Journal of Machine Engineering, 2023, Vol. 23, No. 4, 122–136

ISSN 1895-7595 (Print) ISSN 2391-8071 (Online)

Received: *31 March 2023* / Accepted: *11 October 2023* / Published online: *18 October 2023*

tool wear, feature extraction,
preprocessing,
recurrent neural networks

Fabio C. ZEGARRA[1],
Juan VARGAS-MACHUCA[1],
Alberto M. CORONADO[1*]

A COMPARATIVE STUDY OF CNN, LSTM, BiLSTM, AND GRU ARCHITECTURES FOR TOOL WEAR PREDICTION IN MILLING PROCESSES

Accurately predicting machine tool wear requires models capable of capturing complex, nonlinear interactions in multivariate time series inputs. Recurrent neural networks (RNNs) are well-suited to this task, owing to their memory mechanisms and capacity to construct highly complex models. In particular, LSTM, BiLSTM, and GRU architectures have shown promise in wear prediction. This study demonstrates that RNNs can automatically extract relevant information from time series data, resulting in highly precise wear models with minimal feature engineering. Notably, this approach avoids the need for excessively large window sizes of data points during model training, which would increase model complexity and processing time. Instead, this study proposes a procedure that achieves low prediction errors with window sizes as small as 100 data points. By employing Bayesian hyperparameter optimization and two preprocessing techniques (detrend and offset), RMSE errors consistently fall below 10. A key difference in this study is the use of boxplots to provide a better representation of result variability, as opposed to solely reporting the best values. The proposed approach matches more complex state-of-the-art methods and offers a powerful tool for wear prediction in engineering applications.

1. INTRODUCTION

Machine tools are essential for shaping solid workpieces, particularly those made of metal. Cutting tools play a crucial role in the milling process [1], but their wear increases with use due to various thermomechanical factors that they encounter [2]. Tool wear is a major contributor to machine tool downtime, accounting for up to 20% of such instances [3]. Dull or damaged cutters can negatively impact surface quality and result in unplanned downtime [4], [5] which can significantly increase manufacturing costs [6]. To address this issue, implementing an accurate and reliable Tool Condition Monitoring (TCM) system has been suggested. Such a system can help reduce costs by 10% to 40% and extend the useful life of milling tools by minimizing downtime [7]. Failure to monitor tool wear and accumulated damage can lead to tool breakage, accounting for 7% to 20% of total milling

[1] Laboratorio de Sistemas Inteligentes, EPIME, Universidad Nacional Tecnológica de Lima Sur (UNTELS), Peru

[*] E-mail:acoronado@untels.edu.pe

https://doi.org/10.36897/jme/174019

machine downtime [3, 7, 8]. Additionally, tool changes represent 3% to 12% of total processing costs [9]. Predicting tool wear in manufacturing processes with accuracy can lead to a range of benefits, including reduced costs, improved product quality, environmental gains, and enhanced worker safety. Additionally, accurate prediction can provide a competetive advantage in the marketplace [10].

Machining processes have been extensively studied, utilizing both direct and indirect monitoring methods [11]. Direct monitoring methods employ optical microscopes and image processing software, offering high measurement precision under ideal operating conditions [12]. However, these methods incur high costs due to the equipment required and the accuracy of the measurement can be impacted by the presence of cutting fluid and chips on the cutting tool surface [13, 14]. Indirect monitoring methods, in contrast, estimate cutting tool wear based on signals measured by one or more sensors. These signals may include force on the spindle, vibrations, motor current, and acoustic emission. By processing the corresponding time series, relevant features can be extracted, which can indicate the health state of the machine tool.

Over the past few years, significant advances have been made in applying deep learning methods to predict machine tool wear [15]. Deep learning is known for its ability to extract hierarchical representations from input data by constructing deep neural networks with multiple layers of nonlinear transformations. Various deep learning models have been developed, including denoising auto-encoder [16], bi-directional long short-term memory (BiLSTM) [17], transformer-based neural network [18], convolutional long short-term memory (ConvLSTM) [19, 20], gated recurrent units (GRUs) [21–23], among others.

Various published works have utilized deep learning algorithms trained on features extracted from time series measurements. The studies discussed below draw upon data from the 2010 PHM data challenge, where features were extracted from time and frequency domains. Time domain features such as RMS, variance, maximum, skewness, kurtosis, and peak-to-peak were extracted, alongside frequency domain features including spectral asymmetry, kurtosis, and spectral power. Researchers developed a bi-directional GRU network, which was applied to a locally extracted feature sequence, resulting in RMSE values of 5.4, 8.3, and 8.2 for datasets c1, c4, and c6, respectively [22]. Additionally, a deep heterogeneous GRU model was implemented, achieving RMSE values of 4.66, 8.73, and 6.94 for datasets c1, c4, and c6, respectively [21]. A physics-guided GRU model was also used, producing RMSE values of 5.009, 9.581, and 8.66 for datasets c1, c4, and c6, respectively [24]. Finally, a transformer-based neural network was trained for maximum wear prediction, yielding RMSE values of 6.116 and 9.553 for datasets c1 and c6, respectively [18].

In contrast to the studies mentioned above, some research works in deep learning networks utilize data without prior processing. The following papers also draw upon data from the 2010 PHM data challenge. A long-short term memory network (LSTM) model was developed to predict mean wear and obtained RMSE values of 12.1, 10.2, and 18.9 for datasets c1, c4, and c6, respectively [25]. Additionally, a time-distributed convolutional long-short term memory model (TDConvLSTM) was implemented to predict mean tool wear directly from raw multi-sensor time series data, achieving RMSE values of 8.33, 8.39, and 10.22 for datasets c1, c4, and c6, respectively [19]. A parallel convolutional neural network (PCNN), a deep residual network (DRN), and a bi-directional long-short term memory network (Bi-LSTM) were integrated to predict mean flank wear, resulting in RMSE values of 7.67, 7.84, and 8.6 for datasets c1, c4, and c6, respectively [26]. Convolutional bi-directional long-short term memory (CBLSTM) models were also used, where the convolutional component extracted local features representative of the sequential input, and a bi-directional LSTM encoded temporal information. A CBLSTM model was able to predict maximum tool wear based on raw sensory data of length 100 data points, with RMSE values of 10.8, 8.7, and 9.8 for datasets c1, c4, and c6, respectively [20].

Tool wear is a well-studied problem in the literature, and various feature extraction techniques have been used in conjunction with convolutional and recurrent neural networks in recent years. This work provides a detailed comparison of the prediction results obtained with the most commonly used neural architectures. In addition, relatively simple preprocessing techniques and Bayesian optimization of hyperparameters are introduced to obtain a better overview of the capabilities of each method. The article is organized as follows: Section 2 discusses the proposed methodology and presents the corresponding flowchart. Section 3 describes the experimental data used, which correspond to the PHM 2010 data challenge. Section 4 briefly describes the neural network models used in this study. In Section 5, the results obtained are presented and discussed, with a focus on comparing them not only with each other but also with other relevant results in the literature. Finally, Section 6 summarizes the conclusions.

2. PROPOSED METHODOLOGY

A key advantage of the present approach is that it eliminates the need for manual feature extraction before applying neural network models. The models are designed to process time series data and automatically identify relevant features for accurate predictions. As shown in Fig. 1, the wear prediction procedure incorporates several types of neural network models, including CNN, LSTM, BLSTM, and GRU.

Before training the model, several preprocessing operations were performed. The first operation applied was detrending, followed by offsetting. Detrending is achieved by creating a new time series where each element is calculated by subtracting two consecutive elements in time ($x_t(i) – x_t(i\text{-}1)$). On the other hand, offsetting is done by shifting the wear curves so that they all start at zero. Windows of 10, 25, 50, 100 and 250 points were then used to extract time series that were fed into the neural networks. Four types of time series were used in total: raw time series, detrended time series (without trend), offset time series (with compensation), and detrended and offset time series (detrend + offset).

During the training stage, Bayesian hyperparameter optimization was utilized to determine the optimal values for several hyperparameters. The search space included values for the number of neurons in layer 1 (100, 150, 200, 250, 300), layer 2 (100, 150, 200, 250, 300), dropout rate (0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8), and batch size (24, 32, 64, 128, 256, 512). According to previous tests, 500 epochs were found to be sufficient for most cases [27]. Several optimizers, including rmsprop, adam, adadelta, adagrad, adamax, and nadam, were tested. The Bayesian hyperparameter optimization method used 50 trials. All the previous steps were carried out 10 times (cross validation).

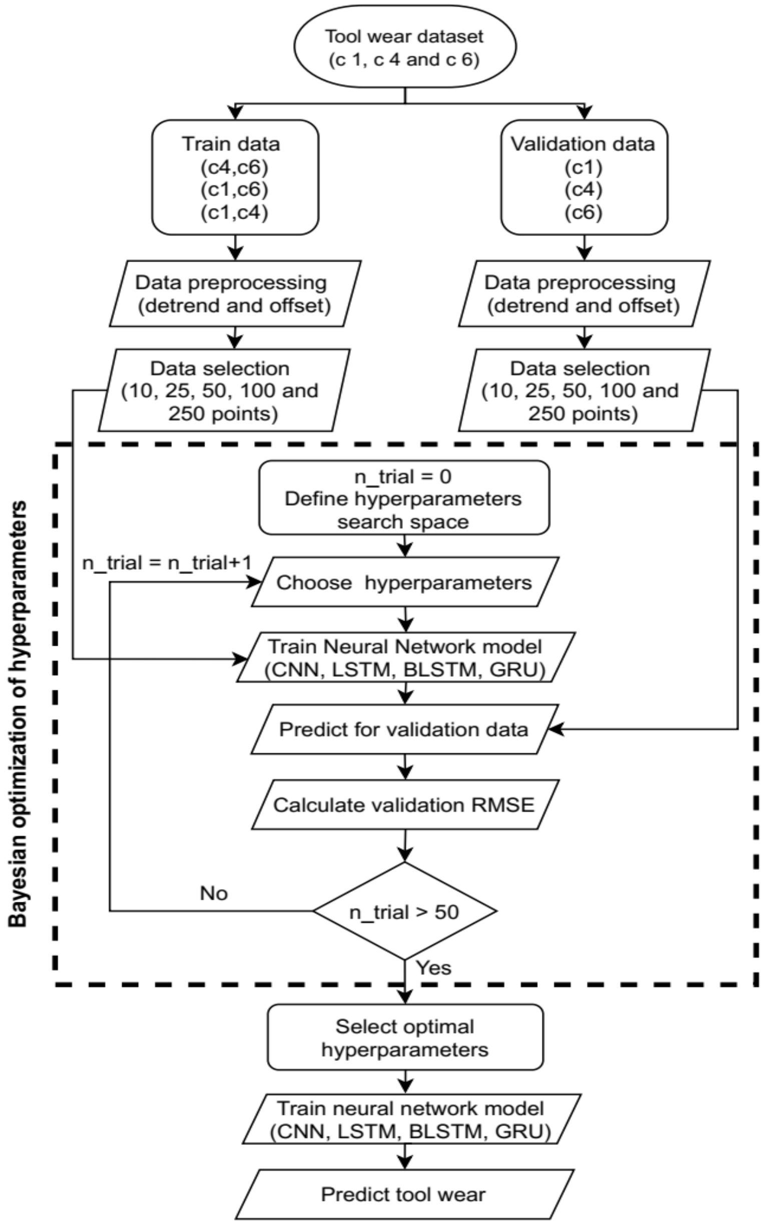

Fig. 1. Flowchart for the proposed tool wear prediction method

3. EXPERIMENTAL DATA

Tool wear refers to the gradual damage and material loss on the cutting edge of a tool during machining operations like milling. It is usually quantified by measuring the wear on the tool's flank face, which is the primary wear zone. This study uses experimental data [28] from the 2010 PHM data challenge to predict wear in a machine tool. The data was collected during a dry milling operation on a high-speed CNC machine, Röders Tech RFM760. A 6 mm ball point tungsten carbide tool with three flanks milled a stainless-steel workpiece in a down mill operation [18]. Figure 2 displays a diagram of the experimental setup. The workpiece was prepared by removing its original coating layer, which contained hard particles. The spindle speed, feed speed, and depth of cut were set to 23,600 rpm, 4.7 m/min, and 0.2 mm, respectively. The length of the workpiece along the feed direction was 108 mm.

During the milling process, cutting forces were measured using a 3-axis quartz platform Kistler dynamometer mounted between the workpiece and the machining table. Vibration accelerations in all three directions were measured using three Kistler piezoelectric accelerometers mounted on the workpiece. Additionally, a Kistler acoustic emission (AE) sensor was mounted also on the workpiece to measure high-frequency stress waves. The outputs of these sensors were captured by a NI DAQ PCI1200 at a sampling rate of 50 kHz.

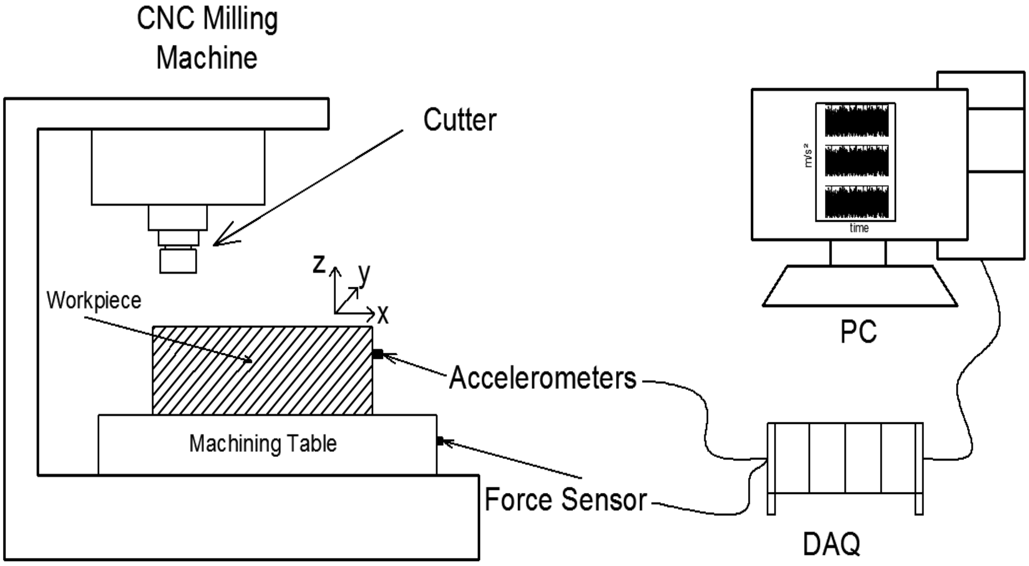

Fig. 2. Diagram of the experimental setup

During the tool wear test, seven time series were recorded: three for forces, three for vibration accelerations, and one for acoustic emission. A user-friendly graphical user interface (GUI) was created using the software National Instruments LabVIEW 8.2 displaying the real-time signals from each sensor during the milling operation. The wear of each flank was measured off-line using a microscope LEICA MZ12 after each surface was finished (each surface finish corresponds to a cut number). Three cutters (c1, c4, c6) were tested in a down milling operation, resulting in 315 data files (one for each cut).

4. MODELS – DEEP LEARNING

4.1. CNN

Convolutional Neural Networks (CNNs) were originally designed for computer vision tasks such as image classification [29]. Their name comes from the mathematical operation of convolution that they use. CNNs have since been applied to problems involving sequential or temporal data. A key feature of CNN-based models is their ability to directly extract complex features from raw data.

The time series data is fed into the convolutional layer where various convolution kernels or feature filters are applied to different local regions of the input to extract their features. These features are stored in a feature map and passed through an activation function. A pooling layer is then applied to reduce the dimensionality of the feature map and prevent overfitting.

4.2. LSTM

One significant variant of recurrent neural networks (RNNs) is the long short-term memory (LSTM) architecture. It addresses the vanishing gradient problem commonly found in traditional RNN models [30]. This issue makes it challenging for RNNs to retain long-term dependencies within a sequence.

LSTM models use three gates to regulate the flow of information within the network: an input gate, a forget gate, and an output gate. These gates utilize the previous hidden state ($\mathbf{h}_{t\text{-}1}$) and current input ($\mathbf{x}_t$) at time t, along with their respective weights and biases. The information is then passed through sigmoid layers corresponding to each gate. The flow of information is calculated using the following equations:

$$\mathbf{g}_t = \tanh\left(\mathbf{W}_g\mathbf{h}_{t-1} + \mathbf{U}_g\mathbf{x}_t + \mathbf{b}_g\right) \tag{1}$$

$$\mathbf{i}_t = \sigma(\mathbf{W}_i\mathbf{h}_{t-1} + \mathbf{U}_i\mathbf{x}_t + \mathbf{b}_i) \tag{2}$$

$$\mathbf{f}_t = \sigma\left(\mathbf{W}_f\mathbf{h}_{t-1} + \mathbf{U}_f\mathbf{x}_t + \mathbf{b}_f\right) \tag{3}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_o\mathbf{h}_{t-1} + \mathbf{U}_o\mathbf{x}_t + \mathbf{b}_o) \tag{4}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t \tag{5}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \tag{6}$$

In an LSTM cell, $\mathbf{g}_t$ represents the candidate hidden state, $\mathbf{i}_t$ is the input gate, $\mathbf{f}_t$ is the forget gate, $\mathbf{o}_t$ is the output gate, $\mathbf{c}_t$ is the memory cell at time t, and $\mathbf{h}_t$ is the hidden state at time t. Additionally, σ represents the sigmoid function, $\odot$ denotes the element-wise product (Hadamard product), $\mathbf{U}_g$, $\mathbf{U}_i$, $\mathbf{U}_f$, $\mathbf{U}_o$ are input weights, $\mathbf{W}_g$, $\mathbf{W}_i$, $\mathbf{W}_f$, $\mathbf{W}_o$ are recurrent weights, and $\mathbf{b}_g$, $\mathbf{b}_i$, $\mathbf{b}_f$, $\mathbf{b}_o$ are biases.

The first step within an LSTM cell involves passing the previous hidden state ($\mathbf{h}_{t\text{-}1}$) and current input ($\mathbf{x}_t$) at time t through the forget gate's ($\mathbf{f}_t$) sigmoid layer. Similarly, for the input gate ($\mathbf{i}_t$), ($\mathbf{h}_{t\text{-}1}$) and ($\mathbf{x}_t$) are passed through its corresponding sigmoid layer. The candidate hidden state ($\mathbf{g}_t$) is then calculated by passing $\mathbf{h}_{t\text{-}1}$ and $\mathbf{x}_t$ through a tanh function.

The forget gate ($\mathbf{f}_t$) controls how much of the previous cell state ($\mathbf{c}_{t\text{-}1}$) is retained or discarded by multiplying it with $\mathbf{c}_{t\text{-}1}$ using the $\odot$ operator. This result is added to the product of the input gate ($\mathbf{i}_t$) and candidate hidden state ($\mathbf{g}_t$) using the $\odot$ operator. The cell state ($\mathbf{c}_t$) at time t is updated by forgetting some of the previous hidden state and adding some of the current input while modulating how much of $\mathbf{c}_{t\text{-}1}$ is kept. The output gate ($\mathbf{o}_t$) then regulates which part of $\mathbf{c}_t$'s tanh is stored. This generates the new hidden state ($\mathbf{h}_t$).

4.3. BiLSTM

A bi-directional long short-term memory (BiLSTM) model is composed of two LSTMs. The first LSTM processes the input in the forward direction while the second LSTM processes it in the backward direction [31]. Each LSTM has its own input, forget, and output gates. The model parameters for each direction are independent. Despite the increased computational cost, this model can process information from both the past and future.

4.4. GRU

A Gated Recurrent Unit (GRU) [32] is a simplified version of an LSTM cell that can be obtained by making specific modifications. In a GRU cell, the flow of information is controlled by gates that differ from those used in an LSTM cell. Unlike an LSTM cell, a GRU cell does not have an output gate. Instead, it has a reset gate ($\mathbf{r}_t$) that controls which part of the old hidden state ($\mathbf{h}_{t\text{-}1}$) will be passed to the candidate hidden state ($\mathbf{g}_t$), modulating how much of the previous hidden state will be remembered. The update gate ($\mathbf{z}_t$) performs the information management of the forget and input gates of an LSTM cell by controlling how much of the old hidden state ($\mathbf{h}_{t\text{-}1}$) and how much of the candidate hidden state ($\mathbf{g}_t$) will be incorporated into the new state. The flow of information is determined by the following equations:

$$\mathbf{g}_t = \tanh\big(\mathbf{W}_g(\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{U}_g\mathbf{x}_t + \mathbf{b}_g\big) \tag{7}$$

$$\mathbf{z}_t = \sigma(\mathbf{W}_z\mathbf{h}_{t-1} + \mathbf{U}_z\mathbf{x}_t + \mathbf{b}_z) \tag{8}$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_r\mathbf{h}_{t-1} + \mathbf{U}_r\mathbf{x}_t + \mathbf{b}_r) \tag{9}$$

$$\mathbf{h}_t = \mathbf{z}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \odot \mathbf{g}_t \tag{10}$$

In a GRU model, the candidate hidden state is represented by $\mathbf{g}_t$, the reset gate by $\mathbf{r}_t$, and the update gate by $\mathbf{z}_t$. The sigmoid function is represented by σ and the element-wise product (Hadamard product) is represented by $\odot$. The hidden state at time t is represented by $\mathbf{h}_t$. The input weights are $\mathbf{U}_g$, $\mathbf{U}_z$, and $\mathbf{U}_r$ while the recurrent weights are $\mathbf{W}_g$, $\mathbf{W}_z$, and $\mathbf{W}_r$. The biases are $\mathbf{b}_g$, $\mathbf{b}_z$, and $\mathbf{b}_r$.

A GRU model has fewer gates to update and reset when calculating a hidden state compared to an LSTM model. This simplicity makes the GRU faster and more efficient.

5. RESULTS

5.1. TIME SERIES PREPROCESSING

To reduce processing times when training neural network models, only limited segments of the time series for forces (*Fx*, *Fy*, and *Fz*) and vibration accelerations (*Ax*, *Ay*, and *Az*) were used. These segments correspond to windows with lengths of 10, 25, 50, 100, and 250 data points. Figure 3 displays the entire raw data set of force and acceleration for cut number 150 of datasets c1, c4, and c6 in panels a and b, respectively. Panels c and d show a window of 100 data points in length extracted from the time series in panels a and b, respectively.

A previous investigation [33] assessed models by extracting features from all available time series, including acoustic emissions. However, it was discovered that features related to acoustic emissions did not significantly enhance predictions, and therefore, were not selected among the top features. As a result, the present study has opted to omit acoustic emissions from the feature extraction process to minimize computational effort required for hyper-parameter optimization. Although acoustic emissions did not improve predictions in the PHM 2010 dataset, it is possible that they may be useful in other datasets under different conditions.

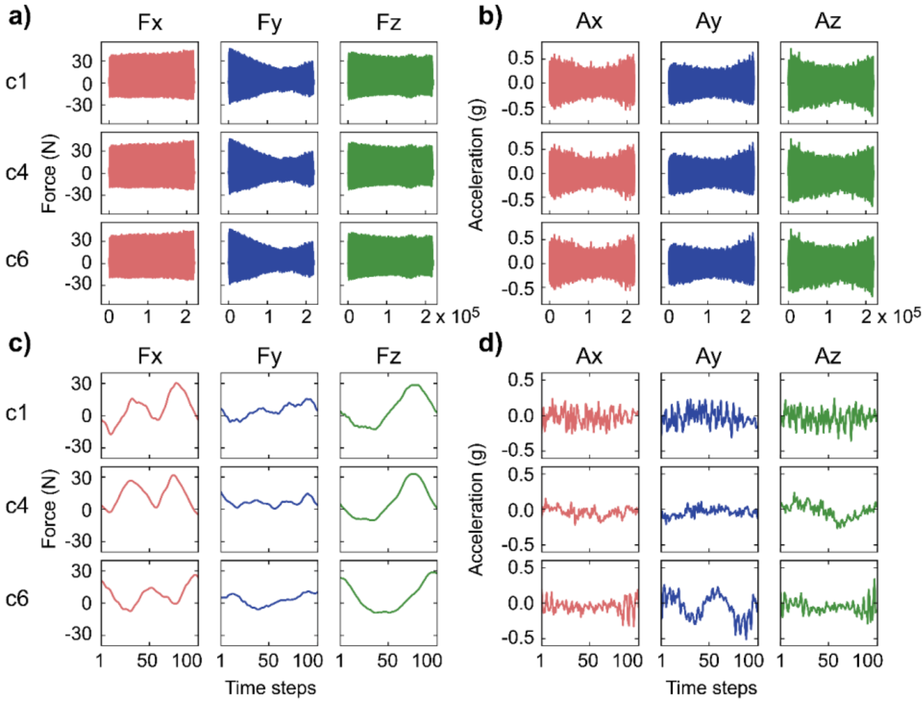

Fig. 3. Raw data of forces (*Fx*, *Fy*, and *Fz*) and vibration accelerations (*Ax*, *Ay*, and *Az*) for a single cut (top) and 100 time step windows (bottom). The data is presented for cut number 150

Tests show that the results are consistent, regardless of where the data extraction windows are located. This means that windows can be taken from any point in the time series without significantly affecting the prediction of tool wear. For the rest of this work, windows from the central part of each cut will be used.

Time series preprocessing, including detrending and offsetting, was used to create four data groups. The first group contains raw time series for forces and vibrations and original wear data (label: raw). The second group contains raw time series and offset wear data, where all three wear curves start at zero (label: offset). The third group includes detrended time series and original wear data (label: detrend). The fourth group consists of detrended time series and offset wear data (label: detrend + offset). To determine the best combination of preprocessing methods, several tests were conducted using the GRU model trained with time series of length 100 points. The four data groups were evaluated for three cutting tools (c1, c4, and c6) using Bayesian optimization of hyperparameters. Results are shown in Fig. 4. The fourth group (label: detrend + offset) generally achieved lower RMSE values for validation data. Thus, detrend + offset preprocessing methods were applied in all subsequent analyses.

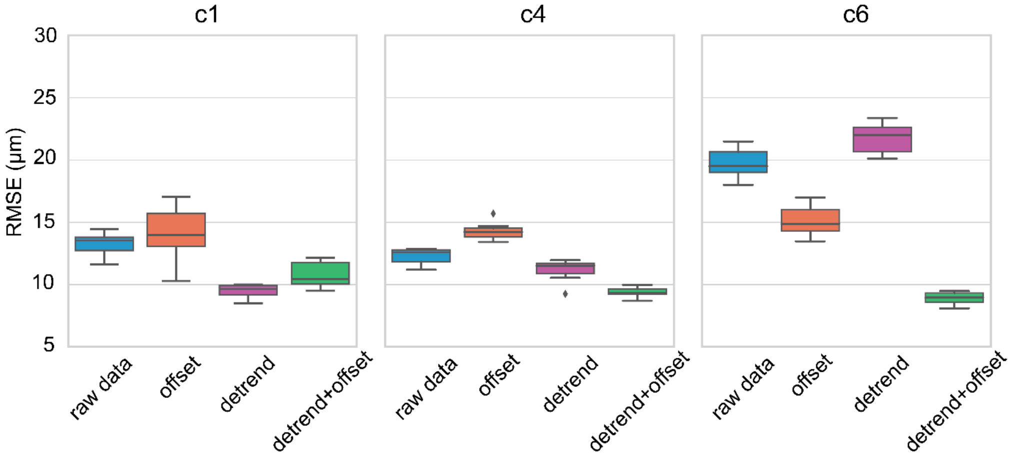

Fig. 4. Four data groups were used to compare preprocessing methods. A GRU model was trained on windows with 100 time steps

5.2. COMPARISON OF NEURAL NETWORK MODELS

Section 4 describes the neural network models used to predict maximum wear for the three edges of the cutting tool. These include CNN (convolutional neural network), LSTM (long-short term memory), BiLSTM (bidirectional long-short term memory), and GRU (gated recurrent unit). The models were trained with 500 epochs and a nadam optimizer. Bayesian hyperparameter optimization was conducted with 50 trials. All procedures were repeated 10 times for cross-validation.

Figure 5 provides a comparative analysis of the neural network models using datasets c1, c4, and c6. The dataset used for validation is indicated by the respective results for each dataset. For instance, the results for dataset c1 were obtained by training the models using datasets c4 and c6, and validating the models using c1. Likewise, for datasets c4 and c6. All the models were fed with time series data, which included forces and vibrations in any of the three axes. These time series data were extracted from windows that had varying numbers of time steps (10, 25, 50, 100 or 250).

In this study, the performance of each model was evaluated by measuring the root mean square error (RMSE) between the predicted and actual values. The results for each model, considering datasets c1, c4, and c6, are presented as boxplots in Fig. 5. Each boxplot includes 10 data points, corresponding to the cross-validation method used.

Upon analysing the results obtained for the CNN model, it can be observed that the c1 dataset exhibited the lowest RMSE values (with a median of approximately 11.5). Furthermore, the window size that yielded the lowest RMSE was 100 time steps. As for the LSTM model, the lowest RMSE values were observed for c4 and c6 datasets (with medians of approximately 9.5), and a window size of 100 time steps also resulted in the lowest RMSE.

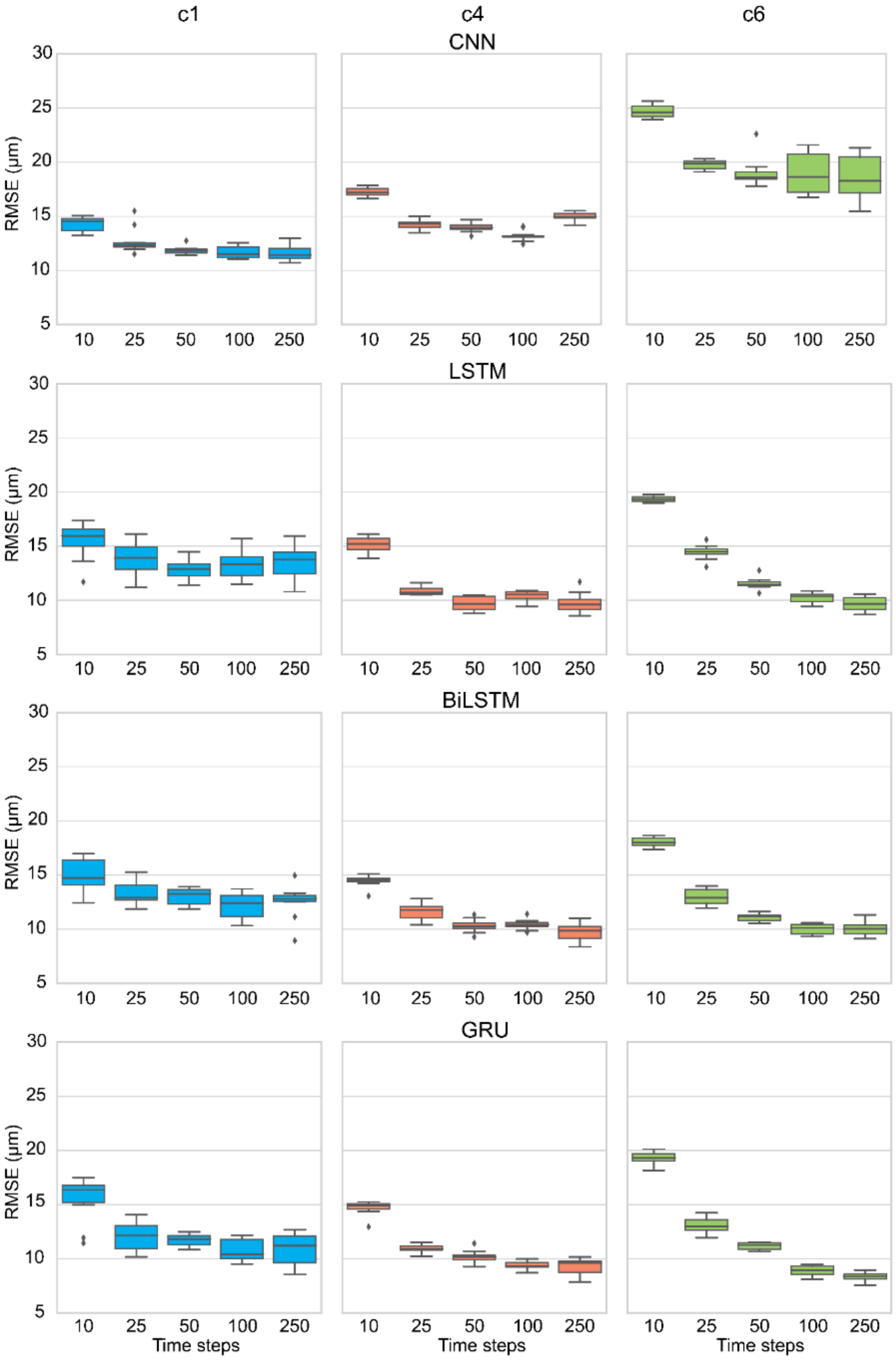

Fig. 5. Comparison of diverse neural network models (CNN, LSTM, BiLSTM and GRU) considering various window sizes (10, 25, 50, 100 and 250 time steps)

Similarly, the BiLSTM model displayed the lowest RMSE values for c4 and c6 datasets (with medians of approximately 10), and the window size of 100 time steps proved to be optimal. Finally, the GRU model demonstrated the lowest RMSE values for the c6 dataset (with a median of around 8.5), and the window size of 100 time steps was once again optimal. In conclusion, based on the overall results, it can be inferred that the GRU model performed the best for all three datasets.

5.3. TOOL WEAR COMPARISON

After analysing the results, it can be concluded that the optimal window size for achieving the best ones is 100 points when using the detrended time series, the offset tool wear, and the GRU model. To extract the hyperparameters corresponding to the first quartile (Q1), GRU boxplots shown in Fig. 5 were used, and the predictions were repeated 100 times using these hyperparameters. The resulting prediction curves for the zero quartile (Q0, lowest value), first quartile (Q1, corresponding to 25%), and second quartile (Q2, median) are shown in Fig. 6.

It is noteworthy that the wear in the cutting tool can be characterized by three regions: the break-in region, where the wear increases rapidly, the steady state region, characterized by a plateau, and the failure region, where wear increases rapidly again. The steady state and fault regions are more difficult to predict for dataset c1, while the steady state region is more difficult to predict for dataset c4. However, the prediction is generally good for dataset c6 since the curves practically overlap.

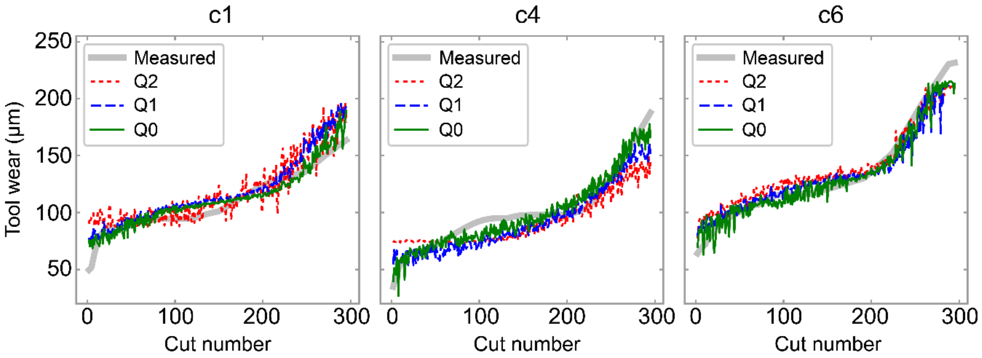

Fig. 6. Prediction of wear curves using time series with detrend and offset, with GRU model. The quartiles Q0, Q1 and Q2 are shown

The lower prediction accuracy during break-in can be improved by collecting more data specifically from that initial tool wear stage. This could involve dedicated cutting trials focused on the first few cuts, increased sampling rate in the break-in period, logging additional sensors, expanding the dataset with more tools exhibiting varying break-in, and synthetically augmenting the break-in data. With a larger, higher-resolution dataset capturing diverse break-in characteristics, the models can be retrained to better generalize across different break-in patterns and improve predictions in this transient phase.

Furthermore, it can be observed that the prediction curve corresponding to quartile zero (best prediction) most closely resembles the data measured for all three datasets.

5.4. COMPARISON WITH OTHER RESULTS IN THE LITERATURE

Table 1 summarizes the results obtained from selected studies that used the same data as the present study. It is worth noting that these studies utilized various neural network models to achieve their results. Notably, some of the studies achieved an RMSE of less than 10 for all three data sets considered (c1, c4, and c6). However, it is important to emphasize that many of these studies only reported their best result, and thus, it is not possible to fully appreciate the range of variability in the results. This limitation is addressed in the present study through the use of boxplots.

Table 1. Summary of the best predictions results (RMSE) obtained in selected studies

Method	RMSE		
	c1	c4	c6
bi-directional GRU [22]	5.4	8.3	8.2
deep heterogeneous GRU [21]	5.009	9.581	8.66
transformer-based neural network [18]	6.116	-	9.553
long short-term memory [25]	12.1	10.2	18.9
time-distributed ConvLSTM [19]	8.33	8.39	10.22
integration of parallel convolutional neural network (PCNN), deep residual networks (DRN) and Bi-LSTM [26]	7.67	7.84	8.6
convolutional bi-directional long short-term memory [20]	10.8	8.7	9.8

Table 2 displays the median RMSE values for the four models investigated in this study: CNN, LSTM, BiLSTM, and GRU. Despite utilizing the median RMSE rather than the lowest RMSE, the results obtained using GRU are comparable to the state-of-the-art outcomes summarized in Table 1. Furthermore, the relative RMSE (RRMSE), which is calculated by dividing the RMSE by the average wear measure, is less than 10% for the GRU model. These findings suggest that using only 100 time series points is sufficient to obtain results that are on par with the state-of-the-art outcomes.

Table 2. Summary of the medians of predictions results (RMSE and RRMSE) obtained in this work

Method	RMSE (RRMSE)		
	c1	c4	c6
CNN	11.53 (10.44%)	13.15 (13.11%)	18.62 (13.93%)
LSTM	13.33 (12.07%)	10.55 (10.52%)	10.37 (7.76%)
BiLSTM	12.42 (11.25%)	10.41 (10.38%)	10.13 (7.58%)
GRU	10.43 (9.45%)	9.32 (9.29%)	8.96 (6.70%)

5.5. APPLICABILITY OF THE METHODOLOGY PRESENTED IN OTHER CONTEXTS

A methodology similar to the one presented in this work has also been successfully applied to another database, Foxconn 2020, demonstrating its adaptability in various contexts. The Foxconn 2020 dataset has more complex tool paths than the PHM 2010 dataset, making prediction more challenging in some aspects [34]. Accurate predictions are achieved through the use of two primary techniques: preprocessing the time series and optimizing hyperparameters with Optuna. Preprocessing methods, such as detrending, help remove signal trends that affect both the PHM 2010 and Foxconn 2020 datasets [33]. Hyperparameter optimization with Optuna is crucial for improving prediction results, as it automatically searches for the best combination of hyperparameters that impact the performance of the machine learning models used.

The proposed methodology can also handle erroneous data, such as the force in the *X* direction of the c1 in the PHM 2010 dataset (Figure 3a from [33]), which shows a deviation between cuts 200 and 250. The preprocessing method detrending can help correct these erroneous data. Additionally, another study [35] found that the Foxconn 2020 dataset also contains measurement errors, which detrending can address, especially for vibration data. However, the applicability of detrending should be evaluated on a case-by-case basis, as it may not be effective in all situations.

In the PHM 2010 dataset [28], tool wear was measured directly by examining the flank face of the cutter under a microscope after interrupting the milling process. While accurate, this method is impractical for real-time monitoring in production, as it requires repeatedly stopping machining. Practical in-process estimation of tool wear requires indirect methods that can monitor changes in sensor data like forces and vibration during cutting. Although the wear measurements used in this study do not reflect real-world manufacturing conditions, the models developed provide a useful benchmark for future methods that can estimate flank wear in real-time from indirect sensor data.

6. CONCLUSIONS

This research aimed to predict the tool wear of a CNC milling machine using various neural network architectures, including CNN, LSTM, BiLSTM, and GRU. To optimize the performance of these models, the impact of two simple preprocessing techniques, detrend, and offset, was investigated. The results showed that implementing both preprocessing techniques significantly enhanced the predictions of the neural networks.

Furthermore, the study analysed the effect of the window size of the time series used for the neural network training. It is worth noting that increasing the amount of data (window size) also increases the processing time and model complexity. The findings indicated that all the models studied here achieved favourable prediction results with window sizes of around 100 data points, ensuring relatively shorter training times.

In addition, Bayesian hyperparameter optimization was successfully utilized to determine the optimal values of three hyperparameters, namely, the number of neurons in layers 1 and 2, as well as the batch size. The study also presented the variability of the obtained predictions through boxplots, which is in contrast to other studies that only present the best-performing value.

Ultimately, the GRU-based model demonstrated the best performance among all the models studied in this research.

ACKNOWLEDGEMENTS

This work was funded by CONCYTEC-FONDECYT in the framework of call E038-01, grant No. 020-2019-FONDECYT-BM-INC.INV.

REFERENCES

[1] KALPAKJIAN S., SCHMID S.R., 2013, *Manufacturing Engineering and Technology,* Prentice Hall International, Pearson/Prentice Hall.

[2] XU H., ZHANG C., HONG G.S., ZHOU J., HONG J., WOON K.S., 2018*, Gated Recurrent Units Based Neural Network for Tool Condition Monitoring*, International Joint Conference on Neural Networks (IJCNN), IEEE, https://doi.org/10.1109/ijcnn.2018.8489354.

[3] KURADA S., BRADLEY C., 1997*, A Review of Machine Vision Sensors for Tool Condition Monitoring*, Comput. Ind., 34/1, 55–72.

[4] JAVED K., GOURIVEAU R., Li X., ZERHOUNI N., 2018*, Tool Wear Monitoring and Prognostics Challenges: A Comparison of Connectionist Methods Toward an Adaptive Ensemble Model*, J. Intel. Manuf., 29/8, 1873–1890.

[5] REHORN A.G., JIANG J., ORBAN P.E., 2005, *State-of-the-art Methods and Results in Tool Condition Monitoring: A Review,* Int. J. Adv. Manuf. Technol., 26/7–8, 693–710.

[6] ZHOU Y., XUE W., 2018, *Review of Tool Condition Monitoring Methods in Milling Processes*, Int. J. Adv. Manuf. Techno*l*., 96/5–8, 2509–2523.

[7] KARANDIKAR J., McLEAY T., TURNER S., SCHMITZ T., 2015, *Tool Wear Monitoring Using Naïve Bayes Classifiers*, Int. J. Adv. Manuf. Technol., 77/9–12, 1613–1626.

[8] VETRICHELVAN G., SUNDARAM S., KUMARAN S.S., VELMURUGAN P., 2015, *An Investigation of Tool Wear Using Acoustic Emission and Genetic Algorithm*, J. Vib. Control, 21/15, 3061–3066.

[9] TETI R., 2002*, Machining of Composite Materials, CIRP Ann.*, 51/2, 611–634.

[10] KUNTOGLU M., et al., 2020*, A Review of Indirect Tool Condition Monitoring Systems and Decision-Making Methods in Turning: Critical Analysis and Trends*, Sensors, 21/1, 108.

[11] SICK B., 2002, *On-line* and *Indirect Tool Wear Monitoring in Turning with Artificial Neural Networks*: *a Review* of *More Than* a *Decade of Research*, Mechanical Systems and Signal Processing, 16/4, 487–546, https://doi.org/10.1006/mssp.2001.1460.

[12] DUTTA S., KANWAT A., PAL S.K., SEN R., 2013, *Correlation Study of Tool Flank Wear* with *Machined Surface Texture in end Milling*, Measurement:, 46/10, 4249–4260, https://doi.org/10.1016/j.measurement.2013.07.015.

[13] GHOSH N., et al., 2007, *Estimation of Tool Wear During CNC* milling *Using Neural Network-Based Sensor Fusion*, Mechanical Systems and Signal Processing, 21/1, 466–479, https://doi.org/10.1016/j.ymssp.2005.10.010.

[14] DROUILLET C., KARANDIKAR J., NATH C., JOURNEAUX A.-C., El MANSORI M.., KURFESS T., 2016, *Tool Life Predictions in Milling Using Spindle Power with the Neural Network Technique*, Journal of Manufacturing Processes:, 22, 161–168, https://doi.org/10.1016/j.jmapro.2016.03.010.

[15] LI X., LIU X., YUE C., LIANG S.Y., WANG L., 2022, *Systematic Review on Tool Breakage Monitoring Techniques in Machining Operations*, *International Journal of Machine Tools and Manufacture*, 176, https://doi.org/10.1016/j.ijmachtools.2022.103882.

[16] ZHAO R., YAN R., CHEN Z., MAO K., WANG P., GAO R.X., 2019, *Deep Learning and its Applications to Machine Health Monitoring*, Mech. Syst. Signal Process., 115, 213–237.

[17] HUANG C.-G., YIN X., HUANG H.-Z., LI Y.-F., 2020, *An Enhanced Deep Learning-Based Fusion Prognostic Method for RUL Prediction*, IEEE Trans. Reliab., 69/3, 1097–1109.

[18] LIU H., LIU Z., JIA W., LIN X., ZHANG S., 2020, *A Novel Transformer-Based Neural Network Model for Tool Wear Estimation*, Meas. Sci. Technol., 31/6, 065106.

[19] QIAO H., WANG T., WANG P., QIAO S., ZHANG L., 2018, *A Time-Distributed Spatiotemporal Feature Learning Method for Machine Health Monitoring with Multi-Sensor Time Series*, Sensors, 18/9, https://doi.org/10.3390/s18092932.

[20] ZHAO R., YAN R., WANG J., MAO K., 2017, *Learning to Monitor Machine Health with Convolutional Bi-Directional LSTM Networks*, Sensors , 17/2, https://doi.org/10.3390/s17020273.

[21] WANG J., YAN J., LI C., GAO R.X., ZHAO R., 2019, *Deep Heterogeneous GRU Model for Predictive Analytics in Smart Manufacturing*: *Application to Tool wear Prediction*, Comput. Ind., 111, 1–14.

[22] ZHAO R., WANG D., YAN R., MAO K., SHEN F., WANG J., 2018, *Machine Health Monitoring Using Local Feature-Based Gated Recurrent Unit Networks*, IEEE Trans. Ind. Electron., 65/2, 1539–1548.

[23] LI W., FU H., HAN Z., ZHANG X., JIN H., 2022, *Intelligent Tool Wear Prediction Based on Informer Encoder and Stacked Bidirectional Gated Recurrent Unit*, Robot. Comput. Integr. Manuf., 77, 102368, https://doi.org/ 10.1016/ j.rcim.2022.102368.

[24] WANG J., LI Y., ZHAO R., GAO R.X., 2020, *Physics Guided Neural Network for Machining Tool Wear Prediction*, J. Manuf. Syst., 57, 298–310.

[25] ZHAO R., WANG J., YAN R., MAO K., 2016, *Machine Health Monitoring with LSTM Networks*, 10th International Conference on Sensing Technology (ICST), IEEE, https://doi.org/10.1109/icsenst.2016.7796266.

[26] XU X., TAO Z., MING W., AN Q., CHEN M., 2020, *Intelligent Monitoring and Diagnostics Using a Novel Integrated Model Based on Deep Learning and* Multi-*Sensor Feature Fusion*, Measurement, 165/108086.

[27] ZEGARRA F.C., VARGAS-MACHUCA J., CORONADO A.M., 2021, *Comparison of CNN and CNN-LSTM Architectures for Tool Wear Estimation*, IEEE Engineering International Research Conference (EIRCON), https://doi.org/10.1109/eircon52903.2021.9613659.

[28] LI X. et al., 2009, *Fuzzy Neural Network Modelling for Tool Wear Estimation in Dry Milling Operation*, PHM_CONF, 1/1, Accessed: Aug. 07, 2021. Available: http://papers.phmsociety.org/index.php/phmconf/article/ view/1403.

[29] LeCUN Y., et al., 1989, *Handwritten Digit Recognition with a Back-Propagation Network*, in *Advances in Neural Information Processing Systems*, Available: https://proceedings.neurips.cc/paper/1989/file/53c3bce66e43be4f20 9556518c2fcb54-Paper.pdf.

[30] HOCHREITER S., SCHMIDHUBER J., 1997, *Long Short-Term Memory*, Neural Comput., 9/8, 1735–1780.

[31] SCHUSTER M., PALIWAL K.K., 1997, *Bidirectional Recurrent Neural Networks*, IEEE Trans. Signal Process., 45/11, 2673–2681.

[32] CHO K., et al., 2014, *Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation*, *arXiv Preprint arXiv:*1406.1078, Available: http://arxiv.org/abs/1406.1078.

[33] ZEGARRA F.C., VARGAS-MACHUCA J., CORONADO A.M., 2021, Tool wear and *Remaining Useful Life (RUL) Prediction Based on Reduced Feature Set and Bayesian Hyperparameter Optimization*, *Prod. Eng.*, 16/4, 465–480.

[34] ZEGARRA F.C., VARGAS-MACHUCA J., ROMAN-GONZALEZ A., CORONADO A.M., 2023, *An Application of Machine Learning Methods to Cutting Tool Path Clustering and RUL Estimation in Machining*, Journal of Machine Engineering, 23, https://doi.org/10.36897/jme/171432.

[35] CASUSOL A.J., ZEGARRA F.C., VARGAS-MACHUCA J., CORONADO A.M., 2021, *Optimal Window Size for the Extraction of Features for Tool Wear Estimation*, IEEE XXVIII International Conference on Electronics, Electrical Engineering and Computing (INTERCON), IEEE, Aug., https://doi.org/10.1109/intercon52678.2021. 9532759.