Transfer learning enabled convolutional neural networks for estimating health state of cutting tools

Marei, M., El Zaatari, S. & Li, W.

DOI 10.1016/j.rcim.2021.102145
ISSN 0736-5845
ESSN 1879-2537

Publisher: Elsevier

Transfer Learning Enabled Convolutional Neural Networks for Estimating Health State of Cutting Tools

Mohamed Marei[1], Shirine El Zaatari[2], Weidong Li[3]*
Faculty of Engineering, Environment and Computing, Coventry University, U.K.
Emails: {mareim@coventry.ac.uk[1], elzaatas@coventry.ac.uk[2], weidong.li@coventry.ac.uk[3]
(Corresponding author) *}

Abstract

Effective Prognostics and Health Management **(**PHM) for cutting tools during Computerized Numerical Control (CNC) processes can significantly reduce downtime and decrease losses throughout manufacturing processes. In recent years, deep learning algorithms have demonstrated great potentials for PHM. However, the algorithms are still hindered by the challenge of the limited amount data available in practical manufacturing situations for effective algorithm training. To address this issue, in this research, a transfer learning enabled Convolutional Neural Networks (CNNs) approach is developed to predict the health state of cutting tools. With the integration of a transfer learning strategy, CNNs can effectively perform tool health state prediction based on a modest number of the relevant images of cutting tools. Quantitative benchmarks and analyses on the performance of the developed approach based on six typical CNNs models using several optimization techniques were conducted. The results indicated the suitability of the developed approach, particularly using ResNet-18, for estimating the wear width of cutting tools. Therefore, by exploiting the integrated design of CNNs and transfer learning, viable PHM strategies for cutting tools can be established to support practical CNC machining applications.

Keywords: Prognostics and Health Management (PHM), Transfer Learning, Convolutional Neural Networks (CNNs), Computerized Numerical Control (CNC)

1. Introduction

In manufacturing industries, unplanned downtime is known to negatively impact profitability, and will be a barrier to implementing lean and zero-defect manufacturing. Also, operational safety could be compromised through unexpected failures, particularly when human operators are involved [1]. To tackle the challenge, predictive maintenance based on Prognostics and Health Management (PHM) has been developed to predict the failure points of working components (such as bearings and cutting tools) [2–4]. Based on that, a component in a manufacturing system can be replaced just before it fails. Thus, component lifetime can be maximized, system downtime can be minimized, and therefore optimal productivity and production quality can be achieved. In Computerized Numerical Control (CNC) machining processes, cutting tool wear leads to various manufacturing problems, ranging from stoppage downtime for redressing and tool replacement, to scraps and reworks of machined components due to degradation in surface quality [5]. Therefore, accurate prediction of the Remaining

Useful Life (RUL) for a cutting tool is essential to mitigate such failures. In CNC machining applications, the flank wear width of a cutting tool (in mm) is defined in ISO8668-2:1989 as a criterion to judge the tool health (the tool is judged to be in a severer wear if the flank wear width is greater than a pre-defined threshold, e.g., 0.4mm). The flank wear is typically the most prominent degradation mode experienced by a cutting tool, in contrast to chip wear or abrasive wear that are less straightforward to distinguish [5]. The RUL is defined as the time remaining until the criterion of the flank wear width is reached.

For tool wear and RUL prediction, physics-based approaches on empirical models have been developed, such as the Taylor, Extended Taylor, Colding, and Coromant Turning model [5]. The Taylor model is used to map the relationship between tool life and cutting conditions such as cutting speed, or additional parameters such as feed rate and depth of cut (Extended Taylor), depending on a fixed wear criterion. Meanwhile, models such as the Sipos tool wear prediction model correlate the wear width to the cutting time in an exponential quadratic relationship, along with additional empirical parameters. However, these approaches are sensitive to variations in machining parameters (e.g., cutting speed, feed rate, cutting depth, cutting tool properties such as the number of teeth), which vary depending on component materials and machining set-up. Moreover, profound expert knowledge of machining processes is also expected to conduct effective and accurate RUL prediction. In contrast to physics-based approaches, data-driven approaches have been developed to leverage historical and real-time data to support decision-making.

Deep learning algorithms (e.g., Convolutional Neural Networks (CNNs)) have been explored to facilitate data-driven approaches (a related review can refer to [6]). For instance, to attain a wide scope of image features from a variety of applications, CNNs models can be trained on millions of images of natural and synthetic object, such as ImageNet [7] (and the subsequent ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [8]), CIFAR-10 and CIFAR-100 [9]. CNNs trained in this way excel at extracting discriminative features and learning hidden relationships between problem parameters (i.e., feature learning), opposed to feature engineering approaches where human experts specify the features to learn. The research community has also had a long-established interest in applying data-driven models (including deep learning algorithms) for PHM on CNC machining processes. Contributions in the relevant areas include sensor fusion techniques [10], integration of different deep learning algorithms for wear prediction [11-12], cyber-physical architecture design for CNC machine tool condition inspection [13], and exploring different sensing approaches for tool wear and RUL prediction [14]. However, the accuracy and reliability of deep learning enabled data-driven approaches may be reduced significantly when data are scarce or insufficiently descriptive of the problem. To address the issue, in recent years, transfer learning enabled approaches have been developed to improve pre-trained deep learning models to perform prediction on new problems with limited available data [15]. The transfer learning strategy is to retain and reuse domain-invariant features learn from a task in one domain (called the source domain), to enhance the performance of another task in its corresponding domain (called the target domain). On the other hand, though transfer learning has shown its great potentials in different applications, there are still limited research works reported on manufacturing applications, especially for estimating the wear prediction of cutting tools based on Visual Inspection (VI).

In this paper, it is aimed to develop a transfer learning enabled CNNs approach for tool wear prediction of cutting tools and further establish PHM based on analyzing the images of healthy and worn tools. The problem of limited data available during machining processes is tackled by integrating transfer learning into CNNs. The main characteristics of this research are summarized as follows:

1. Developing a transfer learning enabled CNNs approach with gradient descent optimization algorithms based on normalized Maximum Mean Discrepancy (MMD) and Mean Square Error (MSE) for algorithm training with learnable weights;
2. Benchmarking the performance of the approach through evaluating several CNNs architectures and transfer learning for tool wear prediction and PHM;
3. Providing recommendations for training techniques and data augmentation based on benchmarking and analysis results.

The rest of the paper is organized as follows. Section 2 reviews some works in PHM, focusing on some approaches that use CNNs to analyze different types of manufacturing data. Meanwhile, transfer learning related works are also reviewed. Section 3 describes the machining tool wear prediction dataset, the proposed methodology, and transfer learning approach. The performance of the approach and the suitability of the implementation are discussed in Section 4, offering some quantitative insights into the applicability of the approach as well as commentary on the results. Section 5 concludes the research and offers ideas for future works based on the proposed methodology.

2. Related Works

A brief review on prior works in deep learning for PHM applications is presented here with a focus on implementations in the manufacturing domain. Relevant literature aligning with CNNs and transfer learning are also highlighted.

2.1. Deep learning for PHM

In recent years, there have been increasing reviews on investigating the development of deep learning approaches for PHM applications [4]. Most deep learning approaches for PHM exploit an inflow of continuous real-time data, such as vibration signals, acoustic emission sensor data, force measurements, temperature, power/current, etc. Alternatively, other approaches were developed and tested with an existing RUL dataset at the validation stage (some examples of these datasets are mentioned in [6]). The C-MAPSS (aero-engine health degradation simulation) tool [10] was used to create datasets extensively studied in prior works on PHM, with varying research perspectives [11]. In addition, the PHM society holds an annual challenge for PHM techniques based on data that it provides, in which the 2010 dataset focuses on high-speed CNC machining [12]. While numerous architectures of deep learning were implemented for PHM, several primary models can be summarized into the following types:

- CNNs and their variants: these approaches use a series of shallow or deep convolutional filters that extract spatial features from images or image-like data. These approaches are particularly efficient at learning

meaningful representations of the data from individual and grouped clusters of filters across several depth channels [6]. Deeper CNNs layers are typically capable of extracting distinct visual features like parts of the general shape of the image, whereas shallower layers typically extract image intensity or color variation across a limited image window [6]. While predominantly used for failure classification in PHM [16], several researchers successfully used CNNs for regression-related feature extraction for RUL prediction [17]. A few approaches were developed to perform both classification and regression [17]. However, few approaches utilized images as input for predicting the health state of cutting tools.

- Recurrent Neural Networks (RNNs) and their variants: these models learn from data sequences with explicit time dependencies between input samples (i.e., sequence prediction problems), due to series of gates within the architecture which retain the outputs from previous inputs. The output from one neuron is fed forward into the adjacent neuron, so that the hidden representation or the output of the neuron is influenced by past inputs [6]. Long Short-Term Memory (LSTM) networks are similar to RNNs but can retain memory over a longer time horizon due to a more complex gate structure that preserves long-term dependencies. A combination of gates with different activation functions determine what portion of the cell state to retain or transform for input into the subsequent layer. LSTM and their variants have achieved widespread success at time series-based prediction problems, therefore are especially popular for PHM applications [18-19].
- Auto-Encoders (AEs) and their variants: AEs are feedforward neural networks comprising an encoder and a decoder. The encoder is trained to find the hidden representation of the input data via a nonlinear mapping of the weights and biases. The decoder attempts to generate an output to the inverse function of the hidden representation to the data, i.e., a nonlinear mapping between the encoder function and the hidden representation. It involves a reconstruction focus, where the encoder discovers a latent feature space with different target objectives, i.e., to reduce data dimensionality (increase sparsity), reduce noise (de-noising), or maximize distribution discrepancy (i.e., variational). To avoid human-assisted feature identification for Tool Condition Monitoring (TCM), Shi et al. used a novel Feature Multi-Space Sparse Auto-Encoder (FMSSAE) to classify four different wear states from TCM data in the time domain, frequency domain and Wavelet domain [20]. The presented methodology achieved 96% accuracy in classification. More recently, CNNs-AEs based strategies were developed with the aim of improving image feature qualities by encoding the image features with the encoder architecture, with the aforementioned objectives.
- Hybrid approaches: which combine and adapt several architecture designs to conduct feature extraction and time-based prediction, e.g., Zhao et al. used a Convolutional Bi-directional Long Short-Term Memory (CBLSTM) to extract local and informative sequence features, followed by a bi-directional LSTM to identify long-term time dependencies, leading to linear regression for target value prediction [21]. The approach was demonstrated to predict tool wear based on raw sensory data.

2.2. Transfer learning enabled deep learning

In the reviewed literature, an ongoing theme is to develop approaches that are trained or validated on publicly available datasets or those collected in individual experiments. Consequently, the replicability of these studies that leverage closed-source data may be called into question [8]. An additional limitation to the data being generated for PHM studies relates to the classic imbalance problem (whereby healthy data samples are much more prominent than faulty data samples) [8]. Meanwhile, the accuracy and reliability of the approaches are significantly hindered by insufficient manufacturing data: a key limitation for most deep learning approaches is their reliance on large quantities of data, typically in the order of ~1000 samples per class (for classification type problems).

Transfer learning has presented its potential to address the above problems [6]. With transfer learning, knowledge acquired from one domain might be retained and reused in a new domain. In general, the methodologies of transfer learning can be classified into the following four categories according to what and how the knowledge is transferred: (1) Instance based transfer learning - the labelled datasets from the source domain are reused in the target domain; (2) Feature based transfer learning - the features in the source domain are reused in the target domain if the features of the source domain match those in the target domain; (3) Parameter based transfer learning - the setting parameters of a machine learning algorithm in the source domain are re-used in the target domain; (4) Relational knowledge-based transfer learning - the relationship between the data from the source domain and the target domain is established, which is the base for knowledge transfer between the two domains.

In essence, transfer learning models repurpose the weights of deep learning approaches learned in classification tasks, corresponding to features in a similar or different domain (e.g., general-purpose image classification challenge datasets like ILSVRC [8] and Places [22]), to perform predictions for a new task. Such models typically achieved remarkable success, leading to a new research direction in exploring this generalizability by evaluating the classification or regression performance in new tasks (i.e., domain adaptation). In particular, for transfer learning, many approaches work well under a pre-requisite condition: the cross-domain datasets are drawn under the same feature distribution. When the feature distribution changes, most deep learning based approaches need to be re-trained from scratch.

In recent years, various research works have been conducted to integrate transfer learning into deep learning algorithms, i.e., deep transfer learning. For instance, Lu et al. developed a deep transfer learning algorithm based on deep neural network and feature based transfer learning for domain adaptation [23]. In the research, the distribution difference of the features between the datasets from the source domain and target domain was evaluated based on the Maximum Mean Discrepancy (MMD) in a Reproducing Kernel Hilbert Space (RKHS). Weight regularization was carried out by an optimization algorithm to minimize the difference of the MMD for the two domains in implementing knowledge transfer. Xiao et al. designed another deep transfer learning algorithm for motor fault diagnostics [24]. In the algorithm, a feature based transfer learning approach was developed to facilitate knowledge learnt from labelled data under invariant working conditions to the unlabeled

data under constantly changing conditions. MMD was incorporated into the training process to impose constraints on the parameters of deep learning to minimize the distribution mismatch of features between two domains.

2.3. Images in PHM applications

Traditionally, images are used within PHM for offline VI when assessing the condition of the damaged component or machine. However, similar image data could be used as a viable tool to predict (or localize) faults within a machine component, particularly if the frequency of such image measurements is sufficiently large. Most approaches use either 2D representations of time- or frequency-domain data (e.g., engine sensor data in [25], vibration signals in bearings in [3]). Comparatively few examples exist where the failure mode of a machine was classified by a pre-trained CNNs model based on visual data. In particular, Janssens et al. utilized pre-trained versions of the VGG-16 CNNs model [26]. The model was re-trained on thermal video images to classify the failure type, firstly by the image intensity, and secondly based on the degree of imbalance of the rotating machine. Their approach combines a CNNs trained to extract spatial features from thermal images, and another trained to extract temporal information from differenced images to represent the motion due to imbalances. In their first case study, their approach attained a 36.67% accuracy improvement over a conventional classification model, in classifying 12 different health conditions. Additionally, they implemented the same methodology to perform binary classification on lubrication state of another system, reporting an accuracy of 86.67% with the feature learning vs 80% with the conventional approach. Subsequent works in CNC tool VI have emerged with CNNs being applied for tool wear classification [27] and segmentation [28], with high accuracy. Another approach integrates both classification and segmentation with over 95% accuracy [29]. Meanwhile, more recent applications have investigated CNNs based tool wear prediction as regression [30].

The above research predominantly relies on modest sized datasets of images to perform tool wear prediction or failure classification for diagnosis. While demonstrating widespread success on intermediate classification or segmentation tasks, these methodologies do not apply directly for tool wear regressions. The methodology developed in this research addresses these shortcoming and leverages transfer learning enabled CNNs for tool wear prediction as a regression task, without intermediate classification outputs, and with minimal pre-processing.

3. Methodology

An overview of the developed methodology is described here first by introducing the overall workflow and then by detailing its constituent components. The overview is also illustrated in Figure 1.

3.1. Problem definition and overall methodology

The objective of this research is to predict the health of a cutting tool, given the image of the tool as an input and the corresponding normalized tool wear measurement as a prediction target. In other words, the objective is to determine:

$$\hat{Y} = w^T \mathrm{X} + b \tag{1}$$

where $\tilde{\boldsymbol{y}}$ is the matrix for the predicted regression output, w and b are the trainable weights and biases of a CNNs model respectively, i.e. trainable parameters, and X is the input matrix for the images.

A pre-trained CNNs model is deployed for predicting the wear state of the cutting tool. The parameters of the CNNs are then adjusted through adaptively learning the images of cutting tools based on a transfer learning process. To facilitate the CNNs model for the prediction, the end layers, which consist of the loss classifier and the classification output layer, are replaced with a designed regression layer stack.

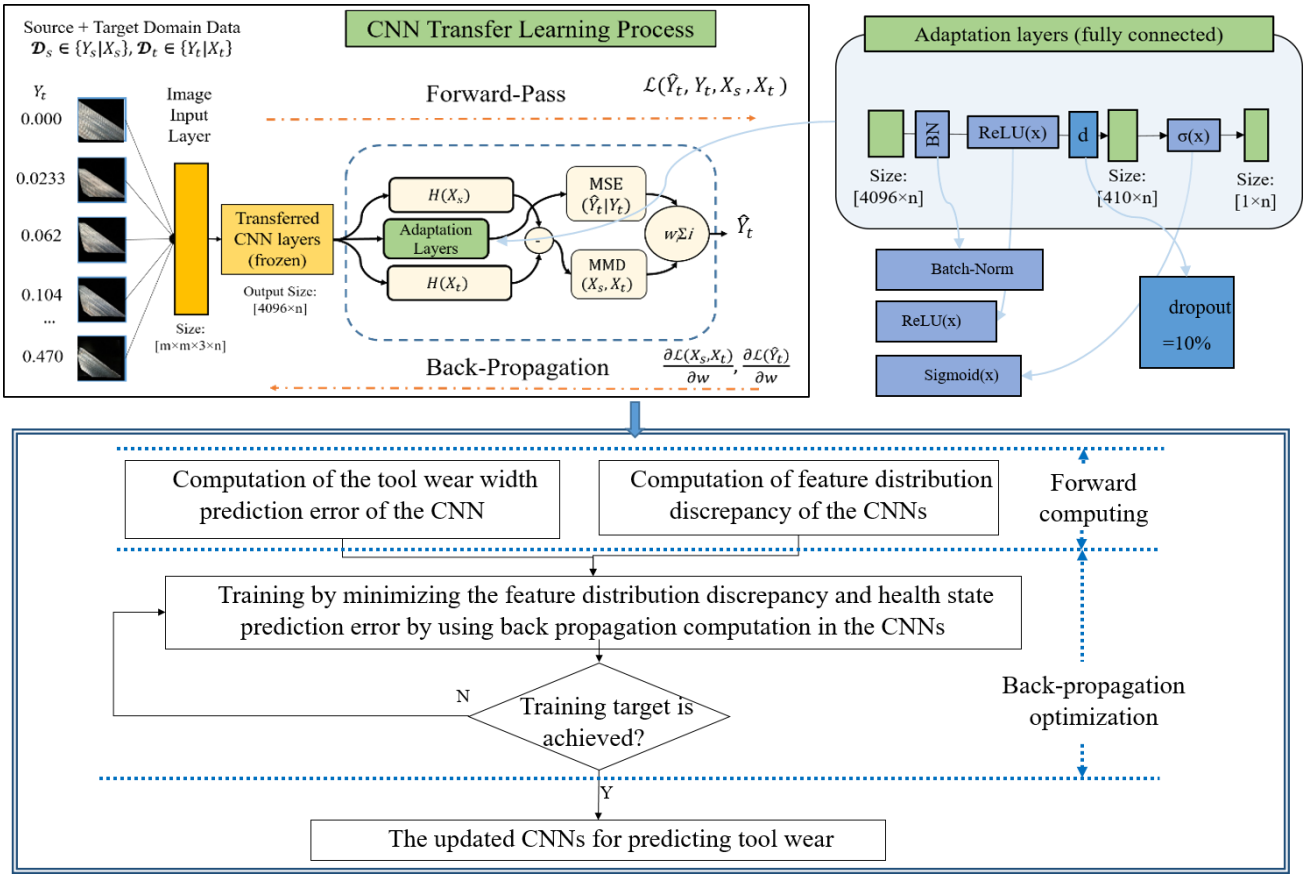

Figure 1. Conceptual overview of the CNNs deep transfer learning process for tool wear prediction.

More details of the developed approach are in the following steps: (1) forward computation of the CNNs is carried out by using the datasets of both the source domain (datasets for pre-training the CNNs) and the target domain (the images for tool heath state) as input for tool RUL prediction; (2) back propagation computation in the CNNs is performed to minimize the feature distribution discrepancy for the two domains and the prediction errors of the tool RUL. Gradient descent optimization algorithms are evaluated and used for the minimization process, and the updated CNNs is deployed for tool RUL prediction. illustrates the above steps, and further details on constituent components are given in the following sub-sections.

3.2. The input data for transfer learning

The dataset used for transfer learning comprises microscope images of the carbide cutting tool flank, collected at set intervals throughout the experiment, along with recorded flank tool wear width v in mm. The experiment conditions are recorded in Table 1. The cutting experiments were conducted under varying

conditions of cutting speed vc, feed rate fd, and cutting width ae. In total, 25 experiments were used to vary these parameters following a Taguchi Design of Experiments (DoE) approach, with 3 factors (vc, fd, and ae), 5 levels per factor, and 1 response variable (i.e., the measured flank wear width, V). The purpose is to analyze the effects of varying these parameters on the longevity of the tools, with a maximum lifespan of 240 m of cutting. A sample of the collected data is shown in Table 2.

In addition to the cutting tool images, the flank wear width (VB) was measured for each tool with a wear threshold of VB = 0.4 mm. Several image views of the cutting tool were recorded, but the analysis was focused on one particular variant of image views, at a magnification factor of 100 and a full side view of the tool flank.

Table 1. The experimental conditions used throughout the 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

Table 2. Sample of recorded tool wear experiment data, demonstrating the tool wear measurement intervals.

Exp	Cutting speed v_c (mm/min)	Feed rate f_d (mm/min)	Cutting width a_e (mm)	Flank Wear $w_{l=0m}$ (mm)	Flank Wear $w_{l=2m}$ (mm)	Flank Wear $w_{l=6m}$ (mm)	Flank Wear $w_{l=10m}$ (mm)	Flank Wear $w_{l=20m}$ (mm)	...	Flank Wear $w_{l=F}$ (mm)
1	150.72	300	1.60	0.000	0.0052	0.0104	0.016	0.0233	…	0.436
2	150.72	870	2.00	0.000	0.0061	0.0078	0.021	0.030	…	1.110*
…	…	…	…	…	…	…	…	…	…	…
25	251.20	2000	3.30	0.000	0.0074	0.012	0.029	0.073	…	0.662*

In total, 327 training images of cutting tools with appropriate tool wear width were used, split into 195 images (59.94%) for training and 132 (40.06%) for validation. To simplify benchmarking, the same pre-shuffled training and validation data were used. Out of the records obtained, experiments #5, 7, 14, 18 and 25 (20% of the dataset) were used for testing, reflecting a comprehensive sample of the different cutting lengths attained under different experimental conditions. The remaining sequences were used in training and observation. The function of image batch processing was implemented to perform a boundary cropping operation to remove the excess image backgrounds, yielding 800×800 pixel images which have been re-sized to each CNN network target input size, within the data augmentation procedure. Examples of the image files are shown in Figure 2.To avoid discarding additional data that could be relevant, the training and validation data labels were normalized between 0 (indicating a healthy tool) and 1 (for a fully worn tool). Figure 3 illustrates the tool life trend of the 25 cutting tools within this experiment. The images of the post-processed cutting tools were captured for Experiment #1. In addition, some tool wear measurements (in particular those corresponding to early failure events, e.g., Experiment #15) had a final values $V < 0.4$ mm; others had much larger values (e.g., Experiment #21) with $v \sim= 1$ mm.

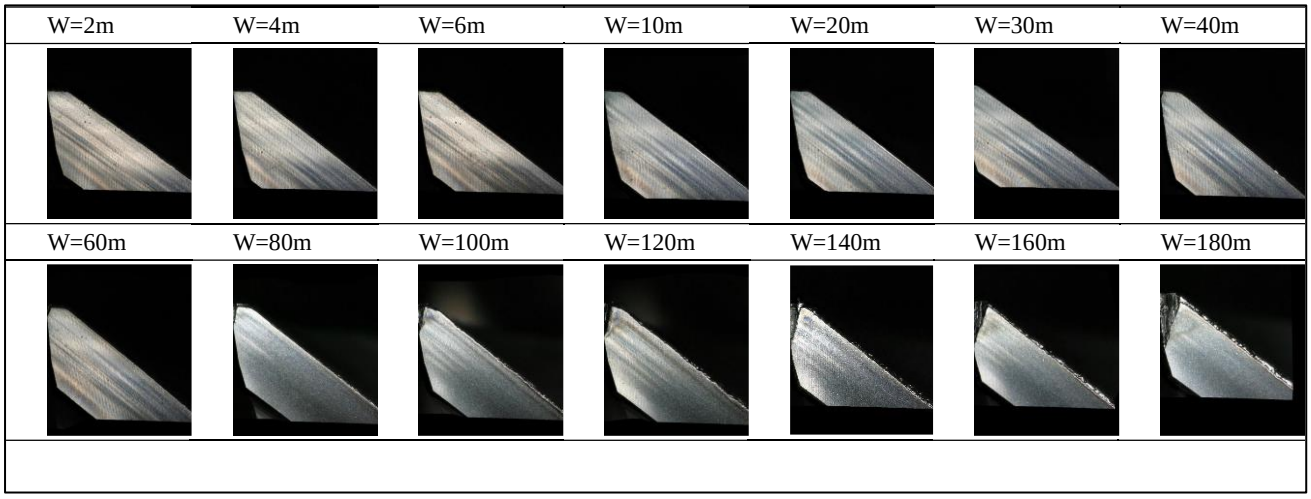

Figure 2. Experiment #1: pre-processed tool wear images recorded at W m cutting intervals. The cutting interval in earlier measurements was kept small to accurately capture the early wear trend

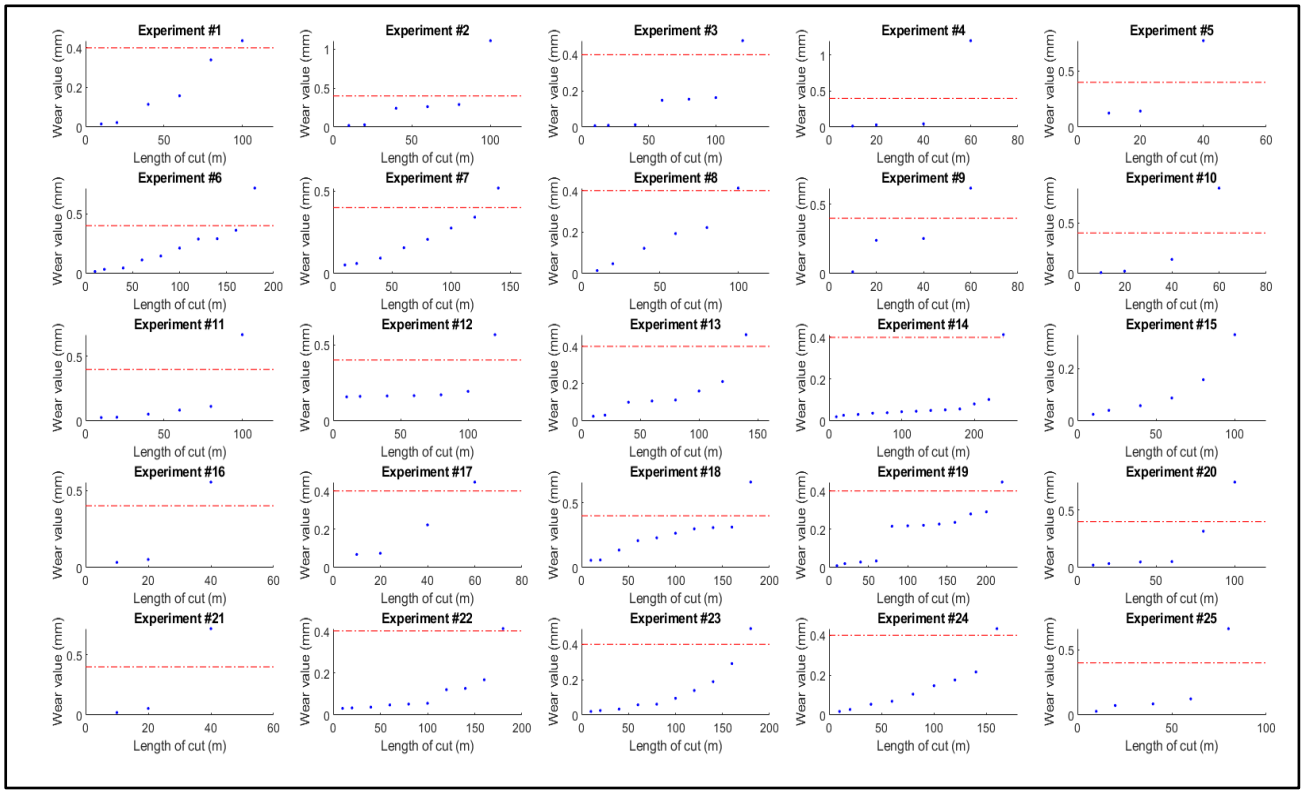

Figure 3. Tool life trends of 25 cutting tool experiments truncated crossing the V_B=0.4mm threshold (marked in red).

3.3. CNNs models and regression stack

For the approach developed in this paper, transfer learning allows the knowledge obtained from original tasks to be repurposed for different tasks. This means that the weights and biases of pre-trained CNNs models could be adjusted or fine-tuned with new training data. While the earlier feature pool layers of CNNs typically extract general image features that are considered safely transferable, specialized features are typically extracted in deeper layers. The degree of specialization often leads to some levels of pre-training bias, where the models retain features learned from the pre-training phase even despite being trained for extensive durations. It is intuitive to select models with a good performance in general classification to be further fine-tuned via transfer

learning. This is because such a model would have been trained to accurately recognize features belonging to a multitude of different classes. Feature transferability is addressed using minimization optimization procedures for MMD and tool wear prediction errors, described in Section 3.4.

When selecting pre-trained CNNs models for transfer learning, another important consideration is the computational complexity of the models. While it is often observed that deeper models tend to outperform shallower ones at certain tasks, it is not always the case. The SqueezeNet architecture, for instance, was able to attain AlexNet-level accuracy on the ImageNet data challenge with nearly 50 times fewer parameters [31]. Therefore, comparing a variety of CNNs models quantitatively is useful to help evaluate their merits and appropriateness of this research. The CNNs models were chosen based on their classification performances in general-purpose classification tasks. Table 3 highlights the model size, input image size, and results from classification challenges for the CNNs models.

Table 3. Pre-trained CNNs models investigated in this study, with performance reported in terms of top-1 and top-5 percentage accuracy when trained on ILSVRC 2012 [8].

Network	Top-1 Accuracy (%)	Top-5 Accuracy (%)	Parameters (Millions)	Input Size
AlexNet	63.3	84.6	61.0	227×227
ResNet-18	71.78	90.58	11.7	224×224
ResNet-50	77.15	93.29	25.6	224×224
ResNet-101	78.25	93.95	44.6	224×224
SqueezeNet	60.4	82.50	1.24	227×227
InceptionV3	78.95	94.49	23.9	299×299

The regression layer stack progressively adapt the outputs of the pre-trained CNNs to make them more suitable for regression-based prediction. It comprises the following layers:

- A 4096-channel fully-connected layer, which function is to further down-sample the outputs of the previous global or max pooling layer (which is a common design choice for CNNs models);
- A batch normalization layer, responsible for normalizing the inputs of the previous layer;
- Rectified Linear Unit (ReLU), which applies a non-linear transformation to the prior layer outputs, given by the function:

$$f(x) = \mathrm{ReLU}(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases} \tag{2}$$

- A 410 fully-connected layer, which down-samples the previous layer inputs;
- A sigmoid activation layer, which transforms the outputs of the previous layer to the range (0, 1) via the sigmoidal activation function;

$$f(x) = \mathrm{sigmoid}(x) = \frac{1}{1+e^{-x}} \tag{3}$$

A regression output loss on the prediction $\hat{\boldsymbol{y}}_i$, and targets $\hat{\boldsymbol{y}}_i$, which, for a mini-batch of N examples, calculates the Mean Squared Error (MSE):

$$\mathcal{L}(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i) = \mathrm{MSE}(\boldsymbol{y}_i, \hat{\boldsymbol{y}}_i) = \frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i)^2 \tag{4}$$

3.4. Transfer learning process

The CNNs model with the regression stack is re-trained on the training dataset of 195 images through the procedure illustrated in Figure 1. In order to transfer the knowledge from the source domain (pre-trained CNNs) to the target domain (trained CNNs for tool RUL prediction), the developed approach should be subject to the condition that features are in similar distributions between domains. To address feature distribution mismatch during transfer learning, an error minimization optimization strategy is applied through back propagation computing on the pre-trained CNNs. In the previous literature, MMD (Maximum Mean Discrepancy) was popularly used to measure the distance metric for probability distribution between two domains. That is, the datasets in the source domain and the target domain are represented as $D_S = \{X_S, P(x_s)\}$ and $D_T = \{X_T, P(x_T)\}$ respectively. Meanwhile, $X_S = \prod_{i=1}^{ns}\{x_s^i, y_s^i\}$ with n_s samples, and $X_T = \prod_{i=1}^{n_t}\{x_T^i\}$ with n_t samples respectively. Their MMDs are defined below:

$$\mathrm{Mean}_H(X_S) = \frac{1}{n_s}\sum_{i=1}^{n_s} H\left(x_s^i\right) \tag{5}$$

$$\mathrm{Mean}_H(X_T) = \frac{1}{n_t}\sum_{j=1}^{n_t} H\left(x_T^j\right) \tag{6}$$

$$\mathrm{MMD_H}(X_S, X_T) = \sup[Mean_H(X_S) - Mean_H(X_T)] \tag{7}$$

where $\sup(\cdot)$ represents the supremum of the aggregate; $H(\cdot)$ is a RKHS (Reproducing Kernel Hilbert Space).

In this research, the MMD is adopted for measuring the feature distribution difference of domain invariant features. To achieve similar distributions from two domains, $\mathrm{MMD}_H(X_S, X_T)$ is considered as the optimization objective to regularize the weights of the CNNs.

Due to the computational cost of calculating the MMD on the feature embeddings, a linear-time approximation of the MMD is used instead, as proposed by Gretton et al [50], taking the form:

$$\mathrm{MMD_l^2}(X_S, X_T) = \frac{2}{M}\sum_{\mathrm{i=1}}^{\frac{M}{2}} h_l(\boldsymbol{z}_i) \tag{8}$$

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

$$h_l(\boldsymbol{z}_i) = k(\boldsymbol{x}_{2i-1}^s, \boldsymbol{x}_{2i}^s) + k\left(\boldsymbol{x}_{2j-1}^t, \boldsymbol{x}_{2j}^t\right) - k\left(\boldsymbol{x}_{2i-1}^s, \boldsymbol{x}_{2j}^t\right) - k\left(\boldsymbol{x}_{2i}^s, \boldsymbol{x}_{2j-1}^t\right) \tag{9}$$

Meanwhile, during the re-weighting process on the CNNs, the prediction error should be minimized as well. Thus, the prediction error is considered as another optimization objective. The overall loss can therefore be calculated based on $\mathrm{MMD_H}(X_S, X_T)$ and MSE. Since $\mathrm{MMD_H}(X_S, X_T)$ and MSE are in different value ranges, normalization is required. In this research, Nadir and Utopia points are utilized to normalize the above objectives. The Utopia point ${z_i}^U$ provides the lower bound of No. i objective obtained by minimizing the objective as below:

$${z_i}^U = min\, f(i) \tag{10}$$

The Nadir point ${z_i}^N$ provides the upper bound of No. i objective by maximizing the objectives:

$${z_i}^N = \max_{1 \ll j < I} f(j) \tag{11}$$

where I is the total number of the objective functions. According to Equations (10) and (11), the normalized MMD and MSE can be calculated as:

$$\mathrm{NMMD_H} = (\mathrm{MMD_{H1}}(X_S, X_T) - {z_1}^u)/({z_1}^N - {z_1}^u) \tag{12}$$

$$\mathrm{NMSE} = (\mathrm{MSE} - {z_2}^u)/({z_2}^N - {z_2}^u) \tag{13}$$

where $\mathrm{NMMD_H}$ and NMSE are the normalized $\mathrm{MMD_H}(X_S, X_T)$ and MSE respectively. Finally, the total loss function $Loss$ can be calculated based on the weighted sum of the two normalized objectives:

$$\mathcal{L}_{total}\left(X_s, X_t, \hat{Y}, Y\right) = w_1 \cdot \mathrm{NMMD_H} + w_2 \cdot \mathrm{NMSE} \tag{14}$$

where w_1, w_2 are the weights of the two objectives, and $\sum_{i=1}^{2} w_i = 1$. The weighting serves to trade off the MMD minimization with the task loss objective. These are therefore set to w_1, w_2 = [0.9, 0.1].

Based on the above process, three variants of training optimization algorithm were investigated and compared, including Stochastic Gradient Descent with Momentum (SGDM), Root Mean Square Propagation (RMSProp) and Adaptive Moments (ADAM) [32]. SGDM has been a popular choice for training ANNs since its inception in 1999, and its subsequent resurgence when used in AlexNet. RMSProp is another popular algorithm for gradient descent training to eliminate the need for learning rate adjustment. ADAM combined the heuristics of both Momentum and RMSProp to achieve faster convergence. The CNNs models were trained according to the procedure illustrated in Algorithm 1.

Algorithm 1: MMD-MSE Computation for CNNs transfer learning.

---

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

end while

The models were trained for 750 epochs in 12 iterations per epoch (9000 iterations total), with a mini-batch size of 16 images. During each pass of training, the model was passed batches of source and target data, simultaneously. The training function was set to shuffle the mini-batch every epoch. To speed up training, validation was done every 40 iterations. During training, image augmentation operations were implemented on

each training mini-batch, to vary the input images by introducing some aspects of visual variation to the images (random translations, rotations, and scaling). This has the effect of inflating the dataset, allowing the CNNs model to consider more examples of the data than are available. The CNNs training procedure was implemented on a HPC computer in MATLAB 2018a, with the Deep Learning Toolbox models for the aforementioned CNNs pre-trained on the ImageNet 2012 dataset. The training was processed on an NVIDIA GPU, with 8GB RAM.

4. Experimental Results and Discussions

4.1. Test results for CNNs training

Table 4 details the benchmarking results for fine-tuning the 6 CNNs models by the transfer learning process, where the 3-run average of each model variant's output was recorded. To evaluate the quality of prediction, the models were assessed with the following performance criteria:

(1) Training time (in seconds).

(2) Mean Absolute Error (MAE), which is defined below:

$$\mathrm{MAE} = \frac{1}{N}\sum_{i=1}^{N} |\boldsymbol{y}_i - \hat{\boldsymbol{y}}_i| \tag{15}$$

(3) Root Mean Square Error (MSE), which is the square root of MSE.

(4) $acc_{10,20,30}$, the accuracies of all predictions are below 10%, 20% or 30% error thresholds from the targets. The threshold of the T percentage accuracy is given by:

$$acc_T = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}_T((\hat{\boldsymbol{y}}_i) \tag{16}$$

$$\mathbb{1}_T(\bar{\boldsymbol{y}}_i) \coloneqq \begin{cases} 1, & (\hat{\boldsymbol{y}}_i \geq T \times |\max(\boldsymbol{y}_i)| \\ 0 & otherwise \end{cases} \tag{17}$$

where the range of T is determined by $T \in [0.1, 0.2, 0.3]$.

Table 4. CNNs test performance results, with lower values of MAE and MSE indicating better performance.

Pre-trained Model	Training Details		Model Performance					
	Optimizer	Learning Rate	MAE	RMSE	acc10 (%)	acc_{20} (%)	acc_{30} (%)	Training Time (s)
AlexNet	adam	4e-5	0.0829	0.1684	81.68	90.84	91.60	2124.8
	sgdm	2e-2	0.0868	0.1723	79.39	90.08	91.60	1940.1
	rmsprop	4e-5	0.0903	0.1726	77.86	90.08	90.84	2027.6
ResNet-18	adam	4e-5	0.0773	0.1654	83.97	90.84	92.37	3358.4
	sgdm	2e-2	0.0820	0.1591	78.63	90.08	92.37	2649.0
	rmsprop	4e-5	0.0791	0.1594	80.15	90.08	92.37	2853.5
ResNet-50	adam	4e-5	0.0868	0.1764	80.92	86.26	90.84	14790
	sgdm	2e-2	0.1124	0.1967	74.05	84.73	90.08	9184.2
	rmsprop	4e-5	0.1050	0.1954	76.34	83.97	90.08	12689
ResNet-101	adam	4e-5	0.0833	0.1657	74.81	89.31	92.37	17750
	sgdm	2e-2	0.0992	0.1565	74.05	89.31	93.89	15122
	rmsprop	4e-5	0.0882	0.1740	77.86	86.26	90.84	16654
SqueezeNet	adam	4e-5	0.0891	0.1732	79.39	88.55	91.60	2151.8
	sgdm	2e-2	0.0868	0.1784	79.39	89.31	91.60	1763.5
	rmsprop	4e-5	0.0882	0.1710	77.68	88.55	91.60	1878.0
InceptionV3	adam	4e-5	0.0886	0.1784	79.39	85.50	91.60	20334
	sgdm	2e-2	0.1040	0.1931	77.10	85.50	90.08	14653
	rmsprop	4e-5	0.0916	0.1728	79.39	87.02	91.60	18780

The threshold of 10% accuracy indicates the percentage of prediction that falls within 10% of this error in either direction. In most cases, due to the proportion of healthy samples compared to faulty or worn tool images, the prediction errors are on the lower end of the range, i.e., ~10% below the target value. The performance measures of 20% and 30% accuracy are considered additional qualitative metrics indicating whether predictions can be accepted.

Table 1. Results for benchmarked pre-trained CNNs models (lower=better except for accuracy (10%, 20% and 30%)).

4.2. Result analysis and observations

In Table 3, comparing these results, ResNet-18 (trained with ADAM) can offer the best performance in terms of average prediction error and acc_{10}, with a reasonable training time considering the number of iterations attempted. ResNet-50 and ResNet-101 variants are both longer to train and generally less accurate based on MAE and MSE. Meanwhile, despite being much deeper than other models, the InceptionV3 variants were amongst the worst performing models considering MAE, RMSE and the accuracy thresholds. Furthermore, the increase in model depth from ResNet-18 to ResNet-101 has increased training time fivefold, without an improvement in performance. This emphasizes a key observation that increase of the model depth does not mean increase in prediction accuracy accordingly. Sample prediction outputs using the three optimizers chosen (ADAM, RMSPROP and SGDM) are illustrated in Figure 4.

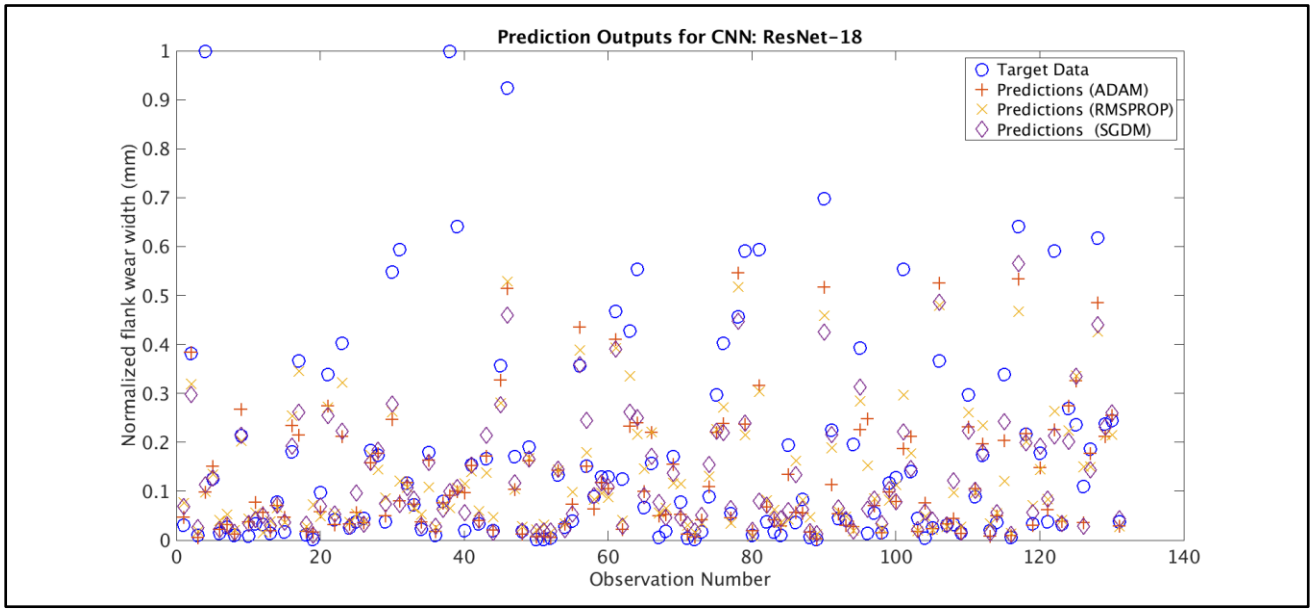

Figure 4. ResNet-18-based predictions vs targets using three optimizer variants.

Figure 5 shows a histogram plot of the prediction outputs of the six CNNs. Some further observations can be made regarding the performance of these models:

- Overall best fit: It shows that ResNet-18 produced the closest prediction output distribution to the validation target data, across the three optimizer training variants.
- Overfitting: All models over-fit the results significantly in the "healthy" categories, with the performance of ResNet-18 (ADAM) being the best out of the compared model variants in terms of overfitting, where the less the model over-fits, the better its performance.

- Generalization performance: Comparatively, ResNet-50 produced the worst general fit results, indicated by its comparatively higher MAE and MSE as well as lower accuracy across all thresholds. This might indicate that the model has a tendency to over-fit the data more strongly than other models. In fact, the generalization performance of SqueezeNet, which is close to 20 times smaller in parameters, is markedly better consider the relative difference in model size.
- Anomalous predictions: With the exception of ResNet-18, all models trained with SGDM have a tendency to produce negative outputs, despite the sigmoid layer (whose function is to force its outputs to be between 0 and 1) being the last layer prior to the regression output layer. This is a property of SGDM which enables it to generalize better than the other training algorithms [46]. However, in doing so the SGDM variants predict results in the reverse direction of what is desired. This contrasts to tool wear width values, which must always be indicated by a positive value.
- Training duration: It is worth mentioning that increasing the number of epochs to 9000 did not have a profound impact on the accuracy. Initial trials with fewer iterations (i.e., 150 instead of 750) yielded similar results for most of the models. It is common to select a short training duration for the fine-tuning process.

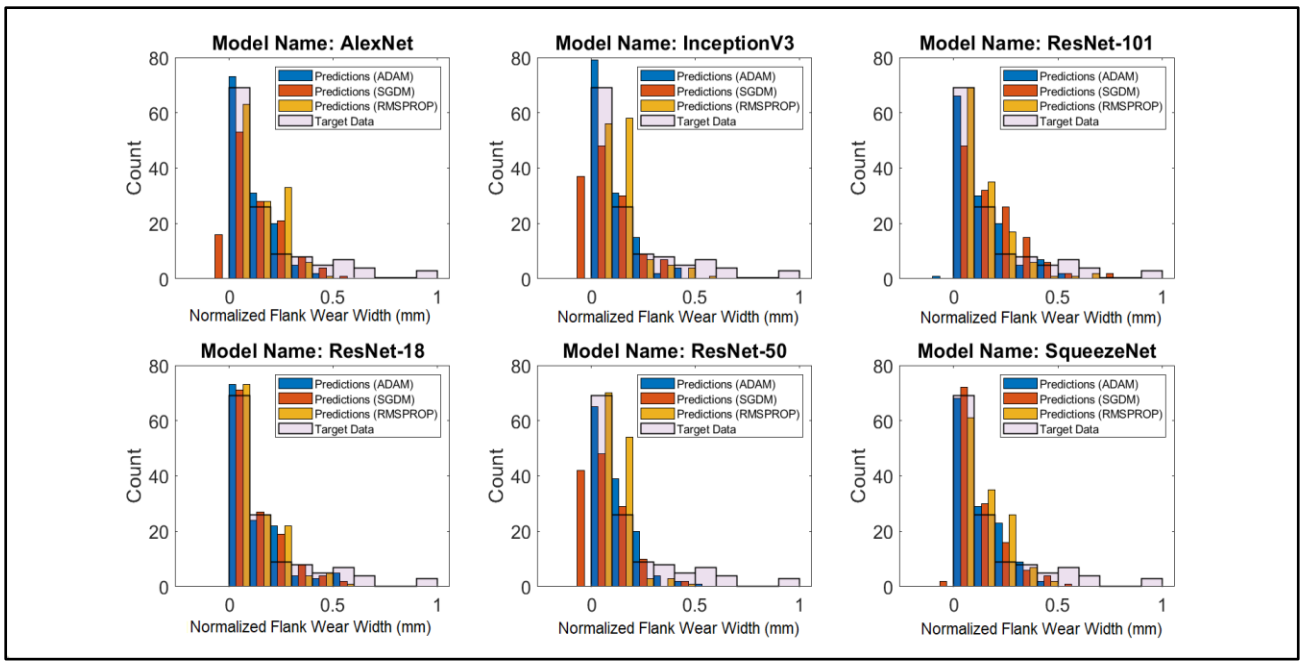

Figure 5. Prediction histogram comparison between six benchmarked CNNs models.

Figure 6 (a-d) compare the results (accuracy, log(training time), MAE, and MSE) from all the model variants (ADAM, SGDM and RMSPROP). ResNet-18 is clearly shown to have the highest average accuracy and lowest MAE, despite being slightly longer to train than AlexNet in training time. ResNet-18 is also amongst the best performing models for RMSE, bested only by ResNet-101 trained with SGDM. It therefore concludes that ResNet-18 is the best performed CNNs at learning a new task (regression output of normalized tool wear state) from images of tools using transfer learning.

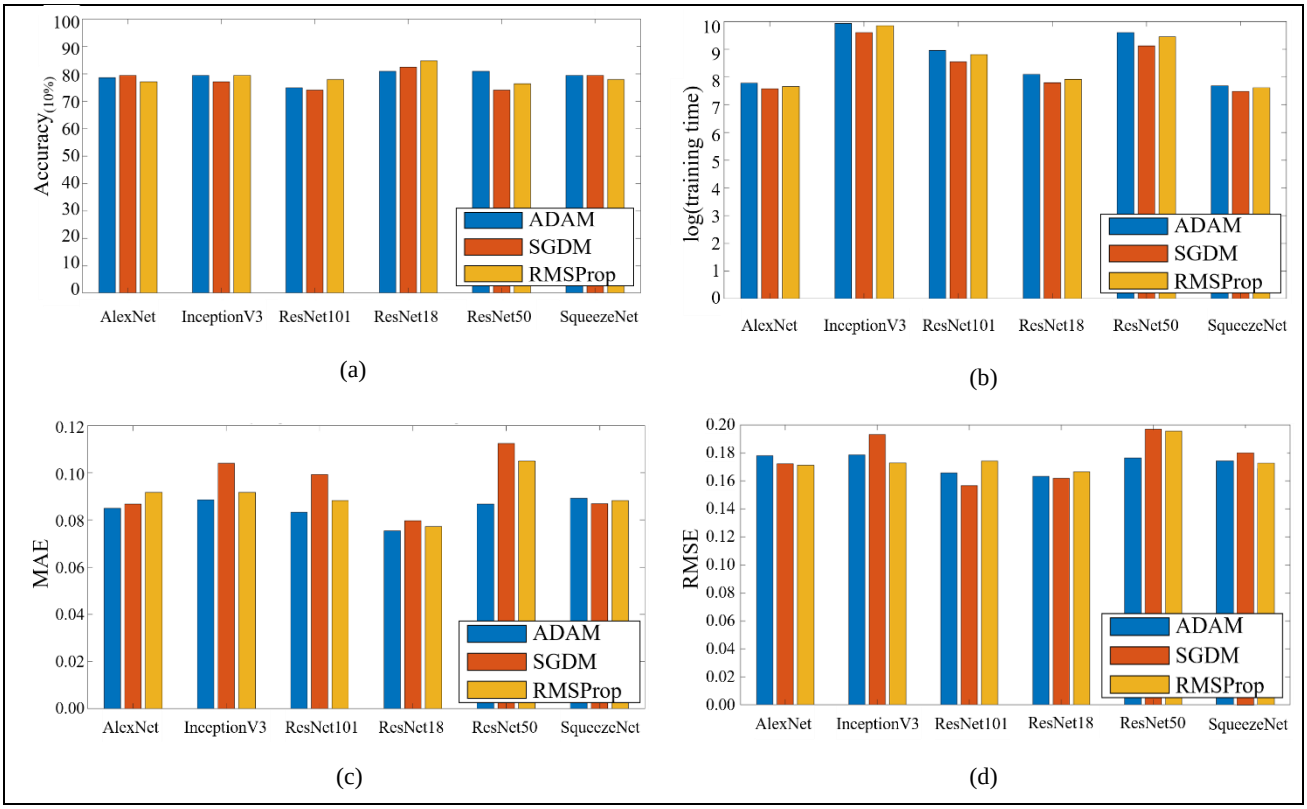

Figure 6. The performance of the test sets of CNNs transfer learning models; (a) accuracy (%), (b) log(training time), (c) MAE, (d) RMSE.

From the above analysis, the prediction workflow of tool health state based on transfer learning enabled ResNet-18 variants can be more effective in early stages (i.e., initial wear, with v>0.2mm). However, data imbalance and overfitting have considerable negative impacts on prediction accuracy, where classes are not uniformly distributed across the dataset. This is evident in the collected data in this research, where many more examples of healthy tools (i.e., $v < 0.4$) are available than those of less healthy tools ($v \geq 0.4$). This is made further apparent in Figure 7 that shows the distribution of the normalized training and testing dataset targets; there are much fewer values close to 1 in the normalized scale, corresponding to v values close to 0.4. Therefore, further investigations should be made:

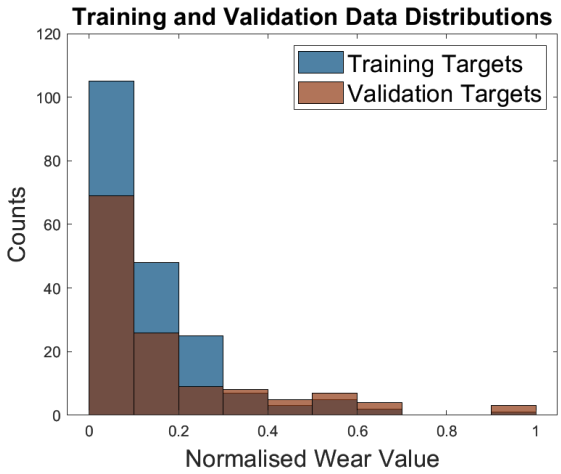

Figure 7. The data distribution of the training and validation target data.

- To address the imbalance between healthy and faulty tool states, classification-then-regression methods could be further explored, where weights are assigned based on class probability. Alternatively, cumulative attributes based methods could help improve accuracy by reformulating the regression problem in a manner similar to classification. Another alternative could be explored based on parameter transfer approaches, where a source task (and its corresponding source domain data) is used to pre-train the model.
- Additional works are required to improve the accuracy of prediction across increasing wear levels (i.e. where the normalized wear value exceeds 0.5). Some additional pre-manipulation of the data need to be implemented, by adding extra safety margins to the hand-measured wear values, for example. Increasing the cost parameters for the regression layer, for example by increasing regularization L2-norm penalties, could reduce overfitting.
- Investigating maximum likelihood estimation methods for regression could help with improving predictions across the full range of expected outputs, thereby reducing prediction bias.

The approach presented in this research could be further incorporated with additional inputs from CNC machining systems such as machining parameters, cutting material databases and cutting tool databases, to develop an intelligent PHM strategy for the machining systems. Additionally, other intelligent strategies such as reinforcement learning could be explored to further enhance the viability of the PHM strategy.

5. Conclusions

Deep learning algorithms have been increasingly applied for PHM due to their great potentials in the applications. Nevertheless, they are still ineffective in practical manufacturing applications as sufficient amounts of training dataset are not usually available. Seeking to overcome these limitations, in this paper, a transfer learning enabled CNNs approach is developed to effectively predict tool wear in CNC machining processes based on a limited number of the images of cutting tools. Quantitative benchmarks and analysis are conducted on the performance of the developed approach using several typical CNNs models and training optimization techniques. Experimental results indicate that the transfer learning approach, particularly using ResNet-18, can predict the health state of the cutting tool (as a normalized value between 0 and 1) with up to 84% accuracy and with a prediction mean absolute error of 0.0773. Based on these results, it demonstrates that the developed approach can achieve effective predictions on the health state of cutting tool in the early stages of tool wear.

A further research work is to integrate additional information to predict the tool RUL for increased accuracy (such as temperature, power dissipation, or current signals from the machine). Additionally, the approach to train the CNNs in this research can be incorporated directly into the PHM module for a CNC machine tool system, with the results of the predictive models being used to provide insights into improving the CNC machining process operations. The applicability of the methodology developed in this approach is not restricted to PHM of CNC machining alone; the methodology of transfer learning could be used for other applications with only limited datasets in a target domain available.

Acknowledgements

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). Special thanks to Dr. Lorena Caires Moreira, who has provided the dataset and corresponding annotations; to Dr. Yuchen Liang and Dr. Ibrahim Almakky, who have provided valuable input on CNNs design and transfer learning.

References

[1] M. Compare, L. Bellani, and E. Zio, "Reliability model of a component equipped with PHM capabilities," *Reliab. Eng. Syst. Saf.*, vol. 168, pp. 4–11, 2017, [DOI: 10.1016/j.ress.2017.05.024].

[2] Y. Liu, X. Hu, and W. Zhang, "Remaining useful life prediction based on health index similarity," *Reliab. Eng. Syst. Saf.*, vol. 185, pp. 502–510, 2019, [DOI: 10.1016/j.ress.2019.02.002].

[3] J. A. Ghani, M. Rizal, M. Z. Nuawi, M. J. Ghazali, and C. H. C. Haron, "Monitoring online cutting tool wear using low-cost technique and user-friendly GUI," *Wear*, vol. 271, no. 9–10, pp. 2619–2624, 2011, [DOI: 10.1016/j.wear.2011.01.038].

[4] Y. Lei, N. Li, L. Guo, N. Li, T. Yan, and J. Lin, "Machinery health prognostics: A systematic review from data acquisition to RUL prediction," *Mechanical Systems and Signal Processing*, vol. 104. pp. 799–834, 2018, [DOI: 10.1016/j.ymssp.2017.11.016].

[5] ISO, "ISO 8688-2: Tool life testing in milling," [Online]. Available: https://www.iso.org/standard/16092.html. [Accessed: 10-Feb-2021].

[6] R. Zhao, R. Yan, Z. Chen, K. Mao, P. Wang, and R. X. Gao, "Deep learning and its applications to machine health monitoring," *Mechanical Systems and Signal Processing*, vol. 115, pp. 213–237, 2019, [DOI: 10.1016/j.ymssp.2018.05.050].

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and F.F. Li, "ImageNet: A large-scale hierarchical image database," in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009, [DOI: 10.1109/cvpr.2009.5206848].

[8] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015, [DOI: 10.1007/s11263-015-0816-y].

[9] A. Krizhevsky, V. Nair, and G. Hinton, "CIFAR-10 and CIFAR-100 datasets." [Online]. Available: https://www.cs.toronto.edu/~kriz/cifar.html. [Accessed: 01-May-2019].

[10] J. Wang, J. Xie, R. Zhao, L. Zhang, and L. Duan, "Multisensory fusion based virtual tool wear sensing for ubiquitous manufacturing," *Robot. Comput. Integr. Manuf.*, vol. 45, pp. 47–58, 2017, [DOI: 10.1016/j.rcim.2016.05.010].

[11] H. Sun, J. Zhang, R. Mo, and X. Zhang, "In-process tool condition forecasting based on a deep learning method," *Robot. Comput. Integr. Manuf.*, vol. 64, 2020, [DOI: 10.1016/j.rcim.2019.101924].

[12] W. Luo, T. Hu, Y. Ye, C. Zhang, and Y. Wei, "A hybrid predictive maintenance approach for CNC machine tool driven by digital twin," *Robot. Comput. Integr. Manuf.*, vol. 65, p. 101974, 2020, [DOI:

10.1016/j.rcim.2020.101974].

[13] R. G. Lins, P. R. M. de Araujo, and M. Corazzim, "In-process machine vision monitoring of tool wear for cyber-physical production systems," *Robot. Comput. Integr. Manuf.*, vol. 61, 2020, [DOI: 10.1016/j.rcim.2019.101859].

[14] B. Cuka and D. W. Kim, "Fuzzy logic based tool condition monitoring for end-milling," *Robot. Comput. Integr. Manuf.*, vol. 47, pp. 22–36, 2017, [DOI: 10.1016/j.rcim.2016.12.009].

[15] K. Weiss, T. M. Khoshgoftaar, and D. D. Wang, "A survey of transfer learning," *J. Big Data*, vol. 3, no. 1, 2016, [DOI: 10.1186/s40537-016-0043-6].

[16] J. W. Oh and J. Jeong, "Convolutional neural network and 2-D image based fault diagnosis of bearing without retraining," in *Proceedings of the 2019 3rd International Conference on Compute and Data Analysis*, pp. 134–138, 2019, [DOI: 10.1145/3314545.3314563].

[17] D. T. Hoang and H. J. Kang, "Rolling element bearing fault diagnosis using convolutional neural network and vibration image," *Cogn. Syst. Res.*, vol. 53, pp. 42–50, 2019, [DOI: 10.1016/j.cogsys.2018.03.002].

[18] C.-S. S. Hsu and J.-R. R. Jiang, "Remaining useful life estimation using long short-term memory deep learning," in *Proceedings of the 2018 IEEE International Conference on Applied System Invention (ICASI), Chiba*, pp. 58-61, 2018, [DOI: 10.1109/ICASI.2018.8394326].

[19] J. Wang, G. Wen, S. Yang and Y. Liu, "Remaining useful life estimation in prognostics using deep bidirectional LSTM neural network, " in *Proceedings of the 2018 IEEE Progn. Syst. Heal. Manag. Conf.*, pp. 1037–1042, 2018, [DOI: 10.1109/PHM-Chongqing.2018.00184].

[20] C. Shi, G. Panoutsos, B. Luo, H. Liu, B. Li and X. Lin, "Using multiple-feature-spaces-based deep learning for tool condition monitoring in ultraprecision manufacturing," *IEEE Trans. Ind. Electron.*, vol. 66, pp. 3794–3803, 2018, [DOI: 10.1109/TIE.2018.2856193].

[21] R. Zhao, R. Yan, J. Wang, and K. Mao, "Learning to monitor machine health with convolutional bi-directional LSTM networks," *Sensors*, vol. 17, no. 2, pp. 273, 2017, [DOI: 10.3390/s17020273].

[22] PHM Society, "2010 PHM Society Conference Data Challenge | PHM Society," 2010. [Online]. Available: http://www.phmsociety.org/competition/phm/10. [Accessed: 30-Apr-2019].

[23] W. Lu, B. Liang, Y. Cheng, D. Meng, J. Yang and T. Zhang, "Deep model based domain adaptation for fault diagnosis, " *IEEE Trans. Ind. Electron.*, vol. 64, pp. 2296–2305, 2017, [DOI: 10.1109/TIE.2016.2627020].

[24] D. Xiao, Y. Huang, L. Zhao, C. Qin, H. Shi and C. Liu, "Domain adaptive motor fault diagnosis using deep transfer learning, " *IEEE Access*, vol. 7, pp. 80937-80949, 2019, [DOI: 10.1109/ACCESS.2019.2921480].

[25] G. S. Babu, P. Zhao, and X. L. Li, "Deep convolutional neural network based regression approach for estimation of remaining useful life," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9642, pp. 214-228, 2016, [DOI: 10.1007/978-3-319-32025-0_14].

[26] O. Janssens, V. Slavkovikj, B. Vervisch, K. Stockman, M. Loccufier, S. Verstockt, R. Van de Walle and S. Van Hoecke, "Convolutional neural network based fault detection for rotating machinery," *Sound Vib.*, vol. 377, pp. 331–345, 2016, [DOI: 10.1016/j.jsv.2016.05.027].

[27] X. Wu, Y. Liu, X. Zhou and A. Mou, "Automatic identification of tool wear based on convolutional neural network in face milling process," *Sensors*, vol. 19, pp. 3817, 2019, [DOI: 10.3390/s19183817].

[28] B. Lutz, D. Kisskalt, D. Regulin, R. Reisch, A. Schiffler and J. Franke, "Evaluation of deep learning for semantic image segmentation in tool condition monitoring," in *Proceedings of the 18th IEEE Int. Conf. Mach. Learn. Appl. ICMLA 2019*, pp. 2008–2013. 2019, [DOI: 10.1109/ICMLA.2019.00321].

[29] T. Bergs, C. Holst, P. Gupta and T. Augspurger, "Digital image processing with deep learning for automated cutting tool wear detection," *Procedia Manuf.*, pp. 947–958, 2020, [DOI: 10.1016/j.promfg.2020.05.134].

[30] X. Li, X. Jia, Y.-L. Wang, S.-J. Yang, H.-D. Zhao and J. Lee, "Industrial remaining useful life prediction by partial observation using deep learning with supervised attention," *IEEE/ASME Trans. Mechatronics*, vol. 25, pp. 1–10, 2019, [DOI: 10.1109/TMECH.2020.2992331].

[31] F.N. Iandola, S. Han, M.W. Moskewicz, K. Ashraf, W.J. Dally and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," in *Proceedings of the Int. Conf. Comput. Vis. Pattern Recognit*, pp. 1–13, 2016, [Online]. Available: https://arxiv.org/abs/1602.07360 [Accessed: 29-April-2018].

[32] S. Ruder, "An overview of gradient descent optimization algorithms," [Online]. Available: https://arxiv.org/abs/1609.04747 [Accessed: 15-May-2019].