PROCESSING TIME ESTIMATION IN PRECISION MACHINING INDUSTRY USING AI

LIM SAY LI

FACULTY OF ENGINEERING
UNIVERSITY OF MALAYA
KUALA LUMPUR

2017

PROCESSING TIME ESTIMATION IN PRECISION MACHINING INDUSTRY USING AI

LIM SAY LI

THESIS SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF INDUSTRIAL ELECTRONICS AND CONTROL ENGINEERING

FACULTY OF ENGINEERING
UNIVERSITY OF MALAYA
KUALA LUMPUR

2017

UNIVERSITY OF MALAYA
ORIGINAL LITERARY WORK DECLARATION

Name of Candidate: LIM SAY LI

Matric No: KQC160002

Name of Degree:

MASTER OF INDUSTRIAL ELECTRONICS AND CONTROL ENGINEERING

Title of Project Paper/Research Report/Dissertation/Thesis ("this Work"):

PROCESSING TIME ESTIMATION IN PRECISION MACHINING INDUSTRY USING AI

Field of Study: ARTIFICIAL INTELLIGENCE

I do solemnly and sincerely declare that:

(1) I am the sole author/writer of this Work;
(2) This Work is original;
(3) Any use of any work in which copyright exists was done by way of fair dealing and for permitted purposes and any excerpt or extract from, or reference to or reproduction of any copyright work has been disclosed expressly and sufficiently and the title of the Work and its authorship have been acknowledged in this Work;
(4) I do not have any actual knowledge nor do I ought reasonably to know that the making of this work constitutes an infringement of any copyright work;
(5) I hereby assign all and every rights in the copyright to this Work to the University of Malaya ("UM"), who henceforth shall be owner of the copyright in this Work and that any reproduction or use in any form or by any means whatsoever is prohibited without the written consent of UM having been first had and obtained;
(6) I am fully aware that if in the course of making this Work I have infringed any copyright whether intentionally or otherwise, I may be subject to legal action or any other action as may be determined by UM.

Candidate's Signature Date:

Subscribed and solemnly declared before,

Witness's Signature Date:

Name:

Designation:

ABSTRACT

Processing time estimation of a machining process is a crucial task in order to gain higher profits, stand out amongst the competition and also grow the customer portfolio in precision machining industry. By having an accurate processing time estimation, a well-planned production schedule can be established and machine capacity availability can be checked to meet customer's estimated time of delivery (ETD). These time estimations are usually done and revised by a tooling process expert. However, the estimation of each and every individual is different based on their knowledge and experiences. In this research, a system is designed to estimate processing time by using artificial intelligence knowledge. Wire electrical discharge machining (WEDM) process is focused and the time taken for the processing is analysed. Input variables such as material type of job, size of copper wire used to run the process, operation mode set for the WEDM machine, number of cuts and the thickness of workpiece are considered as important in estimating the processing time. The objectives of this project are to design a system for processing time estimation, to estimate the processing time required for specific machining process and to verify the accuracy of processing time estimation. Neural Network (NN) model is chosen as the artificial intelligence approach used in this research. Levenberg-Marquardt algorithm is used as the training algorithm. The results show that the data best validation performance is 7.1085 at epoch 27. An AI approach for processing time estimation by implementing desired input parameters and machining data is tested and completed.

Keywords: artificial intelligence, artificial neural network, precision machining, time estimation

ABSTRAK

Pengiraan masa pemprosesan proses pemesinan adalah tugas penting untuk mendapatkan keuntungan yang lebih tinggi, menonjol di antara persaingan dan juga mengembangkan portfolio pelanggan dalam industri pemesinan yang tepat. Dengan mempunyai anggaran masa pemprosesan yang tepat, jadual pengeluaran yang dirancang dengan baik boleh ditubuhkan dan ketersediaan kapasiti mesin boleh diperiksa untuk memenuhi anggaran masa pengiriman pelanggan. Anggaran masa ini biasanya dilakukan dan disemak oleh pakar proses mesin. Walau bagaimanapun, anggaran setiap individu berbeza berdasarkan pengetahuan dan pengalaman mereka. Dalam kajian ini, satu sistem direka untuk menganggarkan masa pemprosesan dengan menggunakan pengetahuan kecerdasan buatan. Proses pemesinan elektrik Wire (WEDM) ditumpukan dan masa yang diambil untuk pemprosesan dianalisis. Pemboleh ubah input seperti jenis pekerjaan, saiz dawai tembaga yang digunakan untuk menjalankan proses, mod operasi yang ditetapkan untuk mesin WEDM, bilangan potongan dan ketebalan benda kerja dianggap sebagai penting dalam mengestimasi masa pemprosesan. Objektif projek ini adalah untuk merekabentuk satu sistem bagi anggaran masa pemprosesan, untuk menganggarkan masa pemprosesan yang diperlukan untuk proses pemesinan tertentu dan untuk mengesahkan ketepatan masa memproses anggaran. Model Neural Network (NN) dipilih sebagai pendekatan kecerdasan buatan yang digunakan dalam kajian ini. Algoritma Levenberg-Marquardt digunakan sebagai algoritma latihan. Hasilnya menunjukkan bahawa prestasi pengesahan data terbaik adalah 7.1085 pada epok 27. Penggunaan AI untuk menganggarkan anggaran masa dengan melaksanakan parameter input yang dikehendaki dan data pemesinan telah dijalankan.

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to all those who helped me in completing this Research Project. The special thanks goes to my project supervisor, Ir. Dr. Chuah Joon Huang who guided me throughout the process of carrying out the works. I would like to express my appreciation towards his ideas, guidance, encouragement, and supervision in ensuring the progress of this project run smoothly as scheduled. My grateful thanks are extended to all my friends for their bright ideas and firm support. Last but not least, I would like to thanks my family member for their continuously support and enthusiastic encouragement throughout my postgraduate study in University of Malaya (UM).

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

AI	-	Artificial Intelligence
ACU	-	Accuracy Priority
ANFIS	-	Adaptive Neuro-Fuzzy Inference System
ANN	-	Artificial Neural Network
CBN	-	Case Based Reasoning
CNN	-	Convolutional Neural Network
EA	-	Evolutionary algorithms
ETD	-	Estimated Time of Delivery
GA	-	Genetic Algorithm
MAPE	-	Mean Absolute Percentage Error
MSE	-	Mean Squared Error
NN	-	Neural Network
SPD	-	Speed Priority
STD	-	Standard
TC	-	Tungsten Carbide
TSK	-	Takagi-Sugeno-Kang
UM	-	University of Malaya
WEDM	-	Wire Electrical Discharge Machining

LIST OF APPENDICES

CHAPTER 1: INTRODUCTION

The brief information of this project “Processing time estimation in precision machining industry using AI” is presented in this chapter.

1.1 Background

Nowadays, lead time estimation has become a vital point in inventory control and factory planning (Ionnou & Dimitriou, 2012). Processing time estimation of a machining process is a crucial task in order to gain higher profits, stand out amongst the competitor and also grow the customer portfolio in precision machining industry (Mucientes, Vidal, Bugarin, & Lama, 2008). It does plays an important role as it would affects the production or manufacturing plan, as well as the machine capacity requirement to meet customer’s estimated time of delivery (ETD). While due date determination affects the estimation problem and decision making, time estimation is treated as one of the service level requirement (Behrouznia, Azadeh, Pichka, Pazhoheshfar, & Saberi, 2011). By having an accurate processing time estimation, a well-planned production schedule can be established (Mucientes, Vidal, Bugarin, & Lama, 2008). The lead time of a machine depends on numerous input variables and the complexity of the machining required, thus time needed to complete the process of a part is a case-by-case basis. Input variables such as material type of job, size of copper wire used to run the process, operation mode set for the WEDM machine, number of cuts and the thickness of workpiece are considered as important in estimating the processing time (Dasgupta, Dutta, Majumder, & Panja, 2017). Thus, a good estimation of the processing time of a part involves the selection of the appropriate settings based on the input variables. The data about the dependencies of processing time with the input variables is also very essential (Mucientes, Vidal, Bugarin, & Lama, 2008).

These estimations are usually done and supervised by a tooling process expert (Jun, Park, & Suh, 2006). Therefore, it is important for the expert to be able to easily obtain the information about the different processing times required for different parts (Mucientes, Vidal, Bugarin, & Lama, 2008).

In precision machining industry, an accurate estimation of processing time is difficult to attain due to most of the products are customized (Mucientes, Vidal, Bugarin, & Lama, 2008). By reducing the processing time, the company is able to reduce the expenses and costs, save inventory, improve the productivity, respond customer's enquiries in the quickest time and meet the quality requirement (Mucientes, Vidal, Bugarin, & Lama, 2008). According to Mucientes et al. (2008), processing time estimation is one of the most crucial jobs in a manufactured goods lifecycle. However, insufficient of preceding manufacturing knowledges and the production complexity makes time estimations harder (Duffie, Bendul, Windt, & Knollmann, 2016).

1.2 Problem Statement

Processing time estimation is an unavoidable task in precision machining industry. With the importance of the task, it is however difficult to obtain a reliable output (Angius, Colledani, & Horvath, 2015). To enhance the manufacturing performances including design stage, manufacturing planning and operational stage, past knowledge is widely constituted in current manufacturing industry (Mourtzis, Doukas, Fragou, Efthymiou, & Matzorou , 2014). These time estimations are usually done and revised by a tooling process expert. However, the estimation of each and every individual is different based on their knowledge and experiences. Underestimation or Overestimation of a process could bring negative impact to the company’s future business opportunity (Florjanič & Kuzman, 2012). The accuracy of the time estimation may be experimental and significance deviations will arise (Mourtzis, Doukas, Fragou, Efthymiou, & Matzorou , 2014). With today’s extremely competitive environment, such deviations should not be allowed. If a job is underestimated, the company may face capital loses. However, if it is overestimated, the job is has high possibility to be assigned to the competitive supplier (Florjanič & Kuzman, 2012). Therefore, a system based on the concrete technical data is proposed to obtain an accurate and reliable processing time estimation. In this research project, wire electrical discharge machining (WEDM) process is focused and the time taken for the processing is analysed.

1.3 Objectives

The objectives of this project are:

a) To design a system for processing time estimation.

b) To estimate the processing time required for specific machining process.

c) To verify the accuracy of processing time estimation.

1.4 Scope

This purpose of this project is to design a system that is able to estimate the processing time required for specific machining process. The specific machining process is defined as wire electrical discharge machining (WEDM). The processing time estimation for the WEDM is determined based on the defined input variables. The input variables are material type of job, size of copper wire used to run the process, operation mode set for the WEDM machine, number of cuts and the thickness of workpiece. Machine capacity and various costs including inventory costs, material costs, labour cost and etc. are not covered in this project

1.5 Project Structure

In Chapter 1, a short introduction regarding on the project is shown. The introduction includes the project background, problem statement, objectives, scope of the project and lastly the project structure.

In Chapter 2, the literature review on processing time estimation is written. Related information regarding on the topics are discussed. For example, processing time, time estimation and previous studies done on processing or lead time estimation.

In Chapter 3, methodology involved in this project is illustrated with the aid of flow chart. The methodologies involved are literature review, project planning, system modelling, and result analysis and discussion.

In Chapter 4, the results of this project are included. The data obtained from the result is described as well. Then, the analysis that carried out for this project is discussed. The experiments to be carried out is also described in details.

In Chapter 5, conclusion of this project are made. The estimated processing time based on the result of the analysis is concluded. Furthermore, suggestion for future work is discussed to improve this project.

CHAPTER 2: LITERATURE REVIEW

This chapter illustrates the basic idea of various approaches used in estimating time in machining and manufacturing industry. This project focused on modelling a system for processing time estimation.

2.1 Processing time

According to James et al. (2013), the processing time of a process involves the considerations of the below criteria:

a) Set up time

b) Operating time (handling time and machining time)

c) Unloading time

d) Miscellaneous time (tool changing and repairing time, checking and inspection, fatigue allowance, cleaning and disposal, et al.)

2.2 Wire Electrical Discharge Machining

Wire electrical discharge machining (WEDM) is widely used in precision machining industry to machine complex and required tight tolerances profile. To meet the machining precision as well as machining time is the main tasks to be solved in WEDM nowadays. In the WEDM machining process, a constantly travelling wire electrode that is made of Tungsten, Copper or Brass is worked as the cutting tool (Conde, Sanchez, Plaza, Olivenza, & Ramos, 2015). Unlike conventional manufacturing machine, WEDM removes the material with electricity by means of erosion of spark. The repetitive sparks between workpiece and wire electrodes indicating the machining is taking place by eroding the workpiece material (Vundavilli, Kumar, & Priyatham, 2012). How WEDM works is illustrated from the top view in Figure 2.1. It clearly shows how the wire electrode cut material by travelling within the workpiece.

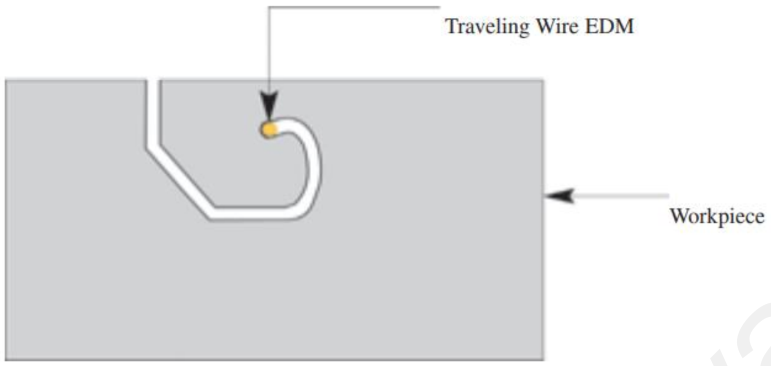

Figure 2.1: Travelling of wire electrodes in workpiece from top view.

On the other hand, Figure 2.2 illustrates the basic operation of WEDM process. In the operation, several components are involved such as roller, wire electrode, workpiece, working table, and etc. (Vundavilli, Kumar, & Priyatham, 2012).

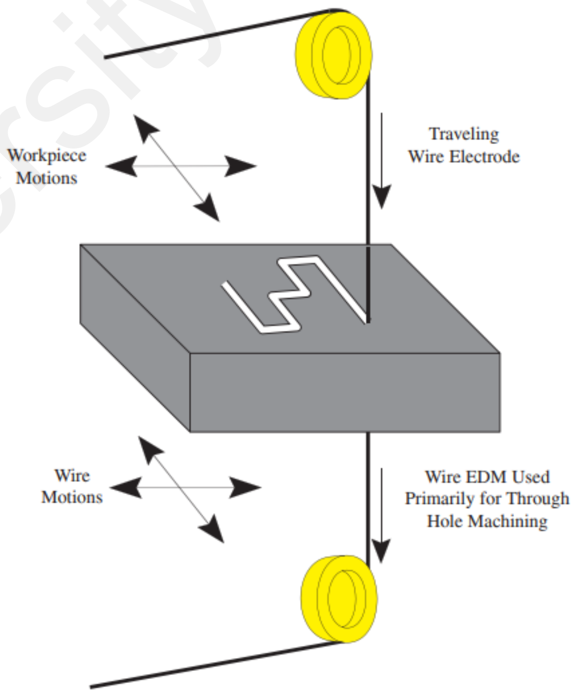

Figure 2.2: Basic Operation of WEDM process (Vundavilli, Kumar, & Priyatham, 2012).

By adjusting the proper voltage, and with the presences of deionized fluid, discharge will happens between the workpiece and the wire electrode (See Figure 2.3).

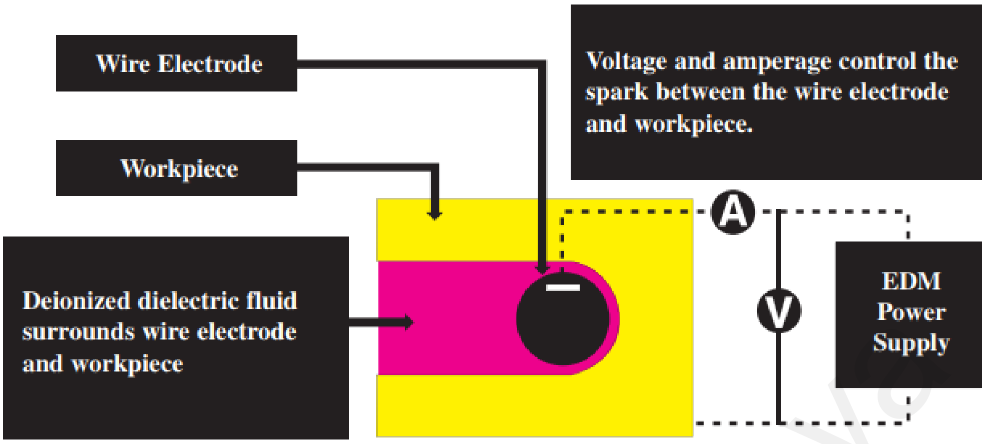

Figure 2.3 (a): Schematic of WEDM process.

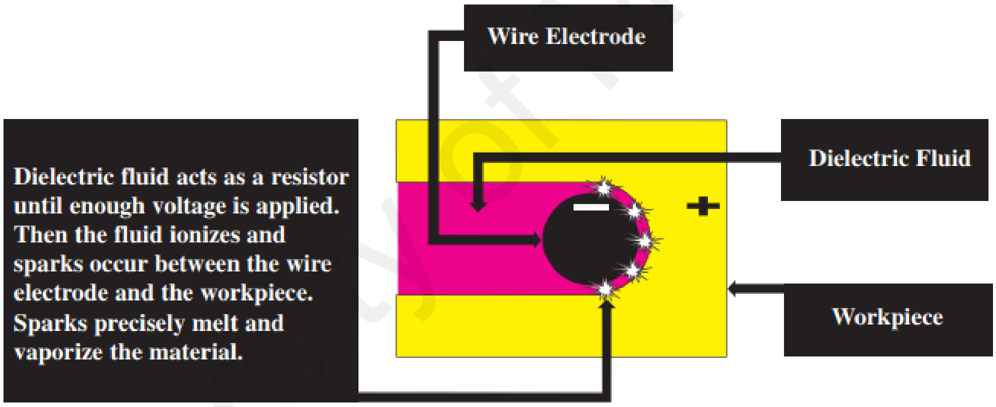

Figure 2.3 (b): Schematic of WEDM process.

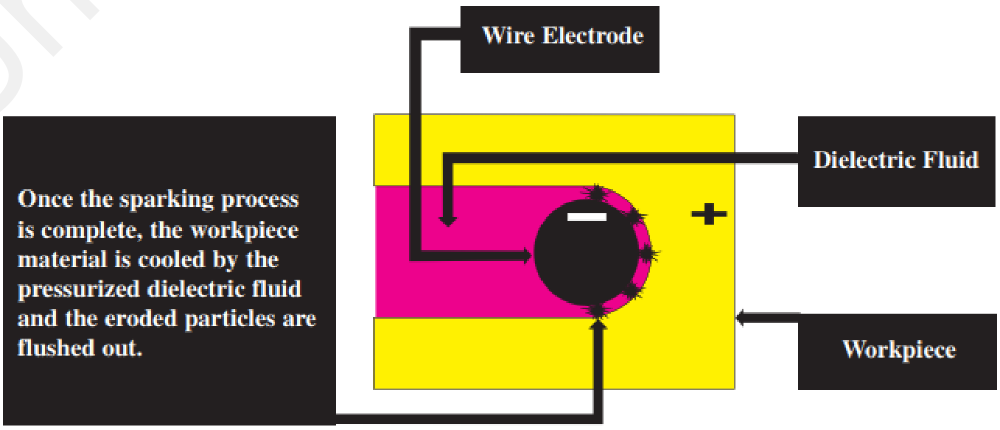

Figure 2.3 (c): Schematic of WEDM process.

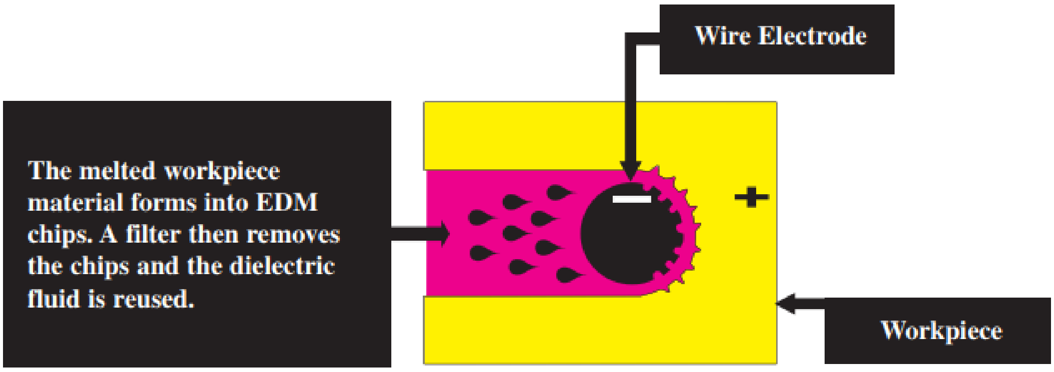

Figure 2.3 (d): Schematic of WEDM process.

The machining process of WEDM is complex as it integrates multiple process parameters. Therefore, it is hard to define the WEDM process with a precise mathematical model (Li, Kong, Lu, Yuan, & Fang, 2007). To model the WEDM process, Li et al. (2007) had used neural-fuzzy inference system associates with the genetic algorithms. An equivalent feed-forward neural network architecture is established and shown in figure below:

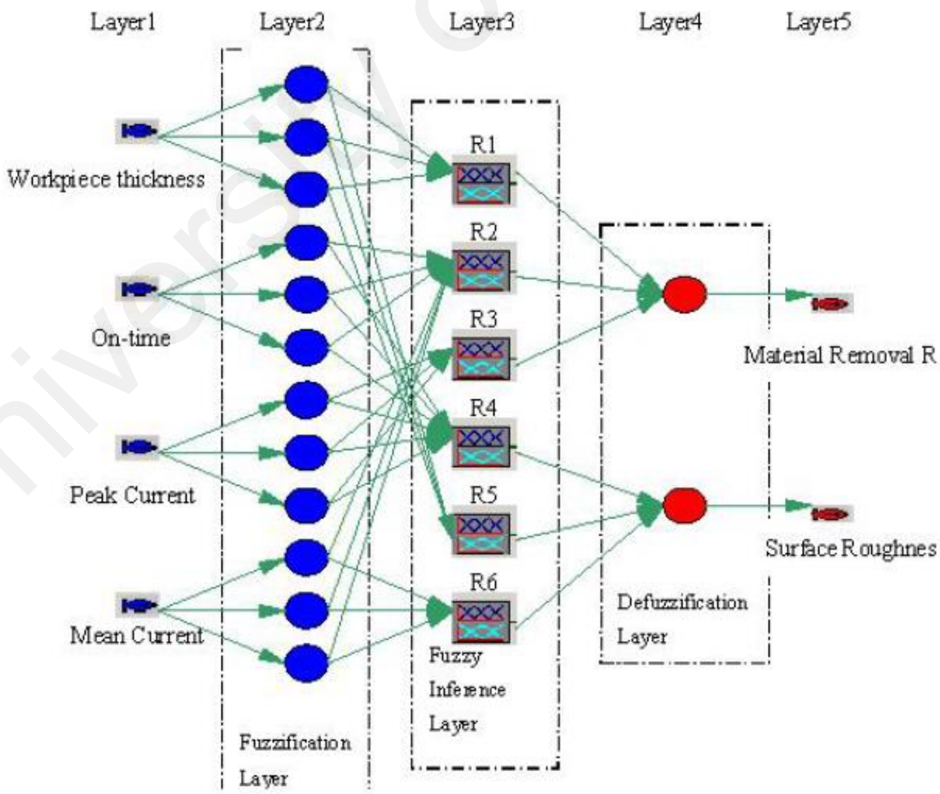

Figure 2.4: Feed-forward fuzzy neural network architecture for WEDM modelling (Li, Kong, Lu, Yuan, & Fang, 2007).

In figure above, Layer 1 shows the input layer where the inputs are various parameters used for setting up the WEDM machining process. Followed by Layer 2 shows the fuzzification layer where the neurons in this layer reflects a function of input membership of the fuzzy rule antecedent. Next, Layer 3 shows the fuzzy inference layer where the training of the parameters of neurons into the final shape were took place. Then, Layer 4 shows the defuzzification layer where the membership function is defuzzified by using center of area method and applied by merging the computational results of previous layer. Lastly, Layer 5 which act as the output layer will compute the result and output after defuzzification. The above model provides a brief image of process knowledge in WEDM process.

2.3 Processing time estimation

Processing time is one of the factor that affects lead time that can be translated to delivery time in the perspective of customer (Berling & Farvid, 2014). Including processing time, job queuing time and transportation time plays a critical role in evaluating the manufacturing performance of a company (Mourtzis, Doukas, Fragou, Efthymiou, & Matzorou , 2014). The processing time estimation are usually used to establish a production plan by scheduling a job based on customer’s order with current machine capacity availability in the production floor (Mucientes, Vidal, Bugarin, & Lama, 2008). A solid time estimation is needed in order to provide accurate estimated delivery time to the customer (Berling & Farvid, 2014). If the delivery date is not met, customer may experience quiet amount of losses and the company will lose their reputation as well. The key of estimation is to properly study and determine all the information required to obtain the total processing time needed (Florjanič & Kuzman, 2012).

2.4 Approaches used for processing time estimation

Several approaches were proposed in the past for the estimation of time as shown in Figure 2.5. The approaches that had been used were simulation, statistical method, queuing theory, logistic curves, stochastic analysis and artificial intelligence.

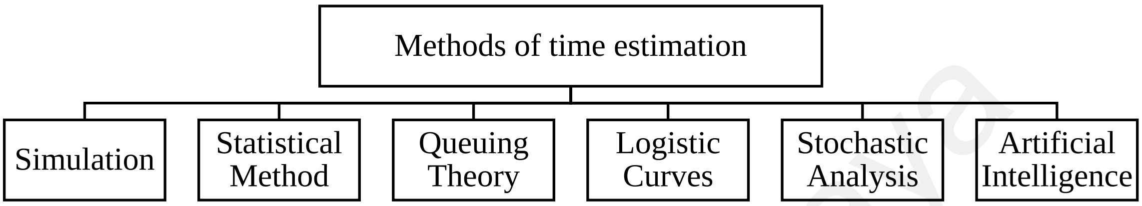

Figure 2.5: Methods of time estimation (Mourtzis, Doukas, Fragou, Efthymiou, & Matzorou , 2014).

According to Mourtzis et al. (2014), the most robust approach among the above is Artificial Intelligence. Nevertheless, in Artificial Intelligence, there are various techniques can be used to do time estimation as shown in Figure 2.6. A brief review of AI approaches are discussed in this sub-section.

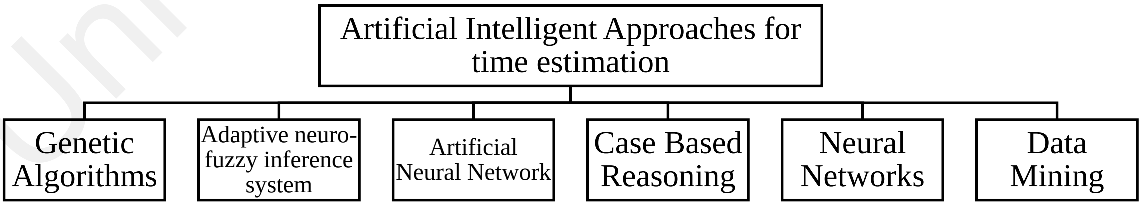

Figure 2.6: Artificial Intelligent approaches for time estimation (Mourtzis, Doukas, Fragou, Efthymiou, & Matzorou , 2014).

2.4.1 Genetic Algorithm (GA)

Mucientes et al. (2008) implemented the Takagi-Sugeno-Kang (TSK) fuzzy rule model for processing time estimation in a manufacturing industry. This method seeks for a great accuracy regression model but at the same time retaining the knowledge structure given by an expert. Besides, the easiness of extracting information of regression functions related to the machine is also very important. Most of the cases, the variety of information required is subjected to the characteristics of the system industry. It is a must for the representation of regression functions to meet the requirement of the expert based on the perspective of information extraction.

In brief, the expert stresses the characteristics below for the representation of the regression functions:

a) The contribution of every input variables to the estimation of processing time must be explicit.
b) The regression function is depends on a new generated variable which denotes the relations between the input variables.

In addition, for certain cases, the knowledge of an expert is structured with the input variables polynomials as below:

$$Processing\ time = 100 \cdot length + 200 \cdot volume$$

The processing time of a machine is defined as a polynomials of several input variables and these variables are joined in various methods:

$$\sum_{i} \propto_i \cdot \prod_{j=1}^{na} x_j^{\delta_{i,j}}$$

where $\propto_i$ represents the coefficients,

x_j represents the input variables where $j = 1, \ldots, na$, and

$\delta_{i,j}$ represents the indicator variable.

To model a specific machine, the polynomials representing the processing time estimation of a class of the input variable are required. Since the input variables are varies, a regression function for every class of the input has to be gained for the learning process. In summary, for each input, the system will choose a regression function for the specific machine and estimate the processing time required.

Thus, Mucientes et al. (2008) proposed a rule model as below:

$$R_k: If\ X_1\ is\ A^1_{l^k_1}, \dots, X_{na}\ is\ A^{na}_{l^k_{na}}$$

$$Then\ Y\ is\ \sum_i \propto_i \cdot f_i(x_1, \dots, x_{na})$$

where X_j represents the linguistic variable,

$A^j_{l^k_j}$ represents the linguistic label of variable where $l^k_j = 1, \dots, nl_j$,

Y represents the output variable, and

$f_i(x_1, \dots, x_{na})$ represents the functions of input variables x_j.

The classification of linguistic labels, functions (f_i), and also coefficients (α_i) with different structures are required for the above TSK rule model.

Evolutionary algorithms (EA) allows the flexibility in the representation of the results, and also enable the management of the interchange between simplicity and accuracy for information extraction. Due to various considerations, genetic programming is chosen as the most appropriate evolutionary algorithm by the author. Each individual is a tree of variable length in genetic programming and allowed to have a dissimilar structure. Moreover, the learning process is based on a set of training data.

For processing time estimation, the genetic programming algorithm is described as below:

a) Initialize the population by generating the rules, evaluating the population and resizing the population A.

b) Repeat the iterative part of the algorithm for *maxIterations* time and start the crossover and mutation, then evaluate the population and resize the population A.
c) Construct the final knowledge.

By implementing genetic programming approach, the algorithm is formed together with token competition in order to sustain the diversity of the population.

2.4.2 Adaptive neuro-fuzzy inference system (ANFIS)

Behrouznia et al. (2008) used an integrated algorithm based on ANFIS model to estimate weekly lead time of a company in Iran. The input parameters introduced in this model for estimation of lead time are processing time summation, fixing times summation as well as breakdowns summation. The cumulative of mentioned parameters are used to determine the operations effectiveness in production lines. The flowchart of ANFIS model is shown as Figure 2.7 below.

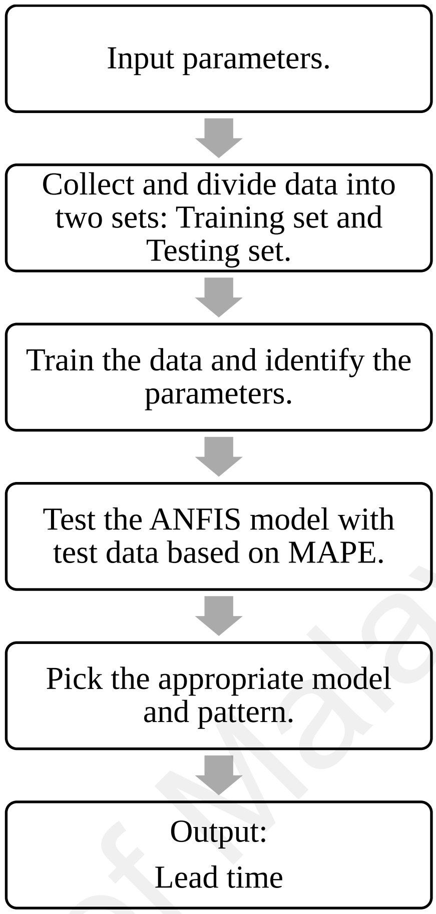

Figure 2.7: Flowchart of ANFIS model (Behrouznia, Azadeh, Pichka, Pazhoheshfar, & Saberi, 2011).

At first, the ANFIS model was tested for two to five membership functions. Later, it was ran for six to seven membership functions in order to obtain desired outputs. The membership functions referred by Behrouznia and the team was G-bell. Mean Absolute Percentage Error (MAPE) was implemented to identify the worst ANFIS model.

2.4.3 Case Based Reasoning (CBR)

Mourtzis et al. (2014) implement CBR model in their studies. The model which is known as knowledge reuse mechanism, is essentially used to compare and contrast the similarities and dissimilarities by adapting previous experiences and knowledges and modify for current cases. It is said to be able to improve the capabilities of problem solving with this CBR model. By using Euclidean distance for numerical features and other categorical features gained, the studies were conducted. Figure 2.8 shows the workflow of time estimation by implementing CBR mechanism.

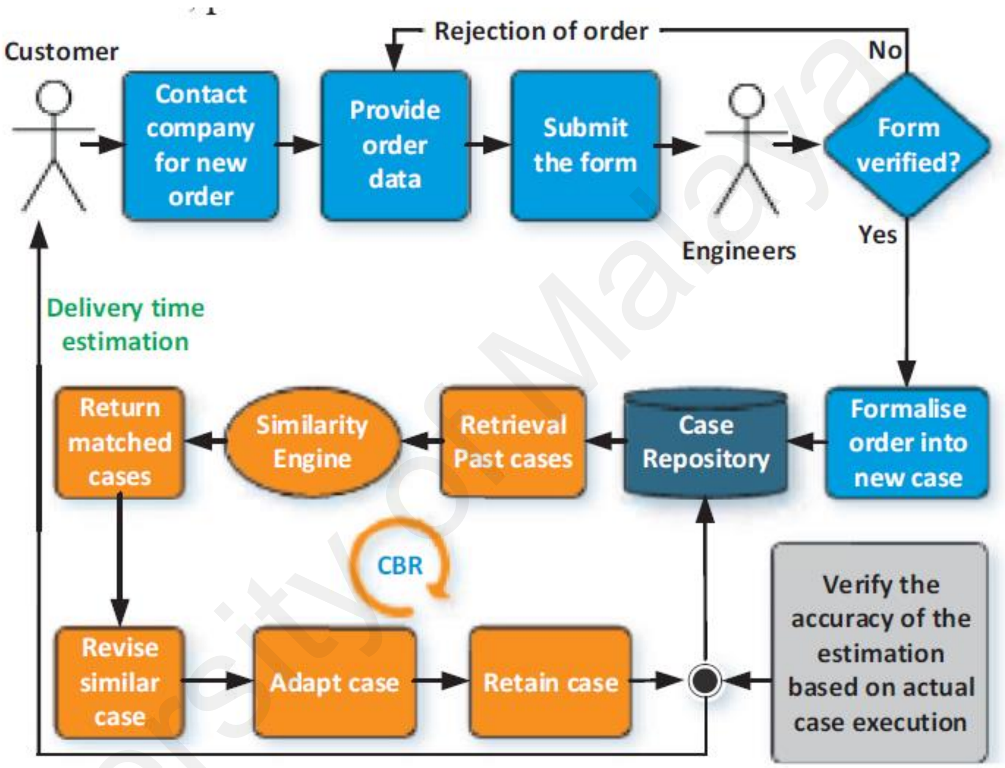

Figure 2.8: Workflow of time estimation by implementing CBR model (Mourtzis, Doukas, Fragou, Efthymiou, & Matzorou , 2014).

CBR as the enhanced version of rule-based approaches. Based on the dynamic memory of similarity engine, the CBR mechanism relates the previous case and new case to learn the pattern and solve new order case. Similarity engine plays a crucial role in this model as it retrieve and adapt the case when new order comes. Figure 2.6 shows how to similarity engine works.

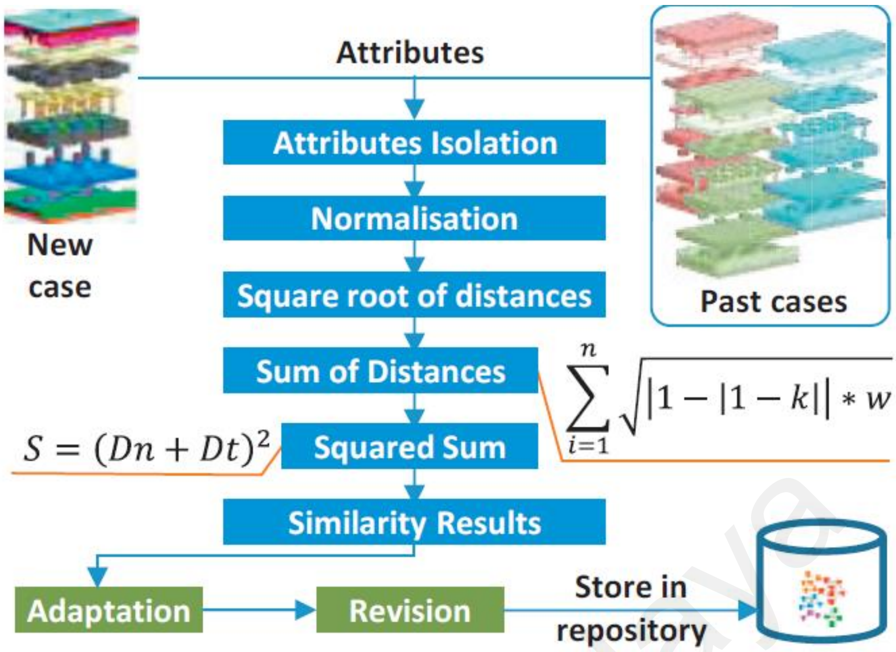

Figure 2.9: Mechanism of similarity engine (Mourtzis, Doukas, Fragou, Efthymiou, & Matzorou , 2014).

When a new order is received, it is revised by using CBR model illustrated in Figure 2.5. With similarity engine shown in Figure 2.6, it is necessary to calculate the similarity measures. The formal representation is described as below:

$$D_n = \sum_{i=1}^{n} \sqrt{\left| \frac{T_{ni}}{T_{ni}} - \left| 1 - \frac{T_{pi}}{T_{pi}} \right| * w_i \right|}$$

where D_n represents the distance of numerical,

n represents the features number,

T_{ni} represents the new order features,

T_{pi} represents the previous order features, and

w_i represents the attribution weights.

To determine the distance between compared alphanumeric attributes by using Euclidean distance, the equation below is used.

$$D_t = \sum_{i=1}^{n} \sqrt{\left|\frac{T_{ni}}{T_{ni}} - |1 - k| * w\right|}$$

where D_t represents the distance of text, and k represents the corresponding value for text attribution.

To combine the numerical and text distance, below equation is used:

$$S = (D_n + D_t)^2$$

According to Mourtzis et al. (2014), in comparing with true values, CBR model is able to solve the problems of high precision.

2.4.4 Data Mining

Data mining is commonly used to generate information by studying the hidden pattern within a set of data provided (Ozturk, Kayaligil, & Ozdemirel, 2006). The data provided are mostly in the tabular form where the columns and rows play different role. The columns are acted as the attributes, while the rows are the instances or transactions (Perzyk, Kochanski, Kozlowski, Soroczynski, & Biemacki, 2013).

The tree structure of data mining is well describe in Figure 2.10. Data mining approach is usually classified by the type of learning, which are supervised learning or unsupervised learning and also attribute in selection and discretization.

As data mining is widely implemented in manufacturing contexts, Ozturk et al. (2006) believed that it has the potential to perform estimation tasks as well.

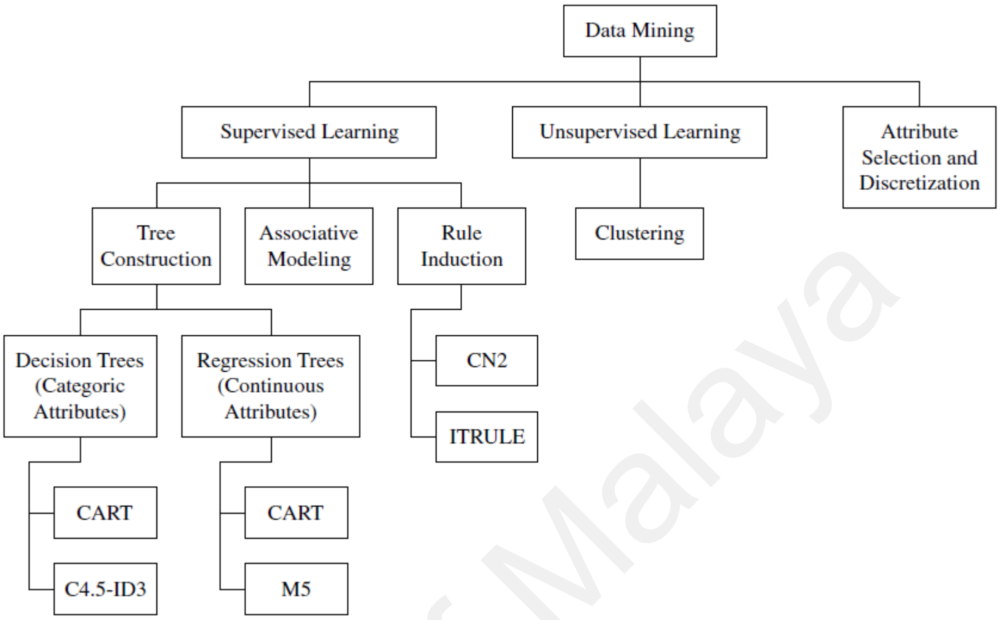

Figure 2.10: Tree structure of data mining (Ozturk, Kayaligil, & Ozdemirel, 2006).

The approach started off with a large amount of attributes that shows the dynamic and static order of the machining company. Ozturk et al. (2006) also established a complex scheme to choose some randomize attributes that have the highest predictive ability. Data mining is known to be efficient and effective for handling a big amount of datasets (Ismail, Othman, & Abu Bakar, 2009). It is proved by Ozturk et al. (2006) that data mining is workable by measuring its performance in the aspects of average absolute error, coefficient of variation, relative error, mean square error, average realised flow time and R^2 value where the model of regression is employed.

2.4.5 Artificial Neural Network (ANN)

Artificial Neural Network approach is widely used in various industrial fields for different applications such as electronics, supply chains, manufacturing planning and scheduling, risk theory, marketing, energy, expert system, food and beverage, and computer software industries (Tehran & Maleki, 2011). Eraslan (2009) and Florjanic and Kuzman (2012) both conducted a study on estimation model for moulding industry by implementing ANN method.

Figure 2.11 shows the solution space of expert estimation model of Florjanic and Kuzman's (2012) study. ANN supported expert estimation possesses narrower solution space compared to unsupported expert estimation. The ANN output is generated from the ANN supported expert estimation.

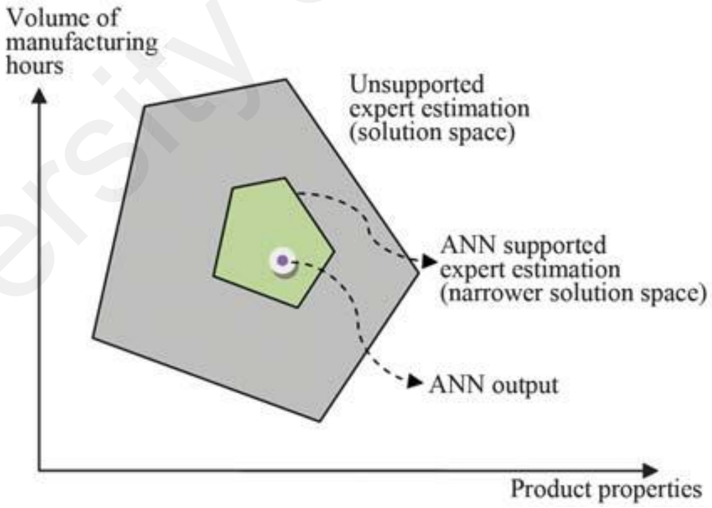

Figure 2.11: Solution space of expert estimation model by using ANN (Florjanič & Kuzman, 2012)

Apart from that, an ANN model is also proposed by Susanto et al. (2012) for estimating lead time in the textile company. ANN model was chosen due to its capability of handling non-linear data (Susanto, Tanaya, & Soembagijo, 2012).

According to Pasini (2015), modelling a complicated system by using ANN approach provides the chance to entirely take nonlinearities into account, even though the number of closed loop existed in the system is not considered as well as the complicated interactions and balance.

2.4.6 Neural Network (NN)

The basic structure of a neural network consists of three main components, which are input layer, hidden layer, and output layer (Zhang & Jia, 2009). The structure can be illustrated as in Figure 2.12.

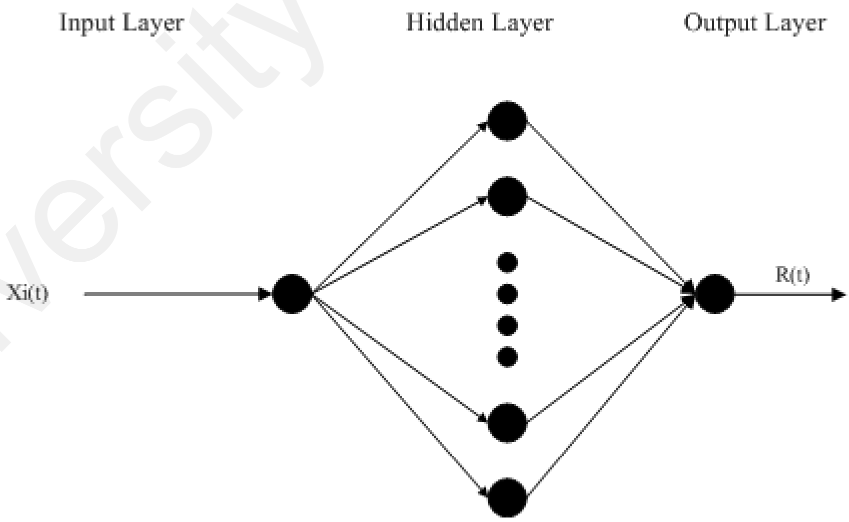

Figure 2.12: Neural Network Structure (Zhang & Jia, 2009).

The number of neurons in input layer is same as the input variables, whereas the number of neuron for output layer will be one (Okubo, Weng, Kaneko, Simizu, &

Onari, 2000). The number of neurons in hidden layer will be the same as the result of calculating by algorithm. Each of these neurons are connected by weight. According to Okubo et al. (2000), NN does not consists of a fix function, it learns from the data stored which make it a considerable method for lead time estimation. Therefore, Okubo et al. (2000) had carried out a study by implementing neural network in lead time estimation. Figure 2.13 shows the lead time estimation model that they had used.

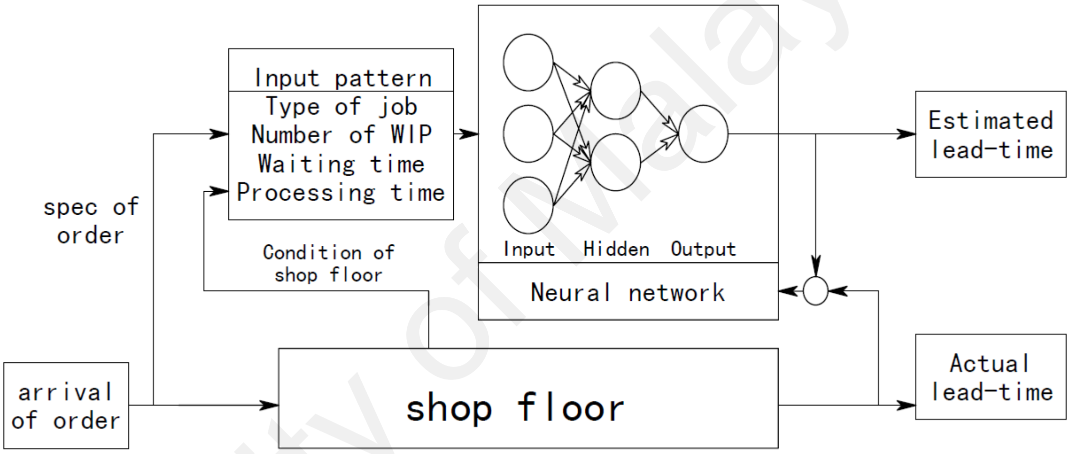

Figure 2.13: Estimation of lead time system (Okubo, Weng, Kaneko, Simizu, & Onari, 2000).

The neural network above is learnt by applying back propagation algorithm. The variance between estimated time and actual time is being feedback to the connection weight between neurons. With the back propagation algorithm, the network is able to learn and reduce the error between the estimated and actual value and hence achieve a higher accuracy of output.

2.5 Summary

In summary, this chapter mainly focus on several approaches used for processing time estimation, which are genetic algorithm, adaptive neuro-fuzzy inference system, case based reasoning, data mining, artificial neural network and neural network. Before these approaches is introduced, a brief idea of processing time, WEDM machining and processing time estimation is also discussed.

CHAPTER 3: METHODOLOGY

This chapter illustrates the methodologies involved in creating the model for processing time estimation in precision machining industry using AI. The methods used to complete the project are shown and described in details. The process flow from the beginning until the entire project completed is included in this chapter.

3.1 Gantt Chart

Table 3.1 illustrate the Gantt chart for this project during Semester 1, Session 2017/2018. The progress of actual activities is monitored so that it did not vary from the planning activities too much.

Table 3.1: Project Gantt chart for Research Project.

No	Activities	Weeks													
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
1	Identify field of research and title.														
2	Identify the project scope, objectives and problem statement.														
3	Study the background of project.														
4	Literature review														
5	Develop the methodologies of project														
6	Preparation for final report														
7	Submission of final report														
8	Preparation and Presentation of Poster														

3.2 Overall Methodology

The flowchart in Figure 3.1 shows the overall methodology upon completing this project. In general, the project is divided into four main parts which require to be done throughout the year. The four main parts are Part A: Literature Review, Part B: Project Planning, Part C: System Modelling, and Part D: Analysis. Each part consists of respective methods to be accomplished which will be further explained in the following sub-titles.

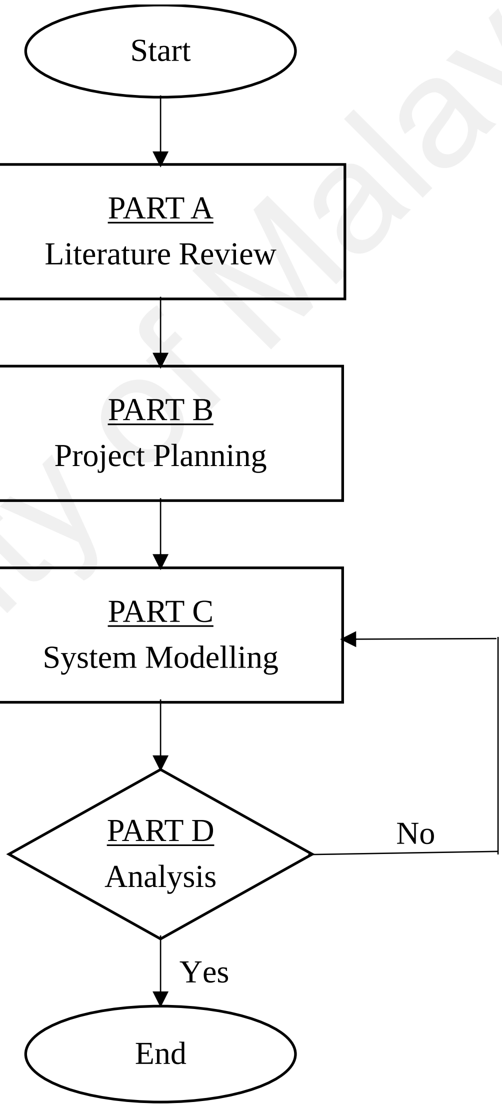

Figure 3.1: Overall methodology.

3.3 Literature Review

Literature review assists in better understanding of a particular topic by gathering data from multiple sources. Through the literature studies that had been carried out, the problem faced in precision machining industry are clearly seen. Thereby, this research work had come out with the objectives. Figure 3.2 illustrates the flowchart of literature review process.

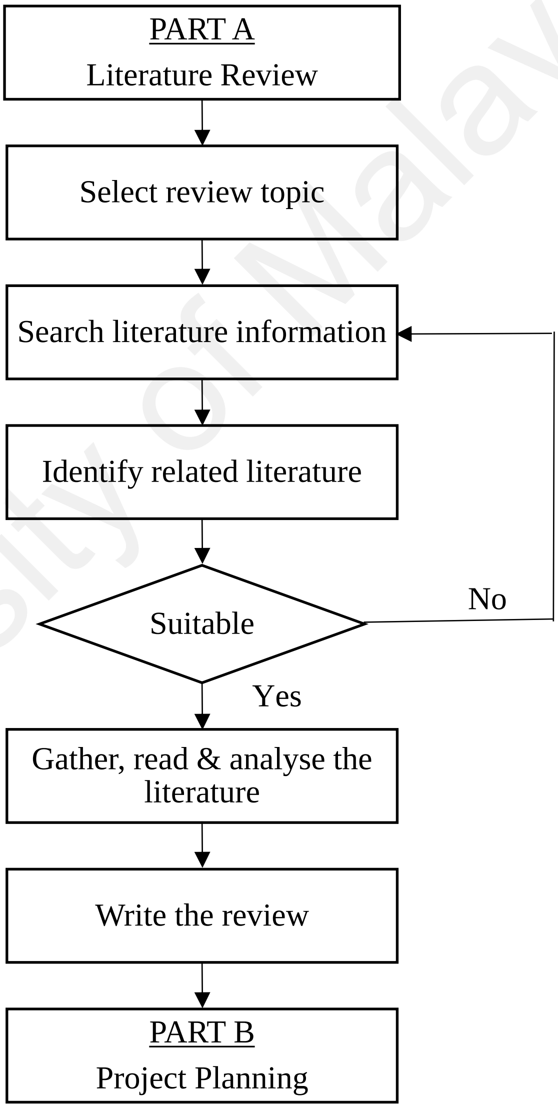

Figure 3.2: Flowchart of literature review process.

From Figure 3.2, the literature review process started with selecting the related topics. In this project, artificial intelligence approach and time estimation are the core subject, therefore these topics are selected for next process. Next, literature information related with the topic selected is searched through several approaches such as keywords, relevant words, and synonyms searches. The literature information are mostly obtained from research papers, internet articles, online journals, review articles and reference book which will be listed later in references. The process continued by identifying the related literature which is a crucial step to get the information that is closest possible to the project. If the literature found is suitable, the research work will then proceed to gather, read and analyse the literature, if not, the research work will turning back to re-look for the literature information. After the literature is analysed, literature review is written based on the topic.

3.4 Project Planning

Project planning shows the groundwork of the project before the modelling and analysis work is started off. The project requires to create the model for processing time estimation in precision machining industry using AI, therefore, establishing the basic concept is important in project planning process. Figure 3.3 demonstrates the activities involved in project planning process.

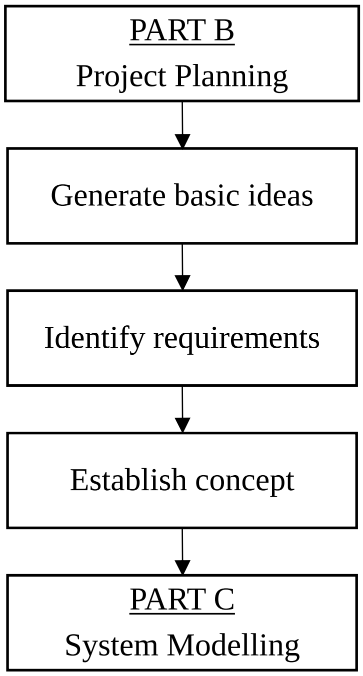

Figure 3.3: Flowchart of project planning process.

3.5 System Modelling

System modelling plays the most important role of the project. Figure 3.4 demonstrates the activities involved in system modelling process. The project started off by selecting the machining process to be studied. In this case, WEDM is being chosen due to its importance in precision machining industry. The factors that could affect the process machining time is determined at this stage. Then, the data for training and testing is collected based on the machining data book of FA-P Series WIRE-CUT EDM SYSTEMS MACHINING CHARACTERISTICS DATA BOOK by Mitsubishi Electric Corporation. The network model is then defined and tested with data collected earlier. The result of the training and testing will be analysed and concluded.

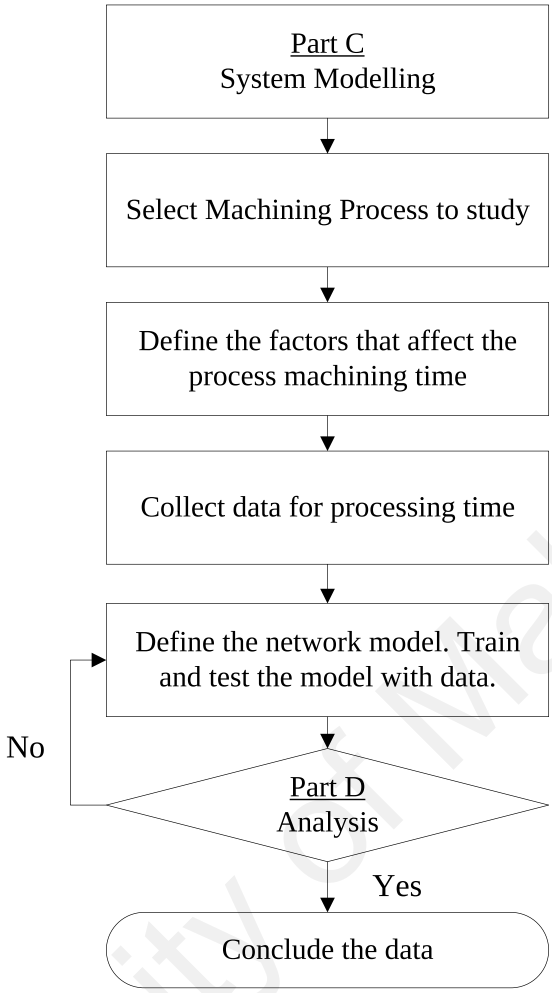

Figure 3.4: Flowchart of system modelling process.

In this project, NN approach is chosen due to its flexible, simple and user friendly tool for system modelling and analysis. To define the network model, it includes the three main processes, which are define input variables, define NN architecture, and evaluate model. The details of each processes are illustrated in Figure 3.5. The input datasets and output of NN model is generated to estimate the processing time. In order to carry out the project, the input variables are defined as the material type of job, wire size for machining, operation mode set on the WEDM machine, the number of cut required and also thickness of the cutting workpiece. Meanwhile the expected output will be processing time. After building the NN architecture, the data are trained and tested to evaluate the workability of the model.

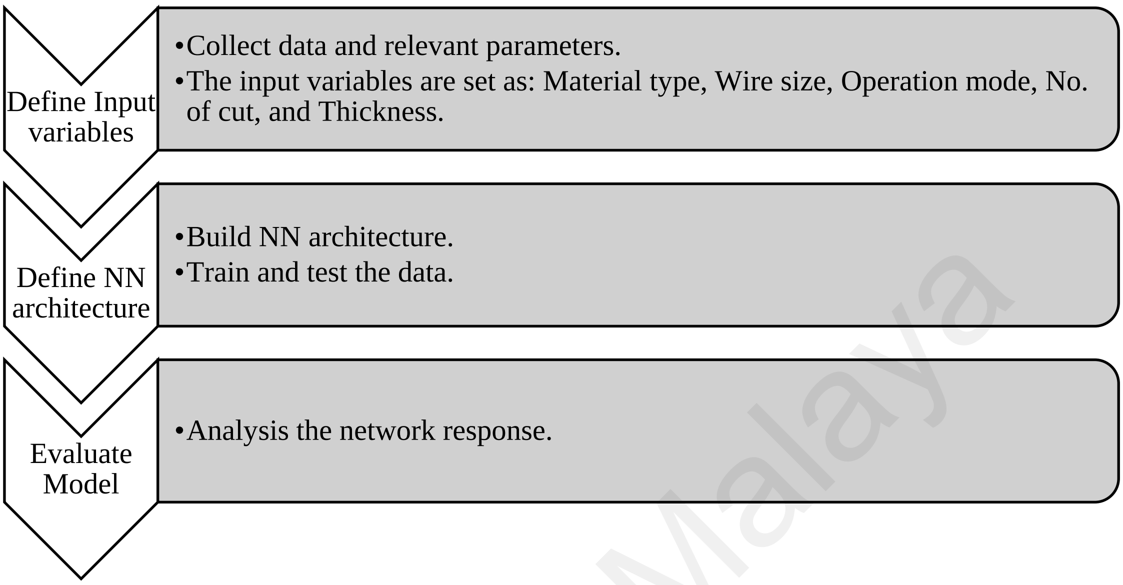

Figure 3.5: Workflow of defining the network model.

3.5.1 Define Input Variables

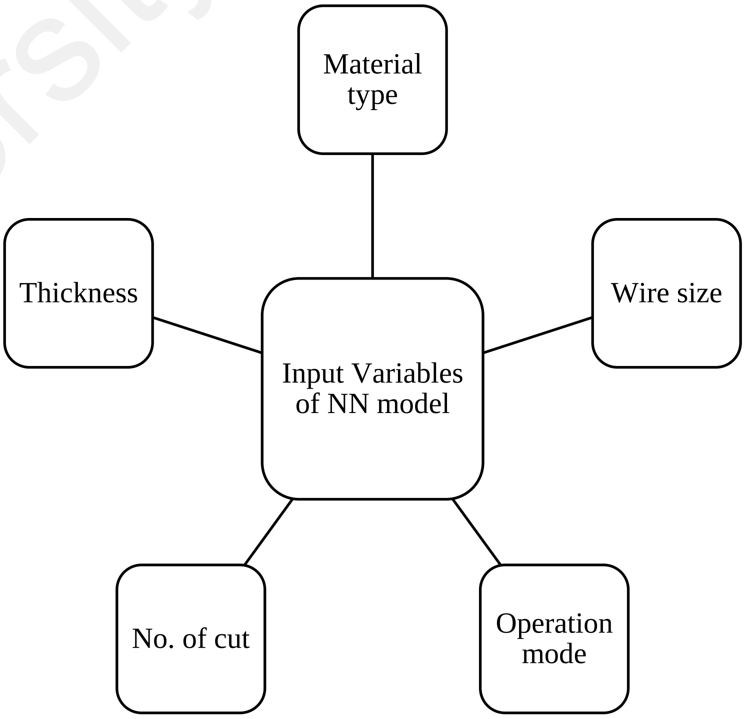

Figure 3.6: Input variables of NN model.

Figure 3.6 shows the input variables of NN model. The input variables are selected based on the importance of these inputs to the processing hour taken for WEDM process. The material type of workpieces that take into count are steel, tungsten carbide, copper and aluminium. Wire sizes used is varies from 0.07mm, 0.10mm, 0.15mm, 0.20mm, 0.25mm, and 0.30mm. Operation modes available are STD (Standard), ACU (Accuracy Priority) and SPD (Speed Priority). Number of cuts are from 1 to 6. And lastly, thickness are varies from 5 to 250mm.

3.5.2 Define Neural Network's Architecture

The architecture of neural network model is illustrated in Figure 3.7. It consists of 5 inputs in the input layer, 3 neurons with the sigmoid activation function in the hidden layer and one neuron with the linear activation function in the output layer.

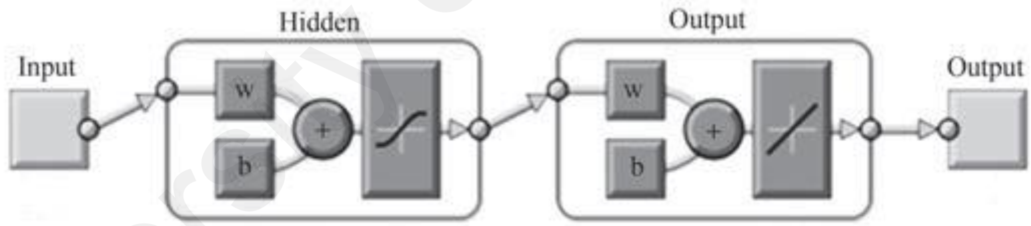

Figure 3.7: Architecture of NN.

3.5.3 Evaluate Model

In this project, dataset gathered divided into two sets for training and testing purpose as illustrated in Figure 3.8.

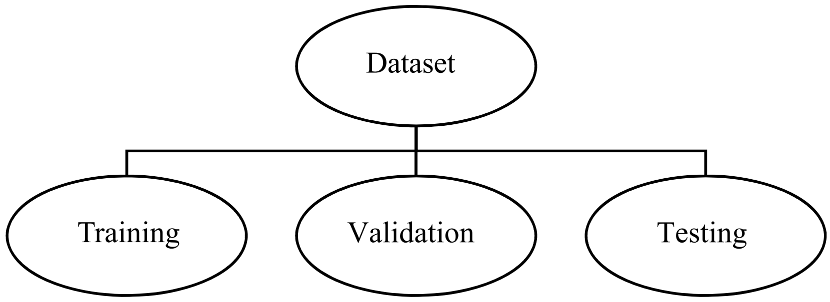

Figure 3.8: Division of datasets.

1200 sets of data are gathered where it 840 sets will be used for training, 180 sets will be used for validation and 180 sets will be used for testing data. In training stage, the 840 sets of data are presented to the network and are adjusted according to its error. Next, in validation stage, the 180 sets of data are used to measure network generalization, and to halt training when the generalization stops improving. Lastly, in testing stage, another 180 sets data which have no effect on the training stage are able to provide independent measure of network performance during and after training.

3.6 Analysis

Software MATLAB is used to run and analyse the system model. The software is chosen based on the capability and solution provided by the software packages. Neural network toolbox in MATLAB provide an easy and convenience platform to perform the analysis of this project. By using the software, architecture of NN model is built and the network is then trained to fit the input and target. Levenberg-marquardt training algorithm is used to train the data. The algorithm was chosen due to it typically requires more memory but less time needed. The training of data will stops when the generalization stops improving, as indicated by an increase in the mean square error of the validation samples.

Last and not least, Error Analysis is conducted to determine the experimental errors happened in carrying out this research project. This is Error Analysis is commonly used in handling errors of experimental in research field (Taylor, 1982). According to Wolfram (2017), there are two types of errors that associated with the result of experiment, which are “precision” and “accuracy”. The analysis is to reduce the both of the two errors, however, in this project, accuracy of the results are focused.

3.7 Summary

In order to carry out this project, four main activities are involved upon completing the project. The four main activities are literature review, project planning, system modelling and analysis. Literature review is carry out by selecting a few review topic at first, then search for literature information, and identify related literature. If the literature is suitable it will proceed with analysing and writing, or else, will back to searching literature information process. Project planning process consists of generate basic ideas, identify the requirements, and establish the concepts. For system modelling, machining process to conduct the study is selected, then factors that affect the process machining time is defined. Last but not least, analysis requires experimental setup, data collection, training and testing of data and result analysis to complete the project. To further improve the result of the training, the number of neurons in hidden layer should be modified so that the system can be training better for estimation.

CHAPTER 4: RESULTS & DISCUSSION

This chapter discusses the results and analysis involved in evaluating the performance of proposed model. The result of the analysis done for this project is shown and discussed. Relevant graphs and tables are recorded and enclosed in this chapter as well.

4.1 Experimental Theory

Figure 4.1 shows the schematic of neuron network in this project. It consists of 3 layer which are input, hidden and output layer. The input parameters defined earlier enters the input layer and undergo training of the NN model.

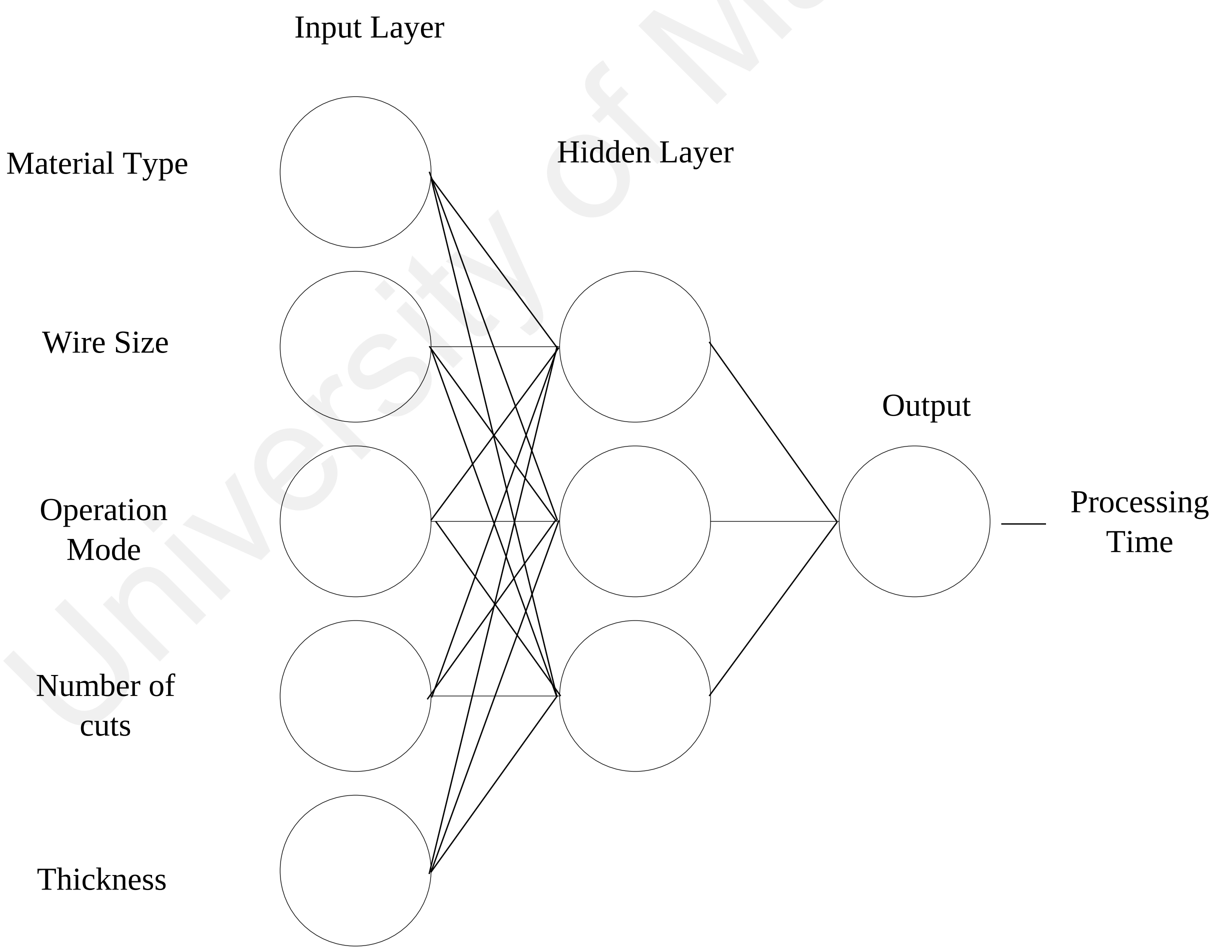

Figure 4.1: Schematic of neuron network for processing time estimation.

As shown as in Figure 4.1, the model is a multilayer feedforward network. The neurons in input layer are consists of linear activation function, whereas the neurons in hidden layer as well as output layer are consists of sigmoid activation function. The purpose of hidden neurons are to interfere between the input and the output with a beneficial manner. The NN architecture is set up as shown in Figure 4.2. As illustrated in Figure 4.1 as well, there will be 5 neurons for the input, and then 3 neurons for the hidden layer and one neuron for the output layer.

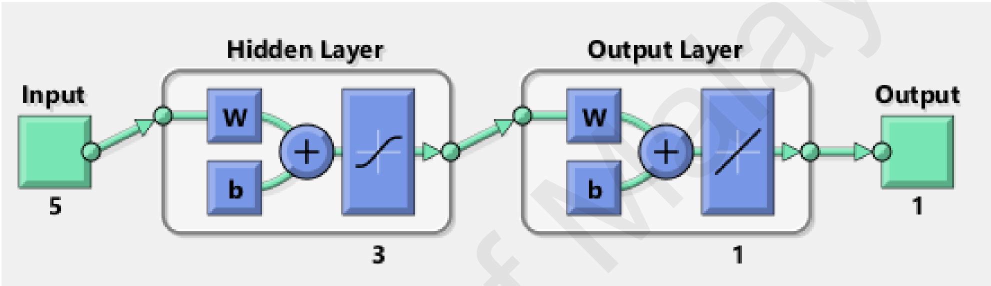

Figure 4.2: NN architecture for processing time estimation.

The Simulink structure of this NN model is also generated and shown in Figure 4.3. From the figure, we can tell that X1 that represents the input of 1200 samples with 5 elements enters the function fitting neural network and Y1 will be the expected output with 1200 samples with 1 element which are processing time.

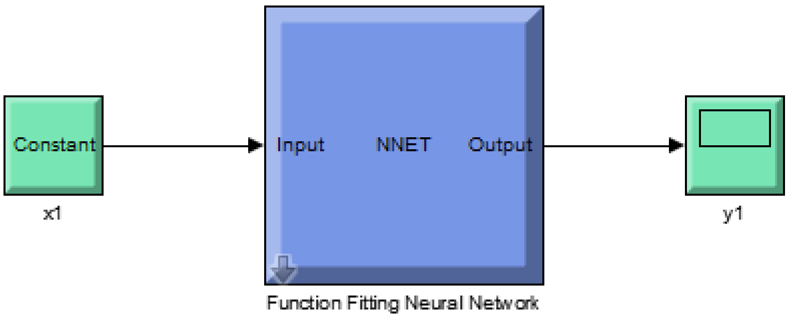

Figure 4.3: Simulink model of NN model.

4.2 Results

Figure 4.4 shows the neural network training interface of MATLAB software. By using the neural network toolbox function, the data was trained and stop at 33 iterations. These toolbox enable the network to retrain and refine to gain the desired result. However, by training the same data of multiple times will produce different result due to different initial conditions and initial sampling. Therefore, it is recommended to stop when the desired output is obtained.

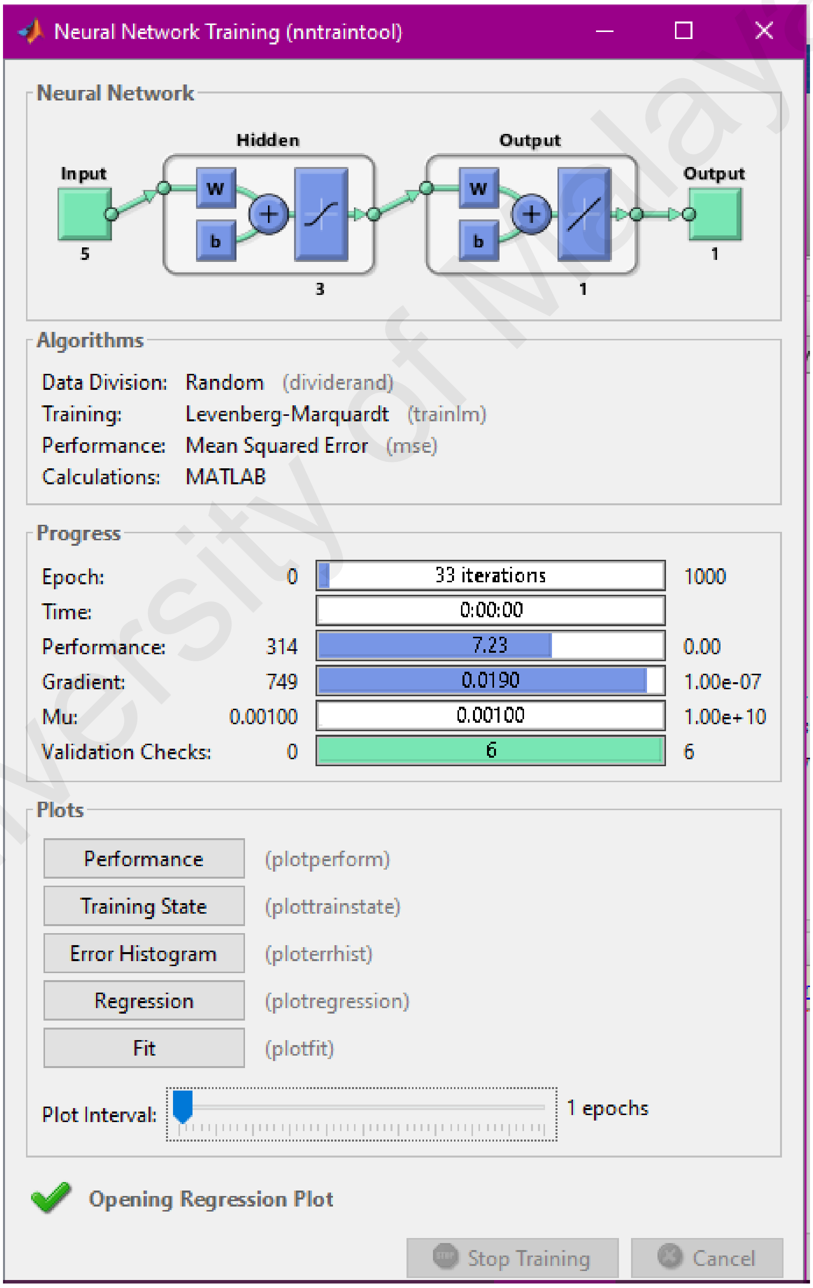

Figure 4.4: Neural network training interface of MATLAB software.

Figure 4.5 shows the plot of Mean Squared Error versus Epochs at 33. The blue lines indicates the train data, green lines indicates the validation data, and the red line indicates the test data. The intersection of two dotted lines across the x-axis and y-axis are the best validation performance of the set which is 7.1085 when the epoch is at 27.

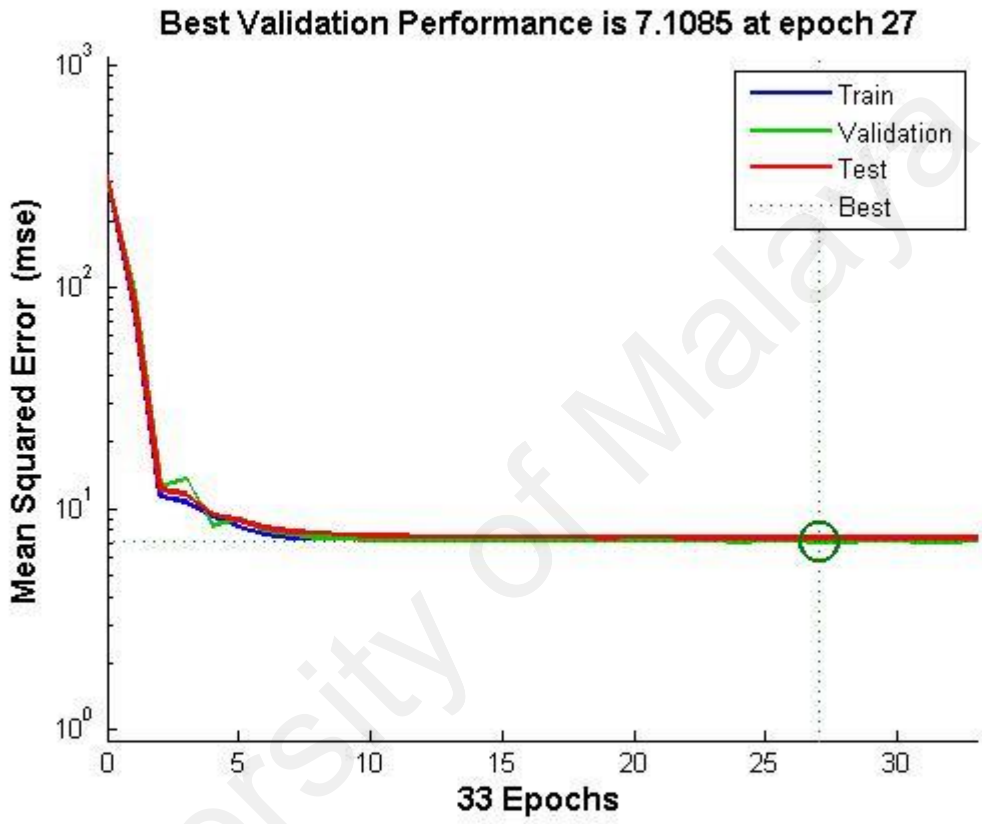

Figure 4.5: Plot of performance.

Figure 4.6 shows the plot of training state which consists of gradient versus epoch, Mu value versus epoch and validation checks versus epoch. When the iteration stops at epoch 33, the gradient obtained is at 0.018975., while the Mu value is at 0.001, and validation checks at 6. The graph pattern fluctuates from epoch 1 to epoch 33.

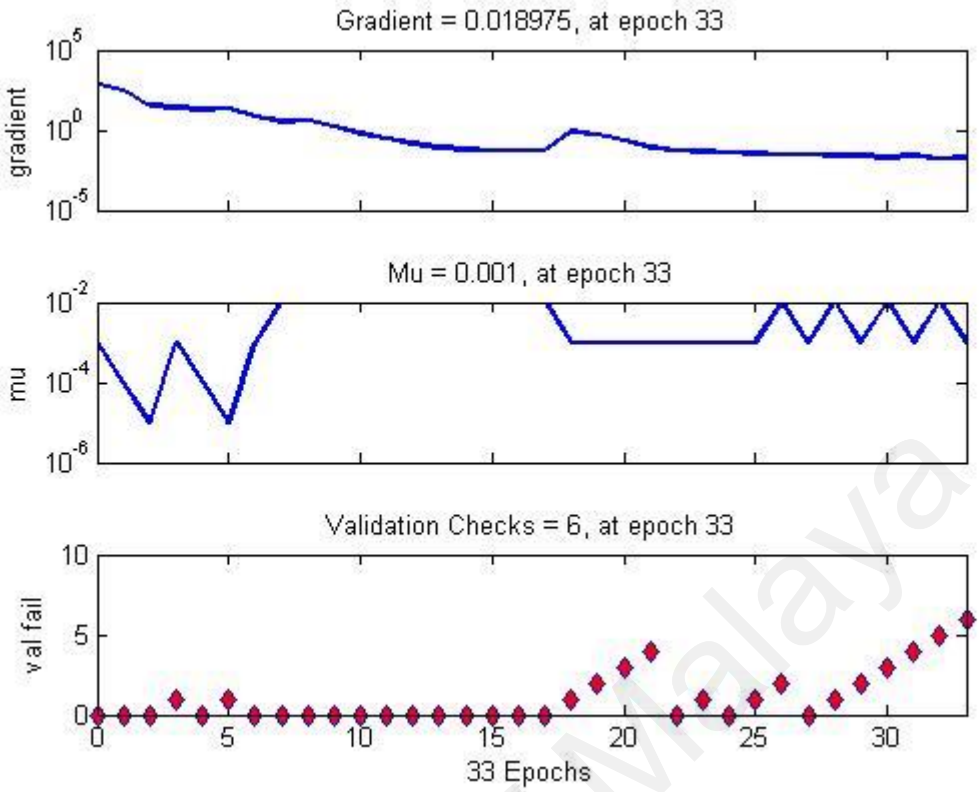

Figure 4.6: Plot of training state.

Figure 4.7 shows the plot of error histogram with 20 Bins, where the instance values versus error values. The blue bar represents the training data, green bar represents the validation data, red bar represents the test data and orange line draws the error at zero value. The graph shows a left shifted bell shape where the zero error lays almost middle of the bell shape. The highest error value is -0.7762, which means the error of the model is pretty low as it is in negative value. However, the error are varies from

-6.602 to 7.379 and the instances varies from 0 to 300. Since the value after 7.379 is very low, it can be negligible.

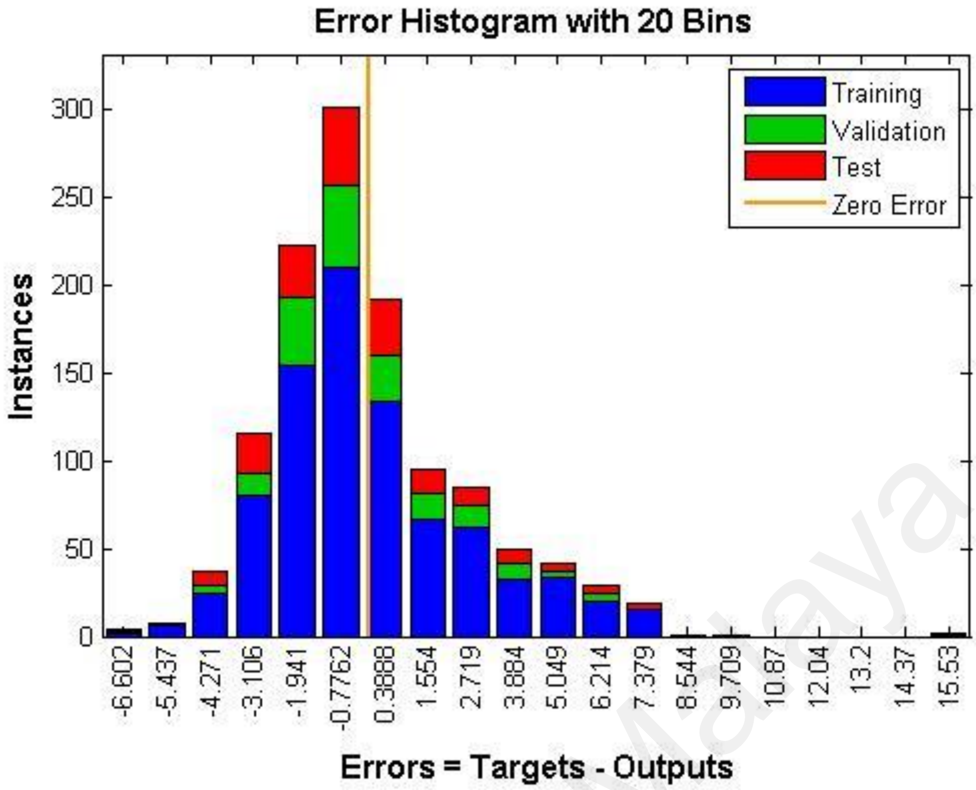

Figure 4.7: Plot of error histogram.

Figure 4.8 shows the plot of regression. The top left plot shows the graph pattern for training data when the regression value is 0.65701 and the data was accumulated at the bottom left of the fitting line. Top right plot shows the graph pattern for validation data when the regression value is 0.70642 and the data is also mainly at the bottom left of the fitting line. Bottom left plot shows the graph pattern for test data when the regression value is 0.64398 and the data shows less accumulated at the bottom left but also appeared all along the fitting line. Bottom right plot shows the graph pattern for all of the training, validation, and test data when the regression value is at 0.6629.

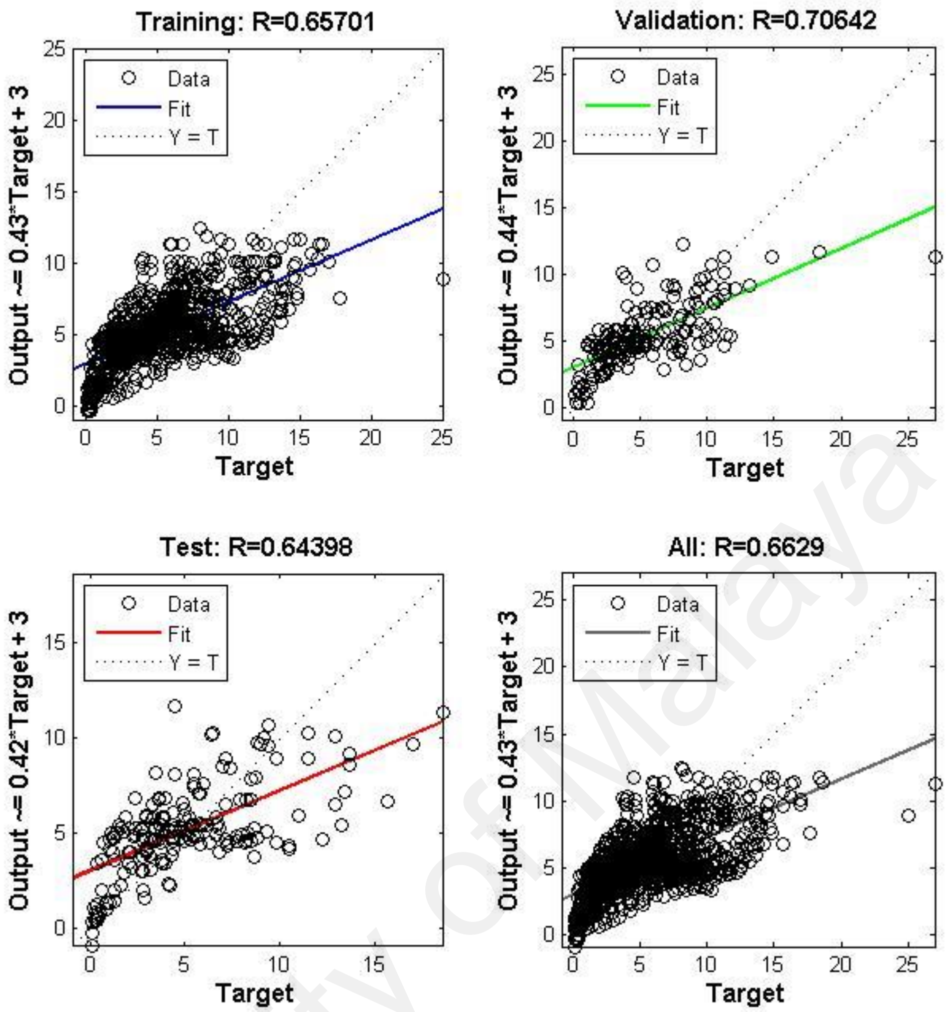

Figure 4.8: Plot of regression.

Figure 4.9 shows the results of training, validating and testing the network. The MSE value for training is 7.22826, validation is 7.10846, and testing is 7.40678.

Results

	Samples	MSE	R
Training:	840	7.22826e-0	6.57007e-1
Validation:	180	7.10846e-0	7.06421e-1
Testing:	180	7.40678e-0	6.43976e-1

Figure 4.9: Results of training, validating and testing the network.

4.3 Discussion

The main advantages of using NN model is based on its ability to define the relationship between several parameters. These parameters are not defined as merely parametric or analytical form. By using NN model for processing time estimation, it allows the precision machining company to improve their estimation accuracy. Most importantly, it helps in shorten the time taken for considering and deciding the processing time, and also ensures the stability of the time estimation result as the main worries of human estimation was underestimate or overestimate the necessary processing time.

4.4 Error Analysis

Error analysis is conducted to measure the accuracy and errors occurred in this project. 20 samples are randomly picked and calculated to determine the error analysis by using the formula stated below:

$$Error\ Analysis\ (\%) = \frac{|estimated\ output - \ actual\ output|}{actual\ output}\ x\ 100\%$$

The formula is used to express the difference between an experimented value and the actual or true value (Helmenstine, 2017). Based on the formula, the error of results of this project is analysed and tabulated in Table 4.1. Besides, the accuracy of the results are also calculated by using the formula below and included in Table 4.1 as well.

$$Accuracy\ (\%) = 100\% - Error$$

The values for inputs (material type, wire diameter, operation mode, number of cuts, and thickness of workpiece) and actual output are from the data that had collected earlier, while the value for estimated output is exported from the Neural Network tool using MATLAB.

Table 4.1: Error Analysis.

No.	Input					Output		Error (%)	Accuracy (%)
	Material Type	Wire Diameter (mm)	Mode	No. of Cuts	Thick-ness (mm)	Actual Time (min)	Estima-ted Time (min)		
1	STEEL	0.20	ACU	6	5	7.40	7.165	3.27	96.73
2	STEEL	0.20	ACU	6	50	3.80	3.869	1.79	98.21
3	STEEL	0.20	SPD	1	40	2.90	2.902	0.09	99.91
4	STEEL	0.20	SPD	1	50	2.30	2.247	2.35	97.65
5	TC	0.20	STD	5	5	9.20	8.859	3.85	96.15
6	TC	0.20	STD	6	30	4.40	4.544	3.18	96.82
7	TC	0.20	ACU	5	50	4.90	5.131	4.5	95.5
8	COPPER	0.20	STD	2	90	3.00	2.882	4.07	95.93
9	COPPER	0.20	STD	3	20	7.40	7.338	0.84	99.16
10	STEEL	0.25	STD	2	80	4.20	4.034	4.11	95.89
11	STEEL	0.25	STD	3	30	6.60	6.940	4.91	95.09
12	STEEL	0.25	ACU	1	10	9.50	9.457	0.45	99.55
13	STEEL	0.25	ACU	4	10	9.40	9.741	3.51	96.49
14	TC	0.25	STD	2	60	4.50	4.581	1.77	98.23
15	TC	0.25	STD	4	70	5.50	5.463	0.67	99.33
16	TC	0.25	ACU	5	60	5.50	5.548	0.87	99.13
17	COPPER	0.25	STD	1	5	11.20	11.009	1.73	98.27
18	COPPER	0.25	STD	5	30	6.60	6.717	1.75	98.25
19	STEEL	0.30	STD	2	125	4.00	4.013	0.34	99.66
20	ALU	0.30	STD	1	100	2.80	2.810	0.36	99.64

*Note: TC = Tungsten Carbide, STD = Standard mode, ACU = Accuracy priority mode, SPD = Speed Priority

From Table 4.1, the error of result from the NN model is varies from 0.09 to 4.91. On the other hand, the accuracy of the NN model estimation is around 95.09 to 99.91.

4.5 Summary

In summary, this chapter evaluate the performance of NN model which consists of three layers which are input layer, hidden layer and output layer. 1200 samples of 5 elements are used as input parameters and 1200 samples of 1 elements are used as expected output. Besides, the 1200 samples are divided into three, where 70% of the 1200 samples are trained with the network model, 15% of the 1200 samples are validated with the network model and 15% of the 1200 samples are tested with the network model by using the Neural Network Tool in MATLAB software. Four plots of graph are generated and analysed. By using formulas in Section 4.4, the error analysis and accuracy is done. From the 20 samples that had analysed, the error of result from the NN model is varies from 0.09 to 4.91, while the accuracy of the NN model estimation is around 95.09 to 99.91.

CHAPTER 5: CONCLUSION & FUTURE RECOMMENDATIONS

The chapter concluded the system for processing time estimation in precision machining industry using AI. The problems faced throughout this project are also analysed and future recommendation of this project is presented in this chapter.

5.1 Conclusion

This project propose a NN model training with Levenberg-Marquardt algorithm for processing time estimation of WEDM machining in precision machining industry. The proposed model was tested to identify how to the expected output and actual output meets the best performance of the model. The Levenberg-Marquardt algorithm is widely used in neural network fitting as it typically requires less time to train and test but requires more memory. When the generalization stops improving, the training of data is stopped as indicated by an increase in the mean square error of the validation samples. However, this training algorithm has a disadvantage which it is less effective for gradient calculation and when noise occurred. Estimation process developed by NN model provides a connection between data-driven method and expert-driven methods.

The implementation of this model provides a comparative image to tooling expert on how the network response shows varies from experience estimation in order to gain the estimation confidence. The major role in this estimation model is to provide an AI approach to do the task of estimation by implementing desired input parameters and machining data.

5.2 Recommendation for future project

It is recommended that the future works to be undertaken in the following aspects:

(a) Implementation and development of exclusively tailored model.

(b) Inclusive of wider perspective, for example job queuing time.

REFERENCES

Angius, A., Colledani, M., & Horvath, A. (2015). Lead-time oriented production control policies in two-machine production lines. *IFAC (International Federation of Automatic Control).* Elsevier Ltd.

Behrouznia, A., Azadeh, A., Pichka, K., Pazhoheshfar, P., & Saberi, M. (2011). Prediction of Manufacturing Lead Time Based on Adaptive Neuro-Fuzzy Inference System (ANFIS). *Innovations in Intelligent Systems and Applications (INISTA), 2011 International Symposium on* (pp. 16-18). Istanbul, Turkey: IEEE.

Berling, P., & Farvid, M. (2014). Lead-time investigationandestimationindivergentsupplychains. *Int. J.ProductionEconomics*, 177-189.

Conde, A., Sanchez, J., Plaza, S., Olivenza, M., & Ramos, J. (2015). An industrial system for estimation of workpiece height in WEDM. *The Manufacturing Engineering Society International Conference, MESIC 2015* (pp. 647-654). Elsevier Ltd.

Dasgupta, A., Dutta, P., Majumder, M., & Panja, S. (2017). Impact of change in machining time, MRR in WEDM modeled by ANN-RSM. *Journal of Industrial Engineering and Advances*.

Duffie, N., Bendul, J., Windt, K., & Knollmann, M. (2016). Lead time instability and its mitigation in production work systems. *CIRP Annals - Manufacturing Technology*, 437-440.

Eraslan, E. (2009). The Estimation of Product Standard Time by Artificial Neural Networks in the Molding Industry. *Mathematical Problems in Engineering*.

Experimental Errors and Error Analysis. (2017). Retrieved from Wolfram: https://reference.wolfram.com/applications/eda/ExperimentalErrorsAndErrorAnalysis.html

Florjanič, B., & Kuzman, K. (2012). Estimation of Time for Manufacturing of Injection Moulds Using Artificial Neural Networks-based Model.

Helmenstine, T. (2017, September 21). *Calculate Percent Error*. Retrieved from Science Notes: https://sciencenotes.org/calculate-percent-error/

Ionnou, G., & Dimitriou, S. (2012). Lead timeestimationinMRP/ERPformake-to-ordermanufacturingsystems. *Int. J.ProductionEconomics*, 551-563.

Ismail, R., Othman, Z., & Abu Bakar, A. (2009). Data Mining In Production Planning and Scheduling: A Review. *2009 2nd Conference on Data Mining and Optimization*, (pp. 154-159).

Jun, H.-B., Park, J.-Y., & Suh, H.-W. (2006). Lead Time Estimation Method for Complex Product Development Process. *Concurent Engineering*, 313-328.

Li, G., Kong, F., Lu, W., Yuan, Q., & Fang, M. (2007). The Neural-fuzzy Modeling and Genetic Optimization in WEDM. *2007 IEEE International Conference on Control and Automation*, (pp. 1440-1443). Guangzhou, China.

Mourtzis, D., Doukas, M., Fragou, K., Efthymiou, K., & Matzorou , V. (2014). Knowledge-based estimation of manufacturing lead time for complex engineered-to-order products. *The 47th CIRP Conference on Manufacturing Systems*, (pp. 499-504).

Mucientes, M., Vidal, J. C., Bugarin, A., & Lama, M. (2008). Processing times estimation in a manufacturing industry through genetic programming. *3rd International Workshop on Genetic and Evolving Fuzzy Systems*, (pp. 95-100). Witten-Bommerholz, Germany.

Okubo, H., Weng, J., Kaneko, R., Simizu, T., & Onari, H. (2000). Production Lead-time Estimation System Based on Neural Network. *Proceedings of Asia-Pacific Region of Decision Sciences.*

Ozturk, A., Kayaligil, S., & Ozdemirel, N. (2006). Manufacturing lead time estimation using data mining. *European Journal of Operational Research*, 683-700.

Pasini, A. (2015). Artificial neural networks for small dataset analysis. *Journal of Thoracic Disease*.

Perzyk, M., Kochanski, A., Kozlowski, J., Soroczynski, A., & Biemacki, R. (2013). Comparison of data mining tools for significance analysis of process parameters in applications to process fault diagnosis. (pp. 380-392). Elsevier Inc.

Susanto, S., Tanaya, P., & Soembagijo, A. (2012). Formulating Standard Product Lead Time at a Textile Factory using Artificial Neural Networks. *2012 International Conference on Uncertainty Reasoning and Knowledge Engineering*, (pp. 99-104).

Taylor, J. (1982). *An introduction to Error Analysis.* University Science Books.

Tehran, M., & Maleki, M. (2011). Artificial Neural Network Prosperities in Textile Applications. *Artificial Neural Networks - Industrial and Control Engineering Applications*, 35-64.

Vundavilli, P., Kumar, J., & Priyatham, C. (2012). Parameter Optimization of Wire Electric Discharge Machining Process using GA and PSO. *IEEE-International Conference On Advances In Engineering, Science And Management*, (pp. 180-185).

Wire EDM Fundamentals. (n.d.). Retrieved from Complete EDM Handbook: http://www.reliableedm.com/Complete%20EDM%20Handbook/Complete%20EDM%20Handbook_2.pdf

Zhang, H.-b., & Jia, Z.-x. (2009). Research in Reliability Modeling of WEDM Based on Neural Network. *International Conference on Measuring Technology and Mechatronics Automation*, (pp. 277-280).

APPENDIX A

```
% Solve an Input-Output Fitting problem with a Neural Network
%   data - input data.
%   MachiningSpeed0x28mm0x2Fmin0x29 - target data.

x = data';
t = MachiningSpeed0x28mm0x2Fmin0x29';

% Choose a Training Function
% For a list of all training functions type: help nntrain
% 'trainlm' is usually fastest.
% 'trainbr' takes longer but may be better for challenging problems.
% 'trainscg' uses less memory. NFTOOL falls back to this in low memory
situations.
trainFcn = 'trainlm';  % Levenberg-Marquardt

% Create a Fitting Network
hiddenLayerSize = 3;
net = fitnet(hiddenLayerSize,trainFcn);

% Choose Input and Output Pre/Post-Processing Functions
% For a list of all processing functions type: help nnprocess
net.input.processFcns = {'removeconstantrows','mapminmax'};
net.output.processFcns = {'removeconstantrows','mapminmax'};

% Setup Division of Data for Training, Validation, Testing
% For a list of all data division functions type: help nndivide
net.divideFcn = 'dividerand';  % Divide data randomly
net.divideMode = 'sample';  % Divide up every sample
net.divideParam.trainRatio = 70/100;
net.divideParam.valRatio = 15/100;
net.divideParam.testRatio = 15/100;

% Choose a Performance Function
% For a list of all performance functions type: help nnperformance
net.performFcn = 'mse';  % Mean squared error

% Choose Plot Functions
% For a list of all plot functions type: help nnplot
net.plotFcns = {'plotperform','plottrainstate','ploterrhist', ...
  'plotregression', 'plotfit'};

% Train the Network
[net,tr] = train(net,x,t);

% Test the Network
y = net(x);
e = gsubtract(t,y);
performance = perform(net,t,y)

% Recalculate Training, Validation and Test Performance
trainTargets = t .* tr.trainMask{1};
valTargets = t  .* tr.valMask{1};
testTargets = t  .* tr.testMask{1};
trainPerformance = perform(net,trainTargets,y)
valPerformance = perform(net,valTargets,y)
```

```
testPerformance = perform(net,testTargets,y)

% View the Network
view(net)

% Plots
% Uncomment these lines to enable various plots.
%figure, plotperform(tr)
%figure, plottrainstate(tr)
%figure, plotfit(net,x,t)
%figure, plotregression(t,y)
%figure, ploterrhist(e)

% Deployment
% Change the (false) values to (true) to enable the following code
blocks.
if (false)
  % Generate MATLAB function for neural network for application
deployment
  % in MATLAB scripts or with MATLAB Compiler and Builder tools, or
simply
  % to examine the calculations your trained neural network performs.
  genFunction(net,'myNeuralNetworkFunction');
  y = myNeuralNetworkFunction(x);
end
if (false)
  % Generate a matrix-only MATLAB function for neural network code
  % generation with MATLAB Coder tools.
  genFunction(net,'myNeuralNetworkFunction','MatrixOnly','yes');
  y = myNeuralNetworkFunction(x);
end
if (false)
  % Generate a Simulink diagram for simulation or deployment with.
  % Simulink Coder tools.
  gensim(net);
end



function [newData1] = importfile(fileToRead1)
%IMPORTFILE(FILETOREAD1)
% Import the file
sheetName='Sheet1';
[numbers, strings, raw] = xlsread(fileToRead1, sheetName);
if ~isempty(numbers)
    newData1.data =  numbers;
end
if ~isempty(strings)
    newData1.textdata =  strings;
end

if ~isempty(strings) && ~isempty(numbers)
    [strRows, strCols] = size(strings);
    [numRows, numCols] = size(numbers);
    likelyRow = size(raw,1) - numRows;
    % Break the data up into a new structure with one field per
column.
    if strCols == numCols && likelyRow > 0 && strRows >= likelyRow
        newData1.colheaders = strings(likelyRow, :);
    end
end
```

```
function [Y,Xf,Af] = myNeuralNetworkFunction(X,~,~)
%MYNEURALNETWORKFUNCTION neural network simulation function.
%
% [Y] = myNeuralNetworkFunction(X,~,~) takes these arguments:
%
%   X = 1xTS cell, 1 inputs over TS timsteps
%   Each X{1,ts} = 5xQ matrix, input #1 at timestep ts.
%
% and returns:
%   Y = 1xTS cell of 1 outputs over TS timesteps.
%   Each Y{1,ts} = 1xQ matrix, output #1 at timestep ts.
%
% where Q is number of samples (or series) and TS is the number of
timesteps.

  % ===== NEURAL NETWORK CONSTANTS =====

  % Input 1
  x1_step1_remove = [2 4];
  x1_step1_keep = [1 3 5];
  x1_step2_xoffset = [0.07;1;3];
  x1_step2_gain =
[8.69565217391304;0.333333333333333;0.00673400673400673];
  x1_step2_ymin = -1;

  % Layer 1
  b1 = [0.18637945985799104;2.3780160824224841;-4.304158387944943];
  IW1_1 = [-10.790248127368182 -2.7328791210512184 -
2.7643931952540148;0.31905020586323773 1.4002369359312619 -
0.2229107733232116;0.3327865748362866 -0.23972605078750661 -
3.2266699142046456];

  % Layer 2
  b2 = -0.66450742039266109;
  LW2_1 = [0.021347788055795461 1.754862932786631 1.7339668296882098];

  % Output 1
  y1_step1_ymin = -1;
  y1_step1_gain = 0.0743494423791822;
  y1_step1_xoffset = 0.1;

  % ===== SIMULATION ========

  % Format Input Arguments
  isCellX = iscell(X);
  if ~isCellX, X = {X}; end;

  % Dimensions
  TS = size(X,2); % timesteps
  if ~isempty(X)
    Q = size(X{1},2); % samples/series
  else
    Q = 0;
  end

  % Allocate Outputs
  Y = cell(1,TS);

  % Time loop
  for ts=1:TS
```

```
    % Input 1
    temp =
removeconstantrows_apply(X{1,ts},x1_step1_keep,x1_step1_remove);
    Xp1 =
mapminmax_apply(temp,x1_step2_gain,x1_step2_xoffset,x1_step2_ymin);

    % Layer 1
    a1 = tansig_apply(repmat(b1,1,Q) + IW1_1*Xp1);

    % Layer 2
    a2 = repmat(b2,1,Q) + LW2_1*a1;

    % Output 1
    Y{1,ts} =
mapminmax_reverse(a2,y1_step1_gain,y1_step1_xoffset,y1_step1_ymin);
  end

  % Final Delay States
  Xf = cell(1,0);
  Af = cell(2,0);

  % Format Output Arguments
  if ~isCellX, Y = cell2mat(Y); end
end

% ===== MODULE FUNCTIONS ========

% Map Minimum and Maximum Input Processing Function
function y =
mapminmax_apply(x,settings_gain,settings_xoffset,settings_ymin)
  y = bsxfun(@minus,x,settings_xoffset);
  y = bsxfun(@times,y,settings_gain);
  y = bsxfun(@plus,y,settings_ymin);
end

% Remove Constants Input Processing Function
function y = removeconstantrows_apply(x,settings_keep,settings_remove)
  if isempty(settings_remove)
    y = x;
  else
    y = x(settings_keep,:);
  end
end

% Sigmoid Symmetric Transfer Function
function a = tansig_apply(n)
  a = 2 ./ (1 + exp(-2*n)) - 1;
end

% Map Minimum and Maximum Output Reverse-Processing Function
function x =
mapminmax_reverse(y,settings_gain,settings_xoffset,settings_ymin)
  x = bsxfun(@minus,y,settings_ymin);
  x = bsxfun(@rdivide,x,settings_gain);
  x = bsxfun(@plus,x,settings_xoffset);
end
```

```
function [y1] = myNeuralNetworkFunction(x1)
%MYNEURALNETWORKFUNCTION neural network simulation function.
%
% [y1] = myNeuralNetworkFunction(x1) takes these arguments:
%   x = 5xQ matrix, input #1
% and returns:
%   y = 1xQ matrix, output #1
% where Q is the number of samples.

%#ok<*RPMT0>

  % ===== NEURAL NETWORK CONSTANTS =====

  % Input 1
  x1_step1_remove = [2 4];
  x1_step1_keep = [1 3 5];
  x1_step2_xoffset = [0.07;1;3];
  x1_step2_gain =
[8.69565217391304;0.333333333333333;0.00673400673400673];
  x1_step2_ymin = -1;

  % Layer 1
  b1 = [0.18637945985799104;2.3780160824224841;-4.304158387944943];
  IW1_1 = [-10.790248127368182 -2.7328791210512184 -
2.7643931952540148;0.31905020586323773 1.4002369359312619 -
0.2229107733232116;0.3327865748362866 -0.23972605078750661 -
3.2266699142046456];

  % Layer 2
  b2 = -0.66450742039266109;
  LW2_1 = [0.021347788055795461 1.754862932786631 1.7339668296882098];

  % Output 1
  y1_step1_ymin = -1;
  y1_step1_gain = 0.0743494423791822;
  y1_step1_xoffset = 0.1;

  % ===== SIMULATION ========

  % Dimensions
  Q = size(x1,2); % samples

  % Input 1
  xp1 = removeconstantrows_apply(x1,x1_step1_keep,x1_step1_remove);
  xp1 =
mapminmax_apply(xp1,x1_step2_gain,x1_step2_xoffset,x1_step2_ymin);

  % Layer 1
  a1 = tansig_apply(repmat(b1,1,Q) + IW1_1*xp1);

  % Layer 2
  a2 = repmat(b2,1,Q) + LW2_1*a1;

  % Output 1
  y1 =
mapminmax_reverse(a2,y1_step1_gain,y1_step1_xoffset,y1_step1_ymin);
end

% ===== MODULE FUNCTIONS ========
```

```
% Map Minimum and Maximum Input Processing Function
function y =
mapminmax_apply(x,settings_gain,settings_xoffset,settings_ymin)
  y = bsxfun(@minus,x,settings_xoffset);
  y = bsxfun(@times,y,settings_gain);
  y = bsxfun(@plus,y,settings_ymin);
end

% Remove Constants Input Processing Function
function y = removeconstantrows_apply(x,settings_keep,settings_remove)
  if isempty(settings_remove)
    y = x;
  else
    y = x(settings_keep,:);
  end
end

% Sigmoid Symmetric Transfer Function
function a = tansig_apply(n)
  a = 2 ./ (1 + exp(-2*n)) - 1;
end

% Map Minimum and Maximum Output Reverse-Processing Function
function x =
mapminmax_reverse(y,settings_gain,settings_xoffset,settings_ymin)
  x = bsxfun(@minus,y,settings_ymin);
  x = bsxfun(@rdivide,x,settings_gain);
  x = bsxfun(@plus,x,settings_xoffset);
end
```