Development of Open-sourced, Self-executable and GUI-based Application Tool Q-Check for Quality Prediction of Resistance Spot Weld using Artificial Neural Network

Muhamad Aiman Raziq
Universiti Teknologi MARA
Yupiter HP Manurung (yupiter.manurung@uitm.edu.my)
Universiti Teknologi MARA
Suhaila Abd Halim
Universiti Teknologi MARA
Cheng Yee Low
Tun Hussein Onn University of Malaysia
Muhammad Saufy Rohmad
Universiti Teknologi MARA
John RC Dizon
Bataan Peninsula State University
Vladimir S Kachinskyi
E.O. Paton Electric Welding Institute

Research Article

Keywords: Resistance spot welding, Artificial Neural Network, Graphical User Interface, Python, Backpropagation

Posted Date: March 14th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-1408226/v1

Development of Open-sourced, Self-executable and GUI-based Application Tool *Q-Check* for Quality Prediction of Resistance Spot Weld using Artificial Neural Network

Muhamad Aiman Raziq[1,2], Yupiter HP Manurung[1,2,*], Suhaila Abd Halim[3], Cheng Yee Low[4], Muhammad Saufy Rohmad[1,5], John RC Dizon[6] and Vladimir S Kachinskyi[7]

[1]Smart Manufacturing Research Institute (SMRI), Universiti Teknologi MARA (UiTM), 40450 Shah Alam, Selangor, Malaysia

[2]School of Mechanical Engineering, Universiti Teknologi MARA (UiTM), 40450 Shah Alam, Selangor, Malaysia

[3]Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA (UiTM), 40450 Shah Alam, Selangor, Malaysia

[4]Innovationlabs.my, Universiti Tun Hussein Onn (UTHM), Batu Pahat, Malaysia

[5]School of Electrical Engineering, Universiti Teknologi MARA (UiTM) 40450 Shah Alam, Selangor, Malaysia

[6]Design, Research, Extension in Additive Manufacturing, Advanced Manufacturing (DR3AM) Center, Bataan Peninsula State University, City of Belanga, Philippines

[7]E.O. Paton Electric Welding Institute, Kiev, Ukraine

**Corresponding E-mail:*
yupiter.manurung@uitm.edu.my

Abstract:

Optimizing Resistance spot welding (RSW), often used as a time and cost-effective process in many industrial sectors, is very time-consuming due to the obscurity inherent within process with numerous interconnected welding parameters. Small changes in values will give effect to the quality of welds which actually can be easily analysed using application tool. Unfortunately, existing software to optimize the parameters are expensive, licensed and inflexible which makes small industries and research centres refused to acquire. In this study, application tool using open-sourced and customized algorithm based on artificial neural networks (ANN) was developed to enable better, fast, cheap and practical predictions of major parameters such as welding time, current and electrode force on tensile shear load bearing capacity (TSLBC) and weld quality classifications (WQC). A supervised learning algorithm implemented in standard backpropagation neural network gradient descent (GD), stochastic gradient descent (SGD) and Levenberg-Marquardt (LM) was constructed using TensorFlow with Spyder IDE in python language. All the display and calculation processes are developed and compiled in the form of graphical user interface (GUI) using Qt designer and PyQt5. The standalone capability was created with customized icon by using Pyinstaller to enable a self-execution in windows platform. Results showed that this low-cost application tool Q-Check based on ANN models can predict with 80% training and 20% test set on TSLBC with an accuracy of 87.220%, 92.865% and 93.670% for GD, SGD and LM algorithms respectively while on WQC 62.5% for GD and 75% for both SGD and LM. It is also expected that tool with flexible GUI can be widely used and enhanced by practitioner with minimum knowledge in the domain.

Keywords: Resistance spot welding; Artificial Neural Network; Graphical User Interface; Python; Backpropagation

1. Introduction

Resistance spot welding (RSW) is widely employed in manufacturing sectors for sheet metal joining because of its high speed and scalability to automation in high-volume, high-rate production. Additionally, RSW is a type of resistance welding that is frequently utilised in place of rivets, screws, soldering, and brazing on a range of goods. The approach is extensively used to combine components made of low carbon steel. High-strength low- alloy steel, stainless steel, nickel, aluminium, titanium and copper alloys are also spot welded commercially [1].

Two or more metal sheets are bonded together using resistance spot welding by putting an electric current across them. The current is transmitted through electrodes that are applied onto the metal surfaces of the components to be welded. The heat generated by the running current melts the metal, resulting in the formation of a welding spot. The amplitude and length of the current influence the amount of heat generated at the location. The current and duration are precisely regulated and matched to the material, thickness, and kind of electrodes. As with any other type of welding, the quality of the joint is specifically correlated to the welding input parameters. A common problem is choosing and implementing the process input parameters to achieve a well-welded joint with the appropriate strength [2].

The problem of choosing the optimal input parameters gives motivation to this study. The objectives related to this study are to develop ANN algorithm and GUI-based application tool to predict the parameters of RSW. The ANN consists of mathematical structures that emulate the biological nervous system's behaviour. It maps non-linear and dynamic designs with parallel, distributed, and adaptive computation [3].

Hamedi et al. [4] standardized three major processes of input parameters: welding current, welding time, and electrode force for spot welding of the body parts. The influence of these parameters on sub-assembly deformation were tested experimentally. Neural networks and multi-objective genetic algorithms were applied to select the optimal welding parameter values that generated the least dimensional variance values in the sub-assemblies.

The ANN can also be referred as a multilayer perceptron (MLP). According to Haruna Chiroma et al. [5], there are many learning algorithms in MLP. One commonly used is the backpropagation algorithm or the backpropagation neural network (BPNN).

However, in BPNN, the GD algorithm requires extended time to converge. Thus, the Levenberg-Marquardt (LM) algorithm is used for accelerating convergence [6]. The supervised learning algorithms used in this study are the BPNN or also known as standard backpropagation neural network algorithm which uses the GD, stochastic gradient descent (SGD), and the LM algorithm. Numerous past research used BPNN to optimize parameter, predict fatigue life and predict weld quality of RSW [6]–[12]. A graphical user interface (GUI) is used to make the ANN algorithm in graphical form that is easy to use by end user to make predictions [13], [14]. A user friendly and easy to use GUI is beneficial to be developed as to effectively deal with the developed ANN algorithm.

In this study, the BPNN algorithms are implemented to optimise the parameters in RSW. These techniques are adapted for the prediction of TSLBC value and classification of quality for resistance spot welding. It is followed by developing an application tool consisting of several GUIs that simplify the computer environment. The tool can be used by other researcher and industry practitioner which enable them to easily use and gather experiment results using the developed tool without prior knowledge in the domain.

2. Experimental Procedure and Data Collection

The experimental investigation stem of this paper is from the previous literature [3]. The chemical composition of the austenitic stainless steel SS304 sheets includes material properties, equipment, test specimen and RSW experimental input and output parameters. The chemical composition of the 304 ASS sheets is shown in Table 1.

Table 1: Chemical composition of SS304 (wt%)

C	Cr	Ni	Si	Mn	Mo	Al	Co
0.08	18.03	8.74	0.426	1.153	0.36	0.003	0.17
Cu	Nb	Ti	V	W	Fe	P	S
0.39	0.02	0.004	0.05	0.03	70.48	0.019	0.002

The SS304 sheets were welded in 50 Hz single-phase equipment with water-cooled truncated cone RWMA Group A Class 2 electrodes with a face diameter of 4.5 mm.

The ultrasonic spot-welding research transducer combines a captive water column delay and a replaceable rubber membrane to ensure a secure coupling to the weld surfaces. The transducer was 20 MHz in frequency and 4.5 mm in diameter. The sheet measured 0.8 millimetres in diameter. The physical properties of the SS304 sheets that consisted of yield strength, tensile strength, total elongation, and microhardness are summarized in Table 2.

Table 2: Mechanical properties of SS304

Yield strength (MPa)	Tensile strength (MPa)	Total elongation (%)	Microhardness (HV, 100 g)
290	675	70	162

Vickers microhardness testing was performed with a 100 g load on a Matsuzawa Seiki MXT 70 tester. Shimadzu UH-F500 kNA universal testing machine was used to conduct tensile shear checks.

The RSW process employed a weld interval between 12 and 2 cycles, with a one-cycle step decrement. The welding current varied from 6.5 to 1.5kA RMS, with a phase decreased of 0.5kA RMS and two electrode force values of 1000N and 1500N. Tensile shear specimens were prepared in accordance to ISO 14273. The dimension of test specimen is shown in Figure 1.

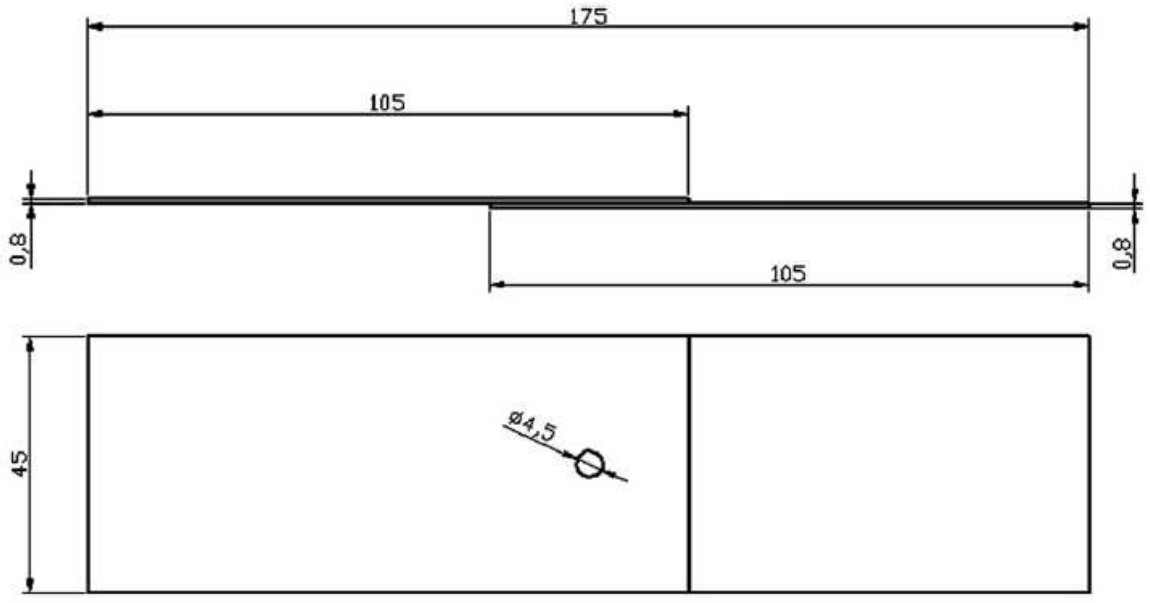


Figure 1: Dimensions of spot-welded tensile shear test specimens (mm)

Based on RSW dataset extracted from [3], there were 36 data available with three inputs as corresponding outputs of TSLBV value (kN) and the quality level. The three input parameters were weld time (cycle), weld current (kA) and electrode force (N). The class value was either Bad, Good, or Worst, with a TSLBC Value for each data point. Table 3 shows the used dataset of RSW.

Table 3: Data Collection of RSW dataset

Weld Time (cycle)	Weld Current (kA)	Electrode Force (N)	TSLBC Value (kN)	Quality level
7	6.16	1500	6.91	GOOD
12	5.64	1500	6.36	GOOD
3	5.51	1500	5.5	BAD
5	4.98	1500	5.75	BAD
8	4.46	1500	5.8	BAD
9	3.93	1500	5.14	BAD
3	4.06	1500	4.01	BAD
3	3.54	1500	3.58	BAD
9	2.88	1500	4.49	BAD
12	6.56	1500	6.5	GOOD
3	6.56	1500	5.55	BAD
8	2.49	1500	3.76	BAD
4	2.49	1500	2.95	BAD
12	1.96	1500	3.42	BAD
11	6.56	1000	6.78	GOOD
7	6.42	1000	6.68	GOOD
9	5.9	1000	6.38	GOOD
12	5.51	1000	6.06	GOOD
5	5.64	1000	6.28	GOOD
9	4.98	1000	6.11	GOOD
6	4.98	1000	5.92	BAD
12	4.59	1000	5.92	BAD
7	4.46	1000	5.98	GOOD

11	3.93	1000	5.17	BAD
8	3.93	1000	5.56	BAD
4	4.06	1000	4.67	BAD
10	3.41	1000	4.93	BAD
4	3.54	1000	4.21	BAD
7	3.01	1000	4.21	BAD
11	1.96	1000	3.63	BAD
3	1.96	1000	2.29	BAD
10	2.49	1000	3.18	BAD
2	2.49	1000	2.7	BAD
5	1.57	1000	1.67	WORST
11	1.7	1500	2.1	BAD
3	1.44	1500	1.36	WORST

Based on literature [3], the quality level was divided into good, bad and worst depending on the TSLBC value. The quality level of the RSW joints was classified based on the TLBC value depending on Table 4.

Table 4: The quality level of RSW joint

TSLBC of the RSW joint	Quality level
5.93kN ≤ TSLBC	Good
1.83kN ≤ TSLBC < 5.93kN	Bad
TSLBC < 1.89kN	Worst

In this study, ANN was applied to predict the TSLBC value using linear regression while multiclassification was used to classify the quality level.

3. Artificial Neural Network with Python

The ANNs are mathematical models that emulate the behaviour of the human nervous system and capable of parallel, distributed and adaptive computation which enables them to map non-linear and complex structures [15]. The ANN consists of a set of connected units generally known as neurons or nodes. The neurons are arranged in

a layer, whereby the signal is transmitted between one neuron and another in each connection. The receiving neuron processes the signal and signals it to its associated downstream neurons within the network.

There are three main types of ANN layers: input, hidden, and output. The input layer receives different information forms, and the data flow from the input layer to the hidden layer to the output layer. Most neural networks are completely interconnected between layers, and the links are weighted. The higher the weight number, the greater the effect of one unit on another. The network learns more about the data as the data move through each unit. As the data transfer from input to output, the resulting output depends on each weight's interconnection inside the network.

The process begins with an input layer consisting of the neuron of each data function to be trained. All data are then multiplied by the weight value and all are summed up by a bias. Weight and bias are, at first, randomly defined. The weighted number is then moved to an activation function that transforms the sum into a result depending on the activation function between a certain lower and upper limit [16].

Multilayer perceptron feed-forward ANN is used to solve complex predictive modelling problems. They usually have an input layer, one or two hidden layers, and a single output layer. Neurons are computational units in the hidden layers that conduct non-linear mapping between inputs and outputs. This study focuses on multilayer perceptron neural network. It takes a set of 3 input values which are electrode force, welding current, and welding time. The prediction on the numerical output is TSLBC and categorical output is the weld quality: worst, bad, good. Figure 2 shows the multilayer perceptron feed-forward network used in predicting TSLBC of RSW.

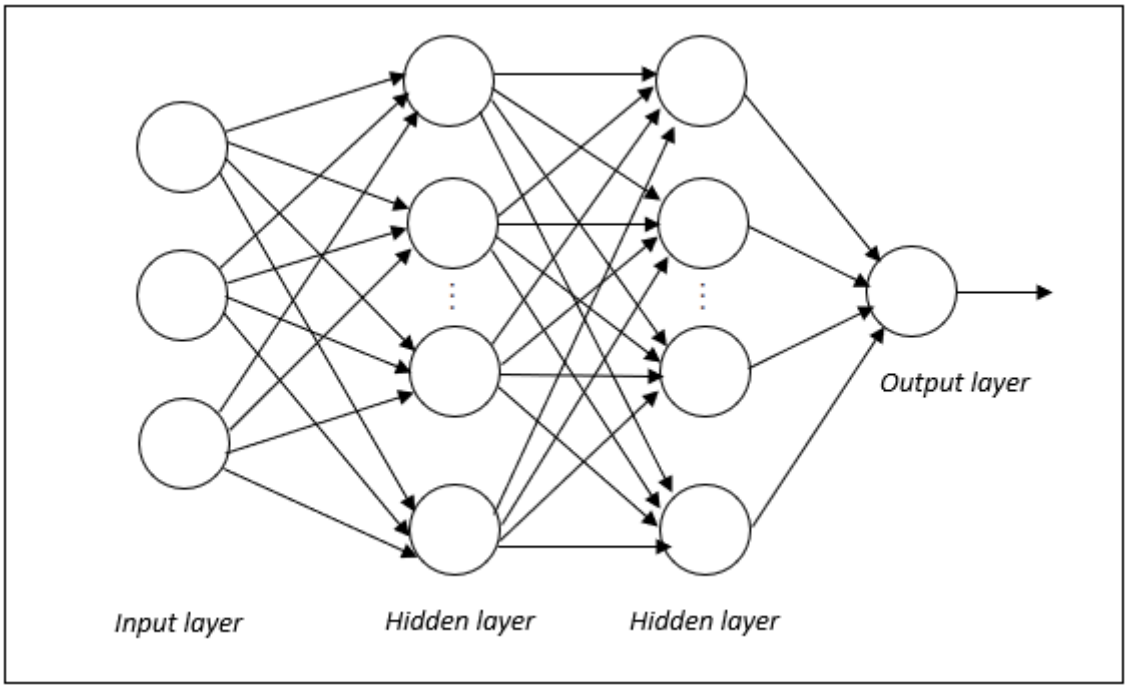


Figure 2: Multilayer perceptron feed-forward network

3.1 Optimizer and Training Algorithm for Back Propagation Neural Network

Training a multilayer perceptron is the process of determining the individual weights' values so that the network can correctly overcome the relationship for modelling. Like other probabilistic neural networks, this neural network needs only a fraction of the training samples. The BPNN is commonly applied to multilayer feed-forward ANN due to its ability to "learn" system properties via nonlinear mapping [17].

The BPNN is an algorithm using gradient descents to measure the loss function. It takes the derivative of the objective function gradient to the weights in the model. The reverse propagation method is carried out using calculus to determine the loss derivative for each weight using the chain rule. The gradient function can be expressed as Equation (3.1).

$$\frac{d}{d\theta_j} J(\theta) = -\frac{2}{n}\sum_{i=1}^{n}(y_t - y_{pred})\, f'(\sum_{i=1}^{k} w_j x_i^l + b)\, x_{w_j}^l \tag{3.1}$$

$\frac{d}{d\theta_j} J(\theta)$ = the gradient/derivative of the objective function J with respect to jth weight and bias

l = layer (l = 0 is input layer)

w_j = j[th] weight

$x^l_{w_j}$ = input of data at j[th] weight for l [th] layer

y_t = true value

y_{pred} = predicted value

n = Number of datapoint

x^l_i = Input data at i[th] input for l [th] layer

k = Total neuron at certain layer

The SGD is also a gradient descent type, using 1 example cost gradient at each iteration instead of using the sum of ALL examples cost gradient. It regularly updates the model parameter. The benefit of the algorithm is to update model parameters so that it converges in less time and needs less memory as the loss values do not need to be stored [18]. The gradient function for SGD can be expressed as Equation (3.2).

$$\frac{d}{d\theta_j} J(\theta) = -2(y_t - y_{pred}) f'(\sum) x^l_{w_j} \tag{3.2}$$

An analytical parameter used to evaluate the network's performance is an objective function that refers to loss function. Loss is defined as the difference between the expected and actual performance values. The ultimate goal is to eliminate error to maximize the accuracy of the model's forecasts. Mean square error (MSE) and cross-entropy are two objective functions commonly used.

Mean squared error (*MSE*) is one of the mostly used objective functions, measured as the average squared difference between predictions and actual values. It is only concerned with the average magnitude of error, irrespective of its direction. The *MSE* can be represented as Equation (3.3).

$$MSE = \frac{1}{PN} \sum_p \sum_n (y_t - y_{pred})^2 \tag{3.3}$$

P is a number of training pattern or data point and N is a number of output nodes or neurons. If the number of training pattern and output node is 1 and 2, then P = 1, N = 2, respectively.

Categorical cross-entropy is often called Softmax loss activation function plus cross-entropy loss. The categorical entropy objective function is used when there are two or more label classes. The categorical cross-entropy can be represented as Equation (3.4).

$$y_{pred_j} = \frac{e^{(x^l_{w_j})}}{\sum_{i=1}^{k} e^{(x^l_i)}} \qquad CE = -\frac{1}{n}\sum_{i=1}^{C} y_t \log\left(y_{pred}\right) \tag{3.4}$$

Weight is used to minimize the prediction error. After the gradient of the objective function achieves global minima, it will maximize the objective function. Hence, the weight needs to be updated. It is a process where the current weight is subtracted with the gradient of the objective function multiplied by the learning rate. The update rule of BPNN is expressed as Equation (3.5)

$$w_{j+1} = w_j - n\frac{d}{dw_j}J\left(w\right) \qquad b_{j+1} = b_j - n\frac{d}{db_j}J\left(b\right) \tag{3.5}$$

w_j is a current weight in a neural network. w_{j+1} is a new weight in a neural network. b_j is a current bias weight in a neural network. b_{j+1} is a new bias weight in a neural network. η is a learning rate.

Another well-known optimizer is the Levenberg Marquardt (LM) which is the combination of a gradient descent method and the Gauss-Newton method designed to minimize a non-linear function [19]. In ANN, LM is suitable for training small and medium-sized problems. The Gauss-Newton approximates the Hessian matrix, *H* expressed as Equation (3.6):

$$H = J^T J \tag{3.6}$$

where *J* is the Jacobian matrix, and the LM introduces another approximation to the Hessian matrix as expressed in Equation (3.7):

$$H = J^T J + \mu I \tag{3.7}$$

where μ is a damping parameter which is always a positive number, and *I* is the identity matrix. The update rule of LM is expressed as Equation (3.8).

$$w_{j+1} = w_j - \left[J^T \left(w_j \right) J \left(w_j \right) + \mu_j I \right]^{-1} J^T \left(w_j \right) v \left(w_j \right) \tag{3.8}$$

where w_{j+1} is a new weight calculated as gradient function, w_j is the current weight using the Newton algorithm, *j* is the iteration index, μ_j is a damping parameter and v is an error vector.

Differ from GD and SGD which are available in Keras libraries, the Python code for LM algorithm is developed and embedded in the application tool. The LM algorithm implemented in ANN models originated from tf.keras.Model. Referring to the lines of codes in Figure 3, the process is comparable on how the models are often trained in Keras. The distinction is that the fit function is invoked on a ModelWrapper class rather than on the model class directly.

```
import levenberg_marquardt as lm
model_wrapper = lm.ModelWrapper(model)
model_wrapper.compile(
    optimizer=tf.keras.optimizers.SGD(learning_rate=0.1),
    loss=lm.SparseCategoricalCrossentropy(from_logits=True),
    metrics=['accuracy'])
model_wrapper.fit(train_dataset, epochs=5)
```

Figure 3: Implemented LM algorithm using Python

As the algorithm shown, the LM optimizer is an advancement of GD and Newton method which can be seen in the separately self-developed model known as *model_wrapper.* This implementation of the LM algorithm refers to the previous work of [20].

The LM approaches gradient descent when the μ is very large, and it approaches the Gauss-Newton algorithm when it μ is very small or nearly zero. The LM is fast, stable, and more robust than the Gauss-Newton as it converges well. But it is a bit slower than Gauss-Newton. The LM converges faster than the gradient descent method, and it requires much memory compared to Gauss-Newton and gradient descent [6].

3.2 Activation Functions and ANN Structure

Activation function, also known as transfer function, is used in neural network to calculate and update the weighted sum and bias.

The activation function for the hidden layers is the Rectified linear units (ReLu) function. The ReLu activation function can be defined as Equation (3.9).

$$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases} \tag{3.9}$$

There are two types of activation functions used in this paper for the output layer since there are two different types of output from the dataset, namely scalar for TSLBC and categorical variable for weld quality. The activation function used to predict the TSLBC of RSW is the Linear activation function as defined in Equation (3.10).

$$f(x) = mx \tag{3.10}$$

In addition, the activation function used to predict the quality of weld is Softmax as defined in Equation (3.11).

$$f(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}} \tag{3.11}$$

where x_i the ith element of input values and j is the total number of input values.

Different activation functions are applied for both hidden and output layers due to their capability and type of prediction.

Table 5 summarises the activation functions of the ANN model used in this study. For both models, ReLu is applied in hidden layer. Linear and Softmax are used in output layer as activation function for linear regression and multi-classification, respectively.

Table. 5: Activation function used for the ANN model

Type of Model	Hidden layer	Output layer
Linear regression	ReLu	Linear
Multi-classification	ReLu	Softmax

The ANN inputs are vectors with three components, one for each RSW parameter of WT, WC, and EF which are used to predict TSLBC and weld quality. Hence, the ANN input layer contains three neurons that represent the three inputs. A supervised learning mechanism is used to learn from the labelled training data. This indicates that each input must be followed by a target. This target is numerical and categorical which contain the value of TSLBC and welding quality of Good, Bad, and Worst obtained using the respective input's welding parameters. As a result, the output layer of the ANN contains one and three neurons for TSLBC value and welding quality, respectively. The number of hidden layers used in this study are in linear regression and multi-classification, while the number of neurons inside the hidden layer is 39 by using the approach as suggested by Pashazadeh et al. [21]. Table 6 summarizes the ANN structures used in this study.

Table 6: ANN structure for numerical and categorical output

Type of Model	Numerical Output	Categorical Output
Linear regression	3-39-1	3-39-3
Multi-classification	3-39-39-1	3-39-39-3

Referring to Table 3, the entire data set namely 36 input/target pairs corresponding to the 36 RSW data is randomly split into two subsets which are:

- **Training subset** (80%) with 26 input/target pairs for training the ANN. The synaptic weights (each link between neurons is associated with a synaptic

weight) are modified periodically to minimize the error between the experimental outputs and their respective targets.

- **Testing subset** (20%) with 8 input/target pairs to evaluate the accuracy of the ANN after the training process.

The whole processes in ANN algorithms are combined to develop an application tool that consists of several GUIs.

4. GUI development using Qt designer and Spyder IDE with self-execution

The GUI brings simplicity to run any computer program by pressing a few buttons. Different development tools such as PyQt5 empower engineers and scientists to make GUI and stand-alone applications for the code to be run efficiently. The development of a GUI can be divided into two phases which are interface design and backend development [22]. The process starts by designing a GUI using the Qt designer tool, which comes up with the PyQt5 module. Simple drag and drop interface, a GUI interface can be quickly built without having to write the code. Figure 4 shows the general page of Qt designer before the design process starts.

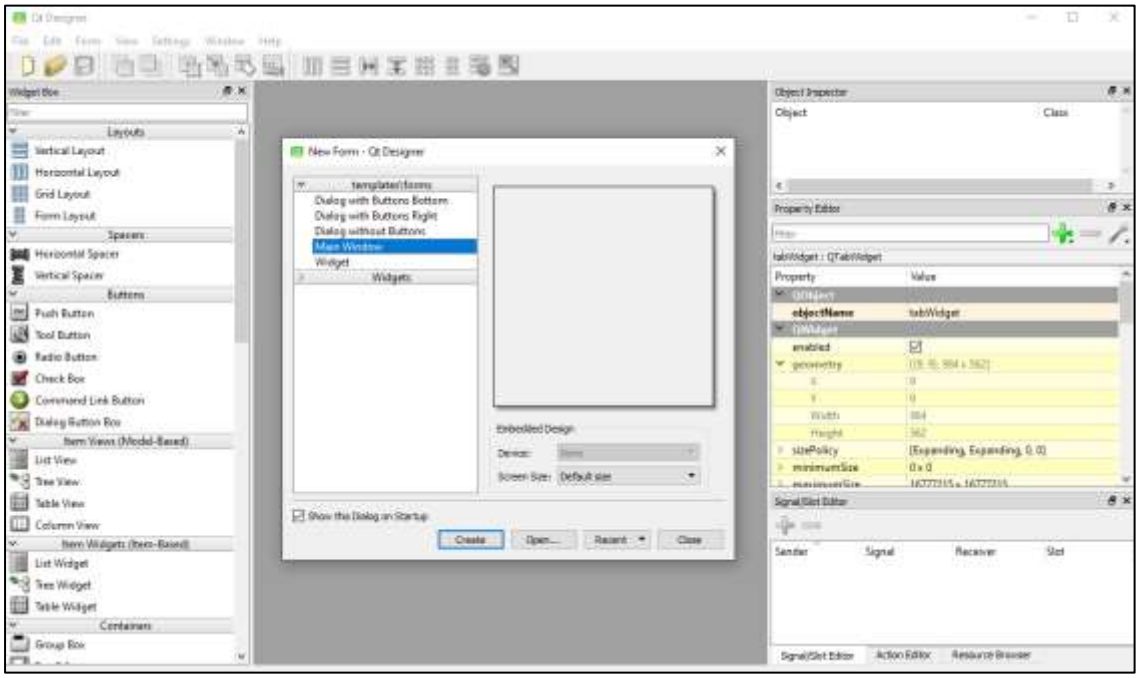


Figure 4: General page of Qt designer

The Qt designer is used to design the GUI while the back-end development that consists of the ANN program is coded using the Spyder IDE. Spyder is a powerful scientific environment written in Python designed by scientists, engineers, and data analysts. It features a unique combination of advanced editing, analysis and debugging. The application tool is developed using a combination of open-source libraries. The libraries are installed using the Anaconda, a distributor of python programming language, which also simplifies package deployment and management. Figure 5 shows the main page of Spyder IDE with its main components. It consists of Code editor, Variable explorer, Help, Plots, Files, IPython console, and History.

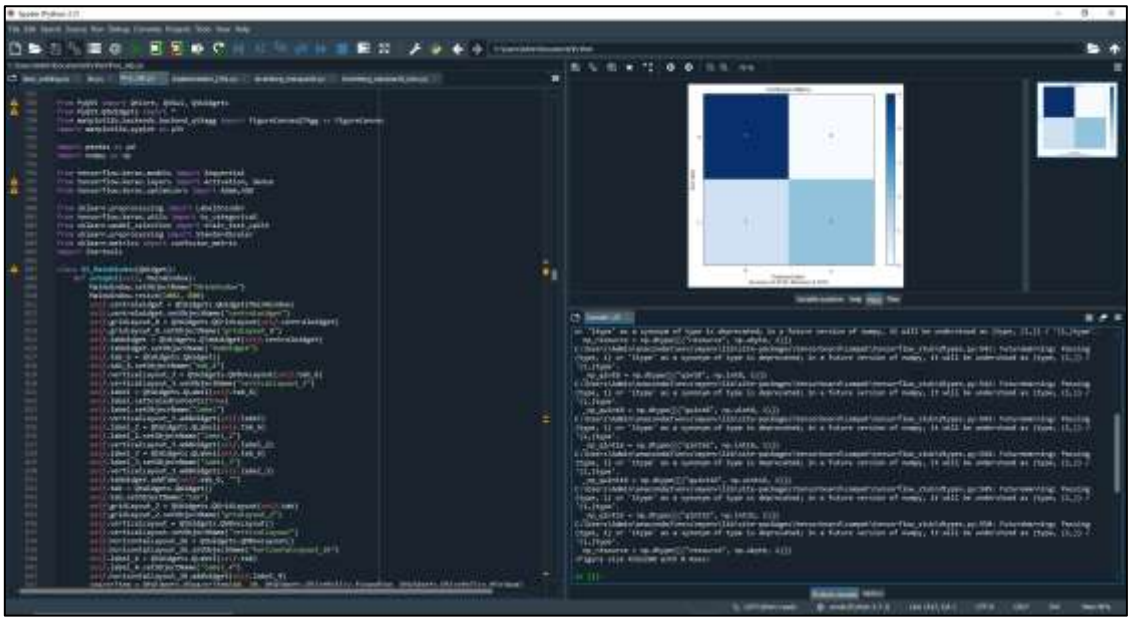


Figure 5: The main interface of Spyder IDE

The use of PyQt in Spyder enables the GUI layout design in Qt designer to interact with each other. In this study, the application tool named *Q-Check* is developed using Qt designer to design the GUI layout and Spyder IDE as the back-end development. The *Q-Check* consists of 6 Tabs in which each tab has its own function to complete the prediction process. Figure 6 demonstrates the general functionalities for Tab 1 until Tab 6 in the *Q-Check* application tool.

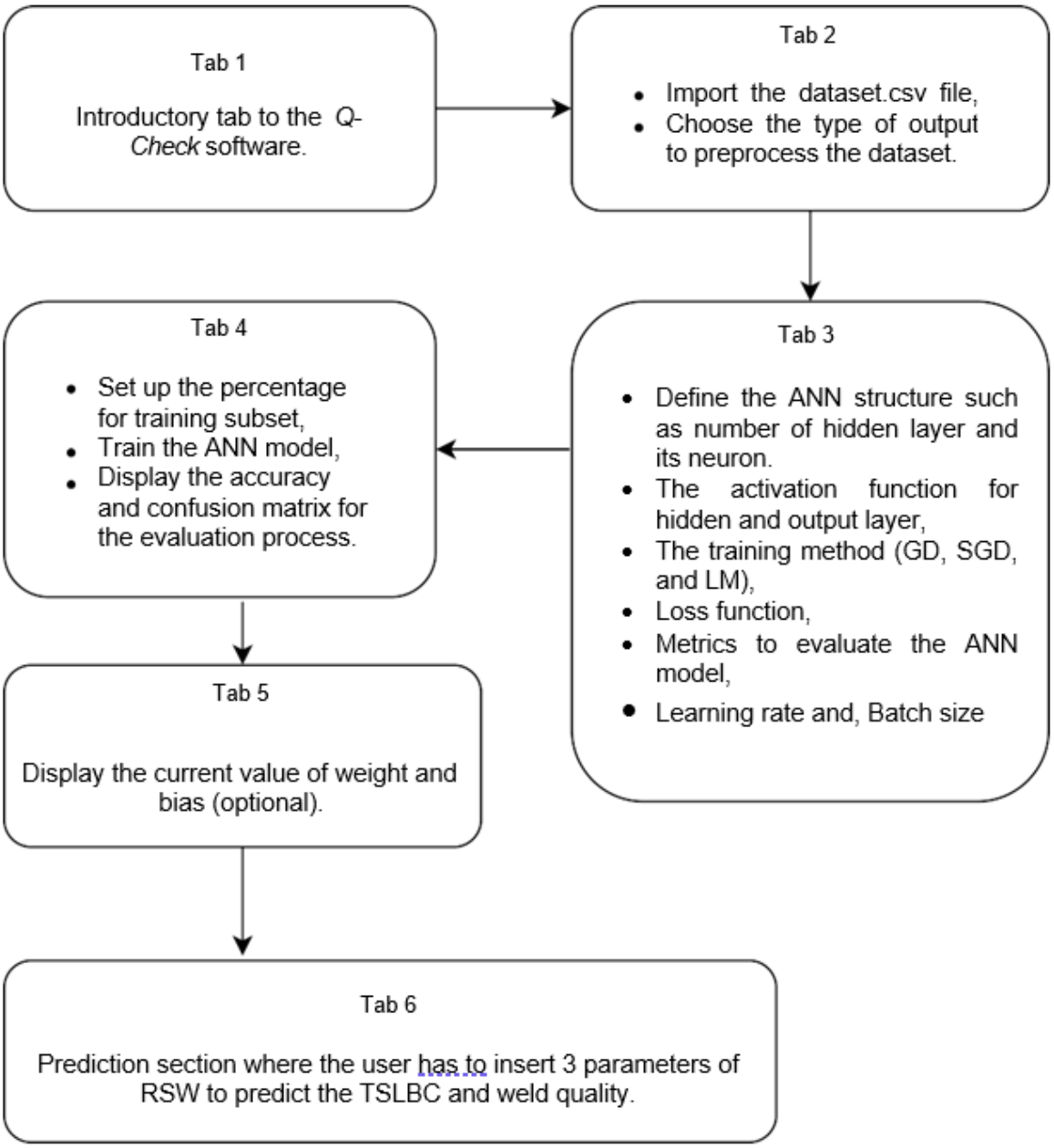


Figure 6: Process flow performed in the *Q-Check* application tool

The layout designs of *Q-Check* with 6 tabs are shown in Figure 7 to Figure 12. The first tab introduces the front page of the application tools as shown in Figure 7.

Figure 7: Introductory tab of *Q-Check* application tool as Tab 1

Figure 8 shows the second tab where the user is able to import the input file and pre-process dataset in the input file for further process. In pre-processing, the user can select which type of true value to be pre-processed before setting up the ANN model.

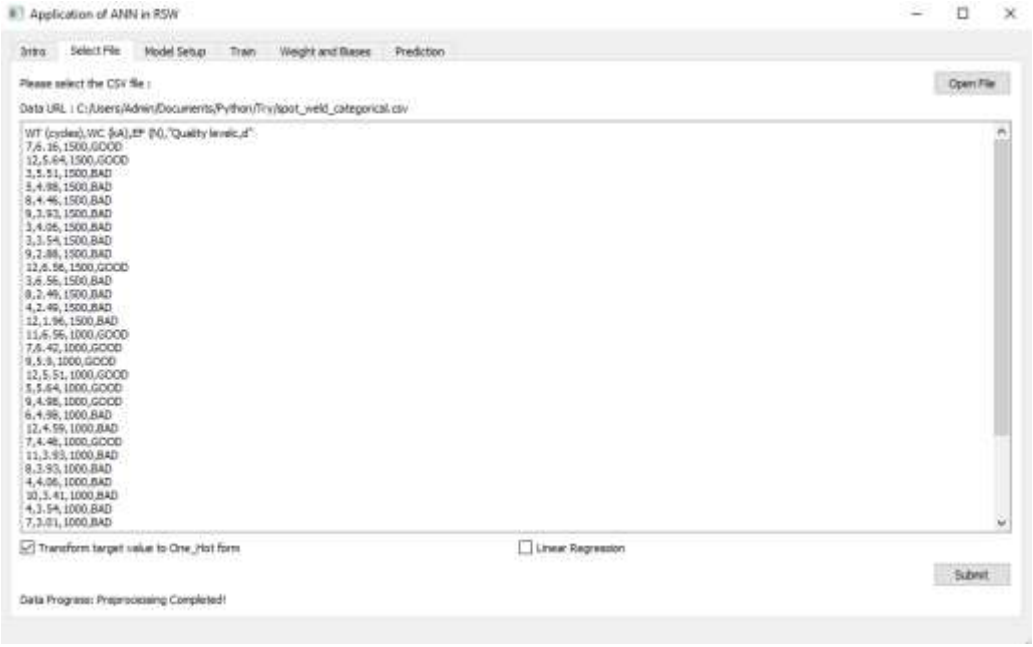


Figure 8: Import and pre-processing of the dataset as Tab 2

The third tab consists the setting up of the model structures and other relevant parameters to implement the ANN model. In this tab, the user is required to choose

the optimizer of GD, SGD and LM training algorithm to train the model. Figure 9 shows the layout of Tab 3 for model setup.

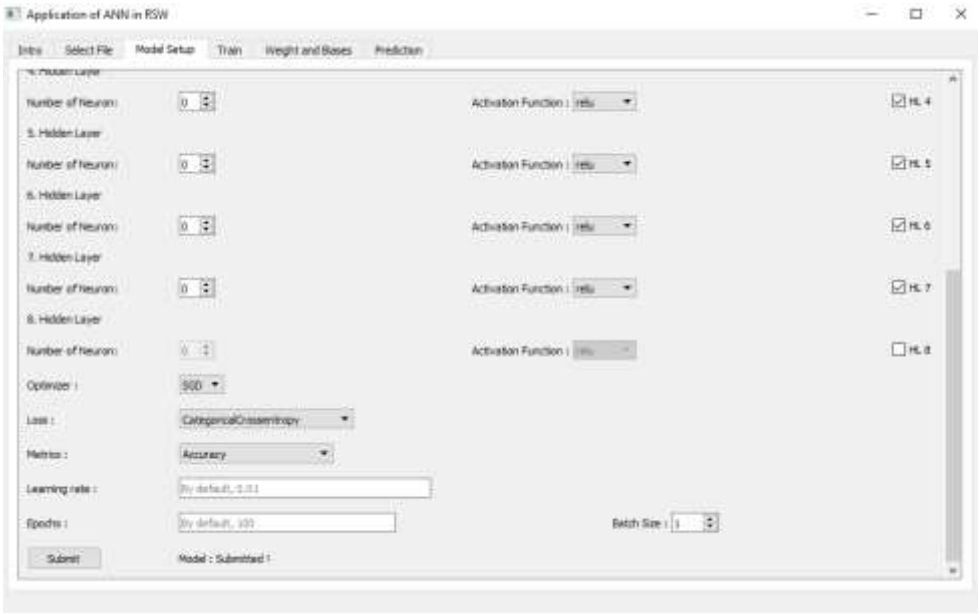
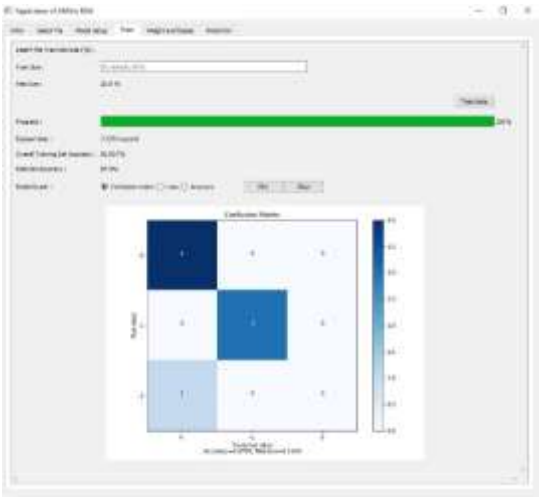


Figure 9: ANN modelling section as Tab 3

The fourth tab as shown in Figure 10 is used to train and evaluate the performance of the ANN model. The training size of the dataset can be set up in this tab and the default value is 80%. This tab shows the value of training and test accuracy once ANN model completes the training process. The confusion matrix can be plotted for multi-classification case.



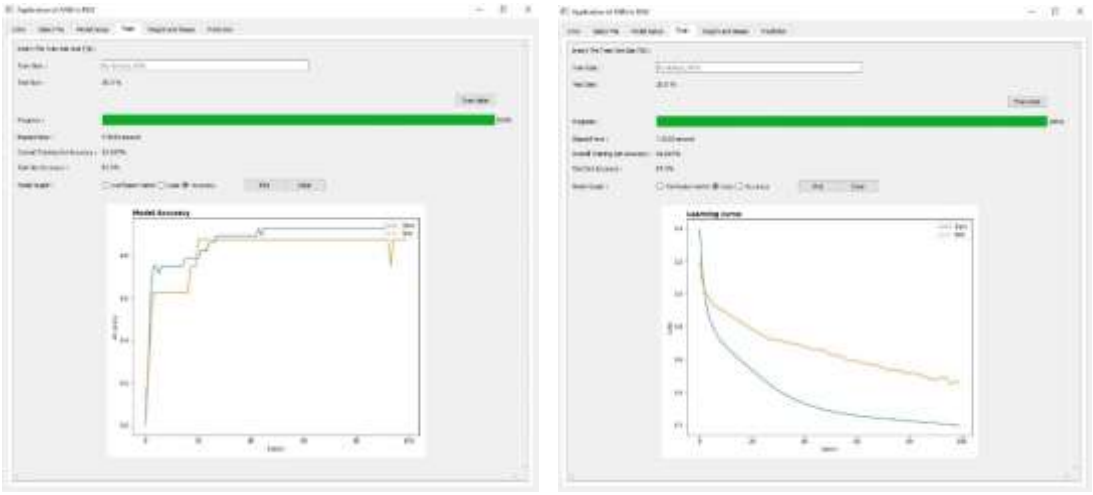


Figure 10: Training and evaluating the ANN model section as Tab 4

The fifth tab as shown in Figure 11 displays the current value of the weight and bias of the model. The weight and bias values for each layer of ANN model are shown and used for research purposes.

Figure 11: Display of current weights and biases of the ANN model as Tab 5

The last tab as in Figure 12 is used to predict the weld quality and TSLBC of RSW with the corresponding input from the user. The user is required to provide three input parameters to make the prediction based on the developed training algorithm.

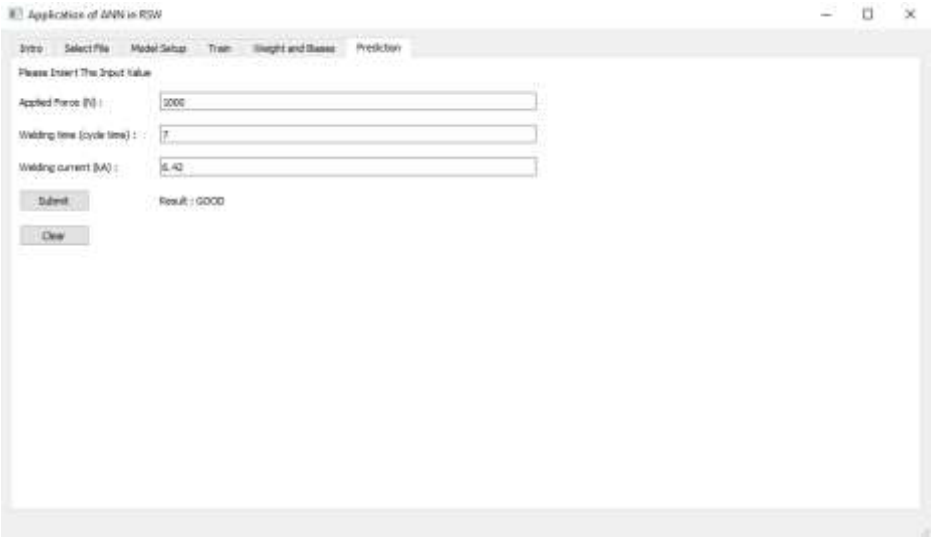


Figure 12: Prediction section as Tab 6

After finishing the designing of all six tabs in Q-Check, the final step of the application tool development is to create an independent executable file by grouping all related files. If an executable file in the .exe extension is made, the program can be run on any computer regardless of whether or not Python is installed. Independent execution is possible by including related libraries in the executable file. *Py2exe*, *Pyinstaller*, and *cx_Freeze* are some programs used to create an executable file from Python code. Pyinstaller is used in this study [23]. However, the standalone application tool can only be used for Windows and OS X user. Figure 13 shows the processes required to produce the executable application tool.

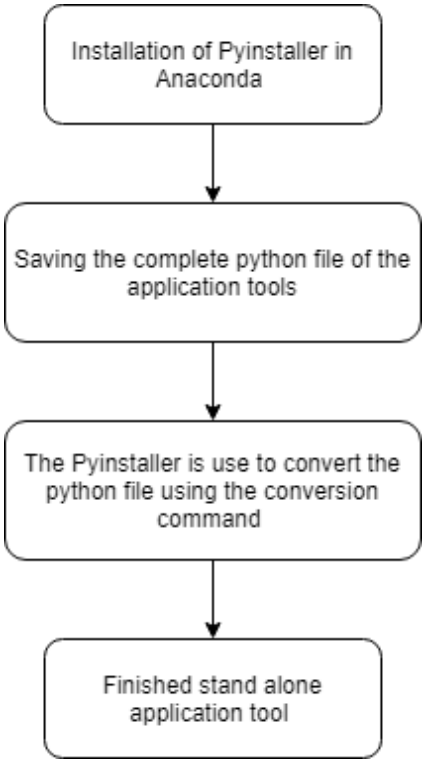


Figure 13: Process of developing standalone application tool

The process starts with the installation of Pyinstaller in Anaconda. Then, the application tool that consists of completed python files are saved before Pyinstaller is used to convert the files to standalone application.

A customized icon is created and used as icon software or graphic editor as a graphical representation of the program. The type of the icon used is in .ico format to apply the icon into the application using the command from the Pyinstaller. Figure 14 shows the icon used for Q-Check application.

Figure 14: Icon for Q-Check

Various types of command are available in Pyinstaller to configure the program properties. For example, -F command is used to create a one-file bundled executable

of the application. The application is located inside disk folder that is generated during execution which is shown in Figure 15.

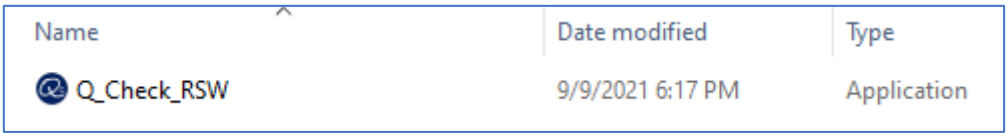


Figure 15: Developed Application with Customized Icon

6. Result and Discussion

The result of the research is an application tool that has the capability of evaluating and predicting the result of TSLBC value and the quality of RSW using GD, SGD and LMBP algorithms. The results obtained in this section are generated from *Q-Check* using the Qt designer from PyQt5 with Spyder IDE. During the training process, the number of epochs for the GD and SGD algorithms is 100 while the LM algorithm is 50, and the value of learning rate used is 0.01.

Table 7 shows the results of *MSE* and accuracy acquired in training and testing of GD, SGD and LM algorithms for numerical output using 2 different ANN structures.

Table 7: *MSE* and accuracy of the linear regression for GD, SGD and LM algorithms.

ANN structure	Training algorithm	MSE		Accuracy
		Training *(%)*	Testing *(%)*	(%)
3-39-1	GD	7.545	12.78	87.220
	SGD	1.554	7.135	92.865
	LM	2.215	6.330	93.670
3-39-39-1	GD	11.749	11.827	88.173
	SGD	1.09	10.716	89.284
	LM	1.569	6.86	93.14

The *MSE* obtained from Table 7 shows that the LM produced the highest accuracy overall while GD gave the lowest accuracy. Based on Table 7, there was a slight variance of accuracy between 1 and 2 hidden layers of ANN structure. The structure

with 1 hidden layer tended to produce an average higher accuracy for all algorithms. Hence, this ANN model is used in predicting the TSLBC of RSW since it has an excellent ability to generalize the 28 input/target pairs employed during the training process. For both structures, the LM gave the highest accuracy while the LM algorithm had a faster rate convergence since it required half value of the epochs than GD and SGD algorithms.

Based on the accuracy result from Table 8, it shows that the ANN model using SGD and LM tended to generalize well for the categorical output of the RSW dataset as it provided an accuracy of 75% for both structures. The GD algorithm gave the lowest performance for both ANN structures with 62.5 % of accuracy.

Table 8: Accuracy of the non-linear model using GD, SGD and LM algorithms

ANN structure	Training algorithm	Train set (%)	Test set (%)
3-39-3	GD	85.714	62.5
	SGD	96.429	75.0
	LM	89.286	75.0
3-39-39-3	GD	78.571	62.5
	SGD	96.429	75.0
	LM	92.857	75.0

In order to summarize the performance of the ANN algorithms, confusion matrix for each algorithm was plotted. Figure 16 shows the result of confusion matrix for GD, SGD and LM algorithms. The number 0 represents Good, 1 is Bad and 2 is Worst which refer to the RSW weld quality that has been encoded earlier.

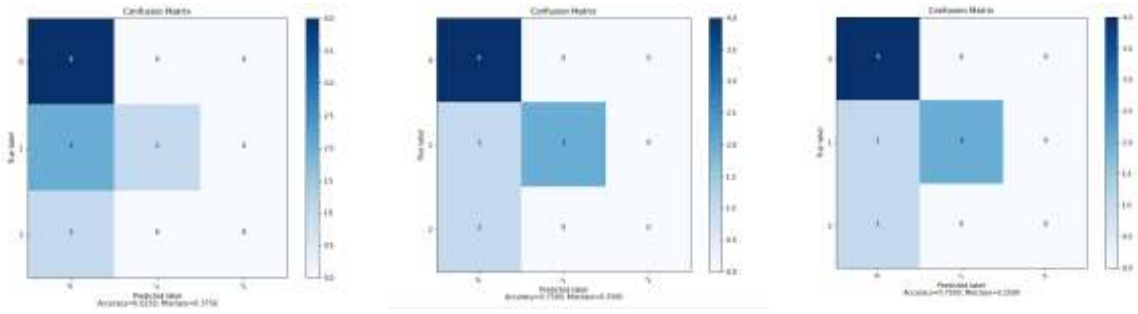


Figure 16: Confusion matrix for GD (left), SGD (middle) and LM (right).

The confusion matrix showed that the ANN model using GD and LM algorithms correctly predicted 6 out of 8 data points during the testing process while GD correctly predicted 5 out of 8 data points.

6. Conclusion and Recommendation

An investigation focusing on the development of an application tool that is able to predict the TSLBC and quality weld of RSW using the ANN model was developed in this study. An application tool was developed using an open-source library and module of Tensorflow and Keras, PyQt5 and Qt designer, Matplotlib, and others. Moreover, the highlight of this study was neglecting the need for an experimental set up in calibrating RSW equipment and materials preparation. The ANN training algorithm proposed in this study was BPNN algorithm which consisted of GD, SGD, and LM. This presented a work approach in which all parameters, geometry, dimensions, and boundary conditions were set in a similar means to ensure the realistic comparison with experimental and conventional experimental study of [3]. According to the results of this study, the following conclusions can be drawn:

1. The GUI-based application tool was successfully developed in predicting the weld quality and TSLBC of RSW using Tensorflow, Keras with the Spyder IDE.
2. The accuracy produced from the prediction of the TSLBC of RSW for GD, SGD, and LM training algorithms was 82.220%, 92.865%, and 93.670%, respectively, for ANN structure 3-39-1. Meanwhile, for ANN structure of 3-39-39-1, the accuracy was 88.173%, 89.284%, and 93.140% for GD, SGD, and LM training algorithms, respectively.
3. For ANN structure of 3-39-1, the accuracy obtained from predicting the quality weld of RSW using the GD, SGD, and LM training algorithms was 62.5 %, 75%, and 75%, respectively. The accuracy obtained for the 3-39-39-3 ANN structure was identical to that obtained with the 3-39-3 structure.
4. The results obtained in predicting TSLBC and the quality weld of RSW showed that the SGD and LM algorithm produced significant results to generalize the ANN model.

5. The LM algorithm had a faster rate convergence than the GD and SGD algorithms since it only used half the epoch than GD and SGD while obtaining the highest accuracy in predicting TSLBC and weld quality.
6. The proposed methodology was relatively successful and straightforward to be used in a variety of other procedures. Additionally, adopting the suggested technique enables engineers to adjust parameters directly using a GUI-based application tool without any prior theoretical understanding of neural computing.

As further recommendations to current research, the following focus can enhance the application with:

1. Adding the various types of machine learning algorithms into the application tool such as Adam, RMSProp, Adadelta and others.
2. Modification to the current GUI by adding more features such as plotting graph tools and exporting the result into excel format for reporting purposes.

Acknowledgement

The authors would like to express their gratitude to the staff members of Smart Manufacturing Research Institute (SMRI), School of Mechanical Engineering as well as the staff of Welding Laboratory, Advanced Manufacturing Laboratory, Advanced Manufacturing Technology Excellence Centre (AMTEx) and Research Interest Group: Advanced Manufacturing Technology (RIG:AMT) at School of Mechanical Engineering, Universiti Teknologi MARA (UiTM) in Shah Alam, Malaysia. This research is financially supported by Technogerma Engineering & Consulting Sdn. Bhd. with research grant project number TEC/SMRI/ANN/Pyhton/01012021.

Conflict of Interest:

The authors declare that they have no conflict of interest.

References

[1] M. Jou, “Real time monitoring weld quality of resistance spot welding for the fabrication of sheet metal assemblies,” *J. Mater. Process. Technol.*, vol. 132, no. 1–3, pp. 102–113, 2003, doi: 10.1016/S0924-0136(02)00409-0.

[2] B. N. Panda, M. V. A. Raju Bahubalendruni, and B. B. Biswal, “Optimization of resistance spot welding parameters using differential evolution algorithm and GRNN,” *2014 IEEE 8th Int. Conf. Intell. Syst. Control Green Challenges Smart Solut. ISCO 2014 - Proc.*, pp. 50–55, 2014, doi: 10.1109/ISCO.2014.7103917.

[3] Ó. Martín *et al.*, “Quality prediction of resistance spot welding joints of 304 austenitic stainless steel,” *Mater. Des.*, 2009, doi: 10.1016/j.matdes.2008.04.050.

[4] M. Hamedi, M. Shariatpanahi, and A. Mansourzadeh, “Optimizing spot welding parameters in a sheet metal assembly by neural networks and genetic algorithm,” *Proc. Inst. Mech. Eng. Part B J. Eng. Manuf.*, vol. 221, no. 7, pp. 1175–1184, 2007, doi: 10.1243/09544054JEM476.

[5] H. Chiroma *et al.*, “Neural networks optimization through genetic algorithm searches: A review,” *Appl. Math. Inf. Sci.*, vol. 11, no. 6, pp. 1543–1564, 2017, doi: 10.18576/amis/110602.

[6] H. L. Lin, T. Chou, and C. P. Chou, “Optimation of resistance spot welding process using Taguchi method and a neural network,” *Exp. Tech.*, vol. 31, no. 5, pp. 30–36, 2007, doi: 10.1111/j.1747-1567.2007.00186.x.

[7] D. Zhao, Y. Wang, D. Liang, and M. Ivanov, “Performances of regression model and artificial neural network in monitoring welding quality based on power signal,” *J. Mater. Res. Technol.*, vol. 9, no. 2, pp. 1231–1240, 2020, doi: 10.1016/j.jmrt.2019.11.050.

[8] X. Wan, Y. Wang, and D. Zhao, “Grey relational and neural network approach for multi-objective optimization in small scale resistance spot welding of titanium alloy,” *J. Mech. Sci. Technol.*, vol. 30, no. 6, pp. 2675–2682, 2016, doi: 10.1007/s12206-016-0232-4.

[9] L. Gong, Y. Xi, and C. Liu, “Embedded artificial neuval network-based real-time

half-wave dynamic resistance estimation during the A.C. resistance spot welding process," *Math. Probl. Eng.*, vol. 2013, 2013, doi: 10.1155/2013/862076.

[10] H. T. Lee, M. Wang, R. Maev, and E. Maeva, "A study on using scanning acoustic microscopy and neural network techniques to evaluate the quality of resistance spot welding," *Int. J. Adv. Manuf. Technol.*, vol. 22, no. 9–10, pp. 727–732, 2003, doi: 10.1007/s00170-003-1599-9.

[11] J. M. Park and H. T. Kang, "Prediction of fatigue life for spot welds using back-propagation neural networks," *Mater. Des.*, vol. 28, no. 10, pp. 2577–2584, 2007, doi: 10.1016/j.matdes.2006.10.014.

[12] H. Pashazadeh, Y. Gheisari, and M. Hamedi, "Statistical modeling and optimization of resistance spot welding process parameters using neural networks and multi-objective genetic algorithm," *J. Intell. Manuf.*, vol. 27, no. 3, pp. 549–559, 2016, doi: 10.1007/s10845-014-0891-x.

[13] G. G. Ahmad, "Using artificial neural networks with graphical user interface to predict the strength of carded cotton yarns," *J. Text. Inst.*, vol. 107, no. 3, pp. 386–394, 2016, doi: 10.1080/00405000.2015.1034930.

[14] Krishnanand, S. Soni, A. Nayak, and M. Taufik, "Development of graphics user interface (GUI) for process planning in extrusion based additive manufacturing," *Mater. Today Proc.*, Mar. 2021, doi: 10.1016/j.matpr.2021.02.306.

[15] Ó. Martín, P. De Tiedra, and M. López, "Artificial neural networks for pitting potential prediction of resistance spot welding joints of AISI 304 austenitic stainless steel," *Corros. Sci.*, vol. 52, no. 7, pp. 2397–2402, 2010, doi: 10.1016/j.corsci.2010.03.013.

[16] C. E. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, "Activation functions: Comparison of trends in practice and research for deep learning," *arXiv*, pp. 1–20, 2018.

[17] B. O. P. Soepangkat, B. Pramujati, M. K. Effendi, R. Norcahyo, and A. M. Mufarrih, "Multi-objective Optimization in Drilling Kevlar Fiber Reinforced Polymer Using Grey Fuzzy Analysis and Backpropagation Neural Network–

Genetic Algorithm (BPNN–GA) Approaches," *Int. J. Precis. Eng. Manuf.*, vol. 20, no. 4, pp. 593–607, 2019, doi: 10.1007/s12541-019-00017-z.

[18] S. Ruder, "An overview of gradient descent optimization algorithms," pp. 1–14, 2016, [Online]. Available: http://arxiv.org/abs/1609.04747.

[19] D. W. Marquardt, "An Algorithm for Least-Squares Estimation of Nonlinear Parameters," *J. Soc. Ind. Appl. Math.*, vol. 11, no. 2, pp. 431–441, Jun. 1963, doi: 10.1137/0111030.

[20] Fabio Di Marco, "Implementation of Levenberg-Marquardt training algorithm." https://github.com/fabiodimarco/tf-levenberg-marquardt (accessed June 17, 2021).

[21] K. G. Sheela and S. N. Deepa, "Review on methods to fix number of hidden neurons in neural networks," *Math. Probl. Eng.*, vol. 2013, 2013, doi: 10.1155/2013/425740.

[22] P. Mishra *et al.*, "MBA-GUI: A chemometric graphical user interface for multi-block data visualisation, regression, classification, variable selection and automated pre-processing," *Chemom. Intell. Lab. Syst.*, vol. 205, no. July, p. 104139, 2020, doi: 10.1016/j.chemolab.2020.104139.

[23] Pyinstaller development team, "PyInstaller." https://www.pyinstaller.org/ (accessed June 17, 2021).

"**Data availability" / "Availability of Data and Materials**"

All data generated or analysed during this study are included in this published article.