The International Journal of Advanced Manufacturing Technology (2022) 118:817–836
https://doi.org/10.1007/s00170-021-07784-y

ORIGINAL ARTICLE

Cutting tool prognostics enabled by hybrid CNN-LSTM with transfer learning

Mohamed Marei[1] · Weidong Li[1,2]

Received: 28 March 2021 / Accepted: 23 July 2021 / Published online: 8 September 2021

Abstract
An effective strategy to predict the remaining useful life (RUL) of a cutting tool could maximise tool utilisation, optimise machining cost, and improve machining quality. In this paper, a novel approach, which is enabled by a hybrid CNN-LSTM (convolutional neural network-long short-term memory network) model with an embedded transfer learning mechanism, is designed for predicting the RUL of a cutting tool. The innovative characteristics of the approach are that the volume of datasets required for training the deep learning model for a cutting tool is alleviated by introducing the transfer learning mechanism, and the hybrid CNN-LSTM model is designed to improve the accuracy of the prediction. In specific, this approach, which takes multimodal data of a cutting tool as input, leverages a pre-trained ResNet-18 CNN model to extract features from visual inspection images of the cutting tool, the maximum mean discrepancy (MMD)-based transfer learning to adapt the trained model to the cutting tool, and a LSTM model to conduct the RUL prediction based on the image features aggregated with machining process parameters (MPPs). The performance of the approach is evaluated in terms of the root mean square error (RMS) and the mean absolute error (MAE). The results indicate the suitability of the approach for accurate wear and RUL prediction of cutting tools, enabling adaptive prognostics and health management (PHM) on cutting tools.

Keywords Remaining useful life · Cutting tool wear · Convolutional neural network (CNN) · Long short-term memory (LSTM) · Transfer learning

1 Introduction

Severer tool wear generates negative impacts on production quality and resource utilisation [1, 2]. During machining operations, a typical practice is to replace cutting tools in a regular period to avoid unexpected tool breakage and severer tool wear, which results in tool waste as replaced tools could still be within their remaining useful lives (RUL) [3]. As a result, the manufacturing cost per workpiece increases greatly due to the excessive number of cutting tools used and the time incurred in replacing worn tools. To address these issues, over the past decade, cutting tool prognostics to estimate their wear conditions and RUL, which is an important part of the prognostics and health management (PHM) for manufacturing assets, have been actively developed [4, 5]. PHM analytics play an increasingly important role in smart manufacturing, which offers critical benefits to real-time assessment of machine and component conditions in machining [6, 7].

In recent years, machine learning (ML) techniques, in particular deep learning (DL) models, have spurred interest in developing such cutting tool prognostics strategies that can exploit large amounts of historical and real-time data [8]. An opportunity to leverage DL models for cutting tool prognostics lies in the available image data that are typically collected for the purpose of visual inspection (VI) of workpieces and cutting tools [9]. While image-based wear estimation has been studied in the past, which can refer to [10] for an extensive review, recent research has sought to apply DL models to image data for wear and RUL prediction of cutting tools. However, in practices, a challenge in training supervised DL models arises from the limited availability of labelled image data for a cutting tool, prompting research into incorporating an effective transfer learning mechanism into DL models. Under transfer learning, a DL model can exploit learned

✉ Weidong Li
weidongli@whut.edu.cn

[1] Faculty of Engineering, Environment and Computing, Coventry University, Coventry, UK

[2] School of Logistics Engineering, Wuhan University of Technology, Wuhan, China

representations in a given source domain and apply this knowledge to a new task, i.e., a target domain. This is useful when labelled image data are insufficient for training the DL model accurately enough for predicting the RUL of a cutting tool. Moreover, it is also worth exploring how machining cutting parameters (i.e., MPPs) such as cutting speed, depth of cut, and feed rate, which affect the wear conditions and lifespans of cutting tools, could augment the image data to further improve the predictive accuracy of the DL model [11].

Motivated by these research foci, in this paper, a novel approach for cutting tool prognostics is developed. Taking multimodal data (VI images and MPPs) of a cutting tool as input, this approach leverages a pre-trained ResNet-18 Convolutional Neural Network (CNN) model to extract features from the VI images, a transfer learning mechanism to adapt the CNN model to a new cutting tool, and a long short-term memory network (LSTM) model to conduct the RUL prediction of the cutting tool based on the image features aggregated with the MPPs. The performance of the developed approach is evaluated and analysed to indicate the suitability of this approach for accurate and adaptive wear and RUL prediction of cutting tools.

The rest of this paper is organised as follows. Firstly, related works in cutting tool prognostics are reviewed in Section 2. In Section 3, the problem formulation and designed architecture for the proposed approach is presented, along with the dataset details. Experiments to validate the approach are described in Section 4. The results and corresponding analysis are highlighted in Section 5. Section 6 concludes the paper.

2 Related works

This section summarises a variety of tool wear and RUL prediction approaches, that is, physics-based and statistical PHM approaches. Furthermore, a review of DL models to predict tool health and estimate RUL is subsequently provided, highlighting issues faced with these approaches.

2.1 Overview of cutting tool PHM approaches

Cutting tools for machining processes undergo several degradation phenomena, such as abrasive wear, crater wear, built-up edge, and tool breakage, due to physical machine-workpiece interactions, frictional forces, and force distributions [12]. The RUL of a cutting tool can be measured as the residual time remaining to reach a specified failure threshold [13]. Flank wear is the most dominant failure mode, where the maximum flank wear width is usually used to define the failure criterion. Tool wear can be measured directly (i.e. measured from image data, collected in-process using high-speed cameras [14] or offline using microscopes), or indirectly (i.e. correlated to measured forces, vibrations, or acoustic emission data). Based on the data collected, tool wear characterisation and RUL prognostics have been studied extensively in the last several decades, with a range of modeling techniques explored. A summary of these approaches, highlighting their strengths and weaknesses, is presented in Table 1.

2.2 DL model-based RUL prediction

Traditional PHM methodologies (including some ML models) require feature engineering to extract meaningful degradation characteristics via physical or virtual health indicators from sensor data [35]. Quantities, such as signal peak, RMS and kurtosis, are statistical features commonly used [35]. As aforementioned, feature engineering requires expert knowledge to analyse data features, select those exhibiting prognostic qualities for RUL prediction, and develop models that learn from those features [36]. To mitigate the need for feature engineering, research into DL methodologies has recently dominated the PHM landscape, exploring feature learning for manufacturing process diagnostics, prognostics, quality control, and decision-making [7]. The reader is referred to [7, 11] for recent state-of-the-art reviews of DL methodologies for tool condition monitoring (TCM) and PHM.

CNN models For structured data such as images, CNN models have been considered the de facto predictive model class, for tasks such as image classification [37] and segmentation [38]. Recently, DL models have been increasingly applied for cutting tool life prediction from VI samples [39]. In contrast to force or dynamometer signals for in-process tool condition monitoring (TCM), charged couple device (CCD) cameras have been previously proposed for on-line cutting tool surface roughness and RUL prediction [9, 40], while others attempted to use microscopic images for offline prediction [7]. Training a CNN model to predict wear from sequence-based data (i.e. sensor signals) has also been explored, where time- and frequency-domain signals can be encoded into structured images using Gramian angular fields (GAF) [41], scalogram continuous wavelet transforms (CWT) [42], or spectrogram short time Fourier transform (STFT) [43], which can be input into the CNN model [44]. Classification and segmentation methods are common for tool wear assessment. For instance, Wu et al. [25] proposed a convolutional automatic encoder (CAE) approach based on the VGG-16 architecture. The CAE classified each input image into one of four wear classes, followed by cascading image processing operations for tool wear estimation. Despite the CNN model’s overfitting tendency, a test classification accuracy of 96.20%, a test error rate of 1.5–7.4%, and mean absolute percentage error (MAPE) of 4.76%, were reported. Recently, a U-Net CNN was used in [27] for semantic image segmentation of tool wear, using four manually selected semantic class labels (undamaged tool body, built-up edge, groove, flank wear region, and

Table 1 Summary of the methodologies for cutting tool wear and RUL prediction

Modelling Process	Strengths	Weaknesses
Physics-based degradation/lifetime models, e.g. modified Taylor (tool life with MPPs); Coromant [2]; Sipos (flank wear and MPPs) [15]; flank wear relationship to cutting force and MPPs [12]	Highly accurate simulations can be used to predict MPPs' impact on cutting tool life; suitable when MPPs are constant	Need to specify failure thresholds relative to failure time; costly to develop sufficient trials with varied experimental MPPs; difficult in estimation of unknown model parameters
Statistical methods *Without covariates*, e.g. Bayesian models (i.e. GPR [16], PF [17, 18], Wiener process [19]) *with covariates*, e.g. proportional hazards models (PH) [20]	Uncertainty bounds for prediction; modelling different degradation process distributions a-priori; deal with MPPs as weighted covariates; assume flexible parameter distributions on covariates; uncertainty bounds for predictions	Choices of model structure can be difficult to develop when underlying process is unknown; feature engineering required on raw data; censored observations require special attentions in modelling process
Machine learning approaches, e.g. linear/nonlinear regression [21], SVM/R [22], random forest [23], fuzzy methods [14]	Capable of accurate predictions with smaller datasets	Raw signals almost always subject to feature engineering or dimensionality reduction; no RUL probability distributions
Deep learning approaches, e.g. back-propagation neural network (BPNN) [24]; CNN (wear classification [25]; wear prediction [26]; wear segmentation [27]); wavelet neural network (WNN) [28]; LSTM (tool RUL prediction); CNN-LSTM [29–31]; GRU [32]; autoencoders [33, 34]	End-to-end trainable with feature learning mechanisms (i.e. no feature engineering); learning from heterogeneous data; generalising across huge datasets (millions of parameters); transfer learning; multi-task approaches; supervised/unsupervised training	Model training resources are difficult for collection (require sufficient labelled data for supervised learning; computation); model overfitting on smaller datasets; regarded as black-box models, so inherently difficult to troubleshoot; negative transfer

background) with an accuracy of 91.5% compared to a state-of-the-art SegNet model's accuracy of 89.9%.

Recurrent network models Despite their widespread success in many applications, CNN is not inherently specialised to handle sequence-based data with explicit time dependencies and potentially varying sequence lengths and are thus prone to vanishing and exploding gradients [45]. Hence, research interest in recurrent network architectures, such as recurrent neural networks (RNN) and LSTM [46], has motivated their application for RUL prediction for cutting tools. More recently, extensions of CNN such as CNN-LSTM with an attention mechanism were used to estimate the RUL of cutting wheels (they are similar to the working mechanisms of cutting tools), which are partially obscured within the machine (i.e., partially observed) [39]. To date, the success of recurrent networks, however, has relied on larger datasets with thousands of observations per sequence.

Transfer learning To address the need for massive labelled datasets for DL model training, DTL seeks to use domain-invariant features for learning novel predictive tasks [47]. Re-purposable CNN-based models for DTL are a mainstream research interest [48–50], whereupon they can be trained on millions of examples within a source task (e.g. image classification), and fine-tuned on a possibly unrelated, downstream predictive task (i.e., target task). However, a common problem in DTL is *negative transfer*, caused when a misalignment between source and target domain marginal feature distributions occurs [18, 35], undermining the performance of the learner [48]. Thus, strategies have been explored to discover an intermediate feature latent space where the distribution discrepancy can be minimised, with feature-based methods such as maximum mean discrepancy (MMD) [51], correlation alignment (CORAL) [52], and transfer component analysis (TCA) [53]. As a sub-problem in DTL, domain adaptation (DA) scenarios emerge where the task space is identical for source and target data, but the input domain is different [52, 54]. DA is advantageous in PHM where source and target data distributions vary due to the nature of the data and process-related parameters [55–58], i.e. one machine to another. However, training CNN-based models with DA on discarded examples from existing data through the use of intermediate tasks has been lacking in research.

2.3 Summary of the reviewed works and proposed approach

In the above review, the available data quantity as well as the need for labels constitutes major challenges for tool wear prediction. Significant manual feature engineering or automated post-processing (i.e. manual wear region segmentation in [27] and cascading image processing operations in [25]) is required to train models and predict the tool wear state and/or RUL. In addition, none of those approaches attempt to use MPPs, which are closely correlated to tool conditions, as inputs to predict the tool wear or RUL. Furthermore, the application of DTL for cutting tool image health and RUL prediction has not been extensively explored, primarily due to "closed-source" data availability [11].

A CNN model has demonstrated a superior performance on vision tasks. In our research, the capability of CNN is leveraged to be applied to support the RUL prediction based on VI images of cutting tools. Meanwhile, this research seeks to integrate transfer learning to further improve performance of the CNN model, and extend the CNN's capability with LSTM-based sequence modelling from heterogeneous data (i.e. image features and MPPs) for predicting the RUL of cutting tools accurately.

3 Methodology

This section presents the cutting tools' RUL prediction approach enabled by a hybrid CNN-LSTM model with a transfer learning mechanism. The dataset to support the developed approach is described, with the machining conditions and experimental setup. The approach comprises the following individual stages. Firstly, the CNN model is designed and pre-trained on a source domain dataset, and then fine-tuned for tool wear state prediction based on the transfer learning strategy for a target domain. Then, RUL prediction uses the extracted features by the CNN model and MPPs jointly feeding into a LSTM model. The flow of the approach is illustrated in Figure 1. The approach was implemented in the MATLAB R2020a with the Deep Learning Toolbox on a HPC cluster with NVIDIA Tesla K80 GPUs.

.

3.1 Dataset for methodology development

3.1.1 Data collection

The dataset to support the developed approach in this research comprises 25 machining experiments conducted using SP210-C4-16015 cutting tools for side milling of AISI 4340 alloy steel jaw-shaped workpieces [59]. The full specifications of the materials, equipment, and parameters are shown in Table 2. A Taguchi (5,3) factorial design was used to evaluate the effect of the MPPs on tool life, as presented in Table 3. For each factorial level, the material removal rate, *MRR*, was also calculated as $MRR = f_d a_e a_p$, where $a_p = 8$mm was the cutting width, kept constant throughout the experiments.

The image dataset contains 2884 1600×1200×3 images of cutting tools under different levels of magnification and from several different angles. The images were collected using a digital microscope via the VHX-5000 communication software. The images of the cutting tool were collected at specific intervals: 0m, 0.2m, 2m, 6m, 10m, 20m, 30m, and 40m, and then every 20m until tool failure (i.e. the flank wear width of the tool exceeding 0.4mm or the tool failed as described in Table 3). Several images were manually annotated to measure the maximum tool wear, and an example is shown in Figure 2. Some tools experienced different modes of failure (either prematurely or shortly after crossing the failure threshold), such as cutting edge tip and cylinder breakage. Therefore, the failure times are determined by interpolating (or extrapolating) the wear trend with a piecewise cubic hermite (PCHIP) interpolant. As a simplifying assumption, the failure mode is assumed to be flank wear for all of the tools within the experiments.

3.1.2 Data pre-processing for the CNN model

The image datasets used to pre-train a CNN model, such as ILSVRC 2012, comprise relatively small square images of size ranging from 220 to 300 pixels wide with the object in focus [47]. However, the input images are 1600×1200 pixels, which are considerably larger than the standard pre-trained CNN input size. In addition, to reduce the impact of negative space on the learned features, the images must be cropped and resized before they can be processed by the CNN model.

Pre-processing The images are first segmented using the fast marching method (FMM) segmentation algorithm [60], resulting in segmentation masks from which the region of interest (ROI) is selected, and a tight bounding box is drawn on the ROI mask. The bounding box determines the cropped window size for the resulting image; this ensures a consistently cropped image whose subsequent compression for input

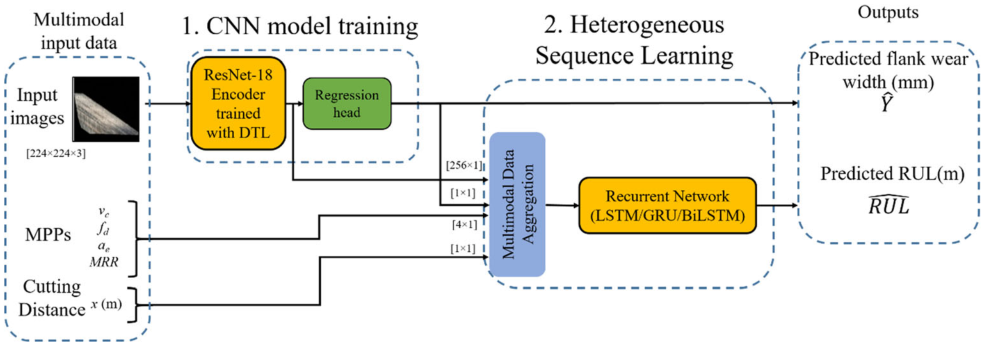


Fig. 1 The proposed tool wear and RUL prediction approach using a hybrid CNN-LSTM with transfer learning, showing the training sub-stages for CNN and recurrent network models

Table 2 Equipment, materials, specification and experimental conditions for machining experiments

Cutting tool model	**Machine tool**	**Dimensions (mm)**	**Tool material**
SP210-C4-16015	MAZAK VCN-430A-II	D16×36×100	AlCrSiN
Digital microscope	**Coolant**	**Tool holder model**	**Workpiece material**
KEYENCE VHX-5000	7% SIcut-Emu1020T	HSK-A 63	$Cr_{36}NiMo_4$
Workshop temperature (°C)	**Humidity (%)**	**Material hardness (HB)**	**Tool overhang (mm)**
27	50	250-300	40

into the CNN model will retain the defining textural characteristics of the worn tool edge. A sample of the image pre-processing operation is demonstrated in Figure 4 (bottom left).

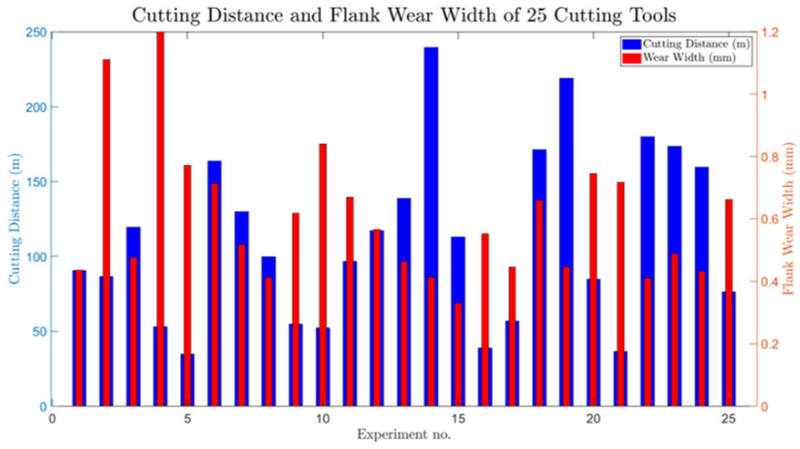
Augmentation The pre-processed images are further augmented in each training mini-batch, by implementing random affine transformations (i.e. scaling between 0.9 and 1.1 aspect ratio, translation within ±30 pixels, and rotation within ±30 degrees) to introduce further randomness within the resulting images. This prevents the CNN model remembering exact details of each example, therefore protecting against model overfitting. A sample augmented batch is shown in Figure 3 (bottom right).

3.2 Design of the CNN model

In this research, a CNN model based on ResNet-18 is trained for tool wear prediction, based on a previous experimental study to validate CNN performance on this predictive task [26]. To predict tool wear as the final output of the CNN model using the target

Table 3 Machining experiments with MPPs (cutting speed: v_c; feed rate: f_d; cutting depth: a_e) and failure conditions. Bold rows indicate test data sequences

Exp.	v_c (m/min)	f_d (mm/min)	a_e (mm)	T_{VB} (min)	L_{VB} (m)	Failure description
1	150.72	300	1.60	302.87	90.77	Cutting edge tip is broken: 0.436mm×0.287mm
2	150.72	870	2.00	99.51	86.49	Edge on Cylinder is broken: 1.110mm×1.2m
3	150.72	1115	2.67	107.25	119.49	Cutting edge tip is broken: 0.477mm×2.2mm
4	150.72	1430	3.30	37.13	53.01	Cutting edge tip is broken: 0.281mm×1.2mm
5	**150.72**	**2000**	**4.00**	**17.45**	**34.81**	**Edge on Cylinder is broken: 1.2mm×0.772mm**
6	184.38	300	2.00	545.63	163.60	Cutting edge tip is broken: 0.713mm×0.311mm
7	**184.38**	**870**	**2.67**	**149.20**	**129.71**	**Cutting edge tip is broken: 0.518mm×0.187mm**
8	184.38	1115	3.30	89.70	99.93	Flank wear: 0.412mm
9	184.38	1430	4.00	38.44	54.89	Edge on Cylinder is broken: 6.180mm×2.450mm
10	184.38	2000	1.60	26.20	52.32	Edge on Cylinder is broken: 0.84mm×1.2mm
11	200.96	300	2.67	322.85	96.77	Cutting edge tip is broken: 0.669mm×0.161mm
12	200.96	870	3.30	134.82	117.20	Flank wear is 0.566mm,
13	200.96	1115	4.00	124.46	138.69	Cylinder edge + tip broken: 0.463mm×0.435mm
14	**200.96**	**1430**	**1.60**	**167.57**	**239.53**	**Flank wear 0.413mm,Stop**
15	200.96	2000	2.00	56.58	113.06	Cutting edge tip is broken
16	218.54	300	3.30	129.68	38.82	Cutting edge tip is broken: 0.553mm×0.29mm
17	218.54	870	4.00	65.40	56.81	Cutting edge tip is broken: 0.446mm×0.2mm
18	**218.54**	**1115**	**1.60**	**153.72**	**171.31**	**Cutting edge tip is broken: 0.660mm×0.168mm**
19	218.54	1430	2.00	153.10	218.85	Cutting edge tip is broken: 0.446mm×0.6mm
20	218.54	2000	2.67	42.43	84.77	Cutting edge tip is broken: 0.745mm×0.3mm
21	251.2	300	4.00	121.94	36.49	Edge on Cylinder is broken: 1.3mm×3.8mm
22	251.2	870	1.60	206.95	179.96	Flank wear 0.411mm,Stop
23	251.2	1115	2.00	155.70	173.52	Cutting edge tip is broken: 0.489mm×0.160mm
24	251.2	1430	2.67	111.63	159.54	Cutting edge tip is broken: 0.433mm×0.12mm
25	**251.2**	**2000**	**3.30**	**38.27**	**76.46**	**Cutting edge tip is broken: 0.662mm×0.55mm**



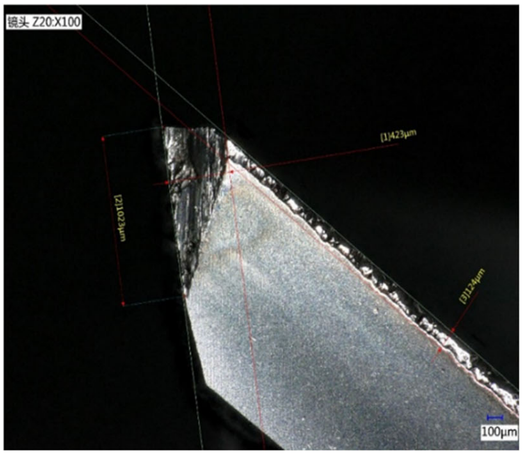


Fig. 2 left: the cutting distances and wear widths of tools #1-25; right: The image of the cutting tool #22 at cutting distance = 180 m, showing measurements of the tool failure region (cutting tool edge broken; size = 0.436×0.287mm)

domain dataset via a transfer learning strategy, two aspects of network adaptation must be implemented:

1. As the target task objective is changed from classification to regression, this entails replacing the CNN classification output layers with layers suitable for tool wear prediction. To that end, the output dense (i.e. fully connected) and softmax activation layers of the CNN model are replaced with appropriate dense and linear or sigmoidal activations, respectively [61]. A series of fully connected layers with a dropout ratio of 0.4 is used, as illustrated in Figure 4. Dropout is an effective regularisation technique that reduces the model's tendency to retain specialised neurons that remember the precise details of training examples (i.e. overfitting) [62]. This is achieved by setting a proportion (i.e. 40%) of the layer parameters to 0 at random.
2. Furthermore, the source and target datasets comprise starkly different data, whose feature distributions may not align well especially for the deepest feature layers, potentially causing negative transfer [48]. This motivates the application of an objective to reduce domain feature discrepancy.

The flows of pre-training and fine-tuning processes of the CNN model are shown in Figure 4. More details pertaining to

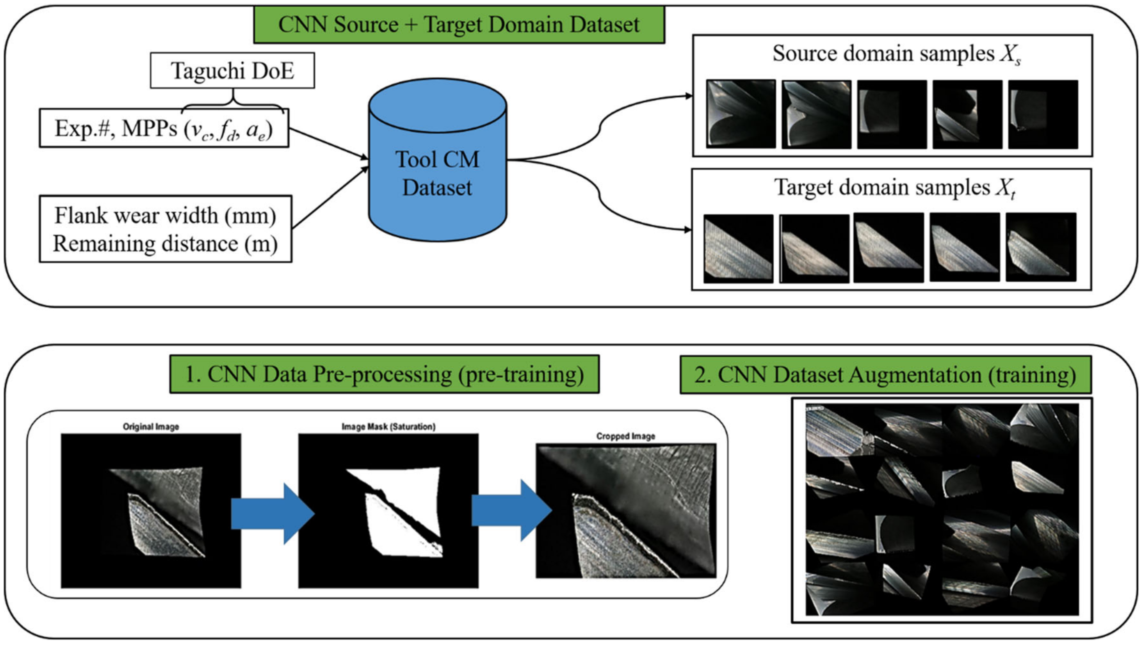

CNN Source + Target Domain Dataset
Taguchi DoE
Exp.#, MPPs (v_c, f_d, a_e)
Flank wear width (mm)
Remaining distance (m)
Tool CM Dataset
Source domain samples X_s
Target domain samples X_t
1. CNN Data Pre-processing (pre-training)
2. CNN Dataset Augmentation (training)
Original Image
Image Mask (Saturation)
Cropped Image

Fig. 3 Overview of the tool VI dataset (top) with source and target domain samples; and the image pre-processing pipeline (bottom) applied to source and target data during training, with pre-training image segmentation (bottom left) and mini-batch data augmentation (bottom right)

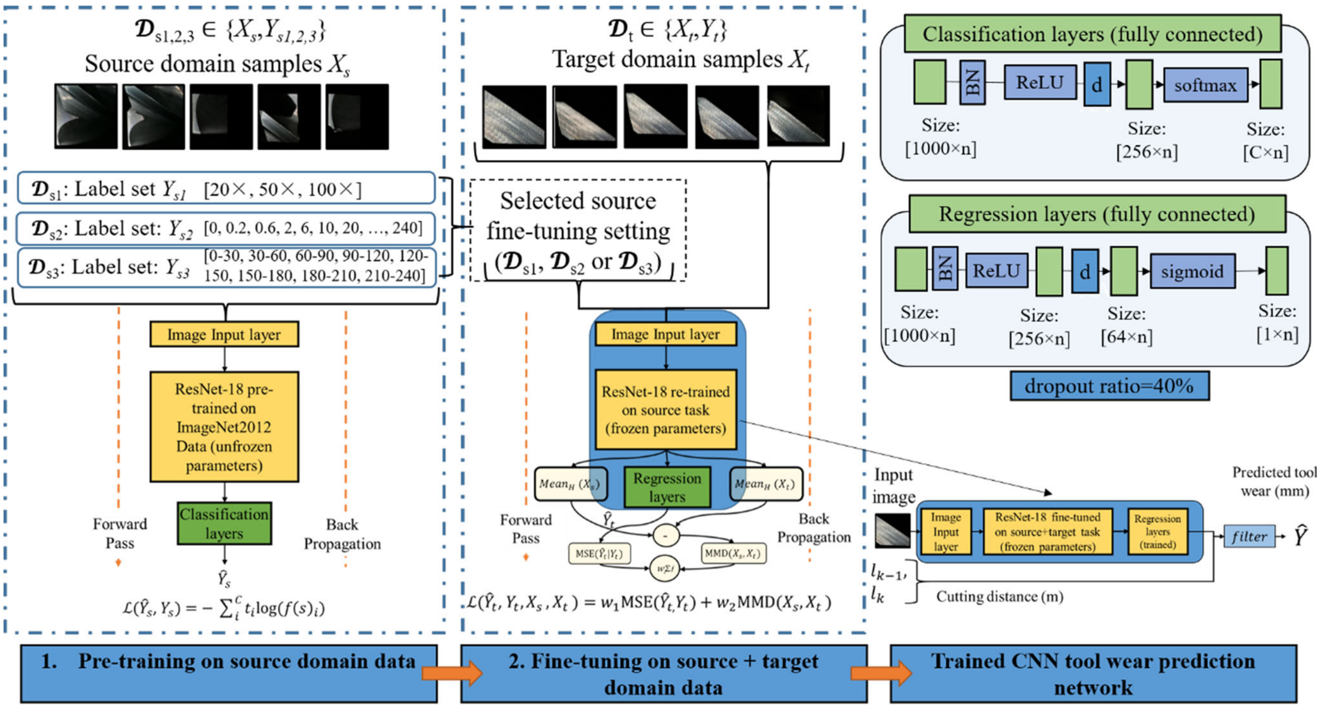

$\mathcal{D}_{s1,2,3} \in \{X_s, Y_{s1,2,3}\}$
Source domain samples X_s
$\mathcal{D}_{s1}$: Label set Y_{s1} [20×, 50×, 100×]
$\mathcal{D}_{s2}$: Label set: Y_{s2} [0, 0.2, 0.6, 2, 6, 10, 20, …, 240]
$\mathcal{D}_{s3}$: Label set: Y_{s3} [0-30, 30-60, 60-90, 90-120, 120-150, 150-180, 180-210, 210-240]
Image Input layer
ResNet-18 pre-trained on ImageNet2012 Data (unfrozen parameters)
Classification layers
Forward Pass
Back Propagation
$\hat{Y}_s$
$\mathcal{L}(\hat{Y}_s, Y_s) = -\sum_i^c t_i \log(f(s)_i)$
$\mathcal{D}_t \in \{X_t, Y_t\}$
Target domain samples X_t
Selected source fine-tuning setting ($\mathcal{D}_{s1}$, $\mathcal{D}_{s2}$ or $\mathcal{D}_{s3}$)
Image Input layer
ResNet-18 re-trained on source task (frozen parameters)
$Mean_H(X_s)$
Regression layers
$Mean_H(X_t)$
$\hat{Y}_t$
MSE($\hat{Y}_t|Y_t$)
MMD(X_s, X_t)
$w_i \Sigma$
$\mathcal{L}(\hat{Y}_t, Y_t, X_s, X_t) = w_1 \mathrm{MSE}(\hat{Y}_t, Y_t) + w_2 \mathrm{MMD}(X_s, X_t)$
Classification layers (fully connected)
BN
ReLU
d
softmax
Size: [1000×n]
Size: [256×n]
Size: [C×n]
Regression layers (fully connected)
BN
ReLU
d
sigmoid
Size: [1000×n]
Size: [256×n]
Size: [64×n]
Size: [1×n]
dropout ratio=40%
Input image
Image Input layer
ResNet-18 fine-tuned on source+target task (frozen parameters)
Regression layers (trained)
filter
$\hat{Y}$
Predicted tool wear (mm)
l_{k-1}, l_k
Cutting distance (m)
1. Pre-training on source domain data
2. Fine-tuning on source + target domain data
Trained CNN tool wear prediction network

Fig. 4 The CNN approach for tool wear prediction, demonstrating the network pre-training stage for $\mathcal{D}_s$ and subsequent fine-tuning on $\mathcal{D}_t$

the model design and training procedure are provided in the following sub-sections.

3.2.1 CNN transfer learning datasets

It is widely established that initialising the CNN weights from a suitable source pre-training setting is more effective than using random initialisation of CNN parameters. Hence, each of the source settings is independently evaluated as a pre-training task (i.e. the network learns one of the possible pre-training tasks, and the downstream task performance is evaluated after fine-tuning from each pre-trained CNN).

Source domain dataset The source domain dataset, $\mathcal{D}_s = \{X_s, Y_s\}$, uses three different categorical label spaces pertaining to the tool VI dataset, which contains all 2,884 image samples obtained in the machining experiments conducted in this research, excluding the subset used as target domain data (327 images). The dataset is used to pre-train the CNN model on one of three separate tasks: classification of tool magnification variant (20×, 50×, 100×), tool cutting distance (0m–240m), and a discrete version of cutting distance (in intervals of 30m) (Table 4).

Target domain dataset The target dataset $\mathcal{D}_t$ comprises 327 distinct samples from the source domain dataset for which associated measurements of the cutting tool flank wear width (in mm) were recorded, showing the flank of the tool at 100x magnification. The tool flank wear width is used as the target output for training the CNN model.

3.2.2 CNN model pre-training and fine-tuning

CNN pre-training stage The CNN model attempts to learn a function $f(x)$ to map the input image, X, to ground truth data (i.e. labels), Y, with the following mapping:

$$Y = f(X) = \theta_{CNN}(X) \tag{1}$$

where $\theta_{CNN}(X)$ is the CNN model function based on CNN parameters applied to image X.

The key strength of using the CNN model to process images lies in the convolutional layers, which learn feature maps across various image spatial and depth levels [63, 64]. In this research, ResNet-18 is used as the base CNN architecture, due to its cheaper computational overhead and lighter weight (i.e. parameter count). Notably, residual connections in the ResNet-18 architecture allow additive merging between more generic and more specialised feature maps [65]. The forward branch of a residual "stack" undergoes a series of convolutional, batch normalisation, and nonlinear activation operations, and the skip (or residual) branch outputs are added directly to the outputs of the forward branch path. Additionally, max pooling operations refine the convolutional features learnt throughout the network by encoding the image regions with the corresponding max feature value over the pooled region. Network layer parameters are learnt via back-

Table 4 Source task description for classification tasks $\mathcal{D}_{s1}$, $\mathcal{D}_{s2}$, and $\mathcal{D}_{s3}$

Source domain	Pseudo-label source	#Classes	Unique class labels
$\mathcal{D}_{s1}$	Image magnification variant	3	(20×, 50×, 100×)
$\mathcal{D}_{s2}$	Image cutting length	20	[0, 0.2, 0.6, 2, 6, 10, 20, …, 240]
$\mathcal{D}_{s3}$	Image cutting length (discrete)	8	[0–30, 30–60, 60–90, 90–120, 120–150, 150–180, 180–210, 210–240]

propagation gradient descent algorithms [66]. In the pre-training stage, the CNN network is trained using categorical cross-entropy loss, defined for a mini-batch of N training examples with class labels $T_{i,j}$ and logit vectors $X_{i,j}$ as:

$$CE = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C}\left(T_{i,j}\log\left(X_{i,j}\right) + \left(1-T_{i,j}\right)\log\left(1-X_{i,j}\right)\right) \quad (2)$$

CNN fine-tuning stage For tool wear prediction, the empirical regression loss metric incorporates a measure of prediction ($\widehat{Y}_i$) deviation from target outputs (Y_i), i.e., the mean square error (MSE) loss function, computed on a mini-batch with N examples as:

$$MSE\left(\widehat{Y}_i, Y_i\right) = \frac{1}{N}\sum_{i=1}^{N}\left(\widehat{Y}_i - Y_i\right)^2 \quad (3)$$

Given a source domain $\mathcal{D}_s = \{X_s, Y_s\}$ where X_s are examples in the source domain and Y_s are source labels to be predicted, the source task can be defined as:

$$T_{s:} : Y_s = \mathbf{W_s} \cdot X_s + \mathbf{b_s} \quad (4)$$

where $\mathbf{W_s}$ and $\mathbf{b_s}$ denote the CNN model's parameters (i.e. weights and biases) to be learned on the task.

The target domain is defined by $\mathcal{D}_t = \{X_t, Y_t\}$, with its target task as:

$$T_{t:} : Y_t = \mathbf{W_t} \cdot X_t + \mathbf{b_t} \quad (5)$$

Each domain (source and target domain) is characterised by its respective feature distribution. Under the assumption that the expectation of these distributions, i.e. $E(Y_s | X_s) = E(Y_t | X_t)$ are equal, the source and target domains are considered sufficiently similar for transfer learning. Therefore, the MMD metric is used to regularise the training loss, using the CNN's final feature-dense layer (i.e. fc_{1000}). The activations extracted from this layer are originally used to classify the input image into one of 1000 different categories from the ImageNet 2012 dataset.

The empirical MMD between source and target domains takes the following formula:

$$MMD^2(X_s, X_t) = \left\| \frac{1}{M}\sum_{i=1}^{M}\varphi\left(X_i^s\right) - \frac{1}{M}\sum_{i=1}^{M}\varphi\left(X_i^t\right) \right\|_{\mathcal{H}}^2 \quad (6)$$

where $\varphi\left(X_i^s\right)$ and $\varphi\left(X_i^t\right)$ denote the source and target feature embeddings, respectively.

The feature embeddings are projected into a Reproducing kernel Hilbert space (RKHS) $\mathcal{H}$ with a Gaussian kernel, according to [67]. An optimal kernel is thus selected from possible candidate kernels $\sigma \in \boldsymbol{\sigma}$ maximising the distribution difference. Notably, the kernel should be sufficiently rich and, simultaneously, restrictive for the MMD to be sufficiently discriminative between source and target domain features. Due to the high cost of MMD computation, a linear-time approximation of MMD [68] is used in this research:

$$MMD_l^2(s,t) = \frac{2}{M}\sum_{i=1}^{\frac{M}{2}} h_l(\mathbf{z}_i) \quad (7)$$

where $\mathbf{z}_i = \left(\mathbf{x}_{2i-1}^s, \mathbf{x}_{2i}^s, \mathbf{x}_{2j-1}^t, \mathbf{x}_{2j}^t\right)$, and $h_l(\mathbf{z}_i)$ is a kernel operator defined on the quad-tuple as follows:

$$h_l(z_i) = k\left(\boldsymbol{x}_{2i-1}^s, \boldsymbol{x}_{2i}^s\right) + k\left(\boldsymbol{x}_{2j-1}^t, \boldsymbol{x}_{2j}^t\right) - k\left(\boldsymbol{x}_{2i-1}^s, \boldsymbol{x}_{2j}^t\right) - k\left(\boldsymbol{x}_{2i}^s, \boldsymbol{x}_{2j-1}^t\right) \quad (8)$$

The MMD statistic is then computed on these feature embeddings. The overall CNN training loss can therefore be described in terms of the empirical regression loss as well as the MMD regularisation:

$$\mathcal{L}\left(X_s, X_t, \widehat{Y}, Y\right) = w_1 MSE\left(Y, \widehat{Y}\right) + \frac{w_2}{R}\sum_{r=1}^{R} MMD_l^2(s,t)_r \quad (9)$$

where the MMD metric is computed for the designated fully connected layer, and r denotes the layer indexes of layers with respect to which the MMD is computed. The weights w_1 and w_2 are chosen as [0.9, 0.1], such that $w_1 > w_2$ and $w_1 + w_2 = 1$. Algorithm 1 demonstrates the overall loss computation.

To train the network via back-propagation gradient descent, the gradient of the in the step 7 is calculated using automatic differentiation, and used to update the CNN model's parameters using the Adam optimiser. At iteration l, the Adam gradient update uses the following routine to calculate the rate of variation of the model loss $E(\theta_l)$:

$$m_l = \beta_1 m_{l-1} + (1-\beta_1)\nabla E(\theta_l) \quad (10)$$

$$v_l = \beta_2 v_{l-1} + (1-\beta_2)[\nabla E(\theta_l)]^2 \tag{11}$$

where m_l and v_l are the momentum and velocity terms for the variation in model gradient, and β_1 and β_2 are decay factors controlling gradient and squared gradient (set to 0.9 and 0.99, respectively). The calculated model parameters are then updated using the model update step:

$$\theta_{l+1} = \theta_l - \frac{\alpha m_l}{\sqrt{v_l} + \epsilon} \tag{12}$$

where ϵ is a loss conditioning hyperparameter, set as a very small value (in this case, 1×10^{-6}).

Algorithm 1: MMD-MSE Computation for CNN transfer learning

```
Initialise $X_i^s, X_i^t, Y_i^s \leftarrow 0$
Compute initial kernel parameter list $\boldsymbol{\sigma} \sim [2^n]$, $-2 \leq n \leq 12$
iteration = 0
set training to true;
while training do
    iteration = iteration + 1
    Compute $i$ forward mini-batch predictions using CNN layers on target data
        $\hat{Y}_i^t = \mathbf{W}_{CNN}(X_i^t) + \mathbf{B}_{CNN}$
    Compute $i$ forward feature embeddings for source and target domain batch w.r.t. layers $l$:
        $\varphi_{s,l}(X_i^s) \leftarrow f(X_i^s, l); \varphi_{t,l}(X_i^t) \leftarrow f(X_i^t, l)$
    Project $\varphi(X_s)$ and $\varphi(X_t)$ into RKHS with chosen Gaussian kernels $\boldsymbol{\mathcal{N}} \sim (0, \boldsymbol{\sigma})$:
        $h_l(\mathbf{z}_i) = k(\mathbf{x}_{2i-1}^s, \mathbf{x}_{2i}^s) + k(\mathbf{x}_{2j-1}^t, \mathbf{x}_{2j}^t) - k(\mathbf{x}_{2i-1}^s, \mathbf{x}_{2j}^t) - k(\mathbf{x}_{2i}^s, \mathbf{x}_{2j-1}^t)$
    Choose optimal kernel parameter $\sigma \in \boldsymbol{\sigma}$ to maximise distribution difference between embeddings
    Compute layer-wise MMD as
        $\mathrm{MMD}_l^2(s,t) = \frac{2}{M}\sum_{i=1}^{\frac{M}{2}} h_l(\mathbf{z}_i)$
    Compute mini-batch loss on $i$ examples:
        $\mathcal{L}(\mathrm{X}_s, \mathrm{X}_t, \hat{Y}, Y) = w_1 MSE(Y, \hat{Y}) + \frac{w_2}{R}\sum_{r=1}^{R} \mathrm{MMD}_l^2(\mathrm{X}_s, \mathrm{X}_t)_r$
end
```

Prediction filter As the wear process is monotonic, i.e. wear can only increase as the cutting distance (l_k) increases (starting from 0), filtering is implemented to ensure the predictions adhere to this degradation pattern. Therefore, given a current sample x_k with its corresponding cutting distance l_k in m, the CNN model output $f(x)$ is filtered according to the following filter function $g(x)$:

$$g(x) = \begin{cases} f(x_k), & k = 1 \\ \max(d_k f(x_{k-1}), f(x_k)), & k > 1 \end{cases} \tag{13}$$

where $d_k = (1 + (2.5\alpha) \cdot \exp(l_{k-k-1}/l_k))$. The value of α is set to 0.03 in this research.

3.3 LSTM model for cutting tool RUL prediction

An LSTM model is designed to learn the temporal dependencies between the sequential data observations, $x_{t,t+1,t+2,t\ldots}$ to predict the current output y_t. The LSTM model contains four distinct *gates*, governing different aspects of the model prediction and state update process. The four gate functions are the input gate (i), the forget gate (f), the cell candidate gate (g), and the output gate (o). The LSTM parameter matrices W, R, and b denote the model input weights (W_i, W_f, W_g, and W_o), recurrent weights (R_i, R_f, R_g, and R_o) and biases (b_i, b_f, b_g, and b_o), respectively [46]. Meanwhile, the GRU unit comprises three internal mechanisms, the reset (r_t), update (z_t), and

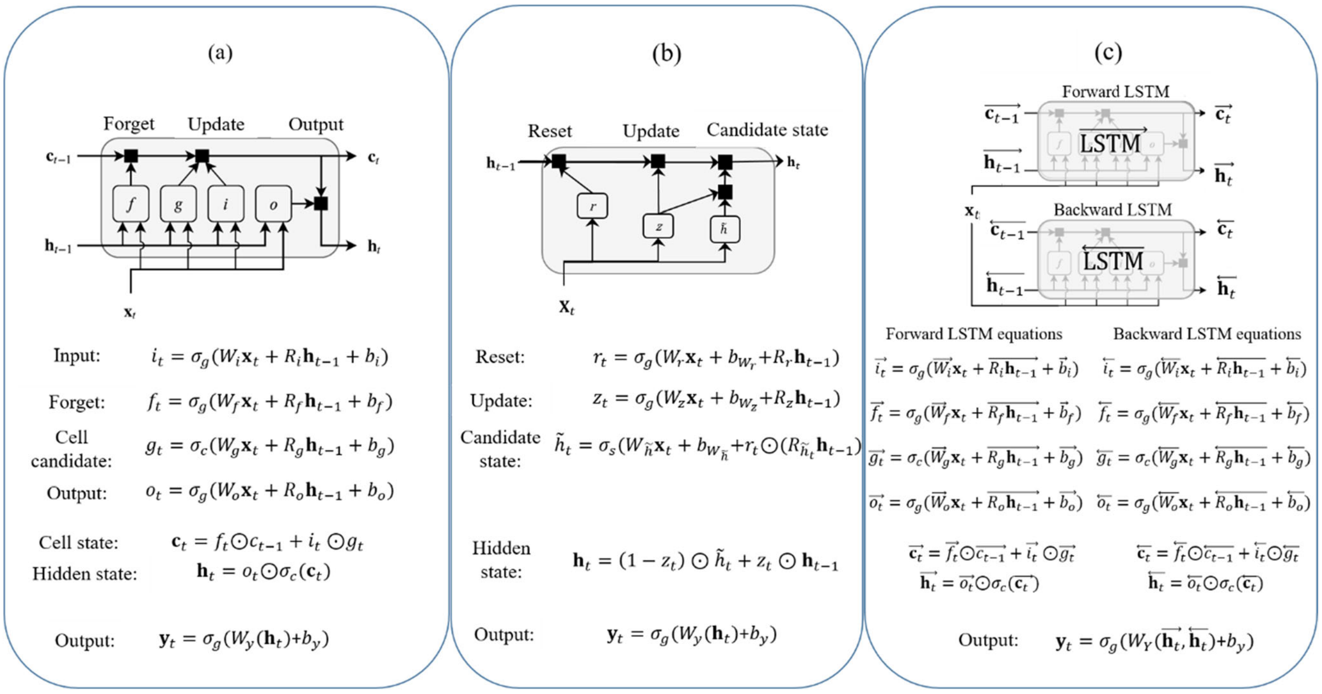

(a)
Forget Update Output
c_{t-1} c_t h_{t-1} h_t x_t
Input: $i_t = \sigma_g(W_i x_t + R_i h_{t-1} + b_i)$
Forget: $f_t = \sigma_g(W_f x_t + R_f h_{t-1} + b_f)$
Cell candidate: $g_t = \sigma_c(W_g x_t + R_g h_{t-1} + b_g)$
Output: $o_t = \sigma_g(W_o x_t + R_o h_{t-1} + b_o)$
Cell state: $c_t = f_t \odot c_{t-1} + i_t \odot g_t$
Hidden state: $h_t = o_t \odot \sigma_c(c_t)$
Output: $y_t = \sigma_g(W_y(h_t) + b_y)$
(b)
Reset Update Candidate state
Reset: $r_t = \sigma_g(W_r x_t + b_{w_r} + R_r h_{t-1})$
Update: $z_t = \sigma_g(W_z x_t + b_{w_z} + R_z h_{t-1})$
Candidate state: $\tilde{h}_t = \sigma_s(W_{\tilde{h}} x_t + b_{w_{\tilde{h}}} + r_t \odot (R_{\tilde{h}_t} h_{t-1})$
Hidden state: $h_t = (1 - z_t) \odot \tilde{h}_t + z_t \odot h_{t-1}$
Output: $y_t = \sigma_g(W_y(h_t) + b_y)$
(c)
Forward LSTM
Backward LSTM
LSTM
Forward LSTM equations
Backward LSTM equations
$\overrightarrow{i_t} = \sigma_g(\overrightarrow{W_i} x_t + \overrightarrow{R_i h_{t-1}} + \overrightarrow{b_i})$ $\overleftarrow{i_t} = \sigma_g(\overleftarrow{W_i} x_t + \overleftarrow{R_i h_{t-1}} + \overleftarrow{b_i})$
$\overrightarrow{f_t} = \sigma_g(\overrightarrow{W_f} x_t + \overrightarrow{R_f h_{t-1}} + \overrightarrow{b_f})$ $\overleftarrow{f_t} = \sigma_g(\overleftarrow{W_f} x_t + \overleftarrow{R_f h_{t-1}} + \overleftarrow{b_f})$
$\overrightarrow{g_t} = \sigma_c(\overrightarrow{W_g} x_t + \overrightarrow{R_g h_{t-1}} + \overrightarrow{b_g})$ $\overleftarrow{g_t} = \sigma_c(\overleftarrow{W_g} x_t + \overleftarrow{R_g h_{t-1}} + \overleftarrow{b_g})$
$\overrightarrow{o_t} = \sigma_g(\overrightarrow{W_o} x_t + \overrightarrow{R_o h_{t-1}} + \overrightarrow{b_o})$ $\overleftarrow{o_t} = \sigma_g(\overleftarrow{W_o} x_t + \overleftarrow{R_o h_{t-1}} + \overleftarrow{b_o})$
$\overrightarrow{c_t} = \overrightarrow{f_t} \odot \overrightarrow{c_{t-1}} + \overrightarrow{i_t} \odot \overrightarrow{g_t}$ $\overleftarrow{c_t} = \overleftarrow{f_t} \odot \overleftarrow{c_{t-1}} + \overleftarrow{i_t} \odot \overleftarrow{g_t}$
$\overrightarrow{h_t} = \overrightarrow{o_t} \odot \sigma_c(\overrightarrow{c_t})$ $\overleftarrow{h_t} = \overleftarrow{o_t} \odot \sigma_c(\overleftarrow{c_t})$
Output: $y_t = \sigma_g(W_Y(\overrightarrow{h_t}, \overleftarrow{h_t}) + b_y)$

Fig. 5 Comparison between **a** LSTM layer; **b** GRU layer; **c** BiLSTM layer

candidate state ($\tilde{h}_t$), which are used to update the hidden state $\mathbf{h}_t$, used to compute the output y_t. Figure 5 compares an LSTM to a GRU in terms of structure and operation. In addition to the unidirectional LSTM model structure, a bidirectional LSTM (i.e. BiLSTM) model comprises forward and backward LSTM units, where the sequence dependencies are learnt along both directions. For each layer type (LSTM, BiLSTM, or GRU), three series model variants and their residual counterparts are explored, as illustrated in Figure 6. Each recurrent series network comprises three successive recurrent and dense layer stacks, with 4d hidden units in the first recurrent layer and 2d fully connected outputs. The final fully connected layer is connected to a "clipped" ReLU activation layer, and the overall model loss function is computed using the MSE loss.

Heterogeneous feature aggregation Given a d×1 aggregated input vector (i.e. CNN lagged outputs, CNN activations, and predictors) for each image sample, the CNN models output a vector of the activations (or features) based on the designated layer. The feature vector is output with respect to the penultimate fully connected layer, i.e. fc_{256}. This yields a feature vector of size 256×1 for each observation, which is subsequently concatenated with the predictor vector for each observation. The resulting output is a vector of size 256+v, where v denotes the length of the predictor vector. The predictors chosen for each observation are the MPPs, (v_c, f_d, a_e, and MRR) and the current cutting distance in m (i.e. the distance at which the tool wear measurement has been taken). The predictors provided to the recurrent model are pre-processed before being input to the network, where each predictor, x_i, is standardised by subtracting its mean and dividing by its standard deviation. In other words,

$$\overline{x} = \frac{x - \mu_x}{\sigma_x} \tag{14}$$

RUL scaling and filtering The RUL targets are scaled between 0 and 1 prior to training the model. At the RUL output, the outputs are reverse-scaled between 0 and the maximum

Fig. 6 left: series stack configurations with (**a**) LSTM layer; (**b**) GRU layer; (**c**) Bi-LSTM layer; right: residual stack configurations with (**d**) LSTM layer; (**e**) GRU layer; (**f**) Bi-LSTM layer

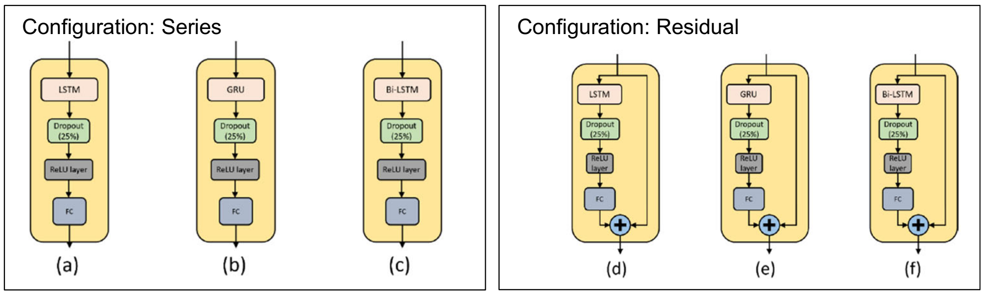


Table 5 Hyper-parameters for source pre-training and target task fine-tuning of the modified ResNet-18 CNN

Hyperparameter /option	Value	Hyperparameter/ option	Value
Learning rate	$8{\times}10^{-6}$	Optimiser	Adam
Mini-batch size	32	Gradient decay factor	0.9
Max # of epochs	100	Squared gradient decay factor	0.995
L_2 regularisation	$2{\times}10^{-3}$	Epsilon	$1{\times}10^{-8}$

predicted cutting distance at $k = 1$, i.e. 240 m. This scaling is crucial to ensure stable training of the LSTM model, as large target values often lead to large error values, which impact the gradient update procedure. To improve the accuracy of RUL prediction at the first-time step, the initial value of the RUL prediction, i.e. at $k = 1$, is obtained by averaging the first m prediction outputs. The RUL prediction is monotone decreasing—its value decreases with cutting distance (until it reaches zero—i.e. tool end-of-life). Therefore, RUL predictions are smoothed to ensure that the current prediction y_k does not exceed $y_{k\text{-}1}$. This is achieved by thresholding the current output using the following filter:

$$y_k : f(x_k, y_{t-1}) = \begin{cases} \frac{1}{m}\sum_{k=1}^{m} y_k, & k = 1 \\ \min(\alpha y_{k-1}, y_k), & k > 1 \end{cases} \quad (15)$$

4 Experiments and results

Each of the three steps of this approach was verified by assessing the predictive capability of each component independently. The first set of experiments validated the use of the CNN model to classify the source domain datasets into corresponding categories. As the CNN model was initially trained on the ILSVRC2012 for general-purpose image classification, this step was performed to adapt the CNN model's parameters in the deeper CNN layers, which are most prone to negative transfer due to fragile co-adaptation [48, 49]. The subsequent stage further adapted the CNN model's parameters learned in the source domain dataset by explicitly aligning the source and target domain feature distributions via MMD-MSE. The aggregated CNN model's features and vector inputs capture the tool degradation in successive image samples and encode additional condition information related to tool wear progression.

4.1 CNN pre-training: source task selection

The following experiments were carried out to test the effectiveness of the CNN model for feature extraction as an independent step. Firstly, the classification performance of the CNN model on each source domain task (i.e. $\mathcal{D}_{s1}$, $\mathcal{D}_{s2}$, and $\mathcal{D}_{s3}$) was tested to ensure its ability to classify the images into the corresponding class, determined by the source task labels. The CNN model was trained using the hyper-parameters recorded in Table 5. Furthermore, Table 6 summarises the results of the test performance of the source domain dataset across the three source task settings proposed. Based on the validation and test results obtained for the evaluated source tasks, the source task $\mathcal{D}_{s1}$ was selected as the pre-training stage for the CNN model, which was subsequently fine-tuned for regression prediction using the MMD-MSE loss. In this stage, an early stopping condition was used to quit training if the validation accuracy stopped improving after six validation iterations. As shown in Table 6, the validation and test accuracy on source domain task setting $\mathcal{D}_{s1}$ are the highest amongst the tested settings. This is likely due to the fact that samples under different magnification settings are easier to distinguish amongst each other (Fig. 7).

4.2 CNN fine-tuning from the source task to the target task

The CNN model's fine-tuning to align the distributions between source and target tasks was performed, with 160 target domain images used to re-train the CNN model for a short duration, and 67 images were used for validation (shown in Figure 8). Before this step, the model output configuration (i.e. classification layers) is replaced with regression layers as per Figure 5. The CNN model is then re-trained with the same hyper-parameters used for source task pre-training (Table 5)(Figure 9). Five sequences were used to validate

Table 6 Source task pre-training results on different source domain tasks

Source domain task	Validation accuracy	Test accuracy	Training time (s)
$\mathcal{D}_{s1}$	100%	94.6%	2220
$\mathcal{D}_{s2}$	68.8%	57.4%	2378
$\mathcal{D}_{s3}$	72.7%	63.9%	2114

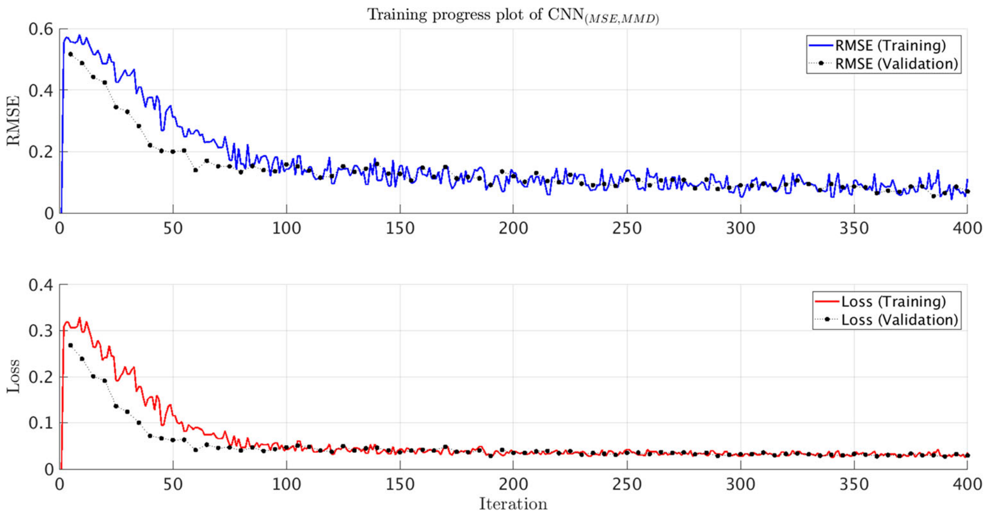


Fig. 7 CNN fine-tuning loss progress, top: RMSE (training and validation); bottom: loss (training and validation)

the CNN model's predictions (#5, #7, #14, #18, and #25). The test outputs for tool wear prediction and the tool wear threshold are shown in Figure 10, and the test performance of the source-target settings for CNN fine-tuning on target dataset is given in Table 7. The results indicate that using $\mathcal{D}_{s1}$ as the pre-training setting with the MMD fine-tuning procedure boosts network performance on the test dataset for tool wear prediction. Also, visualising the feature distributions under t-SNE before and after applying the MMD constraint (Figure 9) illustrates the clustering of feature embeddings closer to (0, 0), showing the domain discrepancy metric has boosted the test performance of the CNN model.

4.3 Cutting tool RUL prediction from aggregated inputs and MPP ablation

The purpose of PHM is the prediction of RUL. It demonstrated that the CNN model with MSE-MMD can predict tool wear

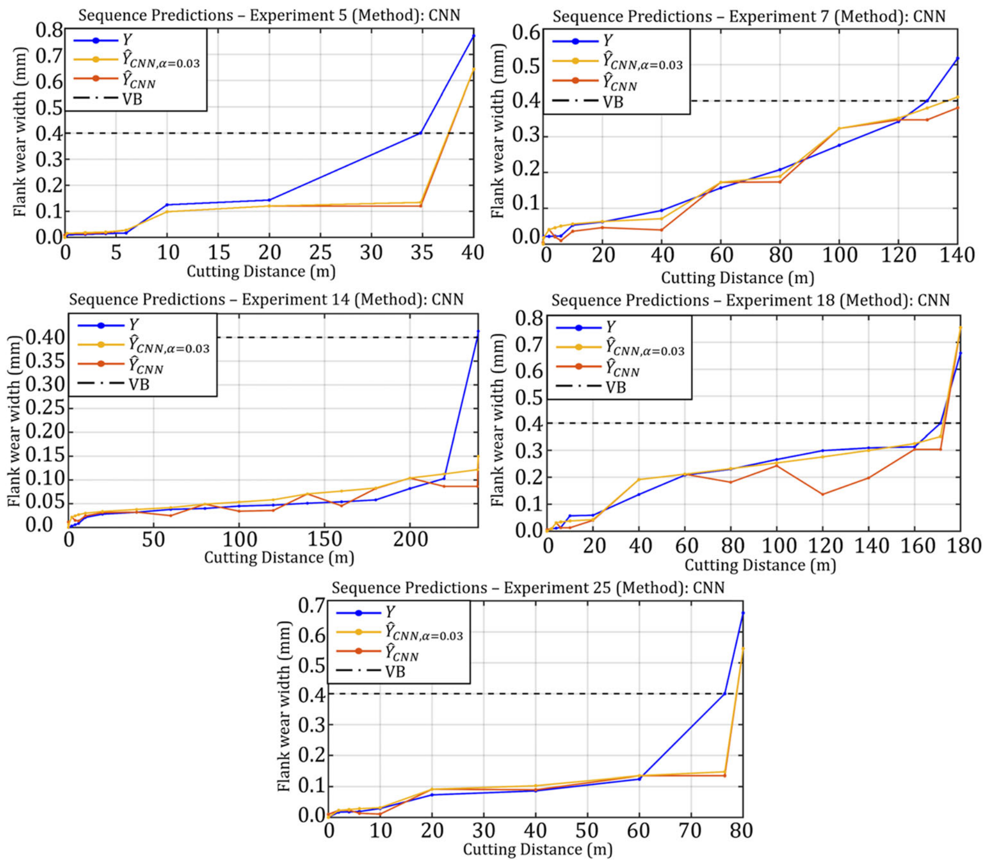


Fig. 8 Fine-tuned CNN results for tool wear prediction, with the CNN predictions after implementing monotonic filtering in yellow (without the filter in red). The predictions from the CNN follow the targets closely across the experiments in the early wear stages but some precision is lost towards the end of the tool life (specifically for experiment #5, 14, and 25)

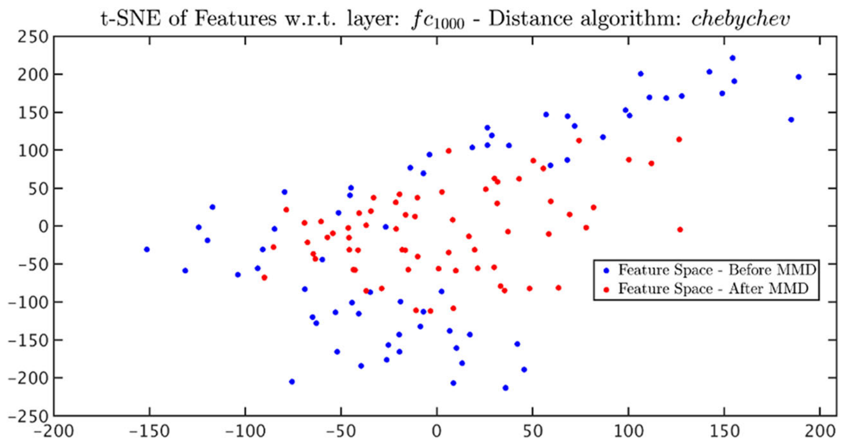


Fig. 9 t-SNE of the trained CNN features on test dataset with source task $\mathcal{D}_{s1}$, with the "Chebychev" distance algorithm and a perplexity value of 30. Applying the MMD constraint to the features at this layer has a clustering effect on the features for the observations, as demonstrated by the convergence of the red points closer to (0, 0) after applying the MMD constraint

with low MAE, RMSE and high R^2 values indicating strong correlation, the RUL prediction models are trained with heterogeneous data using the hyper-parameters in Based on this procedure, the input MPPs were masked off from the training and testing data, and the model trained on the corresponding datasets. The evaluation results in terms of performance metrics (MAE, RMSE and R^2) as well as prognostics metrics are given in Table 10. Predictions with prognostics margins are illustrated in Figure 11. Figure 12 shows the evaluation of the models tested on the MPPs ablation data.

Furthermore, MPPs feature ablation was performed where each model was performed to evaluate the impact of MPPs on the predictive model (Table 8). Given a subset of the MPPs, i.e. v_c, f_d, a_e, and MRR and the v_c is to be masked, a corresponding masking vector is created, with 0 in place of v_c and 1 elsewhere. The masking vector is used to compute the masked input data for the model, i.e.:

$$x_{masked} = x \circ v_{mask} \tag{16}$$

where $\circ$ is the Hadamard product between the vectors. This vector is concatenated with the CNN output feature vector, yielding a vector of size [256+v]. Based on this procedure, the input MPPs were masked off from the training and testing data, and the model trained on the corresponding datasets (Table 9). The evaluation results in terms of performance metrics (MAE, RMSE, and R^2) as well as prognostics metrics are given in Table 10. Predictions with prognostics margins are illustrated in Figure 11. Figure 12 shows the evaluation of the models tested on the MPPs ablation data.

In addition, a k-fold cross-validation scheme was implemented, where the training sequences were split into 4 folds, with 80% of the sequences used for training and the remaining 20% for validation, for a total of 260 training samples. The 5 testing sequences comprise 20% of the dataset (i.e. 67 testing samples), across a variety of MPP conditions. After the model was trained and validated on each fold, the learnt model parameters were used to initialise the model for the subsequent fold; thus, the model layers' learning rates are halved from their initial value (which is set to 1, i.e. the network's base initial learning rate). Three recurrent layer architectures were compared: LSTM, BiLSTM, and GRU. Residual connections were also explored in this work, where a residual path was implemented between each recurrent fully connected network layer pair. To that end, each layer variant was tested with and without a residual connection implementation. The test results for the sequences are summarised in Table 10 predictions with prognostics margins which are illustrated in Figure 10, and k-fold cross-validation scheme implemented on available sequences is shown in Figure 12.

5 Analysis and discussions

5.1 Tool wear prediction and RUL prediction performance analysis

As noted from the procedure of tool wear prediction using the transfer learning enabled CNN, the model gained a considerable performance increase in validation and testing criteria (i.e. MAE and RMSE) due to the incorporation of MMD into the objective of the CNN model (refer to Table 7). Furthermore, a notable gain was acquired from fine-tuning the ResNet-18 model on the source domain dataset, particularly with the $\mathcal{D}_{s1}$ setting.

Both series and residual model variants (i.e. LSTM/BiLSTM and Res-LSTM/BiLSTM) demonstrated a capability to learn RUL prediction from multimodal data (indicated by qualitative fitting performance) and generally strong correlation between predicted and true RUL (indicated by high R^2 values). However, the initialisation performance of the RUL predictions could be substantially improved, as models tend to predict a value close to the mean of the target RUL at time $t = 0$, i.e., the target initial RUL. Methods to exploit the underlying prior distribution of the target RUL could be explored for the initialisation

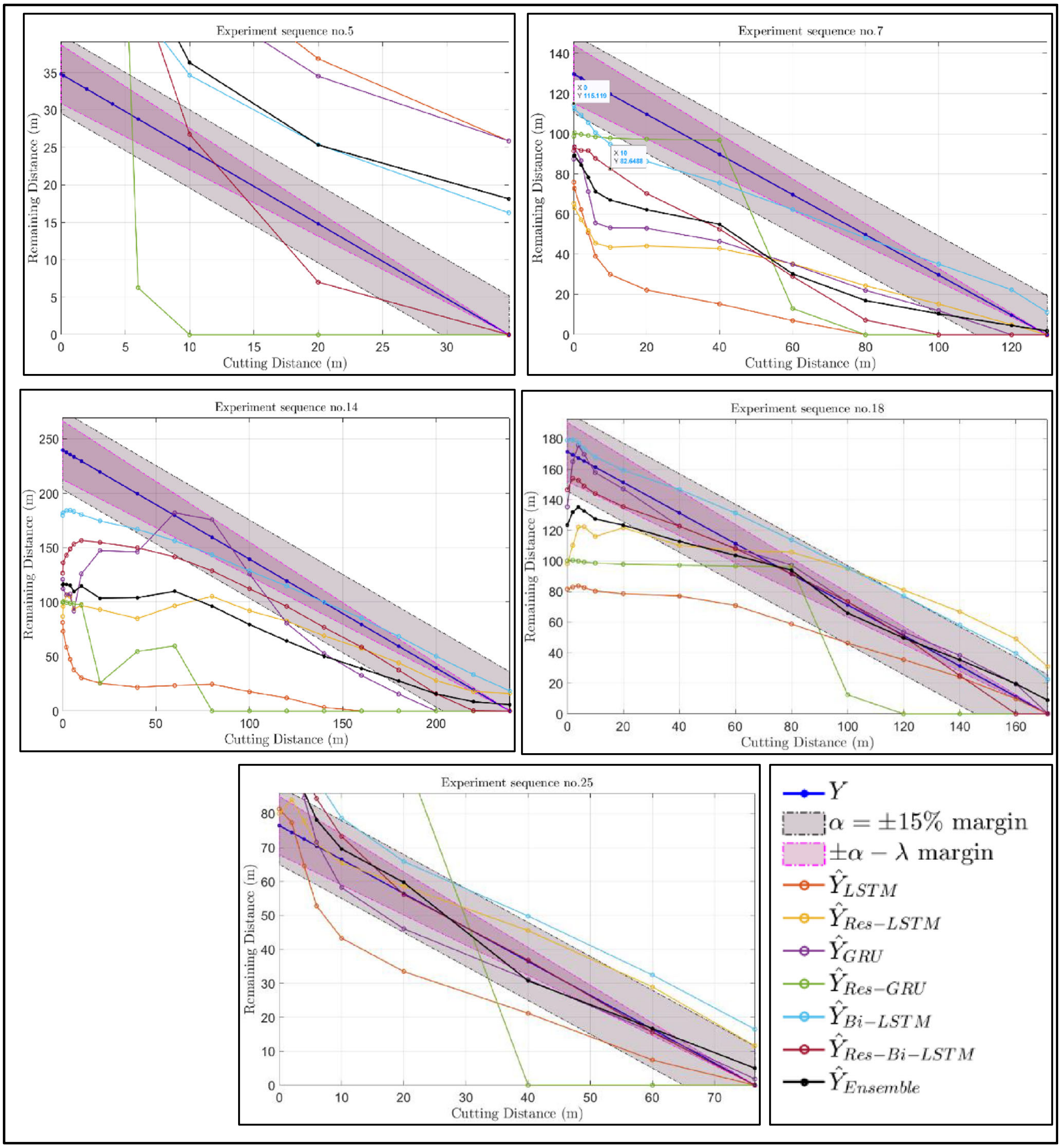


Fig. 10 RUL test sequence prediction outputs using different recurrent model configurations (including series and residual variants), showing α-margin and α-λ accuracy margin. The selection criterion for α-λ accuracy should be used to evaluate models which were capable of progressively more accurate predictions, staying within the margin following t_λ

phase. However, given that independence assumptions of the RUL at a current sample are dependent on the RUL at the previous sample, estimation frameworks are not necessarily applicable to this predictive problem. It is noted that generally, under-estimates of the RUL are preferred to over-estimates. Based on this consideration, the models generally tend to under-estimate RUL for the test sequences (with the exception of the series Bi-LSTM, which over-estimates RUL in 4 out of 5 experiments (sequences #7, #14, #18, and #25)). However, the Res-Bi-LSTM offers markedly better performance in terms of convergence to true RUL particularly in the late stages of prediction.

Table 7 Performance of several source-target settings for CNN fine-tuning on target dataset

CNN model (Loss)	Source-target pre-training task	Validation		Testing		R^2 (testing)
		MAE	RMSE	MAE	RMSE	
ResNet-18 (MMD-MSE)	$\mathcal{D}_{s1}$	**0.0212**	**0.0382**	**0.0520**	**0.0074**	**0.848**
	$\mathcal{D}_{s2}$	0.0221	0.0396	0.0616	0.0102	0.797
	$\mathcal{D}_{s3}$	0.0242	0.0424	0.0634	0.0114	0.722
ResNet-18 (MSE)	$\mathcal{D}_{s1}$	0.0802	0.103	0.167	0.177	0.788
	$\mathcal{D}_{s2}$	0.0824	0.112	0.172	0.182	0.744
	$\mathcal{D}_{s3}$	0.0837	0.124	0.178	0.186	0.692

5.2 Tool RUL PHM analytics and evaluation

Table 11 summarises the PHM analytics results for the test sequence experiments, in terms of the PHM metrics described in [69]. Numerous considerations for the given RUL prediction problem can be used to determine the bounds on metrics such as the Prognostic Horizon (PH) and α-λ accuracy, namely for the specification of the error margin. The α-cycle shows the interval at which the predictions have entered the α-margin of error (i.e. ±15% of true RUL), whereas the

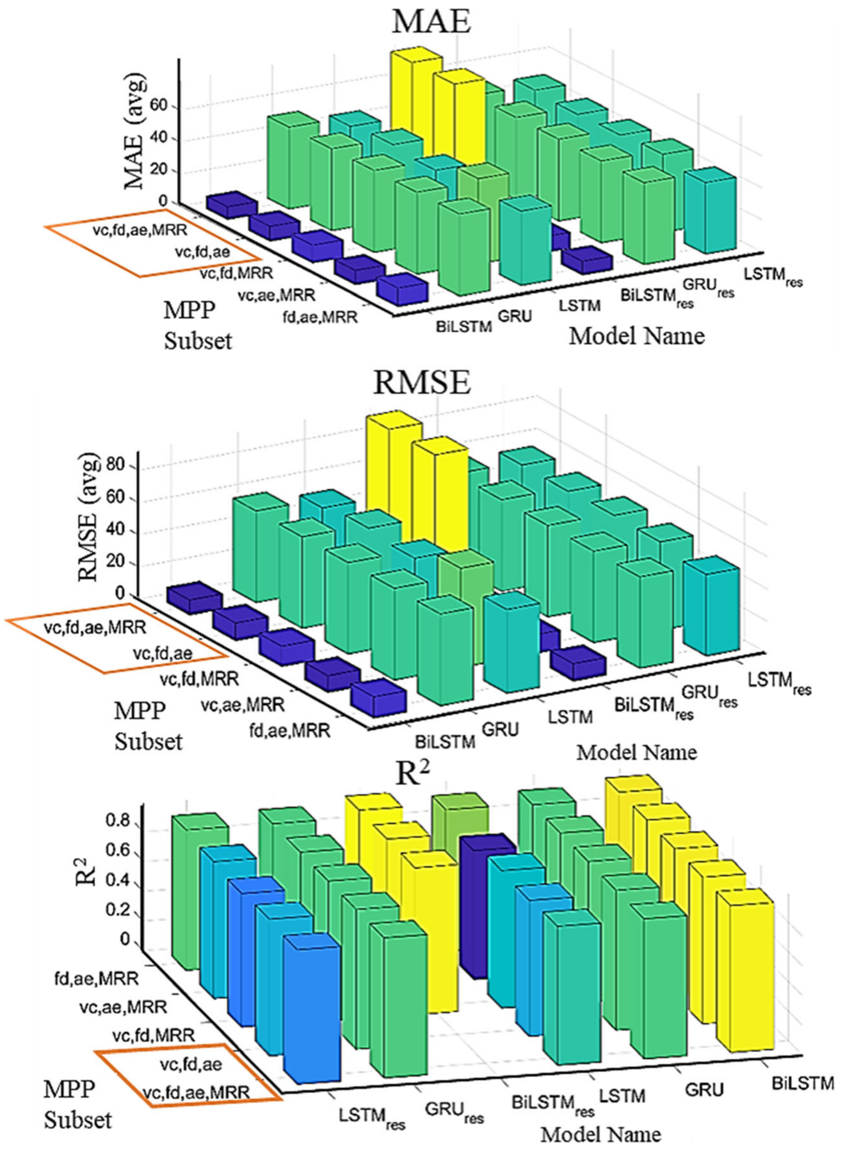


Fig. 11 Performance in terms of MAE, RMSE, and R^2 across model variants and MPPs subsets, with the highlighted models producing zero outputs for test sequences #5 and #14, corresponding to highest MAE, RMSE and zero R^2 values

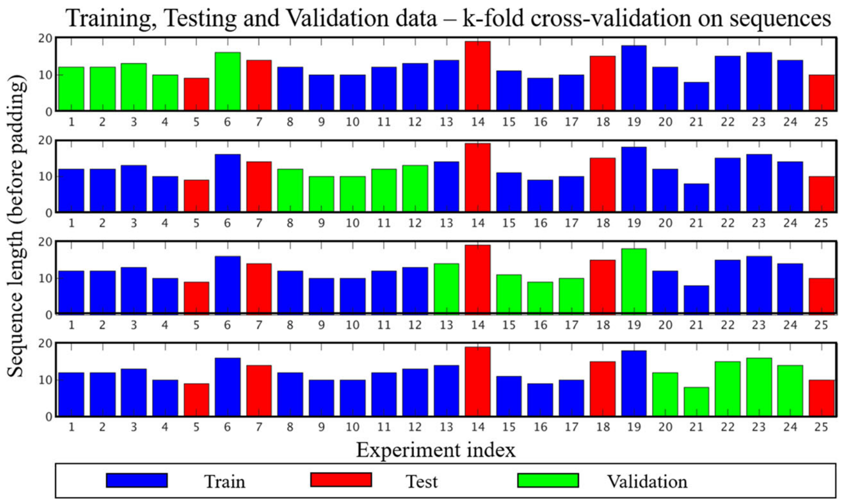


Fig. 12 Illustration of k-fold cross-validation scheme implemented on available sequences

α-λ accuracy shows that the prediction has remained within a decreasing amplitude margin (funnel-shaped), indicating progressively more accurate predictions. The convergence of a prognostics metric such as cumulative relative accuracy (CRA) could furthermore indicate model performance on the given sequence, indicating the accuracy of the model predictions as a whole, by evaluating the Euclidean distance between the point (t_s,0) and points on the error curve (t_C, E_C) [69, 70]. The logarithms of the CRA convergence for each sequence have bene included in Table 11, suggesting that the recurrent models generally retain good prognostic accuracy. The evaluation based on these prognostics metrics indicates that the Res-Bi-LSTM demonstrates the best performance across the tested models and MPP combinations. Nevertheless, some considerations for analysing the system's prognostic performance are needed to assess it suitability in the proposed setting. A summary of this evaluation is provided as follows.

- Determining α and λ: Based on the criticality and timeliness considerations of the intended maintenance action (i.e. tool replacement), a value of α = ±15% has been chosen for this research. It is important to note that lowering the α-threshold is not necessarily always desirable, due to the meaningfulness of the reported RUL at the given α. In other words, too small value of α for a time-critical application could be detrimental to the prognostic action, as the time window would be insufficient to address the impending failure in a timely manner. Also, t_λ is specified as 75% of the tool life of a given sequence. For other RUL prediction applications, the chosen values of α = ±15% and λ = 0.75 may be inappropriate.
- The α-λ accuracy indicates whether a sample's predictions continue to improve with time, as a logical selection metric where models are only selected if they retain RUL prediction accuracy over 75% of the cutting lifespan, i.e. λ = 0.75. As the tool's RUL cannot increase over time (i.e. it is monotone decreasing), adjusted PH scores to compare the models can incorporate a notion of usefulness of predictions when violations in RUL prediction accuracy occur due to noise.
- Furthermore, considering that the RUL in terms of minutes to failure scales with the cutting speed, it is less straightforward to determine the same constant threshold for all experiments. In other words, comparing RUL in units of distance and time could provide a more methodical approach to evaluating an appropriate value for selecting the α-margin.

Table 8 Model training hyper-parameters/options for the multimodal sequence RUL prediction

Hyper-parameter /option	Value	Hyperparameter/option	Value
Optimiser	ADAM	Gradient decay factor	0.90
Initial learning rate	4e-3	Squared gradient decay factor	0.9990
Learn rate drop factor	0.8	L_2 regularisation (global L_2-norm)	1e-4
Learn rate drop period	100	Mini-batch size	5
Max epochs	500	Validation frequency	50

Table 9 MPP subsets for feature ablation to evaluate the impact on RUL prediction

Masked Feature	Feature alias
None	$\{v_c, f_d, a_e, MRR\}$
Cutting speed	$\{v_c, f_d, a_e, MRR\}$
Feed rate	$\{v_c, a_e, MRR\}$
Cutting depth	$\{v_c, f_d, MRR\}$
Material removal rate	$\{v_c, f_d, a_e\}$

Table 10 Summary of RUL prediction model's test performance (comparing series network to alternative with residual connections) showing the top-3 models and MPP feature subset across experiments

Exp	Name	Subset	MAE	RMSE	R^2
5	**Bi-LSTM (residual)**	$\{\boldsymbol{v_c}, \boldsymbol{a_e}, \boldsymbol{MRR}\}$	**4.584**	**5.685**	0.973
	Bi-LSTM (residual)	$\{v_c, f_d, a_e, MRR\}$	12.241	14.765	**0.983**
	Bi-LSTM (series)	$\{v_c, f_d, a_e, MRR\}$	14.286	15.064	0.978
7	**Bi-LSTM (residual)**	$\{\boldsymbol{v_c}, \boldsymbol{a_e}, \boldsymbol{MRR}\}$	**4.589**	**5.944**	**0.997**
	Bi-LSTM (series)	$\{v_c, a_e, MRR\}$	5.302	7.183	0.991
	Bi-LSTM (residual)	$\{v_c, f_d, a_e, MRR\}$	5.949	7.786	0.998
14	**Bi-LSTM (series)**	$\{\boldsymbol{v_c}, \boldsymbol{f_d}, \boldsymbol{a_e}\}$	**5.573**	**6.719**	0.9980
	Bi-LSTM (series)	$\{v_c, f_d, a_e, MRR\}$	10.891	12.070	**0.9983**
	Bi-LSTM (residual)	$\{v_c, f_d, MRR\}$	12.220	14.436	0.991
18	Bi-LSTM (series)	$\{v_c, a_e, MRR\}$	**2.730**	3.872	0.9986
	Bi-LSTM (series)	$\{\boldsymbol{v_c}, \boldsymbol{f_d}, \boldsymbol{a_e}, \boldsymbol{MRR}\}$	2.866	**3.738**	**0.9988**
	Bi-LSTM (residual)	$\{v_c, f_d, a_e, MRR\}$	4.116	5.752	0.9968
25	**Bi-LSTM (residual)**	$\{\boldsymbol{v_c}, \boldsymbol{f_d}, \boldsymbol{a_e}, \boldsymbol{MRR}\}$	3.020	3.778	**0.9919**
	Bi-LSTM (residual)	$\{v_c, f_d, MRR\}$	3.977	5.356	0.9914
	Bi-LSTM (series)	$\{v_c, f_d, MRR\}$	4.833	5.997	0.983

Table 11 Performance across the PHM analytics metrics, for models demonstrating a-lambda accuracy of 1. The results are for models for which α-λ accuracy was achieved, and sorted based on highest RA and PH, and lowest α-cycle and t_λ

Model	Subset	Exp	PH	α	t_λ	RA	$\log_{10}$(Cvg)
LSTM (residual)	$\{v_c, f_d, MRR\}$	5	24.80483	10	20	0.973423	3.491
BiLSTM (series)	$\{v_c, a_e, MRR\}$	7	129.7113	0	100	0.999493	4.180
BiLSTM (residual)	$\{v_c, f_d, MRR\}$	14	239.53	0	180	0.910441	15.037
BiLSTM (residual)	$\{v_c, a_e, MRR\}$	18	171.3049	0	120	0.995661	3.644
LSTM (series)	$\{v_c, f_d, a_e\}$	25	70.4597	6	60	0.87657	4.780

6 Conclusions and future work

This research demonstrates a new approach based on a hybrid CNN-LSTM model with transfer learning for accurate determination of cutting tool wear and RUL prediction. The CNN model, pre-trained on images of cutting tools under different viewing conditions and subsequently fine-tuned on a subset of the VI data, can predict the degree of wear with a low test RMSE, MAE, and high correlation. The model captures the empirical wear characteristics, across a diverse range of experiments with different cutting conditions. To improve the training loss of the CNN model, a transfer learning objective is used to reduce the distribution discrepancy between source and target domain observations. Notably, it was found that the choice of source domain settings can boost the downstream predictive performance of the CNN. In addition, an LTSM model is designed for RUL prediction by projecting the expected end of life of the tool at a given distance measurement interval. The extracted image features are aggregated with the machining process parameters as inputs to recurrent networks, using series and residual variants, for RUL prediction. The performance of the developed approach is evaluated in terms of performance metrics and prognostics analytics. Following MPPs ablation, it was determined that cutting speed was a critical predictor for hybrid RUL prediction, as performance considerably degrades without it. Hence, it is suggested to use the feed rate and cutting depth in the MPP subset, and leaving out MRR as it is proportional to feed rate and cutting depth.

Future research could expand on this work by comparing these results achieved in this research with the real values of cutting tools in more general conditions to validate the research. It is also expected to implement end-to-end training (instead of multi-stage training) of the deep CNN and recurrent models. This would improve usability and extend applicability across image-based PHM applications in which real-time degrading component observations can be made, and operating conditions could be incorporated into the overall model inputs. Furthermore, incorporating visualisation elements to characterise tool wear and prediction uncertainty in terms of prediction probability distributions would further increase the value of the proposed approach and improve its usability. Additionally, incorporating production KPIs such as uptime, availability, and cost per workpiece and impact of PHM knowledge could further enhance the value of the overall PHM integration within the CNC machining production system.

Author contribution Mohamed Marei is responsible for idea and methodology development, algorithm implementation and validation and manuscript writing; Weidong Li is responsible for supervision, idea and methodology discussion, and algorithm check and manuscript refinement; apart from the above contributions, Weidong Li is also responsible for funding support and manuscript finalisation.

Funding This research is funded by Coventry University, Unipart Powertrain Application Ltd. (U.K.), Institute of Digital Engineering (U.K.), and the National Natural Science Foundation of China (Project No. 51975444).

Data availability It is declared that no data or materials are available for this research.

Code availability It is declared that codes are not available for this research.

Declarations

Ethical approval The authors claim that there are no ethical issues involved in this research.

Consent to participate All the authors consent to participate in this research and contribute to the research.

Consent to publish All the authors consent to publish the research. There are no potential copyright/plagiarism issues involved in this research.

Competing interests The authors declare no competing interests.

References

1. Zhao R, Yan R, Chen Z, Mao K, Wang P, Gao RX (2019) Deep learning and its applications to machine health monitoring. Mech Syst Signal Process 115:213–237. https://doi.org/10.1016/j.ymssp.2018.05.050
2. Johansson D, Hägglund S, Bushlya V, Ståhl JE (2017) Assessment of commonly used tool life models in metal cutting. Procedia Manuf 11:602–609. https://doi.org/10.1016/j.promfg.2017.07.154

3. Kurada S, Bradley C (1997) A review of machine vision sensors for tool condition monitoring. Comput Ind 34:55–72. https://doi.org/10.1016/s0166-3615(96)00075-9
4. Lee J, Lapira E, Bagheri B, Kao H an (2013) Recent advances and trends in predictive manufacturing systems in big data environment. Manuf Lett 1:38–41. https://doi.org/10.1016/j.mfglet.2013.09.005
5. Lee J, Lapira E, Yang S, Kao A (2013) Predictive manufacturing system - trends of next-generation production systems. IFAC Proc Vol 46:150–156. https://doi.org/10.3182/20130522-3-BR-4036.00107
6. Tao F, Qi Q, Liu A, Kusiak A (2018) Data-driven smart manufacturing. J Manuf Syst 48:157–169. https://doi.org/10.1016/j.jmsy.2018.01.006
7. Nasir V, Sassani F (2021) A review on deep learning in machining and tool monitoring: methods, opportunities, and challenges. Int J Adv Manuf Technol 115:2683–2709. https://doi.org/10.1007/s00170-021-07325-7
8. Zhao G, Zhang G, Ge Q, Liu X (2016) Research advances in fault diagnosis and prognostic based on deep learning. In: 2016 Prognostics and System Health Management Conference (PHM-Chengdu). IEEE, pp 1–6
9. Kerr D, Pengilley J, Garwood R (2006) Assessment and visualisation of machine tool wear using computer vision. Int J Adv Manuf Technol 28:781–791. https://doi.org/10.1007/s00170-004-2420-0
10. Dutta S, Pal SK, Mukhopadhyay S, Sen R (2013) Application of digital image processing in tool condition monitoring: A review. CIRP J Manuf Sci Technol 6:212–232. https://doi.org/10.1016/j.cirpj.2013.02.005
11. Serin G, Sener B, Ozbayoglu AM, Unver HO (2020) Review of tool condition monitoring in machining and opportunities for deep learning. Int J Adv Manuf Technol 109:953–974. https://doi.org/10.1007/s00170-020-05449-w
12. Park JJ, Ulsoy AG (1993) On-line flank wear estimation using an adaptive observer and computer vision, part 1: Theory. J Manuf Sci Eng Trans ASME 115:30–36. https://doi.org/10.1115/1.2901635
13. ISO (2016) ISO 8688-2: Tool life testing in Milling. In: Int. Organ. Standarization. https://www.iso.org/obp/ui/#iso:std:iso:8688:-2:ed-1:v1:en. Accessed 22 Jun 2019
14. Peng R, Liu J, Fu X, Liu C, Zhao L (2021) Application of machine vision method in tool wear monitoring. Int J Adv Manuf Technol: 1–16. https://doi.org/10.1007/s00170-021-07522-4
15. Sipos Z (1986) Investigation of cutting performance of coated HSS tools made in Hungary. NME, Miskolc Egyetem
16. Kong D, Chen Y, Li N (2018) Gaussian process regression for tool wear prediction. Mech Syst Signal Process 104:556–574. https://doi.org/10.1016/j.ymssp.2017.11.021
17. Wang J, Zheng Y, Wang P, Gao RX (2017) A virtual sensing based augmented particle filter for tool condition prognosis. J Manuf Process 28:472–478. https://doi.org/10.1016/j.jmapro.2017.04.014
18. Wang J, Wang P, Gao RX (2015) Enhanced particle filter for tool wear prediction. J Manuf Syst 36:35–45. https://doi.org/10.1016/j.jmsy.2015.03.005
19. Sun H, Cao D, Zhao Z, Kang X (2018) A hybrid approach to cutting tool remaining useful life prediction based on the Wiener process. IEEE Trans Reliab 67:1294–1303. https://doi.org/10.1109/TR.2018.2831256
20. Aramesh M, Attia MH, Kishawy HA, Balazinski M (2016) Estimating the remaining useful tool life of worn tools under different cutting parameters: A survival life analysis during turning of titanium metal matrix composites (Ti-MMCs). CIRP J Manuf Sci Technol 12:35–43. https://doi.org/10.1016/j.cirpj.2015.10.001
21. Maropolous P, Alamin B (1996) Integrated tool life prediction and management for an intelligent tool selection system. J Mater Process Technol 61:225–230. https://doi.org/10.1016/0924-0136(96)02491-0
22. Benkedjouh T, Medjaher K, Zerhouni N, Rechak S (2015) Health assessment and life prediction of cutting tools based on support vector regression. J Intell Manuf 26:213–223. https://doi.org/10.1007/s10845-013-0774-6
23. Wu D, Jennings C, Terpenny J, Gao RX, Kumara S (2017) A comparative study on machine learning algorithms for smart manufacturing: tool wear prediction using random forests. J Manuf Sci Eng 139. https://doi.org/10.1115/1.4036350
24. Chang WY, Wu SJ, Hsu JW (2020) Investigated iterative convergences of neural network for prediction turning tool wear. Int J Adv Manuf Technol 106:2939–2948. https://doi.org/10.1007/s00170-019-04821-9
25. Wu X, Liu Y, Zhou X, Mou A (2019) Automatic identification of tool wear based on convolutional neural network in face milling process. Sensors (Switzerland) 19:19. https://doi.org/10.3390/s19183817
26. Marei M, El Zaatari S, Li W (2021) Transfer learning enabled convolutional neural networks for estimating health state of cutting tools. Robot Comput Integr Manuf 71:11. https://doi.org/10.1016/j.rcim.2021.102145
27. Lutz B, Kisskalt D, Regulin D, et al (2019) Evaluation of deep learning for semantic image segmentation in tool condition monitoring. In: Proceedings - 18th IEEE International Conference on Machine Learning and Applications, ICMLA 2019. IEEE, pp 2008–2013
28. Ong P, Lee WK, Lau RJH (2019) Tool condition monitoring in CNC end milling using wavelet neural network based on machine vision. Int J Adv Manuf Technol 104:1369–1379. https://doi.org/10.1007/s00170-019-04020-6
29. Zhao R, Yan R, Wang J, Mao K (2017) Learning to monitor machine health with convolutional Bi-directional LSTM networks. Sensors (Switzerland) 17:273. https://doi.org/10.3390/s17020273
30. Wang P, Liu Z, Gao RX, Guo Y (2019) Heterogeneous data-driven hybrid machine learning for tool condition prognosis. In: CIRP Ann. https://www.sciencedirect.com/science/article/pii/S0007850619300083#fig0025. Accessed 10 Jul 2020
31. Zhang X, Lu X, Li W, Wang S (2021) Prediction of the remaining useful life of cutting tool using the Hurst exponent and CNN-LSTM. Int J Adv Manuf Technol 112:2277–2299. https://doi.org/10.1007/s00170-020-06447-8
32. 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. https://doi.org/10.1016/j.compind.2019.06.001, 1
33. Sun C, Ma M, Zhao Z, Tian S, Yan R, Chen X (2019) Deep Transfer learning based on sparse autoencoder for remaining useful life prediction of tool in manufacturing. IEEE Trans Ind Informatics 15:2416–2425. https://doi.org/10.1109/TII.2018.2881543
34. Zou Z, Cao R, Chen W, Lei S, Gao X, Yang Y (2021) Development of a tool wear online monitoring system for dry gear hobbing machine based on new experimental approach and DAE-BPNN-integrated mathematic structure. Int J Adv Manuf Technol 1–14. https://doi.org/10.1007/s00170-021-07470-z
35. Lei Y, Li N, Guo L, Li N, Yan T, Lin J (2018) Machinery health prognostics: a systematic review from data acquisition to RUL prediction. Mech Syst Signal Process 104:799–834. https://doi.org/10.1016/j.ymssp.2017.11.016
36. Si XS, Wang W, Hu CH, Zhou DH (2011) Remaining useful life estimation - a review on the statistical data driven approaches. Eur J Oper Res 213:1–14. https://doi.org/10.1016/j.ejor.2010.11.018
37. Krizhevsky A, Sutskever I, Hinton GE (2017) ImageNet classification with deep convolutional neural networks. Communications of the ACM, In, pp 84–90
38. Yu F, Koltun V, Funkhouser T (2017) Dilated residual networks. In: Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017. IEEE, pp 636–644

39. Li X, Jia X, Wang Y, Yang S, Zhao H, Lee J (2020) Industrial remaining useful life prediction by partial observation using deep learning with supervised attention. IEEE/ASME Trans Mechatronics 25:2241–2251. https://doi.org/10.1109/TMECH.2020.2992331
40. Weis W (1993) Tool wear measurement on basis of optical sensors, vision systems and neuronal networks (application milling). In: Proceedings of WESCON 1993 Conference Record. IEEE, pp 134–138
41. Wang Z, Oates T (2015) Imaging time-series to improve classification and imputation. IJCAI Int Jt Conf Artif Intell 2015:3939–3945 http://arxiv.org/abs/1506.00327
42. Tran M-Q, Liu M-K, Tran Q-V (2020) Milling chatter detection using scalogram and deep convolutional neural network. Int J Adv Manuf Technol 107:1505–1516. https://doi.org/10.1007/S00170-019-04807-7
43. Ma M, Mao Z (2019) Deep recurrent convolutional neural network for remaining useful life prediction. In: 2019 IEEE International Conference on Prognostics and Health Management, ICPHM 2019. Institute of Electrical and Electronics Engineers Inc.
44. Martínez-Arellano G, Terrazas G, Ratchev S (2019) Tool wear classification using time series imaging and deep learning. Int J Adv Manuf Technol 104:3647–3662. https://doi.org/10.1007/s00170-019-04090-6
45. Goodfellow I, Courville A, Bengio Y (2016) Deep learning, 1st–2nd ed. MIT Press. https://www.deeplearningbook.org/.
46. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9:1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735
47. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg AC, Fei-Fei L (2015) ImageNet large scale visual recognition challenge. Int J Comput Vis 115:211–252. https://doi.org/10.1007/s11263-015-0816-y
48. Yosinski J, Clune J, Bengio Y, Lipson H (2014) How transferable are features in deep neural networks? Adv Neural Inf Proces Syst 4: 3320–3328 http://arxiv.org/abs/1411.1792
49. Oquab M, Bottou L, Laptev I, Sivic J (2014) Learning and transferring mid-level image representations using convolutional neural networks. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp 1717–1724
50. FarajiDavar N (2015) Transductive Transfer Learning for Computer Vision Nazli FarajiDavar. University of Surrey
51. Dziugaite GK, Roy DM, Ghahramani Z (2015) Training generative neural networks via maximum mean discrepancy optimization. Uncertain Artif Intell - Proc 31st Conf UAI 2015 258–267. http://arxiv.org/abs/1505.03906.
52. Sun B, Saenko K (2016) Deep CORAL: Correlation alignment for deep domain adaptation. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). pp 443–450
53. Pan SJ, Tsang IW, Kwok JT, Yang Q (2011) Domain adaptation via transfer component analysis. IEEE Trans Neural Netw 22:199–210. https://doi.org/10.1109/TNN.2010.2091281
54. Csurka G (2017) A comprehensive survey on domain adaptation for visual applications. Adv Comput Vis Pattern Recognit:1–35. https://doi.org/10.1007/978-3-319-58347-1_1
55. Xiao D, Huang Y, Zhao L, Qin C, Shi H, Liu C (2019) Domain adaptive motor fault diagnosis using deep transfer learning. IEEE Access 7:80937–80949. https://doi.org/10.1109/ACCESS.2019.2921480
56. Cai H, Jia X, Feng J, Li W, Pahren L, Lee J (2020) A similarity based methodology for machine prognostics by using kernel two sample test. ISA Trans 103:112–121. https://doi.org/10.1016/j.isatra.2020.03.007
57. Li X, Jia XD, Zhang W, Ma H, Luo Z, Li X (2020) Intelligent cross-machine fault diagnosis approach with deep auto-encoder and domain adaptation. Neurocomputing 383:235–247. https://doi.org/10.1016/j.neucom.2019.12.033
58. da Costa PR de O, Akçay A, Zhang Y, Kaymak U (2020) Remaining useful lifetime prediction via deep domain adaptation. Reliab Eng Syst Saf 195:1–30. https://doi.org/10.1016/j.ress.2019.106682
59. Caires Moreira L (2020) Industry 4.0: Intelligent optimisation and control for efficient and sustainable manufacturing. Coventry University, , https://pureportal.coventry.ac.uk/en/studentthesis/industry-40(113974ee-2d36-44dd-8ebe-e102a24510b4).html.
60. Sethian JA (2002) A review of level set and fast marching methods for image processing. Modern Methods in Scientific Computing and Applications. Springer Netherlands, In, pp 365–396
61. Lathuiliere S, Mesejo P, Alameda-Pineda X, Horaud R (2020) A Comprehensive Analysis of Deep Regression. IEEE Trans Pattern Anal Mach Intell 42:2065–2081. https://doi.org/10.1109/TPAMI.2019.2910523
62. Hinton GE, Srivastava N, Krizhevsky A, et al (2012) Improving neural networks by preventing co-adaptation of feature detectors. http://arxiv.org/abs/1207.0580.
63. LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521: 436–444. https://doi.org/10.1038/nature14539
64. Goodfellow I, Bengio Y, Courville A (2016) Chapter 9 - Convolutional Networks. In: Deep Learning, 1st ed. MIT Press, Cambridge, MA, pp 326–366
65. Szegedy C, Ioffe S, Vanhoucke V, Alemi A (2016) Inception-v4, Inception-ResNet and the impact of residual connections on learning. Pattern Recogn Lett 42:11–24. https://doi.org/10.1016/j.patrec.2014.01.008
66. Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. 3rd Int Conf Learn Represent ICLR 2015 - Conf Track Proc. http://arxiv.org/abs/1412.6980.
67. Chen C, Chen Z, Jiang B, Jin X (2019) Joint domain alignment and discriminative feature learning for unsupervised deep domain adaptation. 33rd AAAI Conf Artif Intell AAAI 2019, 31st Innov Appl Artif Intell Conf IAAI 2019 9th AAAI Symp Educ Adv Artif Intell EAAI 2019 33:3296–3303. https://doi.org/10.1609/aaai.v33i01.33013296
68. Gretton A, Borgwardt KM, Rasch MJ et al (2012) A Kernel two-sample test. J Mach Learn Res 13:723–773 http://jmlr.org/papers/v13/gretton12a.html
69. Saxena A, Celaya J, Saha B et al (2009) On applying the prognostic performance metrics. Annu Conf Progn Heal Manag Soc PHM 2009 http://papers.phmsociety.org/index.php/phmconf/article/view/1621
70. Kim N-H, An D, Choi J-H (2017) Prognostics and Health Management of Engineering Systems. Springer International Publishing, Cham. https://doi.org/10.1007/978-3-319-44742-1.