Hindawi
Advances in Polymer Technology
Volume 2022, Article ID 7662264, 14 pages
https://doi.org/10.1155/2022/7662264

Research Article

Deep Learning-Based Predictive Control of Injection Velocity in Injection Molding Machines

Zhigang Ren,[1,2] Yaodong Li,[1,2] Zongze Wu,[3] and Shengli Xie[1,2,4]

[1]*School of Automation and Guangdong Key Laboratory of IoT Information Technology, Guangdong University of Technology, Guangzhou 510006, China*
[2]*Guangdong-Hong Kong-Macao Joint Laboratory for Smart Discrete Manufacturing, Guangzhou 510006, China*
[3]*College of Mechatronics and Control Engineering, Shenzhen University, Shenzhen 518052, China*
[4]*Key Laboratory of Intelligent Detection and The Internet of Things in Manufacturing, Ministry of Education, Guangzhou 510006, China*

Correspondence should be addressed to Zongze Wu; zzwu@gdut.edu.cn

Received 20 April 2022; Revised 24 October 2022; Accepted 26 October 2022; Published 15 November 2022

Academic Editor: Gyorgy Szekely

Rapid and reliable optimal control of injection molding machines (IMMs) is critical for the effective production of injection-molded goods, especially in the situation of restricted computer resources of embedded equipment in IMMs. In this paper, an optimal tracking injection velocity control problem arising in a typical IMM is studied. An effective hybrid intelligent control approach with less computing resources for real-time implementation based on the deep learning (DL) method to mimic the classical model predictive control rule is developed to deal with the tracking control of the injection speed. The proposed method utilizes the gated recurrent unit neural network to learn and predict the optimal time series control process data produced by the traditional model predictive controller. The benefits of this approach over the conventional optimization method are illustrated through simulation results, which show that the convergent DL-based controller can effectively avoid the complex calculation in the control process of IMMs and meet the requirements of more robustness and resist environmental uncertainty to a certain level and can be potentially implemented in embedded hardware much more efficiently and conveniently with a smaller memory footprint and faster computation time.

1. Introduction

Plastic products have become an integral element of all industries and our everyday lives all over the world due to their outstanding flexibility, good durability, low cost, and other advantages [1]. In the field of discrete manufacturing, IMMs are professional operating equipment used to produce plastic components and other polymers. Nearly 70% of plastic products are produced by IMMs, which has become an important field of aerospace, national defense, electronics, and photoelectric communications. They provide important equipment support for high-end manufacturing such as aerospace, power electronics, clean energy, and chip production.

The injection molding process in IMMs is a complex processing technology to make resin materials into plastic products. In this process, the parameter setting of IMMs is very important and closely related to the final product. Any improper changes of the parameter will affect the final quality of the product and produce various defects [1–3]. The injection molding process generally includes six stages: mold clamping, injection, pressure holding, plasticization, cooling, and mold opening. Injection molding is a cyclic process that each process includes above all stages. The simplified structure of a typical IMM is shown in Figure 1. Plastic granules are fed into the IMM through the hopper and melted by friction heating through a heating belt in the barrel. When the molten resin is uniformly heated to the appropriate temperature and fills the material area in the injection unit, the injection and filling can be started. The servo valve causes the reciprocating screw in the injection barrel to move swiftly, allowing the resin material to be injected via the nozzle into the closed metal mold. The screw is then kept

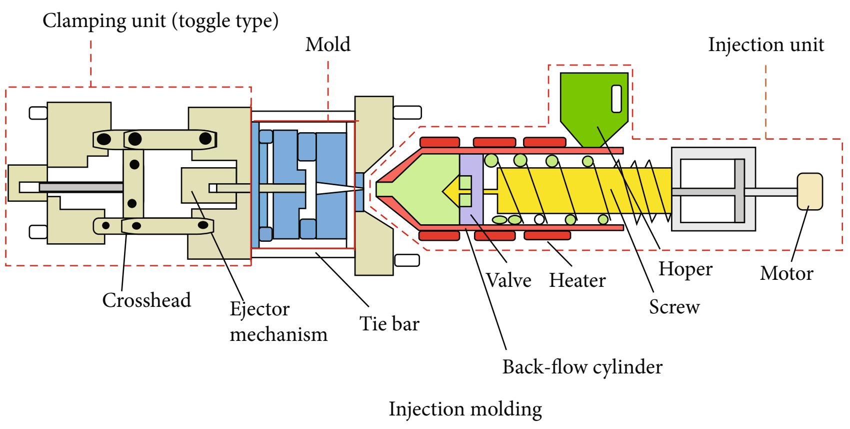


FIGURE 1: A typical simplified structure of an injection molding machine.

at constant pressure, and more molten resin is injected into the mold to cover the gap left by the resin's cooling and shrinkage. When the resin has cooled and completely solidified, the pressure held on the screw is removed. The components are released from the mold with the push of the demolding mechanism and fall into the collecting box once the mold is opened. This process is not only a cyclic process but also a complex parameter change process. For the production process of injection-molded parts, how to achieve precise control of system variables in each production stage is an important prerequisite for ensuring product quality. At different stages, there are also different requirements for parameters of the IMM. As the high-end manufacturing industry puts forward higher and higher requirements for the quality of plastic products, the traditional IMM control technology is gradually unable to meet the requirements in terms of accuracy and efficiency, especially the traditional PID-based control strategies, which have high requirements on the technology of the engineers, and is not easy to resolve problems in production as quickly as possible.

In recent years, there have been many related researches on control technologies in the field of the injection molding process [3–12]. For example, Tan et al. proposed an adaptive control approach for controlling injection speed changes throughout the injection molding process [4]. The strategy shows better performance than the traditional PID control strategy, which can adaptively adjust the mathematical model to deal with parameter changes during the sudden changes in speed. Gao et al. proposed an iterative learning control theory-based robust IMM control approach with flawless performance for systems with unknown initiation and disturbances [5]. Hopmann et al. proposed an iterative learning control system in which the reference tracking of the cavity pressure can be improved over several cycles and recurring disturbances can be automatically corrected to improve component quality even further [7]. With the development of more and more related researches, MPC-based approaches have also been widely employed in the injection molding process control [13–19]. Dubay used a self-optimizing MPC strategy to control the melt temperature in injection molding [17]. Furthermore, Reiter et al. proposed an MPC-based control method based on the IMM physical gray box model and realized the cavity pressure control of the IMM [18]. Hopmann et al. proposed an MPC strategy based on the self-optimizing injection molding process to control the cavity pressure of the IMM and the adaptive adjustment of process parameters [19]. MPC is a mature technology and has become the standard approach for implementing constrained, multivariable control in the process industries today [20]. However, there are still some problems that limit the applications of MPC, especially in the real-time control of complex systems. In the actual control process, due to the complex model or complex constraints, the calculation of MPC is sometimes time-consuming and it is difficult to realize real-time control or have high requirements for equipment. The MPC implementation for IMM systems with limited onboard processing capability, high sampling rates, and strong dynamics is still a difficult task [21], especially in the related embedded devices in the IMM equipment.

With the quick growth of artificial intelligence theories such as deep learning (DL) and machine learning (ML) [22–33], deep neural networks (DNNs) have shown great abilities in various fields [34–37]. In addition to computer vision and natural language processing, DL has broad applications in the control area, particularly in the field of high control accuracy, where some great results have emerged through the combination of DL and control approaches [23, 25, 30, 31, 38]. The powerful learning ability of DNNs can often reproduce complex control processes well, simplify the application of the control theory, and enable it to be quickly deployed in complex models. Existing researches show that DL has better effects on the effective representation and approximation of predictive control laws [39–41]. In this paper, we study how to combine DL with traditional MPC to produce a high-performance predictive controller that is simple in design and more time efficient, while providing the concept of constraint satisfaction to ensure safety. The main purpose is to overcome the shortcomings of traditional MPC in solving efficiency and make full use of the advantages of deep learning networks to realize the design of a data-driven real-time optimal controller. To this end, we propose a DL-based predictive controller which can be applied to the optimal speed curve tracking control problem in a typical IMM system. A gated recurrent unit (GRU) network is designed to study the predictive control process of the controller, so as to replace the original model predictive controller and avoid the complex calculation and save running time brought by the traditional optimization process. The constructed GRU-based neural network module can play the same role as the original MPC controller by learning a large amount of time sequence data, allowing the GRU neural network module to follow changes in the target output and perform optimal control as a result. Furthermore, the problem of long computation time caused by repetitive recursive calculation and rolling optimization in the original model predictive control can be effectively avoided by using the GRU network module, allowing for real-time optimal control of injection molding machines with various tracking output targets. Experimental results also show that our designed DL-based predictive controller is able to accurately approximate the control law with a small memory footprint, very simple implementation, and low computational complexity. In the field of injection molding, particularly in the field of IMM control, this strategy can be quickly deployed in the embedded controller of the IMM without high requirements for equipment hardware performance and has a good application prospect.

The rest of this paper is organized as follows. In Section 2, we introduce the problem description and constraints of injection speed control of an IMM. In Section 3, we propose a DL-based MPC for the injection molding velocity control. In Section 4, numerical experimental results are taken to verify the feasibility of the proposed method. Finally, we conclude this paper in Section 5.

2. Problem Formulation

This section mainly provides a brief introduction on mathematical model of ram velocity, initial state, constraints, and cost function in IMM. We formulate the control task of ram velocity as a dynamic optimal control problem.

2.1. Mathematical Modeling. As a complex control process, resin temperature, cavity pressure, and ram velocity are the key process control parameters that are closely related to the final quality of plastic products [42]. Therefore, many mathematical models describing the changes in these parameters have also been proposed and one of important research directions is the velocity control. During this process, the molten plastic is injected and filled into the mold cavity at a certain speed driven by the control signal. The screw speed control is a process that changes with time, and the injection velocity profile has several zones, as shown in Figure 2. In different zones, the ram speed needs to reach the corresponding value to ensure that the parts meet the production requirements [43]. When the speed curve is set unreasonable, it may cause defects in the plastic parts.

Because the value of injection velocity needs to change with the change of reference trajectory, the control accuracy of the controller is required to be higher. In this paper, we adopt the following fourth-order transfer function model

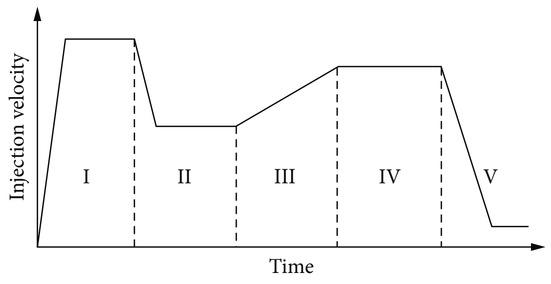


FIGURE 2: Injection velocity profile during the injection molding.

[44] as our control model which is given by relating the valve opening to the injection ram velocity:

$$G(s) = \frac{p}{(s+s_1)(s+s_2)\left[(s+s_3)^2+s_4^2\right]}, \tag{1}$$

where p, s_1, s_2, s_3, and s_4 are configuration parameters that can be set according to different IMM systems. By transforming the above transfer function and discretizing the state equation with the zero-order preservation method, the dynamic model of the studied IMM system can be represented as the following discrete-time state equations:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + D\xi(k), \\ y(k) &= Cx(k), \end{aligned} \tag{2}$$

where $A \in \mathbb{R}^{n_x \times n_x}$, $B \in \mathbb{R}^{n_x \times n_u}$, $C \in \mathbb{R}^{n_y \times n_x}$, and $D \in \mathbb{R}^{n_x \times n_\xi}$ are parameter matrices of the ram speed model in the IMM. $x(k) \in \mathbb{R}^{n_x \times 1}$ is the state vector, $y(k) \in \mathbb{R}^{n_y \times 1}$ is the IMM injection speed measured by the sensor, $u(k) \in \mathbb{R}^{n_u \times 1}$ is the controlling variable which is proportional to the input signal of the servo valve, $\xi(k) \in \mathbb{R}^{n_\xi \times 1}$ is an uncorrelated random sequence representing the process noise, and k is the control time. In this paper, $\xi(k)$ is a Gaussian noise sequence with zero mean and random variance in the range of $[\epsilon, \sigma^2]$. In practice for the IMM, the future noise is difficult to predict; thus, we take the following discrete-time model as the control object:

$$\begin{aligned} x(k+1) &= Ax(k) + Bu(k) + K\widehat{\xi}(k), \\ y(k) &= Cx(k), \end{aligned} \tag{3}$$

where $\widehat{\xi}(k)$ is introduced to describe the model errors along the time horizon k and K is the Kalman filter gain. In sampling time k, we take the error $\widehat{\xi}(k) = y(k) - Cx(k)$. As indicated in [13], if the mathematical model is stable, the observer gain K could be regarded as D. The control objective of the model is to calculate a control trajectory u so that the controlled ram speed y follows the predetermined trajectory y_{ref} as much as possible and minimizes the error between them.

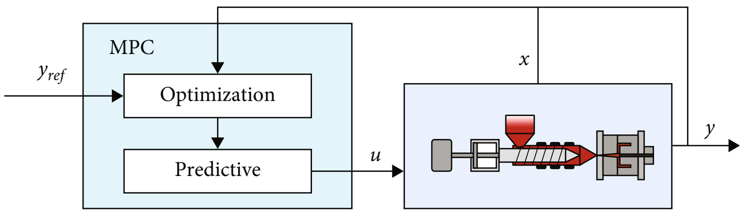


Figure 3: Schematic of the control process in IMM based on MPC.

2.2. Initial State and Constraints. In the actual control process, the initial state of the injection molding is generally slightly different. Most of these differences are caused by external disturbances or noise. To simplify the calculation, we assume that there is no noise before the system works. The initial states in IMM at starting time $k = 0$ are denoted as follows:

$$S(0) = [x(0), y(0), u(0)], \tag{4}$$

where all the initial values of $S(0)$ are set 0. In the IMM, the ram velocity $y(k)$ and the controlling variable $u(k)$ are limited by the machine. Due to the actual physical limitation, $y(k)$ and $u(k)$ in equations (2) and (3) should satisfy the following bound constraints:

$$\begin{aligned} 0 \le y(k) \le y_{\max}, \\ 0 \le u(k) \le u_{\max}, \end{aligned} \tag{5}$$

where $y_{\max}$ and $u_{\max}$ are the given maximum constants of ram velocity and control variable. During the injection molding process, $u(k)$ and $y(k)$ need to meet the above constraints and change within this interval. Through combining the mathematical model (3) and constraints (5), then, we can apply the MPC method to ram speed control in IMM.

2.3. Objective Index. In this paper, our main goal is to control the ram velocity of the IMM system as quickly as possible to converge to the specified target value. In this work, we define the following objective index as the control target:

$$\min_{u(k)} J = \sum_{k=1}^{p} (y(k) - y_{\text{ref}}(k))^T Q (y(k) - y_{\text{ref}}(k)) + \sum_{k=1}^{n} \Delta u(k)^T R \Delta u(k), \tag{6}$$

where $y_{\text{ref}}(k)$, $y(k)$, and $\Delta u(k)$ are the reference tracking output, output of the plant, and control input increment at the discrete time k, respectively. The control input increment satisfies $\Delta u(k) = u(k) - u(k-1)$. Q is the weight matrix of y, and R is the weight matrix of $\Delta u(k)$. The precise control of the dynamic model can finally be attained by minimizing the value of the loss function (8), and the ram velocity $y(k)$ can also quickly converges to the stated target velocity $y_{\text{ref}}(k)$.

3. Deep Learning-Based Predictive Control

In this section, we will introduce our proposed control strategy in three parts: model predictive control, dataset construction, and deep learning architecture design and optimization. First, we will introduce how to use MPC to solve the speed control problem. Second, we will introduce our design idea of the DL network and optimization strategies. Finally, we will introduce how to build an MPC-based dataset and connect the above two parts to form a closed loop.

3.1. Model Predictive Control. MPC is a dynamic optimization approach for determining the optimal control input and output trajectory. The optimality of the prediction vector can be achieved by solving optimization problems in the prediction and control time domains in a cyclical manner. The QP problem is a popular standard type in the MPC. In Figure 3, we illustrate the schematic of the control process in the IMM based on the MPC strategy. By inputting a given reference value y_{ref} to the MPC controller, the controller optimally calculates the optimal control value $u(k)$ at the next moment in combination with the current state variables. The key idea of linear MPC is to predict the trajectory of parameters in prediction horizon through the combination of the current state vector and the mathematical model. Based on MPC, the dynamic model (3) can be reformulated as follows:

$$\begin{aligned} (k+1) &= Ax(k) + Bu(k) + K\hat{\xi}(k), \\ x(k+2) &= A^2 x(k) + ABu(k) + Bu(k+1) \\ &\quad + AK\hat{\xi}(k) + K\hat{\xi}(k+1), \\ &\vdots \\ x(k+p) &= A^p x(k) + A^{p-1} Bu(k) \\ &\quad + A^{p-2} Bu(k+1) + \cdots + Bu(k+p-1) \\ &\quad + A^{p-1} K\hat{\xi}(k) + A^{p-2} K\hat{\xi}(k+1) \\ &\quad + \cdots + K\hat{\xi}(k+p-1), \end{aligned} \tag{7}$$

where p denotes the prediction horizon and n denotes the control horizon.

Combining all the above formulas, we can get the following optimal control problem with constraints for the ram velocity control in IMMs as follows:

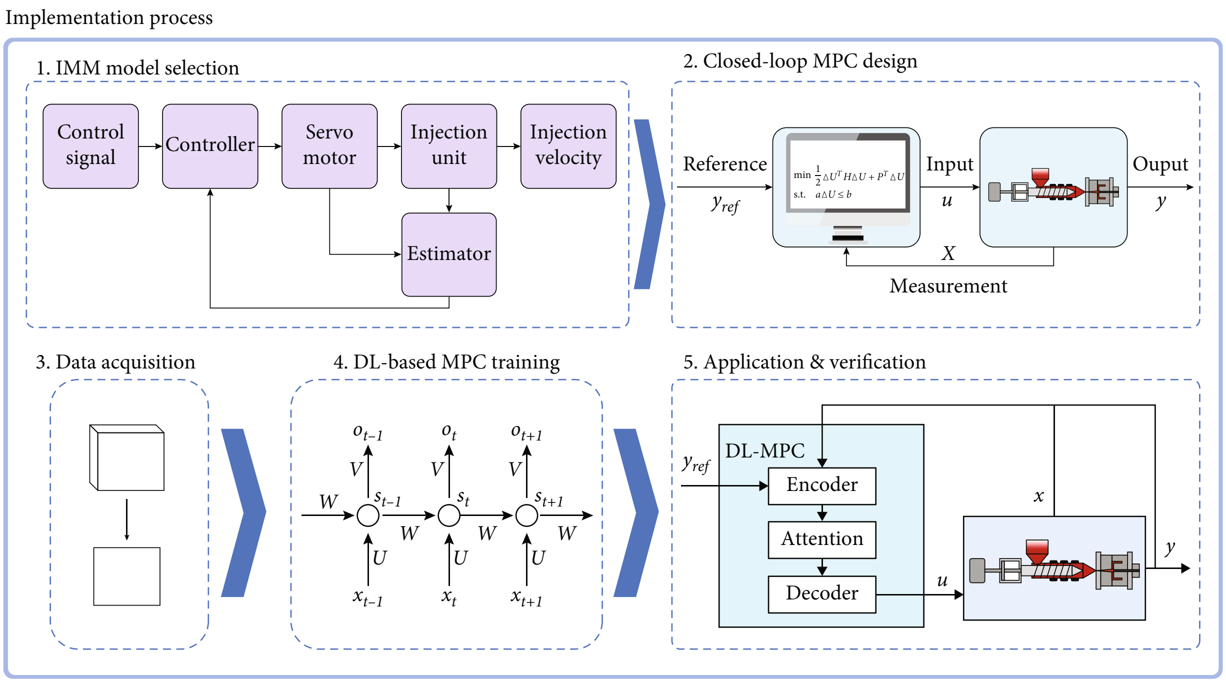


Figure 4: The flowchart of DL-based real-time MPC for IMM.

$$\begin{aligned}
\min_{u(k)} J &= \sum_{k=1}^{p} \left(y(k) - y_{\mathrm{ref}}(k)\right)^T Q \left(y(k) - y_{\mathrm{ref}}(k)\right) + \sum_{k=1}^{n} \Delta u(k)^T R \Delta u(k), \\
\text{s.t.} \quad & x(k+1) = Ax(k) + Bu(k) + K\hat{\xi}(k), \\
& y(k) = Cx(k), \\
& \Delta u(k) = u(k) - u(k-1), \\
& 0 \le y(k) \le y_{\max}, \\
& 0 \le u(k) \le u_{\max}, \\
& k = 1, \cdots, n.
\end{aligned} \tag{8}$$

The MPC-based control sequence $(u(k), \cdots, u(k+n-1))$ over the predict horizon can be solved by solving the QP problem of the above optimization problem.

Although MPC has many advantages compared to traditional control methods, such as higher control accuracy, faster response speed, and better control effects, which are based on a large number of optimized calculations, as the industrial control process becomes more and more complex, the complexity of the model used in MPC is also getting higher and higher, which undoubtedly increases the computational burden of the control process. This problem is particularly obvious in low computing performance devices such as embedded devices. A large number of matrix operations and memory usage in the MPC calculation process are unaffordable for the embedded devices widely used in the industry, so real-time control is hard to guarantee. This difficulty can be efficiently overcome with the advancement of deep learning. The DL-based MPC controller can greatly reduce the consumption of computing resources while achieving the MPC control function and realize the real-time control while ensuring the control accuracy and robustness. To address this problem in the injection molding prediction control process, we propose a DL-based MPC method to realize the real-time rapid control of the ram velocity.

3.2. Deep Learning. DL is a subset of a broader class of machine learning techniques that can be used to efficiently represent highly varying functions and help to extract abstract nonlinear features from data. It has gained popularity in recent years because of its success in resolving difficult problems that were previously unsolvable in the field of artificial intelligence [34, 36, 45, 46]. Many impressive applications have used DL architectures such as convolutional neural networks, deep belief networks, and deep reinforcement learning for supervised as well as unsupervised learning. Using the capabilities of the DL multilayer deep architecture, large labeled datasets can be learned and evaluated. DNNs with numerous layers of nonlinear hidden units and a big output layer can now be taught more successfully owing to advances in machine learning techniques and computer technology.

In this paper, we combine the DL technique with the traditional MPC to produce high-performance predictive controllers that are simple in design and time efficient, while providing the concept of constraint satisfaction to ensure safety. We construct a GRU neural network to learn the optimal time series control process data produced by the traditional model predictive controller and replace the original model predictive controller, avoiding the complex calculations associated with the traditional optimization process and shortening the optimization solution calculation time. The real-time online control of an injection molding machine is thus achieved, based on assuring stability, robustness, and real time.

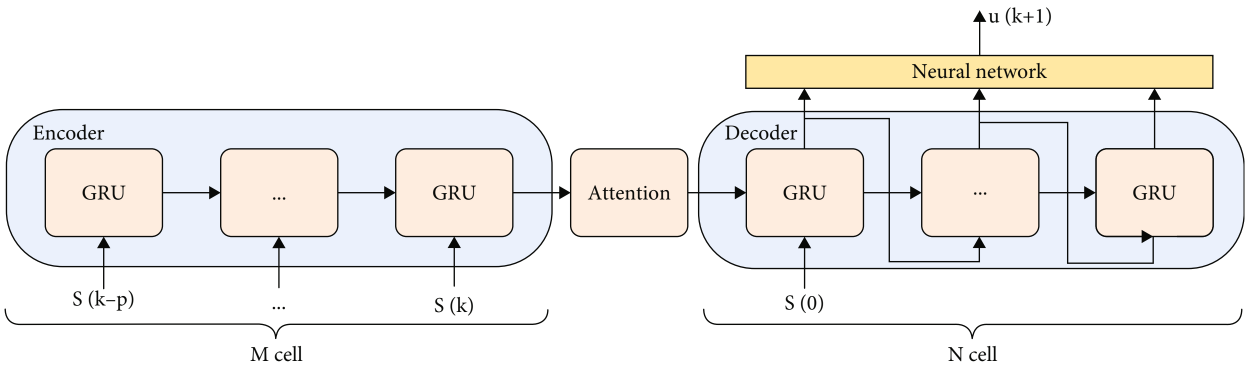


Figure 5: DL-based architecture design.

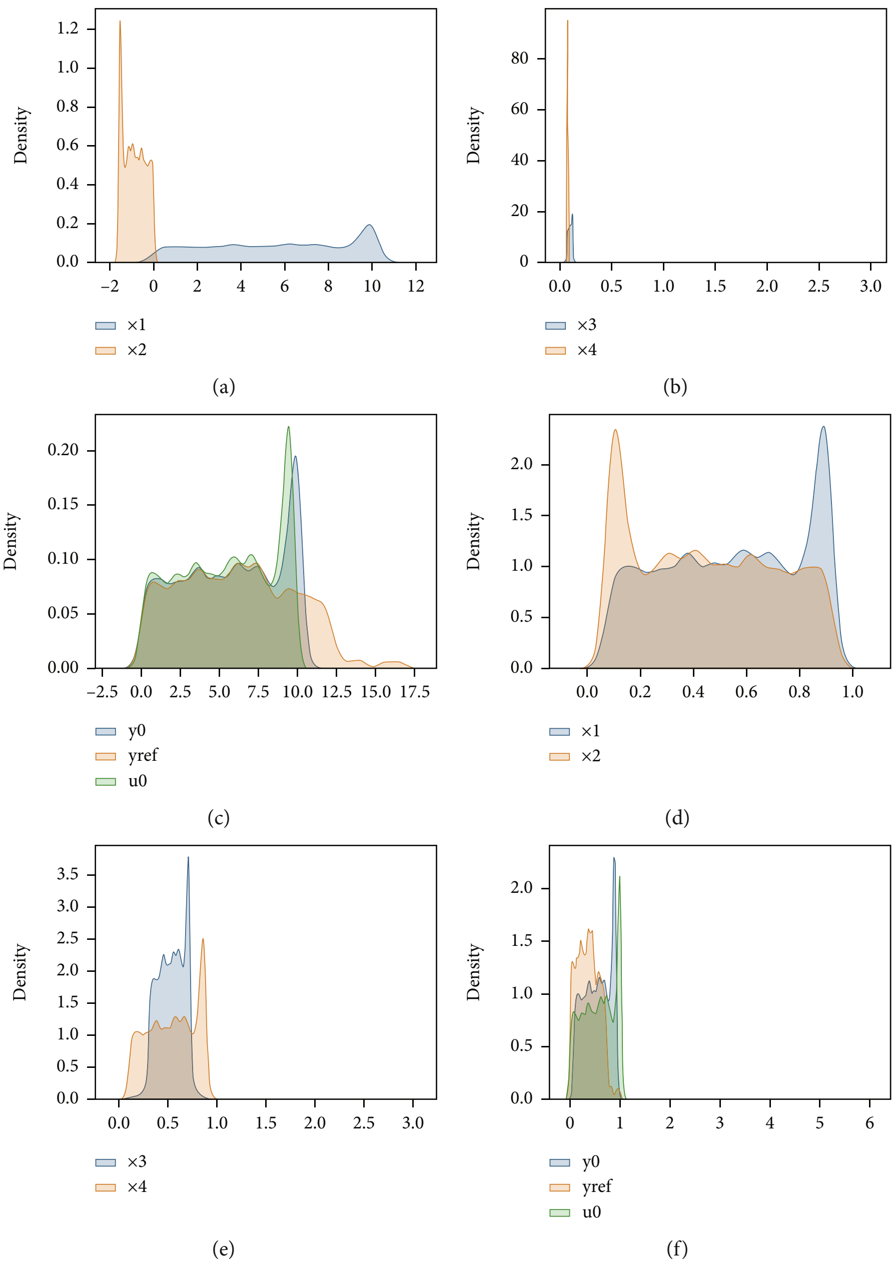


Figure 6: The distribution of dataset. (a–c) The distribution of original data and (d–f) the distribution of normalized data.

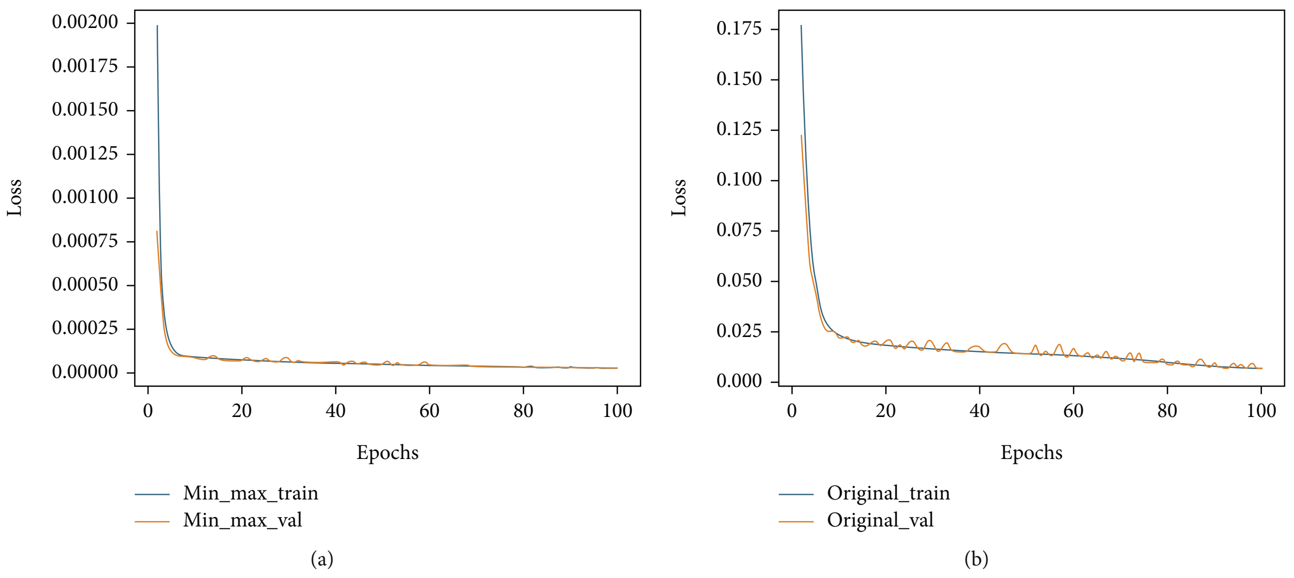


FIGURE 7: Raw and min-max data for network training.

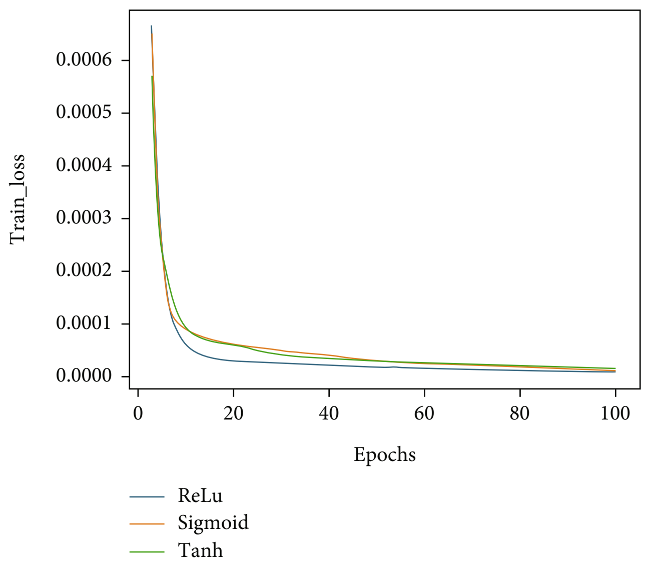
3.3. Deep Learning-Based Architecture Design and Optimization. In this section, we now propose a DL-based architecture design framework and investigate the impact of various network configurations in detail. The overall flowchart of our proposed DL-based MPC control strategy for the IMM is shown in Figure 4, which mainly consists of two parts: offline controller design and training, online controller verification, and application. The former is mainly composed of four parts. We first select the control model, design the MPC controller for the mathematical model, and realize the closed-loop MPC control process simulation according to the MPC control theory. In the process of simulation, the state vector of the control process is collected and an offline dataset is formed, which is used for the subsequent training and tuning of the controller. The latter is to apply the trained controller to the model to verify and analyze its performance. Compared with the MPC controller, the DL-based controller is simpler in structure. The prediction results can be obtained through the forward calculation of the network during the control process, avoiding tedious optimization calculations.

In the framework of DL-based MPC strategy, the most important to-be-designed part is the DL neural network to learn the controller. To this end, we construct a GRU-based neural network to learn the optimal time series control process data produced by the traditional model predictive controller. The DL-based architecture design is mainly composed of the GRU module, sequence-to-sequence (Seq2Seq) module, and attention module, as shown in Figure 5. The module consists of encoder and decoder parts, which are responsible for data acquisition and predicted output, respectively. As indicated in [47–49], the Seq2Seq architecture learns to encode variable-length sequences into fixed-length vector representations and to decode fixed-length vector representations back into variable-length sequences. In deep learning, especially in natural language processing, Seq2Seq networks are often used to solve tasks such as machine translation. In addition, the structure of Seq2Seq performs well in solving problems such as time series prediction [45] and sequence trajectory prediction [50]. This structure allows us to predict the next changes in the model by combining data from multiple past sampling times.

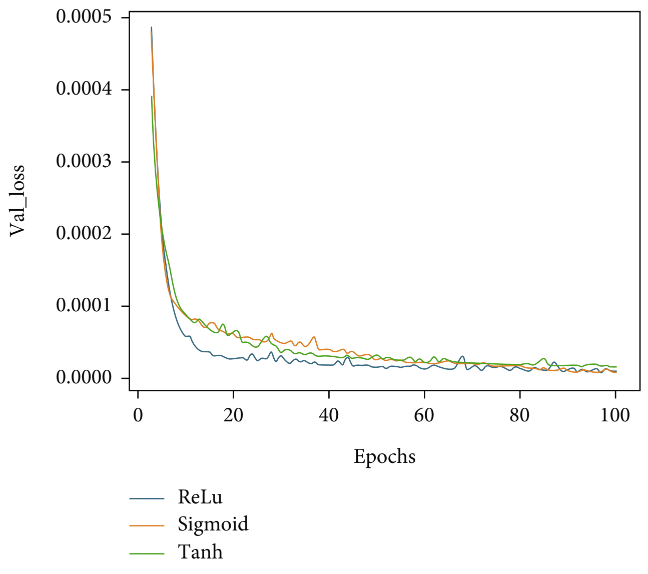
The encoder-decoder learning structure is composed of the GRU which is a popular scheme for learning dependency features of series data. The GRU is an LSTM variant with gating units that control information flow inside the unit but no discrete memory cells. For the learning of time series datasets with a huge amount of data, time series neural networks such as RNN and LSTM often have excellent performance [51, 52]. In comparison to LSTM, the GRU involves less calculation, making it easier for the controller to accomplish real-time control. he GRU consists of an update gate z_k and a reset gate r_k:

$$\begin{aligned} h_k &= (1-z_k)\odot h_{k-1} + z_k \odot \tilde{h}_k, \\ \tilde{h}_k &= \tan h(W_h x_k + U_h (r_k \odot h_{k-1})), \\ z_k &= \sigma(W_z x_k + U_z h_{k-1}), \\ r_k &= \sigma(W_r x_k + U_r h_{k-1}), \end{aligned} \tag{9}$$

where h_k is the activation of the GRU unit at time k, $\tilde{h}_k$ is the candidate activation, e is an element-wise multiplication, and σ is a logistic sigmoid function. W_h, U_h, W_z, U_z, W_r, and U_r are the GRU parameters which are all updated at each training step and stored. To compute the current output y_k and the next activation h_k, the GRU unit uses the previous activation h_{k-1} and the current state x_k. To avoid vanishing gradients, the update gate z_k determines how frequently the unit changes its activation or content. The reset gate r_k forces the unit to read the first symbol in an input sequence, thus erasing the previously computed state.



(a)



(b)

Figure 8: Effect of different activation functions on the network.

In our experiments, the total number of the network inputs is set as 56, including the vector $S = [x, y, y_{\mathrm{ref}}, u]^T$ of the past 6 sampling time, start, and end flags. The output of the GRU network is the predictive control calculation result, which relates to the control input of the next step. To enhance the performance of the network, the attention module is added to the Seq2Seq model. The module can help the network pay more attention to important information in the network input, thereby improving the network's prediction accuracy. The mathematical formula of attention mechanism can be formulated as

$$\mathrm{Attention}(Y, H) = \mathrm{softmax}(W_V \tan h(W_K(S_{t-1}, H))), \quad (10)$$

where Y and H are the predicted output and hidden state of the encoder, respectively. W_K and W_V are weight matrices which can be viewed as multiple single-layer neural networks.

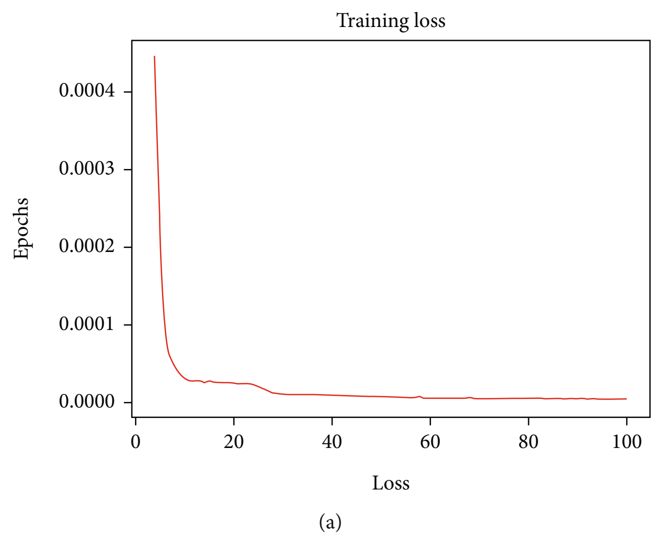
The mean square error (MSE) is chosen as the GRU neural network's loss function, and its mathematical formulation is as follows:

$$\mathrm{Loss}(\hat{u}, u_{\mathrm{ref}}) = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{u}_i - u_{\mathrm{ref}_i}\right)^2, \quad (11)$$

where n indicates the batch size in each training epoch, $\hat{u}$ is the predictive control of the network, and u_{ref} is the true optimal control according to the IMM control input which is obtain by MPC. The depth of the network structure is proportional to the network's training pace. Although a deeper network structure can produce higher convergence results, the time required for training and forward computation increases proportionally and the shallow network performs worse. The architecture of the encoder is 6 hidden layers and that of decoder is 3 hidden layers. The decoder is followed by a single-layer neural network which outputs the prediction results.

3.4. Construction of Datasets. After designing the learning neural network, we need to consider how to build an efficient dataset to train and test our designed DL-based control network. In DL, the training of neural networks mainly relies on large amounts of data. Before designing the DL network and training the controller, we first need to collect data and cover the whole process of ram velocity and all the dynamics should be taken into account. There is a potential correlation between data and data at different sampling times during the injection molding process. In practice, the data that we obtain through the collector is time series data, which contains the mathematical relationship between different parameters. By evaluating the gathered data and determining the internal relationship between the input and output changes over time, we can develop a model to learn and predict the future value of the time series.

In order to efficiently train the designed DL-based MPC controller, we get the data from 360 distinct runs, each of which lasts 1 second and includes 200 steps, resulting in 72000 data pairs. All the runs are under the same initial conditions but different in reference output $y_{\mathrm{ref}}(k)$ and Gaussian noise $\xi(k)$. Each data pair includes state vector $x(k)$, output vector $y(k)$, reference output vector $y_{\mathrm{ref}}(k)$, and the predict control input vector $u(k+1)$ in the sampling time k. The hybrid approach should be able to train the planned network to follow the control rule generated by MPC and attain the required final predictive value, as we hope. The recursive online calculation method is no longer required once we have trained a sufficient number of the GRU for various guiding tasks, with the optimization phase being replaced by a GRU-trained controller.



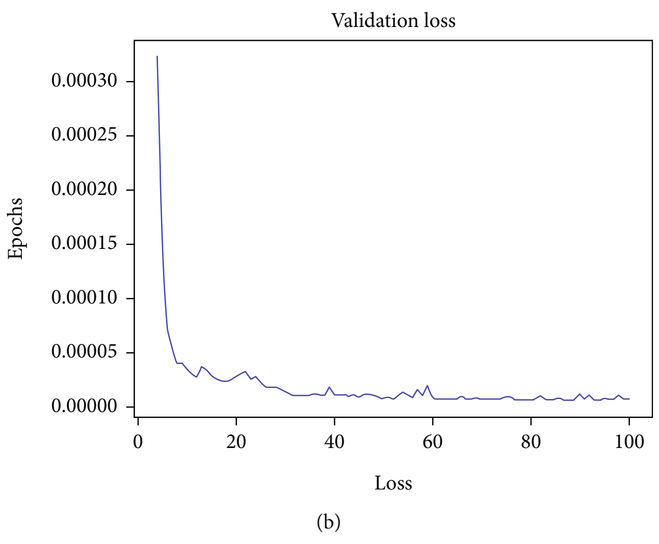


FIGURE 9: Training and validation loss in 100 epochs.

Furthermore, in order to enable the trained network to converge to a stable value faster and avoid the bad local optimal solution during the training process, we carry out a min-max normalization (12) technique for the collected data as follows:

$$X_{\text{min-max}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}}. \tag{12}$$

The min-max normalization can compress the dimension of the raw data to between 0 and 1, thereby enhancing the identifiability of the data and improving the prediction accuracy of the network. It can be seen in Figure 6 that the distribution of various unprocessed data is different, and the range of data size is not concentrated, so that the training process network will be more inclined to fit parameters with larger values. The data preprocessing method provided by min-max normalization can solve this problem well. Figure 7 shows the results of network training using raw and min-max data. The experimental results show that this method can make the network converge to stability more quickly.

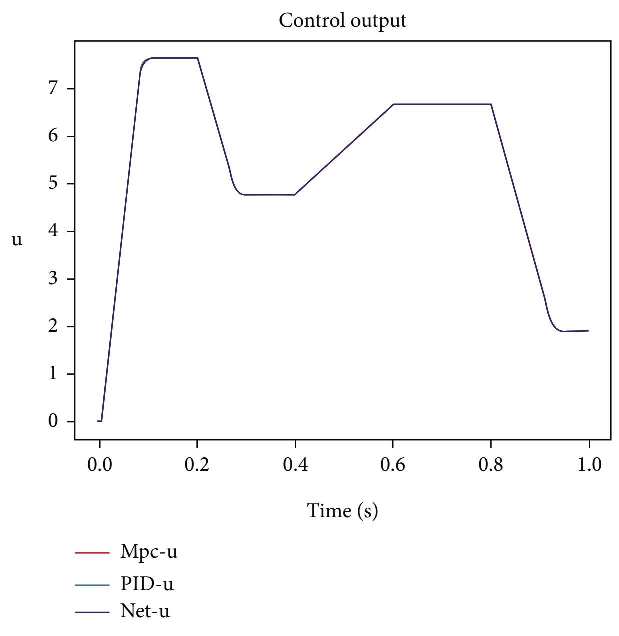
4. Numerical Experiments

The simulation and experimental findings are reported in this part to validate the efficacy of our suggested DL-based control approach. The simulations are run using MATLAB/PyTorch systems, which are powered by an Intel Core i7-8700 processor and 16 GB of RAM. We use 90% of the collected dataset for training and the remaining 10% as the validation set. In our experiments, the encoder and decoder are composed of 6 GRU units and 3 GRU units, respectively, and the hidden state of seq2seq is 5 dimensions. The learning rate and batch size are 0.001 and 64, respectively. The Adam algorithm is used for the neural network optimization, while the ReLU function is used as the activation function. In Figure 8, we show the effect of ReLU, Sigmoid, and tan$h$ functions in the network structure. The results show that the ReLU-based network can converge faster than other commonly used activation functions. Figure 9 shows the training and evaluation errors for the network in 100 epochs, and the errors between them are very small that no overfitting occurs. The network parameters with the lowest MSE during training and verification will be preserved as the best once each round of training is finished and compared to the prior results. Table 1 shows the best results of the training after 100 epochs. The training and verification MSE of the final trained network are basically the same and less than 10^{-3}.

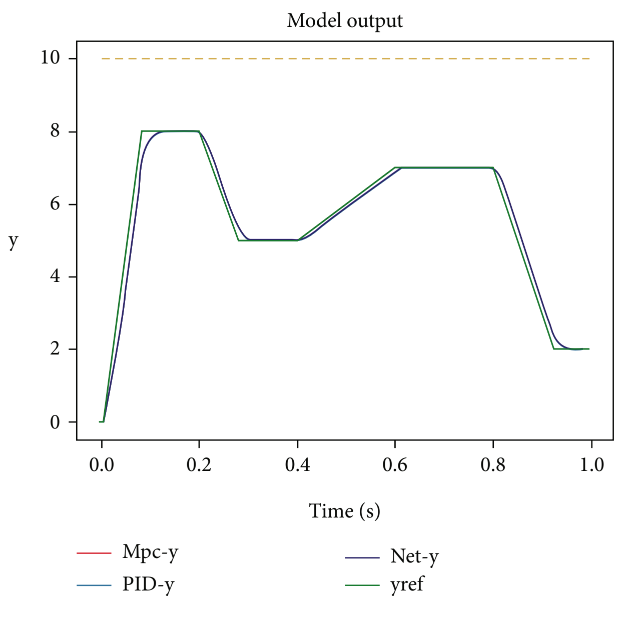
TABLE 1: Training statistics.

Scenario	Amount of data	MSE
Training	64800	4.19×10^{-6}
Validation	7200	5.88×10^{-6}

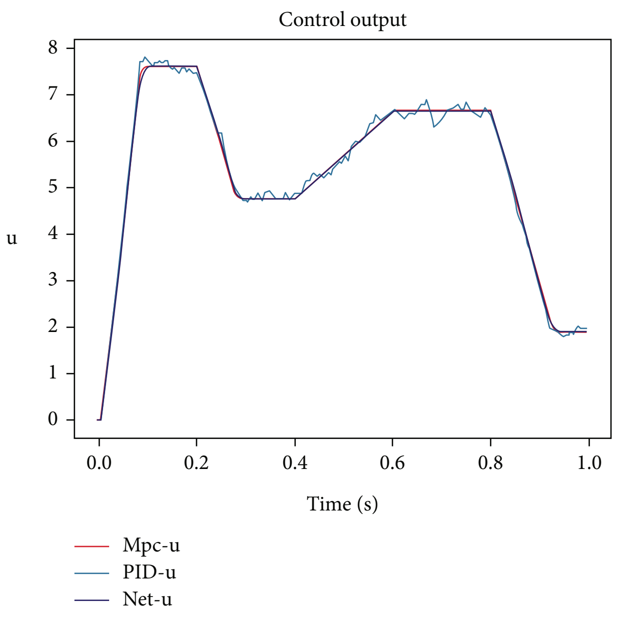
In our experimental simulations, we acquire the simulation running process of different approaches under the same conditions by utilizing different controllers to control the IMM model. In order to further compare the performance of different controllers, we also designed the injection molding simulations based on the widely adopted PID controller. Figure 8 shows the results for 200 steps using an MPC controller, the proposed DL-based controller, and PID controller. As shown in Figure 10, Figure 10(a) is the trajectory of model prediction input u calculated by MPC, DL-based controller, and PID controller and Figure 10(b) is the trajectory of noise ξ, reference y_{ref}, and plant output y corresponding different controllers of u. When the noise ξ is 0, the DL-based controller perfectly reproduces the control process of the MPC controller and PID-based controller. For further comparison, we also generate two sets of running results Figures 10(c)–10(f) under different noise conditions, which have the same conditions but different in noises which variances $Z = (0.1, 0.5)$. The results show that the proposed DL-based controller can achieve more comparable performance than the MPC and PID controllers at each sample time. The



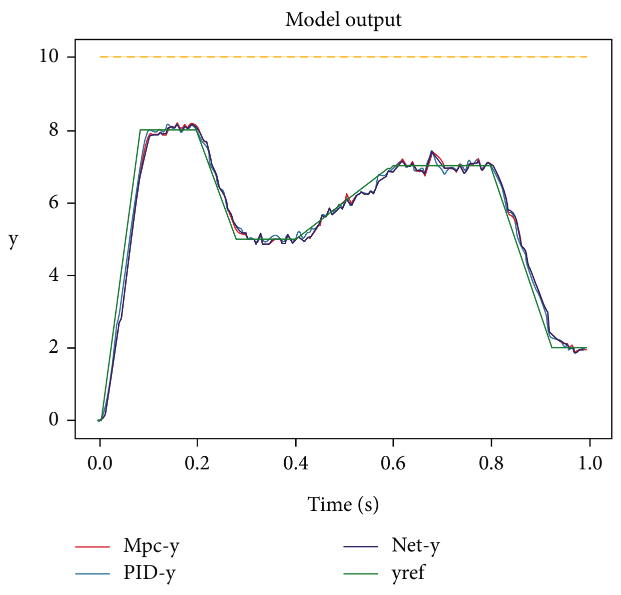
(a)



(b)

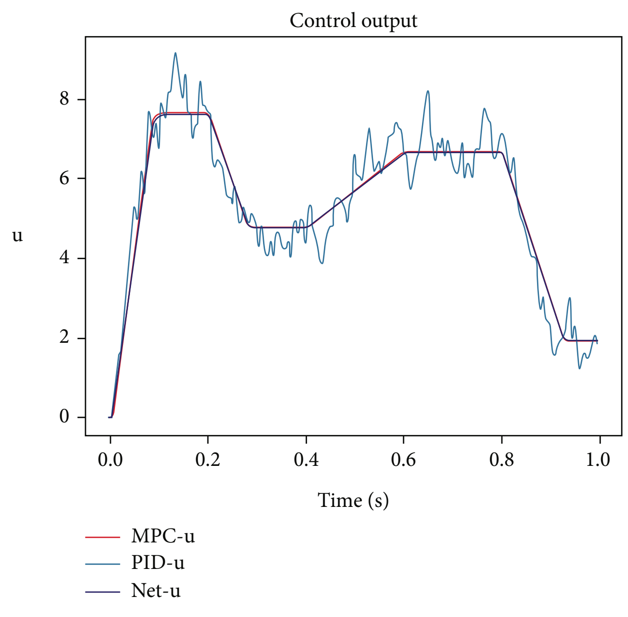


(c)

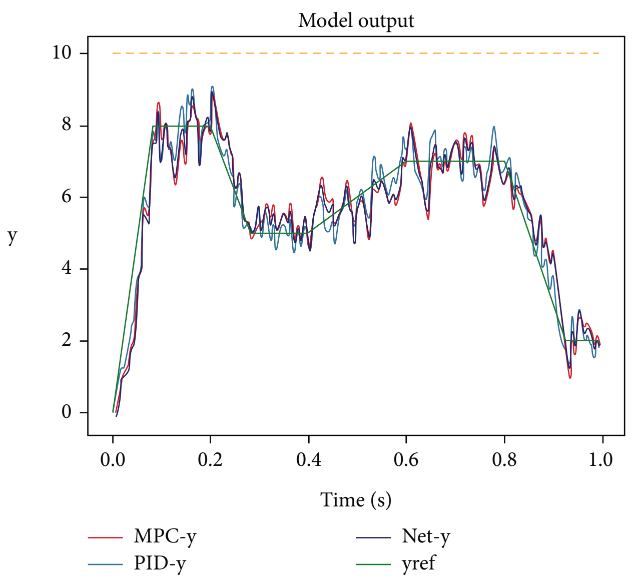


(d)

FIGURE 10: Continued.



(e)



(f)

Figure 10: The control process of different controllers under the same conditions. (a–f) The predictive input u and plant output y under the control of MPC, designed network, and PID from 0 to 1 s. (a, c, and e) The trajectories of u, and (b, d, and f) the trajectories of the reference output y_{ref}, noise ξ, and plant output y.

ram speed y curves under the network and MPC techniques are nearly identical, implying that when the reference y_{ref} changes, the GRU network makes the best decision. Even in the comparison with the PID method, the deep learning method still shows the same or better performance than PID. Moreover, the proposed method does not need to adjust the control parameters. Even in the presence of external interference, the DL-based MPC controller can still track the reference well and the effect is basically the same as that of the MPC controller. However, limited by the MPC strategy that we used, when the reference ram velocity reaches the maximum value with noise, the control curve will fluctuate greatly in order to make the controlled speed change within the constrained range as much as possible.

Figure 11 shows the control process of ram velocity in noiseless and noisy conditions under the constraints. When y is close to the boundary constraint, the network is prone to receive noise interference and frequent fluctuations. Limited by the training data of the model, the training of the network may have the problem of overfitting. We have also taken some measures to address this problem. The first is to expand the coverage of training data to ensure that it can cover most IMM changes. Secondly, one of the important reasons for overfitting is often the high complexity and depth of the model. Compressing the size of the model and reducing the complexity of the network can generally alleviate this problem. Finally, we can reduce the training times of the network and select the best model weight in 100 iterations.

Table 2 illustrates the average tracking error of the two controllers over the course of 30 simulation runs, demonstrating that they have almost identical control capabilities under identical initial conditions and random disturbances. We take the curve of the MPC controller control output as a reference and calculate the average tracking error between the curve under the designed controller and the reference. A very small error is obtained for the proposed DL-based MPC. The computation time for each iteration is considerably decreased from 4.677 ms to 1.691 ms, and the MSE loss is near zero, as shown in Table 2. This finding demonstrates that the DL-based MPC controller has predictive control capabilities comparable to the MPC controller. The current system state predicts the optimal value of the control input at the next moment. The errors mainly appear in the zones that the ram velocity changes. It can be concluded that the controller that we designed basically has the same control performance and robustness as the MPC controller and it consumes less time. In addition, we also made a comparative experiment on the network that contains the attention mechanism and the network that does not contain this mechanism. Under the same conditions, the training MSE of the GRU network without an attention mechanism is 0.000388 and the run time is 1.02 ms. Obviously, compared to the increase in time consumption, the attention mechanism greatly improves the prediction accuracy of the network. The mechanism not only greatly improves the performance of deep neural networks for text or images but is also effective for data-driven control tasks. The key advantage of this DL-based method over the typical MPC-based IMM control method is that it is very simple to construct and calculate, with the highest time consumption coming from the forwarding calculation of the designed network. This unquestionably reduces the control process's computation time and lowers the MPC method's equipment threshold.

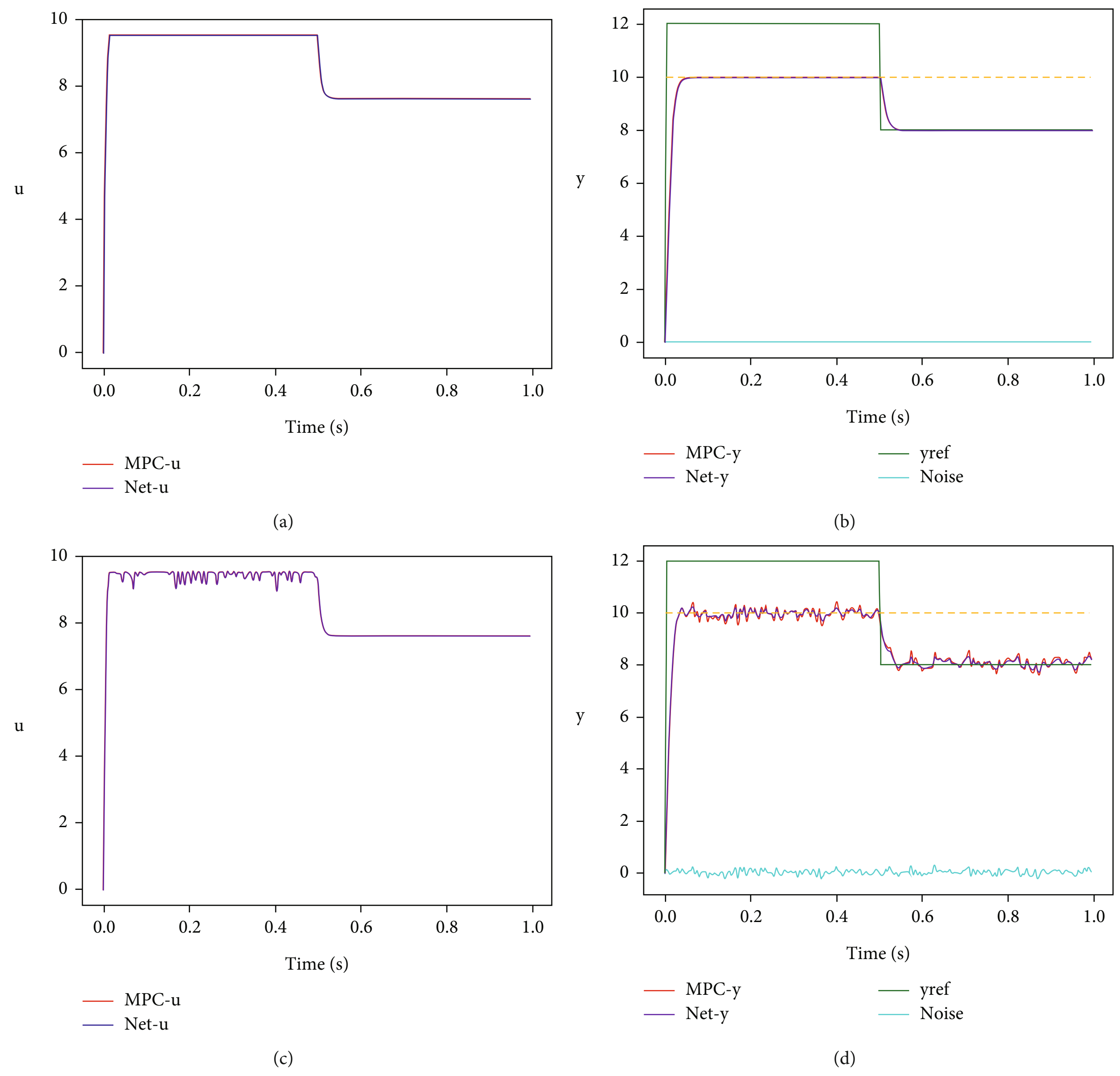


Figure 11: Constrained control process. (a, b) The trajectories of control u and output y when the reference y_{ref} touches the constraint boundary without noise, while (c, d) are the corresponding trajectories under the noise ξ.

Table 2: Average tracking error and run time results of proposed DL-MPC and MPC.

Controller	Tracking error	Run time (ms)
MPC	0	4.677
DL-based MPC	0.006	1.691

5. Conclusions

This paper proposes an effective control strategy with minimal computational efforts for real-time implementation based on the DL technique to approximate the classic MPC strategy to deal with the optimal tracking control of the injection speed in a typical IMM. The system operating data obtained by the classic MPC strategy of the injection speed for the IMM are collected and then used to train the designed network. The trained learning-based controller can output corresponding predictive control results according to the real-time sampled states of the IMM system. Numerical simulation results show that the proposed method is feasible and requires less computational time than the traditional predictive control method. While ensuring the accuracy of predictive control, this method can complete the complex control task of the IMM system well and output the predictive control results quickly, which meets the real-time requirement. With a considerably lower memory footprint and faster calculation time, it can be also implemented in embedded hardware much more efficiently and simply.

In the following work, we will continue to study the DL-based predictive control methods and further improve the experimental methods in this paper, such as considering the impact of the dataset size on the network learning efficiency, comparing the effects of different activation functions and standardized methods, etc. In addition, the proposed method will be applied to more complex injection molding systems combined with such as network compression, network pruning, and other methods, to achieve more efficient real-time predictive control. Note that although the method proposed in this paper mainly focuses on numerical simulation and realization, it is a first and a pretty preliminary attempt at the intelligent control of the injection molding process manufacturing area. In addition, the numerical method and experiment presented in this paper also have a positive role in guiding the subsequent practical experiments. We are also now building a real experimental platform and are trying to implement the proposed strategy for lab-on-chip, and the experimental work is still in progress.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the Key-Area Research and Development Program of Guangdong Province (2021B0101200005), the National Natural Science Foundation of China (62073088 and U1911401), the Guangdong Basic and Applied Basic Research Foundation (2019A1515011606), and the Open Research Fund from Guangdong Laboratory of Artificial Intelligence and Digital Economy (SZ).

References

[1] L. Wang, F. Liu, J. Yu, P. Li, R. Zhang, and F. Gao, "Iterative learning fault-tolerant control for injection molding processes against actuator faults," *Journal of Process Control*, vol. 59, pp. 59–72, 2017.

[2] T. Zou, S. Wu, and R. Zhang, "Improved state space model predictive fault-tolerant control for injection molding batch processes with partial actuator faults using GA optimization," *ISA Transactions*, vol. 73, pp. 147–153, 2018.

[3] M. R. Khosravani and S. Nasiri, "Injection molding manufacturing process: review of case-based reasoning applications," *Journal of Intelligent Manufacturing*, vol. 31, no. 4, pp. 847–864, 2020.

[4] K. K. Tan, S. N. Huang, and X. Jiang, "Adaptive control of ram velocity for the injection moulding machine," *IEEE Transactions on Control Systems Technology*, vol. 9, no. 4, pp. 663–671, 2001.

[5] F. Gao, Y. Yang, and C. Shao, "Robust iterative learning control with applications to injection molding process," *Chemical Engineering Science*, vol. 56, no. 24, pp. 7025–7034, 2001.

[6] K. Yao and F. Gao, "Optimal start-up control of injection molding barrel temperature," *Polymer Engineering & Science*, vol. 47, no. 3, pp. 254–261, 2007.

[7] C. Hopmann, D. Abel, J. Heinisch, and S. Stemmler, "Self-optimizing injection molding based on iterative learning cavity pressure control," *Production Engineering*, vol. 11, no. 2, pp. 97–106, 2017.

[8] Y. Ruan, H. Gao, and D. Li, "Improving the consistency of injection molding products by intelligent temperature compensation control," *Advances in Polymer Technology*, vol. 2019, Article ID 1591204, 13 pages, 2019.

[9] P. Zhao, J. Zhang, Z. Dong et al., "Intelligent injection molding on sensing, optimization, and control," *Advances in Polymer Technology*, vol. 2020, Article ID 7023616, 22 pages, 2020.

[10] K.-C. Ke and M.-S. Huang, "Quality prediction for injection molding by using a multilayer perceptron neural network," *Polymers*, vol. 12, no. 8, p. 1812, 2020.

[11] J. Liu, F. Guo, H. Gao, M. Li, Y. Zhang, and H. Zhou, "Defect detection of injection molding products on small datasets using transfer learning," *Journal of Manufacturing Processes*, vol. 70, pp. 400–413, 2021.

[12] P. Zhao, Z. Dong, J. Zhang et al., "Optimization of injection-molding process parameters for weight control: converting optimization problem to classification problem," *Advances in Polymer Technology*, vol. 2020, Article ID 7654249, 9 pages, 2020.

[13] S. Huang, K. K. Tan, and T. H. Lee, "Predictive control of ram velocity in injection molding," *Polymer-Plastics Technology and Engineering*, vol. 38, no. 2, pp. 285–303, 1999.

[14] S. Wu, Q. Jin, R. Zhang, J. Zhang, and F. Gao, "Improved design of constrained model predictive tracking control for batch processes against unknown uncertainties," *ISA Transactions*, vol. 69, pp. 273–280, 2017.

[15] C. Froehlich, W. Kemmetmüller, and A. Kugi, "Model-predictive control of servo-pump driven injection molding machines," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 5, pp. 1665–1680, 2019.

[16] S. Wu and R. Zhang, "A two-dimensional design of model predictive control for batch processes with two-dimensional (2D) dynamics using extended non-minimal state space structure," *Journal of Process Control*, vol. 81, pp. 172–189, 2019.

[17] R. Dubay, "Self-optimizing MPC of melt temperature in injection moulding," *ISA Transactions*, vol. 41, no. 1, pp. 81–94, 2002.

[18] M. Reiter, S. Stemmler, C. Hopmann, A. Ressmann, and D. Abel, "Model predictive control of cavity pressure in an injection moulding process," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 4358–4363, 2014.

[19] C. Hopmann, A. Ressmann, M. Reiter, S. Stemmler, and D. Abel, "A self-optimising injection moulding process with model-based control system parameterisation," *International Journal of Computer Integrated Manufacturing*, vol. 29, no. 11, pp. 1190–1199, 2016.

[20] M. L. Darby and M. Nikolaou, "MPC: current practice and challenges," *Control Engineering Practice*, vol. 20, no. 4, pp. 328–342, 2012.

[21] D. Liao-McPherson, M. M. Nicotra, and I. Kolmanovsky, "Time-distributed optimization for real-time model predictive

control: stability, robustness, and constraint satisfaction," *Automatica*, vol. 117, article 108973, 2020.

[22] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[23] C. Yang, X. Wang, L. Cheng, and H. Ma, "Neural-learning-based telerobot control with guaranteed performance," *IEEE Transactions on Cybernetics*, vol. 47, no. 10, pp. 3148–3159, 2016.

[24] Z. Wu, S. Liu, C. Ding, Z. Ren, and S. Xie, "Learning graph similarity with large spectral gap," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 51, no. 3, pp. 1590–1600, 2019.

[25] S. Lucia, D. Navarro, B. Karg, H. Sarnago, and O. Lucia, "Deep learning-based model predictive control for resonant power converters," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 1, pp. 409–420, 2020.

[26] R. Hardian, Z. Liang, X. Zhang, and G. Szekely, "Artificial intelligence: the silver bullet for sustainable materials development," *Green Chemistry*, vol. 22, no. 21, pp. 7521–7528, 2020.

[27] M. Kondo, A. Sugizaki, M. I. Khalid et al., "Energy-, time-, and labor-saving synthesis of α-ketiminophosphonates: machine-learning-assisted simultaneous multiparameter screening for electrochemical oxidation," *Green Chemistry*, vol. 23, no. 16, pp. 5825–5831, 2021.

[28] J. Hu, C. Kim, P. Halasz, J. F. Kim, J. Kim, and G. Szekely, "Artificial intelligence for performance prediction of organic solvent nanofiltration membranes," *Journal of Membrane Science*, vol. 619, article 118513, 2021.

[29] N. Jeong, T. H. Chung, and T. Tong, "Predicting micropollutant removal by reverse osmosis and nanofiltration membranes: is machine learning viable?," *Environmental Science & Technology*, vol. 55, no. 16, pp. 11348–11359, 2021.

[30] Z. Ren, J. Lai, Z. Wu, and S. Xie, "Deep neural networks-based real-time optimal navigation for an automatic guided vehicle with static and dynamic obstacles," *Neurocomputing*, vol. 443, pp. 329–344, 2021.

[31] J. Xu, Z. Ren, S. Xie, Y. Wang, and J. Wang, "Deep learning-based optimal tracking control of flow front position in an injection molding machine," *Optimal Control Applications and Methods*, pp. 1–18, 2021.

[32] T. Chen, Z. Ren, G. Lin, and C. Xu, "Learning-PDE-based approximate optimal control for an MHD system with uncertainty quantification," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 11, pp. 7185–7192, 2022.

[33] G. Ignacz and G. Szekely, "Deep learning meets quantitative structure-activity relationship (QSAR) for leveraging structure-based prediction of solute rejection in organic solvent nanofiltration," *Journal of Membrane Science*, vol. 646, article 120268, 2022.

[34] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, 2017.

[35] Q. Zhang, M. Zhang, T. Chen, Z. Sun, Y. Ma, and B. Yu, "Recent advances in convolutional neural network acceleration," *Neurocomputing*, vol. 323, pp. 37–51, 2019.

[36] S. Sengupta, S. Basak, P. Saikia et al., "A review of deep learning with special emphasis on architectures, applications and recent trends," *Knowledge-Based Systems*, vol. 194, article 105596, 2020.

[37] J. Zhang, Y. Li, W. Xiao, and Z. Zhang, "Non-iterative and fast deep learning: multilayer extreme learning machines," *Journal of the Franklin Institute*, vol. 357, no. 13, pp. 8925–8955, 2020.

[38] B. Xu, C. Yang, and Z. Shi, "Reinforcement learning output feedback NN control using deterministic learning technique," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 3, pp. 635–641, 2013.

[39] B. Karg and S. Lucia, "Efficient representation and approximation of model predictive control laws via deep learning," *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3866–3878, 2020.

[40] C. Huré, H. Pham, A. Bachouch, and N. Langrené, "Deep neural networks algorithms for stochastic control problems on finite horizon: convergence analysis," *SIAM Journal on Numerical Analysis*, vol. 59, no. 1, pp. 525–557, 2021.

[41] J. Nubert, J. Kohler, V. Berenz, F. Allgower, and S. Trimpe, "Safe and fast tracking on a robot manipulator: robust mpc and neural network control," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3050–3057, 2020.

[42] H. Karbasi and H. Reiser, "Smart mold: real-time in-cavity data acquisition," in *First Annual Technical Showcase & Third Annual Workshop*, Citeseer, Canada, 2006.

[43] Y. Zou, W. Wu, X. Zhou, G. Wei, and B. Jiang, "A novel method for the quantitative characterization of the simultaneous plasticizing and filling performance in ultrasonic plasticization micro injection molding," *Materials & Design*, vol. 204, article 109680, 2021.

[44] K. Wang, "Injection molding modeling," *Progress Report*, Cornell University, 1984.

[45] S. Du, T. Li, Y. Yang, and S.-J. Horng, "Multivariate time series forecasting via attention-based encoder-decoder framework," *Neurocomputing*, vol. 388, pp. 269–279, 2020.

[46] A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis, "Deep learning for computer vision: a brief review," *Computational Intelligence and Neuroscience*, vol. 2018, Article ID 7068349, 13 pages, 2018.

[47] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014*, pp. 3104–3112, Montreal, Quebec, Canada, December 2014.

[48] K. Cho, B. Van Merriënboer, C. Gulcehre et al., "Learning phrase representations using rnn encoder-decoder for statistical machine translation," 2014, https://arxiv.org/abs/1406.1078.

[49] Y. Keneshloo, T. Shi, N. Ramakrishnan, and C. K. Reddy, "Deep reinforcement learning for sequence-to-sequence models," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 7, pp. 2469–2489, 2020.

[50] R. Quan, L. Zhu, Y. Wu, and Y. Yang, "Holistic lstm for pedestrian trajectory prediction," *IEEE Transactions on Image Processing*, vol. 30, pp. 3229–3239, 2021.

[51] B. Yang, S. Sun, J. Li, X. Lin, and Y. Tian, "Traffic flow prediction using LSTM with feature enhancement," *Neurocomputing*, vol. 332, pp. 320–327, 2019.

[52] Y. Yu, X. Si, C. Hu, and J. Zhang, "A review of recurrent neural networks: LSTM cells and network architectures," *Neural Computation*, vol. 31, no. 7, pp. 1235–1270, 2019.