Contents lists available at ScienceDirect

Chemical Engineering Research and Design

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

Machine learning-based predictive control using noisy data: evaluating performance and robustness via a large-scale process simulator

Zhe Wu[a], *Junwei Luo*[a], *David Rincon*[a], *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*

A R T I C L E I N F O

Article history:
Received 16 January 2021
Received in revised form 7 February 2021
Accepted 9 February 2021
Available online 19 February 2021

Keywords:
Machine learning
Long short-term memory neural networks
Noisy data
Model predictive control
Nonlinear systems
Chemical processes

A B S T R A C T

Machine learning modeling of chemical processes using noisy data is a practically challenging task due to the occurrence of overfitting during learning. In this work, we propose a dropout method and a co-teaching learning algorithm that develop long short-term memory (LSTM) neural networks to capture the ground truth (i.e., underlying process dynamics) from noisy data. To evaluate the performance and robustness of the proposed modeling approaches, we consider an industrial chemical reactor example and use a large-scale process simulator, Aspen Plus Dynamics that does not employ assumptions on reactor properties typically made in the derivation of first-principles models, to generate process operational data that are corrupted by sensor noise which is determined using industrial data. The dropout method is first utilized to reduce the overfitting of LSTM models to noisy data. Then, another approach termed co-teaching method is used to train LSTM models with additional noise-free data generated from simulations of the reactor first-principles model that employs several standard modeling assumptions not made in the Aspen model. Through open-loop and closed-loop simulations, we demonstrate the improvement of model prediction accuracy and of the open- and closed-loop performances under model predictive controllers using dropout and co-teaching LSTM neural network models compared to the LSTM model developed from the standard training process from the noisy data.

1. Introduction

Machine learning has attracted an increasing level of attention in classical engineering fields in recent years due to its ability of analyzing big data from industrial processes. Machine learning techniques such as neural networks and their variants have been successfully applied in process modeling, process monitoring, and fault detection, which fall into the categories of regression and classification problems. Among many types of neural networks, recurrent neural networks (RNN), and long short-term memory (LSTM) networks have become popular for modeling nonlinear dynamic systems from time-series data, and have been incorporated in model predictive control (MPC) to predict evolution of process states when first-principles process models are unavailable. While many research works have studied neural network modeling of chemical processes using noise-free data, learning using noisy data is a practically challenging task due to the high capacity of neural network to fit noisy data (i.e., overfitting). Considering that the sensor measurements in chemical plants are commonly affected by noise in real-time operation, machine learning modeling of chemical processes

* *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.cherd.2021.02.011

using industrial noisy data remains an important research topic.

One way to handle noisy measurements in linear dynamic systems is Kalman filtering, e.g., Patwardhan et al. (2012). Additionally, many other methods such as moving horizon estimation and unscented Kalman filter have been proposed to deal with data noise (Patwardhan et al., 2012). In the state estimation methodology, to establish a correct estimation, a model representation is generally needed and the covariance matrices need to be tuned as well (Lima and Rawlings, 2011). Recently, the effect of learning with raw vibration signals from a laboratory-scale water flow system was studied using machine learning methods (i.e., LSTM and a feed-forward deep neural network) and a linear statistical learning approach (i.e., projection to latent structure, PLS) (Shah et al., 2020). From their findings, it was found a poor performance from both machine learning methods and PLS when using raw vibration data and that further treatment of the data is needed for better model development. When exposing machine learning models to Gaussian noise, Yeo (2019) has shown that machine learning models can efficiently learn the true process dynamics due to the dominant role of the internal states during the prediction step. However, as real-world data noise often follows a non-Gaussian distribution, machine learning models may undergo performance decay if no proper treatment is taken to handle data noise in realistic scenarios.

There are many works in the literature that study machine learning methods with different types of noise. For example, it has been pointed out in Li et al. (2020) that assuming independent identically distributed (i.i.d.) noise is not realistic when modeling some chemical processes with Gaussian process (GP) regression. For that reason, the i.i.d. condition is relaxed to heteroscedastic noise for a machine learning structure in Li et al. (2020). In Hsu and Wang (2009), a Wiener-type recurrent neural network is tested with two types of noise, a white noise and a sinusoidal-type noise in order to evaluate robustness. In Krishnaiah et al. (2006), the effect of Gaussian noise in RNN modeling of chaotic systems using short time-series data has been studied. Additionally, data preprocessing and smoothing techniques can be used to improve data quality. For example, noisy and redundant data are removed for an ensemble-based machine learning approach in a foaming control application in Agarwal et al. (2019). In order to perform data smoothing pretreatment and tackle missing data points, a third-order polynomial is implemented to the experimental data and later used with artificial neural networks to develop a deep reinforcement learning scheme that controls a bioreactor (Ma et al., 2020). In Kadlec et al. (2011), a linear filter is used to denoise the measurements in data-driven soft sensors when implemented with machine learning methods.

In terms of the applications of machine learning modeling approaches, the integration of machine learning models in model predictive controllers have attracted increasing attention in recent years. Machine learning modeling and control of nonlinear processes under noise-free conditions have been explored in Wu et al. (2019a, 2019b) and Hassanpour et al. (2020), where the control performance depends greatly on the model accuracy. To handle noisy data subject to industrial data noise following a non-Gaussian distribution in machine learning modeling of nonlinear processes, Monte Carlo dropout and co-teaching methods have been utilized in Wu et al. (2021) to develop LSTM models to capture the ground truth from noisy data. Specifically, the dropout method uses noisy data only and reduces the overfitting by randomly dropping out model weights at both training and testing phases. Co-teaching method uses noise-free data generated from simulations of first-principles process model in addition to noisy data to improve model performance by accounting for noise-free pattern. However, as first-principles models are imperfect representations of reality due to simplified governing equations and boundary conditions, model mismatch may arise when noise-free data is generated based on first-principles solutions.

To evaluate the performance of the proposed machine leaning modeling approaches in the presence of mismatch between the industrial chemical process and its first-principles model, in this work, we consider a chemical reactor example simulated in the process simulator, Aspen Plus Dynamics, that allows to evaluate model mismatch and controller robustness to industrially-generated data noise. Chemical process simulators are widely used in industrial process design to provide more accurate steady-state and dynamic solutions than first-principles models due to their ready-to-use libraries involving thermodynamics properties, unit operations, and many other features (Feng et al., 2019). In general, chemical process simulators can be classified into equation-oriented (e.g., EMSO software) and sequential modular approaches (e.g., Aspen Plus) (Mendoza et al., 2013). In addition to process design, the integration of process control systems in processes simulators has been explored in the literature Sharifzadeh, 2013.

In this work, we generate noisy data using Aspen dynamic simulations, in which the process state measurements are corrupted by industrial sensor noise. Subsequently, the reactor first-principles model is developed in Matlab to generate noise-free data. The dropout LSTM model is trained using noisy data only, and the co-teaching LSTM model is trained using both noisy and noise-free data. The LSTM models are then incorporated in a Lyapunov-based model predictive controller that optimizes process performance while maintaining closed-loop system stability. Finally, we compare the dropout and co-teaching LSTM models with the LSTM model trained using the standard learning algorithm and demonstrate their superiority in both open-loop and closed-loop operations. The rest of the paper is organized as follows: in Section 2, the class of nonlinear systems, long short term memory networks, and machine-learning-based model predictive control scheme are presented. In Section 3, an overview of the dropout and co-teaching methods are provided. In Section 4, an industrial chemical reactor example is used to demonstrate the efficacy of the proposed machine learning modeling and control approaches.

2. Preliminaries

2.1. *Notation*

The Euclidean norm of a vector is denoted by the operator $|\cdot|$ and the weighted Euclidean norm of a vector is denoted by the operator $|\cdot|_Q$ where Q is a positive definite matrix. x^T denotes the transpose of x. The notation $L_fV(x)$ denotes the standard Lie derivative $L_fV(x){:=}\frac{\partial V(x)}{\partial x}f(x)$. Set subtraction is denoted by "\", i.e., $A\setminus B := \{x \in \mathbf{R}^n | x \in A, x \notin B\}$.

2.2. *Class of systems*

We consider the class of continuous-time nonlinear systems described by the following system of first-order nonlinear ordinary differential equations:

$$\begin{aligned} \dot{x} &= F(x, u) := f(x) + g(x)u, \quad x(t_0) = x_0 \\ y &= x + w \end{aligned} \tag{1}$$

where $x \in \mathbf{R}^n$ is the state vector, $u \in \mathbf{R}^m$ is the manipulated input vector, $y \in \mathbf{R}^n$ is the vector of state measurements that are sampled continuously, and $w \in \mathbf{R}^n$ is the noise vector. The input vector is constrained by $u \in U := \{u_i^{\min} \le u_i \le u_i^{\max}, i = 1, \ldots, m\} \subset \mathbf{R}^m$. $f(\cdot)$ and $g(\cdot)$ are sufficiently smooth vector and matrix functions of dimensions $n \times 1$ and $n \times m$, respectively with $f(0)$ assumed to be zero such that the origin is a steady-state of the nominal (i.e., $w(t) \equiv 0$) system of Eq. (1) (i.e., $(x_s^*, u_s^*) = (0, 0)$, where x_s^* and u_s^* represent the steady-state state and input vectors, respectively). Throughout the manuscript, we assume that the full state measurements are continuously available at all times, and the initial time t_0 is taken to be zero ($t_0 = 0$).

2.3. *Long short term memory (LSTM) model*

Long short-term memory (LSTM) networks are a type of recurrent neural network (RNN) capable of modeling long-term dependencies in sequence prediction problems due to the design of three gates, i.e., the input gate, the forget gate, and the output gate, in the network structure. A schematic of LSTM network structure is shown in Fig. 1 (Chen et al., 2020). In this work, the LSTM model is developed to predict the states of Eq. (1) given the control actions and the past noisy state measurements. Specifically, given the input sequence $m(k)$, $k = 1, \ldots, T$, where T is the number of measured states of the sampled-data system of Eq. (1), the following equations are used to calculate the predicted output sequence $\hat{x}(k)$:

$$i(k) = \sigma(\omega_i^m m(k) + \omega_i^h h(k-1) + b_i) \tag{2a}$$

$$f(k) = \sigma(\omega_f^m m(k) + \omega_f^h h(k-1) + b_f) \tag{2b}$$

$$c(k) = i(k)\tanh(\omega_c^m m(k) + \omega_c^h h(k-1) + b_c) + f(k)c(k-1) \tag{2c}$$

$$o(k) = \sigma(\omega_o^m m(k) + \omega_o^h h(k-1) + b_o) \tag{2d}$$

$$h(k) = o(k)\tanh(c(k)) \tag{2e}$$

$$\hat{x}(k) = \omega_y h(k) + b_y \tag{2f}$$

where $m(k)$, $c(k)$, $h(k)$, $i(k)$, $f(k)$, and $o(k)$ are the input sequence, the cell state, the internal state, the outputs from the input gate, the forget gate, and the output gate, respectively. $\hat{x} \in \mathbf{R}^{n \times T}$ represent the LSTM network output sequences. The weight matrices for the LSTM input vector m, and the hidden state vector in the input gate are represented by ω_i^m and ω_i^h, respectively. Similarly, the weight matrices for the input vector m and hidden state vector h in calculating the cell state c, the forget gate f, and the output gate o are represented by ω_c^m, ω_c^h, ω_f^m, ω_f^h, ω_o^m, ω_o^h, respectively, with b_i, b_f, b_o, b_c representing the bias terms. Finally, the LSTM predicted state is calculated using Eq. (2f) where ω_y and b_y denote the weight matrix and bias vector for the output, respectively. Since the LSTM model uses control actions and past state measurements to predict future states, the input sequence $m \in \mathbf{R}^{(n+m)\times T}$ contains the manipulated inputs $u \in \mathbf{R}^m$ and the past measured states $x \in \mathbf{R}^n$ within a certain period of time (i.e., T). The LSTM model uses the sigmoid activation function $\sigma(\cdot)$ and the hyperbolic tangent function $\tanh(\cdot)$ as the nonlinear activation functions. Additionally, as LSTM networks are a type of recurrent neural networks, we can also present the LSTM model in the form of a continuous-time nonlinear system as follows:

$$\dot{\hat{x}} = F_{nn}(\hat{x}, u) := A\hat{x} + \Theta^T z \tag{3}$$

where $\hat{x} \in \mathbf{R}^n$ is the LSTM state vector, $u \in \mathbf{R}^m$ is the manipulated input vector, and $z = [z_1 \cdots z_{n+m+1}]^T = [\sigma(\hat{x}_1) \cdots \sigma(\hat{x}_n) \;\; u_1 \cdots u_m \;\; 1]^T \in \mathbf{R}^{n+m+1}$ is a vector of both the network states $\hat{x}$ and the inputs u. $\sigma(\cdot)$ represents nonlinear activation functions in each LSTM unit, and “1” represents the bias term. The diagonal matrix $A \in \mathbf{R}^{n\times n}$ and the matrix $\Theta \in \mathbf{R}^{(n+m+1)\times n}$ consist of the LSTM weights that will be optimized.

Training data can be generated from extensive open-loop simulations of the nonlinear system of Eq. (1) under various initial conditions and control actions. The system inputs u are applied in a sample-and-hold fashion, i.e., $u(t) = u(t_k)$, $\forall t \in [t_k, t_{k+1})$, where $t_{k+1} := t_k + \Delta$ and Δ is the sampling period, and the explicit Euler method is utilized with a sufficiently small integration time step $h_c < \Delta$ to integrate the continuous-time nonlinear system of Eq. (1) in simulations. Then, the LSTM model can be trained following the learning process as discussed in Wu et al. (2019a).

Remark 1. The LSTM model of Eq. (3) is built for the network with one hidden layer (i.e., the network is formed with three layers: input layer, one hidden layer, and output layer). However, the LSTM development is not restricted to one hidden layer, and can be extended to multiple hidden layers for better approximation performance. Additionally, it should be noted that the LSTM internal states do not necessarily represent the actual process states in the first-principles model of Eq. (1). In this study, as the LSTM model is developed to predict future states given the past state measurements and manipulated inputs, the LSTM outputs correspond to the process states in the nonlinear system of Eq. (1).

Remark 2. Since the sensor measurements are now corrupted by industrial noise, the evolution of state variables can no longer rely on the current state measurement. Therefore, to reduce the dependence on the current state measurement, the machine learning models in this work are developed accounting for past (noisy) state measurements over a number of sampling periods to provide better predictions in a noisy environment.

Remark 3. As discussed in Wu et al. (2021), the nonlinear activation functions such as tanh and *sigmoid* functions are often used between the LSTM hidden layers to introduce nonlinearities into the network. Specifically, *sigmoid* function is typically used for LSTM input/output/forget gates, and tanh function is often used for the internal state and cell state. The linear unit is only used between the LSTM output layer and the last hidden layer. Since the second derivative of tanh function decays slowly to zero, the vanishing gradient problem in the training of RNN and LSTM models could be tackled by using tanh

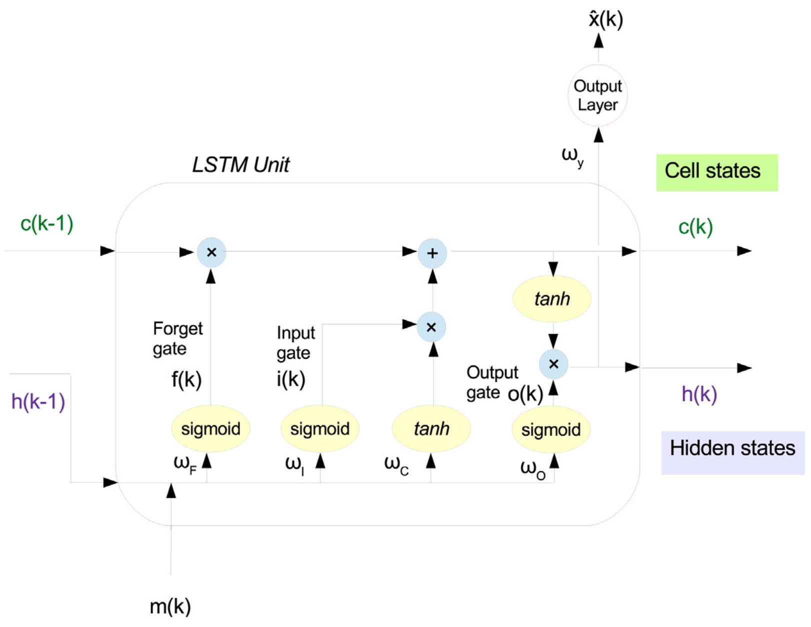

Fig. 1 – Schematic of LSTM units.

function. Although the saturating nonlinear activation function (i.e., tanh) can be limiting, it just means that the gradient near the boundaries will be small, hence taking smaller steps towards convergence; convergence is though guaranteed but at a slower pace. However, a non-saturating activation function like *ReLu* might cause the LSTM model to diverge in the training.

Remark 4. In addition to extensive open-loop simulations with different initial conditions and control actions, we can also generate machine learning datasets by simulating the system of Eq. (1) under a pseudo random input sequence to obtain a single continuous trajectory. Additionally, if the system is operated around an unstable steady-state, the holding period Δ for pseudo random input signals should be carefully chosen to make sure that the state is bounded in the stability region at all times. In Section 4, an Aspen Plus reactor example is used to demonstrate the data generation process using pseudo random input signals.

Remark 5. It should be mentioned that in this work, the LSTM model is trained using noisy data (i.e., the state measurements are corrupted by industrial data noise), which makes it challenging to obtain a well-conditioned LSTM model that can capture the ground truth (i.e., the underlying process dynamics of Eq. (1)) using standard learning algorithms. Therefore, to handle the noisy training data, in Section 3, we propose two methods to improve model performance. The dropout method utilizes noisy data only and improves model performance by reducing overfitting to noisy data. The co-teaching method improves model prediction accuracy by taking advantage of noise-free data generated from computer simulations.

Remark 6. We chose LSTM networks in this work because neural networks such as LSTM and recurrent neural networks (RNN) are often used to approximate nonlinear dynamical systems with time-series data due to the feedback loops in their structures that can exhibit temporal dynamic behavior. In general, any neural network structures face overfitting when dealing with noisy data. To address this issue, a simple neural network with a few number of layers and neurons can be used to reduce overfitting; however, a simple neural network model may be incapable of capturing the underlying process dynamics. It was demonstrated in Gal and Ghahramani (2016b) that by using dropout techniques, a large NN model with a sufficient number of neurons may lead to improved denoised results compared to simple NN models that were commonly used in the past to avoid overfitting.

2.4. *Model predictive control using LSTM models*

The LSTM model is incorporated in Lyapunov-based model predictive controller (LMPC) to provide state predictions in solving the MPC optimization problem. The formulation of LSTM-based MPC is presented as follows:

$$\mathcal{J} = \min_{u \in S(\Delta)} \int_{t_k}^{t_{k+N}} L(\tilde{x}(t), u(t))dt \tag{4a}$$

$$\text{s.t.} \quad \dot{\tilde{x}}(t) = F_{nn}(\tilde{x}(t), u(t)) \tag{4b}$$

$$u(t) \in U, \quad \forall \ t \in [t_k, t_{k+N}) \tag{4c}$$

$$\begin{aligned} &\dot{V}(x(t_k), u) \leq \dot{V}(x(t_k), \Phi_{nn}(x(t_k))), \\ &\text{if} \ \ x(t_k) \in \Omega_\rho \backslash \Omega_{\rho_{nn}} \end{aligned} \tag{4d}$$

$$V(\tilde{x}(t)) \leq \rho_{nn}, \quad \forall \ t \in [t_k, t_{k+N}), \quad \text{if} \ \ x(t_k) \in \Omega_{\rho_{nn}} \tag{4e}$$

where $\tilde{x}$, N and $S(\Delta)$ are the predicted state trajectory, the number of sampling periods in the prediction horizon, and the set of piecewise constant functions with period Δ. $\dot{V}(x, u)$ in Eq. (4d) denotes the time-derivative of V, i.e., $\frac{\partial V(x)}{\partial x}(F_{nn}(x, u))$. The LMPC is implemented in a receding horizon manner, where the first control action $u^*(t_k)$ in the optimal input sequence $u^*(t)$, $\forall \ t \in [t_k, t_{k+N})$ is applied to the system for the next sampling period. Specifically, the LMPC minimizes the time-integral of the cost function $L(\tilde{x}(t), u(t))$ that achieves its minimum value at the steady-state $(x_s^*, u_s^*) = (0, 0)$ accounting for the constraints of Eqs. (4b)–(4e). The control objective of LMPC is to maintain the closed-loop state in the stability region Ω_ρ for all times, and ultimately bound the predicted state in the target region $\Omega_{\rho_{nn}}$, which is a small level set of V around the origin. $\Phi_{nn}(x)$ in Eq. (4d) is a pre-determined control law that renders the origin of the LSTM system of Eq. (3) exponentially stable. When a well-conditioned LSTM model with a sufficiently high model accuracy can be obtained using noise-free training data, the LMPC of Eq. (4) guarantees closed-loop stability of the nonlinear system of Eq. (1). Theoretical results on closed-loop stability can be found in (Wu et al., 2019a).

Remark 7. Due to the sample-and-hold implementation of control actions, the closed-loop system of Eq. (1) cannot be stabilized exactly at the steady-state. In the formulation of MPC of Eq. (4), we require the predicted states to be ultimately bounded in a small level set of Lyapunov function (i.e., $\Omega_{\rho_{nn}}$ in Eq. (4e)) such that the true state will also be bounded in a small neighborhood around the origin. If for any initial state within the stability region Ω_ρ, the closed-loop state remains bounded in Ω_ρ for all times and ultimately converges to a small neighborhood around the origin where it will be maintained thereafter, then we say the system is practically stable under the LSTM-based MPC.

3. Dropout and co-teaching methods

It was demonstrated in Wu et al. (2021) that LSTM models are able to capture the underlying process dynamics using noisy data that is corrupted by Gaussian noise; however, a degraded model performance was noticed when training LSTM models with non-Gaussian noise. To handle industrial noise that follows a non-Gaussian distribution, Monte Carlo dropout method and co-teaching method were utilized in Wu et al. (2021) to learn the ground truth from noisy data. In this section, we present an overview of these two approaches and will evaluate their performance and robustness to industrial data noise using a chemical reactor example in Section 4.

3.1. *Dropout method*

Monte Carlo dropout (MC dropout) method treats neural network weights as random variables and drops out randomly selected weights at both training and testing stages (Gal and Ghahramani, 2016a, 2016b). As the neural network models using dropout method are essentially probabilistic models, an estimate of uncertainty for the model predictions can be obtained by performing Monte Carlo simulations.

Consider the LSTM model in the general form of Eq. (3). Let $\mathbf{W} = \{W_i\}_{i=1}^{L}$ represent the set of weight matrices of all LSTM layers that include both the weights and the bias terms, where $W_i : \mathbf{R}^{K_i \times K_{i-1}}$ is the weight matrix for the ith LSTM layer, and L is the number of layers. The goal of MC dropout method is to obtain the posterior distribution of LSTM weights W, i.e., $p(\mathbf{W})$, based on the training data $(\mathbf{M}, \mathbf{X})$, where $\mathbf{M}$ and $\mathbf{X}$ denote input and output data matrices. Specifically, as discussed in (Gal and Ghahramani, 2016a), the weight matrix W_i is defined as follows:

$$W_i = B_i \cdot \mathrm{diag}(z_i) \tag{5}$$

where z_i, $i = 1, \ldots, L$ are used to represent the weights being dropped out with a certain probability and are a set of binary variables satisfying Bernoulli distribution. B_i are the variational variables to be optimized. After the LSTM model is trained using the MC dropout method, the predictive distribution of LSTM output can be approximated by performing Monte Carlo dropout at test time. The predictive distribution of LSTM model is obtained via multiple realizations as follows:

$$p(\mathbf{x}^*|\mathbf{m}^*, \mathbf{X}, \mathbf{M}) \approx \frac{1}{N_t}\sum_{k=1}^{N_t} p(\mathbf{x}^*|\mathbf{m}^*, \mathbf{W}_k) \tag{6}$$

where $\mathbf{m}^*$ and $\mathbf{x}^*$ are the LSTM input, and the corresponding output in the testing dataset, and $\mathbf{X}$, $\mathbf{M}$ are the LSTM inputs and outputs in the training dataset. N_t is the number of Monte Carlo realizations at the testing phase. Since the LSTM model obtained using the MC dropout method is a probabilistic model, the LSTM output is no longer deterministic given the same input, and needs to be approximated using Eq. (6). Therefore, a probabilistic distribution is obtained from Eq. (6) as an estimate of uncertainty for the model predictions. Additionally, the ground-truth process dynamics can then be approximated using the sample mean of all predictions when the training data is corrupted by noise.

Remark 8. Note that the true process of Eq. (1) evolves deterministically, but our understanding of the process dynamics through machine learning models using dropout method is represented in a probabilistic manner because the training data is noisy. In fact, if we consider the additive sensor noise as an uncertain variable in the process model of Eq. (1), then the Monte Carlo dropout method provides an efficient way to model uncertain process dynamics (i.e., we can consider the LSTM model using the MC dropout method as an uncertain process model with the dropped LSTM weights being uncertain variables) and to predict underlying (nominal) process dynamics through a number of stochastic forward passes.

Remark 9. Since forward prediction is one of the most time-consuming parts in solving MPC regardless of using machine learning models or nonlinear first-principles models (Xie et al., 2015), parallel computing can be utilized to accelerate the computation of multiple forward passes in the Monte Carlo dropout method. Specifically, in parallel computing, each parallel computer node runs one forward prediction only and the host computer node averages the results and sends it to MPC. In Wu et al. (2021), it was demonstrated that the computation time of running multiple forward predictions in parallel is significantly reduced compared to the calculation of the same number of forward predictions in serial mode. Additionally, it was demonstrated in Wu et al. (2019b) that through the implementation of parallel computing for forward predictions, the computation efficiency of the MPC optimization problem was also improved.

Remark 10. Since dropout LSTM model is essentially a probabilistic model, in order to improve closed-loop performance, a stochastic MPC scheme using the dropout LSTM model can be developed to optimize control actions and maintain probabilistic closed-loop stability accounting for the probability distribution of noisy output. The interested readers may refer to Weissel et al. (2009), Mahmood and Mhaskar (2012) and Wu et al. (2018) for the design of model predictive controller for stochastic nonlinear systems.

Remark 11. Dropout techniques are typically used for fully-connected large-scale neural networks as compared to small networks in order for the networks to be able to learn the underlying pattern of data while reducing the overfitting.

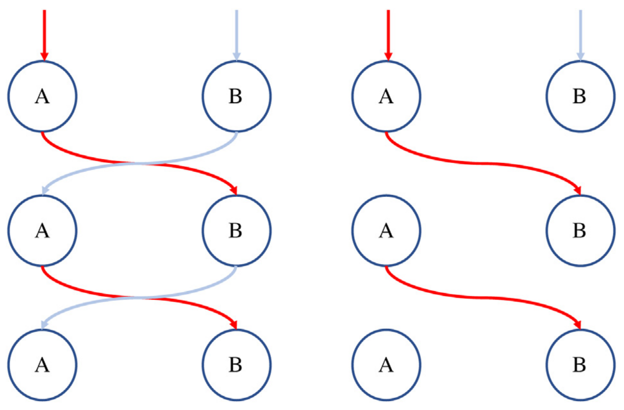

Fig. 2 – The symmetric (left) and asymmetric (right) co-teaching frameworks that train the two networks (A and B) simultaneously.

The dropout rate (i.e., the probability of a neuron being dropped out) plays a key role in the model performance, and is determined during model training to achieve desired training and validation results. Specifically, a small dropout value (e.g., 0.2–0.5) is recommended to start with. Then, it can be increased if overfitting still occurs, or can be decreased if the network fails to learn the process dynamics.

3.2. *Co-teaching method*

The co-teaching method was originally proposed to improve model accuracy in image classification problems, for which the dataset is corrupted by noise (Han et al., 2018). Specifically, data noise in classification problems could cause mislabeled data (for example, an object "A" is mislabeled as object "B"), while in regression problems, data noise could cause a deviation from its ground truth value. In either case, it is challenging for machine learning model to achieve a desired model accuracy with a noisy dataset following the standard learning algorithm. Therefore, the co-teaching method provides an alternative way to train machine learning models under noisy labels by taking advantage of noise-free data and training two models at the same time (Han et al., 2018; Yang et al., 2020). The intuition of co-teaching stems from the observations that neural networks will use a simple pattern to fit training data at the early stage of training process (Han et al., 2018). As a result, when assessing the loss function value under a simple pattern that approximates the relationship between neural network inputs and outputs, the noisy data generally correspond to a larger loss function value, while noise-free data correspond to a smaller value.

Algorithm 1. Co-teaching algorithm

```
for $i = 0$ to $I_{max}$ do
    Select a mini-batch $D_m$ from $D$
    Obtain the small-loss data sequences from model A: $D_A = \{x \in D_m \mid loss(A, x) \leq loss_T\}$
    Obtain the small-loss data sequences from model B: $D_B = \{x \in D_m \mid loss(B, x) \leq loss_T\}$
    Update the weight matrix of model A: $\mathbf{W_A} = \mathbf{W_A} - \eta \nabla loss(A, D_B)$
    Update the weight matrix of model B: $\mathbf{W_B} = \mathbf{W_B} - \eta \nabla loss(B, D_A)$
end
```

Fig. 2 shows two types of co-teaching structures (i.e., symmetric and asymmetric frameworks) that train two networks: A and B, simultaneously. The symmetric co-teaching training method is implemented following Algorithm 1, which is stated as follows: (1) at each training epoch, a mini-batch D_m is selected from the original mixed dataset D. Then, each model checks its data sequences (i.e., each pair of data labeled as input and output), and generates a small dataset (i.e., D_A and D_B) with all the data that has a low loss function value (e.g., $loss(A, x) \leq loss_T$), where $loss_T$ is the threshold for identifying small-loss data sequences; (2) this new small dataset is then sent to the peer network, and the neural network weights $\mathbf{W}_A$, $\mathbf{W}_B$ are updated with a learning rate η; (3) finally, the training is resumed, and the above process is repeated until the end of the training epochs I_{max}. The asymmetric co-teaching method is implemented in a similar way to train two models simultaneously. However, under asymmetric co-teaching framework, noise-free data is used by model A only, and noisy data is used by model B only. At each training epoch, model A injects a subset of noise-free data sequences into model B. Note that the information flows in one direction in the asymmetric co-teaching framework, i.e., from model A to model B only.

Additionally, when using co-teaching method to solve the regression problem of LSTM modeling, the neural network structure needs to be carefully chosen. For example, the number of units in each network plays a role in learning the underlying process dynamics from a mixed dataset of both noisy and noise-free data. If a deep neural network with a large number of layers and neurons is used, the neural network may well fit the noisy data at an early stage (i.e., over-fitting) before effectively learning the ground truth from noise-free data. Additionally, the mixed dataset should be constructed with an appropriate ratio of noise-free data to noisy data. If noise-free data is insufficient, the neural networks may not be able to learn the ground truth, and may overfit the noisy data as training evolves.

In the following section, we use a chemical process example simulated in Aspen Plus Dynamics to illustrate the application of dropout and co-teaching LSTM modeling approaches and evaluate their performance and robustness. Specifically, we will discuss the following steps in this case study: (1) data collection using Aspen simulation and first-principles model simulation, (2) LSTM training process, and (3) development of LSTM-based MPC that drives reactor temperature to its desired set-point. Through open-loop and closed-loop simulations, we demonstrate that the proposed LSTM model using dropout and co-teaching methods outperform the standard LSTM model in terms of more accurate predictions and better closed-loop performance.

Remark 12. As both the dropout and co-teaching methods are developed for a general class of nonlinear systems, they are generally effective in improving the model accuracy that can be achieved under the standard machine learning training process. However, it was observed in Wu et al. (2021) that the improvements of model accuracy under dropout/co-teaching methods vary depending on the noise levels. Therefore, how much improvement the dropout and co-teaching LSTM mod-

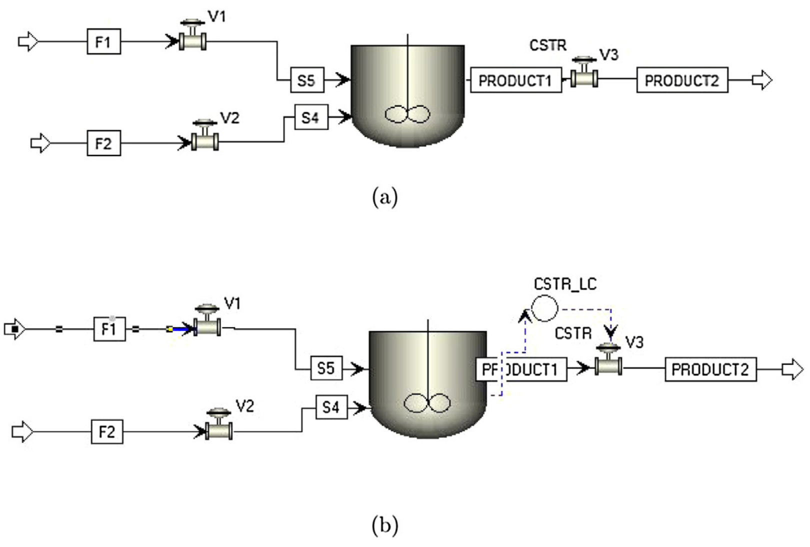

Fig. 3 – (a) Aspen flow sheet of chemical reactor example (steady-state set-up), and (b) Aspen flow sheet of chemical reactor example (dynamic model set-up).

eling approaches can achieve should be evaluated based on both the open-loop tests (i.e., prediction using testing dataset) and the closed-loop simulation under MPC.

Remark 13. Preprocessing of data, optimization of network structure in terms of the number of layers and neurons, and selection of loss functions, learning rate, and optimizers are common techniques for optimizing neural network performance. Beyond these standard tuning methods, the co-teaching method proposed in this work provides a guideline to improve training performance when both noisy and noise-free data are available. It will be shown in Section 4 that using the same mixed dataset with both noise-free and noisy data, the LSTM model using co-teaching method achieves better model accuracy than the LSTM following the standard training process.

Remark 14. The dropout and co-teaching methods improve the training performance in terms of less overfitting because: (1) the dropout method randomly drops neurons at input/hidden/output layer(s) such that more neurons at each layer are forced to learn the multiple characteristics of the neural network, and (2) the co-teaching method takes advantage of noise-free datasets that better characterize the underlying process dynamics to achieve a balanced performance between the noisy and clean pattern.

Remark 15. In practice, dropout has been widely used as an efficient regularization technique in many types of neural networks to reduce overfitting. However, dropout may fail in convolutional neural networks (CNN) because due to the spatial relationships encoded in CNN feature maps, activations could be highly correlated, which renders dropout ineffective. To address this issue, a Bayesian CNN architecture was proposed in Gal and Ghahramani (2015) to improve robustness to overfitting on small data by placing a probability distribution over the CNN's kernels. On the other hand, the co-teaching method is an efficient method to improve model performance provided that noise-free data is available. Since noise-free data of industrial processes is generally unavailable, we can use first-principles models to generate noise-free data from simulations to broaden its application in many process modeling problems in industry. Therefore, the performance of co-teaching method depends heavily on the noise-free data quality.

4. Application to an Aspen Plus Reactor Example

We consider an irreversible, second-order, exothermic reaction using Ethylene (A) and Benzene (B) to produce Ethyl benzene (EB) in a well-mixed, non-isothermal continuous stirred tank reactors (CSTR) (Kamal and Malah, 2017). The CSTR reactor is fed with two Hexane solutions in the feeding flow F_1 and F_2. The two flows have the same inlet temperature T_0, but different volumetric flowrate $F_{vj,in}$, $j = 1, 2$, where $j = 1, 2$ denotes the feeding flow F_1 and F_2. The reactants A and B are contained in each feeding flow separately with inlet molar concentration C_{A0} and C_{B0}. The reactions taking place in the CSTR are:

$$C_2H_4 + C_6H_6 \rightarrow C_8H_{10} \quad (\textit{ethylbenzene}) \tag{7a}$$

$$C_2H_4 + C_8H_{10} \rightarrow C_{10}H_{14} \quad (\textit{di-ethylbenzene}) \tag{7b}$$

$$C_6H_6 + C_{10}H_{14} \rightarrow 2C_8H_{10} \tag{7c}$$

In this study, the reactor model is developed in Aspen Plus and Aspen Plus Dynamics V11. The model is constructed and the steady-state simulations are first performed in Aspen Plus. Then, a dynamic simulation of the reactor process is carried out in Aspen Plus Dynamics to analyze its real-time performance. In Aspen Plus, a main flow sheet is designed with three valves and one CSTR as shown in Fig. 3a.

The valves play a role as a connector of fluid flow and parts by defining the pressure drop in the specific location, which is critical for generating a proper dynamic model. Without reasonable pressure drop in the process provided, Aspen Plus Dynamics cannot identify the source making the fluid flow through the system and may result in failure of dynamic simulation. In this model, the pressure drop at V_1 and V_2 are both 5 bar, and the pressure drop at V_3 is 2 bar.

Hexane is chosen as the solvent in the feeding flow F_1 and F_2 to ensure that the flow is in the liquid phase under the inlet

Table 1 – Parameter values of Aspen model.

$T_0 = 350.0\,K$	$T_s = 322.2\,K$
$F_{v1,in} = 50.0\,m^3/h$	$F_{v2,in} = 23.6\,m^3/h$
$C_{As} = 1.5454\,kmol/m^3$	$C_{Bs} = 4.2714\,kmol/m^3$
$C_{A0} = 4\,kmol/m^3$	$C_{B0} = 5\,kmol/m^3$
$Q_s = -695.1\,kJ/s$	$C_p = 2.41\,kJ/kg\,K$
$V = 60\,m^3/s$	$\rho_L = 683.7\,kg/m^3$
Heat transfer option	*Dynamic*
Medium temperature	298 K
Temperature approach	77.33 K
Heat capacity of coolant	4200 J/kg K
Medium holdup	1000 kg

temperature. Therefore, with a constant inlet volumetric flow rate, the amount of feeding reactants can be manipulated by adjusting the feeding concentration. Process parameter values used in the Aspen model are listed in Table 1, where C_A, C_B, ρ_L, V, and T are the concentration of ethylene, the concentration of benzene, mass density, volume and temperature of the reacting liquid in the CSTR, respectively. C_p is the mass heat capacity of the liquid mixture and is assumed to be constant. C_{As} and C_{Bs} are the steady-state concentration of reactants A and B, and C_{A0}, C_{B0} are the inlet concentration of A and B.

A liquid-only CSTR equipped with a heating/cooling jacket that supplies/removes heat at a rate Q, is considered to carry out three reactions. The initial temperature and pressure of the CSTR are set to be 400 K and 15 bar, which can be automatically adjusted by the steady-state simulation in Aspen. After incorporating the reactions of Eq. (7) in the CSTR, steady-state simulation is performed for analysis of plant behavior.

Before exporting the steady-state model to Aspen Plus Dynamics, the reactor geometry and thermodynamic parameters are defined in the dynamic mode of Aspen. The vessel type, head type, and length of the CSTR in this study are vertical, flat, and 10 m, respectively. The thermodynamic parameter values are listed in Table 1. To ensure that the dynamic mode is set up properly, we run the steady-state simulation again and complete the pressure check with the built-in Aspen Plus Pressure Checker with no error. Subsequently, the steady-state model is exported to Aspen Plus Dynamics with pressure driven chosen as the driven type of dynamic simulation.

4.1. *Dynamic model in Aspen Plus Dynamics*

The flow sheet of the Aspen dynamic model is shown in Fig. 3b. Specifically, a direct acting level controller, where direct means the output signal increases as the input signal increases, is added to the dynamic model to regulate the liquid level at 50% capacity in this study. Note that this controller can be designed following the default setting before exporting the steady-state model or can be manually developed in Aspen Plus Dynamics.

After configuration of the controller, we run a steady-state simulation in Aspen Plus Dynamics to obtain the steady-state for the dynamic model. The steady-state value of Q is −695097.0 W. Then, the heating type of CSTR is changed to constant duty to allow the outside controllers to manipulate Q during the dynamic simulation. Similarly, the volumetric flow rates F_1 and F_2 are fixed, and a steady-state simulation is performed to ensure that the dynamic model reaches the steady-state before collecting data.

Since the Aspen dynamic model can be considered to be a high-fidelity process model for the CSTR, we use Aspen dynamic simulation to generate datasets for neural network

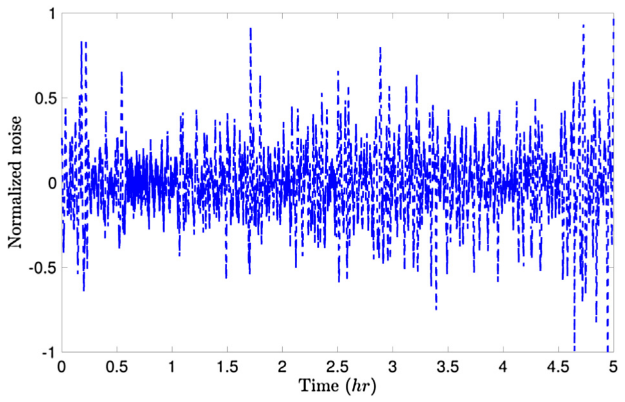

Fig. 4 – Normalized industrial noise from Aspen public domain data.

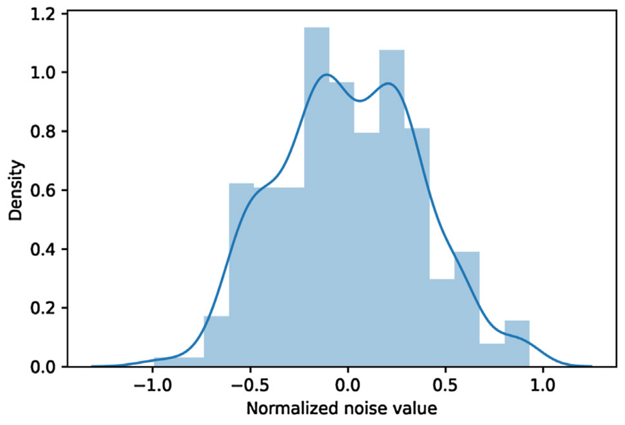

Fig. 5 – Probability density plot of normalized industrial noise in Fig. 4.

training. Industrial noise is introduced on state measurements to represent common sensor variability in chemical plants. Fig. 4 shows the normalized data noise obtained from Aspen public domain data. Fig. 5 shows the probability distribution of the normalized industrial noise in Fig. 4, from which we verify that the noise follows a non-Gaussian distribution.

Open-loop simulation is carried out in Aspen Plus Dynamics using the pseudorandom input signals generated in Matlab. A local Message Passing Interface (MPI) is constructed to link Aspen with Matlab so that the Aspen dynamic model can automatically read the input signals from Matlab and apply them in the dynamic simulations. In the open-loop simulation, the manipulated input variable Q varies within the range of $[-1.0 \times 10^6\,W, -4.0 \times 10^5\,W]$, and is implemented in a sample-and-hold fashion with the value updated every five minutes of the simulation time. We run the open-loop simulation for 15,000 min (simulation time) under pseudorandom input signals of Q with the industrial noise of Fig. 4 added on the temperature measurements. Specifically, since the industrial noise in Fig. 4 is the normalized result from Aspen public domain data, it is amplified by five times and then added on the temperature measurements. All the state measurements (e.g., C_A, C_B, and T) and input value of Q are continuously recorded to build the dataset for neural network training.

4.2. *First-principles model*

While Aspen Plus provides an efficient way that allows engineers to simulate, troubleshoot and optimize process performance and profitability with its high-fidelity process

Table 2 – Parameter values of the first-principles model of CSTR.

$k_1 = 1.528 \times 10^6\ m^3/kmol\,s$	$\Delta H_1 = -1.04 \times 10^5$ kJ/kmol
$k_2 = 2.778 \times 10^5\ m^3/kmol\,s$	$\Delta H_2 = -1.02 \times 10^5$ kJ/kmol
$k_3 = 0.4167\ m^3/kmol\,s$	$\Delta H_3 = -5.50 \times 10^2$ kJ/kmol
$E_1 = 71160$ kJ/kmol	$R = 8.314$ kJ/kmol K
$E_2 = 83680$ kJ/kmol	$E_3 = 62760$ kJ/kmol
$V = 60\ m^3/s$	$\rho_L = 683.7\ kg/m^3$
$C_p = 2.41$ kJ/kg K	

model, Aspen Plus models are typically not used in controller design due to their high computational cost. To reduce the computational time of solving the process model, first-principles models employing simplifying assumptions can be adopted in the design of model-based controllers. Additionally, extensive computer simulations using first-principles model is one of the most efficient data generation methods in machine learning.

In this study, we take advantage of the first-principles model of CSTR to generate noise-free datasets for LSTM training using the co-teaching method. Although the first-principles model may not fully capture the Aspen model dynamics under the same operating conditions, we will demonstrate that the co-teaching method using noisy data from the Aspen model, and noise-free data from the first-principles model solutions is still able to improve prediction accuracy of the LSTM model. While in practice noisy data is provided by chemical plants, and noise-free data is unavailable, the implementation of co-teaching method in this case study implies that the co-teaching LSTM modeling approach can improve prediction accuracy by using noise-free data generated from first-principles models, which broadens its application in many process modeling problems in industry. By applying mass and energy balances, the dynamic model of CSTR is described by the following nonlinear ODEs:

$$\frac{dC_A}{dt} = \frac{F_{v1,in}}{V}(C_{A0} - C_A) - r_1 - r_2 \tag{8a}$$

$$\frac{dC_B}{dt} = \frac{F_{v2,in}}{V}(C_{B0} - C_B) - r_1 - r_3 \tag{8b}$$

$$\frac{dT}{dt} = \frac{F_{v1,in} + F_{v2,in}}{V}(T_0 - T) + \frac{-\Delta H_1}{\rho_L C_p} r_1 + \frac{-\Delta H_2}{\rho_L C_p} r_2 + \frac{-\Delta H_3}{\rho_L C_p} r_3 + \frac{Q}{\rho_L C_p V} \tag{8c}$$

$$r_1 = k_1 e^{\frac{E_1}{RT}} C_A C_B \tag{8d}$$

$$r_2 = k_2 e^{\frac{E_2}{RT}} C_A C_{EB} \tag{8e}$$

$$r_3 = k_3 e^{\frac{E_3}{RT}} C_B C_{DEB} \tag{8f}$$

where r_j, $j = 1, 2, 3$ denote the rate of each reaction in Eq. (7) based on the rate law equation, and C_{EB}, C_{DEB} represent the concentration of C_8H_{10}, and of $C_{10}H_{14}$, respectively. The kinetic parameters for reactions are given in Table 2, where R, k_j, ΔH_j, and E_j, $j = 1, 2, 3$ represent ideal gas constant, pre-exponential constant, enthalpy of reaction, and activation energy of each reaction, respectively.

The manipulated input is the heat input rate Q represented in deviation variable form, i.e., $u^T = [Q - Q_s]$. Similarly, the process states are represented by $x^T = [C_A - C_{As}\ \ C_B - C_{Bs}\ \ T - T_s]$ where C_{As}, C_{Bs}, T_s are the steady-state values of C_A, C_B and T. By representing all the variables in deviation forms, the equilibrium of Eq. (8) is at the origin of state-space. The same pseudorandom signals of Q applied in Aspen simulations are applied to the open-loop simulation of the first-principles model of Eq. (8), where the explicit Euler method is used to integrate the nonlinear ODEs with a sufficiently small integration time step $h_c = 0.05$ min. The input signals are applied in a sample-and-hold fashion with the sampling period $\Delta = 5$ min. In open-loop simulations, process variables are measured every integration time step.

Fig. 6 compares the open-loop state profiles from Aspen simulation and first-principles model solutions under the same input sequences. Although the state profiles are close to each other, small deviations in the evolution of states can be noticed between the two models, which implies the existence of a mismatch between the Aspen model and the first-principles model of Eq. (8). The mismatch is caused by the difference between the balance-based first-principles equations and the equations adopted by Aspen Plus during the simulation. Variation of parameters, such as liquid density, heat capacity, and the energy exchange coefficient, contribute to the model mismatch. Note that these parameters are assumed to be constant in the first-principles process model. With noisy data from Aspen simulation and noise-free data from first-principles solutions, the simulation study in this section provides an insight on the applicability of the co-teaching method in handling real industrial noisy data, for which the corresponding noise-free data is generally unavailable, but can be obtained from computer simulations using first-principles or empirical models.

Remark 16. It should be clarified that in reality, the noisy data is from chemical plants due to common plant variance and sensors variability, and the noise-free data is generated from computational simulations based on first-principles process models. Therefore, when an industrial noisy dataset is available, computer simulations of the first-principles process model will only be used to generate noise-free datasets for the implementation of co-teaching method.

Remark 17. To further reduce the model mismatch, an accurate source of parameters can be utilized for the first-principles model, and a real-time communication between the Aspen database and the first-principles model can be established.

4.3. *Dropout and co-teaching LSTM models*

To reduce the impact of measurement noise in predicting future states, the LSTM models in this work rely on the state measurements over a past period of time to make predictions. The LSTM model is developed with C_A, C_B, T, and Q as inputs to predict the temperature T in the future time. Specifically, when using LSTM models in MPC to predict future states, the LSTM input vector at the current time step $t = t_k$ consists of the state measurements of $C_A(t)$, $C_B(t)$ and $T(t)$ over past five sampling periods, i.e., $\forall t \in [t_{k-5}, t_k]$ and the heat input rate $Q(t)$, $\forall t \in [t_{k-4}, t_{k+1}]$ implemented in a sample-and-hold fashion. Note that the heat input rate within the last sampling period, i.e., $\forall t \in [t_k, t_{k+1}]$ is unknown at the current time step t_k as it is the variable that will be optimized by MPC to meet

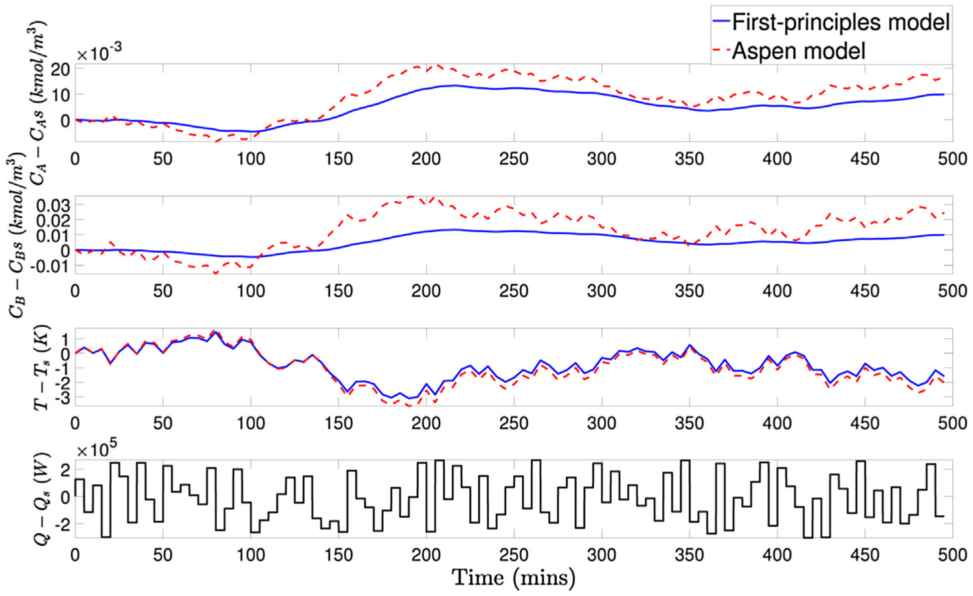

Fig. 6 – State profiles ($C_A - C_{As}$, $C_B - C_{Bs}$, $T - T_s$) from open-loop simulations of Aspen model and of first-principle model, respectively, under the same input sequences of Q.

the control objective. The LSTM output vector at the current time step $t = t_k$ is the predicted temperature $T(t)$ over $t \in [t_{k-4}, t_{k+1}]$. Since the temperature measurements before the current time step are known, only the prediction of $T(t)$ in the last sampling period, i.e., $\forall t \in [t_k, t_{k+1}]$ will be used in MPC to solve the optimization problem.

After running Aspen dynamic simulations and open-loop simulations of the first-principles model of Eq. (8), we obtain a dataset with LSTM inputs and outputs and reshape it to the following tensor dimensions: [2467,500,4] for inputs and [2467,500,1] for outputs, where the first element represents the total number of data sequences, the second element represents the length of each data sequence (i.e., 500 data points correspond to five sampling periods 25 min, in which the data point is collected every integration time step $h_c = 0.05$ min), and the last element represents the dimension of inputs and outputs, respectively (i.e., the LSTM has four inputs: C_A, C_B, T, and Q, and one output: T). Among 2467 data sequences, 494 sets of data are used for validation, and 100 sets of data are used for testing. In the case of the co-teaching method, noiseless datasets are obtained from first-principles solutions under the same operating conditions as performed in Aspen simulations. The high-level application programming interface, Keras, is used to develop the standard, dropout, and co-teaching LSTM networks under the optimization algorithm *Adam*.

4.4. *Open-loop simulation results*

We first carry out an open-loop simulation study to demonstrate the improvement of LSTM model accuracy using dropout and co-teaching methods. Table 3 shows the mean squared errors (MSE) of reactor temperature predicted by LSTM models with respect to the ground truth (i.e., actual temperature value of the nominal system) from testing dataset. The results for standard LSTMs under different datasets are shown in item 1, and those for co-teaching and dropout LSTMs are shown in items 2 and 3, respectively.

Note that all the LSTM models in Table 3 are developed with the same structure in terms of the same number of neurons, layers, epochs, and the type of activation functions and of optimization algorithms. We first consider three types of datasets for standard LSTM models, and the results are reported in item 1 of Table 3. Specifically, in item 1a, the LSTM model is trained following the standard training process with noisy data only (i.e., noisy data from Aspen simulations in Section 4.1); in item 1b, the LSTM model is trained using a mixed dataset consisting of both noisy data from Aspen simulations and noise-free data from simulations of the first-principles model in Section 4.2 ("fp" in Table 3 represents the first-principles model); in item 1c, we consider a scenario where noise-free data is also available from Aspen simulations, thereby the LSTM model is trained using both noisy and noise-free data from Aspen simulations. However, it should be noted that the last scenario is considered only for comparison purposes since the noise-free data from chemical plants (here the Aspen model can be considered as a real chemical process) are generally unavailable. It can be seen from Table 3 that introducing noise-free data into brute force learning of LSTMs (i.e., standard LSTM models) may or may not improve their prediction accuracy. Specifically, when noise-free data from the same process (i.e., from Aspen model) is provided with noisy data, standard LSTM achieves a lower MSE in item 1c than the standard LSTM using noisy data only in item 1a; however, the standard LSTM using a mixed dataset with noise-free data from the first-principles model has a larger MSE due to the mismatch between the Aspen model (i.e., source of noisy data) and the first-principles model (i.e., source of noise-free data). This mismatch, if not handled appropriately, may misguide LSTM training and leads to worse prediction performance.

Table 3 – Open-loop prediction results under industrial noise.

Methods	MSE T
1a) LSTM: only using noisy data	1.8217
1b) LSTM: mixed data (noise-free data from fp)	3.0357
1c) LSTM: mixed data (noise-free data from Aspen)	1.5386
2a) Co-teaching LSTM (noise-free data from fp)	0.8596
2b) Co-teaching LSTM (noise-free data from Aspen)	0.7140
3) Dropout LSTM: only using noisy data	0.8761

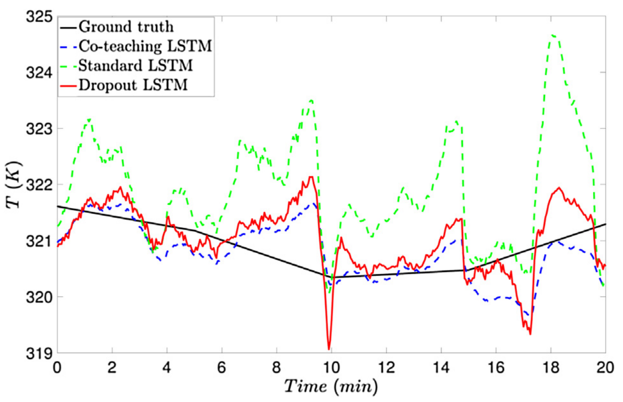

Fig. 7 – Open-loop state profile ($x_3 = T - T_s$) predicted by standard LSTM, co-teaching LSTM, and dropout LSTM, respectively, under the same inputs.

Subsequently, we train LSTM models using the co-teaching method with the same two types of mixed datasets (i.e., the noise-free data from Aspen model, and from first-principles model, respectively). The co-teaching LSTM training is initially equipped with a noisy dataset, and as the training evolves, noise-free data sequences are introduced into the learning process as discussed in the co-teaching algorithm. As shown in Table 3, the two co-teaching LSTM models have lower MSEs than the corresponding standard LSTM models using the same type of mixed dataset. It is also noticed that the co-teaching LSTM using noise-free data from Aspen simulations has the lowest MSE results among all the LSTM models in this study. Additionally, we train the dropout LSTM model using the noisy data only. It is shown in Table 3 that the dropout LSTM model (i.e., item 3) achieves lower MSE than the standard LSTM using the same noisy dataset (i.e., item 1a), which demonstrates the benefits of dropping out weights during training and testing to avoid overfitting to noisy data.

Finally, we show one of the open-loop prediction results in Fig. 7. We compare the temperature profiles predicted by the standard LSTM, dropout LSTM and co-teaching LSTM models which correspond to the models (1b), (2a), and (3) in Table 3. It is demonstrated in Fig. 7 that the co-teaching and dropout results are closer to the ground truth as compared to the standard LSTM model. This is also consistent with the MSE results in Table 3.

4.5. *Closed-loop simulation results*

Finally, we incorporate the LSTM models in the LMPC of Eq. (4), and carry out a closed-loop simulation study to demonstrate the improved closed-loop performance under dropout and co-teaching LSTM models. The control objective of LMPC is to stabilize the reactor temperature at its steady-state T_s by manipulating the heat input rate ΔQ. The LMPC objective function of Eq. (4a) is designed with the following form that has its minimum value at the steady-state:

$$L(x,u) = |x_3|^2_{Q_1} + |u|^2_{Q_2} \tag{9}$$

where Q_1 and Q_2 are the coefficient matrices that represent the contributions of temperature and of control actions (both are in deviation forms) in the MPC objective function. In this example, we choose $Q_1 = 1$ and $Q_2 = 5\times10^{-9}$ to balance the contributions of process state and manipulated input to the objective function. The nonlinear optimization problem of

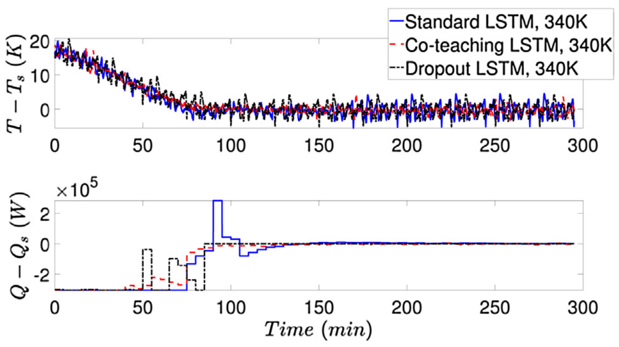

Fig. 8 – Closed-loop state profile ($x_3 = T - T_s$) and manipulated input profile ($u = Q - Q_s$) for the initial condition T = 340 K under the MPC using standard LSTM, co-teaching LSTM, and dropout LSTM, respectively.

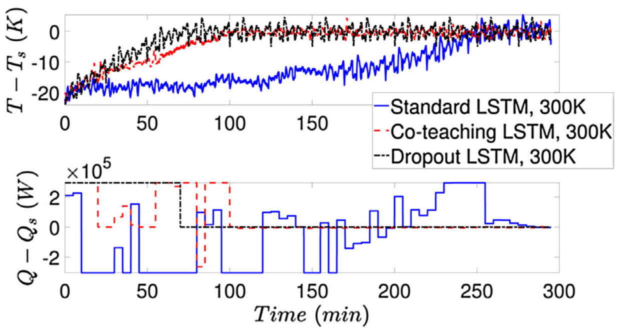

Fig. 9 – Closed-loop state profile ($x_3 = T - T_s$) and manipulated input profile ($u = Q - Q_s$) for the initial condition T = 300 K under the MPC using standard LSTM, co-teaching LSTM, and dropout LSTM, respectively.

LMPC is solved using the python module of the IPOPT software package (Wächter and Biegler, 2006), named PyIpopt with the sampling period $\Delta = 5$ min.

Figs. 8 and 9 show the closed-loop state profiles (with noisy measurements) under LMPC using standard LSTM, dropout LSTM, and co-teaching LSTM models for two different initial conditions. Specifically, Fig. 8 shows that starting from an initial temperature T = 340 K higher than the steady-state value, all the LSTM models drive the temperature to its steady-state within 100 min. However, in Fig. 9, with an initial temperature T = 300 K lower than the steady-state value, the standard LSTM model takes much longer time than the dropout and co-teaching LSTM models to stabilize the temperature at the steady-state.

To further analyze their closed-loop performances in terms of state convergence speed and energy consumption, we use the MPC objective function as an indicator of closed-loop performance as it accounts for both state and input information. It can be seen from Eq. (9) that a lower objective function value implies a faster convergence to the steady-state and less consumption of Q during operation. Therefore, we integrate the objective function value over the closed-loop simulation period t_s, i.e., $L_s = \int_{t=0}^{t=t_s} L(x,u)dt$, for each LSTM model. For the initial condition T = 340 K, L_s is calculated to be 44963.07 for standard LSTM, 39488.3 for dropout LSTM, and 37843.28 for co-teaching LSTM; for the initial condition T = 300 K, L_s is calculated to be 120697.7 for standard LSTM, 40636.26 for dropout LSTM, and 41083.07 for co-teaching LSTM. In both cases, co-teaching and dropout LSTM models have lower L_s values than standard LSTM model, which indicates an improvement in closed-loop performance. Additionally, we test more initial conditions of temperature within [300, 340] K under LMPC. It is demonstrated that for $T_{initial} > T_s$, all the three LSTM models can stabilize the temperature at the steady-state within a short time, while for $T_{initial} < T_s$, co-teaching and dropout LSTM models significantly improve the dynamic responses than standard LSTM (like the one in Fig. 9). For all the tested initial conditions, dropout and co-teaching LSTM models achieve better closed-loop performances with lower values of L_s.

Additionally, we calculate the computation time of solving different ML models as follows: (1) Standard LSTM model: 19.1 s (closed-loop), and 0.0185 s (open-loop), (2) Coteaching LSTM model: 18.61 s (closed-loop), and 0.0173 s (open-loop), and (3) Dropout LSTM model: 120.7 s (closed-loop), and 0.0728 s (open-loop), where 'open-loop' represents the time for one open-loop prediction, and 'closed-loop' represents the time for one MPC iteration. Note that the dropout LSTM model takes much longer time than the other two models because the predictive distribution of LSTM output is approximated by performing Monte Carlo simulations in serial mode. It is expected that the computation time for dropout models can be significantly reduced by employing parallel computing, which has been demonstrated in Wu et al. (2021). Since the computation time of all the ML models are less than one sampling time, which is 5 minutes in the simulation study, it implies that the ML models can be implemented in practice without any computational issues.

Remark 18. It should be noted that closed-loop stability under LMPC is not guaranteed since a sufficiently high model accuracy as required for machine learning models in Wu et al. (2019a) may be unachievable for the proposed LSTM models that receive noisy measurements to predict true states. However, from extensive closed-loop simulation runs, it is demonstrated that with a small noise level, the state trajectories for all types of LSTM models successfully converge to the steady-state from different initial conditions. As the noise level increases, the closed-loop performance gets worse for all LSTM models in the sense that most of the state trajectories can still converge to the steady-state but showing oscillation over time. In this case, the proposed dropout and co-teaching LSTM models can be used to improve closed-loop performance when training with noisy data.

5. Conclusion

In this work, we developed LSTM models using Monte Carlo dropout method and co-teaching technique to predict underlying process dynamics (ground truth) from noisy data. A chemical reactor was modeled and simulated in a large-scale process simulator, Aspen Plus, to demonstrate the application of dropout and co-teaching methods with noisy and noise-free data generated from Aspen simulation and first-principles model solutions, respectively. Then, the closed-loop simulation of the reactor was carried out in Aspen Plus Dynamics under LSTM-based MPC to demonstrate the superiority of dropout and co-teaching LSTM methods over the standard LSTM modeling approach in both open-loop prediction accuracy and closed-loop performance.

Declaration of Competing Interest

The authors report no declarations of interest.

Acknowledgments

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

References

Agarwal, A., Liu, Y., McDowell, C., 2019. 110th anniversary: ensemble-based machine learning for industrial fermenter classification and foaming control. Ind. Eng. Chem. Res. 58, 16719–16729.

Chen, S., Wu, Z., Rincon, D., Christofides, P.D., 2020. Machine learning-based distributed model predictive control of nonlinear processes. AIChE J. 66, e17013.

Feng, Z., Shen, W., Rangaiah, G.P., Dong, L., 2019. Closed-loop identification and model predictive control of extractive dividing-wall column. Chem. Eng. Process.-Process Intensif. 142, 107552.

Gal, Y., Ghahramani, Z., 2015. Bayesian Convolutional Neural Networks With Bernoulli Approximate Variational Inference. arXiv:1506.02158.

Gal, Y., Ghahramani, Z., 2016a. Dropout as a bayesian approximation: representing model uncertainty in deep learning. In: Proceedings of the International Conference on Machine Learning, New York City, New York, pp. 1050–1059.

Gal, Y., Ghahramani, Z., 2016b. A theoretically grounded application of dropout in recurrent neural networks. In: Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain, pp. 1019–1027.

Han, B., Yao, Q., Yu, X., Niu, G., Xu, M., Hu, W., Tsang, I., Sugiyama, M., 2018. Co-teaching: robust training of deep neural networks with extremely noisy labels. In: Proceedings of the Advances in Neural Information Processing Systems, Montreal, Canada, pp. 8527–8537.

Hassanpour, H., Corbett, B., Mhaskar, P., 2020. Integrating dynamic neural network models with principal component analysis for adaptive model predictive control. Chem. Eng. Res. Des. 161, 26–37.

Hsu, Y.L., Wang, J.S., 2009. A wiener-type recurrent neural network and its control strategy for nonlinear dynamic applications. J. Process Control 19, 942–953.

Kadlec, P., Grbić, R., Gabrys, B., 2011. Review of adaptation mechanisms for data-driven soft sensors. Comput. Chem. Eng. 35, 1–24.

Kamal, I., Malah, A., 2017. Aspen Plus Chemical Engineering Applications. John Wiley & Sons, Inc.

Krishnaiah, J., Kumar, C., Faruqi, M., 2006. Modelling and control of chaotic processes through their bifurcation diagrams generated with the help of recurrent neural network models. Part 1. Simulation studies. J. Process Control 16, 53–66.

Li, Z., Hong, X., Hao, K., Chen, L., Huang, B., 2020. Gaussian process regression with heteroscedastic noises – a machine-learning predictive variance approach. Chem. Eng. Res. Des. 157, 162–173.

Lima, F.V., Rawlings, J.B., 2011. Nonlinear stochastic modeling to improve state estimation in process monitoring and control. AIChE J. 57, 996–1007.

Ma, Y., Nore na-Caro, D., Adams, A., Brentzel, T., Romagnoli, J., Benton, M., 2020. Machine-learning-based simulation and fed-batch control of cyanobacterial-phycocyanin production in plectonema by artificial neural network and deep reinforcement learning. Comput. Chem. Eng. 142, 107016.

Mahmood, M., Mhaskar, P., 2012. Lyapunov-based model predictive control of stochastic nonlinear systems. Automatica 48, 2271–2276.

Mendoza, D.F., Palacio, L.M., Graciano, J.E., Riascos, C.A., Vianna Jr., A.S., Le Roux, G.C., 2013. Real-time optimization of an industrial-scale vapor recompression distillation process. model validation and analysis. Ind. Eng. Chem. Res. 52, 5735–5746.

Patwardhan, S.C., Narasimhan, S., Jagadeesan, P., Gopaluni, B., Shah, S.L., 2012. Nonlinear Bayesian state estimation: a review of recent developments. Control Eng. Pract. 20, 933–953.

Shah, D., Wang, J., He, Q.P., 2020. Feature engineering in big data analytics for iot-enabled smart manufacturing-comparison between deep learning and statistical learning. Comput. Chem. Eng. 141, 106970.

Sharifzadeh, M., 2013. Integration of process design and control: a review. Chem. Eng. Res. Des. 91, 2515–2549.

Wächter, A., Biegler, L.T., 2006. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Math. Program. 106, 25–57.

Weissel, F., Huber, M.F., Hanebeck, U.D., 2009. Stochastic nonlinear model predictive control based on Gaussian mixture approximations. In: Informatics in Control, Automation and Robotics. Springer, Berlin, Germany, pp. 239–252.

Wu, Z., Rincon, D., Luo, J., Christofides, P.D., 2021. Machine learning modeling and predictive control of nonlinear processes using noisy data. AIChE J. 66, e17164.

Wu, Z., Tran, A., Rincon, D., Christofides, P.D., 2019a. Machine learning-based predictive control of nonlinear processes. Part I: theory. AIChE J. 65, e16729.

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

Wu, Z., Zhang, J., Zhang, Z., Albalawi, F., Durand, H., Mahmood, M., Mhaskar, P., Christofides, P.D., 2018. Economic model predictive control of stochastic nonlinear systems. AIChE J. 64, 3312–3322.

Xie, W., Bonis, I., Theodoropoulos, C., 2015. Data-driven model reduction-based nonlinear mpc for large-scale distributed parameter systems. J. Process Control 35, 50–58.

Yang, F., Li, K., Zhong, Z., Luo, Z., Sun, X., Cheng, H., Guo, X., Huang, F., Ji, R., Li, S., 2020. Asymmetric co-teaching for unsupervised cross-domain person re-identification. Proceedings of the Thirty-Fourth AAAI Conference on Artificial Intelligence, 12597–12604.

Yeo, K., 2019. Short Note on the Behavior of Recurrent Neural Network for Noisy Dynamical System. arXiv:1904.05158.