International Journal of Current Engineering and Technology
©2020 INPRESSCO®, All Rights Reserved

E-ISSN 2277 – 4106, P-ISSN 2347 – 5161
Available at http://inpressco.com/category/ijcet

Research Article

Applying Machine Learning Algorithm for a Milling Process Simulator for Process Modelling and Optimization

Eng. Nawaf Mohammad H Alamri[*], Dr. Michael Packianather and Eng. Theocharis Alexopoulos

[†]Mechanics, Materials and Advanced Manufacturing, Cardiff University, Queen's Buildings, 14-17 The Parade, Cardiff, CF24 3AA, United Kingdom

Received 08 Dec 2019, Accepted 10 Feb 2020, Available online 12 Feb 2020, **Vol.10, No.1 (Jan/Feb 2020)**

Abstract

Manufacturing industry is currently the heart of data driven revolution which is the cornerstone in transforming the traditional manufacturing systems to highly automated smart manufacturing by embedding new technologies such as Internet of Things (IoT), cyber-physical systems and cloud computing in physical advanced manufacturing processes to measure and monitor real time data. It is necessary to manage data and apply big data analytics to extract meaningful pattern. Also, it is helpful to adopt artificial intelligence techniques in manufacturing context to increase process efficiency within the framework of industry 4.0. The aim of this paper is implementing neural network algorithm for a milling process simulator in order to model and optimize the process. MATLAB was used to build the model based on input and target milling machine data which are material removal rate and spindle load respectively, the objective of the model was to predict spindle load. Then, the network has been tuned to improve the results, the first model achieved a good performance with overall mean squared error value of 0.948 while the second one had bed better performance of 0.879. The tuned model had lower average error value of 0.580 compared to 0.606 in the original model.

Keywords: *Smart Manufacturing, Milling Process, Material Removal Rate; Spindle Load, Artificial Intelligence, Artificial Neural Network.*

1. Introduction

Nowadays there is an increasing trend in the availability of manufacturing data which arise new opportunities for improving the accuracy of the process simulation. (O'Donovan *et al*., 2015) stated that currently manufacturing industry is the heart of data driven revolution which is the cornerstone in transforming the traditional manufacturing systems to highly automated smart manufacturing. The focus of the advanced manufacturing is creating an intelligent manufacturing from real time data that support the accuracy of the decision making. The transformation needs emerging new technologies such as Internet of Things (IoT), cyber-physical systems and cloud computing and embedding them in physical advanced manufacturing processes to measure and monitor real time data. It is necessary to manage the exponential increase in data production and apply big data analytics in order to extract meaningful pattern that helps in decision making. In addition, (De Filippis *et al*., 2017) discussed the uncertainty in the situation of manufacturing processes, they said that it became more sophisticated as there is continuous variability in the processes resulting from the fluctuating in the demand of the customers and the life cycle of the products. Furthermore, they stated that the use of artificial intelligence techniques in manufacturing context allowed to yield revolutionary improvement within the framework of industry 4.0 which is beneficial to increase the efficiency and effectiveness of the processes. One of the most important techniques is artificial neural network (ANN) which can be used to address continuous variation problem along with monitoring, controlling and optimizing the processes and making prediction about their parameters in order to build accurate process simulation.

The aim of this paper is to implement neural network algorithm for a milling process simulator in order to model and optimize the process. The model will be built based on input and target machine data and use this model to learn from the available data and generalize to unseen data in order to extract meaningful features, pattern and information which helps to understand and simulate the machine parameters in addition to support decision making process. Also, it has the ability to deal with large amount of data and detect complex nonlinear relationship between the dependent and independent

[*]Corresponding author's ORCID ID: 0000-0002-5641-0178
DOI: https://doi.org/10.14741/ijcet/v.10.1.11

variables that make it more accurate and give it the edge over other prediction methods.

2. Literature Review

This review will present a detailed explanation about milling process definition, phases, parameters and operations. In addition, practical applications of artificial neural network in manufacturing context will be discussed showing the functionalities of neural network.

2.1 Milling Process

(LearnMech) explained milling process in terms of definition, phases and parameters controlling the machine. It is a machining process in which material from work piece is removed by rotating cutters that consists of multiple cutting edges called teeth. It is applied to all objects from small to large ones and it very common manufacturing process in the industry. The main feature that distinguish this process from any other process is the orientation between the fed direction and tool axis. The milling operation has three different geometric forms, plane surfaces which linear in all three dimensions so it is the simplest and the most suitable form. The second one is two dimensional surfaces where the change of the surface shape happen in two axes and it is linear in the third axis. The last form is the three-dimensional surfaces where the change happens in all three directions.

2.1.1 Phases

The milling process consists of three processes or cutting phases, the first one is milling cutter which can be named end mills as well and they have on their end surfaces special surfaces for cutting can be position in the work piece by drilling. In addition, they have side cutting surfaces on both sides which can be used for peripheral milling. The second phase is surface finish in which any material in the cutting area gets interval regularly as the side cutters that have ridges on them and the distance between them depends on the feed rate, cutter diameter and cutting surfaces quantity. The last process is gang milling when more than two cutters perform uniform or distinct operation in order to make the setup as done in horizontal milling.

2.1.2 Parameters

There are three major parameters that control the milling machine which are cutting speed, feed rate and depth of cut. The speed of the cut is about how fast is the cutting and it is expressed in meters per minute. It depends on cutter diameter of the milling and cutter speed in revolution per minute, spindle speed is selected to give the desired cutter speed in milling machine. The second parameter is feed rate which is the rate with the workpiece under process advances under revolving cutter and can expressed with three different ways, feed per tooth representing the distance between engagement between two successive teeth, feed per revolution which is the travel during one revolution of milling cutter and the third way is the feed per unit of time like feed per minute or feed per second. The last control variable is depth of cut representing the material thickness removed when cutter complete the milling operation from end to end. It is the orthogonal distance and the original and final surface of the workpiece.

2.1.3 Operations

(N. Azudin, n.d) described five general operations involved in the milling process starting with the storing in which the workpiece arrives in the mill and stored according to its quality determined by hardness. Then, cleaning takes place to remove coarse impurities and make the separation according size, shape and weight of the separated material. After that conditioning begins to produce a uniform moisture content that helps to prevent breakup during milling and improve material separation. After conditioning, blending of different batches is done to make a mix has the ability to produce the required quality. Finally, the milling takes place to produce the finished product. The sequence of the described operation can be represented clearly in the following diagram:

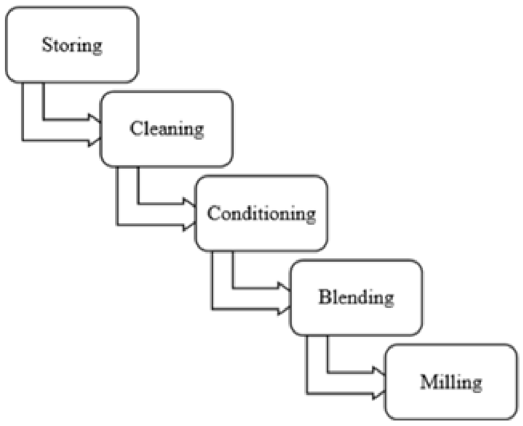


Fig.1 Sequence of milling operations (N. Azudin, n.d)

2.2 Applications of Neural Network in Manufacturing

(De Filippis *et al*., 2017) stated that neural network is an efficient, effective and accurate tool for building a simulator for manufacturing processes to optimize their parameters and predict the properties of the processed product based on the optimum control variables which is beneficial to save cost, time and material resources. The classification for the functionalities of neural network in manufacturing context are illustrated in the shown figure:

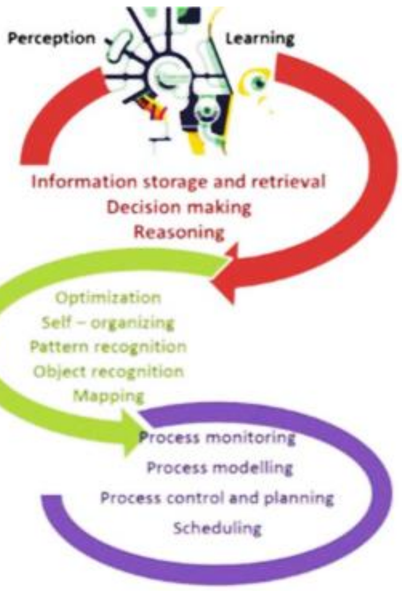


Fig.2 The functionalities of neural network in manufacturing (De Filippis et al., 2017)

The manufacturing applications include modelling, parameters prediction, monitoring and control, in addition to scheduling the processes which solve issues related to operational decision making. The focus will be on the use of artificial neural network to monitor, control and optimize welding processes. The procedure for this application is similar to the steps shown in the previous section by collecting experimental observations and preprocessed it to be ready for network training. Then, establish numerical relationship between the parameters and mechanical features of the part after welding. Finally, time and costs parameters of the process will be evaluated to identify the benefits and costs incurred from the prediction model. However, there are are some important applications to particular production process such as injection moulding process which is a dynamic process as the control variables are the cylinder velocity, the holding, temperatures of melting, the pressure that help to produce the flow of polymer. The process is uncertain and complex making it difficult to correlate between these variables and product quality such as smoothness and geometry accuracy. To solve this issue, multilayer perceptron was used to model the process in order to predict the part quality. Furthermore, arc welding process is another application and the parameters of this process are the temperature of surface, the voltage of welding and the torch speed. Again, it is difficult to correlate between these variables and product quality such as the geometry and weld defects. Also, multilayer perceptron was used for process modelling and product quality prediction.

2.3 Implementation of Neural Network

(Dr. Packianather, 2018) stated that the first step to design a neural network application is data pre-processing including replacing missing data with a nominal value and removing outliers and erroneous data to transform the data to a form suited to the neural network input and to minimize the data so that computations become in short time. After that, partitioning the data takes place by dividing the data into three sets which are: training set that contain enough data with suitable distribution to give accurate answers on data that were not included in training process, test set is selected randomly and used to monitor the performance of the network which helps to decide when to stop training, validation s*et al*so is selected randomly and used to see the generalization capability of the trained network. The optimal iteration will be the one which has the lowest error in test and validation sets. After dividing the data, it is time to select the best neural network architecture based on input type, learning type and data type as will be applied in section 5. Then, training start which consists of presenting the neural network with example data and adjust the internal weights until the desired response achieved. once training is done, the weights are fixed to specific values and the network model can be used to predict the solution for new data.

Furthermore, (Najafabadi *et al*., 2015) discussed deep learning in data mining as the main concept in deep learning algorithms is automated extraction of representation from a huge amount of dataset with a large motivation of artificial intelligence field. The extracted features from the set of data are distributed allowing for a large number of possible configurations which leads to better generalization. The relation between the number of possible configuration and the number of extracted features is exponential, generation of the observed data was based on the interaction of different factors so that obtaining a pattern through some configurations can help to obtain additional patterns of unseen data through new configurations.

3. Methodology

After collecting milling process data, it is necessary to clean and pre-process the dataset using MATLAB software. It has the ability to deal with missing records by removing or replacing them, in addition to detecting the outliers is another key task in cleaning the data, removing them is another feature for using this software. After that, the most challenging step is to identify a suitable neural network technique for building a simulator of milling process based on a predefined dataset containing measurements from key process parameters. Then, coming to modelling stage and starting the MATLAB neural network toolbox phase by creating a script for reading dataset and building the neural network model based on input and target milling machine data which are material removal rate and spindle load respectively, the objective of the model is to predict the spindle load. The code will perform training, validating and testing the network in order to model and optimize the process in addition to build a simulator for predicting the spindle load of the milling machine. The main advantage of creating a new script rather than using the main graphical user interface of MATLAB is having high level of customization of changing the values of the parameters in the training function. After building the model, the results will be interpreted and analyzed, the neural network model will be tuned for further improvement of the prediction. Then, material removal rate in other similar testing data will be considered to get new prediction results for the spindle load using the previous tuned network. The predicted values resulted from original and testing material removal rate will be compared in order to verify the prediction results.

4. Data Preprocessing

After collecting data, it is essentially to be pre-processed and cleaned so that it is ready to be used for going further to the next step which is modelling. First, each attribute will be described and explored, then it will be prepared to be ready for building neural network model.

4.1 Data Understanding

After understanding how a milling machine functions, a more in-depth analysis of the data was carried out. It is important to describe and explore the set of data in order to enhance the understanding about the parameters that control the process of the milling machine.

Looking at the data, it consists of 6653 instances and 11 attributes, the input parameter is material removal rate (MRR) which is defined by (Sandvik.Coromant) as the volume of metal removed per minute, whereas the target variable that will be predicted using neural network model is spindle load (SL) which is used to mount the milling cutters to perform the cutting features as stated in (weissgmbh). Spindle load was measured by direct sensor readings while material removal rate was identified considering the direct sensor readings of spindle speed.

4.2 Data Preparation

After describing, exploring and understanding the attributes included in the set of data, it is important to clean the data by figuring out the missing records and removing the outliers if any in order to prepare the dataset to be ready for modelling stage. Considering the missing values, there are no missing records in the dataset that need to be removed or replaced.

Another key task in cleaning the data is detecting and removing the outliers if any. Before doing so, it is helpful to get the statistical values of minimum, maximum, range, mean and standard deviation for spindle load attribute in order to get an overview about its variability. The minimum value is 0 while the maximum is 26, so the range is 26 which is the difference between the maximum and the minimum. The mean of the data is 3.880605 and the standard deviation is 1.853422. The statistical values reveal that there is a considerable difference between the maximum and average values, which indicates that outliers may exist within the set of data.

In order to remove the outlier values, 'rmoutliers' function in MATLAB software has been used with its default algorithm that considers any value more than three scaled median absolute deviations as an outlier value. However, there are other algorithms that can be used such as defining the outlier as any value more than three standard deviation from the mean or any point outside specific percentile. After removing the existing outliers, the set of data reduced to be 6650, only 3 records have been removed with zeros values in material removal rate attribute and 13, 17 and 26 values in spindle load variable. After pre-processing, the remaining records that will be used for building the neural network model is 6650 observations.

5. Model Building, Results Analysis and Discussion

Once the data has been preprocessed, missing values and outliers have been investigated. Thus, at this stage of implementation, building a neural network model based on the remaining data is required in order to predict the spindle load. Before start making the model, it is necessary to identify which model should be used in order to train, validate and test the network. The decision is based on data type, learning type and the objective of building the model, the set of data contain continuous valued input and learning from data is in input output patterns, so the learning type is supervised learning. The records are arranged in a tabular format where each row represents one observation of spindle load and material removal rate, so it is tabular data. The objective of the neural network is to predict the spindle load. Having it is a prediction problem with tabular continuous data, so the best fit model is multilayer feedforward network with backpropagation algorithm. The network will be built and the algorithm will be applied and tuned in order to improve the results.

5.1 Building the Network Model

The script will be created for reading the dataset, defining input and target variables as arrays and making them as matrix row because each observation is written in a row. Then, selecting the training function which is 'trainscg' that update bias and weight values according to scaled conjugate gradient approach which has been used for prediction problem in one of the examples in (MathWorks) website. However, the training occurred in this function according to specific training parameters, (MathWorks) stated their default values as shown in the following:

- Maximum training epochs: 1000 (net.trainParam.epochs)
- Number of epochs between displays (NaN if there is no displays): 25 (net.trainParam.show)
- Generating command-line output: false (net.trainParam.showCommandLine)
- Show GUI for training: true (net.trainParam.showWindow)
- Performance goal: 0 (net.trainParam.goal)
- Maximum training time in seconds: inf (net.trainParam.time)
- Minimum performance gradient: 1e-6 (net.trainParam.min_grad)
- Maximum validation failures: 6 (net.trainParam.max_fail)
- Weight change for approximation of the second derivative: 5.0e-5 (net.trainParam.sigma)
- Parameter to regulate the indefiniteness of the Hessian: 5.0e-7 (net.trainParam.lambda)

The website also stated that this training function can be used to train any network that has input, weight and transfer functions as derivative functions. The derivatives of performance can be calculated using backpropagation with respect to bias and weight variables. However, this algorithm has a limitation that it does not have the ability to perform a line search at each iteration. The training stops if any of the following conditions occurs:

- Reaching the maximum number of epochs which is 1000
- Exceeding the maximum amount of time
- Minimizing the performance to the goal
- Having performance gradient that falls below the minimum gradient
- Exceeding the maximum fail times for validation performance since the last time it decreased if the validation is used

After that, creating the fitting network with two-layer feedforward network, sigmoid activation function and 10 hidden neurons in 1 hidden layer. Then, setting the maximum validation failure to 50 instead of the default value of 6 to avoid stopping the training earlier without reaching enough number of iterations, this level of customization is the main advantage of creating a new script rather than using the main graphical user interface in MATLAB. After that, dividing the dataset randomly to 80% for training, 10% for validation and 10% for testing. Then, training, validating, testing and viewing the network in addition to making plots about the model which will be explained later in analyzing the results. However, the script is shown in the following:

Building Neural Network Script

%Define input and target variables as arrays and make their samples as matrix row
A = csvread('MRR.csv'); **(defining the input variable)**
B = csvread('SL.csv'); **(defining the target variable)**
x = transpose(A); **(making input samples as matrix row)**
t = transpose(B); **(making target samples as matrix row)**

%Select a training function
trainFcn = 'trainscg'; **(scaled conjugate gradient function)**

%Create a fitting network
hiddenLayerSize = 10; **(10 hidden neurons in 1 hidden layer)**
net = fitnet(hiddenLayerSize,trainFcn); **(fitting the network)**
net.trainParam.max_fail = 50; **(setting maximum validation failure to 50)**

%Setup division of data for training, validation and testing
net.divideParam.trainRatio = 80/100; **(selecting 80% randomly for training)**
net.divideParam.valRatio = 10/100; **(selecting 10% randomly for validation)**
net.divideParam.testRatio = 10/100; **(selecting 10% randomly for testing)**

%Train the network
[net,tr] = train(net,x,t); **(training the network)**

%Test the network
y = net(x); **(producing the predicted value)**
e = gsubtract(t,y); **(calculating the error, the difference between target and output)**
performance = perform(net,t,y); **(calculating the overall mean squared error)**

%View the network
view(net) **(showing the network structure)**

% Make plots
figure, plotperform(tr); **(plotting mean squared error)**
figure, plottrainstate(tr); **(plotting training state)**
figure, ploterrhist(e); **(plotting error histogram)**
figure, plotregression(t,y); **(plotting regression)**
figure, plotfit(net,x,t); **(plotting function fitting)**

After running the code, the following neural network model that consists of one input (material removal rate), 10 hidden neurons, one output layer and one output (spindle load) was shown:

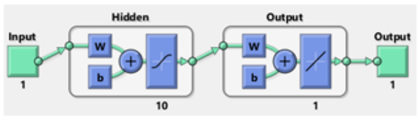


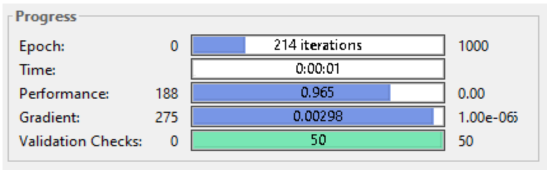
Fig.3 Neural network model

Once the model has been built, it is the stage for presenting, analyzing and interpreting the results of the model in order to achieve the objective from building it which is predicting the spindle load. The neural network has been trained 5 times in order to get better visualization about the network performance in terms of overall mean squared error (mse). However, the following table shows the number of iterations, network performance and best validation performance for each running time:

Table 1 Results of neural network training for 5 times

#	Number of Iterations	Network Performance	Best Validation Performance
1	90	0.9549	0.96878 (epoch 40)
2	164	0.9489	0.96357 (epoch 114)
3	187	0.9500	0.95656 (epoch 137)
4	214	0.9483	0.82105 (epoch 164)
5	127	0.9539	0.98546 (epoch 77)
Average		0.9512	0.93908

In the best performance, the model was running for 214 iterations, it achieved a good performance with overall mean squared error of 0.9483 which is the average squared difference between the target and the output, mse value for training data was 0.965, the gradient value was 0.00298 and the maximum validation failure for the model was 50 as stated previously. The following figure shows the progress results of the model:



```
Progress
Epoch:              0    214 iterations    1000
Time:                    0:00:01
Performance:      188    0.965             0.00
Gradient:         275    0.00298           1.00e-06
Validation Checks:  0    50                50
```

Fig.4 Progress results

The set of data was divided randomly into three divisions, a set for training data which is 80% of the records, another set for validation containing 20% of the observations and the rest 20% for testing as described in the code presented previously. The performance of the model was evaluated based on mean squared error as shown in the following figure:

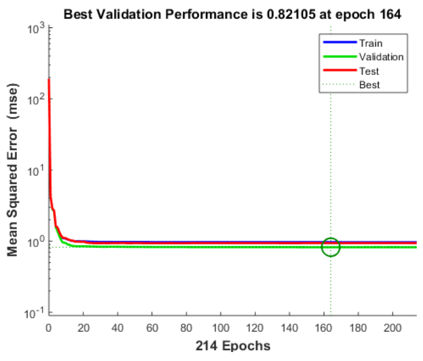


Fig.5 Mean squared error for training, validation and testing

The figure shows that the mean squared error for all three sets was 188 at the beginning, then it declined steeply to reach mse value of 1 in the tenth iteration. After that, the chart experienced almost a steady state condition for all three categories with slightly lower mse value for testing set than training set, the mean squared error for validation set was the minimum and reached the best performance at epoch 164 with mse value of 0.82105. Overall, the model has good capability to generalize to unseen data since the mean squared error is low. Regarding the gradient values and validation checks figure, it is shown in the following:

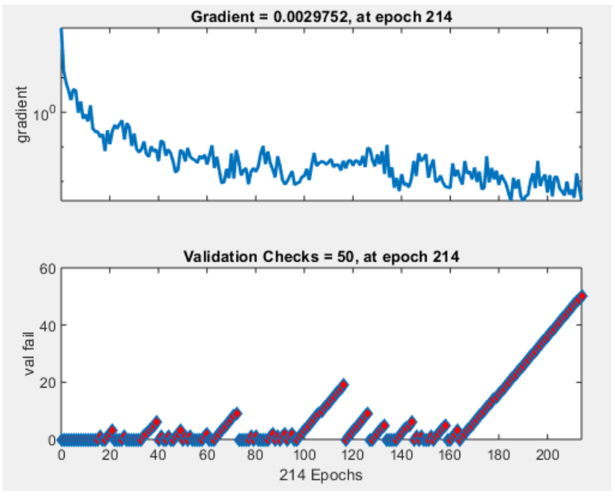


Fig.6 Gradient and validation checks

The first plot shows the training state values and the gradient descent value was fluctuating and decreasing gradually, the last gradient value was 0.00297 at epoch 214. The second plot is about validation checks which is the number of successive iterations that the validation performance fails to decrease, the value was 50 at the last epoch as stated in the code previously (net.trainParam.max_fail = 50) to allow for higher number of iterations as the training will stop when its maximum validation failure exceeds the value of 50.

After evaluating the performance of all three categories, it is important to evaluate the following regression models graphs for training, validation and testing data which shows the correlation between the dependent and independent variables. In figure (7), R value for training set shows that the percent of relationship between the output (predicted SL) and the target (actual SL) was calculated to be 90.54%, it was 92.25% for validation group which is the best percentage among the three sets, R value for testing regression model was 90.48%. Overall, there is a good relationship between the output and the target since the percent of correlation was 90.72%, although the distribution of the data points is not around the fit line because of the randomness in the set of data.

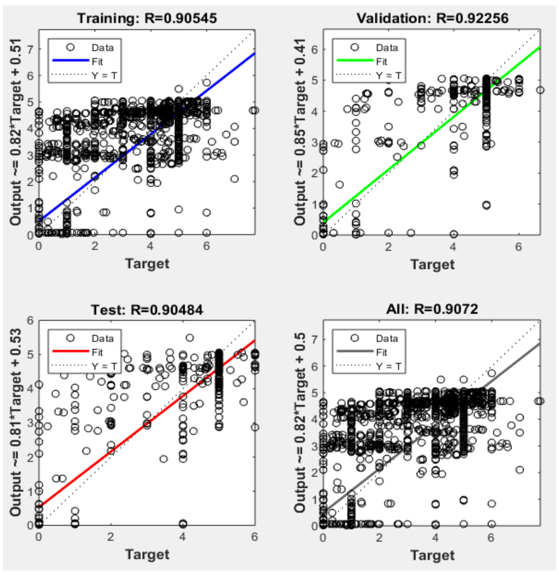


Fig.7 Regression models

The following plot shows the histogram for error which is the difference between the output and the target (target – output):

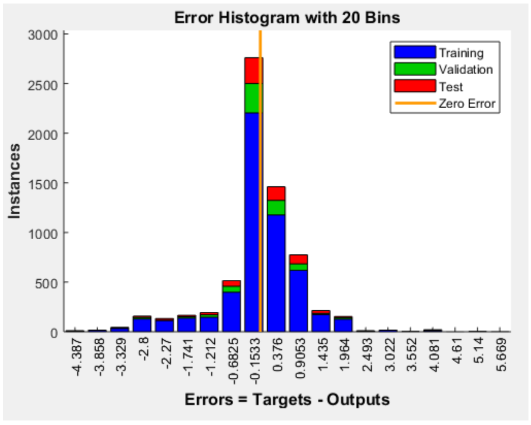


Fig.8 Error histogram

The figure shows that most of the instances have error values close to zero with slightly higher output values since data points are higher in negative side. This conclusion is confirmed by the following figure:

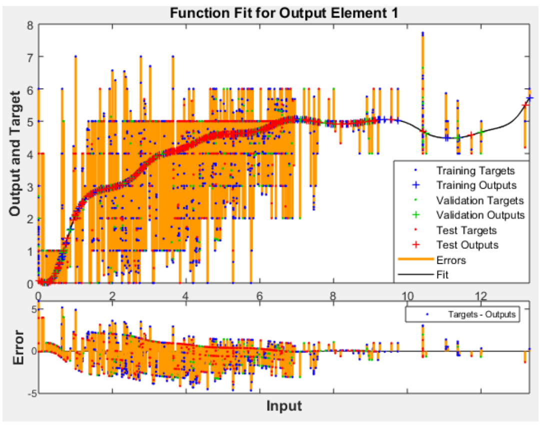


Fig.9 Function fit (input x10000)

The first plot shows the output function across the range of inputs in addition to targets and outputs points associated with inputs values. It is obvious that most of the target and output values are associated with inputs values between 20000 and 60000, the distribution of the data points is not around the fit line because of the randomness in the set of data as described in regression models figure. The error plot at the bottom bar shows the difference between the output and the target where data points are fluctuating around zero line which means that there is no significant difference between the output and the target. For better visualization of the graph, the following sample of data points for input values between 20000 and 40000 will be shown in fit function plot:

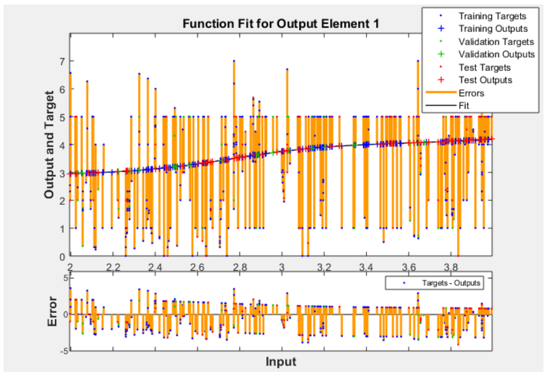


Fig.10 Function fit for a sample (input x10000)

Once the model has been analyzed, it is the stage to look at the predicted values of the spindle load and compare it with the actual value. The following table shows the lowest 20 records in terms of absolute error that express the difference between the actual and predicted value of the spindle load regardless which one is higher.

Table 2 Lowest 20 records in terms of absolute error

MRR	SL	Predicted SL	Absolute Error
74382	5	4.999438	0.000562
74382	5	4.999438	0.000562
66136	5	4.999364	0.000636
66136	5	4.999364	0.000636
66136	5	4.999364	0.000636
66136	5	4.999364	0.000636
66136	5	4.999364	0.000636
66136	5	4.999364	0.000636
66136	5	4.999364	0.000636
89376	5	4.999132	0.000868
89540	5	5.002251	0.002251
89540	5	5.002251	0.002251
89540	5	5.002251	0.002251
89540	5	5.002251	0.002251
55660	4.631579	4.628398	0.003181
89595	5	5.003289	0.003289
89210	5	4.995941	0.004059
89210	5	4.995941	0.004059
89650	5	5.004321	0.004321
66304	5	5.005255	0.005255

The overall average error was calculated to be 0.606 which indicates that there is no significant difference between the actual and predicted spindle load.

5.2 Tuning the Network Model

The network model can be tuned in order to improve the results and have better performance for the neural network model which can be used more accurately to predict the spindle load.

The previous code will be performed similarly in reading the dataset, defining input and target variables as arrays and making them as matrix row, using two-layer feedforward network with sigmoid activation function, setting the maximum validation failure to 50 instead of default value of 6 to allow for higher number of iterations and dividing the data to the same three categories with the same percentages. The difference will be in building the network with 20 hidden neurons instead of 10 in 1 hidden layer and selecting different training function which is 'trainlm' that update bias and weight values according to Levenberg Marquardt optimization approach which can be used for prediction problems as well. It is the most recommended training function in supervised learning as it has the fastest backpropagation algorithm, but it requires more memory than other algorithms.

However, the following table summarizes the comparison between the first model with training function ‘trainscg’ and 10 hidden neurons in 1 hidden layer and the second model with training function ‘trainlm’ and 20 hidden neurons in 1 hidden layer:

Table 3 Comparison between two models

Point of Comparison	Model 1	Model 2
Performance (Overall mse)	0.9483	0.8798
Gradient	0.00298	0.00125
Best validation performance (validation mse)	0.82105	0.76186
R value for training regression model	0.9054	0.9123
R value for validation regression model	0.9225	0.9246
R value for testing regression model	0.9048	0.9190
R value for overall regression model	0.9072	0.9142
Overall average error	0.606	0.580

In addition, the foloiwng graph shows the actual spindle load against the predicted spindle load for a sample of 200 records:

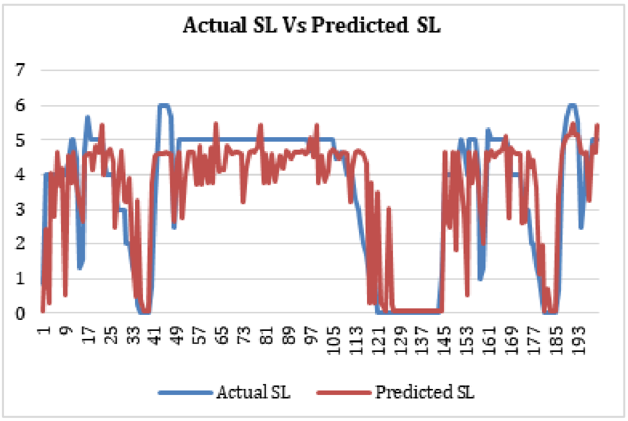


Fig.11 Actual SL Vs Predicted SL

The above graph shows that there is a higher correlation between the actual and predicted values of the spindle load as both charts are closer to each other following similar trend in most of the records with minor difference in some points. Similarly, there are three lags in both charts showing the decreasing trend in some of the successive records.

5.3 Testing the Network Model

The tuned network model that built based on the original data can be tested using other similar set of data produced from the same milling machine in order to verify the prediction results of the spindle load. The network will be recalled again and the values of material removal rate in the testing data will be substituted in model to get the new predicted value and compare it with the values resulted from the model done in section 5.2. The code for recalling the neural network is shown in the following:

Recalling Neural Network Script

```
C = csvread ('Test MRR.csv'); (Reading MRR in testing data)
D = transpose (C); (Making it as matrix row)
F = net (D) (Recalling the network and substituting testing MRR values)
```

It is important to check the predicted values of the spindle load and compare it with the previous prediction using original MRR. The following table shows the lowest 20 records in terms of absolute difference between two predicted values of the spindle load:

Table 4 Lowest 20 records in terms of absolute difference

MRR	SL	Predicted SL (Testing MRR)	Predicted SL (Original MRR)	Absolute Difference
75152	5	4.695782	4.695673979	0.0001082
61160	5	4.678850	4.678716328	0.0001341
59808	5	4.630053	4.630212568	0.0001593
41630	5	4.619537	4.618367371	0.0011693
61544	5	4.698593	4.697378564	0.001214
51198	5	4.585852	4.584554485	0.0012976
51152	5	4.586118	4.584554485	0.0015634
66000	3	5.068284	5.069878095	0.0015945
66000	3	5.068284	5.069878095	0.0015945
47520	5	4.676929	4.678716328	0.0017869
47520	4	4.676929	4.678716328	0.0017869
5324	0	0.359693	0.361520126	0.0018266
5324	0	0.359693	0.361520126	0.0018266
5324	0	0.359693	0.361520126	0.0018266
5324	0	0.359693	0.361520126	0.0018266
5324	0	0.359693	0.361520126	0.0018266
5324	0	0.359693	0.361520126	0.0018266
5324	0	0.359693	0.361520126	0.0018266
5324	0	0.359693	0.361520126	0.0018266
5324	0	0.359693	0.361520126	0.0018266

The overall average difference was calculated to be 1.289 which is the average absolute difference between the predicted values considering the original MRR and testing MRR, it is concluded that the difference between both predicted values is not significant. This conclusion can be confirmed by showing their charts for a sample of 200 records as shown in the following graph:

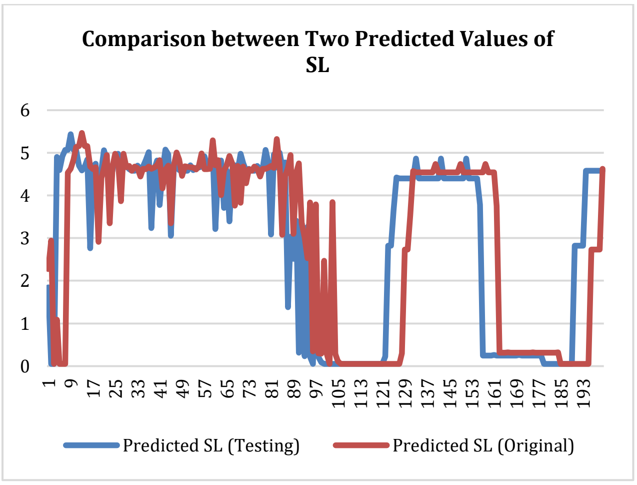


Fig.12 Comparison between two predicted values for SL

The above graph shows that both charts are closer to each other following similar trend in most of the records with minor difference in some points. There is only one lag showing the decreasing trend in some of the successive records, this indicates that there is no significant difference between two prediction values. The following figure shows the predicted values of the spindle load depending on testing material removal rate:

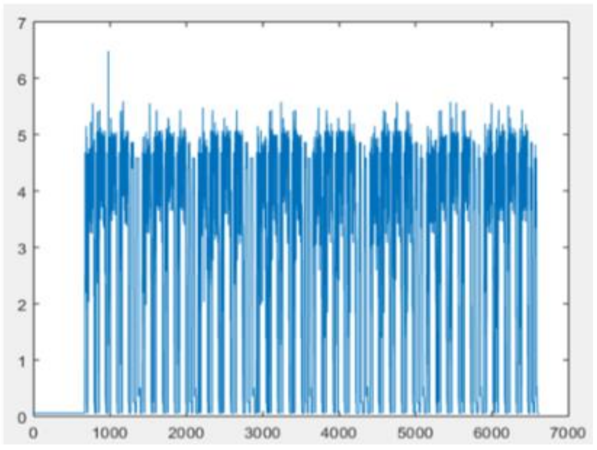


Fig.13 Predicted SL associated with testing MRR

Overall, the output values were slightly higher than the target values which means that the neural network overestimating the predicted values of the spindle load, (tormach) recommended to use spindle load meter that measure and monitor the value of the load and alert in case of having high load value exceeding the maximum possible load set in the device in order to avoid reaching overloading condition that might cause defective parts.

Conclusion

Adopting artificial intelligence techniques in the context of manufacturing is beneficial to increase the efficiency and effectiveness of the processes within the framework of industry 4.0. In this paper, artificial neural network had been applied using MATLAB neural network toolbox to predict spindle load of a milling machine. A script was created to read the dataset and build the network model based on input and target milling machine data which are material removal rate and spindle load respectively. The main contribution of this paper is creating such a custimized script which train, validate and test the data for modelling, optimizing and prediction. It is suitable for tabular data which is the data type used in this paper and can be used for time series data as well. The main advantage of using this script rather than using the main graphical user interface of MATLAB is having high level of customization of changing the values of the parameters in the training function. In addition, a simulator for a milling process was built to predict the values of the spindle load based on material removal rate. In the future, the scope of application could be expanded to include deep learning networks and algorithms to enhance the learning capability with more layers and improve the process of extracting the features automatically.

Acknowledgement

I would like to express my deepest gratitude to my parents and my wife for giving me the composure and the strength to complete this work. I would like to offer my thanks and genuine appreciation to Dr. Michael Packianather and Eng. Theocharis Alexopoulos for guidance, invaluable advice and unlimited support to accomplish this work.

References

O'Donovan, P., Leahy, K., Bruton, K., & O'Sullivan, D. T. (2015). Big data in manufacturing: a systematic mapping study. Journal of Big Data, 2(1), 20.

De Filippis, L. A. C., Serio, L. M., Facchini, F., & Mummolo, G. (2017). ANN Modelling to Optimize Manufacturing Process. In Advanced Applications for Artificial Neural Networks. IntechOpen.

Milling Process – Definition, Milling Manufacturing Processes. [online] Available at https://learnmech.com/milling-process-definition-millingmanufacturing-processes/ [Accessed: 2 July 2019].

N. Azudin (n.d). 3 The Milling Process. [online] Available at https://learnmech.com/milling-process-definition-milling-manufacturingprocesses/ [Accessed: 2 July 2019].

Dr. Packianather (2018), Lecture notes of EN4902/ENT633 course in school of engineering at Cardiff university (Autumn 2018).

Najafabadi, M. M., Villanustre, F., Khoshgoftaar, T. M., Seliya, N., Wald, R., & Muharemagic, E. (2015). Deep learning applications and challenges in big data analytics. Journal of Big Data, 2(1), 1

Milling formulas and definitions. [online] Available at https://www.sandvik.coromant.com/en-gb/knowledge/machiningformulas-definitions/pages/milling.aspx [Accessed: 27 June 2019].

Milling spindles [online] Available at https://www.weissgmbh.com/ en/spindle-abc/term/ milling-spindles-1/ [Accessed: 19 August 2019].

Choose a Multilayer Neural Network Training Function. [online] Available at https://uk.mathworks.com/ help/deeplearning/ug/choose-amultilayer-neural-network-trainingfunction.html?search Highlight= training %20functions&s_tid=doc_srchtitle [Accessed: 16 August 2019].

Trainscg. [online] Available at https://uk.mathworks.com/ help/deeplearning/ref/trainscg.html [Accessed: 16 August 2019].

Trainlm. [online] Available at https://uk.mathworks.com/ help/deeplearning/ref/trainlm.html [Accessed: 16 August 2019].

Spindle Load Meter. [online] Available at https://ftp.tormach.com/store/index.php?app=ecom&ns= catshow&ref=Spi ndle_Load_Meter [Accessed: 31 August 2019].