The International Journal of Advanced Manufacturing Technology (2023) 127:5185–5198
https://doi.org/10.1007/s00170-023-11617-5

ORIGINAL ARTICLE

Anomaly detection in three-axis CNC machines using LSTM networks and transfer learning

Eugene Li[1] · Sanjeev Bedi[1] · William Melek[1]

Received: 24 November 2022 / Accepted: 20 May 2023 / Published online: 6 July 2023

Abstract
There is a growing interest in developing automated manufacturing technologies to achieve a fully autonomous factory. An integral part of these smart machines is a mechanism to automatically detect operational and process anomalies before they cause serious damage. The long-short-term memory (LSTM) network has shown considerable promise in the literature, with applications in the detection of tool wear and tool breakage to name a few. However, these methods require a significant amount of machine-specific training data to be successful, which makes these networks custom to a machine, requiring new networks and new data for each machine. Transfer learning is an approach where we use a network developed with a rich data set on one machine and re-train it with a smaller data set on a target machine. We have implemented this approach for chatter detection with a LSTM network, using sensor data and a rich data set from one machine, and then use a transfer learning methodology, similar sensors, and a smaller data set for the chatter detection algorithm on another machine. This allows for the transfer of knowledge from one machine to be applied to a similar machine, with some local optimization from transfer learning.

Keywords CNC machine · Anomaly detection · LSTM · Transfer learning

1 Introduction

Within the manufacturing environment, the idea of "lights out" manufacturing is one where the processes are all fully automated and require limited human presence on-site as parts are produced. This approach allows for increased productivity and lower upkeep costs [6]. CNC machines can be used in these environments to produce high-quality precision components. However, for this ideal to become a reality, fully autonomous, smart CNC machines must be developed. These smart machines must not only produce high-quality parts, but also predict when anomalies will occur or detect and recover from anomalies during the manufacturing process. This is required to ensure that a safe and continuous operating environment can be maintained without human intervention. Without this ability, the machine may produce unacceptable parts or damage itself or its surroundings; at worst, others in the surrounding area may be injured. Depending on the anomaly, there are different methods of recovering. This paper focuses on anomaly detection, and thus only the detection algorithms are considered.

Anomaly detection algorithms have been used to evaluate the remaining useful tool life for CNC tools, detect tool breakage, and detect chatter. These anomalies are important to detect, as they impact the performance of the machine and the quality of the parts made.

Many different techniques have been applied in the literature to tackle these problems. Sensing techniques using direct and indirect monitoring have both been investigated [29], along with model-based [30] and data driven approaches [15]. These methodologies have shown considerable promise in lab environments, but have yet to be widely implemented in production machines. The methods developed in the lab are specific to a certain machine, material, or process and

Sanjeev Bedi and William Melek contributed equally to this work.

✉ Eugene Li
eugene.li@uwaterloo.ca

Sanjeev Bedi
sbedi@uwaterloo.ca

William Melek
wmelek@uwaterloo.ca

[1] Department of Mechanical and Mechatronics Engineering, University of Waterloo, 200 University Avenue, Waterloo N2L 3G1, Ontario, Canada

require significant investment from the user. The transfer of these techniques from a lab setting to production is limited, as expensive equipment, long development time, and custom algorithms are needed. This limitation highlights the need to develop algorithms with the potential for easier implementation in production.

As computing capabilities have increased, the development of many advanced technologies has become possible. This technological shift has begun to usher in a new age in manufacturing known as Industry 4.0. These industry 4.0 technologies allow for advanced communications and algorithms, including machine learning and cloud computing to be implemented. This technological shift has expanded the potential algorithms that can be applied for anomaly detection.

Amongst the many algorithms that can be applied for anomaly detection, data-driven machine learning algorithms have proven to be an effective option. These approaches allow the system to learn the specific machine dynamics and account for the uncertainty introduced in the system, which is often difficult to model. Machine learning approaches vary from simple algorithms, such as support vector machine (SVM) [19], to more complex deep neural networks, and convolutional neural networks [29, 31]. One particular algorithm that has shown significant potential in this space is the long-short-term-memory algorithm (LSTM). This machine learning approach learns the behaviour of the system and, as the name suggests, is able to store its behaviour in both long and short term memory. The network is then able to use this stored memory to better reflect multi-period characteristics of long-term signals observed during industrial production [9].

The LSTM algorithm has been applied successfully in several works within the literature [22–25]. This approach has demonstrated its viability in lab settings for real time detection of anomalies such as chatter, and has shown potential for monitoring tool wear and tool breakage, as it is adaptive to learning the long term trends of a system, but fast enough to respond to changing dynamics.

Although successful, a limitation of many LSTM approaches is that it requires a sufficiently large and rich data set. These data sets are required for the algorithms to properly capture the specific machine and process dynamics of interest to effectively achieve the desired performance. However, obtaining an appropriate data set can be a hindrance in a production setting, as it is time consuming and potentially expensive. If such anomaly detection could be implemented in production machines in real time, then we can open the doors towards lights out manufacturing.

One approach that has been successful in reducing complications in data collection is to implement transfer learning [18, 26]. In transfer learning, a network is trained on an initial, often larger, set of training data obtained on a source system. The network weights are then frozen and "transferred" to another similar system, that shares the same physical inputs and outputs as the original system. The network is then re-trained on another, smaller set of training data, specific to this second system of interest. This approach allows for considerable efficiencies to be gained, as we are able to re-use much of the network, reducing the amount of data required, while obtaining similar accuracy. This approach significantly reduces the barriers to implementing many neural networks, and has successfully been applied in many different CNC applications [2, 5, 14, 18, 21, 26, 33, 35]. However, this approach has not been implemented using LSTM networks for anomaly detection. By combining the benefits of the LSTM algorithm and transfer learning, we are able to effectively apply these techniques in a scalable approach that can be applied to different materials and machines.

The objective of this paper is to develop an anomaly detection method that can be used on any similar machines with minimal training. This paper proposes an LSTM Autoencoder network that can be trained on a system with a large rich data set, and then re-trained with a smaller data set via transfer learning to detect anomalies during CNC machining. This approach is encapsulated in Fig. 1. This approach allows for networks to be re-trained on different, but similar machines with different materials, without significant resources spent obtaining new training data.

This paper aims to provide the reader with the necessary background to understand LSTM networks and the fundamentals of transfer learning. We then show how these approaches can be applied to transfer the knowledge from a rich data set on one machine, to a similar machine with less information. The organization of this paper is as follows: In Section 2, the necessary background information on the LSTM algorithm and transfer learning will be outlined. In Section 3, the methodology implemented will be discussed. In Section 4, the experimental design and results will be presented. Section 5 will offer a discussion on the findings and Section 6 will provide a conclusion.

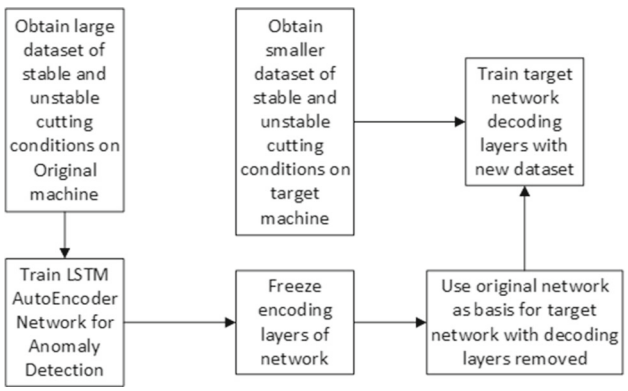

Fig. 1 Flowchart of algorithm design

2 Background

2.1 LSTM

The Long-Short-Term-Memory (LSTM) algorithm is a type of artificial neural network that is gaining popularity in the fields of artificial intelligence and deep learning. The LSTM network is a form of recurrent neural network (RNN), which, as its name suggests, is capable of storing both long term and short term memory about the system being modeled.

In general, a RNN is a special type of neural network that focuses on sequential or time series data. These types of networks are often used in temporal problems, such as language translation, speech recognition, or natural language processing [11]. These sequences are learned by adding connections between the nodes of the network, allowing the network to learn the relationship between each sequence of data, as well as the inter-relationship between the data during the current time step. This approach allows the network to develop a "memory" of the state of the system over time, where the state describes the system or process dynamics in a given instance of time.

Although RNNs are effective, the networks are prone to suffering from exploding and vanishing gradient problems. These problem occur when the networks grow too large due to many sequences being added. As a result of the size of the network, the algorithm is unable to properly update the network weights, preventing it from converging to the correct solution. This effectively causes the system to lose the ability to make connections between newer and older sequences, causing the system to lose its longer term memory.

To address the vanishing gradient limitation, Hochreiter and Schmidhuber developed the LSTM network [10], which was then further refined by Gers in [7]. The LSTM network allows the system to retain short term memory in excess of 1000 time steps, while maintaining long term memory of past events. This is of particular interest, since key events may have occurred with large gaps in-between one another [10].

In an LSTM, each unit is composed of a cell, an input gate, an output gate, and a forget gate. Each cell retains the memory for a time interval, with the gates controlling the flow of information in and out of the cell. The input and output gates, as their names imply, control the flow of information in and out of the cell. The forget gate calculates the proportion of information to be kept, thereby avoiding the state from growing indefinitely and breaking down [7]. An example of this could be a state representing a process variable in a system such as temperature. If the network does not have the ability to "forget" the impact of previous sequences, it may inadvertently increase the temperature beyond reasonable or accurate limits, causing the network to no longer effectively be able to model the dynamics, thus breaking down.

Table 1 Variables used in LSTM

Parameter name	Notation
x_t	Input vector to the LSTM unit
$g_t/\tilde{C}_t$	Cell input activation vector
C_t	Cell state vector
f_t	Forget gate's activation vector
i_t	Input vector to the LSTM unit
o_t	Output gate's activation vector

Following the description of the network outlined in [32], and developed by [10] and [7], the key equations of the LSTM network with forget gate are summarized below. In these equations, the lowercase variables represent vectors, while uppercase variables represents matricies. We begin by defining matrices $W \in \mathbb{R}^{h\times d}$ and $U \in \mathbb{R}^{h\times h}$ and vector $b \in \mathbb{R}^h$, which contain, respectively, the weights of the input and recurrent connections, where the subscript $_q$ can either be the input gate i, output gate o, the forget gate f, or the memory cell c. These weights and bias vector parameters are learned during training, where the superscripts d and h refer to the number of input features and number of hidden units, respectively.

$$g_t = \tilde{C}_t = tanh(W^g x_t + U^g h_{t-1} + b^g)$$
$$f_t = \sigma(W^f x_t + U^f h_{t-1} + b^f)$$
$$i_t = \sigma(W^i x_t + U^i h_{t-1} + b^i)$$
$$o_t = \sigma(W^o x_t + U^o h_{t-1} + b^o)$$

where the operator $\odot$ denotes the Hadamard multiplication, σ represents a sigmoid function and the subscript t indexes the time step. The variables used are summarised in Table 1.

2.2 Transfer learning

Transfer learning is a machine learning method where knowledge from a first system is used as a starting point for training a second related system, with similar dynamics and limited new information [34]. This method allows us to take an existing model trained on one data set and adapt it to predict another similar data set, saving significant time and cost by re-using many of the previous learned weights [4]. This approach is often used in vision processing tasks where there may be similarity in the images. For example, a network trained to identify cars can be used as a starting point to train a network to identify trucks.

The mathematical modelling of transfer learning is expressed in detail in [3], but is presented in brief below. Following the conventions defined by [34], we begin by defining a domain $\mathcal{D} = \{x, P(X)\}$, containing a feature space x and the probability distribution $P(X)$, where $X = x_1, ..., x_n$. A

task can then be presented by $\mathcal{T} = \{\mathcal{Y}, f(x)\}$, where y represents a label space and $f(x)$ a target function. These labels and target functions are learned from the training data consisting of pairs $\{x_i, y_i\}$, where $x_i \in X$ and $y_i \in \mathcal{Y}$.

Given a learning task $\mathcal{T}_S$ from source domain $\mathcal{D}_S$ and a target domain $\mathcal{D}_T$ and learning task $\mathcal{T}_T$, where $\mathcal{D}_S \neq \mathcal{D}_T$, or $\mathcal{T}_S \neq \mathcal{T}_T$, transfer learning aims to improve the predictive function $f_T(\cdot)$ in $\mathcal{D}_T$ using the knowledge in $\mathcal{D}_S$ and $\mathcal{T}_S$. This is accomplished by using the latent knowledge from $\mathcal{D}_S$ and $\mathcal{T}_S$. In most cases the size of $\mathcal{D}_S$ is much larger than $\mathcal{D}_T$ [34].

In practise, this is often accomplished by first developing a deep model based on the source domain data set. Following this, the final layers of the model are removed and replaced with new layers with randomly defined weights, or additional layers are added on top of the source model. The source model layer weights are then frozen and the new data set from the target domain is used to train the new layers [4].

For transfer learning to be successful from the source domain to the target domain, there must be sufficient overlap between the two data sets. This overlap ensures that both the source and target domain are reflecting similar environments, making the transferred information useful. For example, in the context of a CNC machine, if the source data contained only drilled holes but the target data was entirely curved surfaces, one would expect there to be little overlap.

This analysis can be accomplished by comparing the similarity between the training data in both the source and target domains using methods described in works such as [3] or more recently, using statistical measures such as the maximum mean discrepancy (MMD). The MMD metric was first described in [8] and has been used in works such as [36] and [20]. The MMD is a kernel based statistical test that is used to determine if two data sets represent the same distribution. In the context of transfer learning, if both the source and target training data have a low MMD score, then we can conclude that they both represent similar domains and transfer learning will be successful.

Given two distributions X and Y, we assume there is a feature map $\phi(*)$ that maps the distributions to a feature space F. The MMD is calculated as

$$MMD^2(X, Y) = \|\mu_X - \mu_Y\|_F^2 \tag{1}$$

where μ represents the mean of the distribution in the feature space.

This expression has a straightforward description, but is often difficult to implement. Thus, we use an empirical estimation by representing the system with a kernel $k(x_i, x_j)$. For this implementation, we have chosen a Gaussian kernel, as the data appears to be normally distributed. This estimation of the MMD is described as

$$\begin{aligned} MMD^2(X, Y) = & \frac{1}{m(m-1)} \sum_i \sum_{j \neq i} k(x_i, x_j) \\ & -2\frac{1}{m.m} \sum_i \sum_j k(x_i, y_j) \\ & +\frac{1}{m(m-1)} \sum_i \sum_{j \neq i} k(y_i, y_j) \end{aligned}$$

3 Methodology

3.1 Anomaly detection algorithm

CNC machines often operate for extended periods of time with little human intervention. Given the nature of the work for a CNC machine, we are interested in both long term changes in the machine over an extended duration and responding to dynamic changes in a short time horizon. Since we need to use both long term and short term knowledge, we need an intelligent method that is able to respond to both of these demands. The LSTM network provides an ideal opportunity to develop an intelligent monitoring solution capable of assessing the performance of the CNC machine by using both its long term and short term memory. An LSTM can be used for anomaly detection by implementing an encoder-decoder scheme as described in [23].

In this approach, we develop an LSTM network with several layers for encoding, an embedding layer and then a decoding layer; this network can be seen in Fig. 2. For

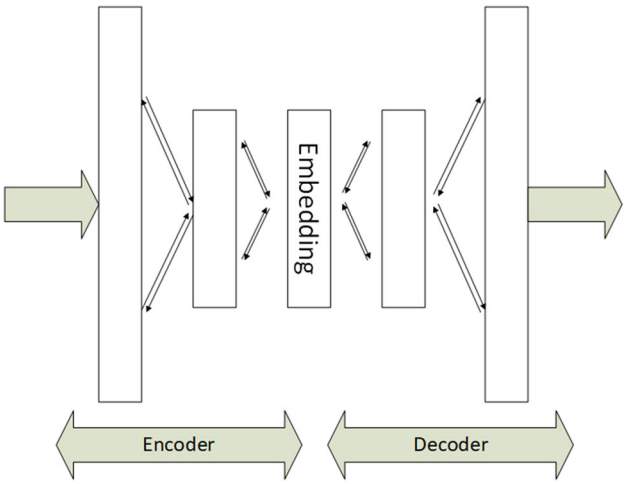

Fig. 2 LSTM autoencoder architecture

anomaly detection, we train this network to re-construct the original signal, implementing a one class classifier approach. This network, if provided with proper signals, will be able to re-construct the original signal with little error. However, if provided with erroneous signals from an anomaly, the network will re-construct the signal poorly, resulting in a large reconstruction error. We use this reconstruction error as an indicator of an anomaly.

This approach for determining the presence of an anomaly using the reconstruction error is possible because of the unique nature of the LSTM-autoencoder network. As outlined in the background section of this paper, the LSTM network is specifically designed to handle sequences of data. With the AutoEncoder framework, the first half of the network, the encoder, reads the input sequences and aims to learn its dynamics. The second portion of the network, the decoder, uses the long-term memory to interpret each step of the sequence and generate the output sequence. If we train the decoder to predict the output sequence based on stable conditions only, then it will only be able to predict stable conditions. Thus, if the network is properly trained to have low bias when given an input feature vector from stable conditions, then if provided with input data from an anomaly condition, it will predict the output sequence poorly. We can leverage this fact to detect anomalies in the system.

In the context of a CNC machine, we are training the system to reconstruct stable cutting conditions. As shown in [16] and [17], the sensor inputs, (e.g accelerometer, microphone, etc) are able to effectively describe the cutting state of the machine. The literature shows that there is a noticeable change in sensor inputs when the machine enters an anomaly state such as chatter. Thus, if the system is in an anomaly condition, the sensor inputs will reflect that, and the network output will also change accordingly.

To determine whether or not an anomaly is present, we plot the reconstruction loss for the system based on the training data provided by stable cutting conditions. This loss distribution allows us to define the mean value, and the three sigma interval to determine where the outliers occur. This allows us to define a threshold for which the reconstruction error would be considered too large, and we can thus conclude is an outlier. For our implementation, the training data was taken from a source machine, where the vibration of the spindle was measured during a wide variety of cutting operations. These operations, under stable conditions, had minimal deviation in the amount of vibration. A histogram for this loss distribution can be seen in Fig. 3, where the mean can clearly be seen and the loss threshold defined.

With this general outline in mind, the network can be designed. Through design iteration, it was found that an

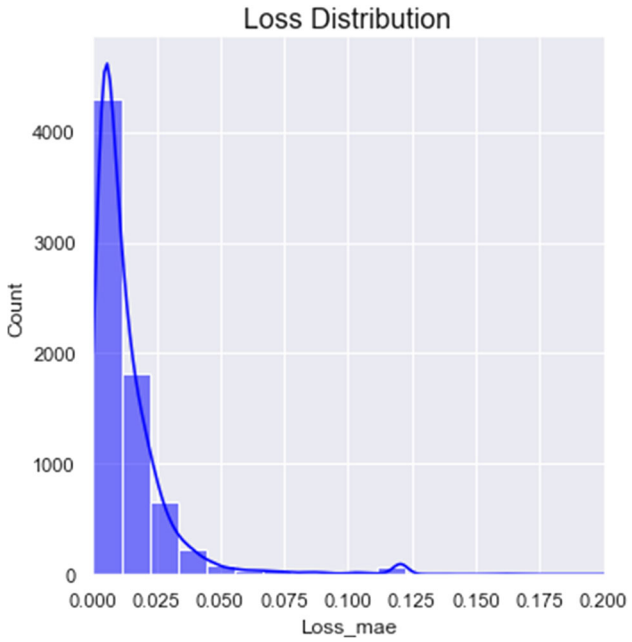

Fig. 3 Histogram of network loss distribution

input layer of three, and then three encoding layers and three decoding layers followed by an output layer produced sufficiently accurate results. The encoding and decoding layers used ReLu activation functions defined as

$$f(x) = max(0, x)$$

and the Mean Absolute Error (MAE) as the error function, defined as

$$MAE = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

The inputs to the network were obtained by taking a moving average window from the sensor inputs. This was implemented to provide some filtering of high frequency noise produced by the accelerometer. The description of the network can be seen in detail in Fig. 4, where the specific size of each fully connected layer is outlined. This architecture results in 45 347 trainable parameters. These parameters were then trained over 150 epochs using the training data.

With this architecture in mind, the LSTM Autoencoder network can be trained using a mini batch gradient descent model as described in [13]. In mini-batch gradient descent, we pick a small batch of data and feed it to the neural network. We then calculate the mean gradient error between the output and data, and then use this gradient error to update the weights, repeating this process for each batch. This is accomplished effectively by using vectored implementations. Once the LSTM Autoencoder network has been

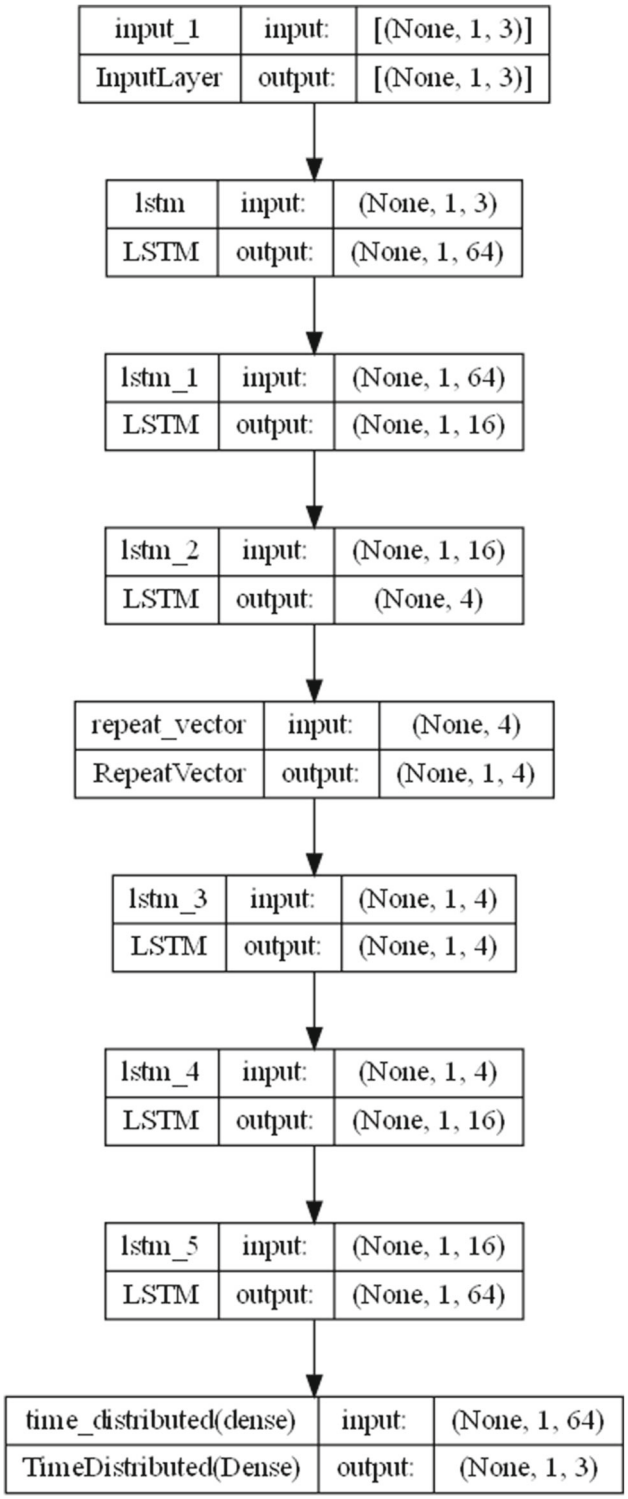

Fig. 4 LSTM autoencoder network architecture

trained for a specific machine and material, it can continuously be fed real-time sensor data and automatically detect if an anomaly has occurred.

3.2 Transfer learning implementation

To implement the transfer learning described in Sect. 2.2, we take a network trained on one system and use it as a starting point for training a different, but similar system. For this implementation we begin with a rich data set with over 200 000 data points for a 3-Axis CNC machine. This data comes from the open source data set described in [37]. In the open source data set, the authors were interested in developing an algorithm for evaluating machine quality. To achieve this, they recorded vibration data during the milling process using a three-axis accelerometer, all from stable cutting conditions, but with a variety of cutting parameters. This process resulted in a very large, very rich data set, that provides significant overlap with the cutting conditions experienced with our three-axis machines. Using this data set, we then developed the anomaly detection network using the methodology described in Sect. 3.1.

Once this network is trained, the encoding layers of the network are frozen. By freezing these layers, we are able to capture the relationship between the sensors and the machine dynamics. As outlined in [26], the general trends of stability lobes in CNC machines are well predicted and can be extended based on specific spindle speeds and depths of cuts. We extend this approach by noting that many of the machine dynamics are generally well predicted, and the network parameters can be adjusted by fine-tuning the weights to match the specific machine and material dynamics.

Since the encoder portion of the network focusing on capturing the relationship between the input signal and the machine, we turn our attention to re-training the decoder portion of the network. This approach allows us to address the specific dynamics required to predict the output sequence for the specific machine. By focusing our attention only on the decoder portion of the network, we are able to minimize the effort required to train our machine learning anomaly detection method.

With the network trained on the rich data set, a smaller data set is obtained on the target machine. This new data set is orders of magnitude smaller than the original data set and can be obtained quickly with a few tests cuts. The specific process followed is explained in greater detail in Sect. 4. Training data is obtained for stable cutting conditions on a desired material by taking simple linear test cuts to calibrate the machine for stable conditions. Unstable conditions were also obtained by inducing chatter in the CNC machine by taking aggressively large cuts during flank milling. The data from these trials was used for verification, but not included during the training of the network. The two data sets are then compared and the MMD metric can be determined. The MMD metric represents the statistical distance between the two data sets and ranges between zero to infinity, with a smaller MMD score representing closer overlap between the

training sets, indiciating a higher likelihood of successful knowledge transfer.

To improve the size of the new data set, data augmentation is applied to create more training data. Data augmentation is a method of producing artificial training data by adding slightly modified copies of existing data, allowing the network to avoid over-fitting by sampling the same training data repeatedly [28]. Through design iteration, it was found that dynamic time warping augmentation provided the greatest impact on improving the LSTM network's training accuracy. It was postulated that this improved the training accuracy of the system the most, as it most accurately represented different cutting scenarios where the same machine may have been moving at different feed rates, thus increasing the potential overlap between the source and target domains.

Once the new data set for the target machine has been obtained, the network can be fine-tuned for the target machine's specific dynamics. To accomplish this, we begin by taking the original network as a starting point for the target machine. We then remove the decoding layers of the original network that have not been frozen and replace them with new decoding layers with random weights. The network is then trained using the new data for the target machine. Once this process is completed, the network can be used to dynamically identify anomalies for the new target machine.

3.3 Experimental platform

Although the LSTM autoencoder network can be implemented in a variety of situations for anomaly detection, for our transfer learning implementation it was crucial that both the original and target machines share similar geometry and sensor inputs. This allows the network to properly establish a mapping between the input sensors and the output of the machine during reconstruction.

For this implementation, we focused on a three-axis CNC machine mounted with an accelerometer. This choice was selected as it is a common configuration seen in many CNC machines used in commercial and research environments. This choice also provided a wealth of open source experimental data that could be used to train and test the anomaly detection networks. Although this was the focus for this implementation, the methodology can be expanded to more complex system with higher degrees of freedom and different sensor inputs, so long as the original and target systems have similarity between the system inputs and process dynamics.

Despite the fact that the type of CNC machine and the sensors for both the original and target systems need to be similar, the training and target material do not need to share much similarity. This is expanded in detail in Sect. 4.

4 Experimental design and results

For this paper, three different sources of training data were used, one of the original system and two from different target systems. The training data for the original system was provided by the open source repository outlined in [37]. This data was obtained from a 3-Axis CNC Milling machine (TMV-720A) and a three axis Dytran-3293A accelerometer with a sample rate of 2048 Hz. This data was collected using a combination of Raspberry Pi 4 and a desktop PC. The data obtained was from milling operations on FDAC, a common pre-hardened high-strength steel, used in many manufacturing operations.

The first target machine is a desktop 3-Axis CNC milling machine designed for hobbyists and education (X-Carve [12]), seen in Fig. 5, and a three axis accelerometer (ADXL335), with a sample rate of approximately 200 Hz. This data was collected using a combination of an ESP32 microcontroller and a desktop PC. Experiments were completed on blocks of machining wax. This material was chosen as its cutting properties were well known, and repeatable experiments could be conducted.

The second target machine used was a Roland MDX-40A 3-Axis CNC milling machine also designed for hobbyists and education, and a three axis accelerometer (MPU9250), with a sample rate of 200 Hz. This data was collected using a Bluetooth module and a desktop PC. Experiments were collected on wood, as it is another material commonly used in CNC machines.

In addition to the accelerometer, additional sensors are mounted in the CNC enclosures to provide a record of the tests being completed. The data from these sensors are not actively used for any of the machine learning algorithms, but can potentially be used in future implementations. These sensors include, incremental encoders on the drive motors and spindle, an array of MEMS microphones (ICS-40180) and a HD camera. These sensors were recorded using separate ESP32 microcontrollers to avoid increasing the latency of the ESP32 controller connected to the accelerometer.

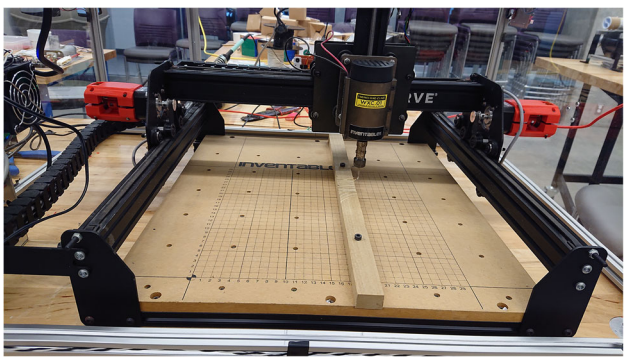

Fig. 5 Desktop CNC X-carve

During each of the tests, G-code commands were sent from the desktop computer to the CNC controller while the microcontrollers and camera collected the sensing information. In this regard, each of the devices acted independently, thus not introducing any latency to the other processes. This resulted in the data being post-processed, but with minor modification, could be processed in real-time.

Simple lateral cuts were completed on the blocks of materials during the collection of training data. These cuts allowed for precise control of the cutting conditions so both stable and chatter cuts could both be obtained. As outlined in previous sections of this paper, the stable cuts were used to train the network during transfer learning, and the chatter cuts were used to test the system's ability to detect anomalies. The chatter conditions were obtained by having a significantly large depth of cut and were verified by manually evaluating the sensor and video data. The training data was obtained by completing 15 lateral cuts, with each cut at approximately five seconds in length. Data augmentation was then applied to create approximately 4000 data points for training.

Once the training data was collected, it was used in post-processing to train the neural network through transfer learning. This was accomplished on a desktop PC using python scripts written with standard Tensorflow libraries. The anomaly detection bounds were obtained using the three-sigma statistical bound determined on the rich data set. This threshold was then carried over to the new target machine with no modification, and the decoding layers were re-trained using the new training data and the mini-batch gradient descent methods described previously in Sect. 3. The network trained for the target machine was then tested by providing sensor information from either stable or chatter conditions. The network evaluated this sensor information and attempted to identify if the machine was in stable cutting or chatter conditions. The results of the network analysis were then manually compared to the sensor and camera data to verify the results of the network. These results can be seen in further detail in Sect. 4.1.

4.1 Experimental results

With the experimental design described in Sect. 4, we were able to obtain experimental data to validate the aforementioned methodology. The first step in verifying the feasibility of the original data set was to decompose the training signals into the frequency domain, as seen in Fig. 6. This allows us to evaluate the data to ensure that it represents stable cutting conditions. As outlined in [1], we find that anomaly conditions, such as chatter, occur at frequencies that are distinct from the spindle vibrations during stable cutting conditions. As the figure shows, the frequencies are evenly distributed over the majority of the frequency bands, indicating that the data is largely composed of stable cutting signals.

With the quality of the training data confirmed, the network is trained and the anomaly detection algorithm tested. Following this, the stable and unstable cutting condition data on the target machine are collected. This data was collected on machining wax and an example of the acceleration signal from a stable cutting condition and a chatter cutting condition can be seen in Fig. 7, with the stable signal on the top subplot and the chatter signal on the second subplot. Due to the soft nature of the machining wax, the differentiation from the two signals is small, with the chatter signal exhibiting higher frequency oscillations due to the regenerative chatter.

The network was then tested using the signals seen in Fig. 7, with the stable signal and chatter signals both being fed to the networks. For validation, the original network is directly deployed on the target machine without applying transfer learning. The results of this network, trained on a different machine and different material can be seen in Fig. 8. In this figure, the blue line indicates the reconstruction error for the stable cut, while the red line represents the

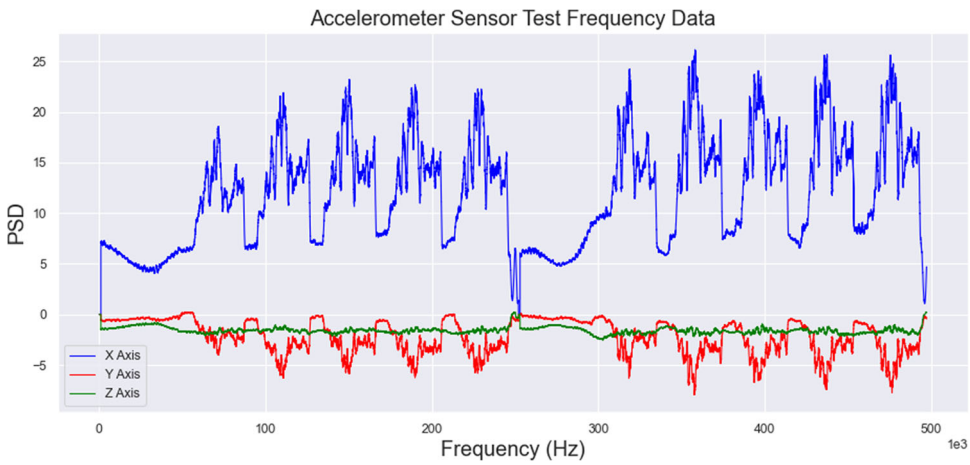

Fig. 6 Accelerometer sensor frequency data

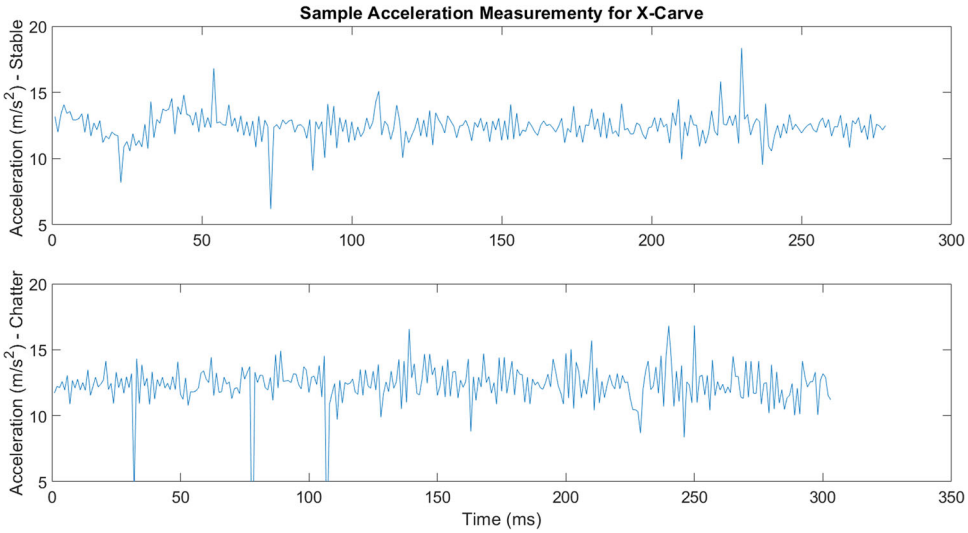

Fig. 7 Sample accelerometer signal from X-carve

reconstruction error for the chatter cut and the green line represents the chatter threshold. As the figure shows, the network is unable to distinguish the signals from the various cuts in any meaningful way.

As part of the design iteration during network development, rather than modify the original network, additional layers were added on top of the network. In this attempt, one fully connected layer with ReLu activation function and MAE error, and one time distributed output layer. These additional layers resulted in adding approximately 22 000 new parameters to be trained using the new data obtained on the X-Carve with approximately 4200 data points. This resulted in nearly 90 000 total parameters, 68 000 which had been trained and frozen from the original network. The MMD score for this data set was found to be less then seven, indicating a fair degree of correlation between the data sets. The results of this experiment can be seen in Fig. 9. As the figure shows, this is a significant improvement over Fig. 8. This network is able to distinguish between the stable and unstable cuts, but it is unable to determine the precise moments when the anomalies occur. Instead, this network classifies the entirety of the unstable cutting test as an anomaly, rather than the specific portions where the anomalies occur.

The next experiments involved removing the decoding layers, and training new decoding layers based on the data obtained from the target machine. This process is described in detail in Sect. 3, and the results of a sample experiment can be seen in Fig. 10. In this approach, the output layer and last three fully connected layers are all removed and new layers added. This results in 22 000 new parameters to be re-trained with the 4200 points of new training data, for a total of 45 000 parameters. As this figure shows, the stable cuts in blue and the unstable cuts in red can be clearly separated. In addition, the portions of the unstable cuts where the chatter anomalies occur, namely after 100 ms, are clearly defined as anomalies, which aligns with the accelerometer signal seen in Fig. 7. The presence of the anomalies was confirmed manually by comparing the recorded videos of the

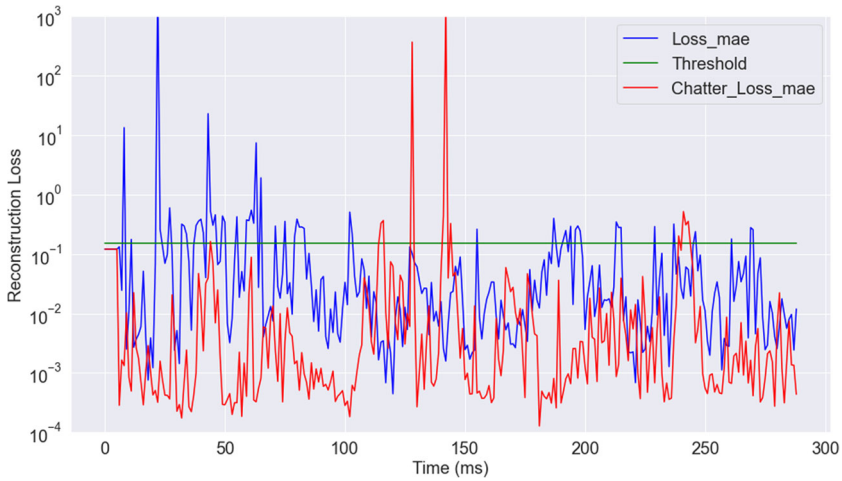

Fig. 8 Anomaly detection for X-carve CNC with LSTM network trained with original model

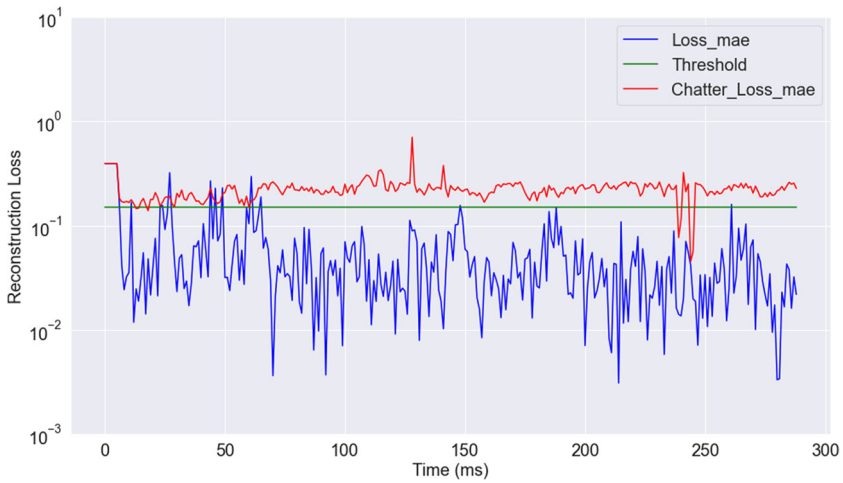

Fig. 9 Anomaly detection for X-carve CNC with LSTM network with additional layers added and trained with transfer learning

cutting experiments and listening for the onset of chatter, as well as verifying with the sensor measurements. These tests showed that the system can detect anomalies, as well as their onset.

This system was then tested by providing the newly trained network with a combination of stable and unstable cutting conditions. Specifically, three stable and two unstable cutting conditions were provided to the network, where the true state was known. For all of the cases, the re-trained system was able to clearly identify if the data originated from stable or unstable cases, and identify when the CNC machine had entered into an anomaly condition.

Following the success of implementing the transfer learning system on the X-Carve, the same approach was implemented on the Roland MDX-40A. The system was trained and tested with 10 different conditions, three from stable conditions and seven from unstable conditions. This network consisted of the last three fully connected layers and output layer being re-trained using approximately 5000 data points, re-training 22 000 parameters. These tests were conducted on wood with varying spindle speeds and depths of cut ranging from 3.0mm to 7.0mm. An example of one of the accelerometer readings from one of the experiments can be seen in Fig. 11. In this figure we can see the transition from stable cutting to chatter conditions, allowing us to establish a ground truth to determine if the system is in a stable condition or chatter condition.

This input was then fed into the transfer learning system to analyze when chatter conditions occurred. The results of this experiment can be seen in Fig. 12. As the figure shows, the network is able to progressively see the reconstruction error signal trend towards the anomaly threshold and is eventually able to clearly determine that an anomaly has occurred. A similar result was seen in all of the experiments, with the network able to correctly distinguish the chatter conditions. A comparison between the binary labels of stable and chatter conditions identified by the transfer learning system, and the ground truth can be seen in Fig. 13. As the figure shows, the

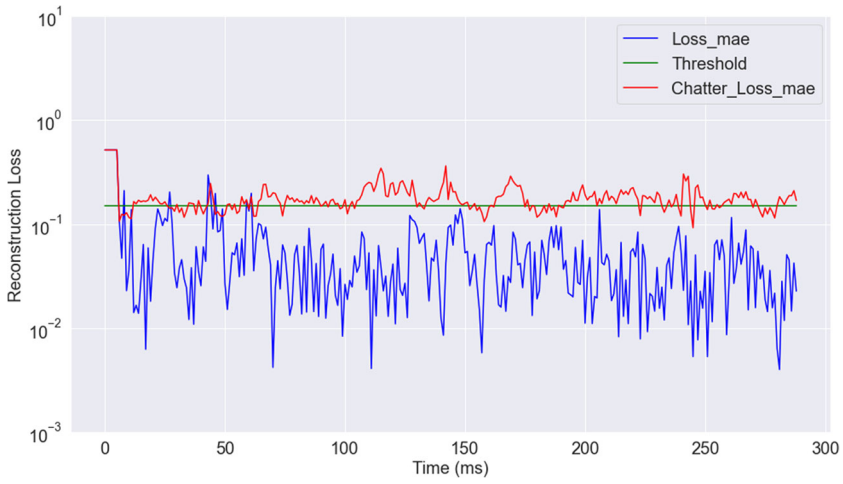

Fig. 10 Anomaly detection for X-carve CNC with LSTM network with decoding layers trained with transfer learning

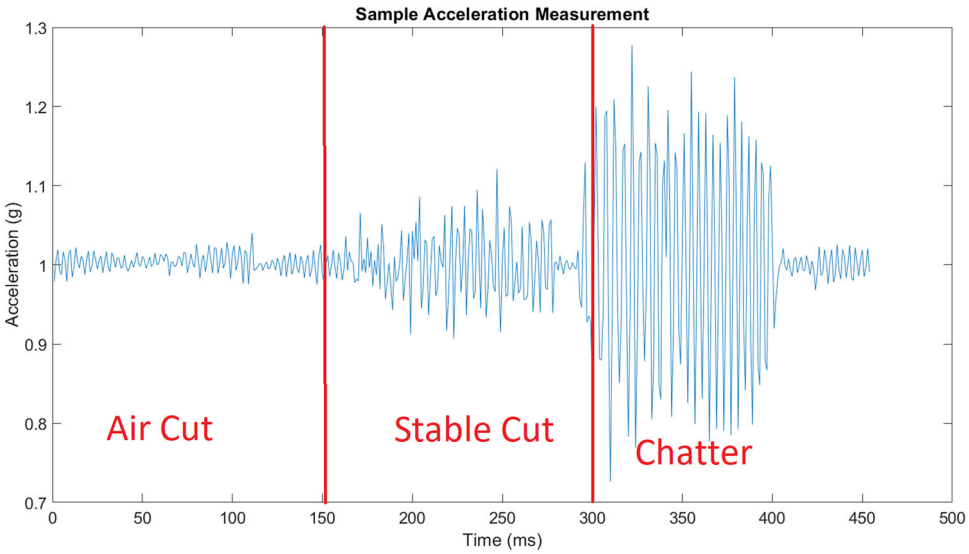

Fig. 11 Accelerometer input signal for Roland MDX-40A

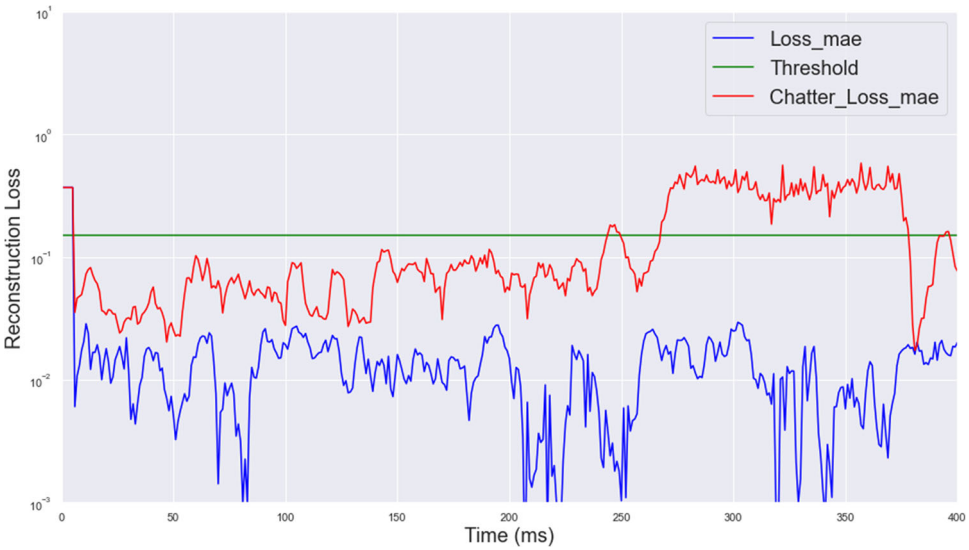

Fig. 12 Anomaly detection for Roland with LSTM network with decoding layers trained with transfer learning

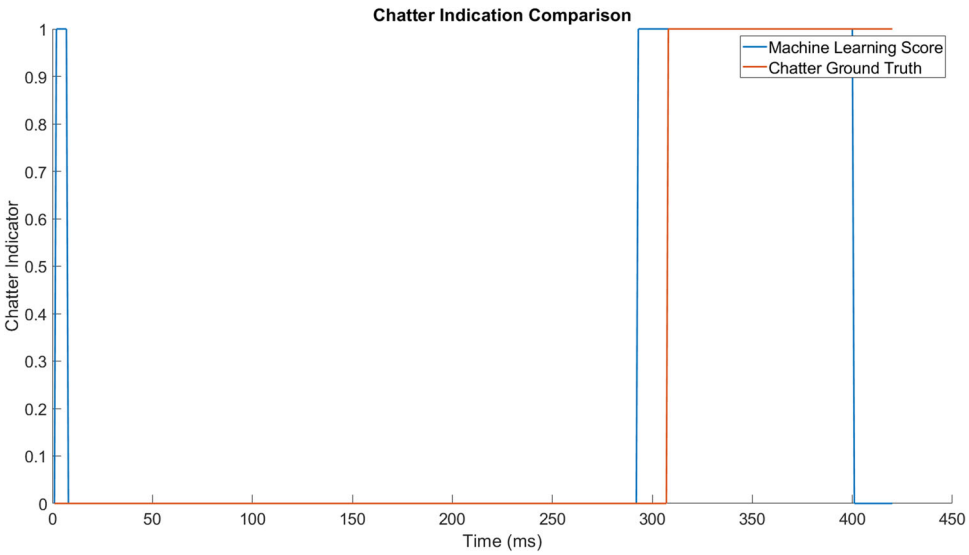

Fig. 13 Chatter label for Roland MDX-40A

Table 2 Confusion matrix for anomaly detection without transfer learning

Confusion Matrix	Anomaly (predicted)	No Anomaly (predicted)
Anomaly (actual)	194	323
Not Anomaly (actual)	142	1031

system identifies the chatter conditions slightly pre-maturely compared to the ground truth, but is still capable of capturing the anomaly.

Using the approach described above, we established indices to perform a quantitative evaluation of the performance of the method. Namely, the confusion matrix for true positive, false negative, false positive and true negative was determined, along with the accuracy, precision and recall. A baseline for the system was developed by obtaining these measures for a system directly trained on the data, without the use of transfer learning. This system had an accuracy of 72.49%, a precision of 57.74% and a recall of 37.52%. The details for this system can be seen in the confusion matrix in Table 2.

The low accuracy of this method is attributed to the high number of false negatives, namely situations where the machine is in a chatter condition, but the system is unable to detect it. This is a highly disadvantageous situation, as it indicates scenarios where the anomaly detection system is not operating as intended.

Using the same testing conditions, the performance metrics for the transferred system were established. The transferred system had an accuracy of 69.86%, recall of 74.85 % and an accuracy of 82.43 %, with the details of the metrics seem in the confusion matrix in Table 3.

These results were deemed acceptable, as they represented comparable results, found in such works as [36], that indicated such accuracy rates were acceptable amongst transferred systems. It is also worth noting that although the accuracy and precision are somewhat low compared to other anomaly detection systems, this is largely due to the high number of false positives, which indicate a somewhat overzealous system. This rate of false positives could likely be lowered with additional tuning of thresholds.

From these experimental results, we are able to see the impact that the transfer learning approach has on implementing the anomaly detection algorithms for the three axis CNC machines. These experiments show the clear potential of this approach and the feasibility for more complex algorithms to be implemented.

5 Discussion

In this paper we have shown that reconstruction error can be used to detect anomalies on a CNC machine. In addition, a network trained for anomaly detection on one system can be re-trained via transfer learning on another system. From our experiments we began with a data set that is over 200 000 data points and trained an anomaly detection system, and then implemented transfer learning with data sets of approximately 5000 data points. These secondary sets of data were collected quickly within a small number of short test cuts, and were able to detect all of the anomaly conditions presented during testing. This approach was applied for the desired material on the different machines, and completed with minimal modifications to the system required. We see that this approach can be quickly applied on two separate machines, with different materials used during our cutting experiments. These experiments show that this approach has the potential to be scaled quickly to individual users.

Although machine learning approaches have been shown to be effective, a common flaw in approaches implemented by other authors is the inability for the algorithm to directly be applied to other systems. We see this in works such as [27] where the authors are able to obtain very promising results on their system, but are unable to obtain comparable results when the networks are implemented on another machine. By showing that transfer learning can be applied for this anomaly detection approach, we can justify the larger initial investment of time and resources to a base system. This is of particular interest for machine manufacturers who have entire families of CNC machines. This approach shows that a large initial investment can be made to develop a sufficiently rich data set, and then apply this data to different, but similar machines. Moreover, the final calibration processes can be left to the end user, as the required training data for transfer learning is very limited in comparison to the initial data sets.

Although it was not attempted, this approach should also be scalable to machines with additional sensors as well. This indicates that more complex anomaly detection algorithms can be implemented with this approach. In addition,

Table 3 Confusion matrix for anomaly detection using transfer learning

Confusion Matrix	Anomaly (predicted)	No Anomaly (predicted)
Anomaly (actual)	387	130
Not Anomaly (actual)	167	1006

this approach is unlikely to be limited to three axis CNC machines. The sensors selected align well with the three axis CNC, as all of the machine and material dynamics required can be captured, but this should also be feasible with other machines and other sensors. For this implementation we chose to use the same type of sensors and a CNC machine with the same degrees of freedom. However, it is not clear how this could be expanded to other systems. This limitation requires future research.

Lastly, as seen in Fig. 12, in some conditions, there is a noticeable trend in reconstruction error towards the anomaly threshold. If the specific situations where these conditions exist can be identified, then it should be possible to apply corrective action (e.g. change in spindle speed) before the anomaly conditions occur. This potential requires future research to directly identify these situations.

6 Conclusion

Anomaly detection is an important step towards making a fully autonomous CNC machine. This technology can lead towards the idea of "lights out" manufacturing, where entire manufacturing facilities can be autonomous. Driven by this motivation, there have been many different techniques applied over the years for anomaly detection.

Recently, the LSTM network has proven to be a useful option, able to overcome many of the short-comings of other approaches. However, a major limitation with LSTM networks are that they require a significant amount of data to properly train a system. To mitigate this short coming, we investigated the potential of using transfer learning to reduce the burden on the end user to collect training data.

This was accomplished by taking a rich data set trained on stable cutting conditions and then applying an anomaly detection algorithm based on finding a three sigma deviation from the reconstruction error between the input and output of the network. We then take this network and obtain a small amount of training data on the target machine for the specific material of interest. Using this data we are able to apply transfer learning such that the target machine is able to implement the previously designed anomaly detection algorithm.

We implemented this on three different materials on two different CNC machines with similar sensors and similar number of machining axis. The results show that applying transfer learning allows us to transport algorithms designed on one machine to another with a limited amount of new training data required. This approach significantly reduces the complication for the end user and allows for complex algorithms designed on one machine to quickly be implemented on another. A use case for this could be for a CNC manufacturer to do significant testing on one type of machine within a family of machines and then design and train a neural network. This could then be given as the base to their users for any machine within the family of machines, with the users completing a calibration procedure and then having a network tuned for their specific system. We believe that these methods developed in the lab could be efficiently transferred to a production setting, but require further testing to validate this approach.

This research shows the power and potential of applying transfer learning to creating a smart CNC machine. As we continue to advance Industry 4.0 technologies, we will develop more intelligent and more connected manufacturing environments. By leveraging these advances, we will be able to not only create a smart CNC machine, but a smart factory and truly achieve the goal of lights-out manufacturing.

Acknowledgements We would like to thank the co-op students, Eric Jessee, Yang Li, and Sarah Ahmed, who helped with the fabrication and data collection for this project. Your assistance was greatly appreciated.

Author Contributions All authors contributed to the study conception and design. Material preparation and data collection were performed by Eric Jessee, Sarah Ahmed, Yang Li, and Eugene Li. The data analysis and the first draft of the manuscript were written by Eugene Li, and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Declarations

Conflict of interest The authors declare no competing interests.

References

1. Altintas Y (2012) Manufacturing automation. Cambridge University Press. https://doi.org/10.1017/CBO9780511843723
2. Ay M, Schwenzer M, Abell D, Bergs T (2021) Recurrent online and transfer learning of a CNC-machining center with support vector machines. In: IEEE International Symposium on Industrial Electronics, Institute of Electrical and Electronics Engineers Inc., vol 2021-June. https://doi.org/10.1109/ISIE45552.2021.9576328
3. Bozinovski S (2020) Reminder of the first paper on transfer learning in neural networks, 1976. Informatica 44:291–302. https://doi.org/10.31449/inf.v44i3.2828
4. Burkov A (2019) The hundred-page machine learning book. Andriy Burkov
5. Deebak BD, Al-Turjman F (2021) Digital-twin assisted: fault diagnosis using deep transfer learning for machining tool condition. Int J Intell Syst. https://doi.org/10.1002/int.22493
6. Erdoğan G (2019) Land selection criteria for lights out factory districts during the industry 4.0 process. J Urban Manag 8(3):377–385. https://doi.org/10.1016/J.JUM.2019.01.001
7. Gers FA, Schmidhuber J, Cummins F (2000) Learning to forget: continual prediction with LSTM. Neural Comput 12(10):2451–2471. https://doi.org/10.1162/089976600300015015, https://direct.mit.edu/neco/article/12/10/2451-2471/6415
8. Gretton A, Borgwardt KM, Rasch M, Schölkopf B, Smola AJ (2008) A Kernel method for the two-sample-problem. Neural Inf Process Syst. www.kyb.mpg.de/bs/people/arthur/mmd.htm. Accessed 8 May 2023

9. Hao G, Kunpeng Z (2020) Pyramid LSTM auto-encoder for tool wear monitoring. In: 2020 IEEE 16th international conference on automation science and engineering (CASE). IEEE, Online Zoom Meeting
10. Hochreiter S, Urgen Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780. http://direct.mit.edu/neco/article-pdf/9/8/1735/813796/neco.1997.9.8.1735.pdf. Accessed 30 Sept 2022
11. IBM (2020) What are recurrent neural networks? https://www.ibm.com/cloud/learn/recurrent-neural-networks. Accessed 30 Sept 2022
12. Inventables (2022) X-Carve Pro. https://www.inventables.com/presales/tech-specs. Accessed 20 Oct 2022
13. Khirirat S, Feyzmahdavian HR, Johansson M (2017) Mini-batch gradient descent faster convergence under data sparsity. In: 2017 IEEE 56th Annual Conference on Decision and Control (CDC), IEEE, Melbourne, Australia. https://ieeexplore.ieee.org/document/8264077. Accessed 20 Oct 2022
14. Kim YM, Shin SJ, Cho HW (2022) Predictive modeling for machining power based on multi-source transfer learning in metal cutting. Int J Precis Eng Manuf - Green Technol 9(1):107–125. https://doi.org/10.1007/s40684-021-00327-6
15. Kounta CAKA, Arnaud L, Kamsu-Foguem B, Tangara F (2022). Review of AI-based methods for chatter detection in machining based on bibliometric analysis. https://doi.org/10.1007/s00170-022-10059-9
16. Kuljanic E, Sortino M, Totis G (2008) Multisensor approaches for chatter detection in milling. J Sound Vib 312(4–5):672–693. https://doi.org/10.1016/j.jsv.2007.11.006
17. Kuljanic E, Totis G, Sortino M (2009) Development of an intelligent multisensor chatter detection system in milling. Mech Syst Signal Process 23(5):1704–1718. https://doi.org/10.1016/j.ymssp.2009.01.003
18. Kuo WF, Huang BM, Lee CH (2020) Development of virtual milling system using data fusion and transfer learning. In: Proceedings - 2020 International Conference on Pervasive Artificial Intelligence, ICPAI 2020, Institute of Electrical and Electronics Engineers Inc., pp 253–257. https://doi.org/10.1109/ICPAI51961.2020.00054
19. Kvinevskiy I, Bedi S, Mann S (2020) Detecting machine chatter using audio data and machine learning. Int J Adv Manuf Technol 108(11–12):3707–3716. https://doi.org/10.1007/s00170-020-05571-9
20. Li WD, Liang YC (2020) Deep transfer learning based diagnosis for machining process lifecycle. Procedia CIRP 90:642–647. https://doi.org/10.1016/J.PROCIR.2020.02.048
21. Li J, Lu J, Chen C, Ma J, Liao X (2021) Tool wear state prediction based on feature-based transfer learning. Int J Adv Manuf Technol. https://doi.org/10.1007/s00170-021-06780-6/Published, https://doi.org/10.1007/s00170-021-06780-6
22. Lindemann B, Maschler B, Sahlab N, Weyrich M (2021). A survey on anomaly detection for technical systems using LSTM networks. https://doi.org/10.1016/j.compind.2021.103498
23. Malhotra P, Ramakrishnan A, Anand G, Vig L, Agarwal P, Shroff G (2016a) LSTM-based encoder-decoder for multi-sensor anomaly detection. 2016 Anomaly Detection Workshop. http://arxiv.org/abs/1607.00148
24. Malhotra P, Ramakrishnan A, Anand G, Vig L, Agarwal P, Shroff G (2016b) LSTM-based encoder-decoder for multi-sensor anomaly detection. In: 2016 Anomaly Detection Workshop. https://doi.org/10.48550/arxiv.1607.00148, http://arxiv.org/abs/1607.00148v2
25. Park D, Hoshi Y, Kemp CC (2018) A multimodal anomaly detector for robot-assisted feeding using an LSTM-based variational autoencoder. IEEE Robot Autom Lett 3(3):1544–1551. https://doi.org/10.1109/LRA.2018.2801475
26. Postel M, Bugdayci B, Wegener K (2020) Ensemble transfer learning for refining stability predictions in milling using experimental stability states. Int J Adv Manuf Technol 107(9–10):4123–4139. https://doi.org/10.1007/s00170-020-05322-w
27. Rahimi MH, Huynh HN, Altintas Y (2021) On-line chatter detection in milling with hybrid machine learning and physics-based model. CIRP J Manuf Sci Technol 35:25–40. https://doi.org/10.1016/j.cirpj.2021.05.006
28. Rashid KM, Louis J (2019) Times-series data augmentation and deep learning for construction equipment activity recognition. Adv Eng Inform 42. https://doi.org/10.1016/J.AEI.2019.100944
29. 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. https://doi.org/10.1007/s00170-020-05449-w/Published, https://doi.org/10.1007/s00170-020-05449-w
30. Shi B, Attia H (2010). Current status and future direction in the numerical modeling and simulation of machining processes: a critical literature review. https://doi.org/10.1080/10910344.2010.503455
31. Siegel B (2020) Industrial anomaly detection: a comparison of unsupervised neural network architectures. IEEE Sensors Lett 4(8). https://doi.org/10.1109/LSENS.2020.3007880
32. Smagulova K, James AP (2019) A survey on LSTM memristive neural network architectures and applications. Eur Phys J Special Topics 228:2313–2324. https://doi.org/10.1140/epjst/e2019-900046-x
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 Inform 15(4):2416–2425. https://doi.org/10.1109/TII.2018.2881543
34. Tan C, Sun F, Kong T, Zhang W, Yang C, Liu C (2018) A survey on deep transfer learning. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 11141 LNCS:270–279. https://doi.org/10.1007/978-3-030-01424-7_27/FIGURES/5, https://link.springer.com/chapter/10.1007/978-3-030-01424-7_27. Accessed 15 Oct 2022
35. Unver HO, Sener B (2022) Exploring the potential of transfer learning for chatter detection. In: Procedia Computer Science, Elsevier B.V., vol 200, pp 151–159. https://doi.org/10.1016/j.procs.2022.01.214
36. Wang J, Zou B, Liu M, Li Y, Ding H, Xue K (2021) Milling force prediction model based on transfer learning and neural network. J Intell Manuf 32(4):947–956. https://doi.org/10.1007/S10845-020-01595-W/FIGURES/6, https://link.springer.com/article/10.1007/s10845-020-01595-w
37. Yang HC (2020) Roughness of milling process. https://doi.org/10.21227/rx49-xs81, https://ieee-dataport.org/open-access/roughness-milling-process. Accessed 15 Aug 2022