Received: 24 April 2019 | Revised: 26 June 2019 | Accepted: 16 July 2019

DOI: 10.1002/aic.16729

PROCESS SYSTEMS ENGINEERING

AIChE JOURNAL

Machine learning-based predictive control of nonlinear processes. Part I: Theory

Zhe Wu[1] | Anh Tran[1] | David Rincon[1] | Panagiotis D. Christofides[1,2]

[1]Department of Chemical and Biomolecular Engineering, University of California, Los Angeles, California

[2]Department of Electrical and Computer Engineering, University of California, Los Angeles, California

Correspondence
Panagiotis Christofides, Department of Chemical and Biomolecular Engineering, University of California, Los Angeles, CA.
Email: pdc@seas.ucla.edu

Abstract

This article focuses on the design of model predictive control (MPC) systems for nonlinear processes that utilize an ensemble of recurrent neural network (RNN) models to predict nonlinear dynamics. Specifically, RNN models are initially developed based on a data set generated from extensive open-loop simulations within a desired process operation region to capture process dynamics with a sufficiently small modeling error between the RNN model and the actual nonlinear process model. Subsequently, Lyapunov-based MPC (LMPC) that utilizes RNN models as the prediction model is developed to achieve closed-loop state boundedness and convergence to the origin. Additionally, machine learning ensemble regression modeling tools are employed in the formulation of LMPC to improve prediction accuracy of RNN models and overall closed-loop performance while parallel computing is utilized to reduce computation time. Computational implementation of the method and application to a chemical reactor example is discussed in the second article of this series.

KEYWORDS

ensemble learning, model predictive control, nonlinear systems, process control, recurrent neural networks

1 | INTRODUCTION

Machine learning has attracted an increased level of attention in model identification in recent years. Among many machine learning techniques, recurrent neural networks (RNN) have been widely used for modeling a general class of nonlinear dynamical systems. Unlike the one-way connectivity between units in feedforward neural networks, there exist feedback loops in RNN architectures that introduce the past information derived from earlier inputs to the current network. The information preserved in the internal states exhibits the memory of an RNN and leads to capturing dynamic behavior in a way conceptually similar to nonlinear state-space ordinary differential equation models. The history of RNN can be traced back to the 1980s, when Hopfield networks were first created for pattern recognition.[1] Since then, many learning algorithms (e.g., backpropagation through time and truncated backpropagation methods) and modern RNN structures (e.g., long short-term memory (LSTM), gated recurrent unit (GRU) and bidirectional RNN) have been developed for various applications including human visual pattern recognition and natural language processing. With the rapid development of computational resources, an explosion of data and open-source software libraries such as Tensorflow and Keras, machine learning techniques have become accessible in classical engineering fields in addition to computer science and engineering. Specifically, the neural network method has been recently used to solve classification and regression problems in process control and operations. For example, in Reference 2 an adaptive fault-tolerant control method based on neural networks was developed for nonlinear systems with unknown actuator fault. In Reference 3 a neural network-based detection system was developed to detect cyber-attacks in chemical processes. In Reference 4 a convex-based LSTM neural network was developed for dynamic system identification. The aforementioned works have demonstrated the potential of neural networks in solving fault-tolerant control, cybersecurity, and real-time control and optimization problems.

As neural networks are able to approximate any continuous function according to the universal approximation theorem, neural networks can also be utilized to derive a nonlinear prediction model for model predictive control (MPC). As an advanced control methodology, MPC has been applied in real-time operation of industrial chemical plants to optimize process performance accounting for process stability, control actuator, and safety/state constraints.[5,6] One key requirement of MPC is the availability of an accurate process model to predict states. The prediction model can be derived as a first-principles model that describes the underlying physiochemical mechanisms or a data-driven model that is developed based on industrial/simulation data. Considering that in most cases it is difficult to obtain a first-principles model that captures complex, nonlinear behavior of a large-scale process, data-driven modeling has historically received significant attention. In References 7 and 8 the multivariable output error state-space algorithm (MOESP) and the numerical algorithm for subspace state-space identification (N4SID) were developed to identify state-space models. In Reference 9 nonlinear autoregressive moving average with exogenous input (NARMAX) method was introduced to identify nonlinear input–output systems. Additionally, modeling through recurrent neural networks has also proven to be successful in approximating nonlinear dynamical systems.[10,11] Extensive research efforts have been made on data-driven modeling, which also contributes to the development of model-based control schemes that utilize data-driven models to predict process dynamics.

Recently, a Lyapunov-based economic MPC that utilized a well-conditioned polynomial nonlinear state-space (PNLSS) model was developed to obtain optimal economic benefits and closed-loop stability.[12] Besides polynomial approximation that is generally easy to solve, neural networks have also been utilized in model-based controller design[13,14] as they may capture better "difficult nonlinearities" via a richer class of nonlinear functions that can be learned effectively. It was found in Reference 13 that feedforward neural networks are easy to implement to approximate static functions, but with slow convergence speed and lack of capturing process dynamics. Neural networks with similar structure such as shape-tunable neural network (SNN) and radial basis function neural network (RBNN) were also developed to approximate nonlinear dynamical systems. For example, in References 15 and 16 RBNN models were incorporated in the design of model predictive control to guarantee system stability. However, at this stage, stability analysis and computation time reduction have not been addressed for model predictive control using recurrent neural networks.

Moreover, given that a single data-driven model may not perfectly represent the process dynamics in the entire operating region, multimodel approaches were utilized in References 17 and 18 by partitioning the region of operation to reduce the reliance on a single model. For instance, a multimodel approach was used for fault identification in an industrial system using estimation techniques in Reference 18. In this case, the region of operation was partitioned around the operation points and each region of operation was represented by a single linear dynamic model. Consequently, the performance of the identified model in each region of operation was limited by the quality of provided data, which may not represent accurately the studied region. To address the above issue, ensemble learning has been proposed to combine the results of multiple models for complex systems. Ensemble learning uses several models that are obtained during a learning step to approximate particular outputs.[19] Compared to a single model prediction, ensemble learning has demonstrated benefits in robustness and accuracy in solving classification or regression problems.[20,21] In Reference 22 ensemble learning-based MPC has proven to be successful in regulating product yield for an industrial-scale fixed-bed reactor with a highly exothermic reaction. Additionally, in References 23 and 24 different ensemble learning methods were introduced to improve model prediction performance of neural network models for a batch polymerization reactor. However, the increasing computation time for multiple machine learning models still remains an important issue for real-time implementation of ensemble regression models.

Motivated by the above, this work develops a Lyapunov-based MPC using an ensemble of RNN models to drive the closed-loop state to its steady state for initial conditions in the closed-loop stability region. Specifically, an RNN model is initially developed to represent a general class of nonlinear processes over a broad region of process operation. Subsequently, the Lyapunov-based MPC using the RNN model that achieves sufficiently small modeling error during training is developed to stabilize nonlinear systems at the steady state, for which sufficient conditions are derived for closed-loop stability. Additionally, ensemble regression modeling and parallel computing are employed to improve prediction accuracy of RNN models and computational efficiency of LMPC optimization problem in real-time implementation. Part I of this two-article series presents the development of the LMPC using an ensemble of RNN models with stability analysis. Part II addresses computational implementation issues, parallel computing, and the application of LMPC to a chemical process example.

The rest of this article is organized as follows: in preliminaries, the notations, the class of nonlinear systems considered and the stabilizability assumptions are given. In section "Recurrent Neural Network," the general form of RNNs, the learning algorithm, and the process of building an RNN and ensemble regression models are introduced. In section "Lyapunov-based MPC using Ensemble RNN Models," a Lyapunov-based MPC is developed by incorporating an RNN model as the prediction model with guaranteed closed-loop stability and recursive feasibility. Stability analysis is performed accounting for the impact of sample-and-hold implementation, bounded disturbances and the modeling error between RNNs and the actual nonlinear system.

2 | PRELIMINARIES

2.1 | Notation

The notation $|\cdot|$ is used to denote the Euclidean norm of a vector. x^T denotes the transpose of x. The notation $L_f V(x)$ denotes the standard Lie derivative $L_f V(x) := \frac{\partial V(x)}{\partial x} f(x)$. Set subtraction is denoted by "\", that

is, $A\backslash B := \{x \in \mathbf{R}^n \mid x \in A, x \notin B\}$. $\emptyset$ signifies the null set. The function $f(\cdot)$ is of class $\mathcal{C}^1$ if it is continuously differentiable in its domain. A continuous function α: $[0,a) \to [0,\infty)$ is said to belong to class $\mathcal{K}$ if it is strictly increasing and is zero only when evaluated at zero.

2.2 | Class of systems

The class of continuous-time nonlinear systems considered is described by the following system of first-order nonlinear ordinary differential equations:

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

where $x \in \mathbf{R}^n$ is the state vector, $u \in \mathbf{R}^m$ is the manipulated input vector, and $w \in W$ is the disturbance vector with $W := \{w \in \mathbf{R}^q \mid |w| \le w_m,\ w_m \ge 0\}$. The control action constraint is defined by $u \in U := \{u_i^{\min} \le u_i \le u_i^{\max}, i = 1,\ldots,m\} \subset \mathbf{R}^m$. $f(\cdot)$, $g(\cdot)$, and $h(\cdot)$ are sufficiently smooth vector and matrix functions of dimensions $n \times 1$, $n \times m$, and $n \times q$, respectively. Throughout the manuscript, the initial time t_0 is taken to be zero ($t_0 = 0$), and it is assumed that f(0) = 0, and thus, the origin is a steady state of the nominal (i.e., w(t) ≡ 0) system of Equation (1) (i.e., $(x_s^*, u_s^*) = (0,0)$, where x_s^* and u_s^* represent the steady-state state and input vectors, respectively).

2.3 | Stabilization via control Lyapunov function

We assume that there exists a stabilizing control law $u = \Phi(x) \in U$ (e.g., the universal Sontag control law[25]) such that the origin of the nominal system of Equation (1) (i.e., w(t) ≡ 0) is rendered exponentially stable in the sense that there exists a $\mathcal{C}^1$ Control Lyapunov function $V(x)$ such that the following inequalities hold for all x in an open neighborhood D around the origin:

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

$$\frac{\partial V(x)}{\partial x} F(x,\Phi(x),0) \le -c_3|x|^2, \tag{2b}$$

$$\left|\frac{\partial V(x)}{\partial x}\right| \le c_4 |x| \tag{2c}$$

where c_1, c_2, c_3, and c_4 are positive constants. $F(x,u,w)$ represents the nonlinear system of Equation (1). A candidate controller $\Phi(x)$ is given in the following form:

$$\varphi_i(x) = \begin{cases} -\dfrac{p + \sqrt{p^2 + q^4}}{q^T q} q & \text{if } q \neq 0 \\ 0 & \text{if } q = 0 \end{cases} \tag{3a}$$

$$\Phi_i(x) = \begin{cases} u_i^{\min} & \text{if } \varphi_i(x) < u_i^{\min} \\ \varphi_i(x) & \text{if } u_i^{\min} \le \varphi_i(x) \le u_i^{\max} \\ u_i^{\max} & \text{if } \varphi_i(x) > u_i^{\max} \end{cases} \tag{3b}$$

where p denotes $L_f V(x)$ and q denotes $L_{gi} V(x)$, $f = [f_1 \cdots f_n]^T$, $g_i = [g_{i1}, \ldots, g_{in}]^T$, ($i$ = 1, 2, …, m). $\varphi_i(x)$ of Equation (3a) represents the i_{th} component of the control law $\varphi(x)$. $\Phi_i(x)$ of Equation (3) represents the i_{th} component of the saturated control law $\Phi(x)$ that accounts for the input constraint $u \in U$. Based on Equation (2), we can first characterize a region where the time-derivative of V is rendered negative under the controller $\Phi(x) \in U$ as $\varphi_u = \left\{x \in \mathbf{R}^n \mid \dot{V}(x) = L_f V + L_g V u < -kV(x), u = \Phi(x) \in U\right\} \cup \{0\}$, where k is a positive real number. Then the closed-loop stability region Ω_ρ for the nonlinear system of Equation (1) is defined as a level set of the Lyapunov function, which is inside φ_u: $\Omega_\rho := \{x \in \varphi_u \mid V(x) \le \rho\}$, where $\rho > 0$ and $\Omega_\rho \subset \varphi_u$. Also, the Lipschitz property of $F(x,u,w)$ combined with the bounds on u and w implies that there exist positive constants M, L_x, L_w, L_x', L_w' such that the following inequalities hold for all x, $x' \in D$, $u \in U$, and $w \in W$:

$$|F(x,u,w)| \le M \tag{4a}$$

$$|F(x,u,w) - F(x',u,0)| \le L_x |x - x'| + L_w |w| \tag{4b}$$

$$\left|\frac{\partial V(x)}{\partial x} F(x,u,w) - \frac{\partial V(x')}{\partial x} F(x',u,0)\right| \le L_x' |x - x'| + L_w' |w| \tag{4c}$$

3 | RECURRENT NEURAL NETWORK

In this work, we develop an RNN model with the following form:

$$\dot{\hat{x}} = F_{nn}(\hat{x},u) := A\hat{x} + \Theta^T y \tag{5}$$

where $\hat{x} \in \mathbf{R}^n$ is the RNN state vector and $u \in \mathbf{R}^m$ is the manipulated input vector. $y = [y_1, \ldots, y_n, y_{n+1}, \ldots, y_{m+n}] = [\sigma(\hat{x}_1), \ldots, \sigma(\hat{x}_n), u_1, \ldots, u_m] \in \mathbf{R}^{n+m}$ is a vector of both the network state $\hat{x}$ and the input u, where $\sigma(\cdot)$ is the nonlinear activation function (e.g., a sigmoid function $\sigma(x) = 1/(1 + e^{-x})$). A is a diagonal coefficient matrix, that is, $A = diag\{-a_1, \ldots, -a_n\} \in \mathbf{R}^{n \times n}$, and $\Theta = [\theta_1, \ldots, \theta_n] \in \mathbf{R}^{(m+n) \times n}$ with $\theta_i = b_i[w_{i1}, \ldots, w_{i(m+n)}]$, i = 1,…, n. a_i and b_i are constants. w_{ij} is the weight connecting the jth input to the ith neuron where i = 1,…, n and j = 1,…, ($m + n$). Additionally, a_i is assumed to be positive such that each state $\hat{x}_i$ is bounded-input bounded-state stable. Throughout the manuscript, we use x to represent the state of actual nonlinear system of Equation (1) and use $\hat{x}$ for the state of the RNN model Equation (5).

Unlike the one-way connectivity between units in feedforward neural networks (FNN), RNNs have signals traveling in both directions by introducing loops in the network. As shown in Figure 1, the states information derived from earlier inputs are fed back into the network, which exhibits a dynamic behavior. Consider the problem of approximating a class of continuous-time nonlinear systems of Equation (1) by an RNN model of Equation 5. Based on the universal approximation theorem for FNNs, it is shown in, for example, References 9 and 20 that the RNN model with sufficient number of neurons is able to approximate any dynamic nonlinear system on compact subsets of the state-space for finite time. The following proposition demonstrates the approximation property of the RNN model:

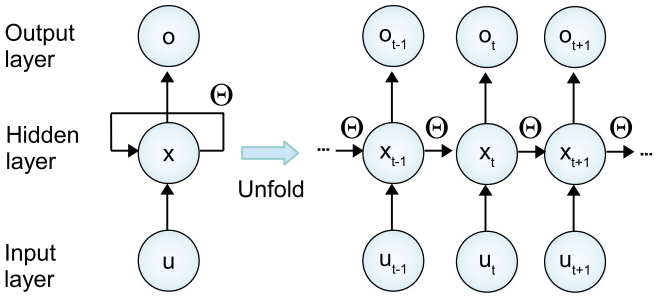

FIGURE 1 A recurrent neural network and its unfolded structure, where Θ is the weight matrix, x is the state vector, u is the input vector, and o is the output vector (for the nonlinear system in the form of Equation (1), the output vector is equal to the state vector) [Color figure can be viewed at wileyonlinelibrary.com]

Proposition 1 [26] *Consider the nonlinear system of Equation (1) and the RNN model of Equation (5) with the same initial condition* $x(0) = \hat{x}(0) = x_0 \in \Omega_\rho$. *For any* $\varepsilon > 0$ *and* $T > 0$, *there exists an optimal weight matrix* Θ^* *such that the state* $\hat{x}$ *of the RNN model of Equation (5) with* $\Theta = \Theta^*$ *satisfies the following equation:*

$$\sup_{t\in[0,T]} |x(t) - \hat{x}(t)| \le \epsilon \tag{6}$$

Remark 1 The RNN model of Equation (5) is developed as a continuous-time network as it is utilized to approximate the input–output behavior of the continuous-time nonlinear system of Equation (1). As discussed in Reference 27 continuous-time RNNs have many advantages, for example, the well-defined derivative of the internal state with respect to time. However, it should be noted that the discrete-time RNNs can be equally well applied to model the nonlinear system of Equation (1), where similar learning procedures and stability analysis can also be derived. Additionally, in our work, the RNN model of Equation (5) and the controller are both simulated in a sample-and-hold fashion with a sufficiently small sampling period Δ.

3.1 | RNN learning algorithm

In this section, the RNN learning algorithm is developed to obtain the optimal weight matrix Θ^*, under which the error between the actual state $x(t)$ of the nominal system of Equation (1) (i.e., $w(t) \equiv 0$) and the modeled states $\hat{x}(t)$ of the RNN of Equation (5) is minimized. Although it is demonstrated in Proposition 1 that RNNs can approximate a broad class of nonlinear systems to any degree of accuracy, it is acknowledged that RNN modeling may not be perfect in many cases due to insufficient number of nodes or layers. Therefore, we assume that there exists a modeling error $\nu := F(x,u,0) - F_{nn}(\hat{x},u)$ between the nominal system of Equation (1) and the RNN model of Equation (5) with $\Theta = \Theta^*$. As we focus on the system dynamics of Equation (1) in a compact set Ω_ρ, from which the origin can be rendered exponentially stable using the controller $u = \Phi(x) \in U$, the RNN model is developed to capture the system dynamics for all $x \in \Omega_\rho$ and $u \in U$. It is noted that the modeling error $\nu(t)$ is bounded (i.e., $|\nu(t)| \le \nu_m$, $\nu_m > 0$) as $x(t)$ and $u(t)$ are bounded. Additionally, to avoid the weight drift problem (i.e., the weights go to infinity during training), the weight vector θ_i is bounded by $|\theta_i| \le \theta_m$, where $\theta_m > 0$. Following the methods in References 10 and 28 the RNN learning algorithm is developed to demonstrate that the state error $|e| = |\hat{x} - x|$ is bounded in the presence of the modeling error ν.

Specifically, the RNN model is identified in the form of Equation (5) and the nominal system of Equation (1) (i.e., $w(t) \equiv 0$) can be expressed by the following equation:

$$\dot{x}_i = -a_i x_i + \theta_i^{*T} y + \nu_i,\ i = 1,\ldots,n \tag{7}$$

The optimal weight vector θ_i^* is defined as follows:

$$\theta_i^* := \arg\min_{|\theta_i| \le \theta_m} \left\{ \sum_{k=1}^{N_d} |F_i(x_k,u_k,0) + a_i x_k - \theta_i^T y_k| \right\} \tag{8}$$

where N_d is the number of data samples used for training. The state error is defined as $e = \hat{x} - x \in \mathbf{R}^n$. Based on Equations (5) and (7), the time-derivative of state error is derived as follows:

$$\dot{e}_i = \dot{\hat{x}}_i - \dot{x}_i = -a_i e_i + \zeta_i^T y - \nu_i, i = 1,\ldots,n \tag{9}$$

where $\zeta_i = \theta_i - \theta_i^*$ is the error between the current weight vector θ_i and the unknown optimal weight vector θ_i^*. ν is the modeling error given by $\nu = F(x,u,0) - Ax - \Theta^* y$. The weight vector θ is updated during the training process as follows:

$$\dot{\theta}_i = -\eta_i y e_i,\ i = 1,\ldots,n \tag{10}$$

where the learning rate η is a positive definite matrix. Based on the learning law of Equation (10), the following theorem is established to demonstrate that the state error e remains bounded and is constrained by the modeling error ν.

Theorem 1 Consider the RNN model of Equation *(5)* of which the weights are updated according to Equation *(10)*. Then, the state error e_i and the weight error ζ_i are bounded, and there exist $\lambda \in \boldsymbol{R}$ and $\mu > 0$ such that the following inequality holds:

$$\int_0^t |e(\tau)|^2 d\tau \le \lambda + \mu \int_0^t |\nu(\tau)|^2 d\tau \tag{11}$$

Proof We first define a Lyapunov function $\tilde{V} = \frac{1}{2}\sum_{i=1}^n \left(e_i^2 + \zeta_i^T \eta_i^{-1} \zeta_i\right)$. Based on Equations (9) and (10) and $\dot{\zeta}_i = \dot{\theta}_i$, the time-derivative of $\tilde{V}$ is derived as follows:

$$\begin{aligned} \dot{\tilde{V}} &= \sum_{i=1}^n \left(e_i \dot{e}_i + \eta_i^{-1} \zeta_i \dot{\zeta}_i \right) \\ &= \sum_{i=1}^n \left(-a_i e_i^2 - e_i \nu_i \right) \end{aligned} \tag{12}$$

It is noted that in the absence of modeling error (i.e., $\nu_i = 0$), it holds that $\dot{\tilde{V}} \le 0$. Following the proof in Reference 10 it is shown that

the state error e_i and its time-derivative $\dot{e}_i$ are bounded for all times. Additionally, as $\tilde{V}$ is bounded from below and $\dot{\tilde{V}}$ is uniformly continuous implied by the fact that the second order derivative $\ddot{\tilde{V}}$ is bounded, it follows that $\dot{\tilde{V}} \to 0$ as $t \to \infty$ according to Barbalat's lemma.[29] This implies that e_i ultimately converges to zero if there is no modeling error term $-e_i\nu_i$ in Equation (12). However, in the presence of modeling error $\nu_i \neq 0$, $\dot{\tilde{V}} \leq 0$ does not hold for all times. Therefore, based on Equation (12), the following equation is further derived:

$$\begin{aligned}\dot{\tilde{V}} &= \sum_{i=1}^{n}\left(-\frac{a_i}{2}e_i^2 - \frac{1}{2}|\zeta_i|^2\right) + \left(\frac{1}{2}|\zeta_i|^2 - \frac{a_i}{2}e_i^2 - e_i\nu_i\right) \\ &\leq -\alpha\tilde{V} + \sum_{i=1}^{n}\left(\frac{1}{2}|\zeta_i|^2 - \left(\frac{a_i}{2}e_i^2 + e_i\nu_i + \frac{1}{2a_i}\nu_i^2\right) + \frac{1}{2a_i}\nu_i^2\right) \\ &\leq -\alpha\tilde{V} + \sum_{i=1}^{n}\left(\frac{1}{2}|\zeta_i|^2 + \frac{1}{2a_i}\nu_i^2\right)\end{aligned} \tag{13}$$

where $\alpha := min\{a_i, 1/(\lambda_m), i = 1,..., n\}$ and λ_m represents the maximum eigenvalue of η_i^{-1}. As the weight vector is bounded by $|\theta_i| \leq \theta_m$, it is derived that $\frac{1}{2}|\zeta_i|^2 \leq 2\theta_m^2$, and it follows that $\dot{\tilde{V}} \leq -\alpha\tilde{V} + \beta$, where $\beta := \sum_{i=1}^{n}\left(2\theta_m^2 + \nu_m^2/2a_i\right)$. Therefore, $\dot{\tilde{V}} \leq 0$ holds for all $\tilde{V} \geq V_0 = \beta/\alpha$, which implies that $\tilde{V}$ is bounded. From the definition of $\tilde{V}$, it is readily shown that e_i and ζ_i are bounded as well. Moreover, based on the fact that $\dot{\tilde{V}} \leq \sum_{i=1}^{n}\left(-\frac{a_i}{2}e_i^2 + \frac{1}{2a_i}\nu_i^2\right)$ derived from Equation (13), we can also derive $\tilde{V}(t)$ as follows:

$$\begin{aligned}\tilde{V}(t) &\leq \tilde{V}(0) + \sum_{i=1}^{n}\left(-\frac{a_i}{2}\int_0^t e_i(\tau)^2 d\tau + \frac{1}{2a_i}\int_0^t \nu_i(\tau)^2 d\tau\right) \\ &\leq \tilde{V}(0) - \frac{a_{\min}}{2}\int_0^t |e(\tau)|^2 d\tau + \frac{1}{2a_{\min}}\int_0^t |\nu(\tau)|^2 d\tau\end{aligned} \tag{14}$$

where a_{min} is the minimum value of a_i, $i = 1,...,n$. Let $\lambda = \frac{2}{a_{\min}}\sup_{t\geq 0}\left(\tilde{V}(0) - \tilde{V}(t)\right)$ and $\mu = 1/a_{\min}^2$. The relationship between $|e|$ and $|\mu|$ shown in Equation 11 is derived as follows:

$$\begin{aligned}\int_0^t |e(\tau)|^2 d\tau &\leq \frac{2}{a_{\min}}\left(\tilde{V}(0) - \tilde{V}(t)\right) + \frac{1}{a_{\min}^2}\int_0^t |\nu(\tau)|^2 d\tau \\ &\leq \lambda + \mu\int_0^t |\nu(\tau)|^2 d\tau\end{aligned} \tag{15}$$

Therefore, it is guaranteed that the state error $|e|$ is bounded and is proportional to the modeling error $|\nu|$. Furthermore, it is noted that if there exists a positive real number $C > 0$ such that $\int_0^\infty |\nu(t)|^2 dt = C < \infty$, then it follows that $\int_0^\infty |e(t)|^2 dt \leq \lambda + \mu C < \infty$. As $e(t)$ is uniformly continuous (i.e., $\dot{e}$ is bounded), it implies that $e(t)$ converges to zero asymptotically.

Remark 2 As the weights may drift to infinity in the presence of modeling error, a robust learning algorithm named switching σ-modification approach was proposed in References 9 and 16 to adjust the weight such that the assumption that the weight vector θ_i is bounded by $|\theta_i| \leq \theta_m$ holds for all times. The switching σ-modification approach was then improved to be continuous in the considered compact set to overcome the problem of existence and uniqueness of solutions. The interested reader may refer to References 9 and 16 for further information.

3.2 | Development of RNN model

In this section, we discuss how to develop an RNN model from scratch for a general class of nonlinear system of Equation (1) within an operating region. Specifically, the development of a neural network model includes the generation of dataset and the training process.

3.2.1 | Data generation

Open-loop simulations are first conducted to generate the dataset that captures the system dynamics for $x \in \Omega_\rho$ and $u \in U$ as we focus on the system dynamics of Equation (1) in a compact set Ω_ρ with the constrained input vector $u \in U$. Given various $x_0 \in \Omega_\rho$, the open-loop simulations of the nominal system of Equation 1 are repeated under various inputs u to obtain a large number of trajectories for finite time to sweep over all the values that (x,u) can take. However, it is noted that due to the limitation of computational resources, we may have to discretize the targeted region in state-space and the range of inputs with sufficiently small intervals in practice as shown in Figure 2. Subsequently, time-series data from open-loop simulations can be separated into a large number of time-series samples with a shorter period P_{nn}, which represents the prediction period of RNNs. Lastly, the dataset is partitioned into training, validation, and testing datasets.

Additionally, it should be noted that we simulate the continuous system of Equation (1) under a sequence of inputs $u \in U$ in a sample-and-hold fashion (i.e., the inputs are fed into the system of Equation (1) as a piecewise constant function, $u(t) = u(t_k)$, $\forall t \in [t_k, t_{k+1})$, where $t_{k+1} := t_k + \Delta$ and Δ is the sampling period). Then, the nominal system of Equation (1) is integrated via explicit Euler method with a sufficiently small integration time step $h_c < \Delta$.

Remark 3 In this study, we utilize the above data generation method to obtain the dataset consisting of fixed-length, time-series state trajectories within the closed-loop stability region Ω_ρ for the nonlinear system of Equation (1). There also exist other data generation approaches for data-driven modeling, for example, simulations under pseudo-random binary input signals (PRBS)[30] that excite the nonlinearity behavior of system dynamics within the targeted region. However, regarding handling unstable steady-states, finite-length open-loop simulations with various initial conditions $x_0 \in \Omega_\rho$ outperform the PRBS method in that the state trajectories can be bounded in or stay close to Ω_ρ within a short period of time, while states may diverge quickly under continuous PRBS inputs.

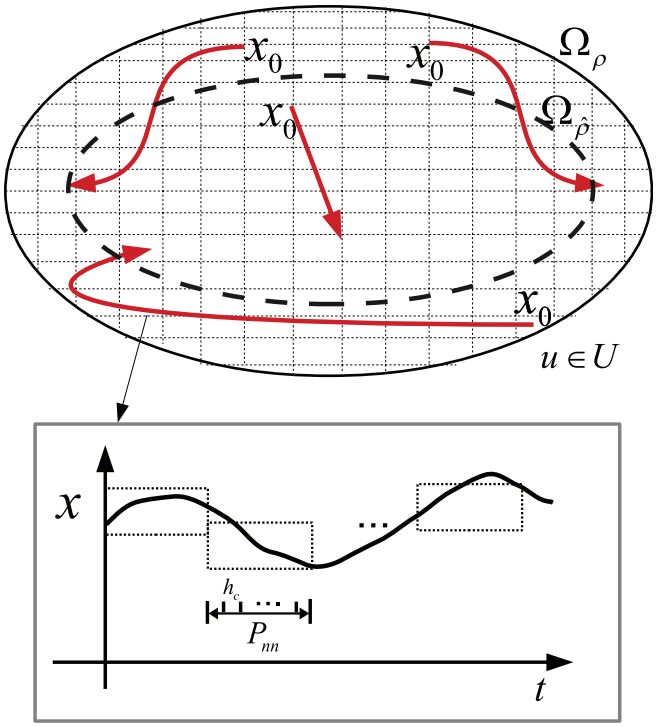

FIGURE 2 The schematic of discretization of the operating region Ω_ρ and the generation of training data for RNNs with a prediction period P_{nn} for all initial conditions $x_0 \in \Omega_\rho$, where h_c is the time interval between RNN internal states, Ω_ρ is the closed-loop stability region for the actual nonlinear system of Equation (1) and $\Omega_{\hat{\rho}}$ is the closed-loop stability region characterized for the obtained RNN model [Color figure can be viewed at wileyonlinelibrary.com]

3.2.2 | Training process

Next, the RNN model is developed using a state-of-the-art application program interface (API), that is, Keras.[31] The prediction period of RNN, P_{nn}, is chosen to be an integer multiple of the sampling period Δ such that the RNN model $F_{nn}(\hat{x},u)$ can be utilized to predict future states for at least one sampling period by taking state measurement $x(t_k)$ and manipulated inputs $u(t_k)$ at time $t = t_k$. As shown in Figure 2, within the RNN prediction period P_{nn}, the time interval between two consecutive internal states x_{t-1} and x_t for the unfolded RNN shown in Figure 1 is chosen to be the integration time step h_c used in open-loop simulations. Therefore, all the states between $t = 0$ and $t = P_{nn}$ with a step size of h_c are treated as the internal states and can be predicted by the RNN model. Based on the dataset generated from open-loop simulations, the RNN model of Equation (5) is trained to calculate the optimal weight Θ^* of Equation (8) by minimizing the modeling error between $F(x,u,0)$ and $F_{nn}(\hat{x},u)$. Furthermore, to guarantee that the RNN model of Equation (5) achieves good performance in a neighborhood around the origin and has the same equilibrium point as the nonlinear system of Equation (1), the modeling error is further required to satisfy $|\nu| \leq \gamma|x| \leq \nu_m$ during the training process.

Specifically, when we train an RNN using open-source neural-network libraries, for example, Keras, the optimization problem of minimizing the modeling error ν is solved using adaptive moment estimation method (i.e., Adam in Keras) with the loss function calculated by the mean squared error or the mean absolute percentage error between predicted states $\hat{x}$ and actual states x from training data. The optimal number of layers and neurons are determined through a grid search. Additionally, to avoid over-fitting of the RNN model, the training process is terminated once the modeling error falls below the desired threshold and the early-stopping condition is satisfied (i.e., the error on the validation set stops decreasing).

Remark 4 In some cases training datasets may consist of noisy data or corrupt data, which could affect the training performance of RNNs in the following manners. On the one hand, noise makes it more challenging for RNNs to fit data points precisely. On the other hand, it is shown in Reference 20 that the addition of noise can also improve generalization performance and robustness of RNNs, and sometimes even lead to faster learning. Therefore, the neural network training with noise remains an important issue that needs further investigation. However, in our work, it should be noted that the open-loop simulations are performed for the nominal system of Equation (1) (i.e., $w(t) \equiv 0$) only. Based on the noise-free dataset, the RNN models are trained to approximate the dynamics of the nominal system of Equation (1) within the closed-loop stability region $\Omega_{\hat{\rho}}$. Additionally, although the RNN model is derived for the nominal system of Equation (1), it will be shown in Section "Sample-and-hold Implementation of Lyapunov-based Controller" that the obtained RNN models can be applied with model predictive control to stabilize the nonlinear system of Equation (1) in the presence of small bounded disturbances (i.e., $|w| \leq w_m$).

Remark 5 In our study, the operating region considered is chosen as the closed-loop stability region Ω_ρ that is characterized for the first-principles model of Equation (1), in which there exist control actions $u \in U$ that render the steady state of Equation (1) exponentially stable for all $x_0 \in \Omega_\rho$. However, in the case that the closed-loop stability region Ω_ρ for the first-principles model of Equation (1) is unknown, the operating region Ω_ρ for collecting data can be initially characterized as a sufficiently large set around the steady state containing all the states that the system might move to. This operating region may be larger than the actual closed-loop stability region Ω_ρ, and therefore, it is likely that no control actions $u \in U$ exist to drive the state toward the origin for some initial conditions x_0 in this operating region. However, this problem can be overcome through the recharacterization of the closed-loop stability region $\Omega_{\hat{\rho}}$ after the RNN is obtained. Specifically, based on the dataset generated in this operating region, the RNN model is derived following the training approach in this section, and subsequently, the actual operating region (i.e., the closed-loop stability region $\Omega_{\hat{\rho}}$ for the RNN model of Equation (5)) will be characterized to ensure that for any $x_0 \in \Omega_{\hat{\rho}}$, there exist control actions $u \in U$ that render the steady state of the RNN model of Equation (5) exponentially stable. Additionally, as shown in Figure 2, the closed-loop stability region $\Omega_{\hat{\rho}}$ for the RNN model of Equation (5) is

characterized as a subset of Ω_ρ (Ω_ρ is a sufficiently large operating region if the first-principles model of Equation (1) is unknown, or represents the closed-loop stability region for the actual nonlinear system of Equation (1) when the first-principles model of Equation (1) is available).

Remark 6 In Figure 2, it is shown that open-loop simulation data is collected for all initial condition $x_0 \in \Omega_\rho$, yet the closed-loop stability region for the obtained RNN model may be characterized as a conservative region $\Omega_{\hat{\rho}} \subseteq \Omega_\rho$ due to model mismatch. The detailed characterization method will be given in Section "Lyapunov-based control using RNN models," and it will be shown that for any initial condition $x_0 \in \Omega_{\hat{\rho}}$, the existence of control actions $u \in U$ that can render the steady state the obtained RNN model of Equation (5) exponentially stable is guaranteed. Furthermore, it will be proven that for any initial conditions x_0 in the closed-loop stability region $\Omega_{\hat{\rho}}$ characterized for the RNN model of Equation (5), the steady state of the actual nonlinear system of Equation (1) can also be rendered exponentially stable under $u \in U$ provided that the modeling error $|\nu|$ is sufficiently small.

3.3 | Ensemble regression modeling

As a single RNN may not perform perfectly over the entire operating region due to insufficient data and inappropriate ratio between the training dataset and validation dataset, the ensemble learning method, which is a machine learning process that combines multiple learning algorithms to obtain better prediction performance,[19] is utilized to construct homogeneous ensemble regression models based on the RNN learning algorithm and the k-fold cross validation discussed in the previous section. Specifically, homogeneous ensemble regression models are derived from the ensemble learning method if a single base learning algorithm is used, while heterogeneous models are produced in the case of multiple learning algorithms. The reasons that ensemble regression models are able to improve the prediction performance are as follows.[19,22] First, a single RNN model that achieves a desired training accuracy may perform poorly in the region that lacks sufficient training data, while ensemble methods can reduce the risk of relying on a single flawed model by aggregating all candidate models. Second, the RNN learning algorithm is known to be a nonconvex, NP-hard problem that can result in locally optimal solutions. Therefore, by using different starting initial weight matrices, ensemble learning methods might be able to avoid getting trapped in a local minimum and obtain a better set of weights for RNN to accurately predict output sequences. Third, in the case that the single regression model cannot represent the true target function, the ensemble learning methods might be able to provide some good approximation. Therefore, by introducing ensemble learning into the development of an empirical model for the nonlinear system of Equation (1), the performance of ensemble regression models is expected to outperform that of a single RNN model in terms of reduced variability and improved generalization performance.

A rich collection of ensemble-based algorithms, for example, Boosting, Bagging and Stacking, have been developed over the past few years.[21] In our work, the stacking method is used to combine the predictions of multiple regression models developed based on the RNN learning algorithm. It is noted that we do not switch RNN models in the operating region. All the RNN models are developed to approximate the process dynamics for the same operating region. Specifically, following the approach of k-fold cross validation, the dataset generated from open-loop simulations is first split into k parts as shown in the dotted box in Figure 3. Then, the RNN model with the general structure shown in Figure 1 is trained using $k-1$ parts as training dataset and the remaining one as validation dataset to predict the nonlinear dynamics of the system of Equation (1). Based on the training dataset, the detailed RNN structure is shown at the bottom of Figure 3. In the input layer, the dimension of input nodes is $m + n$, where x_i, $i = 1,\ldots,n$ represents the real-time state measurements and u_i, $i = 1, \ldots, m$ represents the manipulated inputs at t_k. In the output layer, y_i, $i = 1, \ldots, n$ report the estimation of the states after the prediction period P_{nn}. The output vector and the internal states can be used as the predicted states in the optimization problem of MPC in the next section. Additionally, two hidden layers are used in the RNN model with the optimal number of neurons determined through a grid search.

As shown in Figure 3, the above training process is repeated to produce multiple RNN models using different $k-1$ sets as training dataset, and therefore, a total of k RNN models are developed based on k-fold cross validation. Subsequently, the final predicted states at $t = t_k + P_{nn}$ are calculated by a combiner algorithm that takes all the predictions generated by its constituent RNNs. For the sake of simplicity, in our work, we calculate the average of all prediction results from multiple RNNs as the final prediction results. The open-loop performance of the ensemble measured in terms of the absolute mean error is statistically better than that of a single RNN and the closed-loop performance of the application of ensemble regression models is evaluated in the example in the second article of this series. Additionally, in Figure 3, it is shown that normalizing and rescaling functions are employed before and after the ensemble of k RNN models. Specifically, the input vector, consisting of state measurements $x(t_k) \in \mathbf{R}^n$ and manipulated inputs $u(t_k) \in \mathbf{R}^m$ at $t = t_k$, is first normalized using the input statistics of the training dataset and fed into the cross-validated committee of RNNs. Subsequently, the output vector, which is the estimated states at $t = t_k + P_{nn}$, is generated by rescaling the average of the normalized predicted states using the output statistics of the training dataset.

4 | LYAPUNOV-BASED MPC USING ENSEMBLE RNN MODELS

This section presents the formulation of LMPC that incorporates the RNN model to predict future states with a stability analysis of the closed-loop system of Equation (1). Specifically, the stability of

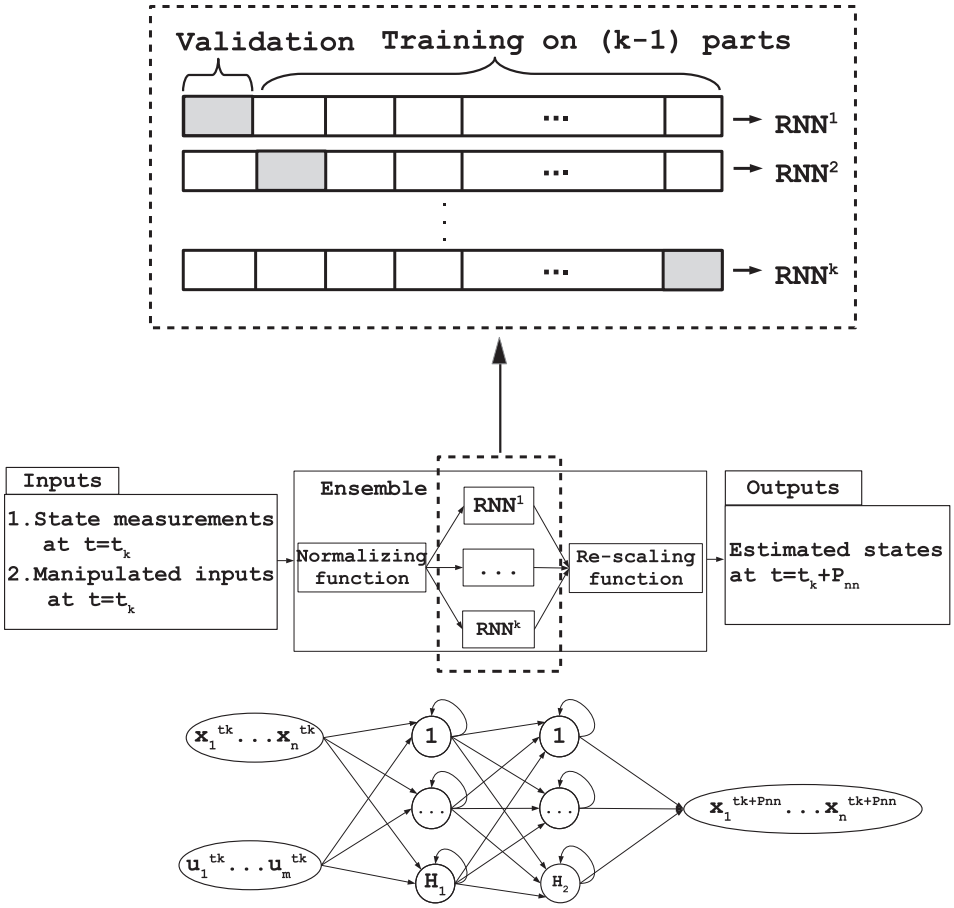

FIGURE 3 The structure of the ensemble regression models based on RNN learning algorithm and *k*-fold cross validation, where $x \in \mathbf{R}^n$ is the state vector, $u \in \mathbf{R}^m$ is the input vector, and H_1, H_2 are the number of neurons in hidden layers

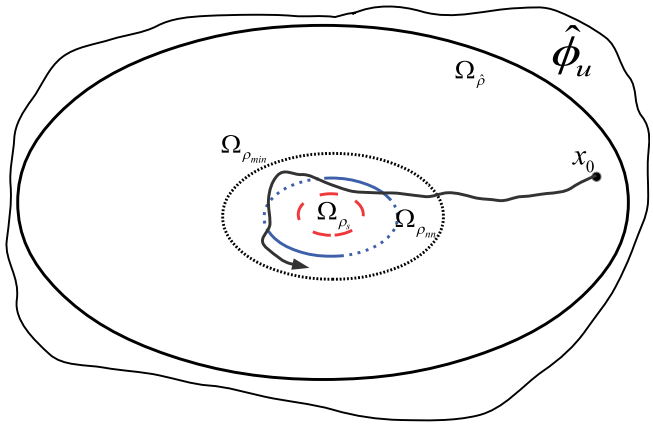

FIGURE 4 A schematic representing the set $\hat{\phi}_u$, the closed-loop stability region $\Omega_{\hat{\rho}}$, and the sets $\Omega_{\rho_{min}}$, $\Omega_{\rho_{nn}}$, Ω_{ρ_s} (going from outside to inside). Under the LMPC of Equation (36), the closed-loop state is driven toward the origin and ultimately bounded in $\Omega_{\rho_{min}}$ for any $x_0 \in \Omega_{\hat{\rho}}$ [Color figure can be viewed at wileyonlinelibrary.com]

the nonlinear system of Equation (1) under a Lyapunov-based controller derived from the RNN model of Equation (5) is first developed. Based on that, the RNN model of Equation (5) is incorporated into the design of LMPC under sample-and-hold implementation of control actions to drive the closed-loop state to a small neighborhood around the origin.

4.1 | Lyapunov-based control using RNN models

A stabilizing feedback controller $u = \Phi_{nn}(x) \in U$ that can render the origin of the RNN model of Equation (5) exponentially stable in an open neighborhood $\hat{D}$ around the origin is assumed to exist for the RNN model of Equation (5) in the sense that there exists a $\mathcal{C}^1$ Control Lyapunov function $\hat{V}(x)$ such that the following inequalities hold for all x in $\hat{D}$:

$$\hat{c}_1|x|^2 \le \hat{V}(x) \le \hat{c}_2|x|^2, \tag{16a}$$

$$\frac{\partial \hat{V}(x)}{\partial x} F_{nn}(x, \Phi_{nn}(x)) \le -\hat{c}_3|x|^2, \tag{16b}$$

$$\left|\frac{\partial \hat{V}(x)}{\partial x}\right| \le \hat{c}_4 |x| \tag{16c}$$

where $\hat{c}_1$, $\hat{c}_2$, $\hat{c}_3$, $\hat{c}_4$ are positive constants, and $F_{nn}(x,u)$ represents the RNN model of Equation (5). Similar to the characterization method of the closed-loop stability region Ω_ρ for the nonlinear system of Equation (1), we first search the entire state space to characterize a set of states $\hat{\phi}_u$ where the following inequality holds: $\dot{\hat{V}}(x) = \frac{\partial \hat{V}(x)}{\partial x} F_{nn}(x,u) < -k\hat{V}(x)$, $u = \Phi_{nn}(x) \in U$, $k > 0$. Starting from $\hat{\phi}_u$, the origin of the RNN model of Equation (5) can be rendered exponentially stable under the controller $u = \Phi_{nn}(x) \in U$. The closed-loop stability region

for the RNN model of Equation (5) is defined as a level set of Lyapunov function inside $\hat{\phi}_u$: $\Omega_{\hat{\rho}} := \left\{ x \in \hat{\phi}_u \mid \hat{V}(x) \le \hat{\rho} \right\}$, where $\hat{\rho} > 0$. It is noted that the above assumption of Equation (16) is the same as the assumption of Equation (2) for the general class of nonlinear systems of Equation (1) as the RNN model of Equation (5) can be written in the form of Equation (1) (i.e., $\dot{\hat{x}} = \hat{f}(\hat{x}) + \hat{g}(\hat{x})u$, where $\hat{f}(\cdot)$ and $\hat{g}(\cdot)$ are obtained from coefficient matrices A and Θ in Equation (5)). The assumptions of Equations 2 and 16 are essentially the stabilizability requirements of the first-principles model of Equation (1) and the RNN model of Equation (5), respectively.

It is noted that $\Omega_{\hat{\rho}} \subseteq \Omega_\rho$ as the dataset for developing the RNN model of Equation (5) is generated from open-loop simulations for $x \in \Omega_\rho$ and $u \in U$. Additionally, there exist positive constants M_{nn} and L_{nn} such that the following inequalities hold for all $x, x' \in \Omega_{\hat{\rho}}$ and $u \in U$:

$$| F_{nn}(x,u) | \le M_{nn} \tag{17a}$$

$$\left| \frac{\partial \hat{V}(x)}{\partial x} F_{nn}(x,u) - \frac{\partial \hat{V}(x')}{\partial x} F_{nn}(x',u) \right| \le L_{nn} | x - x' | \tag{17b}$$

Due to the model mismatch between the nominal system of Equation (1) and the RNN model of Equation (5), the following proposition is developed to demonstrate that the feedback controller $u = \Phi_{nn}(x) \in U$ is able to stabilize the nominal system of Equation (1) if the modeling error is sufficiently small.

Proposition 2 *Under the assumption that the origin of the closed-loop RNN system of Equation (5) is rendered exponentially stable under the controller $u = \Phi_{nn}(x) \in U$ for all $x \in \Omega_{\hat{\rho}}$, if there exists a positive real number $\gamma < \hat{c}_3/\hat{c}_4$ that constrains the modeling error $|\nu| = |F(x,u,0) - F_{nn}(x,u)| \le \gamma |x|$ for all $x \in \Omega_{\hat{\rho}}$ and $u \in U$, then the origin of the nominal closed-loop system of Equation (1) under $u = \Phi_{nn}(x) \in U$ is also exponentially stable for all $x \in \Omega_{\hat{\rho}}$.*

Proof To demonstrate that the origin of the nominal system of Equation (1) can be rendered exponentially stable $\forall x \in \Omega_{\hat{\rho}}$ under the controller for the RNN model of Equation (5), we prove that $\dot{\hat{V}}$ for the nominal system of Equation (1) can still be rendered negative under $u = \Phi_{nn}(x) \in U$. Based on Equation (16b) and (16c), the time-derivative of $\hat{V}$ is derived as follows:

$$\begin{aligned} \dot{\hat{V}} &= \frac{\partial \hat{V}(x)}{\partial x} F(x, \Phi_{nn}(x), 0) \\ &= \frac{\partial \hat{V}(x)}{\partial x} (F_{nn}(x, \Phi_{nn}(x)) + F(x, \Phi_{nn}(x), 0) - F_{nn}(x, \Phi_{nn}(x))) \\ &\le -\hat{c}_3 |x|^2 + \hat{c}_4 | x | (F(x, \Phi_{nn}(x), 0) - F_{nn}(x, \Phi_{nn}(x))) \\ &\le -\hat{c}_3 |x|^2 + \hat{c}_4 \gamma |x|^2 \end{aligned} \tag{18}$$

If γ is chosen to satisfy $\gamma < \hat{c}_3/\hat{c}_4$, then it holds that $\dot{\hat{V}} \le -\tilde{c}_3 |x|^2 \le 0$ where $\tilde{c}_3 = -\hat{c}_3 + \hat{c}_4 \gamma > 0$. Therefore, the closed-loop state of the nominal system of Equation (1) converges to the origin under $u = \Phi_{nn}(x) \in U$ for all $x_0 \in \Omega_{\hat{\rho}}$.

Remark 7 It should be noted that the RNN model of Equation (5) that is trained on the dataset within the operating region Ω_ρ may not be stabilizable at the origin for all $x \in \Omega_\rho$ under $u = \Phi_{nn}(x) \in U$ due to model mismatch. Therefore, in this section, a new closed-loop stability region $\Omega_{\hat{\rho}}$ is characterized for the RNN model of Equation (5) using the controller $\Phi_{nn}(x)$ in the form of Equation (3). Subsequently, it is shown in Proposition 2 that the controller $\Phi_{nn}(x)$ also stabilizes the actual nonlinear system of Equation (1) at the origin for all $x \in \Omega_{\hat{\rho}}$ provided that the modeling error is sufficiently small (i.e., $|\nu| \le \gamma |x| < \hat{c}_3 |x| / \hat{c}_4$). As the closed-loop stability region $\Omega_{\hat{\rho}}$ for the RNN model of Equation (5) guarantees the asymptotic stability of the origin for the nonlinear system of Equation (1) under $u = \Phi_{nn}(x) \in U$, $\Omega_{\hat{\rho}}$ will be taken as the new operating region for the operation of model predictive control in the following sections.

4.2 | Sample-and-hold implementation of Lyapunov-based controller

As the RNN model of Equation (5) will be incorporated in Lyapunov-based MPC design, for which the control actions are implemented in sample-and-hold, in this section, we first derive the following propositions demonstrating the sample-and-hold properties of the Lyapunov-based controller $u = \Phi_{nn}(x)$ in the presence of bounded disturbances (i.e., $|w(t)| \le w_m$). Specifically, the next proposition demonstrates that there exists an upper bound for the error between the state of the actual process of Equation (1) in the presence of bounded disturbances (i.e., $|w(t)| \le w_m$) and the state predicted by the RNN model of Equation (5).

Proposition 3 *Consider the nonlinear system $\dot{x} = F(x,u,w)$ of Equation (1) in the presence of bounded disturbances $|w(t)| \le w_m$ and the RNN model $\dot{\hat{x}} = F_{nn}(\hat{x},u)$ of Equation (5) with the same initial condition $x_0 = \hat{x}_0 \in \Omega_{\hat{\rho}}$. There exists a class $\mathcal{K}$ function $f_w(\cdot)$ and a positive constant κ such that the following inequalities hold $\forall x, \hat{x} \in \Omega_{\hat{\rho}}$ and $w(t) \in W$:*

$$| x(t) - \hat{x}(t) | \le f_w(t) := \frac{L_w w_m + \nu_m}{L_x} \left(e^{L_x t} - 1 \right) \tag{19a}$$

$$\hat{V}(x) \le \hat{V}(\hat{x}) + \frac{\hat{c}_4 \sqrt{\hat{\rho}}}{\sqrt{\hat{c}_1}} | x - \hat{x} | + \kappa | x - \hat{x} |^2 \tag{19b}$$

Proof Let $e(t) = x(t) - \hat{x}(t)$ denote the error vector between the solutions of the system $\dot{x} = F(x,u,w)$ and the RNN model $\dot{\hat{x}} = F_{nn}(\hat{x},u)$. The time-derivative of $e(t)$ is obtained as follows:

$$\begin{aligned} | \dot{e} | &= | F(x,u,w) - F_{nn}(\hat{x},u) | \\ &\le | F(x,u,w) - F(\hat{x},u,0) | + | F(\hat{x},u,0) - F_{nn}(\hat{x},u) | \end{aligned} \tag{20}$$

Following Equation (4b), for all $\forall x, \hat{x} \in \Omega_{\hat{\rho}}$ and $w(t) \in W$, it is obtained that

$$\begin{aligned} | F(x,u,w) - F(\hat{x},u,0) | &\le L_x | x(t) - \hat{x}(t) | + L_w | w(t) | \\ &\le L_x | x(t) - \hat{x}(t) | + L_w w_m \end{aligned} \tag{21}$$

Additionally, the second term $|F(\hat{x},u,0)-F_{nn}(\hat{x},u)|$ in Equation (20) represents the modeling error, which is bounded by $|\nu| \le \nu_m$ for all $\hat{x} \in \Omega_{\hat{\rho}}$. Therefore, based on Equation (21) and the fact that $|F(\hat{x},u,0)-F_{nn}(\hat{x},u)| \le \nu_m$, $\dot{e}(t)$ is bounded as follows:

$$\begin{aligned} |\dot{e}(t)| &\le L_x|x(t)-\hat{x}(t)| + L_w|w_m| + \nu_m \\ &\le L_x|e(t)| + L_w|w_m| + \nu_m \end{aligned} \tag{22}$$

Therefore, given the zero initial condition (i.e., $e(0) = 0$), the upper bound for the norm of the error vector is derived for all $x(t), \hat{x}(t) \in \Omega_{\hat{\rho}}$ and $|w(t)| \le w_m$ as follows:

$$|e(t)| = |x(t)-\hat{x}(t)| \le \frac{L_w w_m + \nu_m}{L_x}\left(e^{L_x t}-1\right) \tag{23}$$

Subsequently, $\forall x, \hat{x} \in \Omega_\rho$, Equation (19b) is derived based on the Taylor series expansion of $\hat{V}(x)$ around $\hat{x}$ as follows:

$$\hat{V}(x) \le \hat{V}(\hat{x}) + \frac{\partial \hat{V}(\hat{x})}{\partial x}|x-\hat{x}| + \kappa|x-\hat{x}|^2 \tag{24}$$

where κ is a positive real number. Using Equations (16a) and (16c), it follows that

$$\hat{V}(x) \le \hat{V}(\hat{x}) + \frac{\hat{c}_4\sqrt{\hat{\rho}}}{\sqrt{\hat{c}_1}}|x-\hat{x}| + \kappa|x-\hat{x}|^2 \tag{25}$$

This completes the proof of Proposition 3.

Subsequently, the following proposition is developed to demonstrate that the closed-loop state $x(t)$ of the actual process of Equation (1) is bounded in $\Omega_{\hat{\rho}}$ for all times, and ultimately can be driven to a small neighborhood $\Omega_{\rho_{min}}$ around the origin under the controller $u = \Phi_{nn}(x) \in U$ implemented in a sample-and-hold fashion.

Proposition 4 *Consider the system of Equation (1) under the controller $u = \Phi_{nn}(\hat{x}) \in U$ that is designed to stabilize the RNN system of Equation (5) and meets the conditions of Equation (16). The controller is implemented in a sample-and-hold fashion, that is, $u(t) = \Phi_{nn}(\hat{x}(t_k))$, $\forall t \in [t_k, t_{k+1})$, where $t_{k+1} := t_k + \Delta$. Let ϵ_s, $\epsilon_w > 0$, $\Delta > 0$ and $\hat{\rho} > \rho_{\min} > \rho_{nn} > \rho_s$ satisfy*

$$-\frac{\hat{c}_3}{\hat{c}_2}\rho_s + L_{nn}M_{nn}\Delta \le -\epsilon_s \tag{26a}$$

$$-\frac{\tilde{c}_3}{\hat{c}_2}\rho_s + L_x'M\Delta + L_w'w_m \le -\epsilon_w \tag{26b}$$

and

$$\rho_{nn} := \max\left\{\hat{V}(\hat{x}(t+\Delta)) \,|\, \hat{x}(t) \in \Omega_{\rho_s}, u \in U\right\} \tag{27a}$$

$$\rho_{\min} \ge \rho_{nn} + \frac{\hat{c}_4\sqrt{\hat{\rho}}}{\sqrt{\hat{c}_1}}f_w(\Delta) + \kappa(f_w(\Delta))^2 \tag{27b}$$

Then, for any $x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$, the following inequality holds:

$$\hat{V}(x(t)) \le \hat{V}(x(t_k)),\ \forall t \in [t_k, t_{k+1}) \tag{28}$$

and the state $x(t)$ of the nonlinear system of Equation (1) is bounded in $\Omega_{\hat{\rho}}$ for all times and ultimately bounded in $\Omega_{\rho_{min}}$.

Proof *Part* 1: Assuming $x(t_k) = \hat{x}(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$, we first prove that the value of $\hat{V}(\hat{x})$ is decreasing under the controller $u(t) = \Phi_{nn}(x(t_k)) \in U$ for $t \in [t_k, t_{k+1})$, where $x(t)$ and $\hat{x}(t)$ denote the solutions of the nonlinear system of Equation (1) in the presence of bounded disturbances and the RNN system of Equation (5), respectively. The time-derivative of the $\hat{V}(\hat{x})$ along the trajectory $\hat{x}(t)$ of the RNN model of Equation (5) in $t \in [t_k, t_{k+1})$ is obtained as follows:

$$\begin{aligned} \dot{\hat{V}}(\hat{x}(t)) &= \frac{\partial \hat{V}(\hat{x}(t))}{\partial \hat{x}}F_{nn}(\hat{x}(t), \Phi_{nn}(\hat{x}(t_k))) \\ &= \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}}F_{nn}(\hat{x}(t_k), \Phi_{nn}(\hat{x}(t_k))) + \frac{\partial \hat{V}(\hat{x}(t))}{\partial \hat{x}}F_{nn}(\hat{x}(t), \Phi_{nn}(\hat{x}(t_k))) \\ &\quad - \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}}F_{nn}(\hat{x}(t_k), \Phi_{nn}(\hat{x}(t_k))) \end{aligned} \tag{29}$$

Using Equations (16a) and (16b), the following inequality is obtained:

$$\begin{aligned} \dot{\hat{V}}(\hat{x}(t)) \le\ & -\frac{\hat{c}_3}{\hat{c}_2}\rho_s + \frac{\partial \hat{V}(\hat{x}(t))}{\partial \hat{x}}F_{nn}(\hat{x}(t), \Phi_{nn}(\hat{x}(t_k))) \\ & - \frac{\partial \hat{V}(\hat{x}(t_k))}{\partial \hat{x}}F_{nn}(\hat{x}(t_k), \Phi_{nn}(\hat{x}(t_k))) \end{aligned} \tag{30}$$

Based on the Lipschitz condition of Equation (17) and the fact that $\hat{x} \in \Omega_{\hat{\rho}}$, $u \in U$, the upper bound of $\dot{\hat{V}}(\hat{x}(t))$ is derived $\forall t \in [t_k, t_{k+1})$:

$$\begin{aligned} \dot{\hat{V}}(\hat{x}(t)) &\le -\frac{\hat{c}_3}{\hat{c}_2}\rho_s + L_{nn}|\hat{x}(t)-\hat{x}(t_k)| \\ &\le -\frac{\hat{c}_3}{\hat{c}_2}\rho_s + L_{nn}M_{nn}\Delta \end{aligned} \tag{31}$$

Therefore, if Equation (26a) is satisfied, the following inequality holds $\forall \hat{x}(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$ and $t \in [t_k, t_{k+1})$:

$$\dot{\hat{V}}(\hat{x}(t)) \le -\epsilon_s \tag{32}$$

By integrating the above equation over $t \in [t_k, t_{k+1})$, it is obtained that $V(\hat{x}(t_{k+1})) \le V(\hat{x}(t_k)) - \epsilon_s\Delta$. So far we have proved that for all $\hat{x}(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$, the state of the closed-loop RNN system of Equation (5) is bounded in the closed-loop stability region $\Omega_{\hat{\rho}}$ for all times and moves toward the origin under $u = \Phi_{nn}(\hat{x}) \in U$ implemented in a sample-and-hold fashion.

However, Equation (32) may not hold when $x(t_k) = \hat{x}(t_k) \in \Omega_{\rho_s}$, which implies that the state may leave Ω_{ρ_s} within one sampling period. Therefore, according to Equation (27a), $\Omega_{\rho_{nn}}$ is designed to ensure that the closed-loop state $\hat{x}(t)$ of the RNN model does not leave $\Omega_{\rho_{nn}}$ for all $t \in [t_k, t_{k+1})$, $u \in U$ and $\hat{x}(t_k) \in \Omega_{\rho_s}$ within one sampling period. If the state $\hat{x}(t_{k+1})$ leaves Ω_{ρ_s}, the controller $u = \Phi_{nn}(x(t_{k+1}))$ will drive the state toward Ω_{ρ_s} over the next sampling period as Equation (32) is

satisfied again at $t = t_{k+1}$. Therefore, the convergence of the state to $\Omega_{\rho_{nn}}$ for the closed-loop RNN system of Equation (5) is proved for all $\hat{x}_0 \in \Omega_{\hat{\rho}}$. It remains to show that the closed-loop state of the actual nonlinear system of Equation (1) can be bounded in $\Omega_{\hat{\rho}}$ for all times and ultimately bounded in a small neighborhood around the origin under the sample-and-hold implementation of the controller $u = \Phi_{nn}(x) \in U$.

Part 2: Following the analysis performed for the RNN system of Equation (5), we first assume $x(t_k) = \hat{x}(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$ and derive the time-derivative of $\hat{V}(x)$ for the nonlinear system of Equation (1) (i.e., $\dot{x} = F(x,u,w)$) in the presence of bounded disturbances (i.e., $|w| \leq w_m$) as follows:

$$\begin{aligned}\dot{\hat{V}}(x(t)) &= \frac{\partial \hat{V}(x(t))}{\partial x} F(x(t), \Phi_{nn}(x(t_k)), w) \\ &= \frac{\partial \hat{V}(x(t_k))}{\partial x} F(x(t_k), \Phi_{nn}(x(t_k)), 0) + \frac{\partial \hat{V}(x(t))}{\partial x} F(x(t), \Phi_{nn}(x(t_k)), w) \\ &\quad - \frac{\partial \hat{V}(x(t_k))}{\partial x} F(x(t_k), \Phi_{nn}(x(t_k)), 0)\end{aligned} \tag{33}$$

From Equation (18), $\frac{\partial \hat{V}(x(t_k))}{\partial x} F(x(t_k), \Phi_{nn}(x(t_k)), 0) \leq -\tilde{c}_3 |x(t_k)|^2$ holds for all $x \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$. Based on Equation (2a) and the Lipschitz condition in Equation (4), the following inequality is obtained for $\dot{\hat{V}}(x(t))$ for all $t \in [t_k, t_{k+1})$ and $x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$:

$$\begin{aligned}\dot{\hat{V}}(x(t)) &\leq -\frac{\tilde{c}_3}{\tilde{c}_2}\rho_s + \frac{\partial \hat{V}(x(t))}{\partial x} F(x(t), \Phi_{nn}(x(t_k)), w) - \frac{\partial \hat{V}(x(t_k))}{\partial x} F(x(t_k), \Phi_{nn}(x(t_k)), 0) \\ &\leq -\frac{\tilde{c}_3}{\tilde{c}_2}\rho_s + L'_x |x(t) - x(t_k)| + L'_w |w| \\ &\leq -\frac{\tilde{c}_3}{\tilde{c}_2}\rho_s + L'_x M\Delta + L'_w w_m\end{aligned} \tag{34}$$

Therefore, if Equation (26b) is satisfied, the following inequality holds $\forall x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_s}$ and $t \in [t_k, t_{k+1})$:

$$\dot{\hat{V}}(x(t)) \leq -\epsilon_w \tag{35}$$

From Equation (35), it is readily shown that Equation (28) holds and the state of the closed-loop system of Equation (1) is maintained in $\Omega_{\hat{\rho}}$ for all times. Also, it follows that the controller $u = \Phi_{nn}(x)$ is still able to drive the state of the actual nonlinear system of Equation (1) toward the origin in every sampling period. Additionally, if $x(t_k) \in \Omega_{\rho_s}$, it is shown in *Part* 1 that the state of the RNN model of Equation (5) is maintained in $\Omega_{\rho_{nn}}$ within one sampling period. Considering the bounded error between the state of the RNN of Equation (5) model and the state of the nonlinear system of Equation (1) given by Equation (19a), there exists a compact set $\Omega_{\rho_{min}} \supset \Omega_{\rho_{nn}}$ that satisfies Equation (27b) such that the state of the actual nonlinear system of Equation (1) does not leave $\Omega_{\rho_{min}}$ during one sampling period if the state of the RNN model of Equation (5) is bounded in $\Omega_{\rho_{nn}}$. If the state $x(t)$ enters $\Omega_{\rho_{min}} \backslash \Omega_{\rho_s}$, we have shown that Equation (35) holds, and thus, the state will be driven toward the origin again under $u = \Phi_{nn}(x)$ during the next sampling period. This completes the proof of Proposition 4 by showing that for any $x_0 = \hat{x}_0 \in \Omega_{\hat{\rho}}$, the closed-loop state trajectories of the nonlinear system of Equation (1) are maintained in $\Omega_{\hat{\rho}}$, and ultimately bounded in $\Omega_{\rho_{min}}$ provided that the assumptions of Proposition 4 are met.

4.3 | Lyapunov-based MPC using RNN models

The Lyapunov-based model predictive control design is given by the following optimization problem:

$$\mathcal{J} = \min_{u \in S(\Delta)} \int_{t_k}^{t_{k+N}} L(\tilde{x}(t), u(t)) dt \tag{36a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = F_{nn}(\tilde{x}(t), u(t)) \tag{36b}$$

$$u(t) \in U, \forall t \in [t_k, t_{k+N}) \tag{36c}$$

$$\tilde{x}(t_k) = x(t_k) \tag{36d}$$

$$\dot{\hat{V}}(x(t_k), u) \leq \dot{\hat{V}}(x(t_k), \Phi_{nn}(x(t_k))), \text{if } x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}} \tag{36e}$$

$$\hat{V}(\tilde{x}(t)) \leq \rho_{nn}, \forall t \in [t_k, t_{k+N}), \text{if } x(t_k) \in \Omega_{\rho_{nn}} \tag{36f}$$

where $\tilde{x}$ is the predicted state trajectory, $S(\Delta)$ is the set of piecewise constant functions with period Δ, and N is the number of sampling periods in the prediction horizon. $\dot{\hat{V}}(x,u)$ is used to represent $\frac{\partial \hat{V}(x)}{\partial x}(F_{nn}(x,u))$. The optimal input trajectory computed by the LMPC is denoted by $u^*(t)$, which is calculated over the entire prediction horizon $t \in [t_k, t_{k+N})$. The control action computed for the first sampling period of the prediction horizon $u^*(t_k)$ is sent by the LMPC to be applied over the first sampling period and the LMPC is resolved at the next sampling time.

In the optimization problem of Equation (36), the objective function of Equation 36a is the integral of $L(\tilde{x}(t),u(t))$ over the prediction horizon. The constraint of Equation (36b) is the RNN model of Equation (5) that is used to predict the states of the closed-loop system. Equation (36c) defines the input constraints applied over the entire prediction horizon. Equation (36d) defines the initial condition $\tilde{x}(t_k)$ of Equation (36b), which is the state measurement at $t = t_k$. The constraint of Equation (36e) forces the closed-loop state to move toward the origin if $x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}}$. However, if $x(t_k)$ enters $\Omega_{\rho_{nn}}$, the states predicted by the RNN model of Equation (36b) will be maintained in $\Omega_{\rho_{nn}}$ for the entire prediction horizon. A schematic of the stability region and a closed-loop state trajectory under LMPC is shown in Figure 4.

Based on the LMPC of Equation (36), the following theorem is established to demonstrate that the LMPC optimization problem can be solved with recursive feasibility, and closed-loop stability of the nonlinear system of Equation (1) is guaranteed under the sample-and-hold implementation of the optimal control actions calculated by LMPC.

Theorem 2 Consider the closed-loop system of Equation *(1)* under the LMPC of Equation (36) based on the controller $\Phi_{nn}(x)$ that satisfies Equation (16). Let $\Delta > 0$, $\varepsilon_s > 0$ and $\hat{\rho} > \rho_{\min} > \rho_{nn} > \rho_s$ satisfy Equation (26) and (27). Then, given any initial state $x_0 \in \Omega_{\hat{\rho}}$, if the conditions of Propositions *3* and *4* are satisfied, there always exists a feasible solution for the optimization problem of Equation (36). Additionally, it is guaranteed that under the LMPC of Equation (36), $x(t) \in \Omega_{\hat{\rho}}, \forall t \geq 0$, *and x(t) ultimately converges to* $\Omega_{\rho_{min}}$ for the closed-loop system of Equation *(1)*.

Proof We first prove that the optimization problem of Equation (36) is recursively feasible for all $x \in \Omega_{\hat{\rho}}$. Specifically, if $x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}}$ at $t = t_k$, the control action $u(t) = \Phi_{nn}(x(t_k)) \in U$, $t = [t_k, t_{k+1})$ calculated based on the state measurement $x(t_k)$ satisfies the input constraint of Equation (36c) and the Lyapunov-based constraint of Equation (36e). Additionally, if $x(t_k) \in \Omega_{\rho_{nn}}$, the control actions given by $\Phi_{nn}(x(t_{k+i}))$, $i = 0,1,\ldots, N-1$ satisfies the input constraint of Equation (36c) and the Lyapunov-based constraint of Equation (36f) as it is shown in Proposition 4 that the states predicted by the RNN model of Equation (36b) remain inside $\Omega_{\rho_{nn}}$ under the controller $\Phi_{nn}(x)$. Therefore, for all $x_0 \in \Omega_{\hat{\rho}}$, the LMPC optimization problem of Equation (36) can be solved with recursive feasibility if $x(t) \in \Omega_{\hat{\rho}}$ for all times.

Next, we prove that given any $x_0 \in \Omega_{\hat{\rho}}$, the state of the closed-loop system of Equation (1) is bounded in $\Omega_{\hat{\rho}}$ for all times and ultimately converges to a small neighborhood around the origin $\Omega_{\rho_{min}}$ defined by Equation (27b) under the LMPC of Equation (36). Consider $x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}}$ at $t = t_k$. The constraint of Equation (36e) is activated such that the control action u is calculated to decrease the value of $\hat{V}(\hat{x})$ based on the states predicted by the RNN model of Equation (36b) over the next sampling period. Additionally, it is shown in Equation (35) that if the constraint of Equation (36e) is satisfied, $\dot{\hat{V}}(x) \leq -\epsilon_w$ holds for $t \in [t_k, t_{k+1})$ after the control action $u^*(t_k)$ is applied to the nonlinear system of Equation (1). Therefore, the value of $\hat{V}(x)$ based on the state of the actual nonlinear system of Equation (1) decreases within the next sampling period, which implies that the closed-loop state can be driven into $\Omega_{\rho_{nn}}$ within finite sampling steps. After the state enters $\Omega_{\rho_{nn}}$, the constraint of Equation (36f) is activated to maintain the predicted states of the RNN model of Equation (36b) in $\Omega_{\rho_{nn}}$ over the entire prediction horizon. As there exists mismatch between the RNN system of Equation (36b) and the nonlinear system of Equation (1), the state of the nonlinear system of Equation (1) may leave $\Omega_{\rho_{nn}}$ under the constraint of Equation (36f). However, if we characterize a region $\Omega_{\rho_{min}}$ that satisfies Equation (27b), it is shown in Proposition 4 that the state $x(t)$ of the nonlinear system of Equation (1), $\forall t \in [t_k, t_{k+1})$ is guaranteed to be bounded in $\Omega_{\rho_{min}}$ if the predicted state by the RNN model of Equation (36b) remains in $\Omega_{\rho_{nn}}$. Therefore, at the next sampling step $t = t_{k+1}$, if the state $x(t_{k+1})$ is still bounded in $\Omega_{\rho_{nn}}$, the constraint of Equation (36f) maintains the predicted state $\hat{x}$ of the RNN model of Equation (36b) in $\Omega_{\rho_{nn}}$ such that the actual state x of the nonlinear system of Equation (1) stays inside $\Omega_{\rho_{min}}$. However, if $x(t_{k+1}) \in \Omega_{\rho_{min}} \backslash \Omega_{\rho_{nn}}$, following the proof we have shown for the case that $x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}}$, the constraint of Equation (36e) will be activated instead to drive it toward the origin. This completes the proof of boundedness of the states of the closed-loop system of Equation (1) in $\Omega_{\hat{\rho}}$ and convergence to $\Omega_{\rho_{min}}$ for any $x_0 \in \Omega_{\hat{\rho}}$.

Remark 8 Theorem 2 shows that closed-loop stability of the nonlinear system of Equation (1) is achieved under the LMPC of Equation (36) that is designed based on the RNN model of Equation (5) and RNN-based constraints. It is noted that the closed-loop state of the nonlinear system of Equation (1) can be driven to a small neighborhood around the origin because the constraints of the LMPC of Equation (36) guarantee the decrease of $\hat{V}$ in each sampling period accounting for the effect of model mismatch including the modeling error ν between the system of Equation (1) and the RNN model of Equation (5), the sample-and-hold implementation of control actions, and the bounded disturbances $w(t)$ in Equation (1). In other words, closed-loop stability can be maintained under the LMPC of Equation (36) if the modeling error ν, the sampling period Δ and the bound of disturbances w_m are sufficiently small such that Propositions 3 and 4 are satisfied.

Remark 9 The performance of the closed-loop system of Equation (1) under the LMPC of Equation (36) depends heavily on the accuracy of the RNN model. Specifically, as the closed-loop stability region $\Omega_{\hat{\rho}}$ is characterized under the RNN model of Equation (5), a well-fitting RNN model leads to a large operating region. Ideally, if the RNN model of Equation (5) fits the system of Equation (1) perfectly, the closed-loop stability region $\Omega_{\hat{\rho}}$ for the RNN model of Equation (5) remains the same as Ω_{ρ} for the actual nonlinear process of Equation (1). Additionally, the small neighborhood around the origin $\Omega_{\rho_{min}}$ that the state of the closed-loop system of Equation (1) ultimately converges to is also characterized accounting for the modeling error between the RNN model of Equation (5) and the nonlinear process of Equation (1). Therefore, the state of the nonlinear process of Equation (1) will be bounded in a small region closer to the origin provided that a reliable RNN model is obtained. Moreover, the speed of convergence of the LMPC optimization problem can be affected by the performance of the RNN model. Specifically, the optimization problem of LMPC of Equation (36) is solved by calculating the optimal solution $u^*(t)$ that satisfies the constraints of Equation (36c)–(36f) and minimizes the objective function of Equation (36a) simultaneously. However, in the case that the RNN model of Equation (5) is not accurate enough, it is possible that the LMPC optimization problem oscillates back and forth and becomes time-consuming due to the opposite gradient descent directions driven by the constraints and the objective function.

4.4 | LMPC using an ensemble of RNN models

As the RNN model accuracy plays an important role in the optimization problem of LMPC of Equation (36), ensemble regression models

introduced in previous section are employed to improve the performance of the closed-loop system of Equation (1) under LMPC. Based on the formulation of LMPC given by Equation (36), the LMPC that incorporates ensemble regression models is developed as follows:

$$\mathcal{J} = \min_{u\in S(\Delta)} \int_{t_k}^{t_{k+N}} L(\tilde{x}(t),u(t))dt \tag{37a}$$

$$\text{s.t.}\quad \dot{\tilde{x}}(t) = \frac{1}{N_e}\sum_{j=1}^{N_e} F_{nn}^{j}(\tilde{x}(t),u(t)) \tag{37b}$$

$$u(t) \in U, \forall t \in [t_k, t_{k+N}) \tag{37c}$$

$$\tilde{x}(t_k) = x(t_k) \tag{37d}$$

$$\dot{\hat{V}}(x(t_k),u) \le \dot{\hat{V}}(x(t_k),\Phi_{nn}(x(t_k))), \text{if } x(t_k) \in \Omega_{\hat{\rho}} \backslash \Omega_{\rho_{nn}} \tag{37e}$$

$$\hat{V}(\tilde{x}(t)) \le \rho_{nn}, \forall t \in [t_k, t_{k+N}), \text{if } x(t_k) \in \Omega_{\rho_{nn}} \tag{37f}$$

where the notation follows that in Equation (36) and N_e is the number of regression models used for prediction. N_e is the number of all available RNN models, and can be determined off-line through trial-and-error to account for computational efficiency. It is shown in Equation (37b) that the states $\tilde{x}(t)$, $t \in [t_k, t_{k+N})$ are now predicted by taking the average of RNN models F_{nn}^{j}, $j = 1,\ldots,N_e$. As the objective function of Equation (37a) and the Lyapunov-based constraints of Equation (37e-37f) are computed based on predicted states from Equation (37b), the application of ensemble regression models in Equation (37b) significantly improves the solution of the optimization problem and thus leads to a better closed-loop performance. Additionally, it is readily shown that closed-loop stability established in Theorem 2 is still guaranteed for the LMPC of Equation (37) because each regression model F_{nn}^{j}, $j = 1, \ldots, N_e$ is trained to satisfy all the conditions and assumptions in Theorem 2.

The LMPC based on ensemble regression models is implemented in the same way as the LMPC of Equation (36), that is, the optimal input trajectory $u^*(t)$ is calculated over the entire prediction horizon $t \in [t_k, t_{k+N})$ but only the first control action $u^*(t_k)$ is applied to the system of Equation (1) over the first sampling period. However, as the optimization problem of the LMPC of Equation (37) is now based on prediction results from multiple RNN models, the computation time for training multiple RNN models and solving the LMPC of Equation (37) both increase as the number of RNN models being used increases, which suggests the further investigation on computational efficiency for the real-time implementation of LMPC using an ensemble regression models.

5 | CONCLUSION

In this work, we developed a Lyapunov-based model predictive controller for nonlinear systems that uses an ensemble of recurrent neural network models to predict the process dynamics. Specifically, an RNN model was first trained using extensive open-loop simulation data to capture process dynamics in a certain operating region such that the modeling error between the recurrent neural network model and the actual nonlinear process model was sufficiently small. Then, the well-fitting RNN model was incorporated in the formulation of LMPC to predict process dynamics. The stability analysis of the closed-loop system under the LMPC using RNN models established the boundedness of closed-loop state in the stability region and ultimate convergence to a small neighborhood around the origin. Additionally, to further improve prediction accuracy of the RNN model, ensemble regression models were utilized in LMPC and parallel computing was introduced to reduce computation time.

ACKNOWLEDGMENTS

Financial support from the National Science Foundation and the Department of Energy is gratefully acknowledged.

ORCID

Panagiotis D. Christofides https://orcid.org/0000-0002-8772-4348

REFERENCES

1. Hopfield JJ. Neural networks and physical systems with emergent collective computational abilities. *Proc Natl Acad Sci*. 1982;79:2554-2558.
2. Shen Q, Jiang B, Shi P, Lim CC. Novel neural networks-based fault tolerant control scheme with fault alarm. *IEEE Trans Cybernetics*. 2014;44:2190-2201.
3. Wu Z, Albalawi F, Zhang J, Zhang Z, Durand H, Christofides PD. Detecting and handling cyber-attacks in model predictive control of chemical processes. *Mathematics*. 2018;6:173.
4. Wang Y. A new concept using LSTM neural networks for dynamic system identification. *Proceedings of the American Control Conference*. Seattle, Washington: IEEE; 2017:5324-5329.
5. Mhaskar P, El-Farra NH, Christofides PD. Stabilization of nonlinear systems with state and control constraints using Lyapunov-based predictive control. *Syst Cont Lett*. 2006;55:650-659.
6. Wu Z, Durand H, Christofides PD. Safe economic model predictive control of nonlinear systems. *Syst Cont Lett*. 2018;118:69-76.
7. Van Overschee P, De Moor B. N4SID: subspace algorithms for the identification of combined deterministic-stochastic systems. *Automatica*. 1994;30:75-93.
8. Viberg M. Subspace-based methods for the identification of linear time-invariant systems. *Automatica*. 1995;31:1835-1851.
9. Billings SA. *Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains*. John Wiley & Sons; 2013.
10. Kosmatopoulos EB, Polycarpou MM, Christodoulou MA, Ioannou PA. High-order neural network structures for identification of dynamical systems. *IEEE Trans Neural Netw*. 1995;6:422-431.
11. Trischler AP, D'Eleuterio GM. Synthesis of recurrent neural networks for dynamical system simulation. *Neural Netw*. 2016;80:67-78.
12. Alanqar A, Durand H, Christofides PD. On identification of well-conditioned nonlinear systems: application to economic model predictive control of nonlinear processes. *AIChE J*. 2015;61:3353-3373.
13. Ali JM, Hussain MA, Tade MO, Zhang J. Artificial intelligence techniques applied as estimator in chemical process systems–a literature survey. *Expert Syst Appl*. 2015;42:5915-5931.

14. Wong W, Chee E, Li J, Wang X. Recurrent neural network-based model predictive control for continuous pharmaceutical manufacturing. *Mathematics*. 2018;6:242.
15. Han H, Wu X, Qiao J. Real-time model predictive control using a self-organizing neural network. *IEEE Trans Neural Netw Learning Syst*. 2013;24:1425-1436.
16. Wang T, Gao H, Qiu J. A combined adaptive neural network and nonlinear model predictive control for multirate networked industrial process control. *IEEE Trans Neural Netw Learning Syst*. 2016;27:416-425.
17. Porfirio CR, Neto EA, Odloak D. Multi-model predictive control of an industrial C3/C4 splitter. *Cont Eng Practice*. 2003;11:765-779.
18. Rodrigues M, Theilliol D, Adam-Medina M, Sauter D. A fault detection and isolation scheme for industrial systems based on multiple operating models. *Cont Eng Practice*. 2008;16:225-239.
19. Mendes-Moreira J, Soares C, Jorge AM, Sousa JFD. Ensemble approaches for regression: a survey. *ACM Comp Surv*. 2012;45:10.
20. Bishop CM. Training with noise is equivalent to Tikhonov regularization. *Neural Comput*. 1995;7:108-116.
21. Zhang C, Ma Y. *Ensemble machine learning: methods and applications*. New York: Springer; 2012.
22. Wu Z, Tran A, Ren YM, Barnes CS, Chen S, Christofides PD. Model predictive control of phthalic anhydride synthesis in a fixed-bed catalytic reactor via machine learning modeling. *Chem Eng Res Des*. 2019;145:173-183.
23. Noor RM, Ahmad Z, Don MM, Uzir MH. Modelling and control of different types of polymerization processes using neural networks technique: a review. *Can J Chem Eng*. 2010;88:1065-1084.
24. Tian Y, Zhang J, Morris J. Modeling and optimal control of a batch polymerization reactor using a hybrid stacked recurrent neural network model. *Industr Eng Chem Res*. 2001;40:4525-4535.
25. Lin Y, Sontag ED. A universal formula for stabilization with bounded controls. *Syst Cont Lett*. 1991;16:393-397.
26. Sontag ED. Neural nets as systems models and controllers. *Proceedings of the Seventh Yale Workshop on Adaptive and Learning Systems*. Yale University; 1992:73-79.
27. Pearlmutter BA. Gradient calculations for dynamic recurrent neural networks: a survey. *IEEE Trans Neural Netw*. 1995;6:1212-1228.
28. Polycarpou MM, Ioannou PA. *Identification and control of nonlinear systems using neural network models: design and stability analysis*. Technical report. University of Southern California; 1991.
29. Narendra KS, Annaswamy AM. *Stable adaptive systems*. Englewood Cliffs, NJ: Prentice-Hall; 1989.
30. Billings SA, Fakhouri SY. Identification of non-linear systems using correlation analysis and pseudorandom inputs. *Int J Syst Sci*. 1980;11:261-279.
31. F. Chollet et al. Keras. https://keras.io, 2015.

How to cite this article: Wu Z, Tran A, Rincon D, Christofides PD. Machine learning-based predictive control of nonlinear processes. Part I: Theory. *AIChE J*. 2019;65:e16729. https://doi.org/10.1002/aic.16729