Received: 22 September 2022 | Revised: 7 November 2022 | Accepted: 8 November 2022

DOI: 10.1002/eng2.12598

RESEARCH ARTICLE

Engineering Reports Open Access WILEY

A transformer-based end-to-end data-driven model for multisensor time series monitoring of machine tool condition

Oroko Joanes Agung'[1] | **Kimotho James**[2] | **Kabini Samuel**[1] | **Murimi Evan**[1]

[1]Mechatronics Engineering Department, Jomo Kenyatta University of Agriculture and Technology, Nairobi, Kenya

[2]Mechanical Engineering Department, Jomo Kenyatta University of Agriculture and Technology, Nairobi, Kenya

Correspondence
Oroko Joanes Agung', Mechatronics Engineering Department, Jomo Kenyatta University of Agriculture and Technology, Nairobi, Kenya.
Email: agunghame@gmail.com

Funding information
African Development Bank Group; Jomo Kenyatta University of Agriculture and Technology

Abstract

Online determination of a cutter's health status is crucial for the attainment of condition-based automated tool change in computer numerically controlled (CNC) machining. Due to the impracticalities associated with direct condition measurements, data-based modeling of monitoring signals provides a viable practical route. However, the highly noisy and redundant nature of the associated data impacts negatively on model's accuracy and typically calls for additional initial preprocessing before modeling. Additionally, the long sequential data entails widely varying condition distributions exhibited by different cutters, even from the same batch on similar machining parameters, posing a challenge to model generalization. An end-to-end model has thus been developed to work directly on unprocessed data to establish global sensitive features from varying distributions for online tool wear estimation in CNC machining. The model utilizes three main functional blocks. First, a data denoising and feature selection block automatically processes raw multisensor data directly, dispensing with scaling or preprocessing of inputs as conventionally done. Each sensor channel's independence is preserved at initial processing ensuring complementary information from different sensors is utilized while simultaneously minimizing existing redundancies. The weighted denoised data is then processed through a transformer encoder block for determination of global dependencies in the time-series sequence, regardless of the time-step position. The learned features are then fed to an upper supervised learning block for association with the monitored wear condition. The developed model works directly on raw noisy data irrespective of scaling differences, saving on preprocessing computational cost. The global associations extracted on long sequences by the transformer-encoder allow for model generalization to varying wear distributions. The parallel processing structure of all channels ensures complementary information is utilized minimizing unforeseen model bias. The model's performance as evaluated on

experimental milling data and further comparison with other reported models on same dataset shows attainment of comparable state-of-art results.

KEYWORDS

deep-learning, end-to-end model, tool condition monitoring, transformer

1 | INTRODUCTION

The health condition of a machine cutter directly impacts the dimensional and surface integrity of the machined part.[1,2] Thus, in order to limit the overhead costs associated with part discarding and machine repair due to damaged cutters, tool condition monitoring is essential. Moreover, future planning on automated tool change based on health condition requires this process. This is especially critical for machining applications involving hard-to-machine materials or extended machining periods.

Direct measurement of a tool's wear condition during machining is not practical due to the constant tool–work interaction which would necessitate continuous intermittent stoppages for relevant measurements to be undertaken. Moreover, the presence of cutting fluid, inaccessibilty of the cutting zone, among many other factors, make it impossible to deploy direct visual-based measuring techniques. The studies reported in References 3-6 utilize visual systems for tool wear image processing in controlled settings but cannot be deployed in practical machining setups due to the aforementioned challenges. Tool condition monitoring (TCM) via modeling of in-direct sensory data and machining parameters provides the practical alternative.[7,8] Traditional physics-based models which relate tool condition to machining conditions, such as cutting speed and feed, have been developed.[9] These models though cannot be used for online TCM as they assume static conditions based on domain knowledge, making them inflexible to update. Moreover, such machining parameters as feed and speed do not change continuously during machining. Thus, data-driven models on sensor signals, independent of cutting parameters, provide a more viable option. The popularity of these models has also been driven by the advancement in storage, sensor and computing technologies.

The monitoring signals used by the data-based models is derived from such sensors as cutting force, vibration, acoustic emission, power, temperature, among many others. Upon data collection, the typical data-driven frame work[10] consists of the steps: feature extraction and reduction, data modeling, and then model prediction. The modeling step involves offline training of the model on historically collected monitoring signals, after which it can then be deployed for predictions on current sampled data. The information contained in a single monitoring sensor channel has temporal associations between the captured data. In order to develop a robust system, complementary redundant information from multiple sensors, that are sensitive to varying faults are utilized.[11-13] The side by side arrangement of the sensor data from multiple channels provides for spatial associations. The conventional data-driven approach utilizes human expert knowledge to manually extract features from the monitoring data for use in model training. The model is thus not trained directly on raw data. This significantly reduces the computational load at train time, and provided with discriminative relevant features, leads to enhanced model performance. Additionally, the sequential steps in the data-driven framework are treated as independent of each other which allows for the use of different optimization algorithms at the relevant stages. The studies in References 14-18 utilize this conventional approach employing tools such as artificial neural networks, support vector regression, hidden Markov models, relevance vector machines, among others. However, the reliance on expert knowledge not only provides an avenue for information loss but is also tedious owing to the huge volume of data usually involved, with no defined way of determining which data features to use. This generally impacts negatively on the models generalization capability and performance. Moreover, the model cannot be jointly optimized as a whole due to the independent stages.

Deep learning approach offers a solution to the weaknesses of the conventional data-driven framework through building of end-to-end models capable of working on raw data directly to provide the monitored condition.[19,20] These models automate the feature extraction and reduction stage, preserving information and generally leading to comparatively better model performance. Different deep architectures such as the deep auto-encoder (DAE), deep belief network (DBN), deep convolutional neural networks (DCNN), and deep recurrent networks have been employed for

condition monitoring tasks.[19,21,22] The DAE and DBN can utilize layerwise unsupervised pretraining allowing for training of a very deep fully connected structure on even a small data sample. However, owing to the large number of associated parameters the problem of model overfitting is easily experienced with such architectures and as such are not favored for the TCM task. The favored deep learning architecture for spatial information extraction is the DCNN, which works by passing a number of kernels across sequential data in order to learn important information at different parts in the data. In order to prevent model overfitting, pooling layers are usually utilized in conjuction in a cascaded structure. The weights sharing of convolutional layers reduces model parameters which eases the computational load at train time. However, CNN treats the data as static spatial arrangement thus the time dependency information is ignored. Thus, in order to learn temporal relations, the deep recurrent neural network (RNN) architecture is applied with the long short-term memory (LSTM) cell preferred due to its superior performance over the basic RNN cell.[20] The output of an LSTM cell is a function of its input and the output at the previous time step. The time dependency in sequential time-series data can thus be learned. However, LSTM cells have limited short-term memory thus cannot learn relations in very long sequences without performance drop while they also consequently lose spatial associations in a multisensory system. Different hybrid architectures have thus been developed to utilize the complementary power of CNN and LSTM while attempting to address the individual shortcomings. These involve various variations of CNN-LSTM layer pairings.[21] The CNN is generally used to extract spatial associations in the data while concurrently shortening sequences for use by the subsequent LSTM layers. This architecture allows for processing of long sequential data through stepwise shortening. However, the shortened sequences may not be discriminative enough thus the model's performance is negatively impacted as opposed to learning on comparatively longer sequences. Additionally, temporal convolutional networks (TCN) have also been developed utilizing dilated causal convolutions thus dispensing with the recurrent architectures.[19] The long-range temporal dependency problems still exist though even for this architecture.

In order to address the long-range dependency short comings, attention mechanisms were developed especially for networks utilizing encoder–decoder architectures specifically for natural language processing (NLP) tasks.[23] An attention mechanism is a neural network that learns to select only a valuable portion of the provided input that the model should focus on at each time step. This is achieved by differentially weighting each part of the input and paying attention on aggregated score. These mechanisms led to marked improvement on NLP tasks. However, they were initially used in conjunction with LSTMs and not a replacement. A novel improvement on the attention mechanisms which allows for dispensing off with the LSTMs is the transformer architecture.[24] The transformer uses multihead attention which consists of several attention layers running in parallel, which allows the model to jointly attend to information from different representation subspaces at different time steps. Its full typical structure comprises of an encoder–decoder architecture with positional encodings incorporated for enhanced processing of sequential data. The transformer is thus able to learn comparatively long range associations in sequential data completely outperforming the previously aforementioned preferred architectures. It is currently the state-of-art in NLP especially for machine translation tasks. Even though performance-wise superior in the reported tasks its been deployed, the transformer is computationally expensive owing to the huge volume of associated parameters. However, depending on the particular task at hand, different part-elements of its structure can be adopted for utilization.

The wear estimation task in a multi-sensor TCM system is thus faced with a couple of challenges. First, is the huge volume of data with high redundancy and noise. The complementary information provided from multiple sensor channels inexplicably contains redundancies while the significantly high sampling rates of data capture contributes to the elevated noise. These features have to be eliminated or minimized before or at model training as they negatively impact the accuracy of the trained model. Thus, the typical approach to deep modeling is to first preprocess the raw data by either standardization or normalization before use in model training to minimize impact of noise and redundant information. Secondly, different cutters, even from the same batch, exhibit varying wear distributions even on similar work pieces and machining parameters which poses a problem at modeling time as it provides for varying train-test distributions. Deep models provide enhanced performance results but require train-test data to be from similar or close distributions for this to be attained. This thus calls for a processing architecture capable of learning global associations in long sequential data as opposed to the relations in the particular ordering of time stamps which gets easily forgotten as the sequence gets longer. This allows for better model generalization capacity and provides an alternate processing technique for comparatively longer sequences. Finally, deep models preferential bias toward the more dominant signals in a multichannel input system negates the multisensory approach need in the first place as information from less dominant channels

are ignored. This results from the manner of backpropagation training in which the optimization algorithm penalizes the associated nodes of the less dominant input data more. Thus it calls for a data processing architecture that incorporates information flow from all input channels deep into the model. The work reported in this study addresses these challenges.

The developed model consists of three main parts based on functionality; data denoising and feature selection, transformer encoder, and supervised learning layer. Raw data from multiple sensors is first segmented automatically along the sensor channels preserving independence of each. This data is then passed to a linear encoding layer to provide for higher dimensionality in feature extraction. The output is then passed to a feature selection network (FSN) which utilizes gated residual networks (GRN) to permit the model to apply nonlinearity only where necessary. Data are processed through the GRN in parallel, both as individual independent channels and as a combined block. The result from the parallel processing is then differentially weighted, with the result being weighted denoised data. This output is then fed to a transformer encoder network which consists of a multihead attention block followed by a feed-forward network. The utilization of only the encoder structure without positional encodings incorporation significantly lowers the associated computational cost while preventing data overfitting by deploying a simplified structure. A global pooling layer is then utilized to obtain global features of the data which are then passed along to the final supervised learning layer block to provide the association between the global features and the monitored variable.

The main contributions in this paper include the use of the denoising network which enables the model to operate on raw data directly. This eliminates the conventional need to always first preprocess the input data before feeding to a deep model. The model can thus work directly on noisy redundant sensor data irrespective of difference in scale. Additionally, the transformer encoder block is able to learn global dependencies in time-series data without regard to position of time-stamp. This provides an alternative route for processing spatiotemporal associations in time-series data devoid of the challenges of gradient explosion and vanishing, and the short-term memory deficiencies associated with LSTMs. The weighted global association of relations should allow for better generalization at test time to cater for different train–test data distributions. Moreover, the parallel processing structure utilized for input data analysis deep into the model ensures information from all monitoring sensor channels is used and weighted appropriately minimizing bias. The rest of this paper is organized into the following sections; related work, theory, methodology, experiments and discussion, conclusion and finally references.

2 | RELATED WORK

2.1 | CNN, LSTM use in condition monitoring

The use of two-dimensional (2D) CNN to process raw time-series sensory information requires that the data either first be encoded into 2D spatial images or transformed to a time-invariant domain.[25-27] These approaches however add an extra computational overhead into the model. Alternatively instead, one-dimensional (1D) convolutional layers can be used to process raw time-series data directly.[28] They work by sliding multiple filters across 1D sensor channel data to generate relevant feature maps. Moreover, in order to capture time dependency, temporal convolution networks have been used, with the performance closely comparable to the more favored LSTM networks. LSTM networks provide comparative superior performance for temporal relations tasks.[29-31] However, the LSTM suffers long range dependency problems due to its small short-term memory. Thus, the use of hybrid CNN-LSTM architecture is favored.[32] The CNN is used for spatial information extraction while consequently shortening the sequences via pooling layers in a stacked configuration. This then allows the LSTM to process shorter sequences significantly improving model performance. In order to harness the power of convolutional and recurrent cells in one step, a convolutional-LSTM (ConvLSTM) can be used.[33,34] The ConvLSTM is simply an LSTM cell with the usual matrix multiplications replaced with the convolution operation. The consequence of this is a neural layer capable of learning both spatial and temporal information in one go. This minimizes the chances of information loss across different stages in the model.

In general, irrespective of the configuration, the aforementioned models require the data first be preprocessed for scaling purposes to enable gradients flow during back propagation at train time. Moreover, long-range dependency issues is still a problem that limits the model's ability to process long input sequences.

2.2 | Attention mechanisms use in condition monitoring

Attention mechanisms were originally developed to address limitations of context representations and long-range dependency deficiencies of LSTM-based encoder–decoder networks. This was specifically for machine translation and related NLP tasks. Although faced with a widely different data domain, these mechanisms have started finding limited use in the TCM task. In Reference 35, an attention layer is used in a hybrid CNN-Bidirectional LSTM model network to assign weights to each time step output of the BiLSTM layer in order to selectively filter and focus on critical information from a large number of output features before being fed to the upper supervised layer. Other recent works reporting use of attention for condition monitoring tasks is in Reference 36. On the other hand, the attention-only based transformer architecture has led to attainment of state-of-art accurate results in machine translation tasks and even in computer vision.[37] The use of this architecture in TCM is not widely reported to the best knowledge of this author, though one recent such work is in Reference 38. However, although performance-wise superior to other preceding architectures reported to date, the transformer is computationally expensive owing to the large number of parameters and structure.

The work reported in this paper uses a variation of only the encoder portion of the transformer architecture without positional encodings encompassment in order to harness the global attention prowess while minimizing the computational load.

3 | THEORY

3.1 | Convolution

The CNN is typically made up of a stack of convolutional, activation, and optional pooling layers in sequence. Each neuron in an upper convolutional layer is only connected to a small number of neurons in the previous layer, making up the receptive field. The convolutional layer operates by sliding multiple filters across an input and outputs one feature map per filter. The neurons in a feature map share the same parameters reducing the number of parameters in the model. The output of a neuron in a 2D convolutional layer is the weighted sum of all inputs plus a bias term as provided in Equation (1).[39]

$$z_k = \sum_{u=0}^{f_h-1} \sum_{u=0}^{f_w-1} \sum_{v=0}^{f'_n-1} x_{ijk} + b_k, \tag{1}$$

where f_h and f_w are the filter height and width, respectively, f'_n is the number of feature maps in the previous layer, x is the input vector, b_k is the bias term for feature map k, w_{uvk} is the connection weight between neurons in feature map k and input vector.

The 1D-convolutional layer operates much the same way as the 2D layer with the critical difference being in that the strided sliding shift from one receptive field to next is only along one direction. For time-series signal analysis, this is equivalent to processing only along the time dimension. The output of the convolutional layer is then usually nonlinearized through an activation function, the choice of which is one of rectified linear unit (ReLU) or its variants, hyperbolic tangent (tanh), or logistic function. If the dimension of the input sequence to a 1D-convolutional layer is $n \times l \times d$, where n is the number of data samples, l is the number of time steps, and d is the number of input channels, then the output dimension is given by $n \times \left(\frac{l-p+f}{s} + 1\right) \times k$, where p is the padding used, s is the stride and k is the number of filters used. Padding allows for preservation of sequence length dimension. This new representation of the data captures better abstract and informative knowledge than the original input representation. The pooling layer is optionally used to apply a sliding maximum or average window for sequence dimensionality reduction.

3.2 | Recurrence

The output y_t of a typical recurrent cell, such as an LSTM, is a function of its input x_t and the output at its previous time step h_t. The LSTM cell extends this simple functionality by addition of long-term memory. Figure 1[40] shows a typical

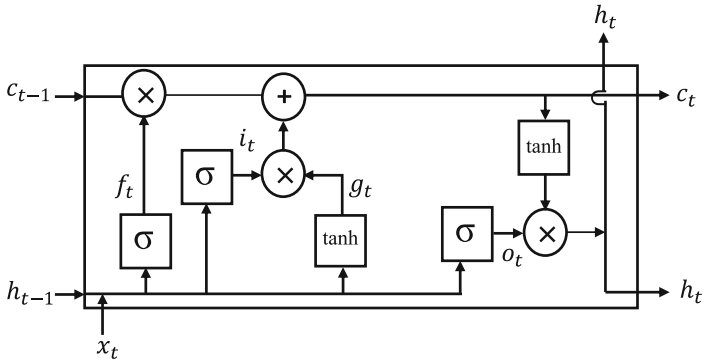

FIGURE 1 Schematic of an long short-term memory cell

LSTM cell. For a single instance of input data, the cell's long-term state, short-term state, and output at each time step are given by Equations (2).[40]

$$\begin{aligned} i_t &= \sigma(W_{xi}X_t + W_{hi}h_{t-1} + b_i) \\ f_t &= \sigma(W_{xf}X_t + W_{hf}h_{t-1} + b_f) \\ o_t &= \sigma(W_{xo}X_t + W_{ho}h_{t-1} + b_o) \\ g_t &= \tanh(W_{xg}X_t + W_{hg}(r_t \otimes h_{t-1}) + b_g) \\ c_t &= f_t \otimes c_{t-1} + i_t \otimes g_t \\ y_t &= h_t = o_t \otimes \tanh(c_t) \end{aligned} \tag{2}$$

where W is the weight matrices of each of the four layers, x_t is the input vector, b is the bias term for each of the four layers. The status of the long-term memory cell c_t is controlled via three gates. The input gate i_t controls which parts of the main output g_t should be added to the long-term state, whereas the forget gate f_t controls which parts are to be erased, with the output gate o_t controlling which parts of the long-term state should be read and output at this time step. The main layer g_t analyzes current inputs x_t and the previous short-term state h_{t-1}. An LSTM cell can learn to recognize an important input, store it in the long-term state, preserve it for as long as it needed, and retrieve it whenever needed.

3.3 | Attention

The attention mechanism permits a network to focus only on a portion of presented input representation at each time step. This is achieved via weighting the combination of all encoded input representations with the most significant vectors assigned the highest scores. This introduces an element of memory storage in the network as represented by the attention weights through time. Attention weights are calculated by normalizing the output score of a feed-forward neural network described by the function that captures the alignment between input and output element at each time step. Equation (3)[24] describes the attention operation.

$$h_t = \sum_i \alpha_{t,i} y_i, \tag{3}$$

with $\alpha_{t,i} = \text{softmax}(e_{t,i})$ and $e_{t,i} = a\left(s_{i-1}, h_j\right)$.

Initially, attention mechanisms were introduced to operate in conjunction with RNN and CNN. However, attention-only networks, such as the transformer, have been shown capable of capturing dependencies in sequential data dispensing with the aforementioned networks. The transformer architecture is as shown in Figure 2.[24]

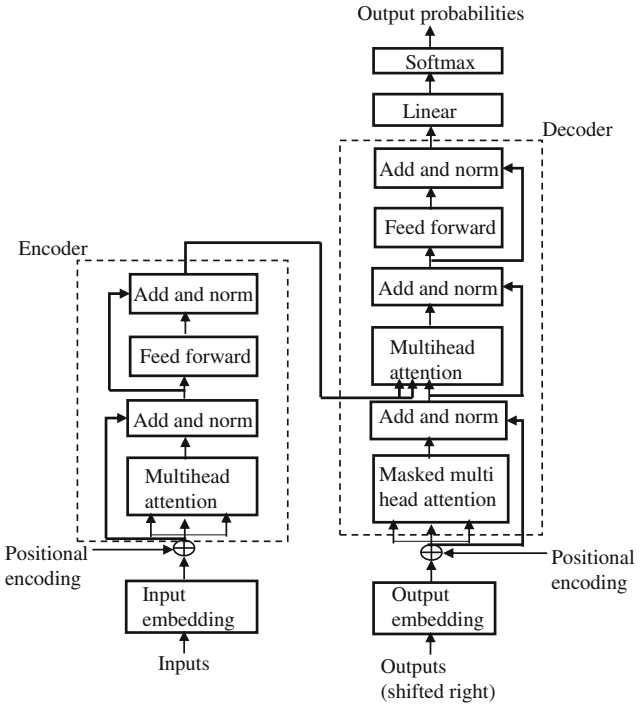

FIGURE 2 Schematic representation of the transformer architecture

The transformer utilizes separate multihead self attention in the encoder and decoder blocks as well as an encoder–decoder attention. Self-attention permits an input sequence to attend to itself thereby learning global dependencies between elements in the sequence without regard to position or order of time steps. Multihead runs multiple attention computations in parallel allowing for focus to be applied to same parts of a sequence differently learning different representations. The output of these parallel attention calculations are then combined to produce a final score.

Multihead attention is based on scaled dot product attention as described in Equation (4),[24] which uses a key, value and search query parameters.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_{\text{keys}}}}\right)V, \tag{4}$$

where Q, K, and V are the query, key and value matrices respectively. The query is used to search over keys of all context representations of the input elements. Each key is related to a particular value that encodes the specific input element.

The work in this paper utilizes only the transformer encoder portion of the architecture for determination of long-range global dependencies in time-series signals for the tool wear prediction task.

4 | METHODOLOGY

4.1 | Notation

The monitoring data for the tool wear prediction task is a time-series of real values from N different sensor channels, and is denoted $X = \{x_i, ..., x_L\}$, where L is the number of data samples. Each input data sample is a 2D tensor $x_i \in R^{l \times d}$ where l is the number of time steps and d is the sensor channels. At each time step j there are d different values. For each input sample, there is a corresponding output wear value $y \in R^3$ of three real values of flank wear width for each flute of the cutter. The wear monitoring task is thus formulated as a time-series regression prediction task of output value y for each input data sample x_i.

4.2 | Proposed model

The proposed model architecture is as shown in Figure 3. It comprises of three main blocks based on functionality, that is, data denoising and feature selection, transformer encoder, and supervised learning layer block. In a multisensor monitoring system, each independent channel provides useful information which is complementary to the others. The combined data though possesses significant redundancies. Additionally, the data capture is done at a high frequency for real-time monitoring success but this inadvertently results in elevated noise in the data. Moreover, different sensor channels are variedly scaled which leads to data outliers and scaling disproportionality. These elements need to be addressed as they not only make model training unstable, they also negatively impact performance. The initial denoising and feature selection block of the proposed model addresses these challenges, with additional benefits provided.

The main processing unit in this block is a GRN, whose configuration is as shown in Figure 3. It comprises of a stack of time-distributed fully connected neural layers, exponential linear unit activation, dropout, gated layer and a skip connection. The GRN permits the model to only apply data nonlinearization only where necessary. This enables the learning of both simple and complex data associations. The input data to this block is first segmented along the sensors channel in order to preserve each channels independence at initial processing. Each channel is then linearly encoded to increase data dimensionality for features determination. The encoded data is then fed to the FSN. The FSN parallel processes this data by independently applying GRN to each encoded channel while simultaneously doing the same to the concatenated combination followed by softmax weighting. The two outputs of the parallel processing are then differentially weighted to provide final result. The FSN allows the model to remove any unnecessary noisy inputs. The weighted output provides for a better representation of input data by minimizing redundancies.

The denoised weighted output of the first block is then fed into a transformer encoder block. The main purpose of this block is to determine global dependencies in the provided sequence. Multi-head self attention is utilized for sequence

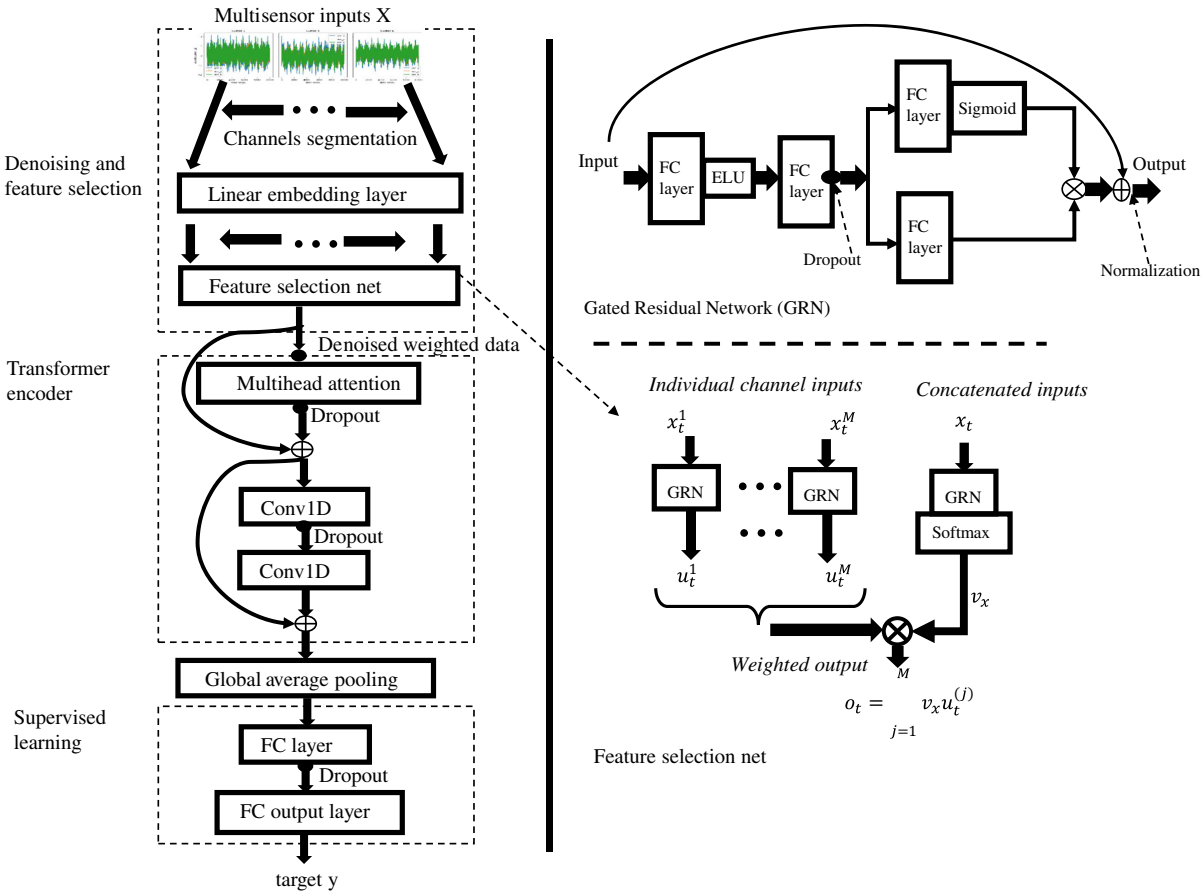

FIGURE 3 Schematic representation of developed model's architecture

representation. No positional encoding is used thus the associations determined are irrespective of the sequence time step order.

Finally, the features generated from the transformer encoder are fed to a supervised learning layer which comprises of two fully connected layers and a dropout layer. The output of the model is the corresponding wear value for each input data sample. Dropout is utilized to introduce randomness at train time to prevent data over-fitting.

4.2.1 | Data denoising and feature selection

The input data sample x_i of dimension $l \times d$ is first segmented along the d domain resulting in d tensors of shape $l \times 1$. Each $l \times 1$ input is linearly encoded with encoding size e a hyper-parameter for the encoding layer. The resulting encodings are d tensors of shape $l \times e$. These are fed in parallel to the FSN. The weighted output of FSN is a single dimension of size e for each instance of input data sample. The input data samples are fed into the model as batches thus weighted output of FSN is $b \times e$ regardless of the number of input features, where b is the batch size. The ELU activation function used in the GRN and the softmax in the FSN are provided by Equations (5)[39] and (6),[39] respectively;

$$\mathrm{ELU}_\alpha(x) = \begin{cases} \alpha(\exp(x) - 1) & \text{if } x < 0 \\ x & \text{if } x \geq 0 \end{cases}, \tag{5}$$

$$\mathrm{softmax}(x)_k = \frac{\exp(x)}{\sum_{j=1}^{k} \exp(x_j)}, \tag{6}$$

where x is the input vector.

4.2.2 | Transformer encoder

Due to limited computational resources available for model training in this work, the input sequence to the transformer encoder is first passed through a 1D convolutional layer for dimensionality reduction with the resultant sequence dimension being $b \times n_l \times e$, where n_l is the new sequence length. Data propagation through the transformer encoder results in features sequence of same dimension as input sequence. The hyper-parameters of choice for the transformer block are the number of heads, head size for the multihead attention layer and the number of transformer blocks.

4.2.3 | Supervised learning

The features tensor fed into this layer block passes through two fully connected layers to output three real values corresponding to predicted tool flank wear width. Linear activation function described by Equation (7)[39] is used in the output regression layer.

$$y = Wv + b, \tag{7}$$

where W is the connection weight, b the bias term associated with the layer and v the input features. The mean of the square of the error between the predicted values and the ground truth wear values is back propagated during model training for parameters adjustments.

5 | EXPERIMENTS AND DISCUSSION

In order to determine the effectiveness of the proposed model for the tool wear prediction task, its performance was tested on publicly available milling wear data from three monitoring sensors.

5.1 | Data description

The data used for the tool wear prediction task in this study is for a dry surface milling process, from the 2010 data challenge by the Prognostics and Health Management society. The monitoring signals are from force, vibration and acoustic emission sensors, with the first two having three channels each, for x, y, and z axes measurements. The input data thus comprise of seven channels. A Kistler quartz 3-component platform dynamometer was used for force measurements, with three Kistler piezo accelerometers and a Kistler acoustic emission sensor used for vibrations and acoustic emissions measurements respectively. The offline measured output is the flank wear width of three-flute ball nose tungsten carbide cutters, obtained through a LEICA MZ12 microscope after each milling cut run. A total of six cutters were used in the experiments, but only three cutter histories, labeled c1, c4, and c6, have both monitoring data and associated measured wear. A total of 315 cutting tests using each cutter, on a three-axis high-speed CNC machine, were conducted. The experimental setup used as described in Reference 41 is as shown in Figure 4.

The input data corresponding to one cut is considered a data sample. The time series measurements corresponding to different data samples vary in length, with some having over two hundred thousand time steps. In this study, the original data sequence length was down-sampled to a representative 20,000 time steps for each data sample before further applying a sliding window for attaining a shorter sequence length while concurrently increasing the training samples count. The sliding window was adopted due to the high frequency capture of the signals thus minimal wear exists across windowed cut samples. The data was acquired through a DAQ NI PCI1200 data acquisition card at a sampling frequency of 50 kHz/channel. The experimental measurements were obtained under constant machining conditions indicated in Table 1.

Due to the machining parameters being constant in the experiments, they were not considered for use in modeling in this study, since the model would be unable to capture any correlation with tool wear.

5.2 | Model settings

The sliding window size adopted determines the length and number of data samples downsampled from original data set and serves as an initial crucial hyperparameter. Too short a sequence and not much discriminative information can

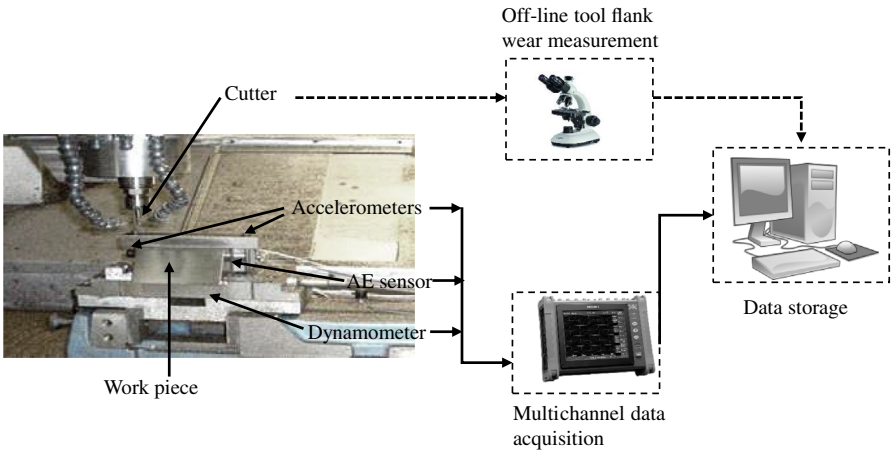

FIGURE 4 Schematic representation of experimental setup used in data collection

TABLE 1 Machining parameters

Parameter	Value	Units
Spindle speed, n	10,400	rpm
Feed rate, v_f	1555	mm/min
Radial depth of cut, a_e	0.125	mm
Axial depth of cut, a_p	0.2	mm

TABLE 2 Proposed model hyper-parameters

Model block	Hyper-parameter	Value
Denoising/feature selection	Encoding size	16
	Dropout rate	0.4
Transformer encoder	Head size	128
	Number of heads	4
	Conv1D filters	1×1
Supervised layer	FC nodes	32
	Output nodes	3
	Dropout rate	0.4

TABLE 3 Training/testing domain

Train set	Test set	Notation
C4, C6	C1	C4C6/C1
C1, C6	C4	C1C6/C4
C1, C4	C6	C1C4/C6

be derived whereas too long a sequence increases the computational processing load without much additional information captured. Values of 100, 200, 500, 1000, and 2000 sequential time-stamps were experimented on with results of cross-validated experiments used in window size selection. Values below 500 resulted in data samples with insufficient dicriminative information whereas higher values produced better results but at a price of enhanced computational load. The median value 500 was thus adopted. The increased training data samples was additionally utilized in minimizing model parameter uncertainty at train time. The summary of the main hyper-parameters, per functional block, for the proposed model are as indicated in Table 2.

Random experimentation on different values and datasets for sensitivity and performance for the main hyper-parameters was carried out with encoding size values of 16, 32, 64, and 128, transformer head size of 64, 128, 256, and 512, number of transformer heads of 1, 2, 4, and 8, and supervised layer nodes of 16, 32, 64, and 128. Higher value choices for each of the hyper-parameters significantly leads to models parameters count explosion which inadvertently risks data overfitting and increases computational processing load for model training and testing. Performance of value variations on different experimentations aided in selection. No joint model hyper-parameter optimization was performed, so the selection of optimal values is still an open avenue for this study. The low count of hyper-parameters required for model tuning simplifies the proposed model in operation. The training and testing regime adopted a three-fold setting whereby two sets, from histories $C1$, $C4$, and $C6$, are used for training with the third for testing. The adopted training/testing setup is as illustrated in Table 3.

The loss function utilized in model training is the mean squared error between ground truth and predicted values, as given by Equation (8).[39]

$$\text{loss} = \frac{1}{n}\sum_{i=1}^{n}\left|y_{\text{truth}_i} - y_{\text{pred}_i}\right|^2. \tag{8}$$

The adaptive momentum estimation (Adam) optimization function was used for model weight updates at train time, with an exponentially decaying learning rate from an initial value of 0.01. The choice of initial learning rate value was from random experimentation. The adopted indices for evaluating model performance were the mean absolute error (MAE) and mean absolute percentage error (MAPE) between the truth and predicted wear values, as given by Equations (9) and (10), respectively.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|y_{\text{truth}_i} - y_{\text{pred}_i}\right|. \tag{9}$$

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{y_{\text{truth}_i} - y_{\text{pred}_i}}{y_{\text{truth}_i}} \times 100\%. \tag{10}$$

The model was developed using Tensorflow Keras® deep learning library in Python® environment. The computing resource utilized was an Intel® Core i5 3GHz 4GB RAM CPU, with additional hardware acceleration provided via a GPU through the Google® Colab platform.

5.3 | RESULTS AND DISCUSSION

The performance of the proposed model under the three train/test regimes was evaluated. Its performance on wear progression determination for the three cutter histories as compared to the ground truth data is as shown in the wear progression plots of Figure 5.

It can be seen that the model is able to track the wear trends closely for the three cutter histories, to within an average MAPE boundary of 6%. The model's performance shows that the absolute error variation is comparatively slightly elevated during the initial rapid wear phase as compared to the constant and final failure phases. This however does not penalize the model negatively as crucial diagnostic and prognostic decision information such as condition-based tool change is taken in the final wear phase. The model's performance on different indices is as summarized in Table 4.

The model's performance on the MAE metric was further compared to three other reported models utilizing the same monitoring data. The models are the time distributed convolutional-LSTM (TDConvLSTM),[33] temporal convolutional network (TCN),[42] and bi-directional LSTM (BiLSTM).[43] The TDConvLSTM uses convolutional-LSTM layers for processing both spatial and temporal dependencies in the data in one layer rather than using two separate steps. The TCN on the other hand uses dilated convolutional layers utilizing causal padding to extract the time dependencies without peeping into the future. The BiLSTM on the other hand uses BiLSTM layers to learn time dependencies in sequences from both directions. The aforementioned models reported significant state-of-art results on the same

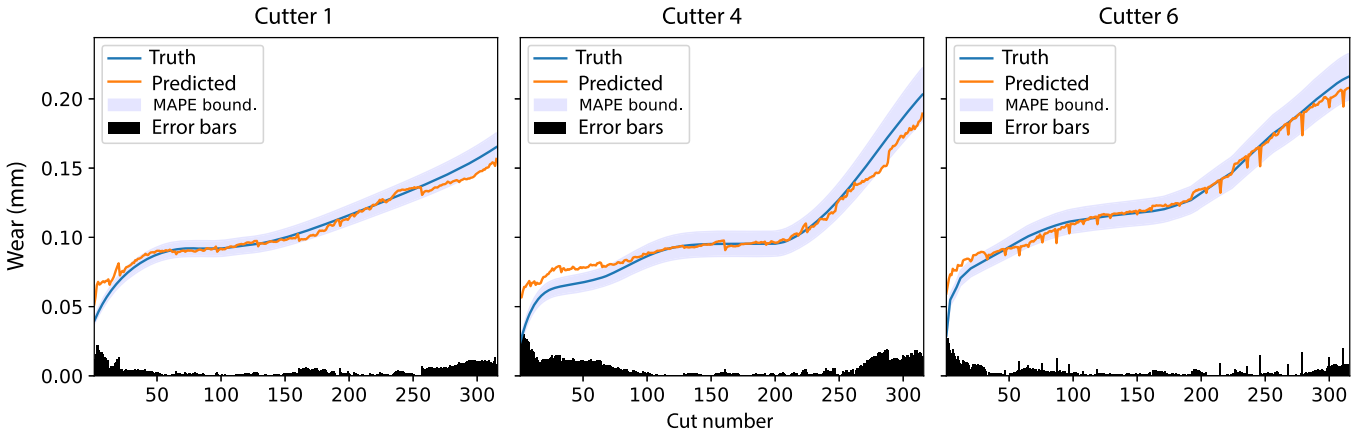

FIGURE 5 Regressive wear plots; predicted versus truth data

TABLE 4 Model performance evaluation on different indices

Index	C1	C4	C6	Units
MSE	8.0	10.9	16.1	$\times 10^{-5}\text{mm}^2$
RMSE	8.9	10.4	12.6	μm
MAE	5.7	7.3	8.5	μm

Abbreviations: MAE, mean absolute error; MSE, mean squared error; RMSE, root mean squared error.

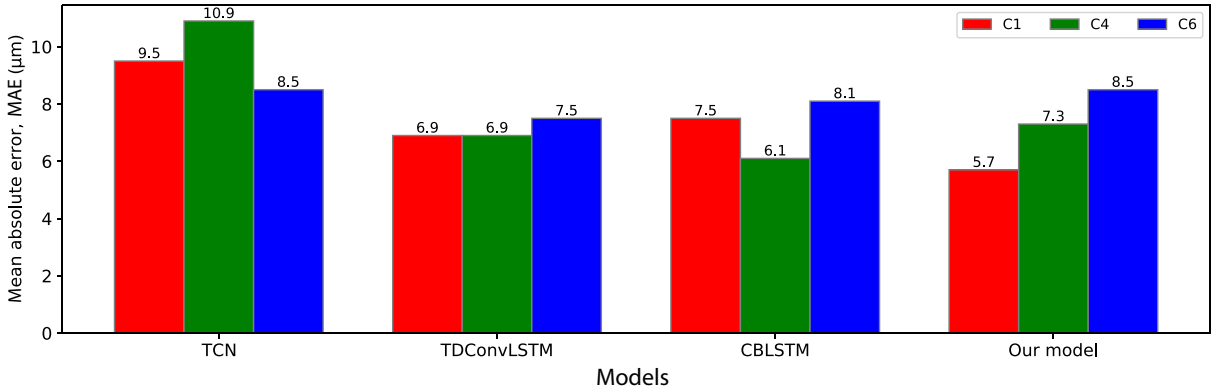

FIGURE 6 Evaluated mean absolute error comparison on different models

dataset while utilizing a similar train/test regime as used in this paper. This allows for a baseline validation comparison. The dataset used for training all the aforementioned baseline models was first preprocessed though before feeding to the model. The model developed in this study works directly on raw noisy data as input. The performance of the developed model versus the comparison models on the MAE metric is summarized using bar charts in Figure 6. It is seen that the model's performance is well comparable to other reported work. The elimination and minimization of noise and redundancies in raw sensory data coupled with parallel processing of all monitoring data for extraction of long-range global dependencies enabled the model's comparatively good performance. The attained results is without model hyper-parameters optimization being carried out, thus enhanced performance is expected with parameter optimization.

Even though the transformer encoder of the developed model does not utilize positional encodings for time stamp mapping, the model's performance proves that the extracted global dependencies of the processed time series feature vectors are a good abstract mapping of the input monitoring sensor signals, as compared to the favored LSTM or TCN networks, without the overhead computational costs. The processing of global relations as opposed to the short-term memory deficient recurrent approaches allows for automatic handling of comparatively long input sequences. Moreover, additional benefits of transformer use such as model interpretability, though not explored in this study, can now be harnessed. Additionally, prior inputs preprocessing is eliminated as evidenced by the denoising and scaling capabilities of the initial feature processing block. The model can thus be used for the real-time tool wear monitoring task.

6 | CONCLUSION

An end-to-end deep model has been developed for the tool wear monitoring task. The model has three main functional blocks, that is, data denoising and feature selection, transformer encoder, and supervised learning. The denoising and feature selection block enables the model to process raw multisensor data directly without need for preprocessing or scaling, as is conventional with deep models. On the other hand, the transformer encoder allows learning of global dependencies in a time-series sequence without regard to positional or time steps order. This provides a good alternative to the conventionally used recurrent networks. The supervised learning block is used to relate learned features to the monitored tool condition. The models' performance was evaluated on experimental data from a CNC milling process, with further validation involving results comparison with other reported models utilizing same data-set. The model is able to track tool flank wear within an average MAE of 5.7, 7.3, and 8.5μm for the three cutters under evaluation. The overall prediction accuracy derived from the MAPE metric translates to an average 93% across all the cutters considered. The performance attained is well comparable to other state-of-art results on the same dataset.

Future work will involve performing model hyper-parameters optimization and introduction of positional encoding for the transformer encoder in order to further associate sequential time stamp inputs, as well as model interpretability.

AUTHOR CONTRIBUTIONS

Oroko Joanes Agung': Conceptualization(equal); data analysis(lead); methodology(equal); software(lead); validation(equal); writing-oroginal draft(lead); writing-review and editing(equal). **Kimotho James**: Conceptualization(equal); data analysis(supporting); methodology(equal); supervision(lead); validation(equal); writing-review and editing(equal). **Kabini Samuel**: Conceptualization(equal); data analysis(supporting); methodology(equal); supervision(supporting); validation(equal); writing-review and editing(equal). **Murimi Evan**: Conceptualization(equal); data analysis(supporting); methodology(equal); supervision(supporting); validation(equal); writing-review and editing(equal).

ACKNOWLEDGMENTS

We sincerely thank the Google® Co., for providing access to GPU infrastructure needed for carrying out this research, and Jomo Kenyatta University of Agriculture and Technology (JKUAT) for facilitation of the study.

FUNDING INFORMATION

This research was funded by the financial support of the African Development Bank (AfDB) scholarship fund.

CONFLICT OF INTEREST

The authors declare no potential conflict of interests.

DATA AVAILABILITY STATEMENT

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

ORCID

Oroko Joanes Agung' https://orcid.org/0000-0002-0205-2985

REFERENCES

1. Akhavan Niaki F, Mears L. A comprehensive study on the effects of tool wear on surface roughness, dimensional integrity and residual stress in turning IN718 hard-to-machine alloy. *J Manuf Process*. 2017;30:268-280. doi:10.1016/j.jmapro.2017.09.016
2. Dutta S, Kanwat A, Pal SK, Sen R. Correlation study of tool flank wear with machined surface texture in end milling. *Measurement*. 2013;46(10):4249-4260. doi:10.1016/j.measurement.2013.07.015
3. Dai Y, Zhu K. A machine vision system for micro-milling tool condition monitoring. *Precis Eng*. 2018;52:183-191. doi:10.1016/j.precisioneng.2017.12.006
4. García-Ordás MT, Alegre E, González-Castro V, García-Olalla O, Barreiro J, Fernández-Abia AI. AZIBO shape descriptor for monitoring tool wear in milling. *Proc Eng*. 2015;132:958-965. doi:10.1016/j.proeng.2015.12.583
5. Mikołajczyk T, Nowicki K, Kłodowski A, Pimenov DY. Neural network approach for automatic image analysis of cutting edge wear. *Mech Syst Signal Process*. 2017;88:100-110. doi:10.1016/j.ymssp.2016.11.026
6. Zhang C, Zhang J. On-line tool wear measurement for ball-end milling cutter based on machine vision. *Comput Ind*. 2013;64(6):708-719. doi:10.1016/j.compind.2013.03.010
7. Zhou Y, Xue W. Review of tool condition monitoring methods in milling processes. *Int J Adv Manuf Technol*. 2018;96(5):2509-2523.
8. Ambhore N, Kamble D, Chinchanikar S, Wayal V. Tool condition monitoring system: a review. *Mater Today Proc*. 2015;2(4):3419-3428. doi:10.1016/j.matpr.2015.07.317
9. Johansson D, Hägglund S, Bushlya V, SJE. Assessment of commonly used tool life models in metal cutting. *Proc Manuf*. 2017;11:602-609. doi:10.1016/j.promfg.2017.07.154
10. Hassan M, Damir A, Attia H, Thomson V. Benchmarking of pattern recognition techniques for online tool wear detection. *Proc CIRP*. 2018;72:1451-1456. doi:10.1016/j.procir.2018.03.201
11. Lauro CH, Brandão LC, Baldo D, Reis RA, Davim JP. Monitoring and processing signal applied in machining processes – A review. *Measurement*. 2014;58:73-86. doi:10.1016/j.measurement.2014.08.035
12. Pimenov DY, Kumar Gupta M, da Silva LR, Kiran M, Khanna N, Krolczyk GM. Application of measurement systems in tool condition monitoring of milling: A review of measurement science approach. *Measurement*. 2022;199:111503. doi:10.1016/j.measurement.2022.111503
13. Wang J, Xie J, Zhao R, Zhang L, Duan L. Multisensory fusion based virtual tool wear sensing for ubiquitous manufacturing. *Robot Comput Integr Manuf*. 2017;45:47-58. doi:10.1016/j.rcim.2016.05.010.

14. Hesser DF, Markert B. Tool wear monitoring of a retrofitted CNC milling machine using artificial neural networks. *Manuf Lett*. 2019;19:1-4. doi:10.1016/j.mfglet.2018.11.001
15. Yu J, Liang S, Tang D, Liu H. A weighted hidden Markov model approach for continuous-state tool wear monitoring and tool life prediction. *Int J Adv Manuf Technol*. 2017;91(1):201-211.
16. Yang Y, Guo Y, Huang Z, et al. Research on the milling tool wear and life prediction by establishing an integrated predictive model. *MeasurementMeasurement*. 2019;145:178-189. doi:10.1016/j.measurement.2019.05.009
17. Xu G, Zhou H, Chen J. CNC internal data based incremental cost-sensitive support vector machine method for tool breakage monitoring in end milling. *Eng Appl Artif Intell*. 2018;74:90-103. doi:10.1016/j.engappai.2018.05.007
18. Ravikumar S, Ramachandran KI. Tool wear monitoring of multipoint cutting tool using sound signal features signals with machine learning techniques. *Mater Today Proc*. 2018;5(11):25720-25729. doi:10.1016/j.matpr.2018.11.014
19. Khan S, Yairi T. A review on the application of deep learning in system health management. *Mech Syst Signal Process*. 2018;107:241-265. doi:10.1016/j.ymssp.2017.11.024
20. Zhao R, Yan R, Chen Z, Mao K, Wang P, Gao RX. Deep learning and its applications to machine health monitoring. *Mech Syst Signal Process*. 2019;115:213-237. doi:10.1016/j.ymssp.2018.05.050
21. Liu R, Yang B, Zio E, Chen X. Artificial intelligence for fault diagnosis of rotating machinery: a review. *Mech Syst Signal Process*. 2018;108:33-47. doi:10.1016/j.ymssp.2018.02.016
22. Gouarir A, Martínez-Arellano G, Terrazas G, Benardos P, Ratchev S. In-process tool wear prediction system based on machine learning techniques and force analysis. *Proc CIRP*. 2018;77:501-504. doi:10.1016/j.procir.2018.08.253
23. Luong T, Pham H, Manning CD. Effective approaches to attention-based neural machine translation. Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing; 2015:1412-1421; Association for Computational Linguistics, Lisbon, Portugal.
24. Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. In: Guyon I, Von Luxburg U, Bengio S, eds. *Advances in Neural Information Processing Systems*. Curran Associates, Inc.; 2017.
25. Kothuru A, Nooka SP, Liu R. Application of deep visualization in CNN-based tool condition monitoring for end milling. *Proc Manuf*. 2019;34:995-1004. doi:10.1016/j.promfg.2019.06.096
26. Cao XC, Chen BQ, Yao B, He WP. Combining translation-invariant wavelet frames and convolutional neural network for intelligent tool wear state identification. *Comput Ind*. 2019;106:71-84. doi:10.1016/j.compind.2018.12.018
27. Wang P, Liu Z, Gao RX, Guo Y. Heterogeneous data-driven hybrid machine learning for tool condition prognosis. *CIRP Ann*. 2019;68(1):455-458. doi:10.1016/j.cirp.2019.03.007
28. Ince T, Kiranyaz S, Eren L, Askar M, Gabbouj M. Real-time motor fault detection by 1-D convolutional neural networks. *IEEE Trans Ind Electron*. 2016;63(11):7067-7075. doi:10.1109/TIE.2016.2582729
29. Wielgosz M, Skoczeń A, Mertik M. Using LSTM recurrent neural networks for monitoring the LHC superconducting magnets. *Nucl Instrum Methods Phys Res Sect A*. 2017;867:40-50. doi:10.1016/j.nima.2017.06.020
30. Zhao R, Wang D, Yan R, Mao K, Shen F, Wang J. Machine health monitoring using local feature-based gated recurrent unit networks. *IEEE Trans Ind Electron*. 2018;65(2):1539-1548. doi:10.1109/TIE.2017.2733438
31. Zhao R, Wang J, Yan R, Mao K. Machine health monitoring with LSTM networks. Proceedings of the 2016 10th International Conference on Sensing Technology (ICST); 2016:1-6.
32. Shi X, Chen Z, Wang H, Yeung DY, Wong WK, Woo WC. Convolutional LSTM network: a machine learning approach for precipitation nowcasting. Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1, NIPS'15; 2015:802-810; MIT Press, Cambridge, MA.
33. Qiao H, Wang T, Wang P, Qiao S, Zhang L. A time-distributed spatiotemporal feature learning method for machine health monitoring with multi-sensor time series. *Sensors*. 2018;18(9):2932. doi:10.3390/s18092932
34. Hall S, Newman ST, Loukaides E, Shokrani A. ConvLSTM deep learning signal prediction for forecasting bending moment for tool condition monitoring. *Proc CIRP*. 2022;107:1071-1076. doi:10.1016/j.procir.2022.05.110
35. Chen Q, Xie Q, Yuan Q, Huang H, Li Y. Research on a real-time monitoring method for the wear state of a tool based on a convolutional bidirectional LSTM model. *Symmetry*. 2019;11(10):1233. doi:10.3390/sym11101233
36. Zeng Y, Liu R, Liu X. A novel approach to tool condition monitoring based on multi-sensor data fusion imaging and an attention mechanism. *Meas Sci Technol*. 2021;32(5):055601. doi:10.1088/1361-6501/abea3f
37. Xu K, Ba J, Kiros R, et al. Show, attend and tell: neural image caption generation with visual attention. In: Bach F, Blei D, eds. *Proceedings of the 32nd International Conference on Machine Learning, Proceedings of Machine Learning Research*. Vol 37. PMLR; 2015: 2048-2057.
38. Liu H, Liu Z, JiaW LX, Zhang S. A novel transformer-based neural network model for tool wear estimation. *Meas Sci Technol*. 2020;31(6):065106. doi:10.1088/1361-6501/ab7282
39. Skansi S. *Introduction to Deep Learning*. Springer; 2018.
40. Sak H, Senior AW, Beaufays F. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. ArXiv. 2014. abs/1402.1128.
41. Li X, Lim B, Zhou J, et alFuzzy neural network modelling for tool wear estimation in dry milling operation. Proceedings of the Annual Conference of the PHM Society; 2009.

42. Van Herreweghe M, Verbeke M, Meert W, Jacobs T. A machine learning-based approach for predicting tool wear in industrial milling processes. In: Cellier P, Driessens K, eds. *Machine Learning and Knowledge Discovery in Databases*. Springer International Publishing; 2020:414-425.
43. Zhao R, Yan R, Wang J, Mao K. Learning to monitor machine health with convolutional bi-directional LSTM networks. *Sensors*. 2017;17(2):273. doi:10.3390/s17020273

AUTHOR BIOGRAPHY

Oroko Joanes Agung'. The corresponding author is a member of the academic staff in the department of Mechatronics Engineering at Jomo Kenyatta University of Agriculture and Technology, Nairobi, Kenya. His research interests are in computer numerically controlled machining, artificial intelligence and computer control.

SUPPORTING INFORMATION

Additional supporting information can be found online in the Supporting Information section at the end of this article.

How to cite this article: Joanes Agung' O, James K, Samuel K, Evan M. A transformer-based end-to-end data-driven model for multisensor time series monitoring of machine tool condition. *Engineering Reports*. 2022;e12598. doi: 10.1002/eng2.12598