Pattern Analysis and Applications (2023) 26:1013–1025
https://doi.org/10.1007/s10044-023-01143-1

INDUSTRIAL AND COMMERCIAL APPLICATION

Deep learning-based cutting force prediction for machining process using monitoring data

Soomin Lee[1] · Wonkeun Jo[1] · Hyein Kim[2] · Jeongin Koo[2] · Dongil Kim[3]

Received: 11 August 2022 / Accepted: 24 January 2023 / Published online: 27 March 2023

Abstract

Machining is a critical process in manufacturing industries. With the increase in the complexity and precision of machining, computer systems, such as computerized numerical control, machining monitoring systems (MMSs), and virtual machining (VM), have been incorporated in modern machining processes. In this study, a deep learning-based cutting force prediction method was proposed. MMS and VM data were collected from real-world machining processes. Next, the prediction of the cutting force using five deep learning-based methods, including the long short-term memory (LSTM) and temporal convolutional networks, were analyzed and compared with values measured with a tool dynamometer. The experimental results revealed that the proposed LSTM model, including bidirectional and residual structures, outperformed other benchmark models in terms of predicting the cutting force. Furthermore, the proposed method trained only with MMS data exhibited excellent performance with a root-mean-square error of 12.55 and R^2 of 0.99 on average. Thus, the cutting force required at each point can be predicted accurately, and the method can become a reference for further studies.

Keywords Deep neural network · Long short-term memory · Machining process · Cutting force prediction · Virtual machining

1 Introduction

Machining, wherein a desirable shape of product is obtained by cutting raw materials, is a critical process of manufacturing. In the machining process, a cutting tool attached to a spindle connected to a motor rotates with high speed. The tool forcibly cuts unnecessary parts from raw materials to obtain the desired shape [1]. The process parameters of the machining process are the revolutions per minute (RPM) of the motor and feed rate, which is the speed of the movements. Machining processes includes milling, drilling, boring, and rotating, which differ with the types of cutting tool and desired shapes [2].

With the increase in the complexity of manufacturing processes and level of precision, automated machining systems have become essential for accurate machining processes. An automated machining system comprises computerized numerical control (CNC), which facilitates precise machining using computer programs [3, 4]. Additional computer systems, such as virtual machining (VM) and machine monitoring systems (MMSs), have been developed for machining to improve the precision, accuracy, and productivity. In VM and MMS, the key process information of the machining process is obtained using process parameters as input for simulation [5] and monitoring process data during the machining process [6], respectively.

The cutting force, which is the physical force generated by a cutting tool when in contact with the raw material in a machine, is a critical machining parameter. Precision, chatter, tool lifecycle, and tool breakage can be predicted by

✉ Dongil Kim
d.kim@ewha.ac.kr

Soomin Lee
sxxmin.lee@g.cnu.ac.kr

Wonkeun Jo
jowonkun@o.cnu.ac.kr

Hyein Kim
hyeinkim@kitech.re.kr

Jeongin Koo
jikoo@kitech.re.kr

[1] Department of Computer Science and Engineering, Chungnam National University, 99 Daehak-ro, Yuseong-gu, Daejeon 34134, Republic of Korea

[2] Smart Manufacturing System R&D Department, Korea Institute of Industrial Technology, 89 Yangdaegiro-gil, Cheonan 31056, Republic of Korea

[3] Department of Data Science, Ewha Womans University, 52 Ewhayeodae-gil, Seodaemun-gu, Seoul 03760, Republic of Korea

monitoring the cutting force [7]. Generally, a tool dynamometer is used to measure the cutting force directly [8]. However, the tool dynamometer requires an expensive and complex setup, and cannot be used in all production systems [7]. In VM, the cutting force is calculated without a dynamometer using a mathematical simulation model and machining setup parameters, such as the feed rate and RPM [9, 10]. However, some critical coefficients to be set in VM require intensive actual cutting experiments. The cutting force calculated by VM may be inaccurate if the machine and tool conditions change. Figure 1 displays a comparison between cutting forces measured by the tool dynamometer and those calculated by VMs. The *y*-axis cutting forces obtained by VM were inaccurate for some points because the variations in the tool and machine conditions were not considered.

With considerable advancement in the computational power of computers and the availability of large data, deep neural networks (DNNs) [11] are being incorporated in modern manufacturing processes [12–15]. In addition, considerable data can be collected in modern CNC-based machining process, and DNN-based methods can be utilized to predict the cutting forces. In some studies, the cutting parameters were used as the input variables to predict the average of the cutting force as output variables [16, 17]. Another study focused on predicting the coefficients of VM for calculating the cutting force [18].

We proposed a DNN-based cutting force prediction method for machining processes. In the proposed method, two main data sources, namely MMS and VM, were used for obtaining the input variables. First, we collected real-world machining data from actual machining processes, which are easily collected from the MMS in modern CNC-based machining process, from different cutting parameters (settings). Then the simulated cutting force values for each cutting parameter were obtained from VM. The actual cutting force to be utilized as target values in supervised learning were obtained from a tool dynamometer. Preprocessing included data sampling and time-series synchronization. DNN-based methods, including recurrent neural networks (RNNs) and convolutional neural networks (CNNs), were used to predict the real-time cutting force by training the collected data. We evaluated the various architectures of RNNs by including bidirectional and residual structures. Furthermore, the DNN models trained without VM data were examined to construct a fully data-driven prediction model.

The main contributions can be summarized as follows.

1. *The proposed method incorporates modern DNN techniques, which have been rarely applied previously for cutting force prediction in machining processes.* We used long short-term memory (LSTM), temporal convolutional networks (TCN), bidirectional structures, and residual networks, which are recent DNN techniques. In contrast to VM, the proposed method directly predicts the cutting force because of its fully data-driven approach.
2. *The proposed method predicts the cutting force in the time domain with high resolution.* In most previously reported cutting force prediction methods, the average or peak of the cutting forces within the process unit are predicted. By contrast, the proposed method predicts the cutting force corresponding to every input point of MMS data. The predicted cutting forces at each point may prove to be a guideline for further studies.
3. *The proposed method can predict the cutting force without VM data.* The experimental results revealed comparable performances could be obtained using only the data collected by the MMS. Thus, the proposed method can predict the cutting force without any other

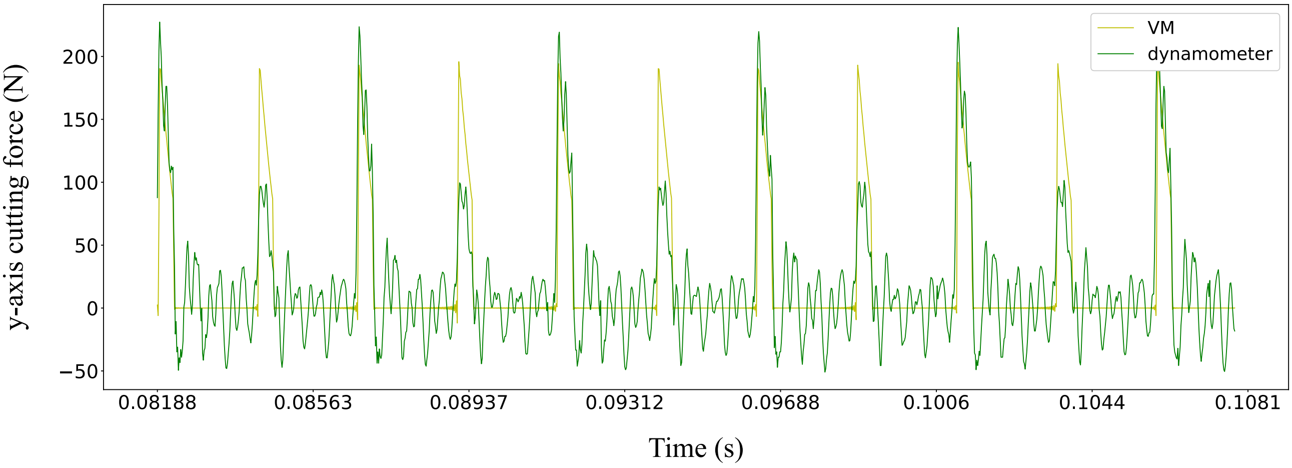


Fig. 1 Comparison of the cutting forces (VM vs. dynamometer)

simulations, but with only monitoring data in real-world machining processes under various machine and tool conditions.

The remainder of this paper is organized as follows. Section 2 presents a review of the related work, and Sect. 3 discusses the proposed method. Section 4 presents the experimental results of applying the proposed method to real-world machining process data. Finally, Sect. 5 concludes this paper by summarizing the proposed method and its limitations, and by identifying directions for future studies to be conducted on this topic.

2 Related work

2.1 Cutting force in machining processes

The cutting force is a critical parameter of machining process optimization [19]. As mentioned in Sect. 1, a tool dynamometer is typically used for measuring the cutting force in machining processes. However, a tool dynamometer is expensive and requires a time-consuming setup. Therefore, mathematical methods, such as VM or a machine learning method, which do not require a tool dynamometer, have been developed. Cutting can be categorized into various types, such as turning and milling. Studies have been conducted to calculate cutting parameters, such as the RPM, feed rate, or peripheral factors, collected from a specific type of the cutting process.

2.2 Mathematical model-based cutting force measurement

In the conventional method, a mathematical model is used to calculate the cutting force coefficients and corresponding cutting force. However, the performance of these methods depends on the materials and cutting types. These parameters are then used to calculate the cutting forces acquired through numerical approximation. Specifically, the linear or nonlinear average cutting coefficient model [20], orthogonal-to-oblique model [21], semi-mechanistic model [22], and Kienzle force model [23] have been proposed for cutting force estimation in the milling process. Thus, a mathematical cutting force model is used for virtual simulation such as VM. However, because of the simplified assumptions of the model, conditions, such as vibrations and instability, are not considered in the virtual simulation of machining processes [24].

2.3 Machine learning-based cutting force prediction

For machine learning-based methods, the input variables are determined based on the experimental environment and proposed method. The output variables can be set as the cutting force or coefficients associated with the cutting force depending on the purpose. Generally, parameters, such as the RPM and feed rate, are used as the input variables for cutting force prediction [25].

Radhakrishnan and Nandan [26] revealed that neural networks exhibit superior cutting force prediction performance compared with linear regression model using cutting parameters including the RPM, feed rate, and depth of cut as input variables in a milling process unit. Zuperl et al. [27] presented a generalized neural network using 10 input parameters, such as the type of the material, dynamometer of tool, and feed rate. Rai et al. [28] used a neural network by applying more than 10 input variables. Irgolic et al. [29] used only the axial depth of cut and feed rate as input variables, which are related to the cutting forces. Other research efforts proposed hybrid approaches with a neural network, where input variables collected from two mathematical model-based simulation and cutting parameters were used [30, 31]. Wang et al. [17] proposed a transfer learning-based cutting force prediction method. In their method, a neural network, comprising four hidden layers that were pretrained through simulation data, was fine-tuned using data acquired at the time intervals in experiments at each learning step.

The aforementioned studies have used neural networks comprising fully connected layers. Because the cutting force varies at any moment, most studies have predicted a maximum of the cutting force [27, 32, 33] or an average of the cutting force [17, 26] in the process. Xu et al. [18] proposed ForceNet using cutter-workpiece engagement (CWE) [34] geometry representation. By using the sequence information of CWE, cutting forces corresponding to the input variables can be predicted.

2.4 Deep learning model for time-series

2.4.1 LSTM

The DNN comprises multilayers, allowing the demanded output to be extracted, while passing through layers sequentially from a first layer starting with the input. In multilayer perceptrons, the relationship between fully connected layers and time-series data is ignored. By contrast, RNNs, an extension of DNNs, have been widely used for considering the statistical properties of time-series. Furthermore, LSTM [35], which captures a long-term dependency in a time-series and is a novel architecture of the RNN, has become a dominant method [36, 37].

Figure 2 displays an LSTM unit. An LSTM unit includes three gates (namely input, forget, and output) to control the flow of data and state of the unit. These gates are calculated as follows:

$$\begin{aligned} i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\ o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \\ \tilde{C}_t &= \tanh(W_c \cdot [h_{t-1}, x_t] + b_i), \end{aligned} \tag{1}$$

where i_t, f_t, and o_t are the outputs of the input, forget, and output gates at time-step t, respectively. Here, W and b represent the weights and biases for each gate, respectively. Additionally, h_{t-1}, x_t, σ, and tanh are the output value of the previous LSTM unit, input value of the current LSTM unit, sigmoid activation function, and hyperbolic tangent function, respectively. $\tilde{C}_t$ has a value range of [− 1,1] by a tanh function, and is point-wise multiplied with i_t. Furthermore, the point-wise multiplication is performed between C_{t-1} and f_t. By adding these values, the current state is calculated. Finally, we can obtain h_t as follows:

$$\begin{aligned} C_t &= f_t * C_{t-1} + i_t * \tilde{C}_t \\ h_t &= o_t * \tanh(C_t). \end{aligned} \tag{2}$$

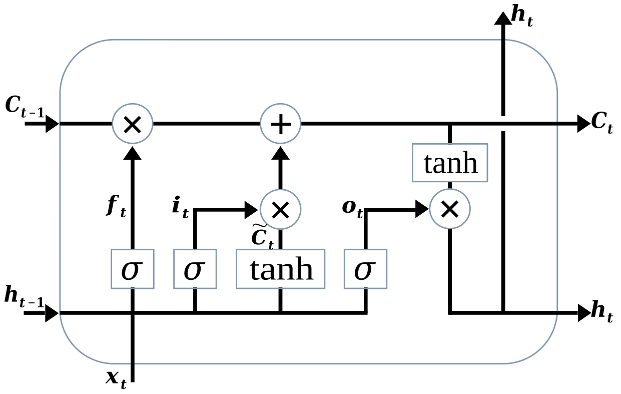


Fig. 2 LSTM unit

2.4.2 TCN

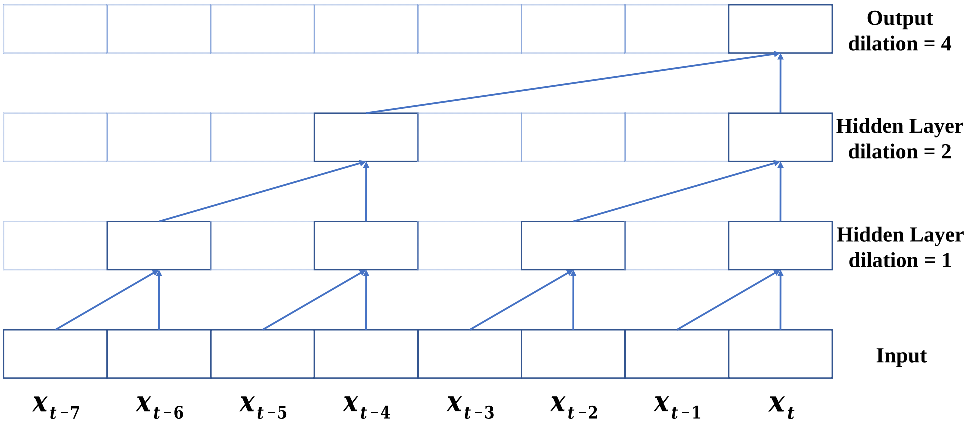
CNN is one of the popular structures to capture the local information of input variables [11, 38]. Convolution can be designed to extract key local information from the input variables. Specifically, TCN can be used for capturing the time-dependent information in time-series data [39–41]. Figure 3 displays one of the convolution types, called dilated causal convolution, used in the TCN. By using the causal convolution, information from the previous points can be reflected in the next hidden layers. In addition, by applying the dilation, dilated causal convolution helps to reflect the previous long-term information of input time-series from more deeper model.

3 Proposed method

3.1 Overview

We proposed a novel DNN-based cutting force prediction method. The overall process of the proposed method is displayed in Fig. 4. First, training data for input variables are collected from MMS and VM, and the output variables are collected from a tool dynamometer. Next, data preprocessing is performed for applying the input data to a regression model. The proposed DNN-based regression model predicts the x- or y-axis cutting force, corresponding to all points of the input variables. The final purpose of the proposed method is to predict the cutting force with monitoring data

Fig. 3 Dilated causal convolution



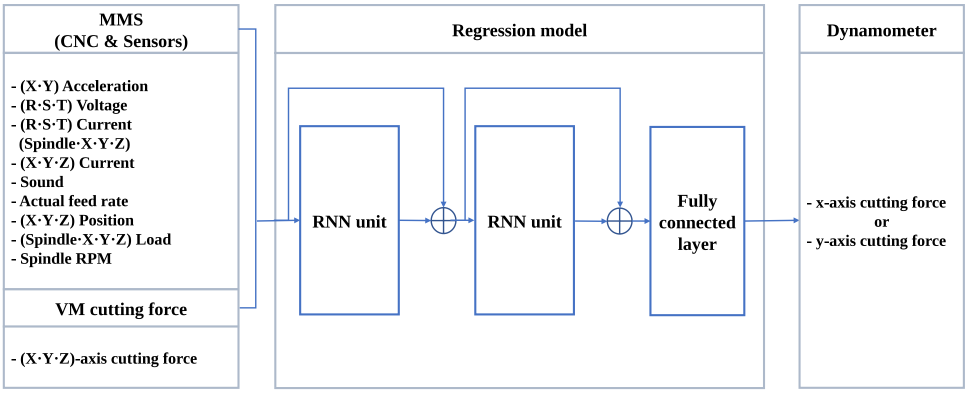


Fig. 4 Block diagram of the proposed method

Table 1 Cutting parameters included in our collected data

Cutting parameter	Value				
RPM	3000	3500	4000	4500	5000
Feed rate (mm/min)	300	350	400	450	500
a_p (mm)	1	2.25	3.5	4.75	6
a_e (mm)	1	2	3	4	

Table 2 Data collected from multiple sources

	Source	Variable
Input	MMS	Time
		(X.Y) Acceleration
		(R.S.T) Voltage
		(R.S.T) Current (Spindle.X.Y.Z)
		(X.Y.Z) Current
		Sound
		Actual feed rate
		(X.Y.Z) Position
		(Spindle.X.Y.Z) Load
		Spindle RPM
	VM	Simulated cutting force (x-, y-, and z–axis)
Output	Tool dynamometer	($X.Y.Z$)-axis cutting force

that are easier to collect and reflects the machine and tool conditions. Furthermore, the proposed method aims to construct a unified model that can be applied to all data by one model.

3.2 Data collection

The data were collected from a real-world machining system of a Korean national research institute, during 2020–2021. The machining process was conducted on the slot milling process of a DMG-MoriSeiki machine. Milling is the most common machining process that involves the use of a multitooth cutter to remove metal from a workpiece surface to create flat and angular surface and grooves. Indexable end mil, ϕ16, HSK63, was used as the cutting tool. SM45C carbon steel was used as the raw material to be cut. The main cutting parameters to be set for milling were the RPM, feed rate, a_p, and a_e. The RPM controls the rotation speed of the motor, and the feed rate is the velocity of the cutting tool. Moreover, a_p and a_e are the depth of the cut in the axial and radial directions, respectively. The MMS and VM data were collected from the 500 cutting parameter combinations, as summarized in Table 1. We collected all 31 time-series provided by MMS, which occur during the machining process, including x-, y-, and z-axis positions; three-phase current; load; and x- and y-axis acceleration. The raw data were collected by a pass for each of the cutting parameter combinations in the milling process, and each combination had 40,000 observations on average. The VM data, including simulated x-, y-, and z-axis cutting forces corresponding to each cutting parameter, were collected from the CutPro milling simulation package [42]. For the output variables, the x- and y-axis cutting forces measured with a tool dynamometer were used as the reference. The input and output variables are summarized in Table 2. Although we obtained the x-, y-, and z-axis cutting forces from the tool dynamometer, we only selected the x- and y-axis cutting forces as the target variables. The machining process was conducted on a two-dimensional flat surface; hence, the z-axis cutting force had no meaningful variance and was not of interest to the experiment.

3.3 Data preprocessing

The MMS comprises a CNC system and sensors. Voltage and current sensors were used in the CNC system to monitor the machining process. However, the data were collected by each source at a different data collection frequency. The signal frequency is unified to use all the collected data as input variables. To apply to the proposed method, Fig. 5 displays an approach for integrating each signal frequency. We fit all

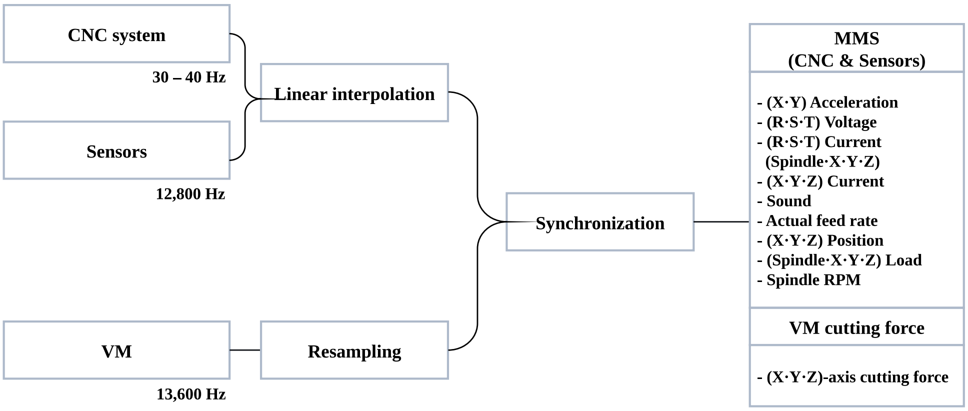


Fig. 5 Preprocessing of multivariate time-series data

signal frequencies to 12,800 Hz to ensure consistency with the signal frequency of the output variables measured from the tool dynamometer. Linear interpolation was applied to the data collected from the CNC system and sensors within the MMS. The simulation cutting force data collected from VM had a higher frequency than that in the data collected from the MMS, and the signal frequency of VM differed from that of the tool dynamometer. To use the data collected from VM as input variables, resampling was applied at 12,800 Hz. Therefore, 20,499,968 observations were obtained for the entire cutting parameter combinations. In addition, z-normalization was applied to the input variables according to each cutting parameter combination. All data generated in the machining process were multivariate time-series, and their length was too long to be trained from a model. Therefore, we sampled the multivariate time-series data of 128 length.

3.4 DNN-based regression model

DNN-based regression models were used to predict the cutting force at the each point in time with 128-length multivariate time-series data as the input variables, which can be represented by a vector, $X_T{}^{(l,d)} = [x_{t-(l-1)}, x_{t-(l-1)}, \dots, x_t]^{(l,d)}$. Here, l is the time-series length, which is fixed to 128 in the proposed method, and d is the number of time-series, which indicates the size of dimension. The regression models based on LSTM were used to consider the sequential characteristics of $X_T{}^{(l,d)}$.

We summarize Eqs. (1) and (2) to be a simple notation as presented as in Eq. (3). In Eq. (3), d' is the number of hidden units, which is fixed to 256 units in the proposed method. $H^{(l,d')}$ is the final output vector of an LSTM layer. Two LSTM layers were designed to consider the bidirectional LSTM (BiLSTM) [43] structure in the proposed method. A BiLSTM considers both the forward and backward directions of $X_T{}^{(l,d)}$ to calculate hidden vectors, $\overrightarrow{H}^{(l,d')}$ and $\overleftarrow{H}^{(l,d')}$, respectively.

$$H^{(l,d')} = \mathrm{LSTM}\left(X_T{}^{(l,d)}\right) \tag{3}$$

Equation (3) can be expressed in detail considering the direction of the sequence. In Eq. (4), represents the $\mathrm{LSTM}_{\mathrm{forward}}(\cdot)$ function that determines the hidden vector from the forward or backward vector of the input variables, and $\overrightarrow{H}^{(l,d')}$ denotes a hidden vector returned through the forward sequence information. By contrast, $\mathrm{LSTM}_{\mathrm{backward}}(\cdot)$ is a function that considers the backward sequence information of the input variables, and $\overleftarrow{H}^{(l,d')}$ denotes a hidden vector returned through the backward sequence information.

$$\begin{aligned} \overrightarrow{H}^{(l,d')} &= \mathrm{LSTM}_{\mathrm{forward}}(X_T{}^{(l,d)}) \\ \overleftarrow{H}^{(l,d')} &= \mathrm{LSTM}_{\mathrm{backward}}(X_T{}^{(l,d)}) \end{aligned} \tag{4}$$

To design the BiLSTM, the hidden vectors, as derived from Eq. (4), should be aggregated. We proposed to concatenate the two hidden vectors relative to the dimension axis in the proposed BiLSTM. A layer normalization [44] was applied to the concatenated vector for normalizing the distributions of the intermediate layers. Equation (5) is expressed as follows:

$$Z^{(l,2d')} = \mathrm{Dropout}\left(\mathrm{LN}\left(\mathrm{concat}\left(\overrightarrow{H}^{(l,d')}, \overleftarrow{H}^{(l,d')}\right)\right)\right), \tag{5}$$

where concat(·), LN(·), and Dropout(·) indicate a function that concatenates two vectors, the layer normalization, and the dropout unit for preventing overfitting (fixed to 0.3), respectively. Here, $Z^{(l,2d')}$ can be obtained through Eq. (5). Figure 6 displays the structure of the RNN unit with two stacked LSTM layers in the proposed method to obtain $Z^{(l,2d')}$. Equations (4) and (5) summarize $Z^{(l,2d')}$ as a function, $F_{\mathrm{RNN}_{\mathrm{unit}}}(\cdot)$. A residual structure was used for the regression model. The regression model was stacked by two RNN units, and has a skip connection. The residual structure is a widely used to improve the performance and the convergence of neural networks [45, 46]. A generalized formulation is expressed in Eq. (6) for applying the residual structure in the proposed method:

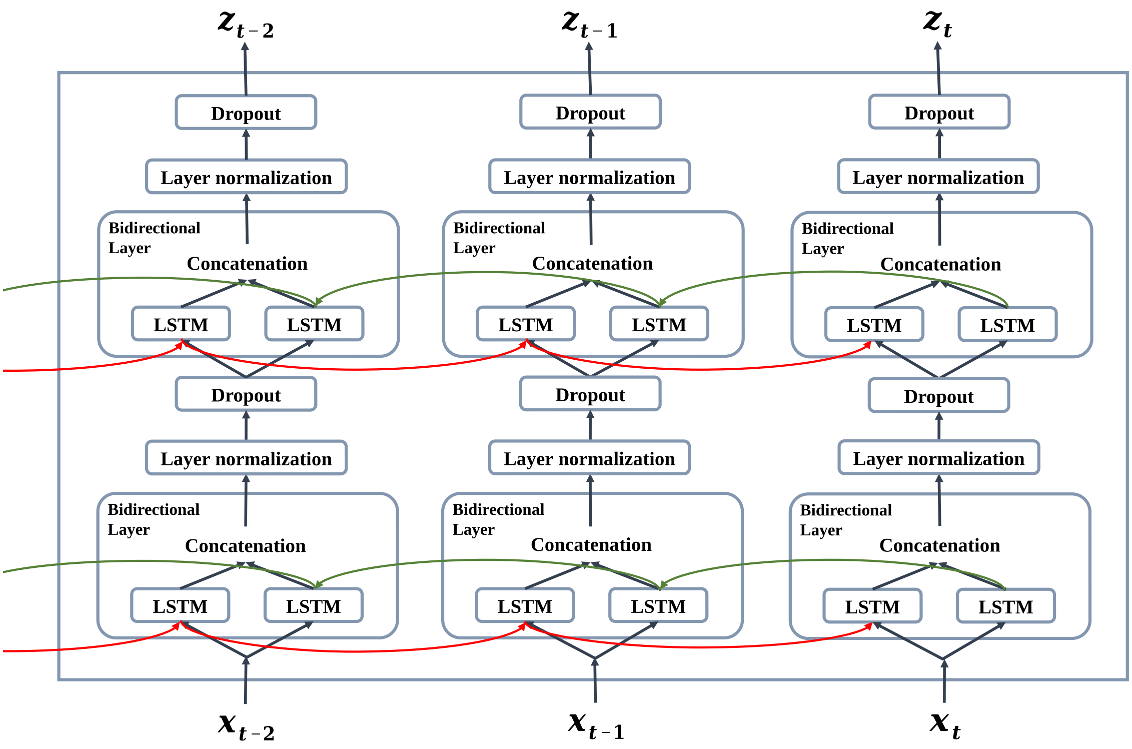


Fig. 6 Detailed architecture of the proposed stacked BiLSTM units

$$
\begin{aligned}
Z_1^{(l,2d')} &= F_{\mathrm{RNN}_{\mathrm{unit}}}\left(X_T^{(l,d)}\right)(i=1)\\
Z_i^{(l,2d')} &= F_{\mathrm{RNN}_{\mathrm{unit}}}\left(Z_{i-1}^{(l,2d')}\right)+Z_{i-2}(i=2,3,\cdots,L-1)\\
Z_L^{(l,2d')} &= F_{\mathrm{RNN}_{\mathrm{unit}}}\left(Z_i^{(l,2d')}\right)+Z_{i-1}(i=L),
\end{aligned}
\tag{6}
$$

where L was fixed at 2 in the proposed regression model. Finally, the sequence that has passed completely to the residual structure returns the predicted x- or y-axis cutting force, $\hat{Y}^{(l,1)}$, by passing through a fully connected layer as follows:

$$\hat{Y}^{(l,1)} = FC_{\mathrm{layer}}\left(Z_L^{(l,2d')}\right). \tag{7}$$

Table 3 List of the regression models used in the experiments

Regression model	Bidirectional structure	Residual structure	Name
Multi-layer perceptron	×	×	MLP
CNN-based model	×	○	TCN
LSTM-based model	×	×	LSTM-o
	×	○	LSTM-R
	○	×	BiLSTM
	○	○	BiLSTM-R (Proposed)

4 Experiments

4.1 Experiment setup

As summarized in Table 2, the input variables and actual cutting forces were used to train the regression models. The inclusion of VM data as the input was examined in the experiments. Six regression models, including the proposed method, were compared: four LSTM-based models, a CNN-based model, and a multi-layer perceptron (MLP). Table 3 summarizes the six regression models and components included in each model. The four LSTM-based models had the same settings except the inclusion of the bidirectional and residual structures. BiLSTM-R (the proposed method) employed the bidirectional and residual structures. In contrast, LSTM-R and BiLSTM employed only the residual and bidirectional structure, respectively, whereas LSTM-o employed the original LSTM structure. For the CNN-based model, a TCN was employed. The TCN contains casual and dilated convolution for reflecting the previous information in a time-series, and the residual structure is used. Note that the MLP employs fully connected layers. In previous studies, MLPs have been used to predict the cutting force using cutting parameters, such as the feed rate and depth of cut, as the input variables [16, 29]. In our experiments, we compared the prediction performance of the MLP, TCN, and LSTMs using time-series monitoring data.

The hyperparameters set for each model are summarized in Table 4. LSTM unit indicates an LSTM module as in Fig. 6. Table 5 summarizes the search results for hyperparameters to determine the architecture of the proposed method. The proposed method presents improvements as the number of BiLSTM units and the size of hidden nodes increase. We set the number of BiLSTM units to two and the size of hidden nodes in a BiLSTM unit to 256. In addition, the proposed method used a dropout ratio of 3.0 based on a search between 0.1 and 0.3. The number of learnable parameters for TCN was set to be similar to that of LSTM-o by manipulating the number of temporal blocks [47] and channels. The architecture of the MLP was set to be similar to that considered in [16]. The rectified linear unit for activation functions [48] was employed for each model. For normalization, layer normalization was used for each model except the MLP, which employed batch normalization [49]. All regression models were implemented with Python using Keras API in TensorFlow. The weights of the models were optimized using an adaptive moment estimation (Adam) algorithm. The loss function was the commonly used mean squared error (MSE), and was used in the regression problem.

Ten-fold cross-validation was performed to evaluate the performance of each model. For the 500 cutting parameter combinations, the training and testing data were divided in a balanced manner from a stratified split. Root MSE (RMSE), normalized RMSE (NRMSE), and R^2 were used as evaluation criteria. In Eqs. (8), (9), and (10), y, $\hat{y}$, and $\bar{y}$ indicate the actual cutting force, cutting force predicted by the regression model, and average value of y, respectively. When calculating these three evaluation metrics, the macro strategy of considering the average value for each fold was used.

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2} \tag{8}$$

$$\mathrm{NRMSE} = \frac{\mathrm{RMSE}}{y_{\max} - y_{\min}} \tag{9}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y}_i)^2} \tag{10}$$

Table 4 Hyperparameter setting for each model

Model	# of units or blocks	Set for a unit or a block	Training parameters (shared)	# of learnable parameters
MLP	Fully connected layer, 2	Hidden size: 256	Batch size: 256 dropout rate: 0.3 learning rate: 0.001 early stopping criterion: 5	141,569
TCN	Temporal block, 8	# of channels: 256, kernel size: 3		3,450,113
LSTM-o	LSTM unit, 2	Hidden size: 256		3,161,857
LSTM-R	LSTM unit, 2	Hidden size: 256		3,161,857
BiLSTM	LSTM unit, 2	Hidden size: 256		8,935,425
BiLSTM-R (proposed)	LSTM unit, 2	Hidden size: 256		8,935,425

Table 5 Hyperparameter search for model architecture of the proposed method (BiLSTM-R)

# of BiLSTM units and set for a unit	*x*-axis cutting force			*y*-axis cutting force		
	RMSE	NRMSE	R^2	RMSE	NRMSE	R^2
1, [64]	36.2729	0.5693	0.9772	11.3945	0.1522	0.9806
1, [128]	19.4829	0.1642	0.9932	7.0168	0.0577	0.9926
1, [192]	14.1628	0.0867	0.9964	6.4421	0.0486	0.9937
1, [256]	16.0971	0.1121	0.9953	6.3676	0.0475	0.9939
2, [64, 64]	27.3344	0.3233	0.9869	10.9658	0.1410	0.9820
2, [128, 128]	30.5097	0.4027	0.9837	6.9653	0.0568	0.9927
2, [192, 192]	23.5624	0.2402	0.9902	6.6708	0.0521	0.9933
2, [256, 256]	**12.5562**	**0.0682**	**0.9972**	**5.4844**	**0.0352**	**0.9955**

Bold text in the Table indicates the best result of the comparative settings or models.

Table 6 Experimental results of each model trained with the MMS and VM data

Model	*x*-axis cutting force			*y*-axis cutting force		
	RMSE	NRMSE	R^2	RMSE	NRMSE	R^2
MLP	251.9727	27.4732	−0.0911	88.3100	9.1449	−0.1543
TCN	36.2773	0.5694	0.9774	15.0827	0.2667	0.9664
LSTM-o	35.7926	0.5543	0.9778	10.8538	0.1381	0.9824
LSTM-R	31.2546	0.4226	0.9830	7.2618	0.0618	0.9921
BiLSTM	15.1472	0.0992	0.9959	6.0988	0.0436	0.9944
BiLSTM-R (proposed)	**14.5764**	**0.0919**	**0.9962**	**6.0168**	**0.0424**	**0.9946**

Bold text in the Table indicates the best result of the comparative settings or models.

Table 7 Experimental results of each model trained only with the MMS data

	x-axis cutting force			*y*-axis cutting force		
Model	RMSE	NRMSE	R^2	RMSE	NRMSE	R^2
MLP	197.0717	14.7997	0.3326	61.4318	3.7803	0.4416
TCN	29.5185	0.3770	0.9850	16.8316	0.3322	0.9581
LSTM-o	47.4291	0.9734	0.9611	8.5506	0.0857	0.9891
LSTM-R	31.1535	0.4199	0.9831	7.0940	0.0590	0.9925
BiLSTM	13.3569	0.0771	0.9968	5.601	0.0367	0.9953
BiLSTM-R (proposed)	**12.5562**	**0.0682**	**0.9972**	**5.4844**	**0.0352**	**0.9955**

Bold text in the Table indicates the best result of the comparative settings or models.

4.2 Experimental results

Tables 6 and 7 detail the cross-validated experimental results for the entire dataset with and without VM data as the input, respectively. The proposed method (BiLSTM-R) with the bidirectional and residual structures outperformed other benchmark models for all experiments. Moreover, the trained models without VM data revealed comparable performance to those with VM data, except for LSTM-o. The results revealed that the cutting force can be predicted using real-time monitoring data without other guidelines from mathematical simulation data. The proposed method without VM data achieved the best performance. The *x*-axis cutting force ranged from −641.3 to 1669.8, and the *y*-axis cutting force ranged from −259.2 to 593.7. Although the proposed method produced a higher RMSE value for the *x*-axis cutting force compared to that for the *y*-axis cutting force, we can confirm that the proposed method accurately predicts both cutting forces based on the NRMSE. Thus, the fully data-driven approach is promising. The LSTM-based regression model outperformed the MLP and TCN. Specifically, the MLP is not suitable for learning using time-series data, and the TCN performs similar to LSTM-o. Nevertheless, BiLSTM and BiLSTM-R present better performance than the TCN. As presented in Tables 6 and 7, BiLSTM and BiLSTM-R with the bidirectional structure outperformed the other regression models. By using a residual structure, BiLSTM-R exhibited slightly improved performance than BiLSTM. Considering the improvement in performance from LSTM-o to BiLSTM and from LSTM-o to LSTM-R, we can conclude that the bidirectional and residual structure improved the model performance. Finally, the proposed method employing both structures outperformed the other benchmark methods.

As mentioned, the model was trained and evaluated with the 10-fold cross-validation on the entire dataset, including all combinations of cutting parameters. However, because regression models underestimate the deviation of the predictors [50], the prediction performance can differ when predicting the cutting force at the end of range. For instance, if a model trained with RPM values of 3000–5000, the accuracy can differ when predicting RPM values of 4000 and 5000. Therefore, the overall experimental results are divided according to the accuracy for predicting the cutting force for each cutting parameter. Figures 7 and 8 reveal the *RMSE* evaluated for predicting the cutting force on each cutting parameter value except for the MLP model. As displayed, the prediction error increased as the cutting parameters increased, except for the RPM. Moreover, the TCN exhibited considerable deviation in the predictive performance with the variation in the cutting parameters. LSTM-o and

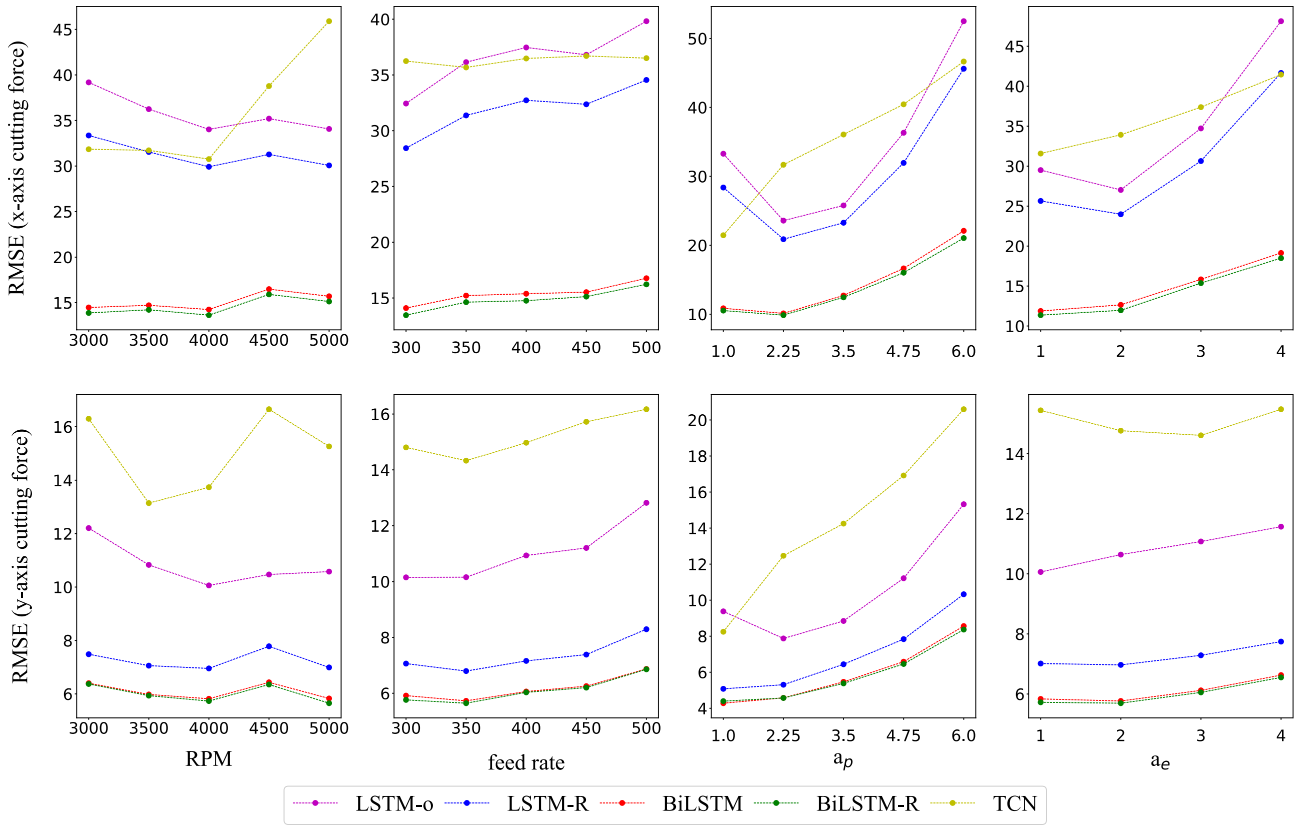


Fig. 7 RMSE evaluated for predicting the cutting force on each cutting parameter (with VM data)

LSTM-R without a bidirectional structure exhibited large deviations. BiLSTM-R outperformed other benchmark methods.

Figure 9 compares the actual *y*-axis cutting forces and predicted ones by the BiLSTM-R without VM. Compared to Fig. 1, the proposed method can predict the cutting force more accurately than VM because of various factors occurring in an actual machining process being considered, such as the vibration of tools. Furthermore, using only the data collected by MMS, the regression model could be an alternative method without using the data calculated by the mathematical simulations of VM.

5 Conclusion

A novel DNN-based cutting force prediction method was proposed for machining processes. Input variables were collected from MMS and VM, whereas the actual cutting force data were collected from a tool dynamometer. The input variables were preprocessed using frequency synchronization. We proposed an LSTM regression model with bidirectional and residual structures to predict the cutting force. Finally, we constructed a fully data-driven model without VM, and revealed that the proposed method outperformed the other benchmark methods. The proposed method has three main contributions: (1) application of modern DNN techniques including the stacked LSTM with bidirectional and residual structures, which have been rarely applied before for cutting force prediction in machining processes using only monitoring data of the time domain; (2) prediction of the cutting force in the time domain with high resolution; and (3) a fully data-driven method without mathematical simulation of the VM model for the cutting force prediction.

The experiments were conducted on data collected on SM45C. Each set of data of the machining process included various cutting parameters. Four LSTM-based models, an MLP, and a TCN model were evaluated. The experimental results revealed that the LSTM-based method with the bidirectional and residual structures achieved the best performance, even when it was trained without VM

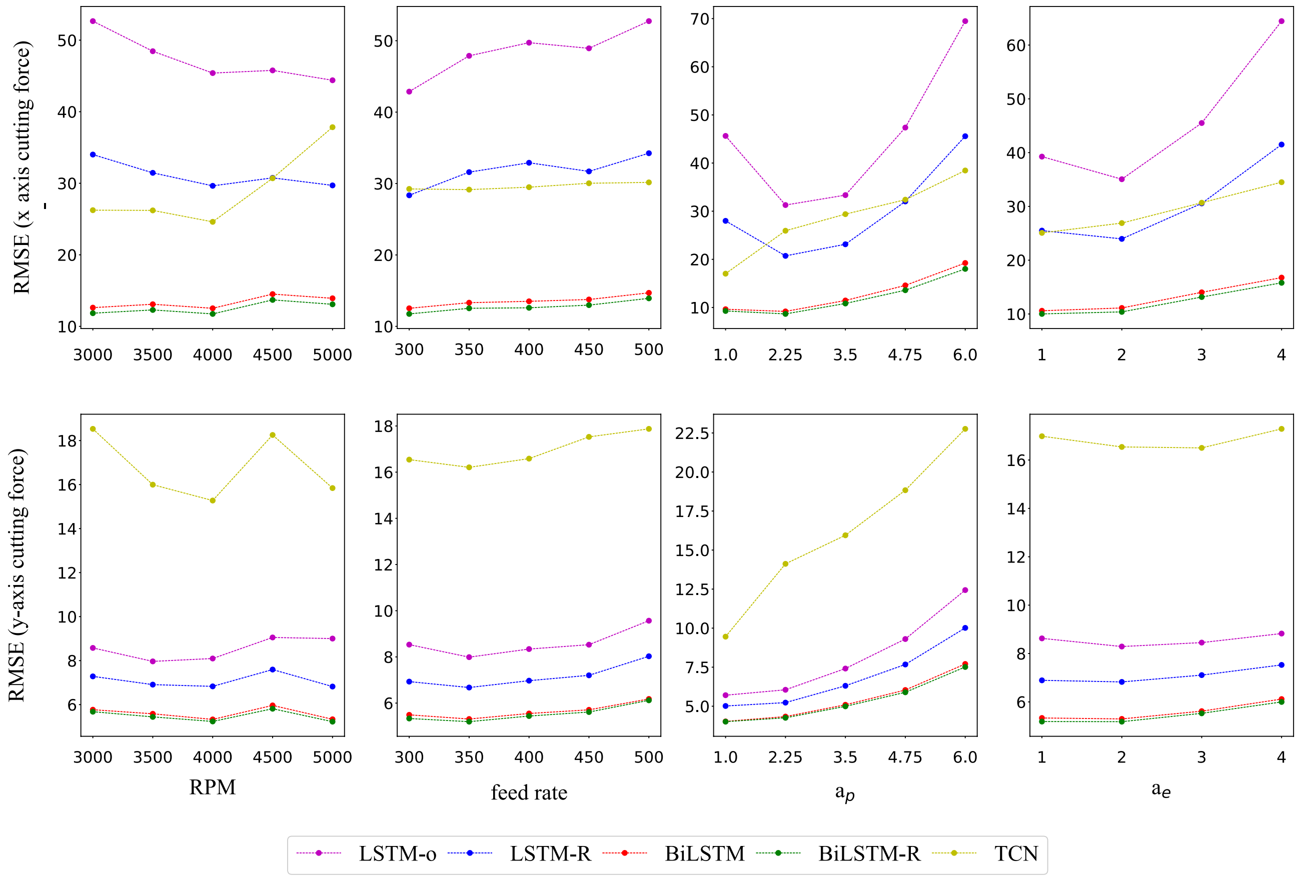


Fig. 8 RMSE evaluated for predicting the cutting force on each cutting parameter (without VM data)

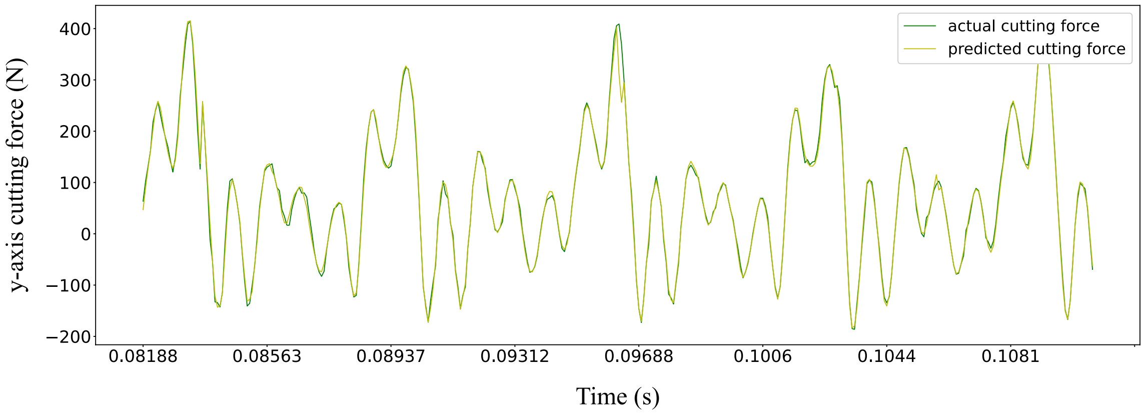


Fig. 9 Predicted cutting forces vs. actual cutting forces

data. Empirically, we confirmed that no additional effort, such as VM, was needed, and the monitoring data were sufficient to predict the cutting force.

Nevertheless, the proposed methods have some limitations, which can be overcome in future studies. First, the prediction errors tended to increase with an increase in the values of the cutting parameters. Therefore, the proposed method requires more data from large values of the process parameters to avoid extrapolation. We plan to reduce the error by adding another linear model that captures the overall trend of the cutting force. Second, other regression models or structures can be evaluated for comparison with the proposed method. Finally, the predicted cutting force at each point may indicate the amount of damage to a tool. With the predicted cutting force, the method can be extended to the prediction of tool life and breakage.

Data availability The datasets generated and analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

References

1. Chandrasekaran M, Muralidhar M, Krishna CM, Dixit U (2010) Application of soft computing techniques in machining performance prediction and optimization: a literature review. Int J Adv Manuf Technol 46(5):445–464
2. Ulsoy AG (2006) Monitoring and control of machining. Cond Monit Control Intell Manufact 1–32
3. Altintas Y (2012) Manufacturing automation: metal cutting mechanics, machine tool vibrations, and CNC design. Cambridge University Press, Cambridge
4. Moreira LC, Li W, Lu X, Fitzpatrick ME (2019) Supervision controller for real-time surface quality assurance in cnc machining using artificial intelligence. Comput Ind Eng 127:158–168
5. Soori M, Arezoo B, Habibi M (2013) Dimensional and geometrical errors of three-axis cnc milling machines in a virtual machining system. Comput Aided Des 45(11):1306–1313
6. Mourtzis D, Vlachou E, Zogopoulos V, Fotini X (2017) Integrated production and maintenance scheduling through machine monitoring and augmented reality: an industry 4.0 approach. In: IFIP International conference on advances in production management systems. Springer, pp 354–362
7. Sousa VF, Silva FJ, Fecheira JS, Lopes HM, Martinho RP, Casais RB, Ferreira LP (2020) Cutting forces assessment in CNC machining processes: a critical review. Sensors 20(16):4536
8. Kang I-S, Kim J-H, Hong C, Kim J-S (2010) Development and evaluation of tool dynamometer for measuring high frequency cutting forces in micro milling. Int J Precis Eng Manuf 11(6):817–821
9. Kadir AA, Xu X, Hämmerle E (2011) Virtual machine tools and virtual machining-a technological review. Robot Comput Integr Manuf 27(3):494–508
10. Barbosa JAG, Osorio JMA, Nieto EC (2014) Simulation and verification of parametric numerical control programs using a virtual machine tool. Prod Eng Res Dev 8(3):407–413
11. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521(7553):436–444
12. Shah D, Wang J, He QP (2020) Feature engineering in big data analytics for IOT-enabled smart manufacturing-comparison between deep learning and statistical learning. Comput Chem Eng 141:106970
13. Yoon S, Kang S (2022) Semi-automatic wafer map pattern classification with convolutional neural networks. Comput Ind Eng 166:107977
14. Okarma K, Fastowicz J (2020) Improved quality assessment of colour surfaces for additive manufacturing based on image entropy. Pattern Anal Appl 23(3):1035–1047
15. Kim D, Kang P, Lee S-K, Kang S, Doh S, Cho S (2015) Improvement of virtual metrology performance by removing metrology noises in a training dataset. Pattern Anal Appl 18(1):173–189
16. Vaishnav S, Agarwal A, Desai K (2020) Machine learning-based instantaneous cutting force model for end milling operation. J Intell Manuf 31(6):1353–1366
17. Wang J, Zou B, Liu M, Li Y, Ding H, Xue K (2021) Milling force prediction model based on transfer learning and neural network. J Intell Manuf 32:947–956
18. Xu K, Li Y, Zhang J, Chen G (2021) Forcenet: an offline cutting force prediction model based on neuro-physical learning approach. J Manuf Syst 61:1–15
19. Strafford K, Audy J (1997) Indirect monitoring of machinability in carbon steels by measurement of cutting forces. J Mater Process Technol 67(1–3):150–156
20. Adem KA, Fales R, El-Gizawy AS (2015) Identification of cutting force coefficients for the linear and nonlinear force models in end milling process using average forces and optimization technique methods. Int J Adv Manuf Technol 79(9):1671–1687
21. Lee P, Altintaş Y (1996) Prediction of ball-end milling forces from orthogonal cutting data. Int J Mach Tools Manuf 36(9):1059–1072
22. Lamikiz A, De Lacalle LL, Sanchez J, Salgado M (2004) Cutting force estimation in sculptured surface milling. Int J Mach Tools Manuf 44(14):1511–1526
23. Vargas B, Zapf M, Klose J, Zanger F, Schulze V (2019) Numerical modelling of cutting forces in gear skiving. Procedia CIRP 82:455–460
24. Han Z, Jin H, Fu H (2015) Cutting force prediction models of metal machining processes: a review. In: 2015 International conference on estimation, detection and information fusion (ICEDIF). IEEE, pp 323–328
25. Al-Zubaidi S, Ghani JA, Che Haron CH (2011) Application of ann in milling process: a review. Model Simul Eng 2011:1–7
26. Radhakrishnan T, Nandan U (2005) Milling force prediction using regression and neural networks. J Intell Manuf 16(1):93–102
27. Zuperl U, Cus F, Mursec B, Ploj T (2006) A generalized neural network model of ball-end milling force system. J Mater Process Technol 175(1–3):98–108
28. Rai JK, Villedieu L, Xirouchakis P (2008) Mill-cut: a neural network system for the prediction of thermo-mechanical loads induced in end-milling operations. Int J Adv Manuf Technol 37(3):256–264
29. Irgolic T, Cus F, Paulic M, Balic J (2014) Prediction of cutting forces with neural network by milling functionally graded material. Procedia Eng 69:804–813
30. Königs M, Wellmann F, Wiesch M, Epple A, Brecher C, Schmitt R, Schuh G (2017) A scalable, hybrid learning approach to process-parallel estimation of cutting forces in milling applications. Robert Schmitt Günther Schuh (Publ) 7:425–432
31. Peng B, Bergs T, Schraknepper D, Klocke F, Döbbeler B (2019) A hybrid approach using machine learning to predict the cutting forces under consideration of the tool wear. Procedia Cirp 82:302–307

32. Chen Y, Long W, Ma F, Zhang B (2009) Cutting force prediction of high-speed milling hardened steel based on bp neural networks. In: The sixth international symposium on neural networks (ISNN 2009). Springer, pp 571–577
33. El-Mounayri H, Briceno JF, Gadallah M (2010) A new artificial neural network approach to modeling ball-end milling. Int J Adv Manuf Technol 47(5):527–534
34. Gong X, Feng H-Y (2016) Cutter-workpiece engagement determination for general milling using triangle mesh modeling. J Comput Des Eng 3(2):151–160
35. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
36. Pascanu R, Mikolov T, Bengio Y (2013) On the difficulty of training recurrent neural networks. In: International conference on machine learning. PMLR, pp 1310–1318
37. Kandhare PG, Nakhmani A, Sirakov NM (2022) Deep learning for location prediction on noisy trajectories. Pattern Anal Appl 1–16
38. Albawi S, Mohammed TA, Al-Zawi S (2017) Understanding of a convolutional neural network. In: 2017 international conference on engineering and technology (ICET). IEEE, pp. 1–6
39. Oord AVD, Dieleman S, Zen H, Simonyan K, Vinyals O, Graves A, Kalchbrenner N, Senior A, Kavukcuoglu K (2016) Wavenet: a generative model for raw audio. arXiv preprint arXiv:1609.03499
40. Sharma DK, Brahmachari S, Singhal K, Gupta D (2022) Data driven predictive maintenance applications for industrial systems with temporal convolutional networks. Comput Ind Eng 169:108213
41. Hewage P, Trovati M, Pereira E, Behera A (2021) Deep learning-based effective fine-grained weather forecasting model. Pattern Anal Appl 24(1):343–366
42. Laboratory M.A CUTPRO, Advanced Milling Process Simulation System. https://www.malinc.com/products/cutpro/. Accessed 05 Jan 2021
43. Imrana Y, Xiang Y, Ali L, Abdul-Rauf Z (2021) A bidirectional LSTM deep learning approach for intrusion detection. Expert Syst Appl 185:115524
44. Ba JL, Kiros JR, Hinton GE (2016) Layer normalization. arXiv preprint arXiv:1607.06450
45. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp 770–778
46. He K, Zhang X, Ren S, Sun J (2016) Identity mappings in deep residual networks. In: European conference on computer vision. Springer, pp 630–645
47. Zhang YF, Thorburn PJ, Fitch P (2019) Multi-task temporal convolutional network for predicting water quality sensor data. In: International conference on neural information processing. Springer, pp 122–130
48. Nair V, Hinton GE (2010) Rectified linear units improve restricted boltzmann machines. In: ICML
49. Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. pp 448–456. http://jmlr.org/proceedings/papers/v37/ioffe15.pdf
50. Kim D, Kang P, Cho S, Lee H-J, Doh S (2012) Machine learning-based novelty detection for faulty wafer detection in semiconductor manufacturing. Expert Syst Appl 39(4):4075–4083

Publisher's Note I submitted my acknowledgement commnets as in the title page via the editorial manager. I think the acknowledgement is missing in this proof. So, I'd like to add the acknowledgment as below: This work was supported by Institute for Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2022-0-01200, Training Key Talents in Industrial Convergence Security), the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No.2020R1F1A1075781), and Korea Institute of Industrial Technology (Kitech EO-19-0043). Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.