I&EC research
Industrial & Engineering Chemistry Research

pubs.acs.org/IECR Article

Machine Learning Modeling and Predictive Control of the Batch Crystallization Process

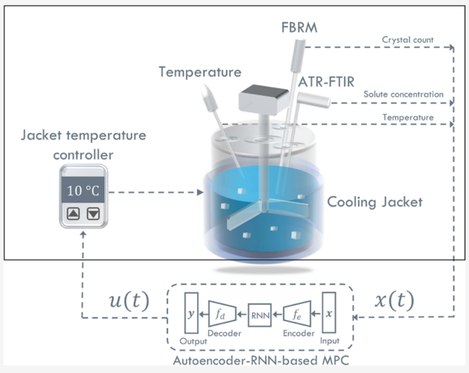
Yingzhe Zheng, Xiaonan Wang,* and Zhe Wu*

Cite This: *Ind. Eng. Chem. Res.* 2022, 61, 5578−5592

Read Online

ACCESS | Metrics & More | Article Recommendations

ABSTRACT: This work develops a framework for building machine learning models and machine-learning-based predictive control schemes for batch crystallization processes. We consider a seeded fesoterodine fumarate cooling crystallization and dissolution process in a batch reactor and present the methodology and implementation of simulation, modeling, and controller design. Specifically, to address the experimental data scarcity problem, we first develop a one-dimensional population balance model based on published kinetic parameters that were obtained empirically to describe the formation of crystals via nucleation, growth, and agglomeration. Then, recurrent neural network (RNN) and autoencoder−RNN (AERNN) models are developed using data from extensive open-loop simulations of the semi-empirical population balance model under various operating conditions to capture the process dynamic behavior. Two model predictive control (MPC) schemes using the respective RNN and AERNN models are developed to optimize the crystallization process with respect to product yield, crystal size, number of fines in the final product, and energy consumption, while accounting for the constraints on manipulated inputs. Through open- and closed-loop simulations, it is demonstrated that the RNN and AERNN models capture the process dynamics well, and the RNN- and AERNN-based MPCs achieved the desired product yield and crystal size with significantly improved computational efficiency.



■ INTRODUCTION

Historically, batch crystallization has been a vital unit operation in the pharmaceutical industry and is instrumental in the separation and purification of intermediate compounds and active pharmaceutical ingredients (APIs) from the liquid solutions. It is deemed an indispensable component of most pharmaceutical processes as more than 90% of APIs are synthesized in the form of crystals.[1] Attributing to the stringent standards imposed by the U.S. Food and Drug Administration (FDA) and the high value-added nature of the final product, crystal size distribution (CSD), purity, shape, and yield are often of major concerns to manufacturers due to their profound impacts on process productivity, downstream processing (e.g., filtration, drying, milling, and tableting), and drug performance (e.g., stability, dissolution rate, and bioavailability). These physicochemical properties are primarily determined by the intricate interplay between the crystallization kinetics of growth, nucleation, agglomeration, and breakage, which are in turn heavily influenced by the crystallization operating conditions. Therefore, selecting the optimal operating and control strategies is of great interest to pharmaceutical manufacturers in attaining a more efficient and greener achievement of the specification targets.

As a trillion dollar industry today, the pharmaceutical sector has experienced a consequential transformation in its manufacturing paradigm in the last decade as the internet of things, artificial intelligence, and advanced computing begin to challenge the traditional practices.[2] Modern pharmaceutical development approaches leverage the quality-by-control (QbC) methodology for an enhanced process understanding and product assurance. QbC emphasizes the use of mathematical and knowledge-based modeling for quantitative and predictive product and process understanding and is widely recognized as a valuable tool for improved mechanistic understanding, process optimization, and robust control.[3−5] However, modeling of pharmaceutical processes, particularly crystallization, can be of great challenge due to their complex, stochastic, and kinetic-driven nature.[6,7] Moreover, as contrary to its ubiquity, crystallization is often not well-understood by practitioners, and this renders optimal control a vision rather than a reality. While previous studies have sought to elucidate the optimization and control of crystallization processes,[8−18] a

Received: January 4, 2022
Revised: March 30, 2022
Accepted: April 1, 2022
Published: April 13, 2022

majority have adopted the first-principles modeling approach [e.g., population balance model (PBM)], which is susceptible to high computation complexity and is hence less amenable for real-time optimal control. Although improved optimization algorithms could be adopted to enhance the calculation time,[15] determination and periodic adjustments of the various crystallization kinetics based on new process data obtained can be cumbersome. A recent pioneering work has employed data-driven models in reducing computational costs.[19] However, the proposed model operates on mean crystal properties and not on the actual CSD.[15] Therefore, with no silver bullet solution for optimizing and controlling pharmaceutical crystallization processes in hand, the development of new approaches is of compelling need. Ascribing to the burgeoning research and development in many process analytical technologies (PATs) and the pervasive nature of data in modern pharmaceutical enterprises, machine learning model-based predictive control (MPC) has emerged as the state-of-the-art optimal control strategy for cooling crystallization processes.

MPC is an optimization-based advanced control method that solves for the optimal control actions based on a predictive model of the process while accounting for inherent process characteristics. The development of accurate and computationally efficient process models has been a long-standing research problem in predictive control of dynamic processes. Recently, machine learning has attracted an increased level of attention in model identification of chemical processes due to its ability to model multidimensional, nonlinear systems from process operational data. Among the various machine learning models, deep neural networks have been widely employed in previous studies in process modeling, optimization, and control. For example, two nonlinear artificial neural network-based predictive controllers have been designed to optimize the performance of a batch sugar crystallization process[20] and a continuous KCl cooling crystallization process,[21] respectively; in addition, artificial neural networks and genetic algorithms have been utilized in the modeling and optimization of a commercial drug crystallization process,[22] and a radial basis functions network model-based predictive controller is realized for optimizing the performance of a crystallization process of ibuprofen from ethanol.[23] However, advanced control methods such as MPC have not been explored and incorporated into some of the optimization frameworks proposed in earlier studies,[22] and others have adopted the conventional feedforward artificial neural networks in constructing surrogate models, which are inadequate in capturing the entire system dynamics embedded in the PBM for use as predictive models in MPC controllers. This forbids the explicit specification of crystal size targets in the objective functions of the optimization problems as crystal size can only be implicitly controlled through optimizing the supersaturation of the crystallization system.[20,22] Furthermore, another class of deep neural networks, recurrent neural networks (RNNs), have received considerable attention due to their outstanding performance in modeling nonlinear dynamic systems using process time series data. Various RNN-based MPCs have been designed and implemented in many recent studies to control chemical plants (e.g., phthalic anhydride synthesis in a fixed-bed catalytic reactor[24] and continuous stirred tank reactors with an irreversible second-order exothermic reaction[25−27]) in real time and optimize process performance accounting for closed-loop stability, safety, and control actuator constraints. However, while previous studies have explored RNN-based modeling, optimization, and control of batch crystallization processes (e.g., cooling crystallization of paracetamol from water[28]), the RNN developed is not capable of capturing the rich dynamics offered by the full PBM as the training data are generated by solving the PBM using method of moments rather than the classes method. The resulting optimization can thus be only performed with respect to mean crystal size, and the incorporation of explicit targets for specific crystal size or its distribution in the objective function is not possible.

Furthermore, in addition to the RNN, an autoencoder (AE) neural network has been widely adopted as a dimensionality reduction technique for preprocessing model input data, which could further alleviate the computational burden of the machine learning model. Functionally, an AE seeks to extract low-dimensional features from data residing in the original high-dimensional space by uncovering complex nonlinear relationships within the data. This effectively reduces the number of neurons and layers required by the RNN model to fit the data in the lower-dimensional latent space and thereby further enhance its computational efficiency. In general, the AE is beneficial for analyzing data sets with a substantial number of features and has found promising application as a model reduction technique for data-driven modeling approaches in different domains (e.g., catalytic tubular reactor system identification,[29] malware detection,[30] image recognition and classification,[31,32] and control of a diffusion-reaction process[33]). However, the incorporation of an AE in machine-learning-based MPC to improve the computational efficiency of the machine-learning-based control of batch crystallization systems characterized via the high-dimensional PBM and solved via the classes method, to the best of the authors' knowledge, has not been previously reported.

Motivated by the above considerations, in this manuscript, we develop a general framework for building machine learning models and developing machine-learning-based predictive control schemes for batch crystallization processes. Specifically, we consider a seeded fesoterodine fumarate (FF) cooling crystallization and dissolution process in a batch reactor and first develop a process model based on published kinetic parameters and experimental data in ref 34. To address the issue pertaining to high computational complexity of this semi-empirical process model (i.e., PBM) in optimizing and controlling the batch crystallization operation, we build an RNN and an AERNN model using simulation data generated from extensive open-loop simulations of the PBM. Subsequently, RNN and AERNN-based model predictive controllers are developed to optimize product yield, crystal size, number of fines in the final product, and energy consumption while accounting for the physical constraints on cooling jacket temperature. Finally, the proposed RNN- and AERNN-based MPC schemes are applied to the batch crystallization process of FF and their performance are compared with that of the MPC using the PBM. The simulation results demonstrate that both the proposed RNN- and AERNN-based MPCs achieve desired closed-loop performance and significantly improve computational efficiency as compared to the MPC using the PBM.

■ MODELING OF SEEDED BATCH FF COOLING CRYSTALLIZATION

This section introduces the background of the batch cooling crystallization system employed in this work. Specifically, the mechanistic model formulation and the basic operating mode of the crystallizer are first presented. Then, an overview of the input−output behavior for this process is used to further explain the data structure used in the neural network model.

Population Balance. FF [isobutyric acid 2-((*R*)-3-diisopropylammonium-phenylpropyl)-4-(hydroxymethyl)-phenyl ester hydrogen fumarate, Figure 1] is a muscarinic antagonist indicated for the treatment of overactive bladder.[35] Its crystalline state can exist in several polymorphic forms (I, A, B, and C), which can be obtained from a variety of solvent combinations. In particular, form I, which resembles a cubic structure, may be produced via seeded cooling crystallization from solutions in 2-butanone,[36] whose mechanisms have been well-investigated in ref 34 and are summarized in a PBM, as shown in eq 1

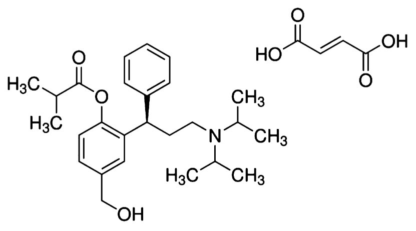


Figure 1. Chemical structure of FF.

$$\frac{\partial \psi}{\partial t} + \frac{\partial (G\psi)}{\partial L} - \frac{\partial (D\psi)}{\partial L} = r_B - r_D \tag{1}$$

where the first term on the left-hand side describes the variation of population density ψ with respect to time. The second and third terms on the left-hand side represent the rates of convective processes along the internal coordinate (crystal size L) through crystal growth G and dissolution D, respectively. The terms on the right-hand side characterize the rates of change in the total number of crystals formed r_B through nucleation and agglomeration and their disappearance r_D as a result of agglomeration and dissolution. The phenomenon associated with crystal breakage was neglected due to the absence of breakage-prone needle-shaped crystals.

Considered as the most detailed and widely adopted approach for modeling pharmaceutical crystallization processes, the PBM entails a comprehensive quantitative description of the time-dependent evolution of the crystal number, size distribution, crystal mass, and solute concentration (which in turn allows the computation of the product yield).[7] At its core, the PBM is a partial integrodifferential equation that encompasses the coupling of all kinetic terms involved in different crystallization phenomena (e.g., nucleation B_N, growth N, and agglomeration A). It is typically solved numerically as closed-form solutions are unavailable. Numerical schemes such as the method of classes developed in ref 37 can be utilized to discretize eq 1, where the size coordinate is divided into r classes between L_0 and L_r, with class i representing the crystal population of size within L_{i-1} to L_i. Following the modeling approach presented in ref 34, eq 1 can thus be re-expressed as a system of r ordinary differential equations with population density ψ replaced by the number of crystals N_i in size class i and w_i representing the width of size class i.

$$\frac{dN_1}{dt} = -\frac{N_1}{2w_1}G_1 + B_N - A_{D,1} \tag{2}$$

$$\frac{dN_i}{dt} = -\frac{N_i}{2w_i}G_i + \frac{N_{i-1}}{2w_{i-1}}G_{i-1} + A_{B,i} - A_{D,i} \tag{3}$$

$$\frac{dN_r}{dt} = \frac{N_{r-1}}{2w_{r-1}}G_{r-1} + A_{B,r} \tag{4}$$

$$\frac{dN_1}{dt} = -\frac{N_1}{2w_1}D_1 + \frac{N_2}{2w_2}D_2 \tag{5}$$

$$\frac{dN_i}{dt} = -\frac{N_i}{2w_i}D_i + \frac{N_{i+1}}{2w_{i+1}}D_{i+1} \tag{6}$$

$$\frac{dN_r}{dt} = -\frac{N_r}{2w_r}D_r \tag{7}$$

The system of eqs 2−7 can be solved numerically using standard integration methods (e.g., Runge−Kutta methods and finite difference method). Equations 2−4 collectively describe the rates of change of the crystal number within each size class i through nucleation B_N, crystal growth G, and agglomeration A in a supersaturated system, whereas eqs 5−7 account for the rates of change of crystal numbers due to crystal dissolution D in an undersaturated system. More specifically, the PBM formulation proposed in ref 34 considers only secondary nucleation due to a very wide metastable zone observed for FF cooling crystallization. Equations 2−4 comprise the size-dependent crystal growth rate G_i, where a two-step growth mechanism consisting of mass transfer from bulk solution to the solid−liquid interface followed by mass integration into the crystal lattice is employed. In contrast, the size-dependent crystal dissolution rate D_i in eqs 5−7 is mechanistically described as the opposite of crystal growth, where molecules detach from the crystal lattice to the solid−liquid interface before diffusing into the undersaturated bulk solution. Finally, the size-dependent crystal agglomeration birth rate $A_{B,i}$ and disappearance rate $A_{D,i}$ in eqs 2−4 are described through a binary mechanism proposed in ref 37, where crystals in size classes j and k ($k \geq j$) aggregate to form crystals of class l where $l > k, j$. Interested readers are referred to ref 34 for a detailed discussion on the PBM formulation and the computation of the various crystallization kinetics involved in seeded batch cooling crystallization of form I FF.

Energy Balance. It is assumed that the enthalpy of crystallization is negligible and the batch crystallizer is perfectly mixed with a uniform temperature distribution. Temperature change in the crystallizer can be estimated using the following ordinary differential equation

$$\frac{dT_r}{dt} = \frac{UA(T_j - T_r)}{m_r c_{p,r}} \tag{8}$$

where the rate of change of crystallizer temperature T_r is determined by the overall heat transfer coefficient U, area of the crystallizer wall A available for heat transfer, difference between jacket temperature T_j and crystallizer temperature T_r, mass of the crystallization mixture m_r, and specific heat capacity of the crystallization mixture $c_{p,r}$.

Mass Balance. The total mass of the crystals formed is computed as the sum of crystal mass values within each size class i

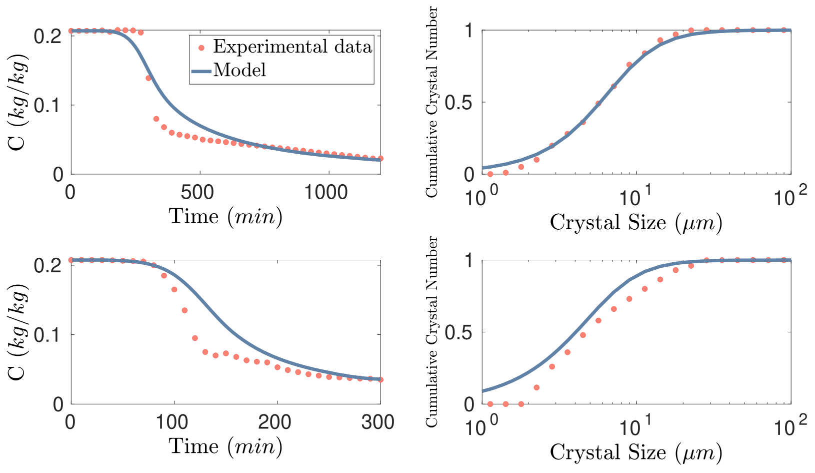


Figure 2. Validating the PBM developed in this work against experimental data reported in ref 34 at different cooling rates and m_{seed}. Top figures: cooling rate ≈ 2.5 °C/h and m_{seed} = 0.01 kg and bottom figures: cooling rate ≈ 10 °C/h and m_{seed} = 0.005 kg.

$$m_{\text{cr}} = k_\nu \rho_{\text{cr}} \sum_{i=1}^{r} N_i S_i^3 \tag{9}$$

where k_ν is the crystal volume factor, ρ_{cr} is the density of crystals, and the total volume of the crystals formed is determined as the sum of the product between number of crystals N_i and average crystal size S_i in each class i.

The bulk solute concentration $C(t)$ at a certain time instance during the crystallization process is calculated by subtracting the mass of crystals $m_{\text{cr}}(t)$ formed at that time from the initial mass of the dissolved solute m_0 and the mass of seed crystals m_{seed}, divided by the mass of the solvent m_{sol}.

$$C(t) = \frac{m_0 + m_{\text{seed}} - m_{\text{cr}}(t)}{m_{\text{sol}}} \tag{10}$$

Model Validation. The PBM comprising eqs 2−10 is developed based on the published crystallization and dissolution kinetic parameters from ref 34 and validated against the experimental data reported for the seeded FF batch cooling crystallization at different operating conditions (e.g., different m_{seed} and cooling rates).[34] It is demonstrated in Figure 2 that the model developed in this work agrees reasonably well with the empirical data obtained from experiments (e.g., solute concentration profiles and cumulative CSD of the final product) under two different operating operations. The discrepancies can be attributed to the assumptions made in the PBM formulation (e.g., the kinetic parameters are assumed to be temperature-independent, the exponent of the total mass of crystals m_{cr} is assumed to be unity in modeling nucleation, and agglomeration of crystals is assumed to be binary)[34] and the different techniques and values adopted in estimating the physicochemical properties of the crystallization system (e.g., the critical volume V_c of the solute in estimating the diffusion coefficients using the Wilke−Chang equation). Nevertheless, as shown in Figure 2, the PBM developed in this work is able to capture the general trends of the concentration and the final CSD profiles of the crystallization process to a reasonable extent, and the simulation results are also found to be in close agreement with the ones reported in ref 34. The model is thus deemed valid for describing the dynamic behavior of the batch system of interest.

■ RNN DEVELOPMENT

Due to the complexity of the PBM, it is computationally impractical to solve it in real-time optimization and control of the batch process operation. To address this issue, reduced-order models have been utilized in the literature for the purpose of real-time control. However, state information and prediction accuracy are inevitably sacrificed by using the model reduction method. Motivated by the above considerations, in this section, we propose a machine learning modeling approach to capturing the dynamic behavior of the batch crystallization process using data from experiments and simulations. We will demonstrate that the machine learning model significantly improves computational efficiency while maintaining a desired model fidelity and prediction accuracy as compared to that of the semi-empirical PBM.

RNN Formulation. The RNN model is utilized to model the batch crystallizer for FF crystallization using simulation data generated from the PBM presented in the previous section. Specifically, the RNN is constructed with input, hidden, and output layers (Figure 3), where the states in the hidden layers $\mathbf{x} \in \mathbf{R}^{d_x}$ are represented as follows

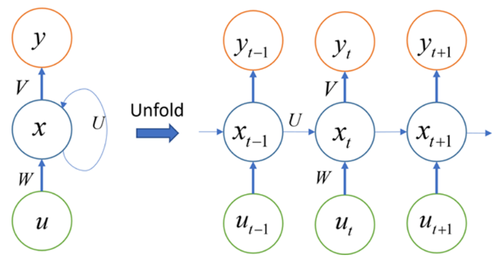


Figure 3. RNN structure.

$$\mathbf{x}_t = \sigma_h(U\mathbf{x}_{t-1} + W\mathbf{u}_t) \tag{11}$$

where $\mathbf{u}_t \in \mathbf{R}^{d_u}$ are the RNN inputs at time t, and the weight matrices $W \in \mathbf{R}^{d_x \times d_u}$ and $U \in \mathbf{R}^{d_x \times d_x}$ are associated with the input and hidden state vectors, respectively. The element-wise nonlinear activation function is denoted by σ_h (e.g., ReLU).

The output layer $\mathbf{y}_t$ is calculated using the following equation

$$\mathbf{y}_t = \sigma_y(V\mathbf{x}_t) \tag{12}$$

where the activation function σ_y and the weight matrix $V \in \mathbf{R}^{d_y \times d_x}$ are associated with the output layer. In regression problems, a linear unit is generally used as the activation function in the output layer. To simplify the notation, the RNN model of eqs 11 and 12 can be represented as the following continuous-time nonlinear system.[38]

$$\dot{\mathbf{x}} = F_{\mathrm{nn}}(\mathbf{x}, \mathbf{u}) := A\mathbf{x} + \Theta^{\mathrm{T}}\mathbf{z} \tag{13}$$

where $\mathbf{x} \in \mathbf{R}^{d_x}$ is the RNN state vector and $\mathbf{u} \in \mathbf{R}^{d_u}$ is the RNN input vector. $z = [z_1,...,z_{d_x},z_{d_x+1},...,z_{d_x+d_u}] = [\sigma(x_1),...,\sigma(x_{d_x}),u_1,...,u_{d_u}] \in \mathbf{R}^{d_x+d_u}$ is a vector of both the network state $\mathbf{x}$ and the input $\mathbf{u}$, and $\sigma(\cdot)$ is the nonlinear activation function. A and Θ are the coefficient matrices consisting of RNN weights.

Contrary to the one-way connectivity between units in feedforward neural networks (FNNs), one dominant advantage of RNNs is that RNNs have signals traveling in both directions (i.e., forward and backward) by introducing loops in the network. This enables the feedback of information derived from earlier inputs into the network, thereby exhibiting a dynamic behavior. Furthermore, based on the universal approximation theorem of FNNs, it is shown in ref 39 that an RNN model with a sufficient number of neurons is capable of approximating any dynamic nonlinear system on compact subsets of the state space for finite time. This makes the RNN an ideal candidate for approximating the PBM comprising the continuous-time nonlinear systems of eqs 2−10.

Autoencoder. The AE is an unsupervised dimensionality reduction algorithm constituting the feedforward artificial neural network architectures.[31] Fundamentally, an AE learns to compress and encode an input data efficiently by minimizing its reconstruction error. The output from an encoder function comprises the complex hierarchical nonlinear features that are of lower dimensions and are sometimes considered as more efficient representations of the original data.[30] Some advantages of incorporating an AE in machine learning tasks include reduced training time, more robust against overfitting, and more convenient data visualization.[30] Figure 4 illustrates the basic architecture of an AE.

A typical AE contains two primary components:

1. Encoder function: parameterizes via $\theta = \{W_e, b\}$, which maps the input data $\mathbf{x} \in \mathbf{R}^{d_x}$ to a reduced representation of the input (also referred to as the **code**) $\mathbf{x}_r \in \mathbf{R}^{d_h}$ of the hidden layer via a function $f_e(\cdot)$, and is mathematically described by eq 14.

$$\mathbf{x}_r = f_e(\mathbf{x}) = \sigma_e(W_e\mathbf{x} + b) \tag{14}$$

where $W_e \in \mathbf{R}^{d_h \times d_x}$ is the weight matrix and $b \in \mathbf{R}^{d_h}$ is the bias. $\sigma_e(\cdot)$ is the nonlinear activation function (e.g., hyperbolic tangent function).

2. Decoder function: parameterizes via $\theta' = \{W_d, b'\}$, which reconstructs the original input data $\mathbf{x}'$ from the encoded representation $\mathbf{x}_r$ using the function $f_d(\cdot)$

$$\mathbf{x}' = f_d(\mathbf{x}_r) = \sigma_d(W_d\mathbf{h} + b') \tag{15}$$

where $W_d \in \mathbf{R}^{d_x \times d_h}$ and $b' \in \mathbf{R}^{d_x}$ are another weight matrix and bias, respectively. $\sigma_d(\cdot)$ is the linear activation function.

The AE learns the optimal weight matrices and biases by minimizing the following mean squared error (MSE) loss function

$$\theta, \theta' = \min_{\theta,\theta'} \frac{1}{n}\sum_{i=1}^{n} L(\mathbf{x}_i, \mathbf{x}_i') \tag{16}$$

where L is the loss function or the reconstruction error, represented by $L(\mathbf{x}_i,\mathbf{x}_i') = \|\mathbf{x}_i - \mathbf{x}_i'\|^2$, and n is the number of training data.

In general, the AE may be regarded as a special type of FNNs and can be trained using similar techniques such as the classic minibatch gradient descent with gradients computed from back-propagation. It closely resembles the principal component analysis (PCA), which is one of the most commonly used linear dimensionality reduction techniques.[40] If only the linear activation function is employed in the AE, the output from the encoder function would directly correspond to the principal components in PCA. Given its flexibility in adopting a wide range of activation functions and generating complex nonlinear representative features from the original input data, the AE is usually preferred to the conventional PCA for more effective dimensionality reduction.

Autoencoder−Recurrent Neural Network. The exploitation of the AE in reducing the dimensionality of input and output data could further aid the computational efficiency of the RNN model. Figure 5 illustrates the structure of the AERNN model proposed in this work. Notably, only one AE is developed for the state variables present in both model input and output data sets to expedite the RNN model training and computation time (i.e., the number of neurons at the input and output layers of the model are now effectively reduced, and hence, less computations are now required for its development). In practice, the model output is converted back to its original space using the decoder function $f_d(\cdot)$ since its latent representation hardly contains any explicable physical information.

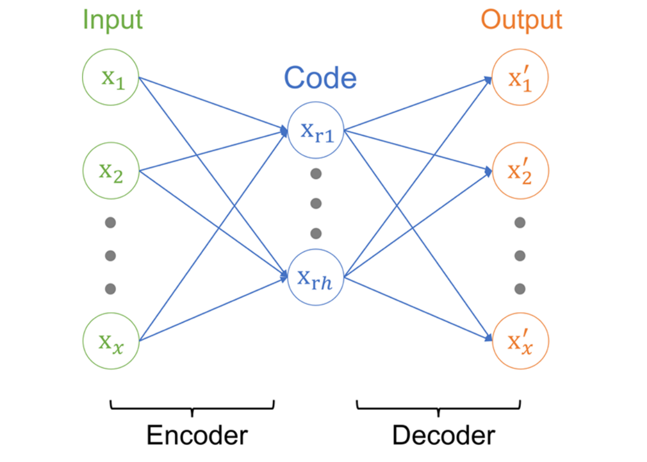


Figure 4. Fundamental structure of an AE.

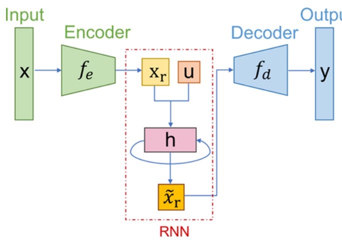


Figure 5. Structure of the AERNN model developed in this work.

Similarly, the AERNN model can be represented as the following continuous-time nonlinear system.

$$\dot{\mathbf{x}}_r = F'_{\text{nn}}(\mathbf{x}_r, \mathbf{u}) := A_r\mathbf{x}_r + \Theta_r^{\text{T}}\mathbf{z}_r \tag{17}$$

where the notations follow those in eq 13, with subscript r denoting the parameters for the reduced-order RNN model using the AE.

Data Generation. To construct an RNN or AERNN model with a desired accuracy, extensive open-loop simulations were first conducted to obtain a rich data set that captures the system dynamics of the batch crystallization process. The system of 2 is solved numerically using the explicit Runge–Kutta method of order 5(4) with an integration time step of h_c = 1 min. The Runge–Kutta method builds upon the Euler method and is one of the most widely employed methods in the temporal discretization for the approximate solutions of ordinary differential equations due to its low approximation error.[41] Adopting similar process conditions to those reported in ref 34, the open-loop simulation was performed with different initial conditions (Table 1) subject to a sequence of

Table 1. Summary of the Initial Conditions Adopted in Open-Loop Simulation

condition	m_{sol} (kg)	m_0 (kg)	m_{seed} (kg)	seed d_{50} (μm)	seeding T (°C)
1	0.482	0.1	0.01	4.2	35
2	0.482	0.1	0.005	6.4	30
3	0.482	0.1	0.005	6.4	35

varying jacket temperatures (generated in a pseudo-random manner) in a sample-and-hold fashion (i.e., the jacket temperature is fed into the system of eqs 2–10 as a piecewise constant function, $T_j(t) \in [-10, 30]$ °C and $T_j(t) = T_j(t_k)$, $\forall\, t \in [t_k, t_{k+1})$, where $t_{k+1} := t_k + \Delta$ with Δ = 30 min indicating one sampling period).

All other operating conditions (e.g., stirring rate) remain constant in each batch run. As the PBM formulation in ref 34 incorporates T_j as the only manipulated input, other variables (e.g., stirring rate) could not be considered. It should be noted that the proposed RNN modeling approach in this work can be generalized to more comprehensive mechanistic models that account for all other operating conditions as manipulated inputs.

With the aforementioned configuration, dynamic state trajectories (C, T_r, N_i) are sampled at every integration time step h_c = 1 min within each sampling period. Simulation data from a total of 1800 (600 for each initial condition) batch runs, each lasting for 15 h, are collected. The state trajectories from each batch run are then discretized, where the time series data are separated into 30 time series samples with a period of Δ = 30 min, which represents the prediction horizon of the RNN model. As a result, 54,000 data sets (18,000 for each initial condition) are obtained and are partitioned into training (60%), validation (10%), and testing (30%) data sets. It is essential to obtain such a large amount of simulation data in order to guarantee a comprehensive representation of the dynamic batch process at different operating conditions and system states (e.g., T_r, C, N_i).

Open-Loop Simulation Results. We first construct a full-order RNN model using all the states and manipulated inputs available in their original high dimension space, followed by the development of an AERNN model with inputs and outputs residing in the latent space of a reduced dimension. Open-loop simulations are carried out in this subsection to evaluate the prediction performance of a full-order RNN model and a reduced-order RNN model (i.e., an AERNN model).

Full-Order RNN Model. The full-order RNN model $F_{\text{nn}}(\mathbf{x},\mathbf{u})$, with $\mathbf{x} \in \mathbf{R}^{42}$ and $\mathbf{u} \in \mathbf{R}^1$ (i.e., the input data have 43 features), is developed using the state-of-the-art application programming interface (API) Keras,[42] to predict future states (i.e., C, T_r, N_i) for one sampling period of Δ = 30 min given the current state measurements and the manipulated input (i.e., T_j). As shown in Figure 3, in each prediction period, the time interval between two consecutive internal states x_{t-1} and x_t for the unfolded RNN is chosen to be the integration time step h_c used in open-loop simulations. Therefore, all the states between t = 0 and $t = \Delta$ with a step size of h_c are treated as the internal states and can be predicted by the RNN model. Finally, the RNN model is designed with two hidden recurrent layers consisting of 160 and 150 recurrent units regularized with dropouts of 0.25 and 0.15, respectively. The activation function is chosen to be the rectified linear activation function (ReLU). Furthermore, to facilitate more efficient training of

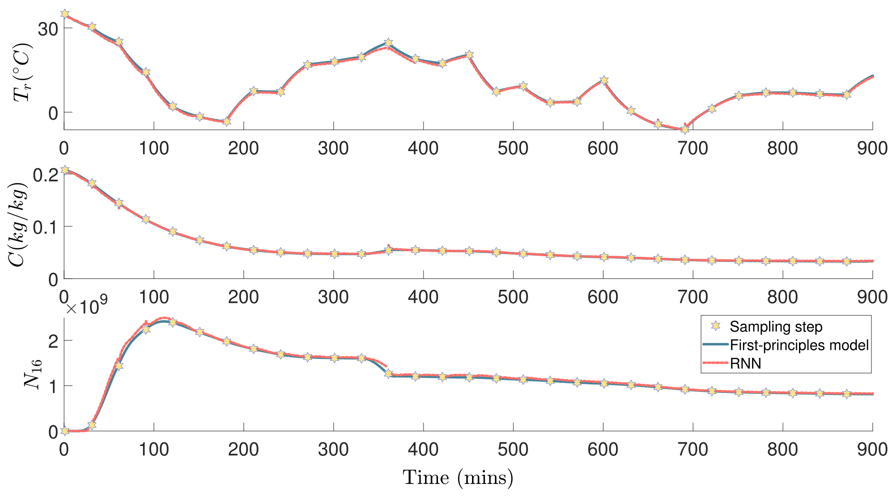


Figure 6. Comparison of crystallizer temperature T_r (top figure), solute concentration C (middle figure), and crystal number N_{16} of size class 16 (bottom figure) predicted using the RNN model and the first-principles model, respectively, where the stars denote the sampling time.

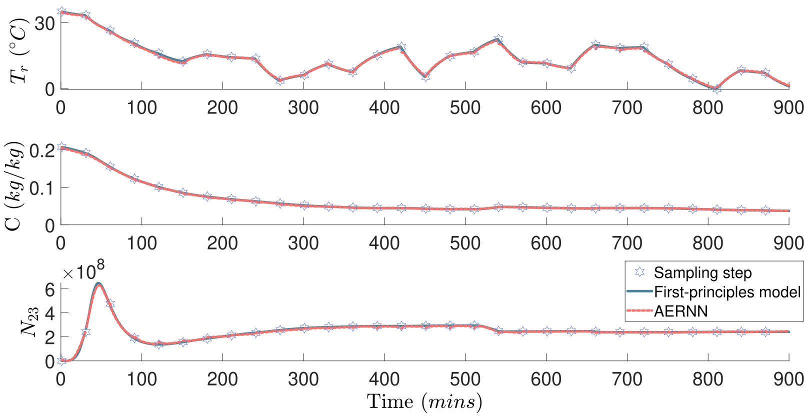


Figure 7. Comparison of crystallizer temperature T_r (top figure), solute concentration C (middle figure), and crystal number N_{23} of size class 23 (bottom figure) predicted using the AERNN model and the first-principles model, respectively, where the stars denote the sampling time.

the RNN, both input and output data are normalized to values between 0 and 1. The resulting RNN model is trained for 500 epochs using the Adam optimizer and attains MSE values of 1.63×10^{-5} and 2.16×10^{-5} on validation and testing data sets, respectively. Figure 6 compares the crystallizer temperature T_r, solute concentration C, and crystal number N_{16} of size class 16 predicted via the RNN model and the first-principles model of eqs 2−10, respectively, under the same pseudo-random jacket temperature profile. While the system dynamics for the full batch simulation of 900 min are generated by solving the first-principles model given the system states (i.e., C, T_r, N_i $\forall$ $i = 1, ..., 40$) at time = 0 subject to a sequence of the pseudo-random manipulated variable T_j, whose value only changes at the time instances denoted by the stars in Figure 6, the RNN model is developed to predict only one sampling period (30 min) forward given the current system states (i.e., C, T_r, N_i $\forall$ $i = 1, ..., 40$) and the manipulated variable (i.e., T_j), and thus, it is recursively applied to predict the entire trajectory for 900 min. The stars in Figure 6 denote the time instances where the state measurements and the manipulated variable are supplied to the RNN model to predict the evolution of the system states in the next sampling period Δ of 30 min. It is demonstrated in Figure 6 that the RNN prediction results and the first-principles model results are in close agreement with each other under various jacket temperatures.

Reduced-Order RNN Model. Next, we construct an AERNN model following the same approach as that of the full-order RNN model except that the input data are compressed to 16 features from 43 features in the original data set. This minimum number of new features (e.g., 16 in this work) that is found to be representative of the original 43 features is determined by gradually increasing the number of neurons in the code layer until no substantial improvement in the MSE was observed. The resulting AERNN model achieved MSEs of 3.45×10^{-5} and 3.29×10^{-5} on validation and testing data sets, respectively. These MSEs are relatively larger than those of the full-order RNN model as modeling accuracy is inevitably compromised with model reduction techniques. Figure 7 compares the crystallizer temperature T_r, solute concentration C, and crystal number N_{23} of size class 23 predicted via the AERNN model and the first-principles model of eqs. 2-10, respectively, under the same pseudo-random jacket temperature profile. Similar to the full-order RNN modeling approach, the system dynamics for the full batch simulation of 900 min are generated by solving the first-principles model given the system states (i.e., C, T_r, N_i $\forall$ $i = 1, ..., 40$) at time = 0 subject to a sequence of the pseudo-random manipulated variable T_j, whose value only changes at the time instances denoted by the stars in Figure 7. The AERNN model is developed to predict only one sampling period (30 min) forward given the current system states (i.e., C, T_r, N_i $\forall$ $i = 1, ..., 40$) and the manipulated variable (i.e., T_j), and thus, it is recursively applied to predict the entire trajectory for 900 min. The stars in Figure 7 denote the time instances where the state measurements and the manipulated variable are supplied to the AERNN model to predict the evolution of the system states in the next sampling period Δ of 30 min. It is demonstrated that the AERNN prediction results are consistent with the results from the first-principles model under various jacket temperatures, and hence, the AERNN model is adequate in describing the dynamics of the batch crystallization process.

In the simulation, it is noted that due to the computational complexity of the PBM, it took much longer to solve for each batch run than that for the RNN and AERNN (Table 2), and this thus poses a great hindrance to its applicability in real-time process optimization and control. However, both the RNN and AERNN models took only less than 0.05 s in predicting the states of each batch run, thereby demonstrating dominant computational efficiencies and are thus considered promising alternatives to the PBM for real-time process optimization and control.

Table 2. Computation Time for Solving a Single Batch Run

process model	computation time (s)
first-principles	41.309
RNN	0.049
AERNN	0.032

Automated Machine Learning.

As a powerful black box tool for modeling nonlinear systems, the neural networks generally have to be constructed with suitable architectures and hyperparameters to achieve the desired performance. However, the selection and identification of a good set of these components for a neural network often require tedious episodes of trial and error and have proven to be challenging even for expert practitioners.[43] Recently, automated machine learning (AutoML) has become an increasingly important research topic in enabling non-machine-learning experts to build and deploy machine learning models more efficiently and effortlessly. While there are several AutoML services available, a majority are not free and are cloud-based, requiring complicated configurations.[44] Moreover, data security and privacy are not guaranteed when using these AutoML services that run on large cloud computing platforms. A great alternative is Auto-Keras, which is an easy-to-use, locally accessible, and deployable open-source API specifically designed for deep learning tasks.[44] The use of AutoKeras is especially useful in determining the optimal architectures and hyperparameters of the RNN (e.g., the number of hidden layers, the number of neurons in each hidden layer, and the learning rate) in this work, which can be laborious to do so manually. Since there is a lack of a systematic method for tuning neural network hyperparameters and the manual tuning approach through a grid search is generally time-consuming, AutoML using AutoKeras provides a convenient and accessible tool in facilitating the efficient construction and deployment of neural networks. The AERNN can also be developed using a similar approach to that of the RNN. This section thus introduces the API AutoKeras as an accessible tool in facilitating the efficient construction and deployment of deep learning neural networks-based MPC.

Fundamentally, AutoKeras utilizes the network morphism-based neural architecture search (NAS) algorithm guided by Bayesian optimization. It outperforms many other search algorithms (e.g., NASNet[45] and PNAS[46]) in terms of computational costs and thus provides an efficient means for exploring the large NAS space. The goal of AutoKeras is to find the optimal neural network f^*, given an NAS space $\mathcal{F}$, that minimizes the cost function on an input data set D.[44]

$$f^* = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \operatorname{Cost}(f(\theta^*), D_{\text{val}}) \tag{18}$$

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \operatorname{Cost}(f(\boldsymbol{\theta}), D_{\text{train}}) \tag{19}$$

where the input data D are divided into D_{train} and D_{val}, Cost(·, ·) is the evaluation metric function (e.g., MSE), and $\boldsymbol{\theta}^*$ is the learned parameter of f.

The search algorithm is guided by the Bayesian optimization algorithm, which uses the Bayes theorem to direct its search of the minimum of the cost function. The Bayesian optimization algorithm consists of two main components: a Bayesian statistical model (e.g., a Gaussian process model) for modeling the cost function and an acquisition function for deciding where to sample next.[47] More specifically, the statistical model provides a Bayesian posterior probability distribution that characterizes the potential values of the cost function at a candidate point x. Any new observation of the cost function at a new point is used to update this posterior distribution. Next, the acquisition function measures the plausible values that would be generated by evaluating the cost function at a new point x based on the current posterior distribution, and this guides the selection of subsequent x that can probably further minimize the cost function. These steps are performed iteratively by AutoKeras to identify the optimal neural architecture where the most promising operations are evaluated and selected each time. A typical workflow for running AutoKeras on local computational resources is briefly described as follows:[44] (1) the user initiates a search for the best neural architecture for the data set; (2) API receives the call, preprocesses the data set, and passes it to a search module; (3) the search module utilizes a Bayesian optimizer to generate a new neural architecture using the CPU; (4) a graph module is called to construct the generated architecture in the RAM; (5) the newly built neural architecture is copied to the GPU for training; (6) the trained model is saved in local storage devices (e.g., hard drives); and (7) the performance of the model is input as feedback to the search module to update the Gaussian process and make it ready to generate the next architecture. Interested readers are referred to ref 44 for a detailed discussion of the AutoKeras algorithm and system architecture.

In practice, the implementation of AutoKeras is relatively straightforward, and to better accommodate the needs of users with varying degrees of expertise, two levels of APIs are designed. (1) task-level: users only need to specify the type of learning task of interest (e.g., image classification, regression, etc.) to use the API; (2) search-level: advanced users can preprocess the data set by themselves and indicate a specific type of neural architecture (e.g., multi-layer perceptron, convolutional neural network, etc.) to better facilitate the API in searching the optimal architecture for their tasks. The AutoKeras is thus a comprehensive API that could facilitate the majority of the deep learning tasks by automatically discovering the model architectures and hyperparameters (e.g., number of layers, number of neurons in each layer, dropout rate, type of optimizer, learning rate, etc.). The RNN models in this work are developed using the TimeseriesForecaster class in the API, as illustrated in the following pseudocode (Algorithm 1), where lookback denotes the range of history steps to consider for each prediction (e.g., lookback = 1 means that the model only uses the values at the time step immediately before the time step it is predicting) and predict_until denotes the end point of the prediction for each sample (e.g., predict_until = 1 indicates that the model only makes prediction for one time step based on the current sample). In contrast, the AE can be constructed readily without the aid of the API as it is relatively simple and more accessible (e.g., by gradually increasing the number of neurons in the code layer until no significant improvement in the MSE is observed).

Algorithm 1: Pseudo-code for constructing AERNN using AutoKeras

```
Require: input and output data to consist of reduced process states $\mathbf{x}_r$ and manipulated
    variable $T_j$ at $t_k$ and future states $\mathbf{x}_r$ at $t_{k+1}$, respectively. They should be 2D arrays
    (e.g., (batch_size, num_features)) with equal batch_size, and properly normalized (e.g.,
    values between 0 and 1)
    lookback ← 1
    predict_until ← 1
    model ← TimeseriesForecaster(lookback=lookback, predict_until=predict_until)
    model.fit(x=input_data, y=output_data)
```

■ RNN-BASED PREDICTIVE CONTROL

In this section, we develop machine-learning-based predictive control schemes to optimize a batch crystallization process of FF. Specifically, the predictive control schemes are formulated as real-time optimization problems that compute the optimal manipulated input (jacket temperature) to optimize a number of process performance considerations, for example, product yield, a particular crystal size of interest, energy consumption, and the number of fines in the final product. Additionally, by incorporating AE and RNN models in MPC schemes, the MPC optimization problems achieved the desired control performance with significantly reduced computation time.

MPC Using a Full-Order RNN Model. We first present the formulation of the MPC using a full-order RNN model as follows

$$\min_{u\in S(\Delta)} \int_{t_k}^{t_{k+N}} L(\tilde{\mathbf{x}}, \mathbf{u})\mathrm{d}t \tag{20a}$$

$$\text{s. t. } \dot{\tilde{\mathbf{x}}}(t) = F_{\mathrm{nn}}(\tilde{\mathbf{x}}(t), \mathbf{u}(t)) \tag{20b}$$

$$\tilde{\mathbf{x}}(t_k) = \mathbf{x}(t_k) \tag{20c}$$

$$\mathbf{u}(t) \in U, \forall \ t \in [t_k, t_{k+N}) \tag{20d}$$

$$|\Delta\mathbf{u}| \le U_\Delta, \forall \ t \in [t_k, t_{k+N}) \tag{20e}$$

where $\tilde{\mathbf{x}} = [C, T_r, N_i] \in \mathbf{R}^{42}$, $i = 1, ..., 40$ and $\mathbf{u} = T_j$ are the predicted states and manipulated inputs, respectively. $S(\Delta)$ is the set of piecewise constant functions with the sampling period Δ. $L(\tilde{\mathbf{x}},\mathbf{u}) = Q_1(C(t))^2 - Q_2(N_{\mathrm{class}}(t))^2 + Q_3(T_j(t) - T_j^{\tau})^2$, $\forall\ t < t_f/2$ and $Q_1(C(t))^2 - Q_2(N_{\mathrm{class}}(t))^2 + Q_3(T_j(t) - T_j^{\tau})^2 + Q_4(C(t) - C_s(t))^2$, $\forall\ t \ge t_f/2$ is the objective function that minimizes the solute concentration $C(t)$ (i.e., maximizing product yield), maximizes the number of crystals in a size class of interest N_{class}, minimizes the number of fines in the final product by limiting nucleation, which is driven by supersaturation [i.e., $C(t) - C_s(t)$], in the second half of the batch process (i.e., $t \ge t_f/2$), and minimizes the energy consumption, which is represented as the deviation of the jacket temperature T_j from its value at room temperature ($T_j^{\tau} = 25$ °C) over the prediction horizon $t \in [t_k, t_{k+N})$, where Q_1, Q_2, Q_3, and Q_4 are the coefficients chosen to balance the scale of each term and to indicate the relative importance of one process specification to another.

Specifically, the number of crystals N_{21} in size class 21 (average size = 11.3 μm) and N_{28} in class 28 (average size = 56.6 μm) are chosen as the crystal sizes of interest as we demonstrate the control strategies implemented in maximizing crystals of differing sizes under two different initial conditions. As larger particle sizes are almost always preferred to facilitate downstream processing, such as filtration, the desired crystal size of the final product is usually much larger than that of the seeds. The full-order RNN model of eq 13 is used as the predictive model in eq 20b, and the measurements of process states at every sampling time are used as the initial condition to predict the evolution of the batch crystallizer dynamics (eq 20c). Equation 20d is the constraint on the jacket temperature that reflects the physical limitations on the coolant supply, and eq 20e is the rate-of-change constraint on the jacket temperature to avoid extreme changes in the manipulated inputs. The jacket temperature is bounded by $U = [-10, 30]$ °C, and $|\Delta u| = |T_j(t_{k+1}) - T_j(t_k)|$, $k = 0, 1, 2, ...,$ is bounded by $U_\Delta = 5$ °C.

We assume that online measurements of the state variables (C, T_r, N_i) are available at each sampling time [e.g., using attenuated total reflection Fourier transform infrared (ATR-FTIR) spectroscopy and focused beam reflectance measurement (FBRM) technique]. The RNN-MPC optimization problem of eqs 20a−20e is solved at every sampling time with the new state measurements received by the controller and applies only the first control action $u(t)$, $t \in [t_k, t_{k+1})$ to the crystallizer, which is represented by the first-principles model of eqs 2−10 in this study. Since the RNN model is developed to predict one sampling period Δ, the RNN model prediction is carried out recursively to predict all the future states within the prediction horizon $t \in [t_k, t_{k+N})$. The MPC is solved using PyIpopt, which is a python connector to the IPOPT software package and is well-suited for solving large-scale nonlinear optimization of continuous systems.[48] IPOPT stands for interior-point optimizer and utilizes the interior-point methods in solving linear and nonlinear convex optimization problems. The initial guess of T_j is specified as 30 °C in all optimization problems for the first sampling period, and the initial guesses for all the subsequent optimization are taken to be the optimal values determined in the previous period.

Remark 1 FBRM is one of the most widely employed PAT for online monitoring of crystal sizes and operates on the principle of laser backscattering.[49] The advantages of FBRM include efficient, online, and in situ crystal size measurements, and it is capable of monitoring even concentrated slurries, thereby manifesting itself as an ideal tool for real-time monitoring of pharmaceutical crystallization processes.[50] A major limitation associated with the use of FBRM for determining the CSD is that its output is in the form of particle counts and chord length distribution (CLD), which cannot be readily converted to CSD. However, a substantial number of studies in recent years have sought to address this challenge by proposing various approaches (e.g., iterative non-negative least squares algorithm[51] and empirical methods to establish the relationships between FBRM measurements and CSD[52]) to convert CLD to CSD for different types of particles. In the event that online measurements of CSD or any other process variables are not available, an alternative approach is to estimate these information using the first-principles models that would have been developed based on experimental data, and the first-principles models can be periodically updated and improved using offline measurements of the actual crystallization process.

Remark 2 Note that all the intermediate steps within one RNN prediction step (i.e., all the integration time steps within one sampling period) are utilized in the calculation of the integral of eq 20a. This improves the MPC optimization process since more dynamic information is utilized in the calculation. Compared to the FNNs that predict one time step only, incorporating all the internal steps is one of the benefits of using RNNs in modeling nonlinear dynamic systems.

RNN-MPC Using the Reduced-Order RNN Model. To further reduce the computation time of solving RNN-MPC, we incorporate an AERNN model in the MPC formulation. The AERNN-based MPC (AERNN-MPC) optimization problem is presented as follows

$$\min_{u\in S(\Delta)} \int_{t_k}^{t_{k+N}} L(\tilde{\mathbf{x}}, \mathbf{u})\mathrm{d}t \tag{21a}$$

$$\text{s. t. } \dot{\tilde{\mathbf{x}}}(t) = f_{\mathrm{d}}(F'_{\mathrm{nn}}(\tilde{\mathbf{x}}_r(t), \mathbf{u}(t))) \tag{21b}$$

$$\tilde{\mathbf{x}}_r(t_k) = f_{\mathrm{e}}(\mathbf{x}(t_k)) \tag{21c}$$

$$\mathbf{u}(t) \in U, \forall \ t \in [t_k, t_{k+N}) \tag{21d}$$

$$|\Delta\mathbf{u}| \le U_\Delta, \forall \ t \in [t_k, t_{k+N}) \tag{21e}$$

where the notations follow those in eq 20a−20e. The reduced-order RNN model (i.e., the AERNN model of eq 17) with the decoder $f_{\mathrm{d}}(\cdot)$ is used in the constraint of eq 21b. Specifically, at each sampling time, the MPC receives the state measurements ($x(t_k) = [C, T_r, N_i](t_k)$) and computes the corresponding states $\mathbf{x}_r(t_k)$ in the latent space using the encoder $f_{\mathrm{e}}(\cdot)$ of eq 14 in the constraint of eq 21c. Subsequently, the AERNN model of eq 17 predicts the future states $\mathbf{x}_r(t)$, $t \in [t_k, t_{k+N})$ in the latent space based on $\mathbf{x}_r(t_k)$ and the manipulated input $\mathbf{u}(t)$, and the decoder of eq 15 is utilized to convert $\mathbf{x}_r(t)$ to the states (i.e., $\mathbf{x}(t) = [C, T_r, N_i](t)$, $t \in [t_k, t_{k+N})$) in the original high-dimensional space.

Remark 3 Since the RNN model is developed to predict one sampling period forward and is recursively invoked to predict all the future states within the prediction horizon $t \in [t_k, t_{k+N})$, the computation efficiency is improved by running prediction in the latent space using the reduced-order RNN model. The encoder and decoder functions are utilized to transform the process state measurements to a low-dimensional latent space at the beginning and convert the AERNN predicted states back to the original high-dimensional space at the end, respectively.

■ CLOSED-LOOP SIMULATION RESULTS

A batch crystallization process of FF based on the published kinetics data is used to illustrate the application of the RNN-based MPC of eqs 20a−20e and AERNN-based MPC of eqs 21a−21e with the objectives of maximizing both the product yield and a particular crystal size of interest while avoiding the formation of a large number of fines in the final product and accounting for the energy consumption and the physical constraints of the cooling jacket. Furthermore, to showcase the flexibility of the machine-learning-based MPC framework in meeting various control objectives, different coefficient values are specified for Q_1, Q_2, Q_3, and Q_4 in the objective functions of eqs 20a and 21a, respectively (Table 3). Closed-loop

Table 3. Summary of the Coefficient Values Specified in the Objective Function of MPC

process	Q_1	Q_2	Q_3	Q_4	relative importance
1	10^6	10^{-15}	0.1	10^8	yield > N_{21}
2	10^5	10^{-12}	0.1	10^8	yield < N_{28}

simulations are carried out for the seeded batch cooling crystallization of FF using both RNN-MPC of eqs 20a−20e and AERNN-MPC of eqs 21a−21e. Additionally, the MPC using the first-principles model of 2 is used as a benchmark case for comparison purposes.

The process states $\mathbf{x} = [C, T_r, N_i] \in \mathbf{R}^{42}$, $i = 1, ..., 40$, are the solute concentration C, crystallizer temperature T_r, and the number of crystals in 40 size classes through logarithmic discretization of crystal size, respectively. The manipulated input is the jacket temperature $\mathbf{u} = T_j$. The MPC is implemented in a sample-and-hold fashion, with a sampling time of $\Delta = 30$ min. The closed-loop simulation is performed with two different process initial conditions (Table 4) and control objectives (Table 5).

Table 4. Summary of the Process Conditions Adopted in Closed-Loop Simulation

process	m_{sol} (kg)	m_0 (kg)	m_{seed} (kg)	seed d_{50} (μm)	initial T_r (°C)
1	0.482	0.1	0.005	6.4	30
2	0.482	0.1	0.01	4.2	35

Table 5. Summary of the MPC Objectives Adopted in Closed-Loop Simulation

process	operation time (h)	crystal size to be maximized
1	10	N_{21} (average crystal size: 11.3 μm)
2	10	N_{28} (average crystal size: 56.6 μm)

Figure 8 shows the closed-loop solute concentration profiles (top figure) and the number of crystals N_{21} in size class 21

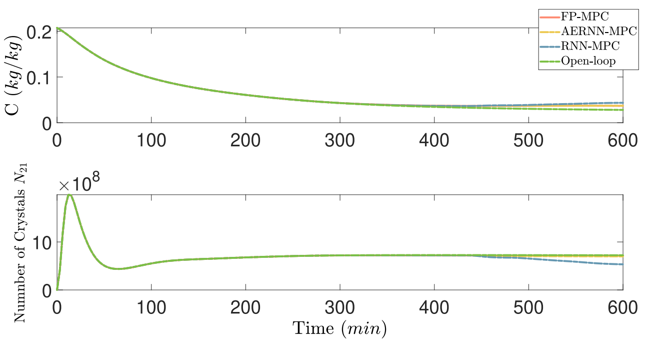


Figure 8. Comparison of closed-loop solute concentration profiles (top figure) and the number of crystals of average size 11.3 μm (bottom figure) under open-loop control (i.e., using maximum cooling after T_j reaches −10 °C) and the MPC using the first-principles model (FP-MPC), the AERNN model (AERNN-MPC), and the RNN model (RNN-MPC), respectively, under process condition 1.

(bottom figure) under process condition 1 with open-loop control and the MPC using the first-principles model (denoted by FP-MPC), the RNN model (denoted by RNN-MPC), and the AERNN model (denoted by AERNN-MPC), respectively. The open-loop control scheme uses the maximum cooling (i.e., $T_j = -10$ °C) after it reaches −10 °C and is subjected to the rate-of-change constraint. It is demonstrated that all MPCs maximize the product yield by decreasing the solute concentration and achieve a desired number of crystals with the size of interest. Overall, the closed-loop performances under both AERNN-MPC and RNN-MPC are very close to that under the FP-MPC, which demonstrates that the RNN-based models provide a sufficiently accurate prediction of the dynamics of the batch crystallizer in the MPC optimization problem as compared to the benchmark case of FP-MPC.

Furthermore, it is noticed in Figure 8 that the number of crystals N_{21} reaches an initial peak before experiencing a descend. This observation can be attributed to the magnitudes of coefficient values Q_{1-4} assigned to the MPCs under process condition 1, which indicates the relative importance of each performance specification in the MPC control objectives (eq 20a for RNN-MPC and eq 21a for AERNN-MPC). In the first half of the crystallization process where time <300 min, a greater emphasis has been placed on product yield [i.e., $Q_1(C(t))^2$ has the largest magnitude]. Ascribed to the addition of seed crystals, which triggers the onset of crystallization and the sharp decrease in crystallizer temperature T_r at this initial stage (Figure 9), crystals that initially clustered in smaller crystal size classes, which correspond to the CSD of seeds and nucleation, begin to grow into larger size classes. With the driving force for crystal growth maximized by the rapid cooling, an incipient surge in N_{21} is observed (Figure 8). As the crystallization process proceeds further, the system starts to establish a trade-off between maximizing product yield and maximizing N_{21}: while further cooling increases product yield, it reduces N_{21} as the number of crystals begins to spread more evenly among the various size classes due to the complex interplay between the various crystallization kinetics (e.g., nucleation, crystal growth, and agglomeration), and this dynamic behavior is also observed in the simulation results

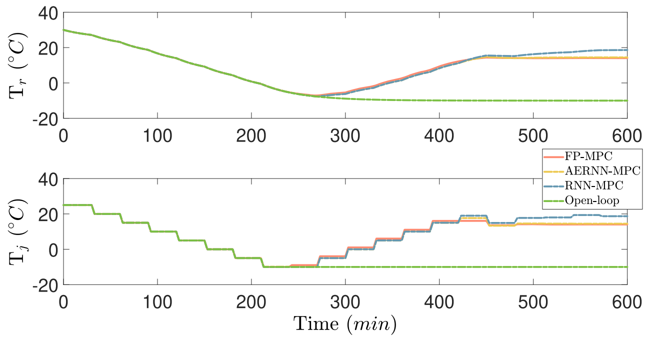


Figure 9. Comparison of closed-loop crystallizer temperature (top figure) and jacket temperature (bottom figure) under open-loop control (i.e., using maximum cooling after T_j reaches −10 °C) and the MPC using the first-principles model (FP-MPC), the AERNN model (AERNN-MPC), and the RNN model (RNN-MPC), respectively, under process condition 1.

reported in ref 34. Since product yield is specified to be relatively more important than N_{21} in the MPC objectives, the controller will cool the crystallization system further to minimize C at the expense of N_{21}, and N_{21} would thus experience a decline. Finally, as observed toward the second half of the crystallization process (i.e., time ≥ 300 min), both C and N_{21} begin to vary slowly (Figure 8), and a new performance specification (i.e., limiting number of fines), which supercedes product yield in terms of relative importance, is incorporated in the MPC objectives (eqs 20a and 21a). At the beginning of the latter half of crystallization (i.e., time ≈ 300 min), the crystallizer temperature T_r attains a value close to its minimum of −10 °C. Through the comparison with the open-loop scheme, it is demonstrated in Figure 8 that no significant improvement in product yield and the number of crystals can be achieved by maintaining the lowest jacket temperature, and the MPC controller thus optimizes the overall performance by increasing T_j to diminish the driving force for nucleation (Figure 10) and to reduce energy consumption (Figure 9). The product yields achieved by the MPCs are 82.1, 79.0, and 82.2% for AERNN-MPC, RNN-MPC, and FP-MPC, respectively, compared to the maximum 86.5% attained with open-loop control.

Similarly, Figure 11 shows the closed-loop crystallizer temperature (top figure) and the jacket temperature profiles

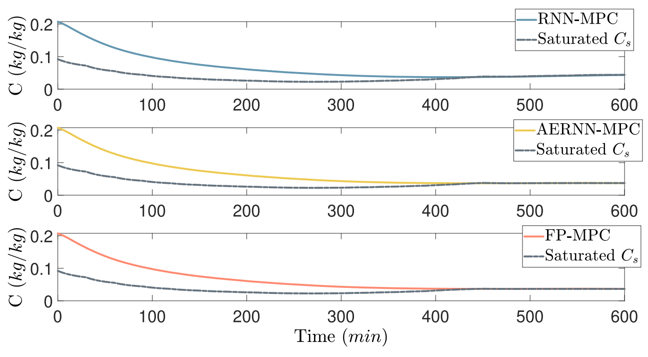


Figure 10. Comparison of closed-loop solute concentration profiles with their corresponding saturated solution concentrations C_s under MPCs using the RNN model (RNN-MPC) (top figure), the AERNN model (AERNN-MPC) (middle figure), and the first-principles model (FP-MPC) (bottom figure), respectively, under process condition 1.

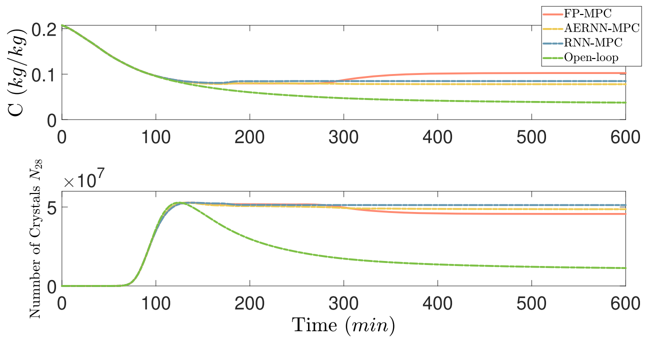


Figure 11. Comparison of closed-loop solute concentration profiles (top figure) and the number of crystals of average size 56.6 μm (bottom figure) under open-loop control (i.e., using maximum cooling after T_j reaches −10 °C) and the MPCs using the first-principles model (FP-MPC), the AERNN model (AERNN-MPC), and the RNN model (RNN-MPC), respectively, under process condition 2.

(bottom figure) under process condition 2 for open-loop control, FP-MPC, RNN-MPC, and AERNN-MPC, respectively. The open-loop control initially decreases T_j and uses T_j = −10 °C for the remaining time of simulation after it reaches the lowest jacket temperature of −10 °C. However, since a different set of coefficient values Q_{1-4} is assigned to the MPCs under process condition 2 (Table 3), disparate control schemes are proposed by the MPCs. In the first half of the crystallization process where time < 300 min, the number of crystals N_{28} in size class 28 has the highest priority ([.e., $Q_2(N_{28}(t))^2$ has the largest magnitude]. While all MPCs decrease the jacket temperature initially, they gradually increase the jacket temperature after around 130 min in order to maximize N_{28} at the expense of product yield (Figure 12). In contrast to the system states observed for process condition 1 (Figure 8), the occurrence of the peak in N_{28} is delayed for process condition 2 (Figure 11). This can be explained by the differing crystal size of interests (i.e., N_{21} vs N_{28}) in both processes. Since N_{21} is in closer proximity to the seed size classes than that of process condition 2, the crystals will take less time to grow to size class N_{21} than to N_{28}, thereby engendering the earlier peak observed in process condition 1. At the second half of crystallization (i.e., after 300 min), the jacket temperature T_j stays relative constant at a value close to its maximum of 30 °C (Figure 12). This phenomenon can be attributed to the attainment of the best possible overall performance toward the latter half of the crystallization process, where both N_{28} and the driving force for nucleation have already been optimized. The product yields achieved by the MPCs are 62.5, 59.5, and 50.6% for AERNN-MPC, RNN-MPC, and FP-MPC, respectively, compared to the maximum 82.0% attained with open-loop control. Therefore, it maybe uneconomical to target an average particle size of 56.6 μm in terms of practical considerations. These observations also agree well with the empirical results reported in ref 34, where a crystal size of approximately 10 μm falls in the range of 80 to 90 percentiles in majority of the experiments conducted, and hence, targeting larger crystals is impractical at a decent yield. Notably, although the MPCs in this work are formulated to optimize the number of crystals in a specific size class of interest, the objective functions eqs 20a and 21a can be adjusted easily in practice to accommodate different perform-

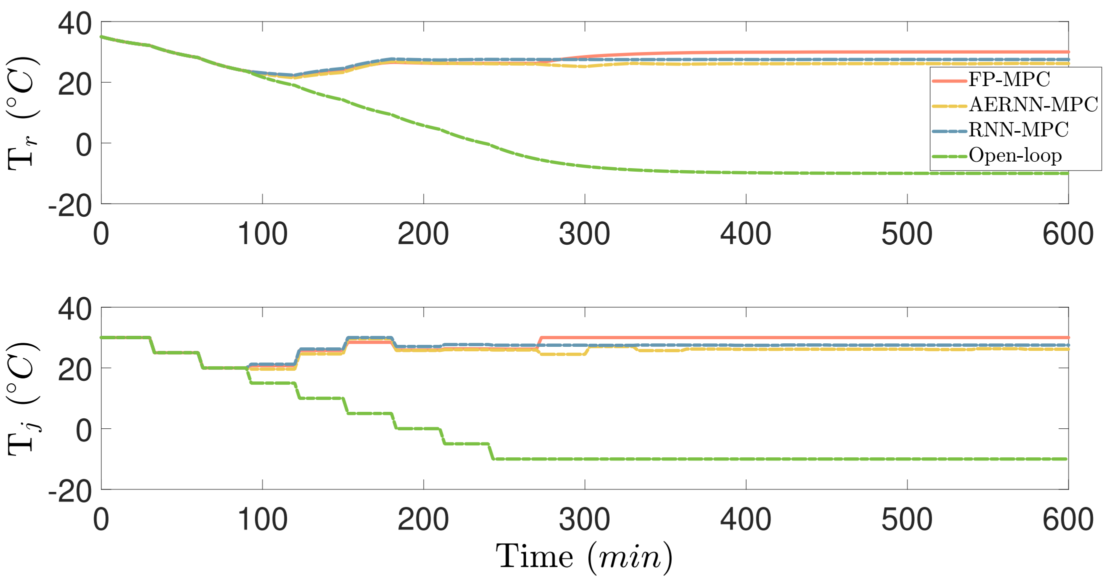


Figure 12. Comparison of closed-loop crystallizer temperature (top figure) and jacket temperature (bottom figure) under open-loop control (i.e., using maximum cooling after T_j reaches −10 °C) and the MPCs using the first-principles model (FP-MPC), the AERNN model (AERNN-MPC), and the RNN model (RNN-MPC), respectively, under process condition 2.

ance specifications (e.g., achieving a particular CSD of interest).

Figure 13 shows the final CSD obtained at the end of the batch simulation with open-loop control and the MPCs using the first-principles model (denoted by FP-MPC), the RNN model (denoted by RNN-MPC), and the AERNN model (denoted by AERNN-MPC), under process conditions 1 (top figure) and 2 (bottom figure), respectively. The final CSDs obtained under process condition 1 are almost identical for FP-MPC and AERNN-MPC, and deviations are observed for those obtained with RNN-MPC and open-loop control. This can be attributed to the difference in closed-loop solute concentration profiles (top figure) observed in Figure 8, which are the key driving force for crystallization and hence the determining factor for the final CSD. The concentration profiles are almost indistinguishable for FP-MPC and AERNN-MPC throughout the batch run, with apparent deviations observed for RNN-MPC and open-loop control after a batch run of 400 min. The final solute concentration achieved at the end of the batch run (i.e., at 600 min) is the highest for RNN-MPC and the lowest for open-loop control. Therefore, in comparison with the final CSDs attained for FP-MPC and AERNN-MPC, the final CSD obtained with RNN-MPC is slightly right-skewed due to a relatively lower driving force for crystal growth and slightly left-skewed for open-loop control due to a relatively higher driving force for crystal growth. The CSDs peaked at a crystal size of approximately 6 μm rather than the desired crystal size of approximately 11.3 μm. This can be mainly ascribed to the formulation of the objective function specified in the MPC optimization problem where the crystal numbers of other size classes were not explicitly controlled. In contrast, the final CSDs obtained under process condition 2 are similar for AERNN-MPC and RNN-MPC, but distinct profiles are observed for FP-MPC and open-loop control. This can be similarly attributed to the disparate closed-loop solute concentration profiles (top figure) observed in Figure 11, with FP-MPC having the highest value and open-loop control with the lowest value toward the latter half of the batch run. The CSDs peaked at a crystal size of approximately 9.5 μm rather than the desired size of approximately 56.6 μm as the objective function specified in the MPC optimization problem does not take into account the number of crystals in other size classes explicitly. It should be noted that formulating an explicit CSD in the MPC objective function requires specific domain knowledge on the plausible CSD achievable by the crystallization system of interest. Furthermore, as only the crystal number of a particular size class of interest and not the detailed CSD is stated in the MPC objective function, the final CSDs attained for FP-MPC, AERNN-MPC, and RNN-MPC are thus not explicitly controlled.

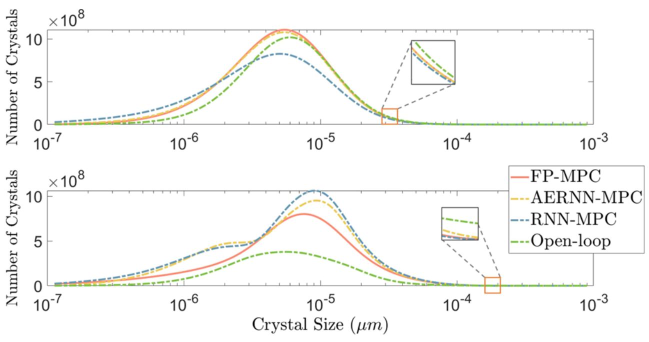


Figure 13. Comparison of the final CSD obtained at the end of the batch simulation with open-loop control (i.e., using maximum cooling after T_j reaches −10 °C) and the MPC using the first-principles model (FP-MPC), the AERNN model (AERNN-MPC), and the RNN model (RNN-MPC), under process conditions 1 (top figure) and 2 (bottom figure), respectively.

Figure 14 shows the optimal values of the objective functions at every sampling period during the batch simulation with the MPCs using the first-principles model (denoted by FP-MPC), the RNN model (denoted by RNN-MPC), and the AERNN model (denoted by AERNN-MPC), under process conditions 1 (top figure) and 2 (bottom figure), respectively. While the optimal values of the objective functions determined by the MPC control scheme under process condition 1 are

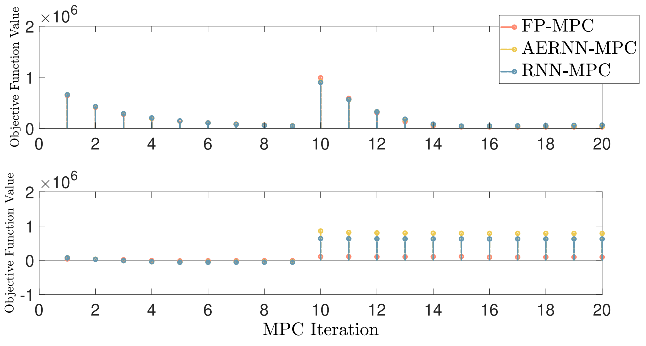


Figure 14. Comparison of the optimal objective function values at every sampling period during the batch run with the MPCs using the first-principles model (FP-MPC), the AERNN model (AERNN-MPC), and the RNN model (RNN-MPC), under process conditions 1 (top figure) and 2 (bottom figure), respectively.

very similar for all controllers, those determined under process condition 2 are considerably different, especially toward the latter half of the batch run. This can be ascribed to the possible nonconvex nature of the optimization problem and hence the existence of several local minima. As the initial guess of the T_j for the current sampling period is taken from the optimal value determined in the previous sampling period, its value may affect the resulting minimum point obtained using the interior-point method. Moreover, due to model mismatch, optimal control action and the initial guess of T_j computed for the current and the next sampling period, respectively, can be different for the three models in each sampling period. This is also evident in the disparate solute concentration profiles (top figure) observed in Figure 11, where the optimal control actions determined by the different controllers start to deviate from one another toward the latter half of the batch simulation. Furthermore, as the values of N_{28} range from 0 to 10^7, the objective function values computed by the MPCs with different models could deviate considerably as even a minor difference between the N_{28} predicted could be of a large magnitude (e.g., in the order of 10^7), and the error is further magnified by specifying maximizing N_{28} as the main objective in the optimization problem of process condition 2 [i.e., $Q_2(N_{28}(t))^2$ having the second largest magnitude in the latter half of the batch run]. Finally, drastic changes in the objective function values are observed between the 9th and 10th sampling periods for both process conditions, and these are attributed to the inclusion of an additional process specification (i.e., minimizing the formation of fines).

As mentioned above, one of the benefits of using RNN models is the computational efficiency of online prediction using the offline-trained RNN model. Therefore, we compare the computation time for solving FP-MPC, RNN-MPC, and AERNN-MPC under the same initial condition using the desktop with Intel Core i7-10700 CPU @ 2.90GHz. Table 6 reports the averaged computation time for solving the MPC optimization problem with different prediction horizons at each sampling step. Since the MPC is solved successively when new state measurements are received at each sampling time with a sampling period of $\Delta = 30$ min, we run the closed-loop simulation for 600 min and calculate the average computation time for solving MPC problems 20 times over the entire simulation period.

Table 6. Computation Time for Solving MPC Once

prediction horizon N	FP-MPC (s)	RNN-MPC (s)	AERNN-MPC (s)
3	575.0	135.3	79.3
5	1963.0	309.1	211.3
10	N/A	1560.6	1397.5

From Table 6, it is shown that the RNN- and AERNN-MPC significantly reduce the computation time as compared to the FP-MPC. Furthermore, as the MPC prediction horizon increases, the computation time for solving the MPC optimization problem increases rapidly. Specifically, for both RNN- and AERNN-MPC, the computation time for solving MPC at each sampling time is rendered to less than one sampling period (i.e., 30 min) for all prediction horizons $N \leq 10$; however, it is noticed that for FP-MPC, the computation time for $N = 5$ has already exceeded 30 min, which makes it infeasible for practical implementation. Also, the FP-MPC with $N = 10$ is unable to obtain the solutions within a reasonable computation time and thus is shown as N/A in Table 6. Since computation time is an important performance metric that determines whether an algorithm can be implemented in practice, it demonstrates the benefits of RNN- and AERNN-MPC in the practical implementation of real-time control of the crystallization process, with AERNN-MPC possessing slightly superior performance. Through the closed-loop simulation study, we demonstrate that both the RNN- and AERNN-MPC are able to accommodate varying performance specifications, achieve desired closed-loop performance in terms of improved product yield, crystal size, number of fines in the final product, and energy consumption, and also significantly reduce the computation time needed for solving the real-time optimization problem as compared to the MPC using first-principles models.

Remark 4 Considering the number of size classes (40) employed in this work, the computational advantage of the AERNN-MPC is relatively minor as compared to that of the RNN-MPC. However, in the event that finer discretized size classes are required (i.e., N_i where $i \gg 40$), the computational efficiency of AERNN-MPC would be more apparent. We have chosen the number of size classes to be 40 in this work to be consistent with the PBM modeling approach proposed in ref 34.

■ CONCLUSIONS

In this work, we developed RNN models for a batch cooling crystallization process of FF using the simulation data from a semi-empirical PBM and designed full-order RNN- and AERNN-MPC to control the batch crystallization process in real time in order to optimize its product yield, crystal size, number of fines in the final product, and energy consumption. Specifically, we first built a PBM based on published kinetic parameters and validated it against the experimental data reported. With insufficient experimental data for constructing robust machine learning models, this semi-empirical model was used to create a rich data set for RNN training by running extensive open-loop simulations under various operating conditions. The open-loop simulation results demonstrated that the full-order RNN model achieved a desired accuracy in predicting future states with much less computation time. To further enhance its computation efficiency, an AERNN model is developed using input data of reduced dimensions, where it achieved a similar prediction performance to that of the full-order RNN model. Subsequently, both the RNN and AERNN models were utilized to predict the dynamics of the batch crystallizer in MPC. It was demonstrated in closed-loop simulations that both the RNN- and AERNN-MPC are formulated with great flexibility, where varying objective functions can be accommodated by assigning different coefficient values to the various performance specifications. As a result, both MPCs are capable of achieving the desired product yield and crystal size while avoiding the formation of a large number of fines in the final product and reducing energy consumption simultaneously. Additionally, the computational efficiency of solving the real-time control problem was significantly improved through the use of RNN and AERNN models in MPC, with the AERNN model having slightly superior performance. The proposed AERNN-MPC framework in this work can be conveniently generalized to most batch crystallization processes as long as the system states can be obtained accurately and periodically throughout the batch run. Furthermore, given the data-driven nature of the AERNN modeling approach proposed, it can be constructed directly from real-world plant operational data even without a sufficiently detailed mechanistic understanding of the process of interest, thereby potentially transforming real-time optimal control from a vision into reality.

AUTHOR INFORMATION

Corresponding Authors

Xiaonan Wang − *Department of Chemical and Biomolecular Engineering, National University of Singapore, Singapore 117585, Singapore; Department of Chemical Engineering, Tsinghua University, Beijing 100084, China;* orcid.org/0000-0001-9775-2417; Email: wangxiaonan@mail.tsinghua.edu.cn

Zhe Wu − *Department of Chemical and Biomolecular Engineering, National University of Singapore, Singapore 117585, Singapore;* orcid.org/0000-0002-2923-149X; Email: wuzhe@nus.edu.sg

Author

Yingzhe Zheng − *Department of Chemical and Biomolecular Engineering, National University of Singapore, Singapore 117585, Singapore*

Complete contact information is available at: https://pubs.acs.org/10.1021/acs.iecr.2c00026

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

Financial support from Pfizer Asia Manufacturing Pte. Ltd. under its project "Data-Driven Knowledge Discovery In Pharmaceutical Manufacturing Through Batch Process Optimization" (grant no.: R-279-000-624-592) and the NUS Start-up Grant (grant no.: R-279-000-656-731) is gratefully acknowledged.

REFERENCES

(1) Alvarez, A. J.; Myerson, A. S. Continuous plug flow crystallization of pharmaceutical compounds. *Cryst. Growth Des.* **2010**, *10*, 2219−2228.

(2) Arden, N. S.; Fisher, A. C.; Tyner, K.; Yu, L. X.; Lee, S. L.; Kopcha, M. Industry 4.0 for pharmaceutical manufacturing: Preparing for the smart factories of the future. *Int. J. Pharm.* **2021**, *602*, 120554.

(3) Simone, E.; Zhang, W.; Nagy, Z. K. Application of Process Analytical Technology-Based Feedback Control Strategies To Improve Purity and Size Distribution in Biopharmaceutical Crystallization. *Cryst. Growth Des.* **2015**, *15*, 2908−2919.

(4) Su, Q.; Ganesh, S.; Moreno, M.; Bommireddy, Y.; Gonzalez, M.; Reklaitis, G. V.; Nagy, Z. K. A perspective on Quality-by-Control (QbC) in pharmaceutical continuous manufacturing. *Comput. Chem. Eng.* **2019**, *125*, 216−231.

(5) Su, Q.; Ganesh, S.; Reklaitis, G. V.; Nagy, Z. K. *Continuous Pharmaceutical Processing*; Springer, 2020; pp 395−427.

(6) Xu, S.; Chen, Y.; Gong, J.; Wang, J. Interplay between Kinetics and Thermodynamics on the Probability Nucleation Rate of a Urea-Water Crystallization System. *Cryst. Growth Des.* **2018**, *18*, 2305−2315.

(7) Ochi, Y.; Cai, Z.; Horie, T.; Komoda, Y.; Tung, K.-L.; Ohmura, N. Representative shear rate for particle agglomeration in a mixing tank. *Chem. Eng. Res. Des.* **2021**, *171*, 73−79.

(8) Shi, D.; Mhaskar, P.; El-Farra, N. H.; Christofides, P. D. Predictive control of crystal size distribution in protein crystallization. *Nanotechnology* **2005**, *16*, S562.

(9) Kwon, J. S.-I.; Nayhouse, M.; Christofides, P. D.; Orkoulas, G. Modeling and control of protein crystal shape and size in batch crystallization. *AIChE J.* **2013**, *59*, 2317−2327.

(10) Yang, Y.; Pal, K.; Koswara, A.; Sun, Q.; Zhang, Y.; Quon, J.; McKeown, R.; Goss, C.; Nagy, Z. K. Application of feedback control and in situ milling to improve particle size and shape in the crystallization of a slow growing needle-like active pharmaceutical ingredient. *Int. J. Pharm.* **2017**, *533*, 49−61.

(11) Cao, Y.; Acevedo, D.; Nagy, Z. K.; Laird, C. D. Real-time feasible multi-objective optimization based nonlinear model predictive control of particle size and shape in a batch crystallization process. *Control Eng. Pract.* **2017**, *69*, 1−8.

(12) Marcellos, C. F. C.; Durand, H.; Kwon, J. S.-I.; Barreto, A. G., Jr.; da Cunha Lage, P. L.; de Souza, M. B., Jr.; Secchi, A. R.; Christofides, P. D. Optimal enantiomer crystallization operation using ternary diagram information. *Comput. Aided Chem. Eng.* **2018**, *44*, 499−504.

(13) de Moraes, M. G. F.; de Souza, M. B., Jr.; Secchi, A. R. Dynamics and MPC of an evaporative continuous crystallization process. *Comput. Aided Chem. Eng.* **2018**, *43*, 997−1002.

(14) Liu, Y. C.; Acevedo, D.; Yang, X.; Naimi, S.; Wu, W.-L.; Pavurala, N.; Nagy, Z. K.; O'Connor, T. F. Population balance model development verification and validation of cooling crystallization of carbamazepine. *Cryst. Growth Des.* **2020**, *20*, 5235−5250.

(15) Szilágyi, B.; Agachi, P. Ş.; Nagy, Z. K. Chord length distribution based modeling and adaptive model predictive control of batch crystallization processes using high fidelity full population balance models. *Ind. Eng. Chem. Res.* **2018**, *57*, 3320−3332.

(16) Shi, D.; El-Farra, N. H.; Li, M.; Mhaskar, P.; Christofides, P. D. Predictive control of particle size distribution in particulate processes. *Chem. Eng. Sci.* **2006**, *61*, 268−281.

(17) Liu, J.; Muñoz de la Peña, D.; Christofides, P. D.; Davis, J. F. Lyapunov-based Model Predictive Control of Particulate Processes Subject to Asynchronous Measurements. *Part. Part. Syst. Char.* **2008**, *25*, 360−375.

(18) Kwon, J. S.-I.; Nayhouse, M.; Christofides, P. D.; Orkoulas, G. Protein crystal shape and size control in batch crystallization: Comparing model predictive control with conventional operating policies. *Ind. Eng. Chem. Res.* **2014**, *53*, 5002−5014.

(19) Griffin, D. J.; Grover, M. A.; Kawajiri, Y.; Rousseau, R. W. Data-driven modeling and dynamic programming applied to batch cooling crystallization. *Ind. Eng. Chem. Res.* **2016**, *55*, 1361−1372.

(20) Damour, C.; Benne, M.; Grondin-Perez, B.; Chabriat, J.-P. Nonlinear predictive control based on artificial neural network model for industrial crystallization. *J. Food Eng.* **2010**, *99*, 225−231.

(21) Rohani, S.; Haeri, M.; Wood, H. C. Modeling and control of a continuous crystallization process Part 2. Model predictive control. *Comput. Chem. Eng.* **1999**, *23*, 279−286.

(22) Velásco-Mejía, A.; Vallejo-Becerra, V.; Chávez-Ramírez, A. U.; Torres-González, J.; Reyes-Vidal, Y.; Castañeda-Zaldivar, F. Modeling and optimization of a pharmaceutical crystallization process by using

neural networks and genetic algorithms. *Powder Technol.* **2016**, *292*, 122−128.

(23) Öner, M.; Montes, F. C. C.; Ståhlberg, T.; Stocks, S. M.; Bajtner, J. E.; Sin, G. Comprehensive evaluation of a data driven control strategy: Experimental application to a pharmaceutical crystallization process. *Chem. Eng. Res. Des.* **2020**, *163*, 248−261.

(24) Wu, Z.; Tran, A.; Ren, Y. M.; Barnes, C. S.; Chen, S.; Christofides, P. D. 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.

(25) Wu, Z.; Tran, A.; Rincon, D.; Christofides, P. D. Machine Learning-Based Predictive Control of Nonlinear Processes. Part II: Computational Implementation. *AIChE J.* **2019**, *65*, No. e16734.

(26) Wu, Z.; Christofides, P. D. Control Lyapunov-barrier function-based predictive control of nonlinear processes using machine learning modeling. *Comput. Chem. Eng.* **2020**, *134*, 106706.

(27) Wu, Z.; Rincon, D.; Christofides, P. D. Process structure-based recurrent neural network modeling for model predictive control of nonlinear processes. *J. Process Control* **2020**, *89*, 74−84.

(28) Benyahia, B.; Anandan, P. D.; Rielly, C. Control of Batch and Continuous Crystallization Processes using Reinforcement Learning. *Comput. Aided Chem. Eng.* **2021**, *50*, 1371−1376.

(29) Qing, X.; Jin, J.; Niu, Y.; Zhao, S. Time−space coupled learning method for model reduction of distributed parameter systems with encoder-decoder and RNN. *AIChE J.* **2020**, *66*, No. e16251.

(30) Wang, W.; Zhao, M.; Wang, J. Effective android malware detection with a hybrid model based on deep autoencoder and convolutional neural network. *J. Ambient Intell. Hum. Comput.* **2019**, *10*, 3035−3043.

(31) Mallick, P. K.; Ryu, S. H.; Satapathy, S. K.; Mishra, S.; Nguyen, G. N.; Tiwari, P. Brain MRI image classification for cancer detection using deep wavelet autoencoder-based deep neural network. *IEEE Access* **2019**, *7*, 46278−46287.

(32) Chen, M.; Shi, X.; Zhang, Y.; Wu, D.; Guizani, M. Deep feature learning for medical image analysis with convolutional autoencoder neural network. *IEEE Trans. Big Data* **2021**, *7*, 750−758.

(33) Zhao, T.; Zheng, Y.; Gong, J.; Wu, Z. Machine Learning-Based Reduced-Order Modeling and Predictive Control of Nonlinear Processes. *Chem. Eng. Res. Des.* **2022**, *179*, 435−451.

(34) Trampuž, M.; Teslić, D.; Likozar, B. Crystallization of fesoterodine fumarate active pharmaceutical ingredient: Modelling of thermodynamic equilibrium, nucleation, growth, agglomeration and dissolution kinetics and temperature cycling. *Chem. Eng. Sci.* **2019**, *201*, 97−111.

(35) Michel, M. C. Fesoterodine: a novel muscarinic receptor antagonist for the treatment of overactive bladder syndrome. *Expet Opin. Pharmacother.* **2008**, *9*, 1787−1796.

(36) Kerekes, P. Crystalline forms of fesoterodine fumarate and fesoterodine base. US Patent 20,120,220,655 A1, 2012.

(37) Marchal, P.; David, R.; Klein, J. P.; Villermaux, J. Crystallization and precipitation engineering-I. An efficient method for solving population balance in crystallization with agglomeration. *Chem. Eng. Sci.* **1988**, *43*, 59−67.

(38) Wu, Z.; Tran, A.; Rincon, D.; Christofides, P. D. Machine Learning-Based Predictive Control of Nonlinear Processes. Part I: Theory. *AIChE J.* **2019**, *65*, No. e16729.

(39) Schäfer, A. M.; Zimmermann, H. G. Recurrent neural networks are universal approximators. *International Conference on Artificial Neural Networks*, 2006; pp 632−640.

(40) Xiu, X.; Yang, Y.; Kong, L.; Liu, W. Laplacian regularized robust principal component analysis for process monitoring. *J. Process Control* **2020**, *92*, 212−219.

(41) Cui, W.; Li, Y.; Sun, Z. A Parallel Computer Numerical Simulation Method Based on Coincident Coefficients. *J. Phys. Conf.* **2020**, *1486*, 042038.

(42) Chollet, F.; et al. *Keras*, 2015. https://keras.io (accessed Dec 16, 2021).

(43) Hutter, F.; Kotthoff, L.; Vanschoren, J. *Automated Machine Learning: Methods, Systems, Challenges*; Springer Nature, 2019.

(44) Jin, H.; Song, Q.; Hu, X. Auto-keras: An efficient neural architecture search system. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019; pp 1946−1956.

(45) Zoph, B.; Le, Q. V. Neural architecture search with reinforcement learning. 2016, arXiv:1611.01578. https://arxiv.org/pdf/1611.01578.pdf (accessed Dec 16, 2021).

(46) Liu, C.; Zoph, B.; Neumann, M.; Shlens, J.; Hua, W.; Li, L.-J.; Fei-Fei, L.; Yuille, A.; Huang, J.; Murphy, K. Progressive neural architecture search. *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018; pp 19−35.

(47) Frazier, P. I. A tutorial on Bayesian optimization. 2018, arXiv:1807.02811 https://arxiv.org/pdf/1807.02811.pdf (accessed Dec 16, 2021).

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

(49) Heinrich, J.; Ulrich, J. Application of Laser-Backscattering Instruments for In Situ Monitoring of Crystallization Processes−A Review. *Chem. Eng. Technol.* **2012**, *35*, 967−979.

(50) Leyssens, T.; Baudry, C.; Escudero Hernandez, M. L. Optimization of a crystallization by online FBRM analysis of needle-shaped crystals. *Org. Process Res. Dev.* **2011**, *15*, 413−426.

(51) Li, M.; Wilkinson, D.; Patchigolla, K. Comparison of particle size distributions measured using different techniques. *Part. Sci. Technol.* **2005**, *23*, 265−284.

(52) Li, H.; Grover, M. A.; Kawajiri, Y.; Rousseau, R. W. Development of an empirical method relating crystal size distributions and FBRM measurements. *Chem. Eng. Sci.* **2013**, *89*, 142−151.