathcal{H}_{h,k}}\sum_{i=1}^{m}\epsilon_i h(\mathbf{x}_i)\right] \quad (20)$$

Using the definition of Rademacher complexity for the scalar-valued function class $\mathcal{H}_{h,k}$, we have the following:

$$\sqrt{2}L_r\frac{1}{m}\sum_{k=1}^{d_y}\mathbb{E}\left[\sup_{h\in\mathcal{H}_{h,k}}\sum_{i=1}^{m}\epsilon_i h(\mathbf{x}_i)\right]$$
$$= \sqrt{2}L_r\frac{1}{m}\sum_{k=1}^{d_y}m\mathcal{R}_{S_s}(\mathcal{H}_{h,k}) = \sqrt{2}L_r\sum_{k=1}^{d_y}\mathcal{R}_{S_s}(\mathcal{H}_{h,k}) \quad (21)$$

Therefore, using Eq. (18), we can derive the bound on the Rademacher complexity of the loss function as follows:

$$\mathcal{R}_{S_s}(\mathcal{G}) \leq \sqrt{2}L_r\sum_{k=1}^{d_y}\mathcal{R}_{S_s}(\mathcal{H}_{h,k})$$
$$\leq \sqrt{2}L_r\sum_{k=1}^{d_y}\frac{2B_X(B_W)^d\sqrt{d+1+log(d_x)}}{\sqrt{m}}$$
$$\leq 2\sqrt{2}L_r d_y\frac{B_X(B_W)^d\sqrt{d+1+log(d_x)}}{\sqrt{m}} \quad (22)$$

Lastly, we can substitute Eq. (22) into Eq. (15), and obtain the generalization error bound as seen in Eq. (19). □

### 4.3. Implications of generalization error bound for different loss functions

As seen in Eq. (6), there are two parts to the loss function of the FNN, and each part is being monitored separately during training. As explained in Section 4, both loss functions $L_1$ and $L_2$ are locally Lipschitz continuous functions satisfying the following inequalities:

$$|L_1(\mathbf{y},\hat{\mathbf{y}}_2) - L_1(\mathbf{y},\hat{\mathbf{y}}_1)| \leq L_{r1}|\hat{\mathbf{y}}_2 - \hat{\mathbf{y}}_1| \quad (23a)$$

$$|L_2(\hat{\mathbf{y}}_2) - L_2(\hat{\mathbf{y}}_1)| \leq L_{r2}|\hat{\mathbf{y}}_2 - \hat{\mathbf{y}}_1| \quad (23b)$$

where $L_{r1}$ and $L_{r2}$ denote the local Lipschitz constant for loss functions $L_1$ and $L_2$ respectively. Note that $L_1$ is a function assessing the MSE between the true output $\mathbf{y}$ and the predicted output $\hat{\mathbf{y}}$, and $L_2$ is a function of the predicted output $\hat{\mathbf{y}}$ only (the explicit form of $\frac{\partial B}{\partial x}$, hence $L_f B(x)$, are not known ahead of time).

Therefore, we can develop the generalization error bound with respect to each loss function, and explain their respective implications. Here, we replace the general notations of FNN inputs $\mathbf{x}$ and output $\mathbf{y}$ with the specific variables under consideration in our case, which include states of the nonlinear system of Eq. (1) $x$ as the inputs, barrier function value $B$ as the true output, and $\hat{B}(x)$ as the predicted output. The expected loss of $L_1$ is upper bounded by the following inequality with probability of at least $1-\delta$:

$$\mathbb{E}[L_1(\hat{B}(x),B)] \leq \mathcal{O}\left(L_{r1}d_y\frac{B_X(B_W)^d\sqrt{d+1+log(d_x)}}{\sqrt{m}}\right)$$
$$+3\sqrt{\frac{\log(\frac{2}{\delta})}{2m}} + \frac{1}{m}\sum_{i=1}^{m}L_1(\hat{B}(x_i),B_i) \quad (24)$$

Since $L_1$ evaluates error between true FNN output (i.e., $B$) and predicted FNN output (i.e., $\hat{B}(x)$) in terms of MSE, the upper bound on $|\hat{B}-B|$ is:

$$|\hat{B}-B| \leq \sqrt{\mathcal{O}\left(L_{r1}d_y\frac{B_X(B_W)^d\sqrt{d+1+log(d_x)}}{\sqrt{m}}\right)+3\sqrt{\frac{\log(\frac{2}{\delta})}{2m}}+\frac{1}{m}\sum_{i=1}^{m}L_1(\hat{B}(x_i),B_i)} \quad (25)$$

We can further develop a bound on the value of $\hat{B}(x)$, which holds with probability of at least $1-\delta$ as follows:

$$|\hat{B}| \leq |\hat{B}+B-B|$$
$$\leq |B| + |\hat{B}-B|$$
$$\leq |B| + \sqrt{\mathcal{O}\left(L_{r1}d_y\frac{B_X(B_W)^d\sqrt{d+1+log(d_x)}}{\sqrt{m}}\right)+3\sqrt{\frac{\log(\frac{2}{\delta})}{2m}}+\frac{1}{m}\sum_{i=1}^{m}L_1(\hat{B}(x_i),B_i)} \quad (26)$$

Given the conditions of Eqs. (3a) and (3c), the true barrier function $B$ take values of $+1$ for unsafe $x$, and $-1$ for safe $x$; therefore, $|B| \leq 1$ for all $x$ in the operating region. In order to ensure that $\hat{B}$ satisfies $\hat{B} \leq 0$ for all safe $x$, and $\hat{B} > 0$ for all unsafe $x$, the upper bound on the modeling error of the barrier function output must be less than 1, thus,

$$\sqrt{\mathcal{O}\left(L_{r1}d_y\frac{B_X(B_W)^d\sqrt{d+1+log(d_x)}}{\sqrt{m}}\right)+3\sqrt{\frac{\log(\frac{2}{\delta})}{2m}}+\frac{1}{m}\sum_{i=1}^{m}L_1(\hat{B}(x_i),B_i)} \leq 1 \quad (27)$$

The FNN model must be trained and built by selecting the appropriate number of samples $m$, the depth of the network $d$, the bound on the weight matrices $B_W$ such that this bound on the modeling error is satisfied.

Moreover, the generalization error bound of $L_2$ represents the upper bound of the expected value of $L_2$ when applied on testing data that has not been studied by the FNN. The generalization error bound of $L_2$ can be written as follows:

$$\mathbb{E}[L_2(\hat{B}(x)] \leq \mathcal{O}\left(L_{r2}d_y\frac{B_X(B_W)^d\sqrt{d+1+log(d_x)}}{\sqrt{m}}\right)$$
$$+3\sqrt{\frac{\log(\frac{2}{\delta})}{2m}} + \frac{1}{m}\sum_{i=1}^{m}L_2(\hat{B}(x_i)) \quad (28)$$

where the term $\frac{1}{m}\sum_{i=1}^{m}L_2(\hat{B}(x_i))$ represents the empirical loss of $L_2$ resulting from $m$ data samples from the training dataset. As described in Section 3.1, we monitor $L_2$ during training and only stop training when $L_2$ reaches 0 for all training data samples. Therefore, $\frac{1}{m}\sum_{i=1}^{m}L_2(\hat{B}(x_i)) = 0$. Furthermore, by the law of large numbers, with sufficiently large number of data sample size, the sample mean can sufficiently approximate the real expected value. In this case, we can use the testing dataset empirical loss to approximate the expectation of $L_2$, which assesses $ReLu(L_f\hat{B}(x)+\tau_l)$ for $x$ values that have not been studied by the FNN. We can further simplify Eq. (28) to the following form by utilizing the fact that the empirical loss of $L_2$ on the training dataset is 0:

$$\mathbb{E}\left[\frac{1}{N_{I_f}^{test}}\sum_{i=1}^{N_{I_f}^{test}}ReLu\left(L_f\hat{B}(x_i)+\tau_l\right)\right]$$
$$\leq \mathcal{O}\left(L_{r2}d_y\frac{B_X(B_W)^d\sqrt{d+1+log(d_x)}}{\sqrt{m}}\right)+3\sqrt{\frac{\log(\frac{2}{\delta})}{2m}} \quad (29a)$$

$$\mathbb{E}\left[ReLu(L_f\hat{B}(x)+\tau_l)\right] \leq \mathcal{O}\left(L_{r2}d_y\frac{B_X(B_W)^d\sqrt{d+1+log(d_x)}}{\sqrt{m}}\right)$$
$$+3\sqrt{\frac{\log(\frac{2}{\delta})}{2m}} \quad (29b)$$

where $x_i$ for $i = 1, \ldots, N_{l_f}^{test}$ represents safe states in the testing dataset at which $L_g\hat{B}(x_i) = 0$. In order to meet the condition of Eq. (3b) for testing data points that have not been previously studied by the FNN, the following inequality must hold:

$$\mathcal{O}\left(L_{r2}d_y\frac{B_X(B_W)^d\sqrt{d+1+log(d_x)}}{\sqrt{m}}\right)+3\sqrt{\frac{\log(\frac{2}{\delta})}{2m}}-\tau_l\leq 0 \quad (30)$$

By carefully choosing the number of layers to the FNN (depth $d$), the number of training sample size $m$, the upper bounds on weight matrices $B_W$, as well as the upper bound on the input vector $B_X$ by selecting the range of states considered in the compact set in state space appropriately, we build a FNN that satisfies Eq. (30), and in turn, ensures that $L_f\hat{B}(x) \leq 0$ in the operating region for which we consider the states are constrained within with probability $1-\delta$.

## 5. Probabilistic stabilization and safety via Control Lyapunov-Barrier Function

The Control Lyapunov-Barrier Function (CLBF) in the form of a weighted average of CLF and CBF was proposed in Romdlony and Jayawardhana (2016), and it shows that when a CLBF exists for the system of Eq. (1), there exists a controller $u = \Phi(x)$ that keeps the closed-loop state bounded within a level set of the CLBF and outside of the unsafe set $\mathcal{D}$ for all times for any initial condition $x_0 \in \mathcal{X}_0$. This work is further extended in Wu and Christofides (2019), Wu et al. (2019a) to account for input constraints in the system and the constrained CLBF was presented. Furthermore, a constrained CLBF-MPC where the prediction model inside the MPC was developed using an ensemble of Recurrent Neural Network (RNN) models was proposed in Wu and Christofides (2020). Based on this work, we proposed a machine-learning-based CLBF-MPC in Chen et al. (2021) where the CBF is built using an FNN model to characterize the safety status of the states inside the operating region, and the MPC uses an RNN model for its predictions. In this work, we provide statistical analysis on the probability of stabilization and safety of a CLBF-based controller where the CBF is built using an FNN, first under the control law $u = \Phi(x) \in U$ for the nonlinear system of Eq. (1), then under the CLBF-MPC where MPC uses the first-principles model in the form of ODE as described by Eq. (1) to predict future states. The FNN-CBF $\hat{B}$ can be shown to meet the conditions outlined in Eq. (3) in probability with proper model construction, parameter selection, and post-training verification. Therefore, it can be readily used as a valid CBF in the design of CLBF. The constrained CLBF built using the FNN-CBF $\hat{B}$ is defined as follows:

**Definition 6.** Given a set of unsafe points in state-space $\mathcal{D}$, a proper, lower-bounded and $\mathcal{C}^1$ function $\hat{W}(x) : \mathbf{R}^n \to \mathbf{R}$ is a constrained CLBF if $\hat{W}(x)$ has a minimum at the origin and also satisfies the following properties:

$$\hat{W}(x) > \rho, \quad \forall\, x \in \mathcal{D} \subset \phi_{uc} \quad (31a)$$

$$\left|\frac{\partial \hat{W}(x)}{\partial x}\right| \leq r_W(|x|) \quad (31b)$$

$$L_f\hat{W}(x) < 0,$$
$$\forall\, x \in \{z \in \phi_{uc} \backslash (\mathcal{D} \cup \{0\} \cup \mathcal{X}_e) \mid L_g\hat{W}(z) = 0\} \quad (31c)$$

$$\mathcal{U}_\rho := \{x \in \phi_{uc} \mid \hat{W}(x) \leq \rho\} \neq \emptyset \quad (31d)$$

$$\overline{\phi_{uc} \backslash (\mathcal{D} \cup \mathcal{U}_\rho)} \cap \overline{\mathcal{D}} = \emptyset \quad (31e)$$

where $f$ and $g$ are from the nonlinear model in Eq. (1), $\rho \in \mathbf{R}$ is a constant, $r_W$ is a class $\mathcal{K}$ function, $\mathcal{X}_e := \{x \in \phi_{uc} \backslash (\mathcal{D} \cup \{0\}) \mid \partial\hat{W}(x)/\partial x = 0\}$ is a set of states for the nonlinear model of Eq. (1) where $L_f\hat{W}(x) = 0$ (for $x \neq 0$) due to $\partial\hat{W}(x)/\partial x = 0$. If $\hat{W}(x)$ exists for the nonlinear system of Eq. (1) as defined in Eq. 6, then there exists a control law $u = \Phi(x) \in U$ such that the origin of the system is rendered asymptotically stable within a region $\phi_{uc}$, which is defined as the union of the origin, and the set $\mathcal{X}_e$, and the set for which the time-derivative of $\hat{W}(x)$ is negative with constrained inputs: $\phi_{uc} = \{x \in \mathbf{R}^n \mid \{0\} \cup \mathcal{X}_e \cup \dot{\hat{W}}(x(t), \Phi(x)) = L_f\hat{W} + L_g\hat{W} \cdot u < -\alpha_W|\hat{W}(x) - \hat{W}(0)|, u = \Phi(x) \in U\}$, and $\alpha_W > 0$ is a real constant used to characterize the set $\phi_{uc}$. An example of such control law $\Phi(x)$ takes the form of the Lyapunov-based universal Sontag law (Lin and Sontag, 1991) with the Lyapunov function $V(x)$ replaced by the CLBF $\hat{W}(x)$; details can be found in Wu et al. (2018, 2019a), Wu and Christofides (2019).

### 5.1. Design of constrained CLBF

The design of CLBF can be carried out following the practical design guidelines in Wu et al. (2019a), by first designing valid CLF and CBF that meet their conditions outlined in Eq. (2) and Eq. (3), respectively. This design method is further expanded and proven in Chen et al. (2021) in the case of FNN-based CBF and RNN-based process model, and it was shown that through a FNN-CBF $\hat{B}(x)$ that meets all its required conditions, the resulting machine-learning based $\hat{W}(x)$ has a global minimum at the origin and is able to meet all its requirements of Eq. (31). The proof for the following proposition can be found in Wu and Christofides (2019) and Chen et al. (2021) and will be omitted here. In this work, we have introduced the statistical analysis on the generalization error of the FNN-CBF $\hat{B}$. Accounting for the general expected error of $\hat{B}$, the FNN-CBF $\hat{B}$ is shown to meet all the requirements of Eq. (3) with probability of $1-\delta$ if the two conditions on the modeling error bound shown in Eqs. (27) and (30) are met. Therefore, the properties of the resulting CLBF $\hat{W}$ as well as the associated safety and stabilizability properties of the CLBF-based controller will also hold with probability $1-\delta$.

**Proposition 1.** *Consider the $\mathcal{C}^1$ FNN-CBF $\hat{B}(x) : \mathbf{R}^n \to \mathbf{R}$, trained using the dataset $S_s$ consisting $m$ i.i.d data samples satisfying Assumptions 1–4, and has a resulting loss function errors constrained by Eqs. (27) and (30). Given an open set $\mathcal{D}$ of unsafe states for the system of Eq. (1), assume that there exists a $\mathcal{C}^1$ CLF $V : \mathbf{R}^n \to \mathbf{R}_+$, such that the following conditions hold:*

$$\mathcal{D} \subset \mathcal{H} \subset \mathcal{H}' \subset \phi_{uc}, \ 0 \notin \mathcal{H}, \ 0 \notin \mathcal{H}' \quad (32)$$

$$\hat{B}(x) = -\eta < 0, \ \forall x \in \mathbf{R}^n \backslash \mathcal{H}'; \ \hat{B}(x) > 0, \ \forall x \in \mathcal{H}' \quad (33)$$

*where $\mathcal{H}$ and $\mathcal{H}'$ are both compact and connected sets within $\phi_{uc}$, and $\mathcal{H}'$ encloses $\mathcal{H}$ with sufficient margin accounting for modeling errors in $\hat{B}(x)$. Consider $\hat{W}(x)$ designed as $\hat{W}(x) := V(x) + \mu\hat{B}(x) + \nu$, and satisfies:*

$$\left|\frac{\partial \hat{W}(x)}{\partial x}\right| \leq r_W(|x|) \quad (34)$$

$$L_f\hat{W}(x) < 0,$$
$$\forall\, x \in \{z \in \phi_{uc} \backslash (\mathcal{D} \cup \{0\} \cup \mathcal{X}_e) \mid L_g\hat{W}(z) = 0\} \quad (35)$$

$$\mu > \frac{c_2c_3 - c_1c_4}{\eta}, \quad (36a)$$

$$\nu = \rho - c_1c_4, \quad (36b)$$

$$c_3 := \max_{x\in\partial\mathcal{H}'} |x|^2, \tag{36c}$$

$$c_4 := \min_{x\in\partial\mathcal{D}} |x|^2 \tag{36d}$$

*then, with probability of at least $1-\delta$, the control law $\Phi(x)\in U$ (Lyapunov-based Sontag control law with $V(x)$ replaced by $\hat{W}(x)$) guarantees that, for any initial state $x_0\in\phi_{uc}\backslash\mathcal{D}_{\mathcal{H}'}$, where $\mathcal{D}_{\mathcal{H}'} := \{x\in\mathcal{H}' \mid \hat{W}(x)>\rho\}$, the state is bounded in $\phi_{uc}\backslash\mathcal{H}$ and does not enter the unsafe region $\mathcal{H}$ for all $t>0$.*

**Proof.** Through the selection of parameters $\mu$ and $\nu$, the conditions of Eqs. (31a) and (31e) can be met. The proofs for these two conditions are shown in Wu et al. (2019a) and will be omitted here. We will focus on how the conditions of Eqs. (31b) and (31c) can be met. Given that Eqs. (27) and (30) are met, $\hat{B}(x)$ satisfies the CBF properties presented in Eq. (3) with probability at least $1-\delta$. From Eqs. (2b) and (7), as well as the way the CLBF is constructed $\hat{W}(x) := V(x)+\mu\hat{B}(x)+\nu$, we have the following:

$$\begin{aligned}\left|\frac{\partial\hat{W}(x)}{\partial x}\right| &= \left|\frac{\partial V}{\partial x}+\mu\frac{\partial\hat{B}}{\partial x}\right| \\ &\le r_V(|x|)+\mu r_B(|x|) \\ &\le r_W(|x|)\end{aligned} \tag{37}$$

where $r_W$, as the weighted sum of two class $\mathcal{K}$ functions $r_V$ and $r_B$, is also a class $\mathcal{K}$ function. Thus, it is shown that Eq. (31b) is satisfied. Similarly, for all $x\in\{z\in\phi_{uc}\backslash(\mathcal{D}\cup\{0\}\cup\mathcal{X}_e) \mid L_g\hat{W}(z)=0\}$, Eq. (31c) can be also shown to hold with the following derivation:

$$L_f\hat{W}(x) \quad = L_fV(x)+\mu L_f\hat{B}(x) < 0 \tag{38}$$

Thus, Eqs. (31b) and (31c) are both satisfied. In addition, the global minimum of $V(x)$ is at the origin, i.e., $V(0)=0$, and $V(x)>0$ for all $x\in\mathbf{R}^n\backslash\{0\}$. With a sufficiently small modeling error as characterized by its generalization error bound, $\hat{B}(x)=-1$ for all $x\in\phi_{uc}\backslash\mathcal{H}'$, where $\{0\}\in\phi_{uc}\backslash\mathcal{H}'$, and $\hat{B}(x)=+1$ for all $x\in\mathcal{H}'$ in probability. Hence, $\hat{B}(x)$ also has a global minimum at the origin in probability. Since $\hat{W}(x)$ is a weighted average of $V(x)$ and $\hat{B}(x)$, the global minimum of $\hat{W}(x)$ is at the origin. Therefore, it has been demonstrated that a CLBF $\hat{W}(x)$ and the control law $u=\Phi(x)\in U$ exist that satisfy all conditions of Eq. (31) with probability $1-\delta$, and guarantee the safety and asymptotic stability of the states for all $x_0\in\phi_{uc}\backslash\mathcal{D}_{\mathcal{H}'}$. □

We specify the set of initial conditions considered in our study as $\mathcal{U}_\rho$, which is a level set of $\hat{W}(x)$ as described in Eq. (31d). Since $\dot{\hat{W}}(x)=0$ for $x=0$ and $x=x_e\in\mathcal{X}_e$, and $\dot{\hat{W}}(x)<0$ within the set $\phi_{uc}\backslash(\mathcal{X}_e\cup 0)$ under the control law $u=\Phi(x)\in U$, it holds that $\dot{\hat{W}}(x)\le 0$ for all $x\in\mathcal{U}_\rho$. We know that $\hat{W}(x)$ is a proper function, therefore the level set of $\hat{W}(x)$, $\mathcal{U}_\rho$, is a compact, forward invariant set. For any initial condition $x_0\in\mathcal{U}_\rho$, the closed-loop state $x(t)$ is bounded in $\mathcal{U}_\rho$ under the continuous control law $u=\Phi(x)\in U$. Furthermore, since the set $\mathcal{U}_\rho$ has no intersection with the set $\mathcal{D}_{\mathcal{H}'}$, the closed-loop state will not enter the unsafe set $\mathcal{D}_{\mathcal{H}'}$ characterized by Proposition 1.

For bounded unsafe sets (e.g., the entire unsafe region occurs as an obstacle in the middle of the operating region), there are stationary points in state space (in addition to the origin), denoted as $x_e\in\mathcal{X}_e$ where $\hat{W}=0$, which can be considered as saddle points. When states reach these stationary points, the continuous controller $u=\Phi(x)\in U$ is incapable of steering the states away from these points and they will remain there and become trapped. Thus, we design discontinuous control actions $u=\bar{u}(x)\in U$ that can drive the states away from $x_e$ in a path of decreasing $\hat{W}(x)$. Once the states leave $x_e$ under $\bar{u}(x)$, then the controller $u=\Phi(x)\in U$ is able to continue driving the state towards the origin asymptotically since $\dot{\hat{W}}(x)<0$ for all $x\in\mathcal{U}_\rho\backslash(\mathcal{X}_e\cup 0)$. In the case of unbounded unsafe sets, the origin will be the only stationary point in state-space, therefore the CLBF-based control law $u=\Phi(x)\in U$ is able to ensure asymptotic stability and safety.

### 5.2. Sample-and-hold implementation of CLBF-based controller

We have shown that if there exists a constrained CLBF $\hat{W}$ built from FNN-CBF $\hat{B}$ that meets the conditions of Eq. (31) and a set of control law $u=\Phi(x)\in U$ that is continuously implemented, the closed-loop state can be maintained within the safe region for all times. This CLBF-based control law $u=\Phi(x)\in U$ is used to design CLBF-based constraints in MPC. As the MPC is executed every sampling period $\Delta$, the control law will be implemented in a sample-and-hold manner. Therefore, we will now discuss the impact of sample-and-hold application of control actions on the probabilistic stability and safety of the nonlinear system of Eq. (1).

We consider the region $\mathcal{U}_\rho\backslash(\mathcal{U}_{\rho_s}\cup\mathcal{B}_\delta(x_e))$, where $\rho_s<\rho_{\min}<\rho$, and prove that for all $x(t_k)$ in this region, $\dot{W}(x(t),u(t))<-\epsilon$ where $u(t)$ is applied in a sample-and-hold manner $u(t)=u(t_k)=\Phi(x(t_k))$, $\forall\ t\in[t_k,t_k+\Delta')$. Since this region is a bounded region within $\phi_{uc}$ and the functions $f(\cdot)$ and $g(\cdot)$ are continuous, we have the following inequalities:

$$\dot{\hat{W}}(x(t_k)) < -\alpha_W|\hat{W}(x)-\hat{W}(0)| < -\alpha_W\rho_0 \tag{39a}$$

$$|x(t)-x(t_k)|\le k_1\Delta',\ \ \forall\ t\in[t_k,t_k+\Delta') \tag{39b}$$

where $k_1$ is a positive real number and $\Delta'>0$ represents a sampling period, where the sampling period of the CLBF-based controller and CLBF-MPC $\Delta$ will be taken from the range $\Delta\in(0,\Delta^*]$. Eq. (39a) comes from the definition of the region $\phi_{uc}$, and $\rho_0 := \min_{x\in\mathcal{U}_\rho\backslash(\mathcal{U}_{\rho_s}\cup\mathcal{B}_\delta(x_e))}|\hat{W}(x)-\hat{W}(0)|$, and $\hat{W}(0)$ is the minimum of $\hat{W}(x)$ which is found at the origin. Furthermore, since $\hat{W}(x)$ is a $\mathcal{C}^1$ function that meets the property of Eq. (31b), and considering the fact that $f(\cdot)$ and $g(\cdot)$ are sufficiently smooth functions, we have the following inequalities:

$$|L_f\hat{W}(x(t))-L_f\hat{W}(x(t_k))|\le k_2|x(t)-x(t_k)| \tag{40a}$$

$$|(L_g\hat{W}(x(t))-L_g\hat{W}(x(t_k)))u(t)|\le k_3|x(t)-x(t_k)| \tag{40b}$$

where $k_2$ and $k_3$ are positive real numbers. With these inequalities established, the following proposition is presented to show that with sufficient conditions, the controller $u=\Phi(x)\in U$ designed based on the FNN-based CLBF $\hat{W}(x)$ and the discontinuous control law $u=\bar{u}(x)\in U$ are able to guarantee closed-loop stability and safety for the nonlinear system in Eq. (1).

**Proposition 2.** *Consider the nonlinear system of Eq. (1) with a FNN-based CLBF $\hat{W}(x)$ designed based on a valid CLF $V(x)$ and a valid FNN-CBF $\hat{B}(x)$ that satisfies* Eq. (3) *with probability of at least $1-\delta$. There exists $\epsilon>0$, $\Delta'>0$, $\Delta''>0$, $\rho>\rho_{\min}>\rho_s$ that satisfy:*

$$\Delta' < \frac{\alpha_W\rho_0-\epsilon}{k_1(k_2+k_3)},\ \ 0\le\epsilon<\alpha_W\rho_0 \tag{41a}$$

$$\rho_{min} := \max_{\Delta t\in[0,\Delta'')}\{\hat{W}(x(t_k+\Delta t)) \mid x(t_k)\in\mathcal{U}_{\rho_s},\ \ u\in U\} \tag{41b}$$

$$\Delta^* = \min\{\Delta',\Delta''\} \tag{41c}$$

*such that, for any $x(t_k)\in\mathcal{U}_\rho$, under the sample-and-hold application of either $u(t)=\Phi(x(t_k))\ \forall t\in[t_k,t_{k+1})$ where $t_{k+1}=t_k+\Delta$ and $\Delta\in$*

$(0, \Delta^*]$, *or* $u(t) = \bar{u}(x(t_k)) \in U$ *when* $x(t_k) \in \mathcal{B}_\delta(x_e)$, $\hat{W}(x)$ *is guaranteed to decrease over one sampling period with probability of at least* $1-\delta$, *and* $x(t)$ *is bounded in* $\mathcal{U}_\rho$ *for all times and ultimately converges to* $\mathcal{U}_{\rho_{\min}}$.

**Proof.** We first consider the case of bounded unsafe sets in state space. We will first prove that the closed-loop state trajectory $x(t)$ will be bounded in $\mathcal{U}_\rho$ and will enter $\mathcal{U}_{\rho_s}$ in finite steps under the sample-and-hold implementation of control actions $u = \Phi(x) \in U$ or $u = \bar{u}(x) \in U$ if $x \in \mathcal{B}_\delta(x_e)$. Then we will prove that once the state enters $\mathcal{U}_{\rho_s}$, i.e., $x(t_k) \in \mathcal{U}_{\rho_s}$, $x(t)$ will stay in $\mathcal{U}_{\rho_{\min}}$ for $t = [t_k, t_k + \Delta'')$.

Under the sample-and-hold implementation of $u(t)$, for $x(t_k) \in \mathcal{U}_\rho \backslash (\mathcal{U}_{\rho_s} \cup \mathcal{B}_\delta(x_e))$, we can write $\dot{\hat{W}}(x)$ as follows:

$$\begin{aligned}\dot{\hat{W}}(x(t), u(t)) &= \dot{\hat{W}}(x(t_k), u(t_k)) + (\dot{\hat{W}}(x(t), u(t)) \\ &\quad -\dot{\hat{W}}(x(t_k), u(t_k))) \\ &= L_f\hat{W}(x(t_k)) + L_g\hat{W}(x(t_k))u(t_k) \\ &\quad +\left(L_f\hat{W}(x(t)) - L_f\hat{W}(x(t_k))\right) \\ &\quad +\left(L_g\hat{W}(x(t)) - L_g\hat{W}(x(t_k))\right)u(t) \end{aligned} \tag{42}$$

Substituting Eqs. (39a), (40) and (39b), we derive the following inequality:

$$\dot{\hat{W}}(x(t), u(t)) < -\alpha_W \rho_0 + k_1(k_2 + k_3)\Delta' < -\epsilon \tag{43}$$

which sufficiently shows that under sample-and-hold implementation of control actions $u(t)$, $\dot{\hat{W}}(x)$ can be rendered negative for any $x(t_k) \in \mathcal{U}_\rho \backslash (\mathcal{U}_{\rho_s} \cup \mathcal{B}_\delta(x_e))$, and $\hat{W}(x(t)) < \hat{W}(x(t_k)) \le \rho$, therefore bounded within $\mathcal{U}_\rho$ $\forall t > t_k$. Within finite steps, $x(t)$ will eventually enter $\mathcal{U}_{\rho_s}$.

For bounded unsafe sets where stationary points in state-space exist, consider $x(t_k) \in \mathcal{B}_\delta(x_e)$. $x(t_{k+1})$ can be driven to a smaller level set of $\hat{W}(x)$ under the discontinuous control law $u = \bar{u}(x) \in U$ which decreases $\hat{W}(x)$ over one sampling period; i.e., $\hat{W}(x(t_{k+1})) < \hat{W}(x(t_k))$. Within finite sampling periods, the closed-loop state will eventually leave $\mathcal{B}_\delta(x_e)$, and will never return since the control law $u = \Phi(x) \in U$ will take over and ensure that $\hat{W}(x(t)) < \hat{W}(x(t_k))$ for all $t > t_k$.

Once the state enters the set $\mathcal{U}_{\rho_s}$, $x(t_k) \in \mathcal{U}_{\rho_s}$, the definition of $\mathcal{U}_{\rho_{\min}}$ in Eq. (41b) shows that the trajectory $x(t)$ will stay in $\mathcal{U}_{\rho_{\min}}$ for $t \in [t_k, t_k + \Delta'')$. We choose a maximal sampling period $\Delta^*$ which is the minimum of $\Delta'$ and $\Delta''$ as described by Eq. (41c), and choose a sampling period $\Delta \in (0, \Delta^*]$. Within $t \in [t_k, t_k + \Delta)$, under the sample-and-hold implementation of $u = \Phi(x) \in U$ or $u = \bar{u} \in U$, we are able to show that, with probability at least $1-\delta$, for $x(t_k) \in \mathcal{U}_\rho \backslash \mathcal{U}_{\rho_s}$, $x(t)$ moves towards the origin into smaller level sets of $\hat{W}$ and eventually into the level set $\mathcal{U}_{\rho_s}$, and for $x(t_k) \in \mathcal{U}_{\rho_s}$, $x(t)$ remains in $\mathcal{U}_{\rho_{\min}}$. Since the CLBF properties on $\hat{W}(x)$ are satisfied with a probability of at least $1-\delta$, the closed-loop stability and safety of the system under the sample-and-hold implementation of CLBF-based control laws also follow the same probability.

In the case of unbounded unsafe sets, stationary points other than the origin $\mathcal{B}_\delta(x_e)$ do not exist, therefore, the sample-and-hold control actions $u = \Phi(x) \in U$ are able to drive closed-loop state towards smaller level sets of $\hat{W}(x)$ since $\dot{\hat{W}}(x, \Phi(x)) < 0$ holds, and similarly, will be bounded within $\mathcal{U}_{\rho_{\min}}$ eventually. □

### 5.3. FNN-CLBF-based MPC

Given the probabilistic stability and safety analysis provided by the sample-and-hold implementation of FNN-CLBF-based control laws $u = \Phi(x) \in U$, the FNN-CLBF-based MPC is formulated as follows:

$$\mathcal{J} = \min_{u \in S(\Delta)} \int_{t_k}^{t_{k+N}} l(\tilde{x}(t), u(t))dt \tag{44a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = f(\tilde{x}(t)) + g(u(t)) \tag{44b}$$

$$\tilde{x}(t_k) = x(t_k) \tag{44c}$$

$$u(t) \in U, \ \forall\ t \in [t_k, t_{k+N}) \tag{44d}$$

$$\dot{\hat{W}}(x(t_k), u(t_k)) \le \dot{\hat{W}}(x(t_k), \Phi(x(t_k))) \quad \text{if } x(t_k) \notin \mathcal{B}_\delta(x_e) \text{ and } \hat{W}(x(t_k)) > \rho_{min} \tag{44e}$$

$$\hat{W}(\tilde{x}(t)) \le \rho_{min}, \ \forall\ t \in [t_k, t_{k+N}), \ \text{if } \hat{W}(x(t_k)) \le \rho_{min} \tag{44f}$$

$$\hat{W}(\tilde{x}(t)) < \hat{W}(x(t_k)), \ \forall\ t \in (t_k, t_{k+N}), \quad \text{if } x(t_k) \in \mathcal{B}_\delta(x_e) \tag{44g}$$

where the state trajectory predicted by the ODE model of Eq. (1) is represented by $\tilde{x}(t)$, the number of sampling periods in the prediction horizon is denoted by $N$, and $S(\Delta)$ is a piecewise constant function with a sampling time $\Delta$. This optimization problem of Eq. (44) is solved by the MPC every time a new measurement is received (every $\Delta$), and the optimization problem has an objective function Eq. (44a) that is in the form of the integral of $l(\tilde{x}(t), u(t)) = \tilde{x}^T Q \tilde{x} + u^T R u$ over the prediction horizon. Here, $Q$, $R$ are positive definite weight matrices. The objective function is formulated this way such that it has a minimum at the origin. Eq. (44d) describes the constraints imposed on the input vector along the predicted trajectory. It is assumed that state measurements are received at every sampling period. As seen in Eq. (44c), the initial condition of the predicted state trajectory in Eq. (44b) are obtained from the feedback state measurements at $t = t_k$. The constraints of Eqs. (44e)–(44g) are used to ensure closed-loop stability and safety. When $x(t_k) \notin \mathcal{B}_\delta(x_e)$ and $\hat{W}(x(t_k)) > \rho_{\min}$, the constraint in Eq. (44e) decreases $\hat{W}(\tilde{x})$ at a rate at least of the rate achieved by the CLBF-based controller $u = \Phi(x) \in U$. When $\hat{W}(x(t_k)) \le \rho_{\min}$, Eq. (44f) maintains the closed-loop state trajectory over the prediction horizon inside the level set $\mathcal{U}_{\rho_{\min}}$. If $x(t_k) \in \mathcal{B}_\delta(x_e)$, Eq. (44g) is activated to decrease $\hat{W}(x)$ over the next sampling period so that the state will escape the saddle point within finite steps. The first control action $u^*(t_k)$ of the optimized input trajectory $u^*(t)$ will be applied in a sample-and-hold manner for the next sampling period. After that, the horizon will move forward one sampling period, and the above optimization problem is solved again.

The CLBF used in the CLBF-MPC of Eq. (44) is one constructed using an FNN-based CBF $\hat{B}(x)$, which is well-trained and designed to satisfy modeling error constraints in Eqs. (27) and (30). Subsequently, with probability at least $1-\delta$, $\hat{B}(x)$ meets the conditions of Eq. 3, CLBF meets the conditions of Eq. (31) via the design method presented in Proposition 1, and therefore, probabilistic safety and stability under the CLBF-based control laws are provided. The following theorem will demonstrate that probabilistic stability and safety can be established under the CLBF-MPC of Eq. (44).

**Theorem 3.** *Consider the nonlinear system of* Eq. (1) *with a constrained CLBF* $\hat{W}(x)$ *built following Proposition 1 using a FNN-CBF* $\hat{B}(x)$ *that satisfies* Eqs. (27) *and* (30) *and meets the conditions of*

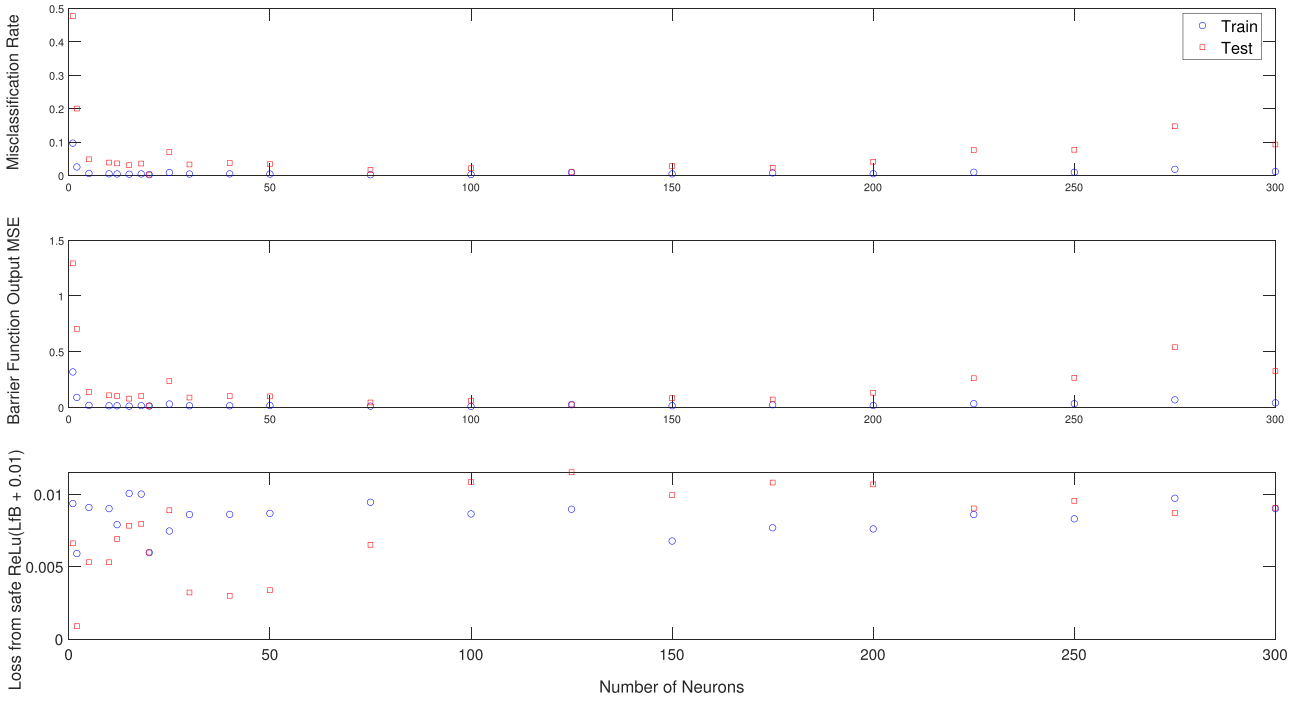


**Fig. 1.** Generalization performance for the FNN models for characterizing bounded unsafe regions utilizing various neurons.

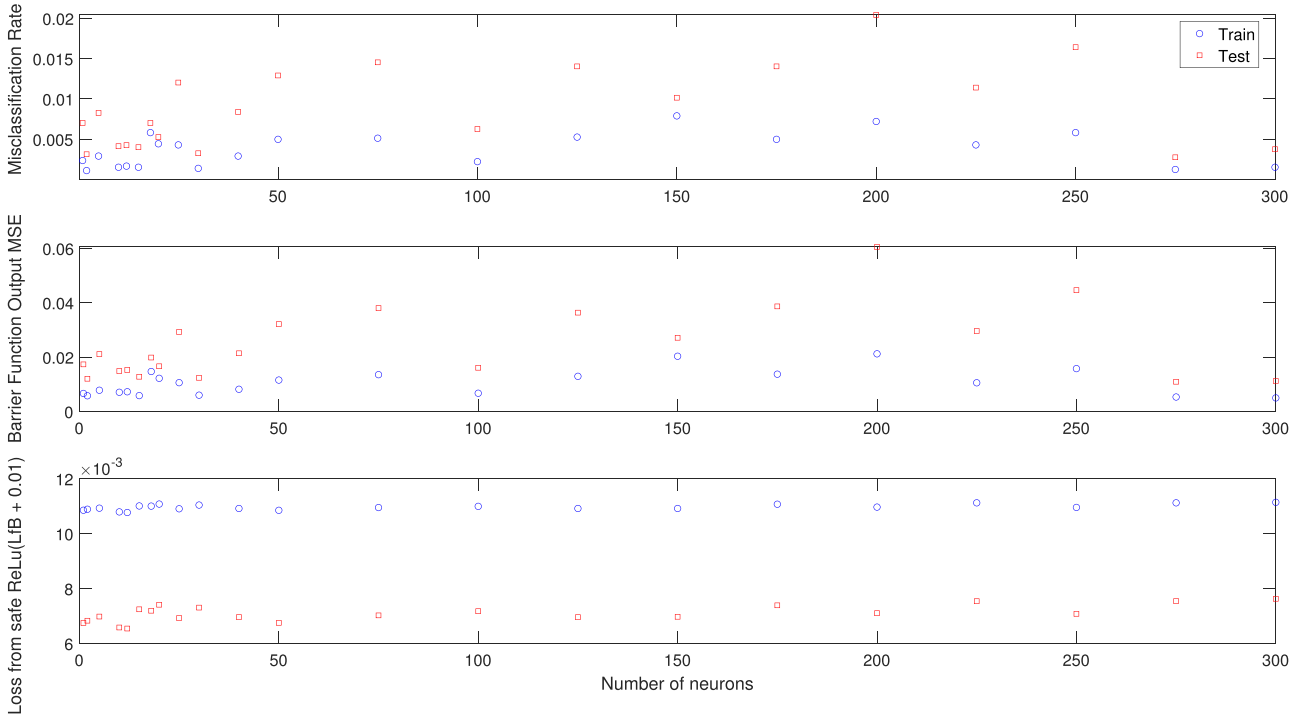


**Fig. 2.** Generalization performance for the FNN models for characterizing unbounded unsafe region utilizing various neurons.

*Eq. (3) with probability of at least* $1-\delta$. *Let* $\Delta > 0$, $\epsilon > 0$, $\rho > \rho_{\min} > \rho_s$ *satisfy the requirements in Proposition 2. Given* $x_0 \in \mathcal{U}_\rho$, *with probability of at least* $1-\delta$, *recursive feasibility can be guaranteed for the optimization problem of Eq. (44), and the closed-loop state is bounded in* $\mathcal{U}_\rho$, $\forall\ t \geq 0$, *and converges to* $\mathcal{U}_{\rho_{\min}}$ *as* $t \to \infty$.

**Proof.** There always exists a feasible solution for the CLBF-MPC optimization problem since sample-and-hold implementation of the CLBF-based control law $u = \Phi(x) \in U$ (when $x(t_k) \in \mathcal{U}_\rho \backslash \mathcal{B}_\delta(x_e)$) and the discontinuous control law $u = \bar{u} \in U$ (when $x(t_k) \in \mathcal{B}_\delta(x_e)$ in the case of bounded unsafe sets) provide one such solution that satisfy the constraints of Eqs. (44d)–(44g) for all $x(t_k) \in \mathcal{U}_\rho$. This has been proven in Proposition 2. The properties Eq. (33) ensure that the CBF $\hat{B}$ is able to discern the unsafe region from the safe region accurately with a probability of at least $1-\delta$. Furthermore, it has been shown in Proposition 1 that $\dot{\hat{W}}(x) \leq 0$ is held with probability at least $1-\delta$ in the region $\mathcal{U}_\rho$.

For unbounded unsafe sets, there are no stationary points in the operating region other than the origin. For any $x_0 \in \mathcal{U}_\rho \backslash (\mathcal{B}_\delta(x_e) \cup$

$\mathcal{U}_{\rho_{\min}}$), Eq. (44e) forces the optimal control action calculated by the FNN-CLBF-based MPC $u^*(t_k)$ to decrease $\hat{W}(x)$ at a rate at least as fast as that achieved by the control law $\Phi(x(t_k))$. Therefore, $u^*(t_k)$ will drive the closed-loop state towards the origin and into $\mathcal{U}_{\rho_{\min}}$ within finite steps. After that, Eq. (44f) ensures that the closed-loop state remains inside $\mathcal{U}_{\rho_{\min}}$. We can conclude that the closed-loop state under the CLBF-MPC will be bounded in $\mathcal{U}_\rho$ for $t > 0$ and eventually be bounded in $\mathcal{U}_{\rho_{\min}}$, thus will not enter the unsafe set $\mathcal{D}$ for all times since the safe set $\mathcal{U}_\rho$ has no intersection with the unsafe set $\mathcal{D}$.

In the case of bounded unsafe sets, when the closed-loop state reaches a stationary point, $x(t_k) \in \mathcal{B}_\delta(x_e)$, Eq. (44g) is activated to ensure that the optimal solution of the MPC drives the state away from the stationary point in a direction of decreasing $\hat{W}$. After the state escapes the neighborhood around the saddle point, Eqs. (44e)–(44f) will continue to ensure that $x(t)$ is bounded in $\mathcal{U}_\rho$ and eventually converges to $\mathcal{U}_{\rho_{\min}}$ without entering the bounded unsafe set. □

When Eq. (44e) is activated, the FNN-CBF is used to predict the corresponding barrier function value $\hat{B}$ based on $x(t_k)$. This $\hat{B}(x(t_k))$ prediction is shown to satisfy the CBF properties of Eq. (3) with probability of at least $1-\delta$, and therefore stability and safety properties enforced by Eq. 44e are achieved with a probability of at least $1-\delta$. When Eqs. (44f) or (44g) are activated, FNN predictions of the barrier function are carried out for the entire trajectory $\hat{B}(\tilde{x}(t))$ for $t \in [t_k, t_{k+N}]$. Each of the FNN inputs, $\tilde{x}(t)$, are calculated based on the ODE model of Eq. (1), which are accurate assuming there are no modeling mismatches. The predictions $\hat{B}(\tilde{x}(t))$ based on $\tilde{x}(t)$ are therefore independent predictions and do not affect one another. At each time step of the trajectory in the MPC prediction horizon, the probability of the actual closed-loop state being maintained inside $\mathcal{U}_{\rho_{\min}}$ (in the case of Eq. (44f)), or the actual closed-loop state being driven around the unsafe set in the direction of decreasing CLBF (in the case of Eq. (44f)) is at least $1-\delta$. However, to ensure that the entire trajectory satisfies its safety and stability properties, the probability will be reduced (specifically, $(1-\delta)^N$ for $N$ time steps in the prediction horizon). Although the overall probability of stability and safety for this predicted trajectory is reduced, the stability and safety properties of the system under the first control action $u^*(t_k)$ of the FNN-CLBF-MPC for the current time step $t = t_k$ is guaranteed with probability $1-\delta$. When a new feedback measurement is received, the MPC is executed again and computes a new control action to be applied that ensures stability and safety with probability $1-\delta$ over the next sampling step.

**Remark 2.** In this study, we study the generalization error bound of the FNN-CBF and the probabilistic closed-loop stability and safety properties of the FNN-CLBF-MPC where the MPC uses the first-principles model for prediction. In our previous work in Chen et al. (2021), we have also developed FNN-CLBF-MPC systems where the MPC can use a prediction model of the nonlinear process built using recurrent neural networks (RNN). Similar to the FNN used in this study, with a neural-network-based model, there exists an expected error in the predicted output $\hat{x}$ of the nonlinear system that can be upper-bounded following machine learning theory; this has been developed in Wu et al. (2021). In our previous work (Chen et al., 2021), we have discussed design methods with data generation and unsafe region characterization to account for both modeling error in the FNN-CBF and in the RNN process model, as well as with numerical approximations of the predicted vector and matrix functions $\hat{f}$ and $\hat{g}$. We have demonstrated through theoretical development as well as closed-loop simulations that with adequate design and verification of the FNN-CBF as well as sufficient boundedness of the modeling and numerical errors, closed-loop stability and safety can be achieved for FNN-CLBF-MPC using both first-principles and RNN models. In this work, we have only conducted closed-loop studies on FNN-CLBF-MPC using a first-principles model as the focus of this manuscript is on the generalization error upper bound of the FNN model. We can easily extend the statistical stability and safety analysis to FNN-CLBF-MPC using RNN models by following the work in Wu et al. (2021), where we can further specify the upper bound on the modeling error of the RNN process model as it depends on a number of factors such as sample size, weight matrix bounds, input length, and network complexity, and in turn construct the RNN to meet Lyapunov-based stability properties in probability.

## 6. Application to a chemical process example

### 6.1. Preliminaries

A chemical process example is simulated to demonstrate the effectiveness of the FNN-based CLBF in ensuring the closed-loop stability and safety of a nonlinear process, and to demonstrate how various aspects of FNN design and training may impact the outcome of the FNN model. The system we consider is a continuously stirred tank reactor (CSTR) which is non-isothermal and assumed to be well-mixed, undergoing a second-order, exothermic, irreversible reaction converting reactant $A$ into product $B$. There is a heating jacket equipped to remove and supply heat. The process dynamics can be modelled by material and energy balances as shown below:

$$\frac{dC_A}{dt} = \frac{F}{V_L}(C_{A0} - C_A) - k_0 e^{-E/RT} C_A^2 \tag{45a}$$

$$\frac{dT}{dt} = \frac{F}{V_L}(T_0 - T) - \frac{\Delta H k_0}{\rho_L C_p} e^{-E/RT} C_A^2 + \frac{Q}{\rho_L C_p V_L} \tag{45b}$$

where the two states of the system, $C_A$ and $T$, are the concentration of $A$ in the tank and the temperature inside the tank, respectively. $V_L$, $F$, $T_0$ represent the volume of the reacting fluid in the reactor, volumetric flow rate of the feed, and temperature of the feed, respectively. $Q$ denotes the heat jacket input rate, and $C_{A0}$ denotes the feed concentration of reactant $A$. It is assumed that the reacting liquid has a constant heat capacity $C_p$ and density $\rho_L$. Other constants such as the pre-exponential constant, ideal gas constant, enthalpy and activation energy of the reaction are denoted by $k_0$, $R$, $\Delta H$, and $E$, respectively. The values of these process parameters are given in Wu et al. (2019b).

The CSTR process is stabilized at its unstable equilibrium point $(C_{As},\ T_s) = (1.954\ kmol/m^3,\ 401.9\ K)$ by the CLBF-based MPC, which manipulates the inputs $C_{A0}$ and $Q$ with corresponding steady-state values $(C_{A0_s}\ Q_s) = (4\ kmol/m^3,\ 0\ kJ/hr)$. The manipulated inputs have the following bounds: $|\Delta C_{A0}| \le 3.5\ kmol/m^3$ and $|\Delta Q| \le 5 \times 10^5\ kJ/hr$, which originate from physical constraints. The states and the inputs of the system are represented in deviation variable for the subsequent analyses such that the equilibrium point of Eq. (45) is at the origin, i.e., $[\Delta C_A = C_A - C_{As},\ \Delta T = T - T_s]$, and $[\Delta C_{A0} = C_{A0} - C_{A0_s},\ \Delta Q = Q - Q_s]$. For simplicity of notation, the state and input vectors are represented in the following forms: $x^T = [\Delta C_A\ \Delta T]$ and $u^T = [\Delta C_{A0}\ \Delta Q]$. The CLBF-MPC is executed every sampling period where $\Delta = 10^{-3}\ hr$, where the nonlinear optimization problem of Eq. (44) is solved using the python module PyIpopt. To simulate the CSTR process and predict the state trajectory inside the MPC, the system of ODE of Eq. (45) is solved using the explicit Euler method with an integration time step of $h_c = 10^{-5}\ hr$.

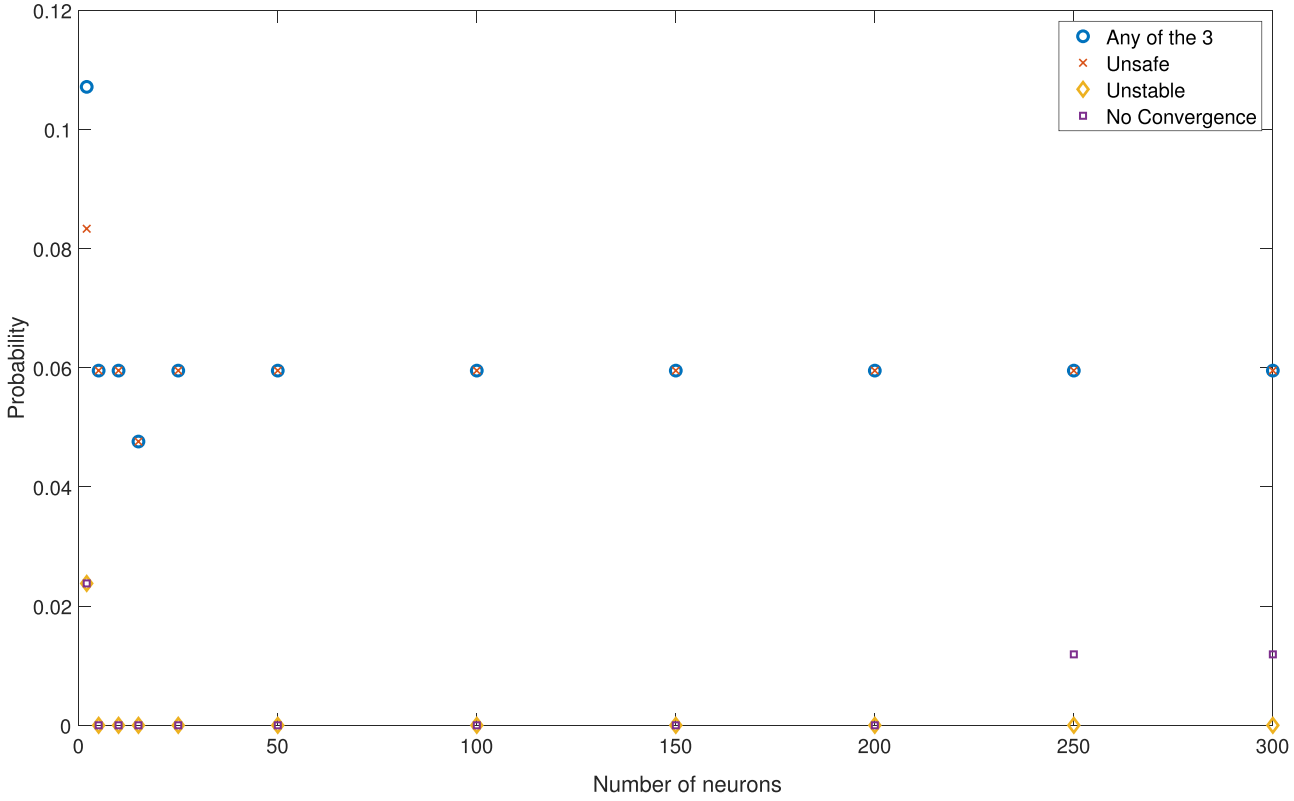


**Fig. 3.** Probabilities of unsafe, unstable, and non-convergent behavior under closed-loop control of the FNN-based CLBF-MPC for FNN models trained with varying neurons in the case of bounded unsafe region.

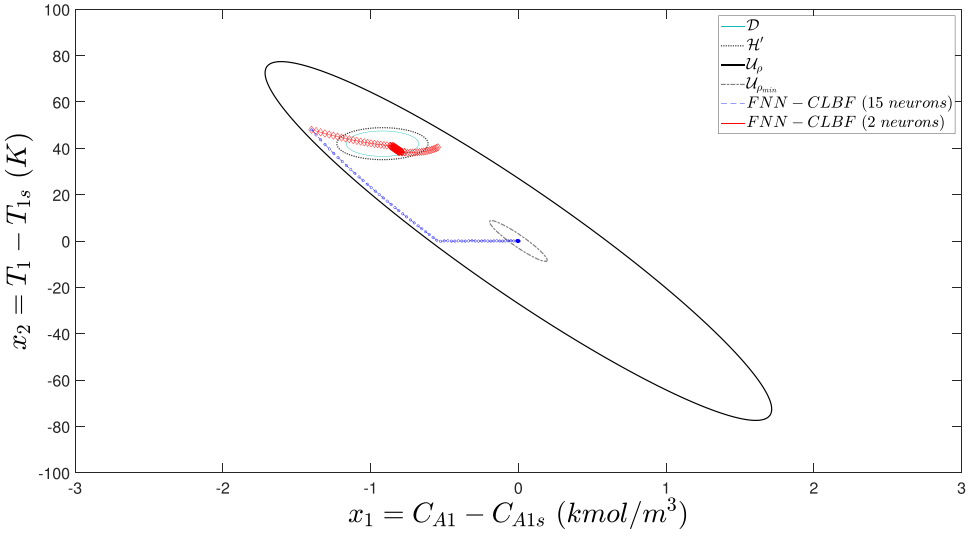


**Fig. 4.** Closed-loop state trajectories under CLBF-MPC with FNN-based barrier function trained with 15 neurons (blue) vs. 2 neurons (red), where states classified as safe by each FNN model are labelled by circle markers, and states classified as unsafe by each FNN model are labelled by diamond markers.

We use the following positive definite $P$ matrix to build a CLF $V(x)$ in the form of $V(x) = x^T Px$:

$$P = \begin{bmatrix} 1060 & 22 \\ 22 & 0.52 \end{bmatrix} \tag{46}$$

where the values of the $P$ matrix are determined via extensive closed-loop simulations of the process. The unsafe region $\mathcal{D}$ can be either bounded or unbounded, and is a set within the stability region such that the state may enter the unsafe region on its path while it is driven towards the origin under a control law that does not consider safety constraints. The CLBF-MPC accounts for these unsafe regions in state-space and is capable of navigating the states around the unsafe set and towards the equilibrium point thereafter.

### 6.2. Development of the FNN model for barrier function

The control barrier function within the CLBF is built using an FNN model, which takes $x$ as inputs and computes the value $\hat{B}(x)$. In this study, we consider the cases of both bounded and unbounded unsafe regions. First, the bounded unsafe set is considered, where the unsafe region is defined as $\mathcal{D}_b := \{x \in$

$\mathbf{R}^2 \mid F_b(x) = \frac{(x_1+0.92)^2}{1} + \frac{(x_2-42)^2}{500} < 0.06\}$. $\mathcal{H}_b$ is defined as $\mathcal{H}_b := \{x \in \mathbf{R}^2 \mid F_b(x) < 0.07\}$ such that it satisfies $\mathcal{D}_b \subset \mathcal{H}_b \subset \phi_{uc}$ in Proposition 1. The unsafe region is an ellipse embedded in the operating region to demonstrate the challenging case of a bounded unsafe set obstructing the trajectory of the closed-loop state. Practically, the unsafe sets may not be easily represented in a closed form function. However, based on engineering knowledge or simulations, one may collect sufficiently dense data in the operating region with corresponding labels indicating whether the data point is safe or unsafe. Following this, we can obtain respective sets of data samples that are labelled as unsafe and safe, and can be subsequently used for model training. In our study, after specifying the region of unsafe operation, we generate training data for the FNN model. This is done by first specifying a region which the system is likely operated within, in this case, we specify $V(x) \leq 368$, which is a level set of CLF characterized as the stability region in the absence of unsafe sets under the use of Lyapunov-based control laws. Then, we specify $\mathcal{H}'_b := \{x \in \mathbf{R}^2 \mid F_b(x) < 0.0952\}$, which is a larger compact set that encloses $\mathcal{H}$ with enough contingency accounting for modeling and numerical error. Similarly, we also consider the case of unbounded unsafe sets, which have the unsafe region defined as follows: $\mathcal{D}_u := \{x \in \mathbf{R}^2 \mid F_u(x) = x_1 + x_2 > 47\}$. Since both the unsafe and the safe sets from which we sample must be compact, we first approximate this unbounded region with a sufficiently large compact set within the operating region $\mathcal{D}_{u'} := \{x \in \mathbf{R}^2 \mid F_u(x) \geq 46 \text{ and } V(x) \leq 368\}$. We then characterize $\mathcal{H}'_u \supset \mathcal{D}_{u'}$ as $\mathcal{H}'_u := \{x \in \mathbf{R}^2 \mid F_u(x) > 45 \text{ and } V(x) \leq 368\}$.

Data points that fall in the set $\mathcal{H}'$ are labeled as "unsafe", while data points outside of this set are labeled as "safe". Both the safe and the unsafe regions are discretized into the same number of data samples, where the samples are labeled with a target output of $B(x) = +1$ if $x$ belongs to the unsafe set, and $B(x) = -1$ if $x$ belongs to the safe set. The inputs to the FNN model are the vector of state measurements $x$, and the FNN model produces $\hat{B}(x)$ values that classify $x$ as being safe or unsafe.

The following three case studies are examined: varying the number of neurons in the FNN, varying the number of layers in the FNN, and varying the number of training sample size in the FNN. We construct numerous FNN models under each scenario to study the impact of the structure and training of FNN on the generalization error of the resulting model. In all models we construct, the activation functions used in all hidden layers are $tanh(\cdot)$, and the cost functions of Eq. (6) are used, where both loss functions are monitored separately during training. The FNN undergoes 500 epochs of training. $L_2 = 0$ and $L_1$ no longer decreasing for 100 consecutive epochs are the two criteria to trigger early-stopping of training.

Once a FNN-CBF is built, it must be verified that the conditions of Eq. (3) must hold for all $x$ in their respective compact sets, by examining whether the strict inequalities of Eq. (3) hold for a tightened bound as described in Theorem 1. For example, it has been shown that for a 3-hidden-layer FNN with 10 neurons in each layer, $\hat{B}(x) \geq 0.5751$ for all discretized $x \in \hat{\mathcal{H}}$, where $\hat{\mathcal{H}}$ is the unsafe region characterized by the predictions of the FNN model, and $\hat{B}(x) \leq -0.0033$ for all discretized $x \in \mathcal{U}_\rho \backslash \hat{\mathcal{H}}$. It is shown that $\hat{\mathcal{H}}$ is a superset of $\mathcal{D}$, since there are safe points outside the boundary of $\mathcal{D}$ being misclassified as unsafe by the FNN, but there are no unsafe points being misclassified as safe. Therefore, the conditions of Eqs. (3a) and (3c) are proven to hold in a continuous sense. To prove that the condition of Eq. (3b) also holds, we examine $L_f\hat{B}(x)$ values for all discretized $x$ in the safe set for which $L_g\hat{B}(x) = 0$. This can be seen from the error metrics in Fig. 6, where for a 3-hidden-layer model, the errors from $ReLu(L_f\hat{B}(x) + 0.01)$ for all safe $x$'s at which $L_g\hat{B}(x) = 0$ in both training and testing datasets are below $1.18 \times 10^{-6}$, which means that $L_f\hat{B} \leq -0.01 - 1.18 \times 10^{-6}$. Thus, the condition of Eq. (3b) is proven to be true in a continuous sense.

Once the control barrier function is verified, the CLBF $\hat{W}(x)$ is characterized with the following parameters: $c_1 = 0.001$, $c_2 = 100$, $c_3 = 49.38$, $c_4 = 35.21$, $\mu = 5000$, $\rho = 0$, and $\nu = -0.0352$ following the guidelines in Proposition 1. The stability and safety region $\mathcal{U}_\rho$ is therefore defined according to Eq. (31d).

### 6.3. Analysis on generalization performance and closed-loop stability and safety

The generalization performance is assessed via three metrics: the misclassification rate calculated as the ratio of misclassified samples over the total number of samples in the training and testing data sets, the MSE between the predicted and true barrier function output, and loss function calculated from $ReLu(L_f\hat{B}(x) + \tau_l$ for all safe $x$'s in each data set, where $\tau_l = 0.01$ is a positive constant used to ensure the negative definiteness of $L_f\hat{B}(x)$.

Within each case study of FNN models trained using different width, depth, and sample size, both bounded and unbounded unsafe sets are studied. In addition to studying the generalization performance of these FNN models, closed-loop simulations are also ran and compared, and the probability of stability and safety has been investigated. We also run closed-loop simulations with these FNN models and assess its probability of unsafe, unstable, or non-convergent behavior. Unsafe behavior is defined as the closed-loop state entering the unsafe region $\mathcal{D}$ any time during its trajectory from the initial condition to the final state. Unstable behavior is when the closed-loop state exits the stability operating region any time during the simulation period. Non-convergence occurs when the final state at the end of the simulation period is not within the terminal set $\mathcal{U}_{\rho_{\min}}$, or when the state exits the terminal set after entering it for the first time. We discretize the operating region evenly to generate a set of $x_0$ used as initial conditions for closed-loop studied. We run closed-loop simulations starting from 83 different initial conditions in the operating region $\mathcal{U}_\rho \backslash (\mathcal{U}_{\rho_{\min}} \cup \mathcal{B}_\delta(x_e))$ for the case of bounded unsafe sets, and from 74 different initial conditions in the operating region $\mathcal{U}_\rho \backslash \mathcal{U}_{\rho_{\min}}$ for the case of unbounded unsafe sets. The probability of each of these three undesirable behaviors is calculated by tabulating the number of occurrences out of the total number of initial conditions ran.

#### 6.3.1. Varying number of neurons

In this case study, the FNN is trained with different number of neurons within 1 hidden layer, where the number of neurons (or the width of the FNN) varies from $n_w = 1$ to $n_w = 300$. In the case of the bounded unsafe set, by discretizing the boxes around both the safe and the unsafe regions along each dimension of the state vector by a mesh grid size of 350 by 350, we obtain 20,472 unsafe samples and 25,695 safe samples. In the case of the unbounded unsafe set, since it is a simpler case where the boundary of safety is linear, we discretize the safe and unsafe regions by 150 by 150, resulting in a dataset of 3021 unsafe and 4198 safe samples respectively. 70% of these samples are used for training, and 30% are used for testing. The generalization performance for FNN models with various number of neurons to address the presence of bounded and unbounded unsafe sets are shown in Figs. 1 and 2, respectively.

In the case of bounded unsafe regions, the drop in misclassification rate and barrier function output MSE are prominently shown as the number of neurons increases from $n_w = 1$ to $n_w = 10$ for both training and testing datasets. The misclassification rate and output MSE for the training dataset stay consistently low for $n_w \geq 10$, where its misclassification rate is maintained below 0.018 and MSE output is maintained below 0.067 (this high point occurs

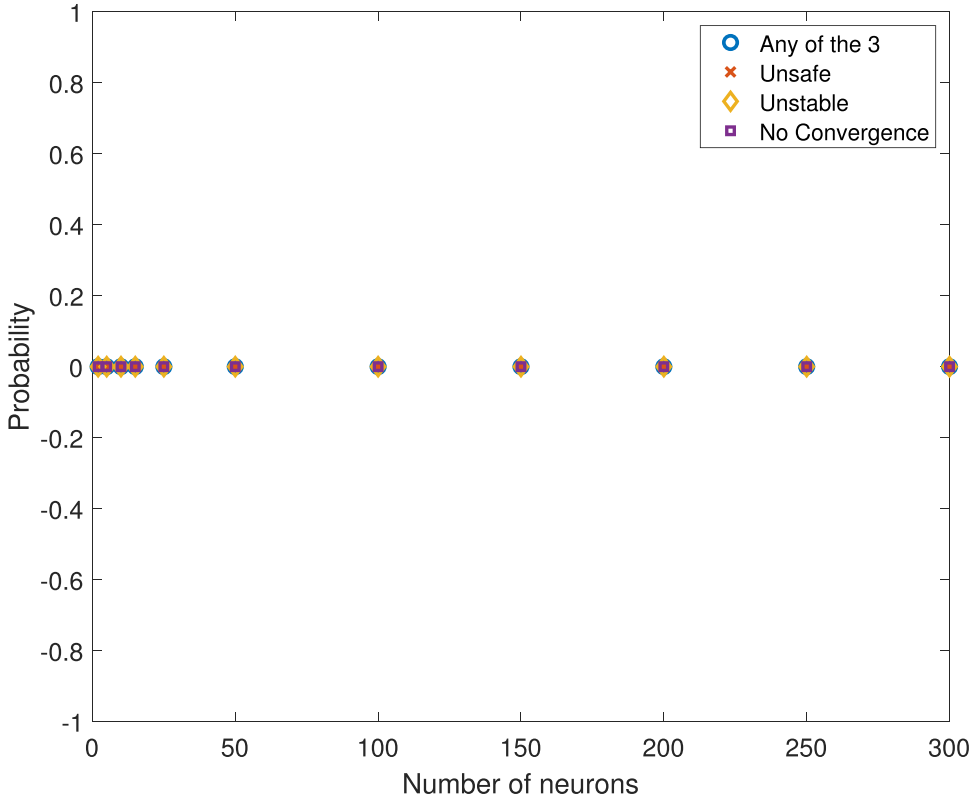


**Fig. 5.** Probabilities of unsafe, unstable, and non-convergent behavior under closed-loop control of the FNN-based CLBF-MPC for FNN models trained with varying neurons in the case of unbounded unsafe regions.

at $n_w = 300$). Misclassification rate and output MSE in the testing dataset are consistently higher than the training dataset for all variations of $n_w$, which is expected as there exists a gap between the expected error and the empirical error as shown in the generalization error analysis in this work. For the testing dataset, misclassification rate stays below 0.04 and output MSE stays below 0.11 for $n_w \in [10, 200]$ except for the one-off case at $n_w = 25$, which has a testing data misclassification rate of $7 \times 10^{-2}$ and an output MSE of $2.3 \times 10^{-1}$. Sometimes one-off cases of FNN models occur where their resulting errors are higher than other FNN models of similar structure due to the stochastic nature of FNN training and prediction. For $n_w > 200$, it is seen that the testing errors in misclassification rate and output MSE increase as $n_w$ increases while the training errors for these two metrics stay consistently low. This is expected as the FNN model is essentially overparametrized by too many number of neurons, and while this improves the model's ability to learn and fit existing data, it becomes overfitted and therefore producing increasingly larger errors when applied to other data samples that do not exist in the training set. The third error metric is the loss calculated from $ReLu(L_f\hat{B}(x) + 0.01)$ for all safe $x$ that satisfy $L_g\hat{B}(x) = 0$ in both training and testing datasets. Although there are no obvious trends in the relation between this error and the number of neurons, it is observed that the error in the training set stays below $1.008 \times 10^{-2}$, while the highest error in the testing set is at $1.154 \times 10^{-2}$.

In the case of unbounded unsafe sets, all three error metrics achieve relatively low values compared to the case of bounded unsafe sets due to the less challenging nature of unbounded unsafe sets similar to a linear boundary. There are no obvious trends of errors increasing or decreasing as the number of neurons increase because the errors are already maintained at a low level (misclassification rate is kept under $2 \times 10^{-2}$, output MSE is kept under $6 \times 10^{-2}$, and $L_f\hat{B}(x)$ is kept under $1.107 \times 10^{-2}$, accounting for both training and testing errors). However, we do observe that the gap between training and testing error generally increases as $n_w$ increases beyond 50. This may be due to model overfitting where the model is again parametrized with too many neurons.

The probabilities of unsafe, unstable, and non-convergent closed-loop behavior under the control of FNN-based CLBF-MPC built using FNN models with varying number of neurons in the case of bounded unsafe set are shown in Fig. 3. In addition, the figure also shows the probability of any of these three behaviors occurring. It is demonstrated that the probability decreases drastically for $n_w > 2$ and reaches its minimum at $n_w = 15$. It is also noted that the instances of non-convergence also increases for $n_w \geq 250$, which is consistent with the trend of testing error and generalization error gap increasing for overfitted models with $n_w > 200$.

To better illustrate how FNN models trained with insufficient number of neurons may impact the closed-loop performance of the FNN-CLBF-MPC, Fig. 4 compares two state trajectories starting from the same initial condition, one under an FNN barrier function trained with 15 neurons (blue), and one under an FNN barrier function trained with 2 neurons (red). The FNN barrier function trained with 2 neurons, which has much higher errors and probabilities of instability and violation of safety, falsely identifies all states within this time-series trajectory as "unsafe" (labeled by diamond markers), including the initial condition $x_0$. Therefore, it is shown to produce a closed-loop trajectory that fails to navigate the state around the unsafe region. The closed-loop state enters the unsafe region and struggles to leave within the simulation period. This shows an instance of unsafe and non-convergent behavior amongst the 83 runs of closed-loop simulations starting from different initial conditions. On the other hand, with an FNN barrier function trained with 15 neurons, starting from the same initial condition, the closed-loop state of the CSTR process is able to con-

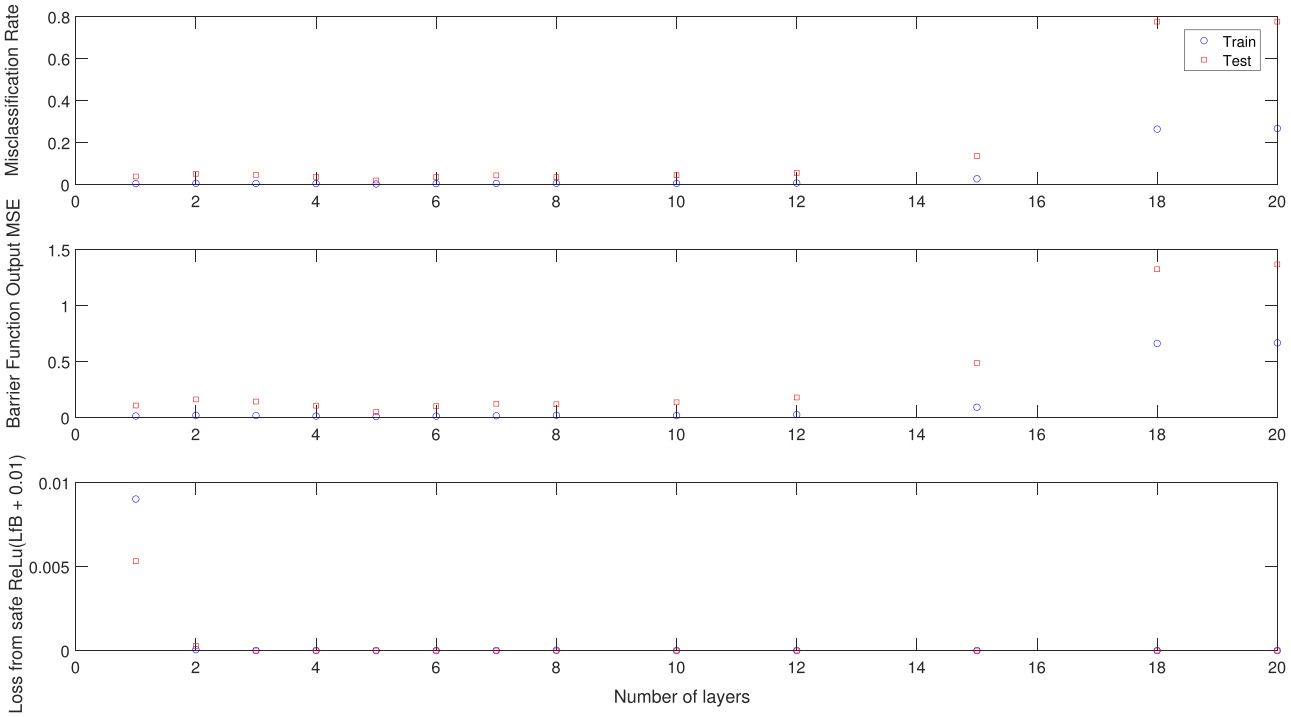


**Fig. 6.** Generalization performance for the FNN models for characterizing bounded unsafe regions utilizing various layers.

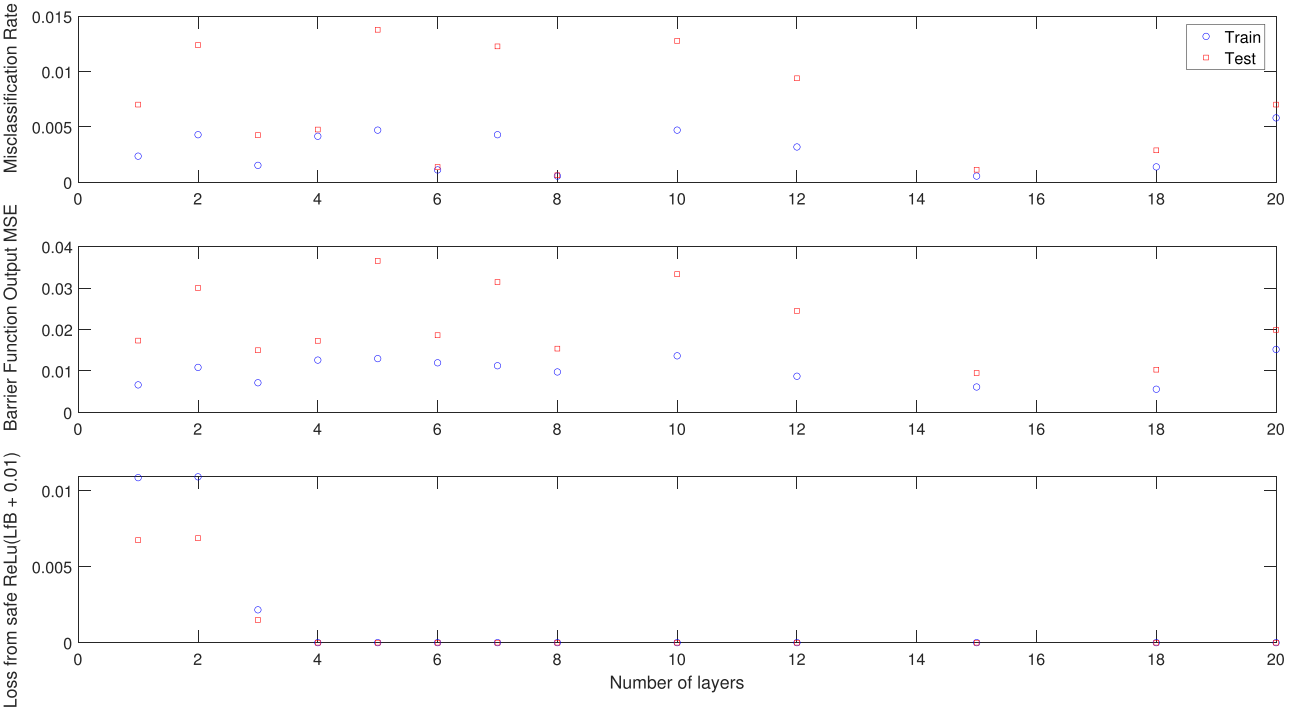


**Fig. 7.** Generalization performance for the FNN models for characterizing unbounded unsafe region utilizing various layers.

verge to the terminal set within the simulation period and avoid entering the unsafe region $\mathcal{H}'$. All states within this trajectory are correctly classified as "safe", which is labeled by the circle markers.

The probabilities of unsafe, unstable, and non-convergent closed-loop behavior in the presence of unbounded unsafe regions are shown in Fig. 5. Since all models have low misclassification rate and low barrier function output MSE, the number of occurrences of such unsafe, unstable, or non-convergent trajectories is zero for various FNN models trained with different number of neurons. In other words, all of the closed-loop simulation runs starting from 74 different initial conditions are able to converge to the terminal set within the simulation period while not entering the unsafe region and not exiting the stability region.

### 6.4. Varying number of layers

The relation between model depths and generalization performance are also studied, where FNN models with various number

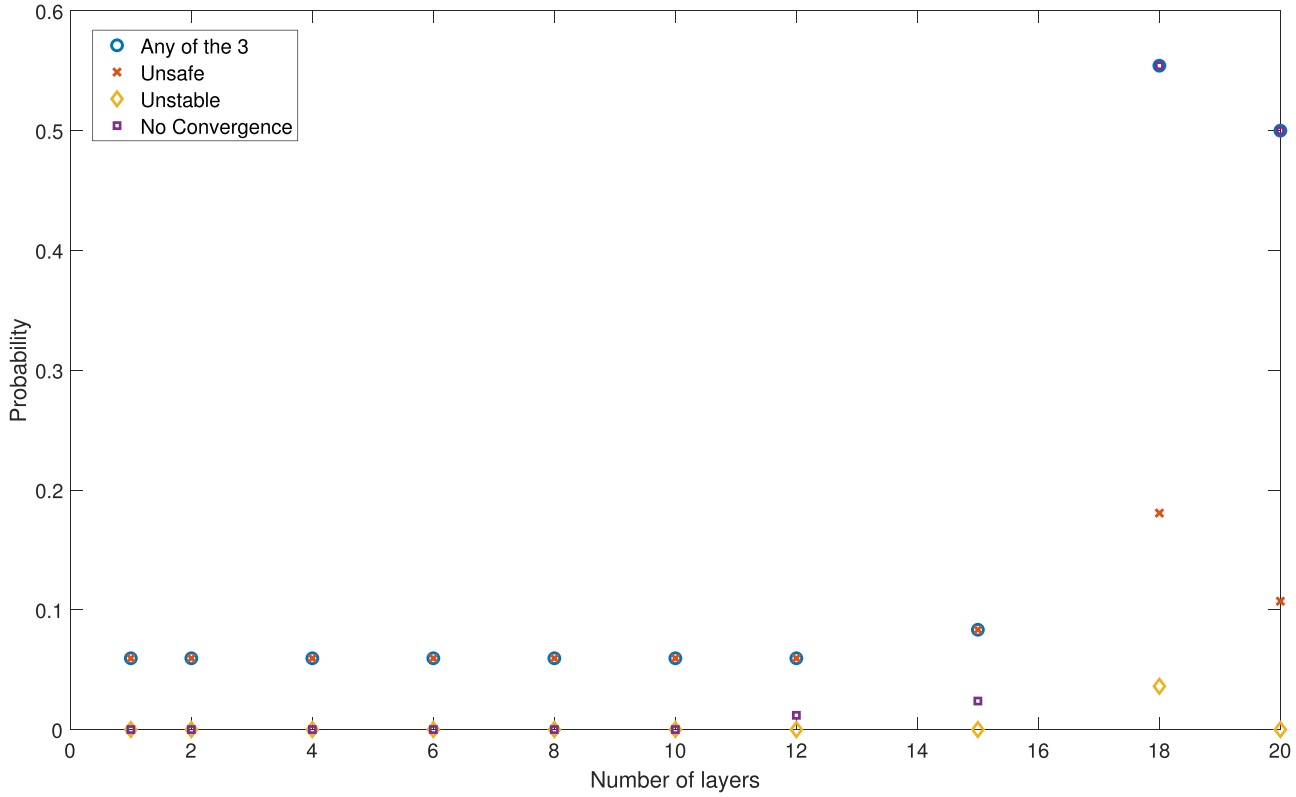


**Fig. 8.** Probabilities of unsafe, unstable, and non-convergent behavior under closed-loop control of the FNN-based CLBF-MPC for FNN models trained with varying layers in the case of bounded unsafe region.

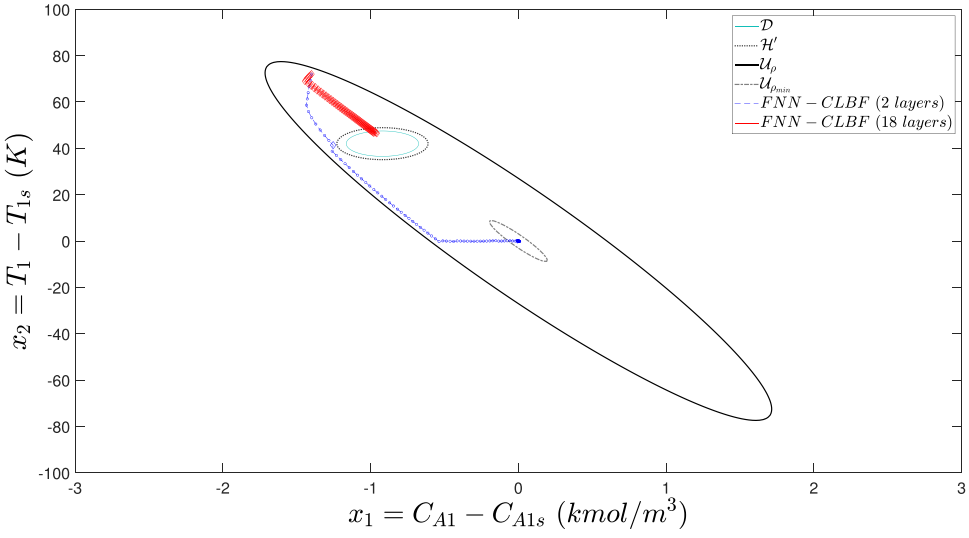


**Fig. 9.** Closed-loop state trajectories under CLBF-MPC with FNN-based barrier function trained with 2 layers (blue) vs. 18 layers (red), where states classified as safe by each FNN model are labelled by circle markers, and states classified as unsafe by each FNN model are labelled by diamond markers.

of layers from $n_l = 1$ to $n_l = 20$ are constructed with 10 neurons within each hidden layer. The same data generation and sampling method is used as in the case study of varying number of neurons. The generalization performance for FNN models with various number of layers for both cases of bounded and unbounded unsafe sets are shown in Figs. 6 and 7, respectively.

In the case of bounded unsafe sets, the misclassification rate and the output MSE for the testing dataset are maintained below $5.6 \times 10^{-2}$ and $1.79 \times 10^{-1}$ respectively for layers $n_l = 1$ to $n_l = 12$, and the testing errors are shown to be higher than the training errors for all layers. For layers $n_l \geq 15$, the generalization error gap between testing and training error drastically increases, which can be attributed to the model being overfitted, thus unable to generalize to new data as effectively. It is also observed that both training and testing error increase as the number of layer increases for $n_l \geq 15$. This is a common phenomenon that has been seen in neural networks with increasing depth; some possible explanations include: the network may be not able to find an appropriate mapping between two consecutive layers and becomes hard to optimize, or higher-level layers may lose access to important lower-level layer features. However, this remains a topic that is continuously studied by researchers. For the third error metric which

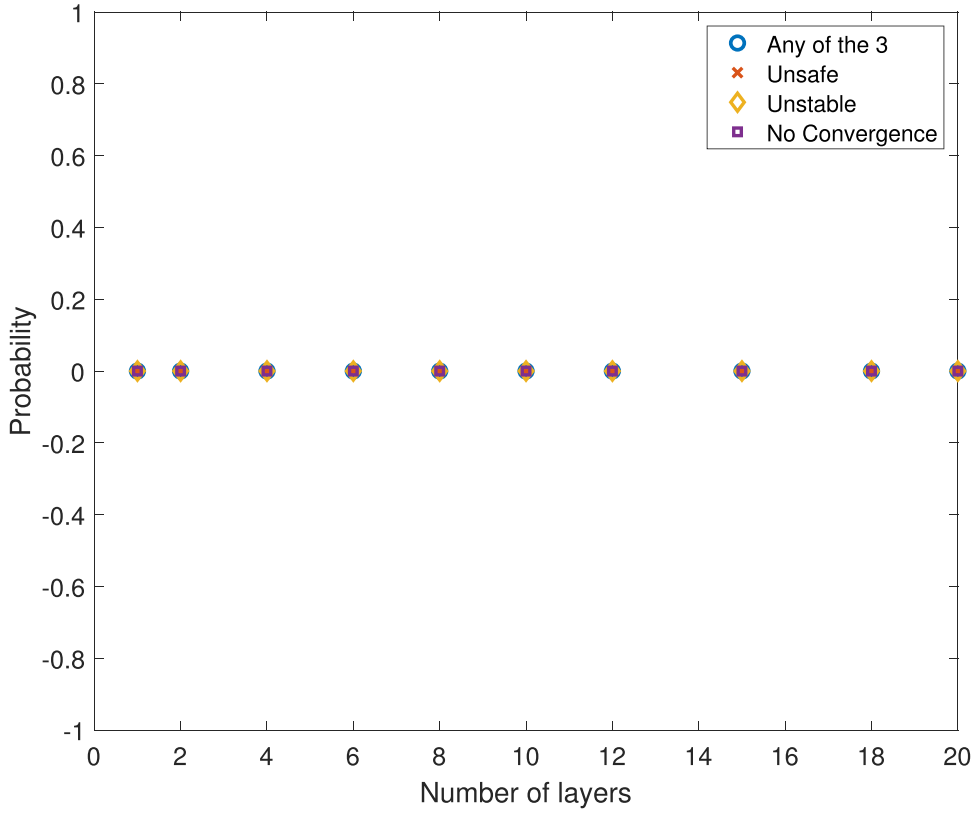


**Fig. 10.** Probabilities of unsafe, unstable, and non-convergent behavior under closed-loop control of the FNN-based CLBF-MPC for FNN models trained with varying layers in the case of unbounded unsafe regions.

assesses the negative definiteness of $L_f\hat{B}$, all models produced an average $ReLu(L_f\hat{B}(x) + 0.01)$ of less than 0.01 in both training and testing datasets, and this error is maintained under $2.8 \times 10^{-4}$ in both training and testing sets for models with $n_l \geq 2$. This shows that an FNN of at least 2 layers is needed.

In the case of unbounded unsafe sets, the resulting training and testing misclassification rate and output MSE are again sporadic because their values are already low for all layers. The highest misclassification rate and output MSE are $1.37 \times 10^{-2}$ and $3.65 \times 10^{-2}$ respectively, which are obtained at $n_l = 5$. For the first two error metrics, the testing dataset consistently yields a higher error than the training dataset, which agrees with the theoretical development of Section 4. The training loss from $ReLu(L_f\hat{B} + 0.01)$ is oddly higher than the testing losses for $n_l = 1, 2, 3$. The highest loss of this error metric is 0.0109 for training and $6.8 \times 10^{-3}$ for testing at $n_l = 2$. For $n_l \geq 3$, $L_f\hat{B}(x) < 0, x \in \{\mathcal{U}_\rho | L_g\hat{B}(x) = 0\}$ holds for both training and testing datasets. For this particular study, Fig. 7 shows that it is best to choose an FNN built with 4 layers.

Closed-loop probability studies are also conducted for both bounded and unbounded unsafe sets. For bounded unsafe sets, the probability of non-convergent behavior starts increasing for $n_l \geq 12$, and the probability of unsafe, unstable behavior starts increasing for $n_l \geq 15$. The probabilities are plotted against varying FNN depth in Fig. 8. This is consistent with the generalization error performance, where the model becomes overfitted as the number of layer increases beyond $n_l \geq 15$, and the training error, the testing error, as well as the generalization error gap all increase. The larger the generalization error gap, the less likely that closed-loop stability and safety can be guaranteed, thus the occurrences of unstable, unsafe, and non-convergent behavior increase.

We further demonstrate the difference in closed-loop performance between two models trained with different number of layers for systems with bounded unsafe sets in state space. Fig. 9 shows two state profiles under the FNN-CLBF-MPC, one of them has an FNN barrier function trained with 2 layers (blue), and the other one has an FNN barrier function trained with 18 layers (red). Along the red-colored state trajectory, all state values are falsely identified as "unsafe" by the 18-layer FNN barrier function, causing the closed-loop state to move very slowly, eventually into the unsafe region and unable to escape. The blue-colored trajectory starts from the same initial conditon, and is driven inside the terminal set while avoiding the unsafe set successfully. Along this trajectory, only one state at $x^T = [-1.2537, 41.3475]$, which is outside the unsafe region, is being falsely identified as "unsafe". This is because the predicted unsafe region $\hat{\mathcal{H}}$ characterized by the FNN barrier function $\hat{B}(x)$ constructed using 2 layers turns out to be a superset of $\mathcal{H}'$, which allows the MPC to act preemptively before the state actually enters $\mathcal{H}'$.

Similarly, the probability of unsafe, unstable, and non-convergent behaviors for the case of unbounded unsafe regions are shown in Fig. 10. Due to the consistently low modeling error, the probability of such behavior is zero across all variations of FNN depth.

### 6.5. Varying training data sample size

Lastly, the number of training data sample size is varied to examine its impact on the generalization error and closed-loop performance. The same data generation method is applied with the exception of varying the discretization grid size along each dimension of $x$ from $n_d = 10, 20, 30, \ldots, 450, 500$. The resulting datasets range from having a total sample size of $m = 38$ to $m = 94251$. The same neural network structure is used, consisting of 2 hidden layers of 10 neurons each. Figs. 11 and 12 illustrate the three error

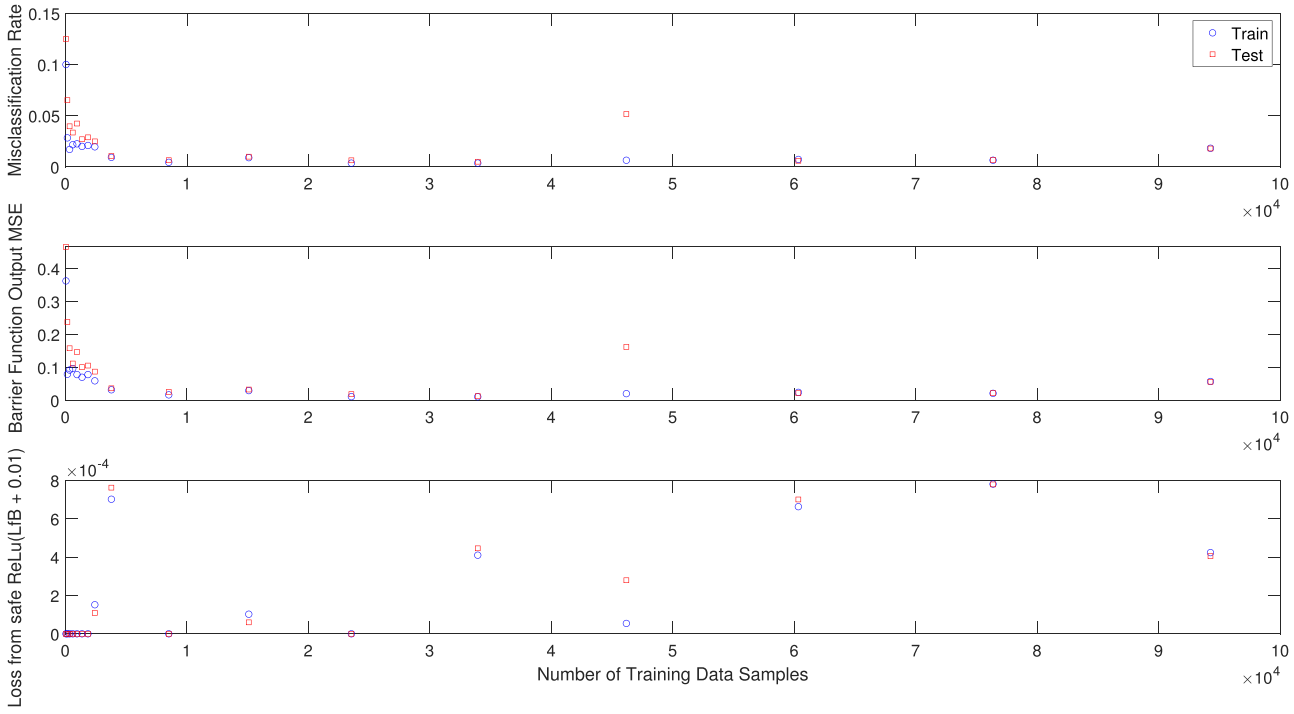


**Fig. 11.** Generalization performance for the FNN models for characterizing bounded unsafe regions utilizing various data sample size.

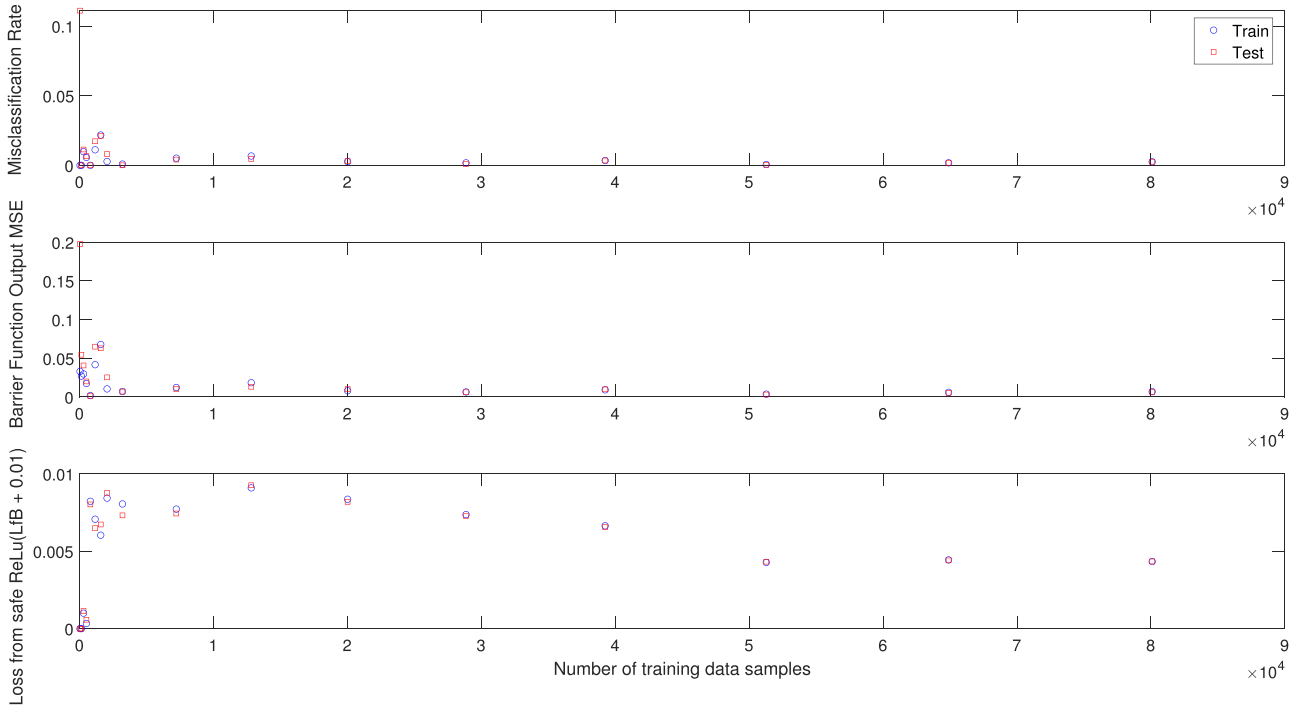


**Fig. 12.** Generalization performance for the FNN models for characterizing unbounded unsafe region utilizing various data sample size.

metrics in training and testing datasets for the bounded and unbounded unsafe sets respectively.

In the case of bounded unsafe regions, it is seen that for training data sample size between $m = 38$ to $m = 3775$, both training and testing sets produce high misclassification rate and the output MSE. The testing errors are higher than the testing errors with a generalization error gap, and the magnitude of these errors as well as the generalization error gap between training and testing sets decrease as the sample size increases. This is aligned with theoretical derivations as larger sample size results in improved model accuracy and reduced generalization error. The generalization gap, which captures the difference between expected error (testing dataset) and empirical error (training dataset), is roughly proportional to $\frac{1}{\sqrt{m}}$ as indicated by Eq. (19). This is consistent with the trend observed here where the decrease is drastic when $m$ is small, and reaches a plateau as $m$ increases to larger values. For data sample size $m \geq 8499$, both training and testing errors in the first two metrics stay consistently low below 0.018 and 0.056

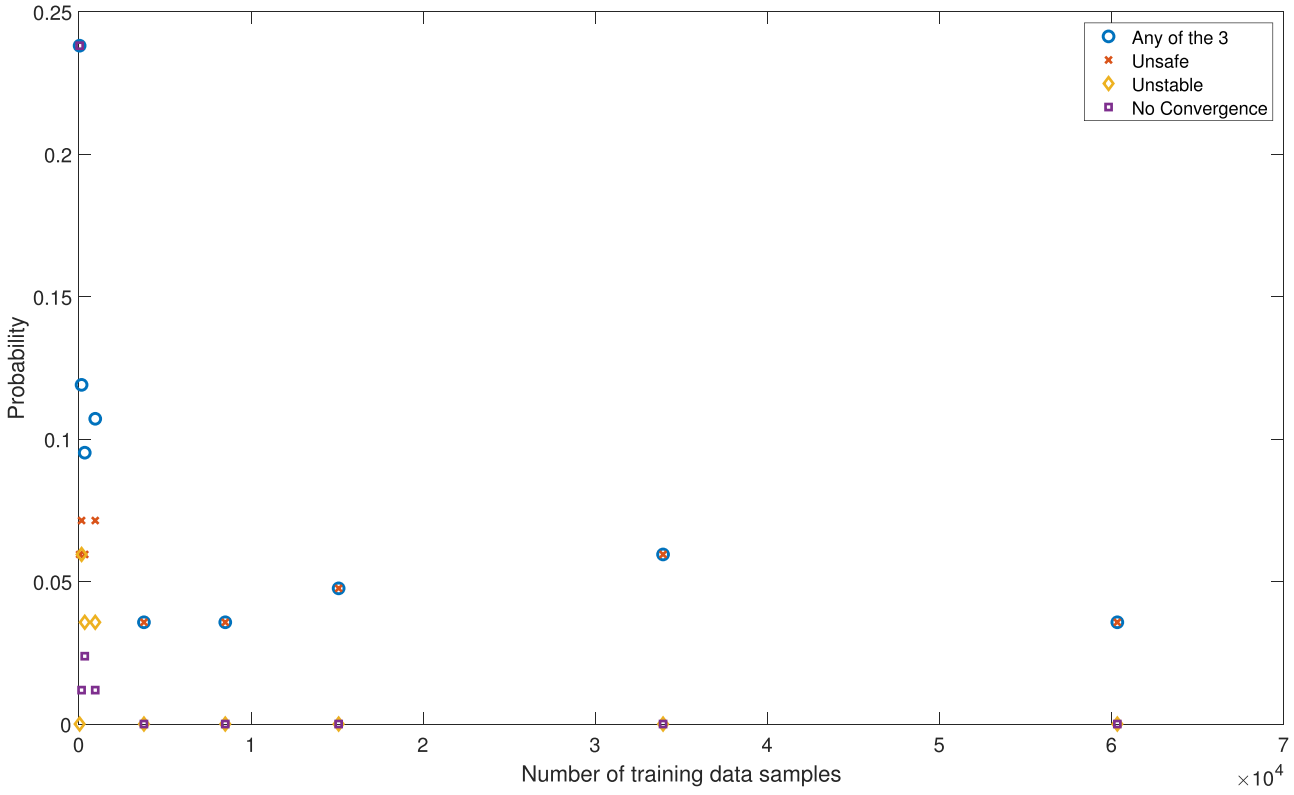


**Fig. 13.** Probabilities of unsafe, unstable, and non-convergent behavior under closed-loop control of the FNN-based CLBF-MPC for FNN models trained with varying data sample size in the case of bounded unsafe region.

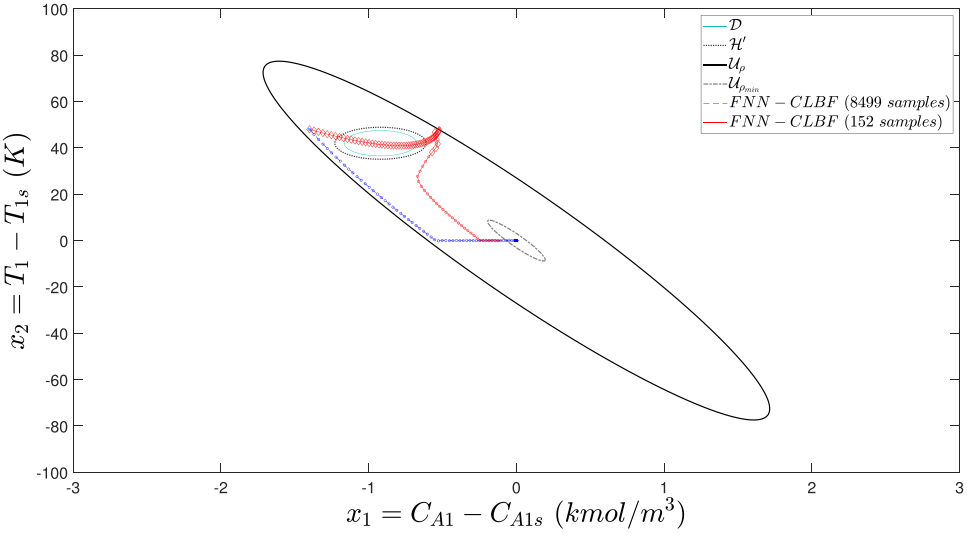


**Fig. 14.** Closed-loop state trajectories under CLBF-MPC with FNN-based barrier function trained with 8499 data samples (blue) vs. 152 data samples (red), where states classified as safe by each FNN model are labelled by circle markers, and states classified as unsafe by each FNN model are labelled by diamond markers.

for misclassification and MSE respectively, and no significant improvement is seen beyond $m \geq 8499$. For the losses calculated from $ReLu(L_f\hat{B}(x) + 0.01)$, it is observed that both training and testing errors are able to achieve extremely low values for $m = 38$ to $m = 3775$ where the misclassification rate and MSE are high. This may be because the data samples are too few for the FNN to learn the underlying relation between input and output, and therefore it fails to minimize $L_1$ and only stresses on satisfying $L_2$. The maximum $ReLu(L_f\hat{B}(x) + 0.01)$ for all models is $7.79 \times 10^{-4}$ and therefore the expected $L_f\hat{B}(x)$ stays below 0 for all models.

In the case of unbounded unsafe regions, it is similarly seen in Fig. 12 that the testing error and generalization error gap for misclassification rate and output MSE at the smallest sample size $m = 38$ is drastically higher than the other FNN models trained with larger training sample size, and they reach a low, stable level after $m \geq 3199$. For the loss term of $ReLu(L_f\hat{B}(x) + 0.01)$, all losses stay below $9.27 \times 10^{-3}$, which means that the expected $L_f\hat{B}(x) < 0$ for all models.

We also simulate closed-loop runs starting from various initial conditions within the operating region to assess probabilities

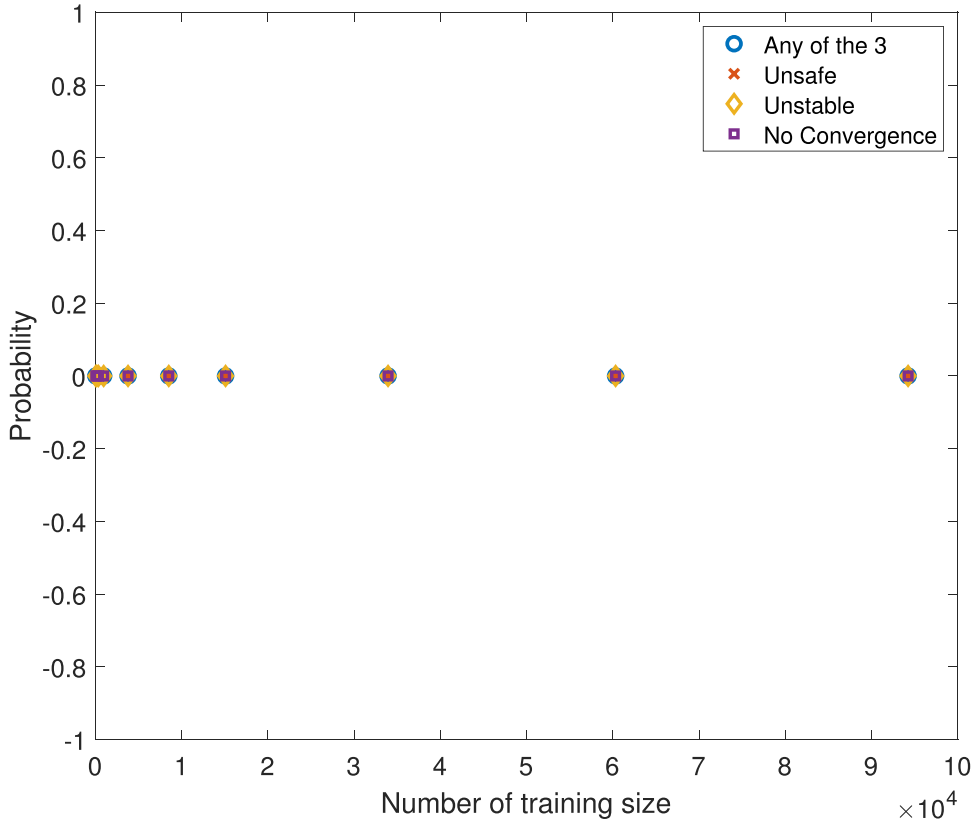


**Fig. 15.** Probabilities of unsafe, unstable, and non-convergent behavior under closed-loop control of the FNN-based CLBF-MPC for FNN models trained with varying data sample size in the case of unbounded unsafe regions.

of unstable, unsafe, and non-convergent behaviors. It is shown in Fig. 13 that the probabilities of unsafe and non-convergent instances drop to 0 for $m \geq 3775$, and the probability of unsafe instances is also in general lower when the sample size is larger. This is better demonstrated in the comparison of two closed-loop trajectories shown in Fig. 14 where the closed-loop state under an FNN model trained with 152 samples (red) and under an FNN model trained with 8499 samples (blue) are plotted together. The FNN model trained with 152 samples incorrectly classifies the initial condition as well as many states around the unsafe region as "unsafe" (diamond markers), and the closed-loop state under this FNN-CLBF-MPC enters the unsafe region and eventually traverses across the unsafe region, exiting on the other side. The closed-loop state continues to migrate towards terminal set, where eventually the FNN model correctly identifies the state as being "safe" (labeled by circle markers), and the closed-loop state ultimately is driven inside the terminal set. The closed-loop state along the trajectory controlled by the FNN-CLBF-MPC trained with 8499 samples are all correctly classified as "safe", and the MPC is able to quickly drive and maintain the state inside the terminal set in a stable and safe manner. Closed-loop simulations are also conducted starting from various initial conditions inside the operating region for the unbounded unsafe sets, and the probabilities of any of these undesirable instances occurring are zero for all models, as shown in Fig. 15.

**Remark 3.** As shown in Theorem 2, the generalization performance of the FNN model depends on a number of factors, including the sample size $m$, the network weight matrix bounds $B_W$, the bound on the possible values of state vector as inputs to the FNN $B_X$, the depth of the neural network $d$, the output dimension $d_y$ and the input dimension $d_x$. In this study, we have demonstrated case study results on the impact of neural network hypothesis class complexity (depth and width) and the training sample size $m$ on the overall generalization and closed-loop performance. As an extension to this study, one may also study the impact of $B_W$, $B_X$, which have been investigated in (Wu et al., 2021), or $d_x$, by adjusting the number of input features if possible.

## 7. Conclusion

A machine-learning-based Control Lyapunov-Barrier Function is used to design model predictive controllers for nonlinear systems with the presence of bounded and unbounded unsafe sets. Specifically, an FNN model is used to construct the Control Barrier Function, for which the generalization error bound is analyzed using the Rademacher complexity method from statistical machine learning theory. Subsequently, probabilistic stability and safety is established for CLBF-based control laws designed using this FNN-CBF, which is then extended to the sample-and-hold implementation of an FNN-CLBF-MPC. We demonstrate the impact that structural complexity and sample size of the FNN model have on the generalization performance, as well as the probabilities of closed-loop stability and safety in the cases of bounded and unbounded unsafe sets through a chemical reactor example.

## Declaration of Competing Interest

The authors have no affiliation with any organization with a direct or indirect financial interest in the subject matter discussed in the manuscript

## CRediT authorship contribution statement

**Scarlett Chen:** Conceptualization, Methodology, Software, Writing – original draft. **Zhe Wu:** Conceptualization, Methodology,

Writing – original draft. **Panagiotis D. Christofides:** Writing – review & editing.

## References


Althoff, M., Le Guernic, C., Krogh, B.H., 2011. Reachable set computation for uncertain time-varying linear systems. In: Proceedings of the 14th international conference on Hybrid systems: computation and control, pp. 93–102. Chicago, USA

Ames, A.D., Grizzle, J.W., Tabuada, P., 2014. Control barrier function based quadratic programs with application to adaptive cruise control. In: Proceedings of the 53rd IEEE Conference on Decision and Control, pp. 6271–6278. Los Angeles, California

Ames, A.D., Xu, X., Grizzle, J.W., Tabuada, P., 2016. Control barrier function based quadratic programs with application to automotive safety systems. arXiv preprint arXiv:1609.06408.

Ames, A.D., Xu, X., Grizzle, J.W., Tabuada, P., 2017. Control barrier function based quadratic programs for safety critical systems. IEEE Trans. Automat. Control 62, 3861–3876.

Bobiti, R., Lazar, M., 2016. A sampling approach to finding Lyapunov functions for nonlinear discrete-time systems. In: In Proceedings of the 2016 European Control Conference (ECC), pp. 561–566. Aalborg, Denmark

Chen, S., Wu, Z., Christofides, P.D., 2021. Machine-learning-based construction of barrier functions and models for safe model predictive control. AIChE J. e17456.

Clark, A., 2019. Control barrier functions for complete and incomplete information stochastic systems. In: Proceedings of the 2019 American Control Conference, pp. 2928–2935. Philadelphia, USA

Eryarsoy, E., Koehler, G.J., Aytug, H., 2009. Using domain-specific knowledge in generalization error bounds for support vector machine learning. Decis. Support Syst. 46, 481–491.

Golowich, N., Rakhlin, A., Shamir, O., 2018. Size-independent sample complexity of neural networks. In: Proceedings of the Conference On Learning Theory, pp. 297–299. Stockholm, Sweden

Jakubovitz, D., Giryes, R., Rodrigues, M., 2019. Generalization error in deep learning. In: Compressed Sensing and its Applications. Springer, pp. 153–193.

Jin, W., Wang, Z., Yang, Z., Mou, S., 2020. Neural certificates for safe control policies. arXiv preprint arXiv:2006.08465.

Khojasteh, M.J., Dhiman, V., Franceschetti, M., Atanasov, N., 2020. Probabilistic safety constraints for learned high relative degree system dynamics. In: Learning for Dynamics and Control. PMLR, pp. 781–792.

Lin, Y., Sontag, E.D., 1991. A universal formula for stabilization with bounded controls. Syst. Control Lett. 16, 393–397.

Lindemann, L., Hu, H., Robey, A., Zhang, H., Dimarogonas, D.V., Tu, S., Matni, N., 2020. Learning hybrid control barrier functions from data. arXiv preprint arXiv:2011.04112.

Liu, S., Kumar, A.R., Fisac, J., Adams, R.P., Ramadge, P., 2021. Probf: learning probabilistic safety certificates with barrier functions. arXiv preprint arXiv:2112.12210.

Luo, W., Sun, W., Kapoor, A., 2020. Multi-robot collision avoidance under uncertainty with probabilistic safety barrier certificates. Adv. Neural Inf. Process. Syst. 33, 372–383.

Maurer, A., 2016. A vector-contraction inequality for rademacher complexities. In: Proceedings of the International Conference on Algorithmic Learning Theory, pp. 3–17. Bari, Italy

Mitra, S., Wongpiromsarn, T., Murray, R.M., 2013. Verifying cyber-physical interactions in safety-critical systems. IEEE Secur. Priv. 11, 28–37.

Mohri, M., Rostamizadeh, A., Talwalkar, A., 2018. Foundations of Machine Learning. MIT press.

Prajna, S., Jadbabaie, A., 2004. Safety verification of hybrid systems using barrier certificates. In: Proceedings of the 7th International Workshop, HSCC, Vol. 2993, pp. 477–492. Philadelphia, Pennsylvania

Ratschan, S., She, Z., 2007. Safety verification of hybrid systems by constraint propagation-based abstraction refinement. ACM Trans. Embed. Comput. Syst. 6, 573–589.

Richards, S.M., Berkenkamp, F., Krause, A., 2018. The lyapunov neural network: adaptive stability certification for safe learning of dynamical systems. In: Proceedings of Conference on Robot Learning, pp. 466–476. Zurich, Switzerland

Robey, A., Hu, H., Lindemann, L., Zhang, H., Dimarogonas, D.V., Tu, S., Matni, N., 2020. Learning control barrier functions from expert demonstrations. In: Proceedings of the 59th IEEE Conference on Decision and Control, pp. 3717–3724. Jeju Island, South Korea

Romdlony, M.Z., Jayawardhana, B., 2016. Stabilization with guaranteed safety using control Lyapunov–barrier function. Automatica 66, 39–47.

Sontag, E.D., 1989. A 'universal' construction of Artstein's theorem on nonlinear stabilization. Syst. Control Lett. 13, 117–123.

Sontag, E.D., 1992. Neural nets as systems models and controllers. Yale University, pp. 73–79.

Srinivasan, M., Dabholkar, A., Coogan, S., Vela, P.A., 2020. Synthesis of control barrier functions using a supervised machine learning approach. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 7139–7145. Las Vegas, USA

Valiant, L.G., 1984. A theory of the learnable. Commun. ACM 27 (11), 1134–1142.

Wieland, P., Allgöwer, F., 2007. Constructive safety using control barrier functions. IFAC Proc. Vol. 40, 462–467.

Wu, Z., Albalawi, F., Zhang, Z., Zhang, J., Durand, H., Christofides, P.D., 2019. Control Lyapunov-barrier function-based model predictive control of nonlinear systems. Automatica 109, 108508.

Wu, Z., Christofides, P.D., 2019. Handling bounded and unbounded unsafe sets in control Lyapunov-barrier function-based model predictive control of nonlinear processes. Chem. Eng. Res. Des. 143, 140–149.

Wu, Z., Christofides, P.D., 2020. Control lyapunov-barrier function-based predictive control of nonlinear processes using machine learning modeling. Comput. Chem. Eng. 134, 106706.

Wu, Z., Durand, H., Christofides, P.D., 2018. Safe economic model predictive control of nonlinear systems. Syst. Control Lett. 118, 69–76.

Wu, Z., Rincon, D., Gu, Q., Christofides, P.D., 2021. Statistical machine learning in model predictive control of nonlinear processes. Mathematics 9, 1912.

Wu, Z., Tran, A., Rincon, D., Christofides, P.D., 2019. Machine learning-based predictive control of nonlinear processes. part II: computational implementation. AIChE J. 65, e16734.

Xu, X., 2016. Control sharing barrier functions with application to constrained control. In: Proceedings of the 55th Conference on Decision and Control, pp. 4880–4885. Las Vegas, USA

Xu, X., Tabuada, P., Grizzle, J.W., Ames, A.D., 2015. Robustness of control barrier functions for safety critical control. IFAC-PapersOnLine 48 (27), 54–61.

Yaghoubi, S., Fainekos, G., Sankaranarayanan, S., 2020. Training neural network controllers using control barrier functions in the presence of disturbances. In: Proceedings of the 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC). IEEE, pp. 1–6. Rhodes, Greece

Zhao, H., Zeng, X., Chen, T., Liu, Z., 2020. Synthesizing barrier certificates using neural networks. In: Proceedings of the 23rd International Conference on Hybrid Systems: Computation and Control, pp. 1–11. Sydney, Australia