ESTIMATION OF WEIBULL PARAMETERS USING ARTIFICIAL NEURAL NETWORK

By

Md. Sujauddin Mallick

B.Sc. in Mechanical Engineering,1998

Bangladesh University of Engineering & Technology

An MRP

presented to Ryerson University

in partial fulfillment of the

requirements for the degree of

Master of Engineering

in the program of

Mechanical and Industrial Engineering

Toronto, Ontario, Canada

© Md. Sujauddin Mallick 2019

AUTHOR'S DECLARATION

I hereby declare that I am the sole author of this MRP. This is a true copy of the MRP, including any required final revisions.

I authorize Ryerson University to lend this MRP to other institutions or individuals for the purpose scholarly research.

I further authorize Ryerson University to reproduce this MRP by photocopying or individuals or by other means, in total or part, at the request of other institutions for the purpose of scholarly research.

I understand that my MRP made electronically available to the public.

Abstract

Estimation of Weibull Parameters Using Artificial Neural Network

Master of Engineering Project, 2019

Md. Sujauddin Mallick

Mechanical and Industrial Engineering

Ryerson University

Weibull distribution is an important distribution in the field of reliability. In this distribution usually there are two parameters. The usual parameter estimation method is maximum likelihood estimation. Maximum likelihood estimation requires mathematical formulation and prior assumption. Non parametric method such as neural network does not require prior assumption and mathematical formulation. They need data to formulate the model. In this report feed forward neural network with back propagation is used to estimate the parameters of a two-parameter Weibull distribution based on four Scenarios. The Scenario consists of training and test data set. Training and test data set generated through simulated time to failure events using *wblrnd* function in MATLAB. The input to the network is time to failure, and the output is shape and scale parameters. The network is trained and tested using trainbr algorithm in MATLAB. The network performed better on Scenario 2 which has the larger number of training examples of shape and scale.

Acknowledgements

The author would like to thank Dr. Sharareh Taghipour for offering a project on artificial neural network for Weibull parameter estimation. The author would like to express his sincere gratitude for her time and continued guidance for completing the project. Her timely advice and assistance kept me inspired and confident to complete the project.

Finally, the author would like to thank his family for their inspiration and support.

Table of contents

List of Tables

List of Figures

1. Introduction

There is no denying the fact that Weibull distribution is an essential distribution in the field of reliability. It is used to model for not only increasing failures, but also for decreasing failures and commonly used in reliability engineering, medical study, quality regulation, financial affairs, and particle size explanation. It can take the attributes of other form of distributions such as exponential distribution and the normal distribution. In statistical literatures, the Weibull distribution usually appears in the form of two parameters [1].

The probability density function is given below [1].

$$f(t) = \frac{\beta}{\theta}\left(\frac{t}{\theta}\right)^{\beta-1} \quad (1)$$

Then, The Cumulative Density Function will be [1].

$$F(t) = \int_{-\infty}^{\infty} f(t)\,dt = \int_{0}^{\infty} f(t) = \frac{\beta}{\theta}\left(\frac{t}{\theta}\right)^{\beta-1} dt = 1 - e^{-\left(\frac{t}{\theta}\right)^{\beta}} \quad (2)$$

The Reliability function will be [1].

$$R(t) = e^{-\left(\frac{t}{\theta}\right)^{\beta}} \quad (3)$$

t=time variable

ß=Shape Parameter

Ɵ=Scale Parameter

For ß=1, The shape of Probability Density Function (PDF) is close to exponential distribution.

For ß>3, The shape of Probability Density Function (PDF) is close to normal distribution

The shape parameter is the most important parameter in the Weibull distribution. It explains how the data is distributed. By using this parameter, interpretations can be made about a population's failure attributes. The scale is also labeled as characteristic life. It is the 63.2 percentile of the data. It influences not only the mean but also the spread [1].

Parameter estimation is the process of estimating parameters from reliability data. It can also be expressed as a classification problem. There are two types of methods in estimating parameters in a Weibull distribution. They are graphical and analytical method. Graphical method is simple but the possibility of error is higher. Analytical method is of three types. They are maximum likelihood estimation, least square method and the method of moments. Maximum likelihood estimation is a generally used technique. It is a technique which calculates the parameters of a model provided observations. The maximum likelihood estimator has less chance of error in comparison with graphical method, however, it needs a lot of iterations. Method of moments is the technique of estimating population parameters. It begins with deriving equations which is linked to population moments. After that sample is drawn and the population moments are estimated from the sample. In least square method, vertical distance is minimized between the data points. It is introduced in scientific problems [2].

There are various literatures which discussed the parameter estimation of a Weibull distribution. The commonly used method is the maximum likelihood estimator. Watkins et al. [3] introduced maximum likelihood estimator for estimating the Weibull parameters for time to failure data. In another work, Flygen et al. [4] presented the maximum likelihood technique for estimating the Weibull parameters for interval based data. Least square method is discussed in some studies. Zhang et al. [5] used least square technique to estimate Weibull parameters. He compared two Least Square (LS) regression technique for estimating the Weibull parameters. Bütikofer et al. [6] used least square technique to compare the assessment of a two-parameter Weibull distribution. Murthy et al. [7] used least square fit technique for estimating Weibull parameters to investigate wind speed difference.

The analytical method requires presumption and mathematical formulation. Non- parametric method such as artificial neural network does not require the prior assumption and mathematical formulation [8].This neural network can do parameter estimation, even if the data is small. An Artificial neural network is a data- driven approach. It has the features like estimation, learning from instances and simplification. Because of this, it has gained popularity among researchers.

There are over fifty categories of ANN. Among them feed forward network is the simplest and widely used one [10].

There is also fewer studies available for parameter estimation of reliability data using a feed forward neural network. One such study is the work of Ming C. Liu [9] et al. who used back propagation neural network for the parameter estimation of a two-parameter Weibull distribution. He generated simulated failure data for training the neural network.

The report is based on the scientific paper of Liu et al. [9]. In this report, independent failure data is generated using Weibull random numbers in MATLAB. After that, feed forward neural network was applied to estimate the shape and scale parameter of the Weibull distribution in neural network toolbox in MATLAB. This estimation is not done before in MATLAB to the best of author's knowledge.

The report is organized as follows. Section 2 describes the fundamental of artificial neural network. Section 3 discusses the common method of teaching artificial neural network naming back propagation algorithm. Neural network tool box and training algorithm in MATLAB are discussed in Section 4. Section 5 describes the mechanism of generating simulated data based on two scenarios. The constructions of the proposed feed forward neural network and the training mechanism are discussed in Section 6. Section 7 displays a snapshot of training in MATAB for the two scenarios. Section 8 discusses the performance of the proposed network. Section 9 concludes with the positive and negative aspects of artificial neural for estimation of two parameters of a Weibull distribution. An appendix is given in Section 10.

2. Fundamental of Artificial Neural Network

A neural network is a computer program that is built on the structure of a human brain, and simulates its actions. It has a parallel-distributed construction, and, has natural tendency for keeping investigational knowledge. This knowledge is used when required [8].

2.1 Neurons

It is the base for functioning of a neural network. Input signal is received here. Then activation function handles the signal and generates an output signal [8].

2.2 Biological neuron

There are four basic parts of a biological neuron. They consist of Cell body, Dendrites, Synapse and Axon (Figure 1). Cell body of a neuron is termed as soma. Dendrites are like channels. In the channels, signals are collected from attached neurons. With these dendrites neurons collect signals from several neurons. If the aggregate impulses go above a definite threshold, the neuron is likely to stimulate and 'fire'. After that, the axon is electrically active. It acts as output which sent impulse to a neighboring cell. Synapses are the joining points between the dendrites and axon [11].

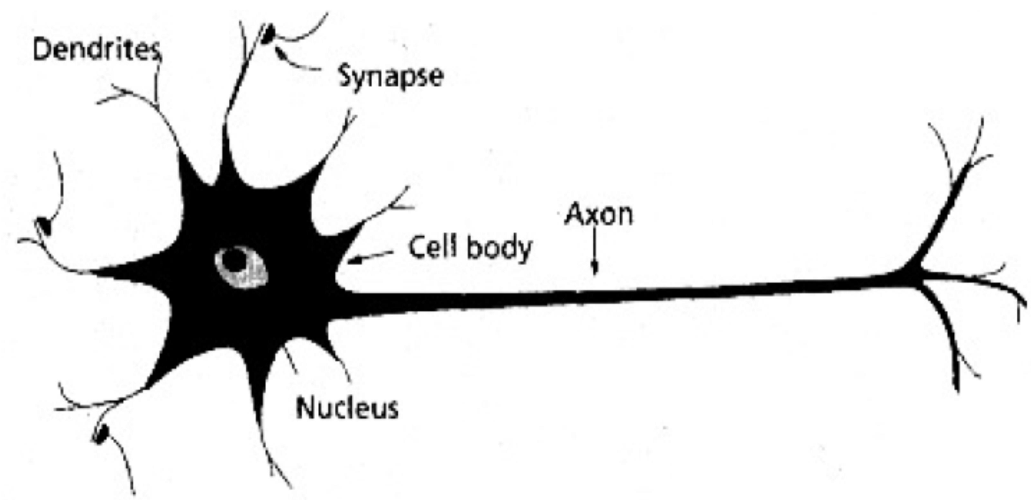

Figure1. Biological Neuron [11].

2.3 A General Artificial Neural Network Model:

The model of neuron is shown in Figure 2. It is the base for designing artificial neural network [12].

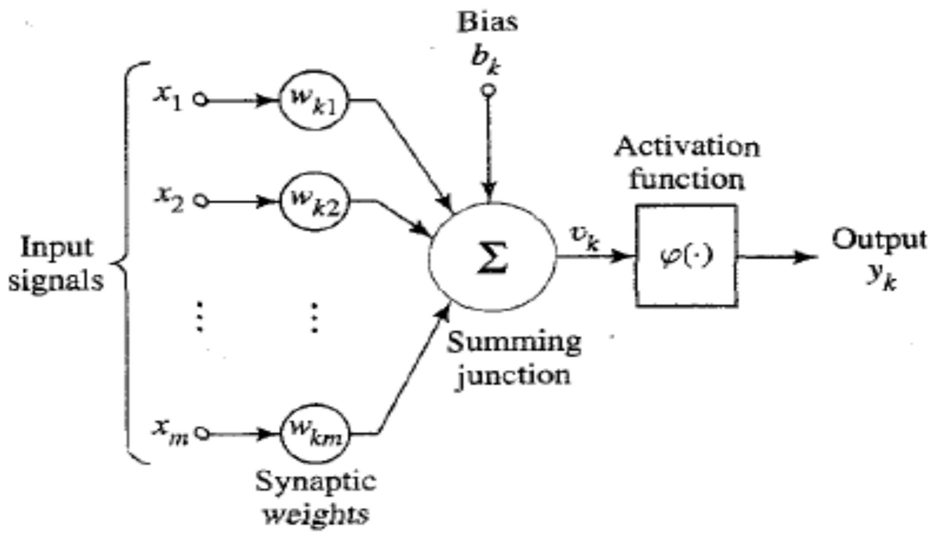

Figure2. Model of neuron [12].

Let,

Input signals	$x_1, x_2, \ldots\ldots x_m$
Synapic weights	$w_{k1}, w_{k2}, \ldots, w_{km}$
Bias	b_k
Activation function	$\Phi(\cdot)$
Linear output	u_k
Output signal	y_k

Input signal xj is fed into the network. Afterwards, it is multiplied by the synaptic weight w_{kj}. The linear output u_k and output signal y_k can be written as [12].

$$u_{k=}\textstyle\sum_{j=1}^{m} w_{kj}\, x_j \tag{4}$$

$$y_k = \emptyset(u_k + b_k) \tag{5}$$

$$y_k = (u_k + b_k) \qquad (6)$$

2.4 Activation functions

In the activation function multiplication is done between input signals and weights. It is labeled as Φ. It is likely to replicate to simulate the firing attribute of neurons, which is added at output end of any neural network. It is also used to determine the output of neural network like yes or No [12].

There are three types of activation function: They are Threshold function, Piecewise-linear and sigmoid [12].

2.4.1 Threshold function:

The threshold function is given below in Equation (7) [12].

$$\Phi(v) = \begin{cases} 1, & if\ \ v \geq 0 \\ 0, & if\ v < 0 \end{cases} \qquad (7)$$

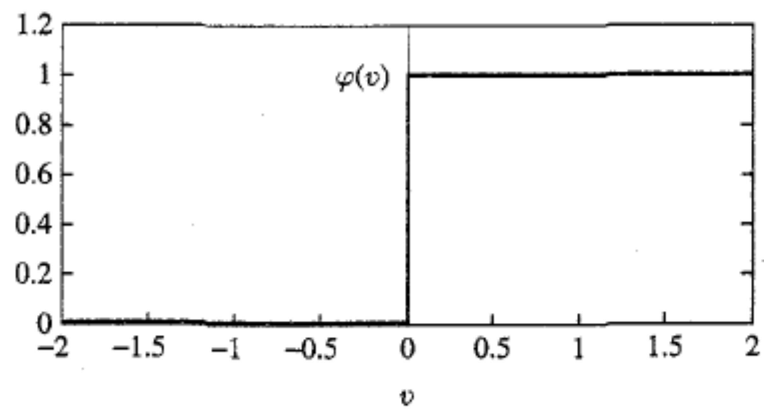

Figure 3.Threshold function [12].

In engineering, the above function calls Heaviside function. It can be designated as unit step function (Figure 3). By applying this threshold function, the output of neuron k is

$$y_k = \begin{cases} 1, & if\ \ v_k \geq 0 \\ 0, & if\ v_k < 0 \end{cases} \tag{8}$$

v_k denotes the local field of neuron

$$v_k = \sum_{j=1}^{m} w_{kj} x_j + b_k \tag{9}$$

2.4.2 Piecewise linear function

The function is given below (Figure 4) [12].

$$(v) = \begin{cases} 1, & if\ v\ \geq \frac{1}{2} \\ v, & if + \frac{1}{2} > x > -\frac{1}{2} \\ 1 & if\ v \leqslant\ -\frac{1}{2} \end{cases} \tag{10}$$

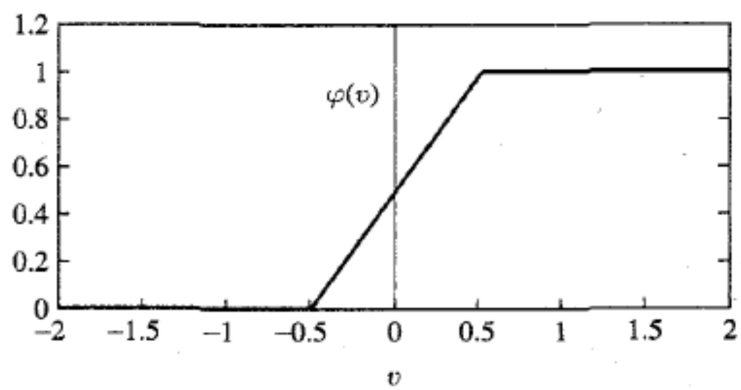

Figure 4.Piece wise function [12].

2.4.3 Sigmoid function

Due to its smooth and restricted nature, it is used in neural network. It is monotonic function that displays a balance between linear and nonlinear nature (Figure 5). The sigmoid function is given below [12].

$$f(x) = \frac{1}{1 + e^{-av}} \tag{11}$$

a is the slope parameter

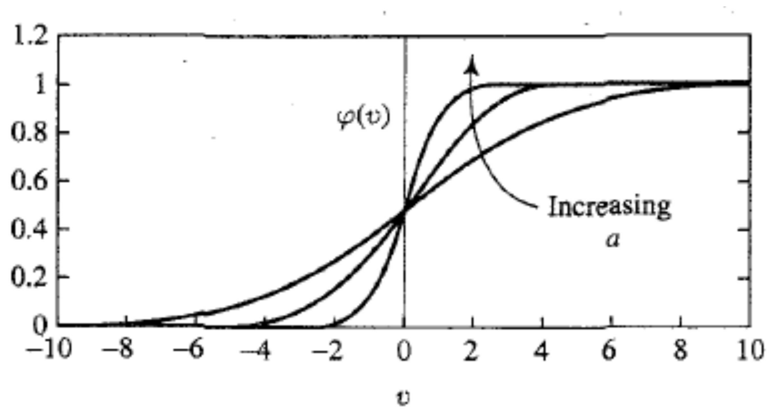

Figure 5.Sigmoid function [12].

Sigmoid functions with various slopes can be obtained if the slope parameter a is varied. This function will be threshold function when a becomes infinity. It is seen previously that the threshold function has the value either 0 or 1. Unlike threshold function, sigmoid function has continuous value from 0 to 1. Another aspect of sigmoid function is that it is differentiable; however, there is no differentiation in threshold function [12].

In equation (7), (10) and (11) activation function varies from 0 to +1. Sometimes this range is from -1 to +1 [12].

Now from equation (7) the threshold function is given below [12].

$$(v) = \begin{cases} 1 & if\ v > 0 \\ 0 & if\ v = 0 \\ -1 & if\ v < 0 \end{cases} \tag{12}$$

Equation (12) is also called the signum function. The Hyperbolic tangent function may be used for an equivalent form of the sigmoid function. It is shown below [12].

$$\emptyset(v) = \tanh(v) \tag{13}$$

2.5 Network architecture:

The most common type of neural network is called a feed forward neural network. It is shown in Figure 6. It may comprise of various neurons. The neurons are labelled as nodes. The nodes are organized in layers. A feed –forward neural network is designed in such a way that information flow is unidirectional. No cycles formed between the nodes [13].

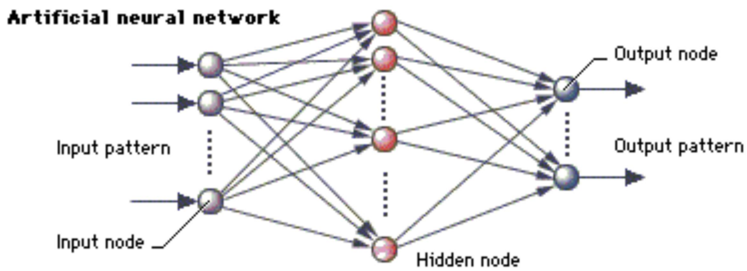

Figure 6.Simple feed forward neural network [13].

There are three forms of nodes on a feed –forward neural network

1. **Input Nodes:** Data is fed into input nodes. They are jointly depicted as "Input Layer". Calculation is not performed in input nodes [13].
2. **Hidden Nodes:** It has indirect connections with the data. The hidden node is doing calculation and information transmission from the input layer to output layer. The group of hidden nodes is called "Hidden Layer". There may be a one input and output layer in feed forward neural network. The network may have zero hidden layer [13]
3. **Output Nodes:** The group of output nodes are called Output Layer. Computation and information transmission is performed from network to user [13].

3. Learning algorithm in neural network

Learning is the ability to improve behavior based on learning. The purpose of the learning rule is to train the network to perform some task [12]. They are of three types:

Supervised learning: Network is given, input and corresponding output. This is called labeled data. External teacher is giving input-output to the network. Input output pattern is fed to the network [14]. In this report, supervise learning is used.

Unsupervised learning: In this case only input is given. There is no label on the data. The Cluster is done based on input data. It is also called self-organization which means that system will likely to develop its representation based on the input data [14].

Reinforcement learning: It is the intermediate form of learning between supervised and unsupervised learning algorithms. After the feedback response is received by the learning machine from the environment, grading is done based on the environmental response [14].

There is a different learning algorithm for feed- forward neural network. Back propagation is one of the popular algorithms. Here network is provided with some training examples. Then the target output is compared with the network output over a definite time through weight adjustment. The backward transmission error is performed by correlating the actual output with desired output during the training. The network performance is optimized by fine tuning weights in the backward route. Training technique is performed as long as the desired output is provided by the network. The algorithm is likely to give reasonable results for the unknown data. Because of the generalization property, it can produce good results even if we do not train on all possible input output pairs [8].

Derivation of back propagation algorithm is given below [12]. Figure 7 shows the flow diagram.

Notation

i,j,k	represents various neurons in the neural network
n	no of training examples
$\varepsilon(n)$	squared error function

$\varepsilon_{av}(n)$	average of squared error function
$e_j(n)$	error signal
$dj(n)$	expected output

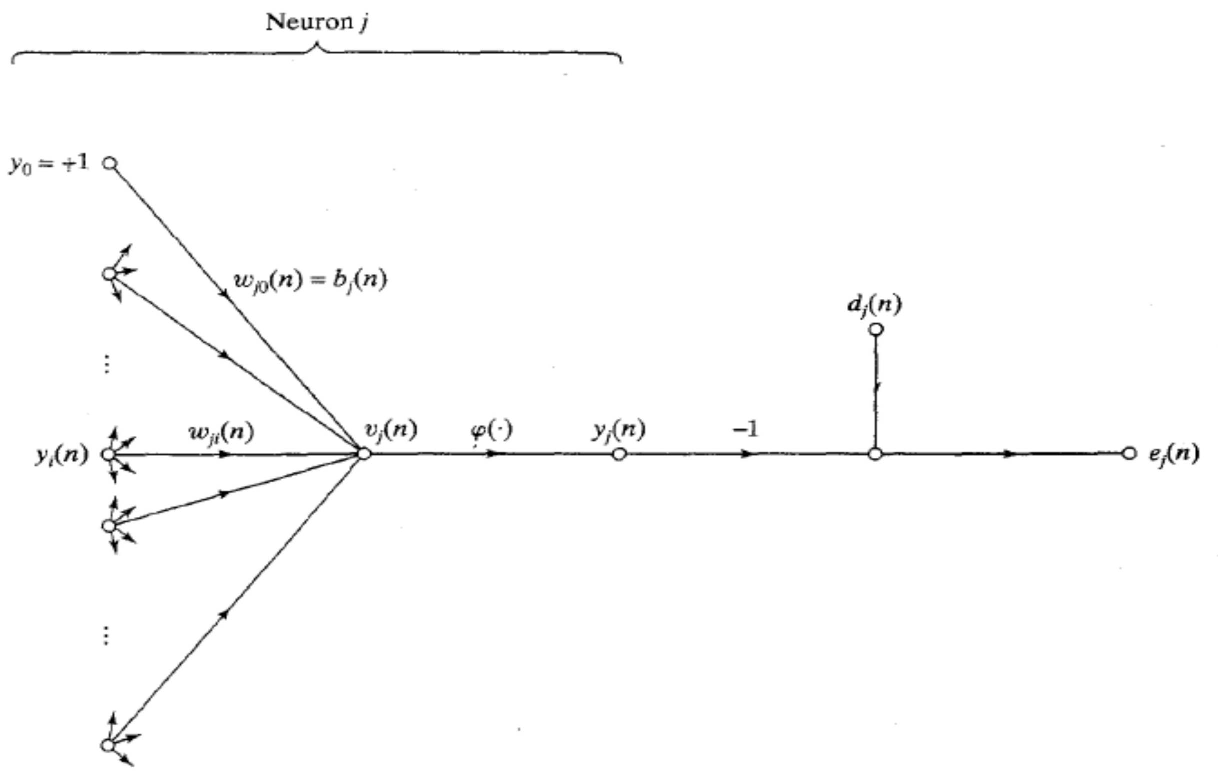

Figure 7. Flow diagram of backpropagation algorithm [12].

$y_j(n)$	functional signal acting at neuron j
w_{ji}	synaptic weight linking between the ith neuron to jth neuron
$v_j(n)$	local field
$\Phi j(.)$	activation function
bj	bias
$x_i(n)$	input vector
$o_k(n)$	total output vector
η_j	learning rate

m_l nodes in layer l

The error signal at output j can be expressed as [12].

$$e_j(n) = d_j(n) - y_j(n) \tag{14}$$

Immediate value of the squared error function for neuron j is [12].

$$\varepsilon(n) = \frac{1}{2}\sum_{j=C} e_j{}^2\,(n) \tag{15}$$

C all the neurons in the output layer

Summing $\varepsilon(n)$ over n and normalizing regard to the size N, Where N represents the total number of instances in the training set [12].

$$\varepsilon_{av} = \frac{1}{N}\sum_{n=1}^{N}\varepsilon_n \tag{16}$$

$\varepsilon(n)$ and ε_{av} is a function of biases and weights. ε_{av} indicates the cost function which is an indicator of learning performance [12].

Local field of neuron j is given below [12].

$$v_{j\,(n)=}\sum_{i=o}^{m} w_{ji}(n)yi(n) \tag{17}$$

At neuron j functional signal on iteration n is [12].

$$y_{j=}\Phi_j(v_j(n)) \tag{18}$$

In synaptic weight $w_{ji}(n)$, correction Δw_{ji} (n) is applied by back propagation algorithm It is relational to the partial derivative. The Gradient is given below by using the chain rule in calculus [12]

$$\frac{ðε(n)}{ðw_{ji}(n)} = \frac{ðε(n)}{ðe_j(n)}\ \frac{ðe_j(n)}{ðy_j(n)}\frac{ðy_j(n)}{ðv_j(\ n)}\frac{ðv_j(\ n)}{ðw_{ji}(n)} \tag{19}$$

$\frac{ðε(n)}{ðw_{ji}(n)}$ signifies a sensitivity element.

Differentiating Equation (15) of both sides,

$$\frac{ðε(n)}{ðe_j(n)} = e_j(n) \tag{20}$$

Differentiating Equation (14) of both sides,

$$\frac{ðe_j(n)}{ðy_j(n)} = -1 \tag{21}$$

Differentiating Equation (18),

$$\frac{ðy_j(n)}{ðv_j(\ n)} = \Phi_j^{'}(v_j(n)) \tag{22}$$

prime denotes differentiation with regard to the argument

Lastly, differentiating Equation (17) with regard to $w_{ji}(n)$

$$\frac{ðv_j(\ n)}{ðw_{ji}(n)} = y_j(n) \tag{23}$$

Using Equation (20), (21), (22), and (23) in (19)

$$\frac{ðε(n)}{ðw_{ji}(n)} = -e_j(n)\ \Phi_j^{'}(v_j(n))\ y_j(n) \tag{24}$$

According to delta rule, correction of Δw_{ji} (n) applied to w_{ji} (n)

$$\Delta w_{ji}\text{ (n)} = -\eta \frac{ð\varepsilon(n)}{ðw_{ji}(n)} \tag{25}$$

In Equation (25), η is learning rate parameter. The meaning of minus sign that it is searching for changing weight, which decrease the value of $\varepsilon(n)$. It is called gradient descent [12].

From equation (25) & (26),

$$\Delta\, w_{ji}(\text{n}) = \eta\, \delta_j(\text{n})\, y_i\ (n) \tag{26}$$

where δ_j is local gradient. The definition of δ_j is given below [12].

$$\begin{aligned}
\delta_j(\text{n}) &= -\frac{ð\varepsilon(n)}{ðv_j(n)} \\
&= -\frac{ð\varepsilon(n)}{ðe_j(n)}\ \frac{ðe_j(n)}{ðy_j(n)} \frac{ðy_j(n)}{ðv_j(\,n)} \\
&= -e_j(n)(-1)\ \Phi_j^{'}(v_j(n)) \\
&= e_j(n)\ \Phi_j^{'}(v_j(n))
\end{aligned} \tag{27}$$

There is a need for modifications in synaptic weights. It is called local gradient. The local gradient is the multiplication of error signal and derivative $\Phi_j^{'}(v_j(n))$ of the accompanying activation function [12].

It can be seen from equations (26) and (27) that the error signal $e_j(n)$ is the main element in computation of weight adjustment$\Delta\, w_{ji}$(n) [12].

4. MATLAB and Neural Network

MATLAB denotes matrix laboratory. The MATLAB software has an interactive environment. In this environment, users can perform various activities, such as, algorithm up gradation, data visualization, and numerical calculation. The MATLAB software has toolboxes. The toolboxes are a group of functions. The toolbox enables users to apply particular technology [15]. In MATLAB, there is a neural network toolbox, which has a collection of functions and structures. It is designed in such a way that code is not needed for activation functions and training algorithms. It is built on network object. The object has the information, such as, layers structure, layers number and linkage between the networks. There are different functions for network creation.They are *newlin, newp,* and *newff*. Newlin is for generating a new layer; *newp* is for generating a perceptron, and *newff* is for generating a feed forward backpropagation network. In the report, *newff* function is used to make a feed forward neural network [16].

As MATLAB is used to model the feed forward neural network, it is important to know some training algorithms with this data mining tool. There are different training algorithms to train a neural network. Some of the training algorithms are given below [17].

traingd: It is the fundamental Gradient descent algorithm. In this algorithm, output error quantified using local search technique. Here gradient error is measured by tuning weights in the direction of descending gradient [17].

traingdm: The algorithm is called Gradient descent back propagation momentum. The algorithm has sharp descent to respond to local gradient and error surface [17].

traingdx : It is called adjusting learning rate. It is quicker in training than *traingd* [17].

trainrp*:* It is called resilience back propagation. The effect of partial derivative magnitude is eliminated through this algorithm. The training algorithm is based on batch mode with quick convergence. There is a need for less storage [17]

traincgf: It is called conjugate gradient with Fletcher-Reeves. It has the minimum requirements for storage among all the conjugate gradient algorithms [17].

traincgp: It is called Conjugate Gradient back propagation with Polak-Riebre. There is a need for marginally larger storage compared to *traincgf*. Convergence is usually quicker [17].

***trainscg**:* It is called Scaled Conjugate Gradient. Line search is not necessary in every iteration. As there is no line search, there is a significant reduction of calculation. But more iteration is needed compared to the other methods [17].

trainbfg: It is called Broyden-Fletcher-Goldfrab-Shanno quasi –Newton method. There is a need for more storage of the estimated Hessian matrix. It has more calculation in each iteration than Conjugate Gradient algorithms; however, converge in less iteration [17].

***trainlm**:* It is called Levenberg-Marquardt back propagation algorithm. The algorithm is quickest for medium sized network. Because of its memory reduction attribute, it is used for large training set [17].

***trainbr*:** It is called Bayesian Regularization algorithm. The algorithm modifies the *trainlm* algorithm and generate networks that generalize well. Optimum network architecture can be determined by this algorithm [17].

Reason for chosing trainbr algorithm:

There are different types of training algorithm. In this report for training neural network, *trainbr* algorithm is used. The reason is as follows: first, it is suitable for noisy and small data; second, In the algorithm, weight and bias values are upgraded in line with the Levenberg-Marquardt optimization and reduces squared errors and weights; third, for generalization the algorithm determines the right combination; fourth, it can give information about effective use of network parameters, and finally, there is no need for separating the validation data set from training data set [17].

5. Data Description

Simulation study is conducted to estimate the Weibull parameters. In the simulation study, random failure data is generated through *wblrnd* function by varying actual shape and scale in MATLAB. Two scenarios were considered:

Scenario 1: It consists of two data sets. They are training and test set. For training, 200 random values of shape and scale taken. Shape value varied between 0.9 to 9.99. Scale value varied between 120 to 5800. For each shape and scale, 20 times to failure events created using the *wblrnd* function in MATLAB. It can be denoted as 200-20. For testing, 50 examples created based on actual shape and scale parameters. Each example has 20 times to failure as independent events. This is denoted as 50-20. In each scenario, logarithm of scale is taken before training. Screenshot of training and test data set are shown in the appendix.

Scenario 2: There are two types of data set which are training and testing data. In Training data 500 random values of shape and scale are taken. Shape value varied between 0.5 to 9.98. Scale value varied between 250 to 4900. Each example has 20 times to failure as independent events. It can be denoted as 500-20. For testing, 50 examples created based on actual shape and scale parameters. In each scenario, logarithm of scale is taken before training. Screenshot for training and test data set are shown in the appendix.

Scenario 3: It is an extension of Scenario 1. In this Scenario, 30 times to failure events created based on each example of shape and scale. This is denoted as 200-30. For testing, 50 examples created based on actual shape and scale. It can be denoted as 50-30.

Scenario 4: It is an extension of Scenario 2. In this Scenario, 30 times to failure events created for each example of scale and shape. It is denoted as 500-30. For testing, 50 examples created based on actual shape and scale. This is denoted as 50-30.

Before feeding data in the neural network for training, data is normalized for decreasing the convergence time. In this report, scaling is performed in MATLAB by using the function *mapminmax*. The coding of *mapminmax* function is given below;

[pn,ps]=mapminmax(p);

[tn,ts]= mapminmax(t);

Input and output are denoted as p and t. pn and tn are normalized input and output. The value of this normalization falls in the interval [-1, 1]. ps and ts are the lowest and highest values of the original inputs and outputs [18].

6. Proposed Neural Network

Feed forward network with back propagation is used to estimate the Weibull parameters. The network structure consists of an input layer, hidden layer and output layer (Figure 8). Input values are time to failure and output values are scale and shape parameters. For the Scenarios, two networks proposed. In one network, there are 20 neurons in the input layer as there are 20 times to failure events in each example of the dataset. In another network, there are 30 neurons in the input layer as there are 30 times to failure events in each example of the data set. In both the network, there are two neurons in the output layer which are scale and shape. Time to failure depicted as TTF. Scale and shape denoted as ϴ and ß. Hidden neuron is selected based on trial and error. By varying hidden neuron different topology of network created, for example, 20-5-2 denotes that the network is configured based on twenty inputs, 5 hidden neurons and 2 outputs.

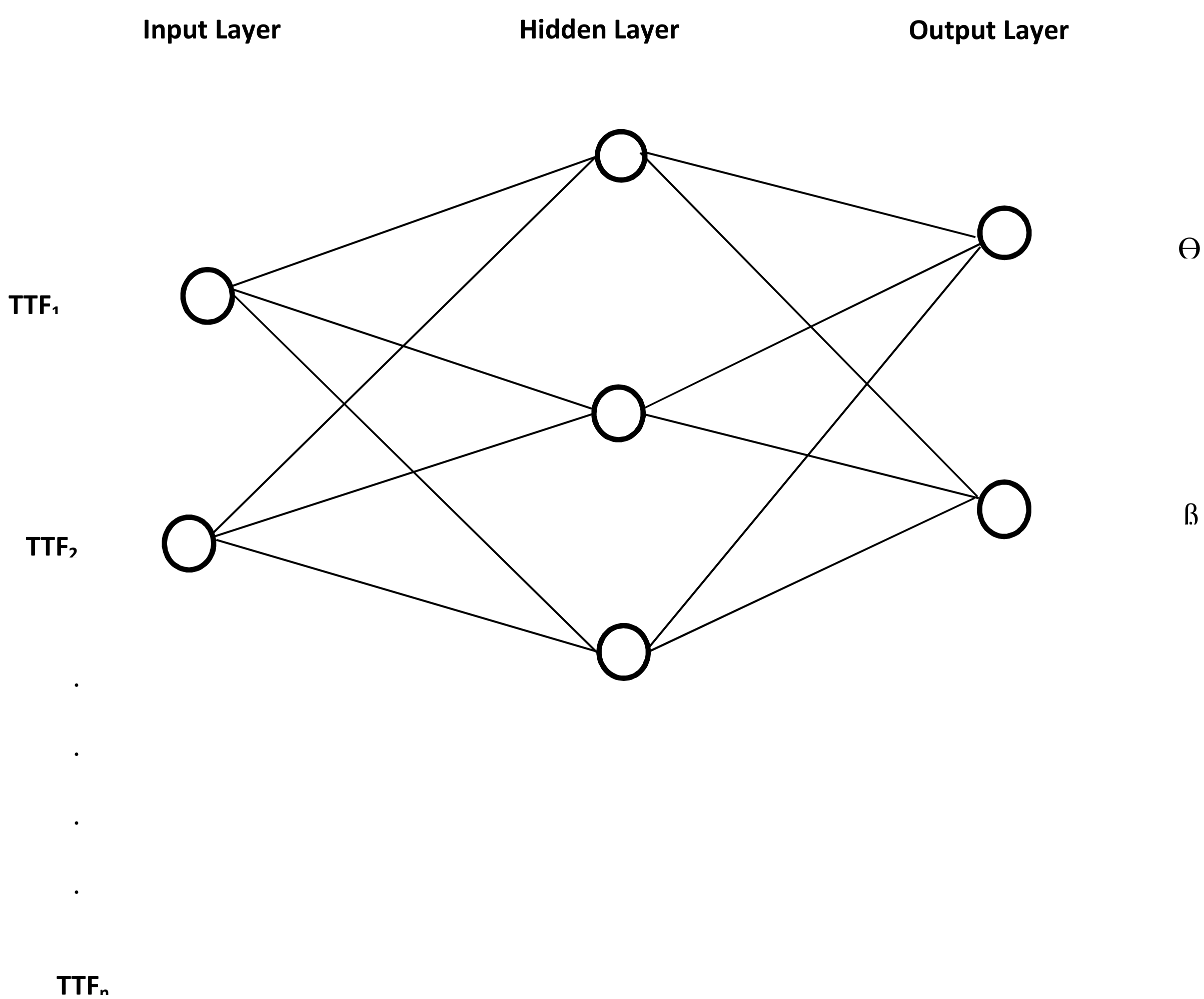

Figure8. Proposed feed forward neural network

7. Snapshot in MATLAB

The snapshot shows the training data set of Scenario 1 and Scenario 2. It shows how the coding is done in forming neural network, training progress window, best training performance, training state and regression.

In Scenario 1, training data set is 200-20. It is experimented in neural network tool box in MATLAB. The sequence is given in the screen shot below. Coding of the neural network in Command window is shown in Figure9.

Command Window

```
>> [pn,ps]=mapminmax(p);
[tn,ts]=mapminmax(t);
net_1=newff([pn],[tn],[5],{'tansig','tansig'},'trainbr');
net_1.divideparam.trainratio=1;
 net_1.trainparam.show=50;
net_1.trainparam.epochs=5000;
net_1.trainparam.goal=1e-06;
network1=train(net_1,pn,tn);
>> a1n=sim(network1,pn);
b1=mapminmax('reverse',a1n,ts);
>> anewn=mapminmax('apply',Test_50,ps);
>> anewn1=sim(network1,anewn);
>> anew1=mapminmax('reverse',anewn1,ts);
>>
```

Figure 9. Code for network 20-5-2

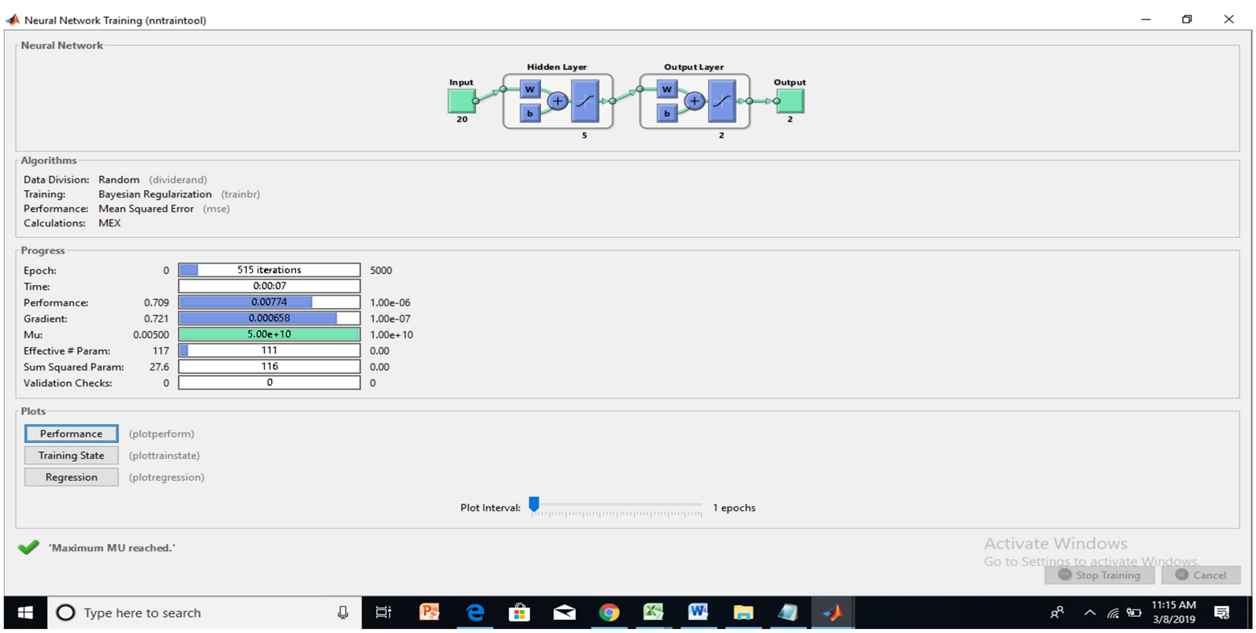

Figure10. Training progress of network 20-5-2

Figure10 represents a summary of the training. In this summary, Epoch, Time, Performance, Gradient, Mu, Effective parameter, Sum Squared Parameter, and validation checks are shown.

Next three useful plots are offered by the training algorithm of neural network which are used to evaluate the performance of neural network (Figure 10). They are: Performance, Training state and Regression. Figure.11 is a performance plot for the best training performance. It shows the best training performance is .0077435 at epoch 513.

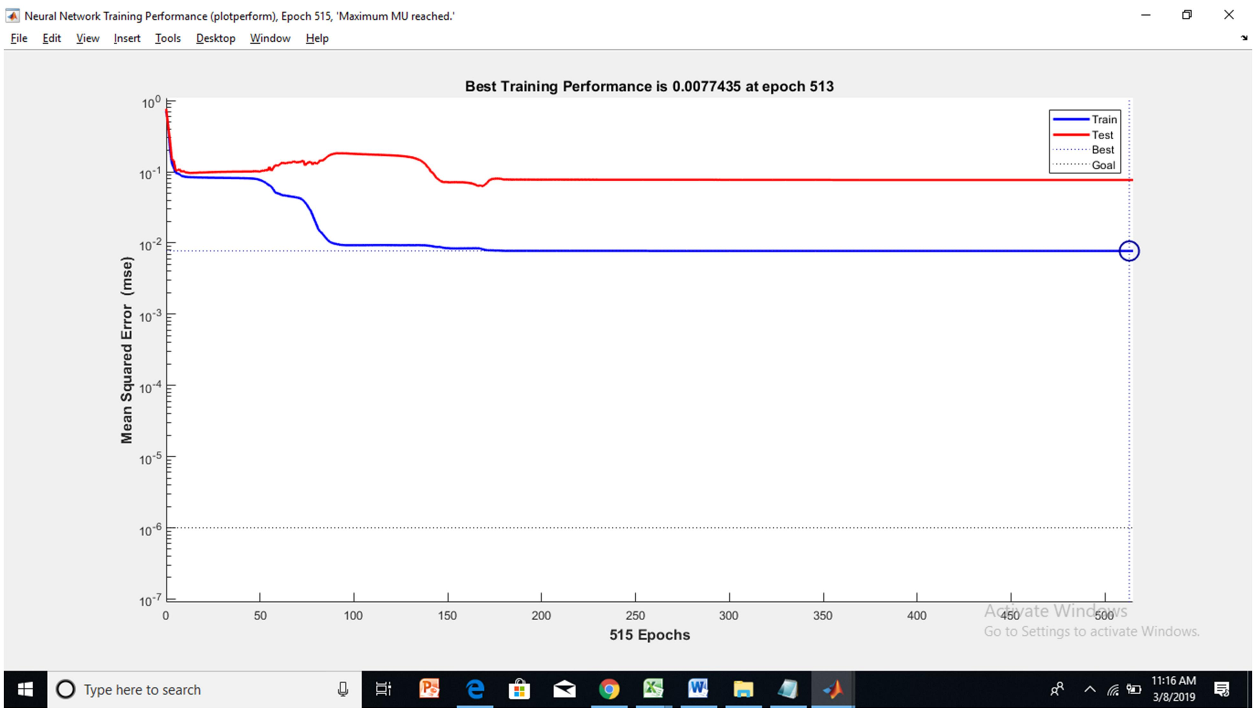

Figure 11. Best training performance plot of network 20-5-2

The training state is shown in Figure 12. It shows graphically the status of the gradient at epoch, Mu, Num parameters, Sum Squared Parameters, and Validation checks.

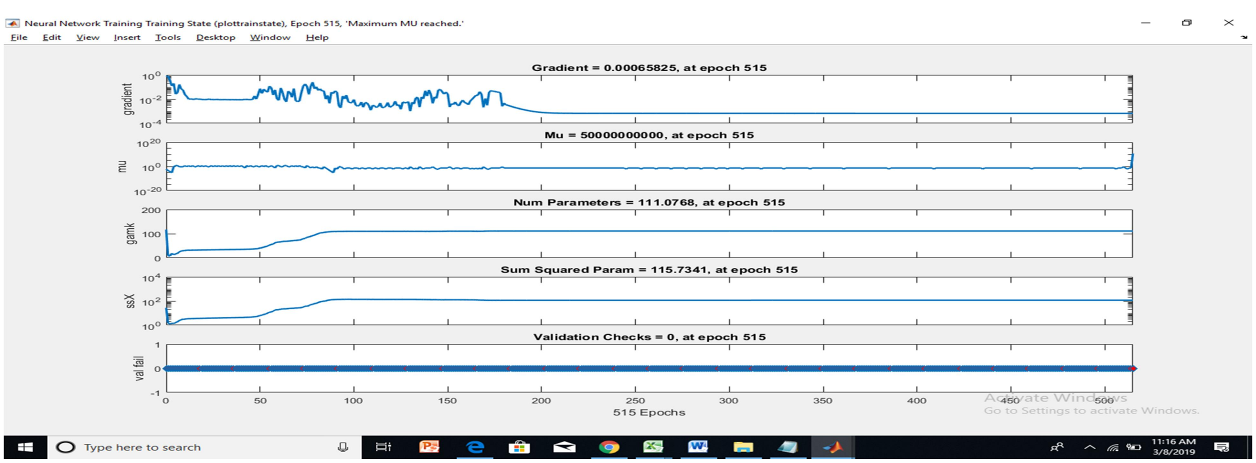

Figure12. Training state plot of network 20-5-2

Finally, regression plot is shown in Figure 13. It shows network outputs with targets for training and testing data.

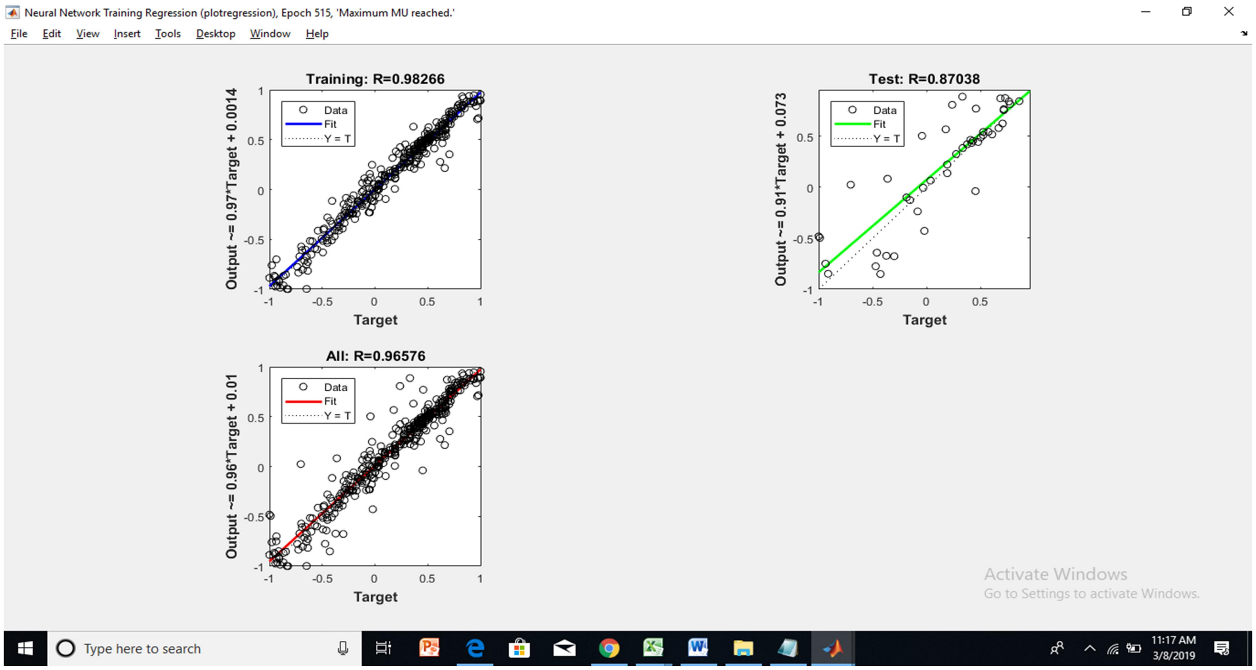

Figure13. Regression plot of network 20-5-2

Similarly, the snapshot is shown in Scenario 2 for training data set 500-20, the network topology is 20-7-2. Here Coding for setting neural network, summary of training, best training performance, training state, & regression plot are shown sequentially.

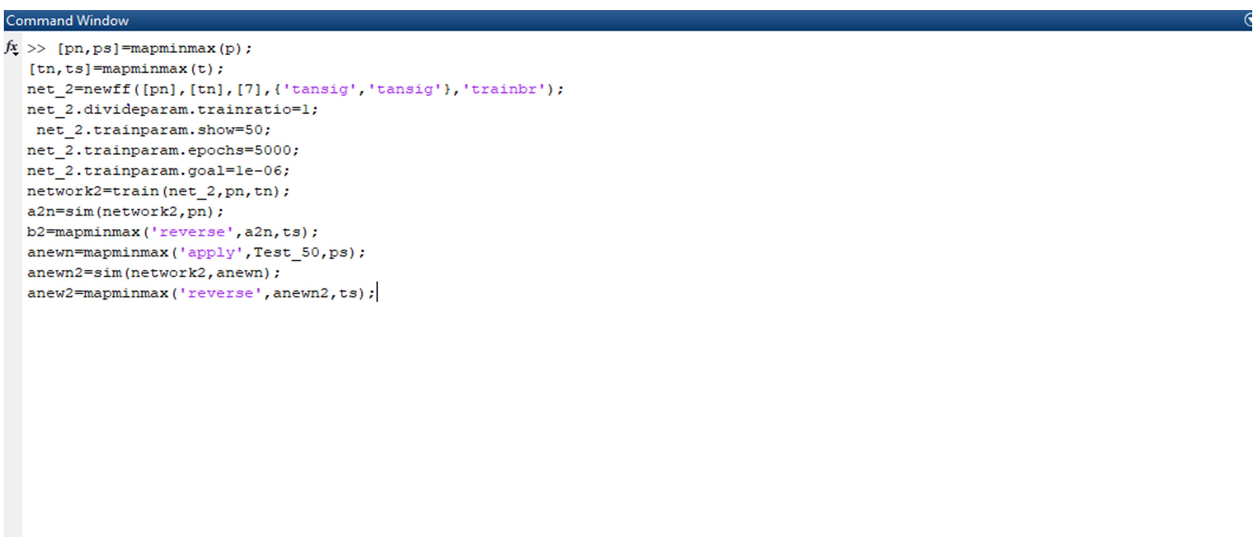

Figure 14.Coding for network 20-7-2

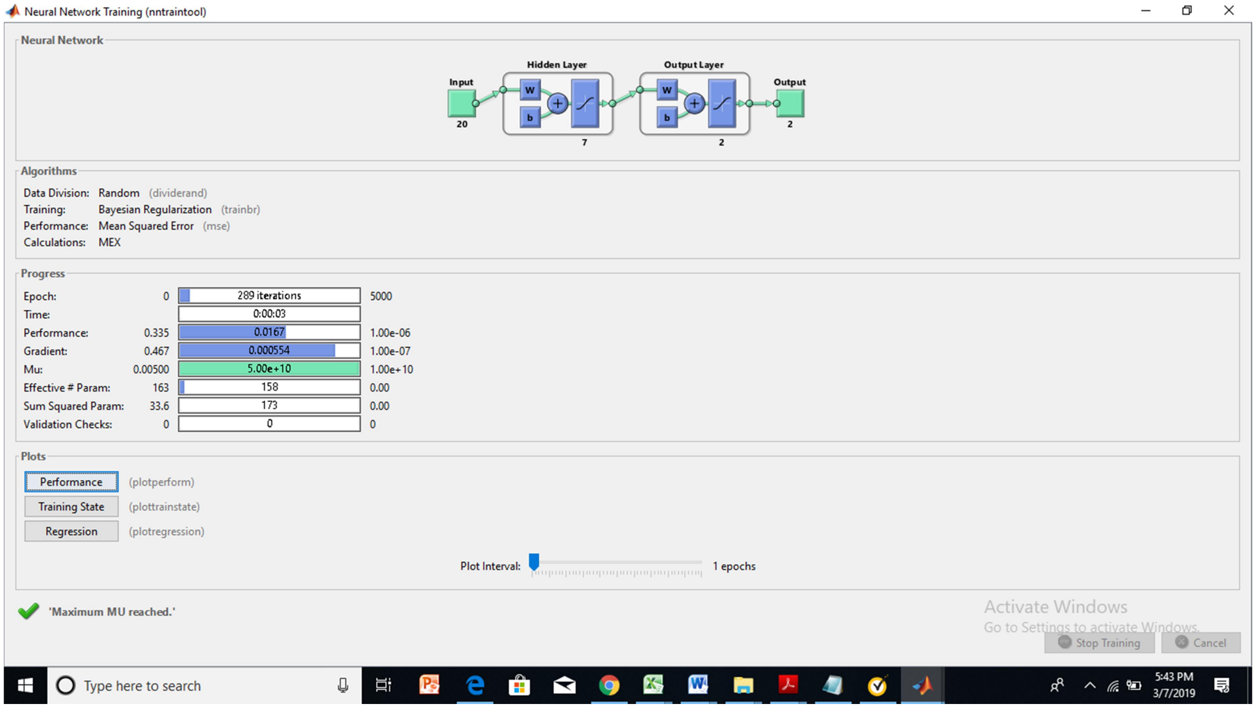

Figure15. Training progress of network 20-7-2

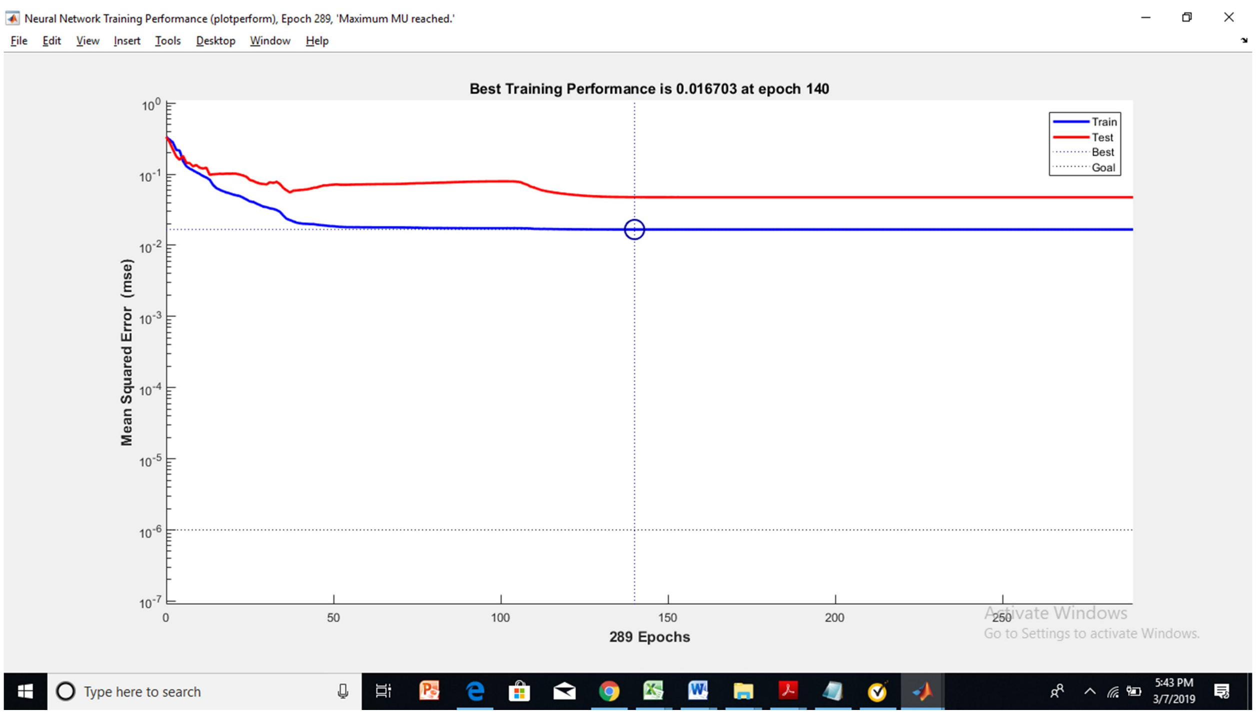

Figure16. Best training performance plot of network 20-7-2

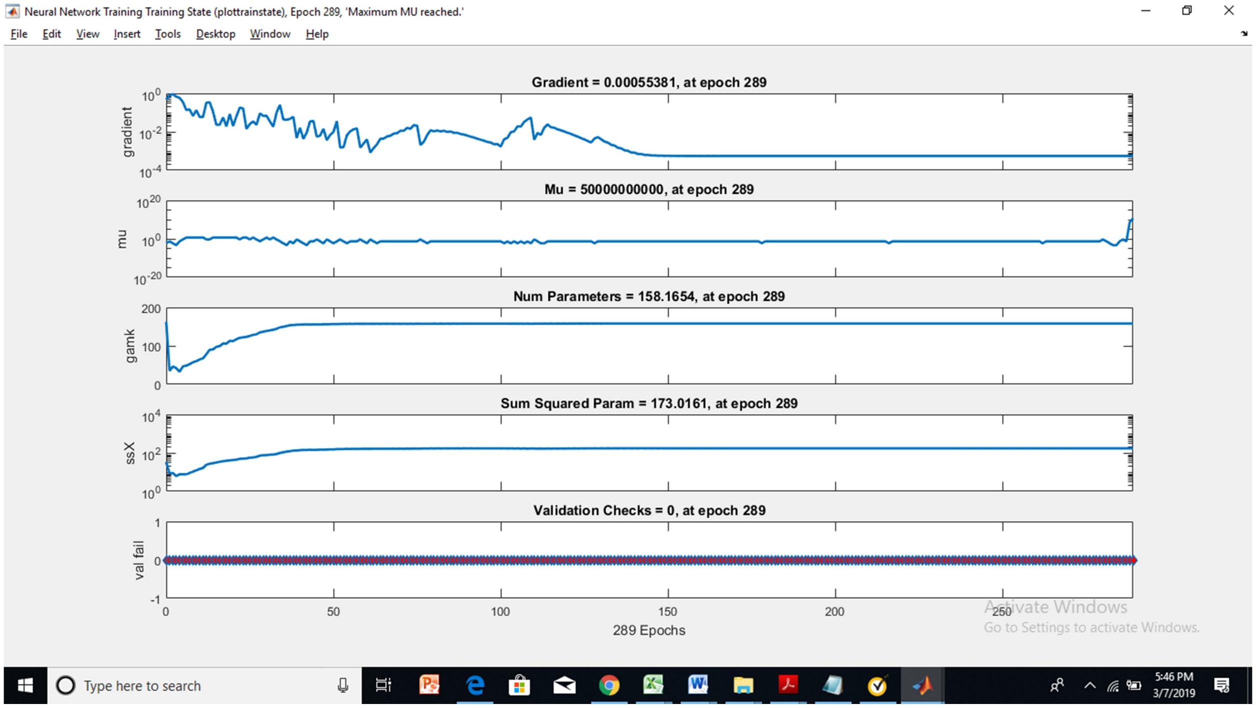

Figure17.Training state of network 20-7-2

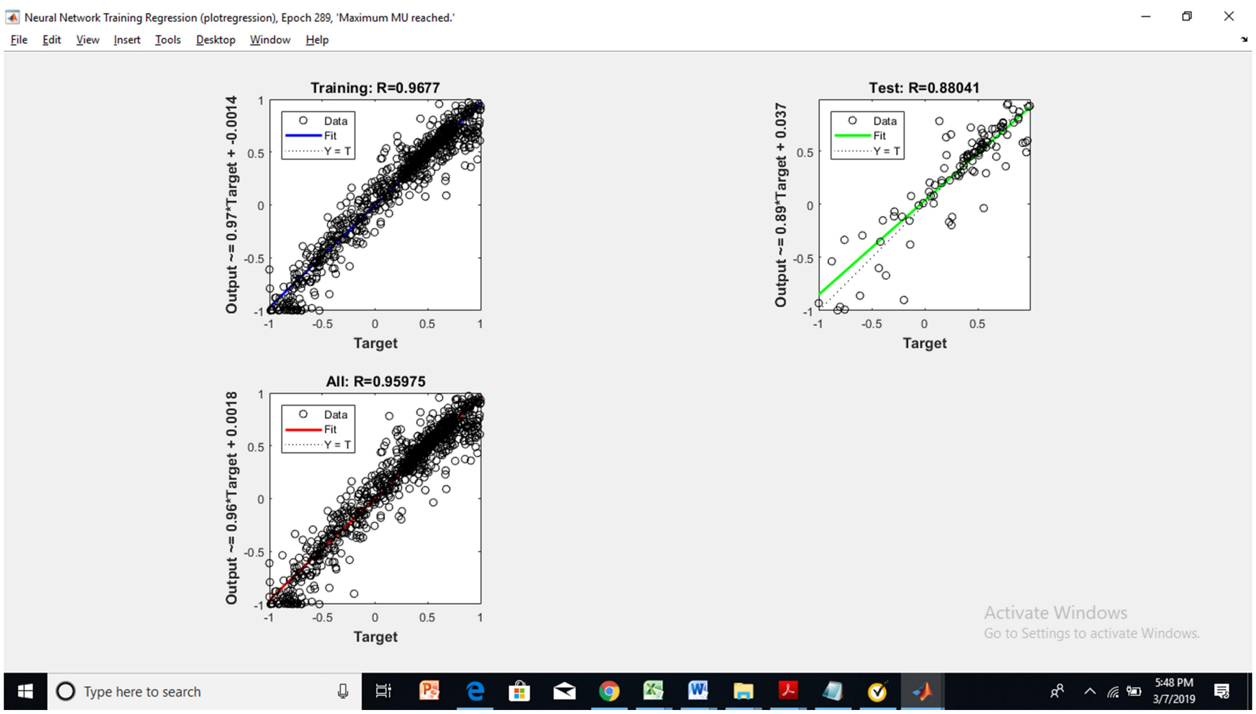

Figure18. Regression plot of network 20-7-2

8. Results and Discussion

Different network topology created based on different hidden neurons. Eight network topologies created for each of the scenarios. To determine the network performance, average percentage error is taken as the indicator of the network performance. It is the sum of the difference between actual output and network output. Then, it is divided by the total no of examples. Results are shown in Tables 1 and 2.

Table 1: Results of Scenario 1

	Network Topology	Training algorithm	Hidden Neuron	Average percentage error (Training)		Average percentage error (Testing)	
				Scale log(Θ)	Shape (β)	Scale log(Θ)	Shape (β)
1	20-5-2	trainbr	5	0.9711	13.97	1.127	25.72
2	20-6-2	trainbr	6	1.157	9.541	1.214	34.35
3	20-7-2	trainbr	7	0.8600	10.09	0.9051	50.29
4	20-9-2	trainbr	9	0.6000	5.799	0.7838	38.43
5	20-11-2	trainbr	11	0.5183	5.025	0.6692	38.65
6	20-13-2	trainbr	13	0.4575	5.752	0.9933	52.76
7	20-15-2	trainbr	15	0.3119	6.852	0.6737	38.06
8	20-17-2	trainbr	17	0.3147	4.667	0.9860	38.97

Table 2: Results of Scenario 2

	Network Topology	Training algorithm	Hidden Neuron	Average percentage error (Training)		Average percentage error (Testing)	
				Scale log(Θ)	Shape (β)	Scale log(Θ)	Shape (β)
1	20-5-2	trainbr	5	1.139	21.54	0.9828	25.72
2	20-7-2	trainbr	7	1.020	16.34	0.9212	22.51
3	20-8-2	trainbr	8	0.9968	16.09	1.209	27.97
4	20-9-2	trainbr	9	0.8242	13.59	0.9335	33.47
5	20-11-2	trainbr	11	0.9214	10.35	1.035	30.98
6	20-13-2	trainbr	13	0.8047	9.961	1.164	33.29
7	20-15-2	trainbr	15	0.7581	11.92	0.9830	31.56
8	20-17-2	trainbr	17	.6673	9.924	1.161	36.70

Table 3: Results of Scenario 3

	Network Topology	Training algorithm	Hidden Neuron	Average percentage error (Training) %		Average percentage error (Testing) %	
				Scale log(Θ)	Shape (β)	Scale log(Θ)	Shape (β)
1	20-5-2	trainbr	5	1.057	7.363	0.8849	40.45
2	20-6-2	trainbr	6	0.9395	7.063	1.143	39.17
3	20-7-2	trainbr	7	0.6774	6.590	1.048	44.55
4	20-9-2	trainbr	9	0.5545	6.730	1.385	31.79
5	20-11-2	trainbr	11	0.3295	2.887	2.567	42.66
6	20-13-2	trainbr	13	0.3842	7.597	1.615	48.18

Table 4: Results of Scenario 4

	Network Topology	Training algorithm	Hidden Neuron	Average percentage error (Training) %		Average percentage error (Testing) %	
				Scale log(Θ)	Shape (β)	Scale log(Θ)	Shape (β)
1	20-5-2	trainbr	5	1.072	21.18	1.307	51.01
2	20-7-2	trainbr	7	1.046	16.53	1.321	60.27
3	20-8-2	trainbr	8	0.9631	17.08	1.174	55.57
4	20-9-2	trainbr	9	.8494	12.26	1.495	49.36
5	20-11-2	trainbr	11	0.6588	11.61	1.078	47.10

Table 1 is the result of the first scenario for training and testing data. From the table it is seen that the average percentage error for scale varies between 0.3119% to 1.157% and 0.6692% to 1.214% in training and testing data. This error is acceptable. But there is variation in average percentage error of shape. It is seen from the table that the percentage error for shape is 4.667% during training for the network topology 20-17-2. It is the least error among all the network topology, however, when it is tested with new data which is not trained earlier, the average percentage error for shape is 38.97%. In another network topology of 20-5-2, the average percentage error during training for shape is 13.97% which is higher than 20-7-2. But the average percentage error for shape during testing of this network is 25.72% which is better than any other network topologies. For topology 20-6-2 and 20-7-2 poor training performance result in poor testing performance of shape. During training, network topology of 20-9-2, 20-11-2 and 20-13-2 has a similar average percentage error for the shape, the testing performance is good for 20-9-2. Finally, it had been found that the network topology 20-5-2 is better than any other network topologies because it has less average percentage error during testing and better prediction capacity.

The results of the second scenario are shown in the Table 2. In training and testing data, average percentage error for scale varies between 0.6673% to 1.02% and 0.9212 to 1.209% for training and testing data. This error is acceptable. There is variation in average percentage error in the shape parameter. Among all the network topologies, 20-17-2 has the lowest average percentage error for the shape which is 9.924%. But the testing performance of this network is 36.70% which is higher than any other network topologies. In another network topology 20-7-2, the average percentage error during training is 16.34%, although higher than network topology of 20-17-2, the prediction performance is the best in comparison with all the other network topologies. The testing performance is 22.51 %. Even though the network topology 20-5-2 has higher average percentage error during training, it has second best testing performance. The training and testing performance respectively are 21.54% and 25.72%. For topology 20-9-2, 20-11-2, and 20-15-2, poor training performance result in poor testing performance. So, 20-7-2 is considered to be better network topology because of its less percentage error during testing.

The results of the third Scenario are shown in Table 3. In this scenario the percentage error for scale varies between 0.3295% to 1.057% and .8849% to 2.567% during training and testing. But the result for shape varies. The network topology 20-11-2 has the minimum percentage error for the shape during training which is 2.887%, however, the testing performance of this network is 42.66%. In another network topology, 20-9-2 the average percentage error for the training is 6.730%, which is higher than the network topology 20-11-2. But the testing performance of this topology is better than any other network topologies. The testing performance is 31.79%. Network topology 20-5-2 and 20-6-2 has a similar average percentage error during training; the testing performance is good for 20-6-2. So, 20-9-2 is better network topology because of its less percentage error during testing.

The results of the fourth Scenario are shown in Table 4. In this scenario the percentage error for scale varies between .6588% to 1.072% and 1.078% to 1.495% during training and testing. This error is acceptable. There is variation in average percentage error in the shape parameter. During training the network topology 20-11-2 has a minimum average performance error for the shape which is 11.61%. The testing performance found in this network is 47.10%, which is better than any other network topologies. The second best testing performance found in network topology of 20-5-2. For network topology of 20-7-2 and 20-8-2, poor training performance results in poor testing performance. So in this scenario, 20-11-2 is considered better network because of its testing performance.

Considering the above mentioned scenario, it can be stated that by increasing the training examples, the prediction capacity of the network can be improved. It is due to the fact that when the network is presented with a lot of examples, it is likely to memorize less from training data. Another point is that less percentage error during training does not necessarily mean that it will predict well. It happens because sometimes there is overtraining done to get better training performance. When the network is tested with new data, it tries to memorize from the training data. As a result the prediction performance got worse. Considering this situation, it is not wise to select network only by looking at average percentage error during training unless the network is tested with new data.

9. Conclusion

In this report feed forward neural network with back propagation is used to estimate the two parameters of a Weibull distribution. The feed forward neural is configured with different hidden neurons to create various network topologies. The feed network is trained and tested on the four scenarios where simulated failure data generated using *wblrnd* function. The training and testing performance is measured through average percentage error. The network performed well in the second scenario as it has higher no of training examples.

It is observed that how a simple network can calculate the shape and scale without any mathematical model. The network requires only a set of logical training examples which is based on input and output. In the study, the input is considered time to failure data and output is taken shape and scale parameters. From that, it can comprehend the complex nonlinear connection through training. The trained network has also the capacity to predict the shape and scale on a new set of data. The prediction of shape parameter helps experts to make inference on the failure attribute of a population.

Although the neural network has the benefit in understanding the relationship between input and output during training, it has drawbacks. Firstly, it needs to train again when there is a change in network configuration. Secondly, overtraining sometimes degrades the performance of the network as this happened in both the scenarios. Because of this overtraining, the network may result in overfitting on the testing data which results in poor prediction. Finally, it is termed as a black box system. It is very difficult to explain physical explanation of the network. The model developer has the difficulty in finding the optimal neural network architecture.

The proposed network can be extended for estimation of a three- parameter of the Weibull distribution. It can be used to calculate the reliability of repairable systems where failure time follows a non-homogenous Poisson process. It can be further be used for censored data.

Appendices

Sample of training and test data

Scenario 1 consists of training and test data sets. Training data is marked as 200-20. Testing data set marked as 50-20. The sample data set is shown in the screenshot.

Shape	Scale	log(Scale)	TTF1	TTF2	TTF3	TTF4	TTF5	TTF6	TTF7	TTF8	TTF9	TTF10	TTF11	TTF12	TTF13	TTF14	TTF15	TTF16
0.9	400	2.602	68.73	30.61	894.67	27.76	168.10	1022.62	525.48	228.24	12.26	9.87	791.21	8.08	12.37	278.89	75.42	841.39
1.2	380	2.580	185.15	1036.10	84.04	40.59	172.46	130.53	138.12	359.60	185.30	610.10	157.70	1065.81	468.01	969.10	769.44	96.99
1	320	2.505	263.63	308.32	85.51	73.33	536.75	228.43	258.68	139.67	109.88	90.06	411.92	123.55	135.35	581.24	681.17	222.86
0.95	120	2.079	32.13	166.65	80.12	40.70	12.37	4.22	70.45	245.77	236.07	165.44	18.99	167.06	22.68	172.61	7.67	126.31
1.1	420	2.623	437.18	90.73	238.13	263.31	45.35	515.33	131.23	133.19	407.16	250.32	993.77	1112.46	277.29	118.90	36.60	803.37
1.2	600	2.778	987.84	176.50	682.54	412.39	978.26	340.96	763.68	293.81	263.31	213.96	496.79	1278.57	829.08	81.15	1015.98	151.26
1.25	550	2.740	1047.80	40.91	2111.64	184.33	152.83	114.59	1134.41	513.09	698.24	165.61	478.71	82.68	842.67	692.01	929.66	955.54
1.3	700	2.845	1744.32	138.75	110.48	224.44	1335.37	876.69	749.33	336.62	1188.94	296.12	1300.45	362.46	534.88	240.74	302.11	120.22
1.15	1200	3.079	2624.10	1545.33	324.03	3595.97	124.42	438.57	897.68	710.45	1646.24	965.79	69.23	773.69	827.01	1670.37	897.03	624.11
1.16	900	2.954	187.09	561.74	94.25	374.32	529.05	228.69	153.95	18.66	5145.72	169.84	486.68	17.13	611.96	690.17	245.38	1261.22
1.3	710	2.851	283.34	438.49	919.46	354.81	1429.23	396.22	360.34	292.01	135.12	32.10	253.86	443.39	96.20	444.86	2092.34	1262.41
1.19	620	2.792	400.04	324.15	1751.96	338.46	627.79	1562.99	467.27	943.24	1154.18	911.74	1072.95	951.90	1628.10	319.20	756.07	414.21
1.5	1200	3.079	1960.37	957.39	352.19	315.42	1435.40	1619.62	825.89	700.28	1097.46	1627.87	170.32	2210.52	2058.69	1874.05	1771.06	732.07
1.32	1100	3.041	446.33	2372.48	261.76	143.28	47.33	264.06	374.73	809.15	1664.91	1032.51	1867.05	2827.67	135.11	1262.63	1278.03	1182.20
1.36	1800	3.255	1208.30	462.87	1873.45	1537.96	3952.27	2694.58	936.57	1938.61	348.11	3144.78	68.15	1260.86	826.29	6.81	2115.10	1639.38
1.5	2200	3.342	3164.72	2233.12	4442.90	1651.33	2331.65	3181.62	2966.36	472.87	1179.29	1829.41	448.25	3792.38	971.92	999.36	1523.90	3123.56
1.6	1300	3.114	329.05	2384.04	1616.27	2541.66	1145.79	3244.89	324.38	1761.74	2229.96	1441.46	1117.59	2179.56	45.12	1381.63	1466.64	2462.98
1.65	1900	3.279	1187.29	3237.32	3322.80	824.08	469.71	1432.90	3076.59	697.52	1995.66	2147.89	902.18	4773.33	3717.92	1096.30	1255.92	1452.83
1.7	2000	3.301	1109.76	1460.17	1910.38	3655.09	881.19	2099.39	1326.87	983.90	3227.22	3056.58	1478.99	1652.79	563.26	830.48	1001.90	3793.80
2.1	1700	3.230	1072.66	2378.46	995.04	2476.79	2442.99	1156.32	1788.35	1130.99	941.07	1266.86	959.82	2028.31	970.20	319.29	672.87	2605.10
2.2	2250	3.352	1168.38	2250.63	2769.80	3378.53	1217.28	2771.17	2193.99	1776.48	2684.13	1554.43	1943.62	3001.32	1189.39	3283.27	2466.47	2654.04
2.25	3000	3.477	4247.57	1599.19	3292.05	2214.28	686.57	2773.67	1914.97	1695.48	2773.14	2045.30	4267.18	911.49	3772.15	3398.08	1548.60	2589.67
2.4	3200	3.505	2174.34	2983.12	2907.75	2386.33	4931.42	3395.09	1819.18	2094.93	4339.07	4306.05	4594.31	6178.94	3005.27	2234.27	2002.00	2644.15
2.5	1400	3.146	1959.20	1829.35	1764.09	1701.49	1479.13	1480.83	1657.55	1593.49	585.80	922.11	1131.67	1727.15	1668.69	2038.86	534.69	916.90

Figure19. Screenshot for training data of Scenario1

Shape	Scale	Log(Scale)	TTF1	TTF2	TTF3	TTF4	TTF5	TTF6	TTF7	TTF8	TTF9	TTF10	TTF11	TTF12	TTF13	TTF14	TTF15	TTF16
1.5	2200	3.342	764.64	470.64	3566.01	443.81	1307.74	3863.80	2591.32	1571.15	271.79	238.72	3312.56	211.69	273.27	1771.93	808.52	3437.04
1.9	1900	3.279	1206.55	3579.98	732.62	462.64	1153.64	967.53	1002.69	1834.91	1207.15	2562.24	1090.26	3644.47	2167.18	3431.97	2966.64	802.04
1.68	1000	3.000	891.06	978.11	455.86	416.04	1360.51	818.19	881.06	610.50	529.27	470.18	1162.19	567.53	599.19	1426.56	1567.83	806.26
1.75	2300	3.362	1124.80	2748.84	1847.11	1278.79	670.04	373.60	1722.50	3394.28	3320.90	2738.00	845.36	2752.51	931.07	2801.83	516.93	2364.86
1.9	700	2.845	716.44	288.28	503.99	534.20	192.96	788.00	356.94	360.03	687.53	518.78	1152.51	1230.30	550.44	337.13	170.42	1018.98
2.3	3200	3.505	4150.73	1690.04	3422.59	2631.39	4129.68	2382.81	3629.17	2204.80	2082.25	1868.56	2899.87	4748.75	3788.15	1126.76	4212.00	1559.31
2.45	1900	3.279	2639.78	504.64	3774.36	1087.75	988.55	853.49	2748.94	1833.84	2146.01	1029.92	1770.08	722.54	2362.07	2136.22	2483.48	2518.51
2.5	2800	3.447	1341.75	2078.43	2852.03	2381.26	2698.45	4089.17	3228.38	3762.39	3456.64	3228.24	2653.19	4346.71	1125.09	888.96	2443.71	2448.22
2.75	3000	3.477	1807.22	2935.84	3408.08	2894.87	4085.95	3877.58	1076.49	970.23	2418.65	4372.05	3433.23	3043.98	1661.73	5043.65	4551.19	3698.27
2.9	1700	3.230	1428.13	1818.76	1115.63	2027.27	1213.00	2039.46	1699.04	1309.38	1051.34	2335.46	690.18	1059.75	1517.88	1594.50	1577.94	1801.44
3	1900	3.279	1444.64	1881.61	1127.14	1628.42	1929.77	755.77	968.52	1600.34	1481.48	1540.17	2208.96	2018.95	1728.61	2159.16	1050.96	2238.78
3.22	2500	3.398	2623.27	1139.73	2371.28	2941.60	1222.69	747.09	2353.73	3192.33	2747.07	2415.11	2039.76	2736.99	2023.43	1789.56	2839.13	3167.23
3.3	3800	3.580	4991.12	4150.13	2407.88	5570.37	1724.95	2675.75	3434.42	3165.57	4242.63	3523.08	1406.23	3261.05	3337.67	4264.19	3433.56	3025.81
3.44	2700	3.431	3812.54	1464.70	1343.85	1756.61	3446.42	2939.70	2770.38	2047.42	3298.42	1950.59	3412.08	2105.44	2438.98	1803.78	1965.41	1387.45
3.52	2100	3.322	2994.92	1485.45	1892.31	1923.48	1092.16	1719.25	1706.58	1228.45	1359.31	1772.34	2441.14	2323.40	1151.05	3010.20	1908.00	2475.45
3.6	3300	3.519	4401.23	2527.33	4542.67	4320.88	2928.78	4176.95	2112.44	2114.58	2415.30	3942.93	2586.66	2936.09	1214.91	2614.29	2174.41	3090.96
3.66	1700	3.230	1981.54	1671.12	1071.54	1122.57	2253.51	1660.79	1505.36	1639.17	1341.50	1379.29	1799.35	1620.99	2510.85	549.86	1992.75	2119.66
3.77	2300	3.362	1036.70	1188.61	3062.65	1677.11	2472.08	2210.33	2010.07	1085.50	2218.54	781.59	2413.50	1747.67	1810.82	2024.19	1753.23	1810.49
3.8	800	2.903	1105.95	692.43	463.54	629.30	913.92	799.41	748.09	280.08	941.29	490.67	644.12	795.21	913.54	766.04	736.35	974.33
3.9	1800	3.255	1954.64	1900.64	1493.39	1935.42	1180.74	637.82	1337.86	1830.39	1535.28	2210.39	993.20	1063.04	1193.18	1941.86	1522.34	2533.67
4.15	1700	3.230	1639.63	2091.21	1447.63	1587.70	1331.23	1326.20	1401.38	2281.94	2155.14	1754.78	1522.77	1382.45	1656.24	1151.39	1302.20	740.90
4.28	4800	3.681	3471.94	4632.93	5888.80	5123.94	5558.02	5075.11	4583.55	4325.15	4532.19	2996.51	4352.32	2468.16	3982.79	2304.05	5213.39	3855.27
4.4	3600	3.556	3865.29	3945.07	2929.37	2403.83	3652.50	2621.78	2910.76	5190.70	3085.32	3558.21	2070.76	5559.43	3393.60	3476.11	3396.92	2653.26
4.89	2300	3.362	2575.24	1828.94	2167.19	2616.69	2334.49	1994.88	2548.52	1801.99	2469.32	1393.24	2431.65	1756.00	2553.62	2406.02	2749.92	2036.29

Figure20. Screenshot for testing data of Scenario 1

Scenario 2 consists of training and test data sets. Training data are marked as 500-20. Testing data set marked as 50-20. The sample data set is shown in the screenshot.

Scale	Shape	log(Scale)	TTF1	TTF2	TTF3	TTF4	TTF5	TTF6	TTF7	TTF8	TTF9	TTF10	TTF11	TTF12	TTF13	TTF14	TTF15	TTF16
300	0.5	2.477	12.60	2.94	1277.62	2.46	63.01	1625.16	490.25	109.27	0.57	0.38	1024.10	0.27	0.57	156.75	14.89	1143.95
250	0.52	2.398	22.52	455.51	119.52	34.67	3.94	0.55	94.48	926.31	860.62	449.49	8.61	457.56	11.92	485.74	1.64	274.53
350	0.55	2.544	379.22	16.33	112.51	137.57	4.08	526.91	34.17	35.20	328.92	124.33	1959.49	2455.48	152.56	28.05	2.66	1280.55
380	0.57	2.580	1085.54	28.91	498.46	172.56	1063.49	115.62	631.44	84.53	67.11	43.35	255.38	1868.60	750.69	5.63	1151.66	20.89
400	0.6	2.602	1531.85	1.78	6595.71	41.02	27.76	15.24	1807.47	346.11	657.63	32.82	299.54	7.72	972.96	645.46	1193.93	1264.21
600	0.62	2.778	30.90	180.42	646.24	312.23	516.96	2762.89	1065.25	1974.76	1403.09	1065.07	482.88	3534.47	15.19	5.87	346.59	349.17
425	0.65	2.628	49.79	387.85	728.99	365.46	1570.50	1258.53	5.56	3.58	170.85	2091.18	752.03	451.99	34.91	3827.82	2478.40	1030.07
450	0.7	2.653	218.61	595.26	78.59	933.22	111.15	956.69	448.94	152.57	61.46	1677.29	10.75	63.52	281.41	345.09	330.49	572.13
600	0.75	2.778	200.53	577.11	74.31	323.74	638.50	15.02	40.51	301.99	221.78	259.07	1096.20	764.97	411.08	1000.64	56.17	1156.60
700	0.8	2.845	849.63	29.65	565.84	1347.22	39.34	5.42	549.18	1872.51	1022.92	609.13	308.63	1007.89	298.81	182.26	1168.04	1813.96
1000	0.82	3.000	2996.03	1425.76	159.43	4660.71	41.65	243.74	665.59	479.44	1558.02	737.48	18.30	540.35	593.29	1590.13	664.92	399.78
650	0.85	2.813	2626.43	54.70	38.60	114.12	1745.49	917.09	721.35	212.14	1461.41	174.37	1676.18	237.55	430.74	127.04	179.80	43.93
800	0.9	2.903	3206.62	206.54	532.35	567.48	62.03	365.84	355.41	98.25	145.97	412.05	1441.36	1187.98	76.17	3271.10	549.83	1522.22
900	0.94	2.954	2711.41	324.01	3060.60	2526.68	569.84	2219.23	163.04	163.68	272.36	1779.54	354.12	575.31	19.60	368.82	182.13	700.48
1200	0.92	3.079	2207.76	1120.92	191.33	230.23	3682.68	1093.63	739.76	1038.07	467.72	522.37	1504.21	993.01	5662.22	13.46	2257.84	2886.46
850	0.95	2.929	35.98	61.90	2648.33	242.71	1131.80	725.91	497.97	43.18	736.66	11.73	1029.05	285.83	329.07	511.99	289.46	328.83
900	1	2.954	3081.08	519.91	113.14	361.54	1492.61	897.47	697.46	16.68	1669.78	140.44	394.98	879.70	1490.29	763.24	656.79	1903.66
490	1.1	2.690	656.30	594.24	252.73	633.70	109.89	12.38	171.13	519.97	278.78	1014.94	59.52	75.73	114.05	641.22	270.54	1646.70
550	1.12	2.740	481.03	1184.82	303.22	426.96	222.26	219.17	268.85	1637.24	1324.69	618.59	365.76	255.64	499.34	129.81	204.83	25.34
600	1.14	2.778	177.84	525.28	1292.66	766.72	1040.46	739.64	504.56	405.79	483.66	102.31	415.45	49.39	297.74	38.15	818.19	263.50
650	1.16	2.813	851.22	919.81	297.39	140.48	686.69	195.24	290.28	2604.51	362.05	621.84	79.78	3378.91	519.58	569.15	521.51	204.29
1000	1.18	3.000	1597.46	386.85	781.55	1706.75	1063.63	554.43	1529.90	363.77	1342.27	125.26	1259.43	326.82	1542.61	1205.33	2096.74	603.71
1200	1.2	3.079	599.06	544.14	620.17	109.27	1743.60	492.56	1628.88	2249.22	672.05	994.51	974.85	573.81	391.78	1249.00	573.69	1075.36

Figure21. Screenshot for training data of Scenario 2

Shape	Scale	log(Scale)	TTF1	TTF2	TTF3	TTF4	TTF5	TTF6	TTF7	TTF8	TTF9	TTF10	TTF11	TTF12	TTF13	TTF14	TTF15	TTF16
0.75	550	2.740	66.44	25.17	1445.05	22.38	194.34	1696.47	763.06	280.51	8.39	6.48	1246.93	5.09	8.49	356.79	74.28	1342.41
0.8	750	2.875	255.09	3376.66	78.00	26.18	229.32	151.00	164.36	690.41	255.39	1525.78	200.52	3522.92	1025.12	3054.48	2160.97	96.71
0.9	900	2.954	725.66	863.57	207.68	175.10	1598.89	618.83	710.53	358.26	274.44	220.02	1191.50	312.62	345.96	1746.84	2083.55	602.09
0.95	850	2.929	227.60	1180.42	567.53	288.28	87.65	29.88	499.01	1740.89	1672.19	1171.86	134.49	1183.33	160.67	1222.68	54.35	894.68
1.2	1300	3.114	1348.67	319.08	772.75	847.36	168.99	1568.10	447.52	453.67	1263.51	808.95	2862.93	3174.86	888.51	408.83	138.82	2355.79
1.25	1100	3.041	1775.27	339.82	1244.89	767.46	1758.74	639.38	1386.64	554.26	498.90	408.78	917.68	2274.17	1500.45	161.17	1823.79	293.03
1.4	2100	3.322	3733.74	206.36	6980.35	791.28	669.35	517.60	4008.10	1973.71	2598.70	719.12	1855.18	386.74	3073.70	2577.98	3355.49	3438.76
1.5	2400	3.380	704.26	1460.52	2474.79	1832.14	2256.69	4511.69	3042.67	3926.94	3409.61	3042.46	2193.96	4995.16	525.12	354.61	1912.92	1918.81
1.85	3500	3.544	1647.69	3389.31	4230.60	3319.24	5540.03	5125.32	762.81	653.62	2541.03	6126.38	4277.10	3576.55	1454.43	7576.26	6503.20	4777.00
2	3200	3.505	2407.82	3418.88	1683.11	4001.58	1900.25	4036.53	3097.45	2123.02	1544.32	4913.04	838.90	1562.27	2630.29	2824.98	2782.55	3371.78
2.5	2000	3.301	1439.59	1976.80	1068.81	1662.05	2037.66	661.60	890.96	1627.72	1483.75	1554.56	2396.35	2151.18	1785.51	2331.67	982.72	2435.23
2.65	1400	3.146	1484.32	539.03	1312.90	1705.95	587.07	322.65	1301.11	1884.22	1569.86	1342.45	1093.35	1562.86	1082.73	932.62	1634.01	1866.24
2.7	1600	3.204	2232.80	1781.99	916.09	2553.47	609.38	1042.15	1413.93	1279.85	1830.65	1458.67	474.74	1327.19	1365.40	1842.03	1413.50	1211.13
2.75	2500	3.398	3849.37	1163.26	1044.47	1460.19	3392.67	2780.66	2581.79	1768.62	3211.42	1664.62	3350.44	1831.54	2201.44	1509.40	1680.46	1087.03
2.95	1500	3.176	2291.10	992.38	1324.72	1350.80	687.54	1181.47	1171.09	791.11	892.67	1225.14	1795.13	1692.31	732.00	2305.06	1337.84	1825.28
3.2	2050	3.312	2834.30	1518.52	2936.98	2776.16	1792.45	2672.34	1241.10	1242.52	1443.01	2504.51	1558.68	1797.49	666.10	1577.42	1282.14	1904.50
3.45	1950	3.290	2294.25	1914.87	1195.07	1255.53	2629.64	1902.32	1714.00	1876.06	1516.76	1562.12	2071.10	1853.99	2949.28	588.84	2308.01	2464.24
3.55	3450	3.538	1480.12	1711.45	4676.24	2466.90	3724.74	3307.33	2990.03	1554.22	3320.38	1096.53	3631.08	2577.27	2676.28	3012.34	2585.98	2675.75
3.65	3100	3.491	4342.99	2667.29	1756.38	2414.60	3560.86	3097.61	2890.85	1039.50	3671.98	1863.51	2473.84	3080.68	3559.34	2963.12	2843.66	3806.26
3.7	1950	3.290	2126.98	2065.09	1601.59	2104.94	1250.32	653.29	1426.29	1984.72	1648.98	2421.33	1041.94	1119.30	1264.21	2112.33	1634.34	2796.00
4.1	3400	3.531	3277.81	4192.99	2889.59	3172.76	2654.52	2644.39	2796.17	4580.29	4322.77	3510.93	3041.46	2757.94	3311.42	2291.85	2595.94	1466.87
4.25	2800	3.447	2020.67	2701.87	3440.09	2990.34	3245.54	2961.64	2672.86	2521.15	2642.71	1742.16	2537.10	1433.02	2320.24	1337.09	3042.92	2245.43
4.52	1750	3.243	1875.42	1913.09	1431.82	1181.12	1774.84	1285.25	1422.96	2498.86	1505.97	1730.22	1021.51	2671.50	1652.26	1691.35	1653.83	1300.27
4.7	3200	3.505	3599.35	2521.14	3007.98	3659.65	3249.95	2759.56	3560.51	2482.50	3445.45	1899.53	3390.78	2416.63	3567.92	3353.61	3853.70	2819.18

Figure22. Screen shot for testing data of Scenario 2

Scenario 3 consists of training and test data sets. Training data are marked as 200-30. Testing data set marked as 50-30. The sample data set is shown in the screenshot.

Shape	Scale	log (Scale)	TTF16	TTF17	TTF18	TTF19	TTF20	TTF21	TTF22	TTF23	TTF24	TTF25	TTF26	TTF27	TTF28	TTF29	TTF30
0.9	400	2.60206	841.3865	339.7328	26.8791	79.24677	11.60969	153.3659	1523.644	53.49585	20.27092	139.5129	96.23102	103.7625	371.6209	153.5278	752.0054
1.2	380	2.5797836	286.9325	318.2656	190.435	155.9319	132.1158	469.0003	171.9359	185.5134	624.8721	713.188	281.0899	26.57986	404.4427	225.8788	531.8504
1	320	2.50515	335.9579	520.5146	442.2299	155.0195	239.3758	334.4294	59.30615	171.4215	191.4687	27.65972	400.7445	89.00082	90.47113	309.2517	181.1033
0.95	120	2.0791812	58.76774	162.7459	48.69705	42.40043	32.62273	94.5427	312.0481	180.5463	9.586796	233.4031	21.05174	72.46037	0.346942	321.2942	96.74167
1.1	420	2.6232493	786.7645	70.34224	241.9809	263.215	764.046	78.91244	213.3571	437.9476	290.6428	386.1767	993.2537	580.4483	821.9666	677.9444	580.3923
1.2	600	2.7781513	1080.272	57.29377	45.15164	366.2438	1422.281	817.3378	620.3506	154.9526	1973.358	1559.374	969.1936	298.2008	227.5223	299.4149	496.3484
1.25	550	2.7403627	629.1569	402.2169	400.1029	152.5882	169.5034	284.957	537.3133	157.0725	379.8333	570.9102	60.18577	109.1481	364.3038	302.7075	332.2914
1.3	700	2.845098	35.13876	602.8996	1282.529	884.0516	642.5935	422.8956	876.037	414.5591	305.8174	959.2574	1257.701	813.1831	775.934	621.9991	518.8649
1.15	1200	3.0791812	624.1072	525.5393	1124.05	1201.257	25.80779	3368.442	192.594	148.8589	331.6956	2490.412	1547.675	1296.021	524.5093	2183.992	453.752
1.16	900	2.9542425	490.4606	479.5756	176.8607	240.4498	537.8814	1421.076	1223.127	145.167	2683.878	672.7982	1482.55	32.92738	353.9268	655.4106	704.5095
1.3	710	2.8512583	592.3165	619.9893	199.4973	1429.299	1217.592	1093.035	676.5586	193.6178	220.7123	1569.973	664.8592	504.1715	640.7763	364.4822	394.1286
1.19	620	2.7923917	546.6077	404.5817	57.45078	553.0651	20.29366	722.2163	259.7418	290.6587	413.6534	262.3691	290.488	980.443	1135.972	1.741663	999.5846
1.5	1200	3.0791812	1977.331	784.3808	1562.919	1164.132	795.3494	1486.771	1382.325	738.4529	1449.065	400.9391	80.85097	554.807	1253.388	793.5247	2046.888
1.32	1100	3.0413927	503.9101	599.2927	2775.628	2319.01	1215.347	778.1625	574.2066	1013.407	323.1095	475.7876	80.80007	777.3392	1201.435	2031.79	643.4548
1.36	1800	3.2552725	903.054	2109.839	913.8907	855.5356	3724.529	2265.562	2420.382	923.8973	487.3077	1886.311	645.2868	905.0373	5880.792	1092.702	1733.271
1.5	2200	3.3424227	1383.298	3073.905	992.9801	2773.263	429.2507	2637.698	912.729	3093.986	2548.159	3938.869	1479.13	1155.668	1572.069	1980.213	1271.479
1.6	1300	3.1139434	1197.352	432.7553	449.48	1576.072	830.9233	885.3598	959.2017	379.2896	1552.808	1415.169	2083.418	228.7899	775.6173	1072.52	786.323
1.65	1900	3.2787536	4357.295	2019.704	1730.715	2236.103	2549.333	708.4236	1714.59	523.0833	1828.393	844.7464	1811.535	743.2983	880.1886	1870.442	2460.926
1.7	2000	3.30103	1878.646	964.4342	756.8254	854.879	2164.766	1520.103	3354.179	3173.301	3000.575	1145.337	1625.345	2696.823	1625.815	2929.345	3742.031
2.1	1700	3.2304489	2048.33	1431.488	579.6832	1283.22	727.4518	962.673	1261.49	1995.039	1106.501	2621.456	1184.76	1117.148	980.8046	608.6922	250.0094
2.2	2250	3.3521825	3710.161	1930.833	2823.268	3148.944	2771.871	3027.055	2837.263	3792.974	1571.158	2504.917	1808.96	1420.597	1906.926	1815.859	2043.651
2.25	3000	3.4771213	2157.915	2310.432	4855.779	927.5308	1799.256	1767.274	4709.3	1292.228	907.395	473.8076	1298.881	1594.961	2505.422	3825.775	2890.612
2.4	3200	3.50515	4021.983	2209.884	3337.386	1261.249	4389.986	500.5498	2615.409	2058.438	135.7608	3506.308	3034.917	2863.694	1852.297	1638.679	4527.925

Figure23. Screenshot for training data of Scenario 3

Shape	Scale	Log(Scale)	TTF16	TTF17	TTF18	TTF19	TTF20	TTF21	TTF22	TTF23	TTF24	TTF25	TTF26	TTF27	TTF28	TTF29	TTF30
1.5	2200	3.342	3437.035	1994.661	435.347	832.8669	263.0747	1237.725	4908.138	657.9286	367.5452	1169.379	935.7843	979.0636	2104.974	1238.508	3213.063
1.9	1900	3.279	1591.104	1698.736	1228.165	1082.501	974.9153	2170.068	1151.402	1208.021	2601.234	2827.745	1570.564	354.1066	1976.299	1367.964	2349.459
1.68	1000	3.000	1029.391	1335.868	1212.355	649.5919	841.3147	1026.601	366.6535	689.6679	736.5985	232.8531	1143.313	466.8667	471.4423	979.8687	712.5956
1.75	2300	3.362	1561.059	2713.71	1409.621	1307.552	1134.109	2020.753	3863.985	2871.01	583.3617	3300.449	894.0993	1749.034	96.26196	3925.722	2046.133
1.9	700	2.845	1006.738	248.7801	508.6963	534.0811	989.804	265.9023	472.939	717.1651	565.6273	666.788	1152.166	844.206	1032.576	923.6068	844.1588
2.3	3200	3.505	4349.034	939.6221	829.8273	2473.427	5020.138	3760.05	3256.176	1579.037	5955.491	5267.044	4109.67	2221.917	1929.44	2226.632	2898.525
2.45	1900	3.279	2034.921	1619.629	1615.281	987.7581	1042.186	1358.463	1877.512	1002.464	1572.998	1936.518	614.4894	832.556	1539.851	1400.998	1469.26
2.5	2800	3.447	590.9028	2590.803	3836.207	3161.371	2678.144	2154.509	3146.435	2132.318	1820.318	3298.476	3797.408	3026.948	2954.037	2633.163	2396.268
2.75	3000	3.477	2282.392	2124.082	2919.084	3001.313	602.3267	4619.238	1395.917	1253.368	1752.23	4071.201	3336.796	3098.142	2122.34	3853.698	1997.546
2.9	1700	3.230	1333.502	1321.584	886.7567	1002.678	1383.651	2040.793	1921.943	819.4058	2631.825	1513.233	2075.657	452.6628	1170.354	1497.467	1541.369
3	1900	3.279	1756.501	1791.602	1096.094	2572.932	2400.265	2290.603	1860.69	1081.977	1145.161	2679.754	1846.678	1638.044	1817.389	1423.207	1472.261
3.22	2500	3.398	2386.266	2135.148	1037.872	2396.646	706.5195	2645.046	1812.591	1889.513	2152.718	1819.345	1889.103	2961.385	3126.992	285.1157	2982.621
3.3	3800	3.580	4768.411	3132.203	4284.943	3747.944	3152.036	4188.755	4052.34	3047.466	4140.129	2308.73	1115.027	2676.043	3875.934	3148.747	4843.938
3.44	2700	3.431	2001.092	2138.73	3851.302	3594.635	2805.316	2364.187	2103.924	2616.375	1687.36	1957.479	991.3604	2363.227	2792.95	3416.804	2197.885
3.52	2100	3.322	1608.721	2232.901	1616.152	1575.471	2781.202	2295.188	2354.562	1622.966	1267.564	2138.347	1412.819	1610.085	3317.977	1731.677	2069.572
3.6	3300	3.519	2719.899	3793.518	2368.989	3634.272	1670.324	3559.166	2287.25	3803.825	3508.317	4206.388	2796.88	2523.587	2868.805	3158.405	2626.032
3.66	1700	3.230	1639.959	1051.037	1068.605	1849.31	1397.89	1437.211	1488.433	992.1588	1837.327	1764.269	2089.265	795.4447	1356.426	1562.894	1364.579
3.77	2300	3.362	1606.992	1133.897	887.8356	2626.78	2754.737	1756.636	2890.376	2046.259	2038.324	1389.308	2111.1	2257.94	1801.754	1670.341	2054.843
3.8	800	2.903	1147.115	821.5072	768.2319	858.6284	908.9227	521.2522	765.1157	456.9332	786.766	562.6471	783.608	532.243	572.7781	794.5721	895.0995
3.9	1800	3.255	1751.55	1309.781	1178.444	1242.716	1863.198	1597.101	2255.049	2201.212	2148.16	1411.703	1644.39	2050.509	1644.598	2125.78	2365.214
4.15	1700	3.230	1854.497	1568.943	1028.952	1490.894	1143.97	1303.756	1479.058	1831.822	1391.287	2080.779	1436.372	1397.519	1315.158	1052.669	694.9481
4.28	4800	3.681	6207.137	4437.016	5393.962	5705.306	5343.262	5590.701	5407.689	6277.971	3990.925	5072.243	4290.779	3789.533	4408.691	4299.183	4568.437
4.4	3600	3.556	3041.817	3149.92	4605.193	1975.208	2771.822	2746.517	4533.623	2340.21	1953.164	1400.989	2346.363	2606.148	3283.169	4076.633	3532.267
4.89	2300	3.362	2573.115	1917.861	2347.946	1456.379	2686.092	925.3178	2083.19	1852.188	487.7186	2405.54	2240.98	2178.01	1758.706	1656.054	2727.189

Figure24. Screenshot for testing data of Scenario 3

Scenario 4 consists of training and test data sets. Training data are marked as 500-30. Testing data set marked as 50-30. The sample data set is shown in the screenshot.

Scale	Shape	log(Scale)	TTF16	TTF17	TTF18	TTF19	TTF20	TTF21	TTF22	TTF23	TTF24	TTF25	TTF26	TTF27	TTF28	TTF29	TTF30
300	0.5	2.477121	1143.95	223.594	2.32466	16.2772	0.51297	53.4226	3331.22	8.02397	1.3989	45.0525	23.0877	26.4415	262.781	53.5242	934.568
250	0.52	2.39794	130.735	166.059	50.763	32.0046	21.8325	406.293	40.0988	47.7865	787.806	1068.83	124.674	0.53954	288.68	75.2685	543.096
350	0.55	2.544068	382.38	847.655	630.259	93.7066	206.467	379.223	16.3331	112.509	137.568	4.08123	526.91	34.1667	35.1998	328.92	124.329
380	0.57	2.579784	657.67	5.7692	380.604	3036.2	86.2711	23.145	1869.73	5250.16	384.125	306.616	301.856	239.811	236.723	155.627	168.859
400	0.6	2.60206	15.2362	1807.47	346.108	657.628	32.8163	299.537	7.71817	972.956	645.457	1193.93	1264.21	15.1124	145.558	169.827	1198.09
600	0.62	2.778151	349.172	684.927	15.9011	596.414	2134.51	63.3652	545.136	1056.39	512.194	2361.93	1872.57	6.36609	4.01493	230.792	3188.87
425	0.65	2.628389	957.518	423.926	132.593	49.8005	1752.82	7.61671	51.6033	256.345	319.334	304.806	550.414	232.83	230.482	36.1027	44.1919
450	0.7	2.653213	909.084	793.61	1015.52	787.761	345.311	561.519	12.1334	352.867	950.987	16.7626	1.73862	341.013	1385.43	694.207	383.882
600	0.75	2.778151	128.193	384.466	268.59	974.311	430.091	7.56057	306.109	339.043	996.288	384.043	220.193	169.174	542.764	600.964	1.66504
700	0.8	2.845098	39.9725	47.1319	785.098	194.161	1279.8	3337.53	152.583	442.685	475.68	39.4265	290.283	280.989	66.1473	103.26	331.85
1000	0.82	3	2813.97	141.084	141.715	254.058	2184.69	343.261	598.718	12.4385	359.649	160.18	750.282	806.609	133.645	3032.05	2351.59
650	0.85	2.812913	1680.7	640.651	1145.65	437.305	711.832	18.9655	34.7831	2314.9	160.153	895.144	544.893	357.58	23.2584	553.926	5.41838
800	0.9	2.90309	797.498	602.646	9.51871	1589.75	101.553	320.395	779.974	1401.06	666.121	563.73	1839.02	393.852	1242.65	760.545	403.074
900	0.94	2.954243	3717.62	761.897	1056.31	1705.15	1604.19	767.204	2245.71	442.708	665.593	305.769	300.708	383.588	3301.48	2565.04	1035.28
1200	0.92	3.079181	1555.18	968.174	739.123	918.731	134.037	760.975	54.3647	503.623	39.4736	1762.37	432.868	1517.56	440.568	399.625	3515.72
850	0.95	2.929419	206.839	968.265	191.234	629.806	3016.32	1520.89	261.281	625.834	1651.19	917.69	408.555	1441.42	242.059	1225.21	64.3905
900	1	2.954243	309.152	1298.65	1912.78	448.853	718.382	701.373	371.316	234.895	944.279	371.228	789.02	154.853	164.539	1224.77	439.775
490	1.1	2.690196	4.90403	717.051	1022.58	1007.59	1231.25	447.45	401.001	492.505	148.941	244.457	143.391	43.4005	18.7663	772.566	909.35
550	1.12	2.740363	177.137	750.11	458.452	228.702	561.671	191.102	515.3	232.185	215.917	458.562	1868.11	601.801	479.354	699.158	848.11
600	1.14	2.778151	134.693	185.836	999.03	112.63	10.7286	419.338	95.4468	344.346	1037	911.186	546.528	202.206	140.865	168.928	675.182
650	1.16	2.812913	165.163	111.186	13.4754	3716.36	122.662	351.492	12.3744	441.974	498.454	177.22	910.883	476.158	92.6861	390.646	139.81
1000	1.18	3	1885.21	197.708	761.503	221.193	1460.45	642.828	519.965	2850.73	543.114	1012.68	2540.79	751.853	1526.77	1871.42	1475.36
1200	1.2	3.079181	1745.7	752.24	612.066	1073.22	1756.82	104.54	2575.27	2356.1	2094.98	1952.08	646.984	735.366	2960.15	132.837	460.128

Figure25. Screenshot for training data of Scenario 4

Shape	Scale	log(Scale)	TTF16	TTF17	TTF18	TTF19	TTF20	TTF21	TTF22	TTF23	TTF24	TTF25	TTF26	TTF27	TTF28	TTF29	TTF30
0.75	550	2.740363	1342.41	452.122	21.53716	78.82584	7.864577	174.0866	2737.48	49.18977	15.35108	155.3917	99.51048	108.9279	503.5133	174.3071	1173.156
0.8	750	2.875061	492.1023	574.8709	266.0762	197.1462	153.751	1028.361	228.2631	255.8284	1581.512	1928.377	477.1487	13.87442	823.5148	343.7154	1241.853
0.9	900	2.954243	950.0047	1545.259	1289.293	402.2583	651.8761	945.2034	138.3099	449.8184	508.6356	59.26464	1155.627	217.138	221.1274	866.4747	478.1341
0.95	850	2.929419	416.2715	1152.783	344.9374	300.3364	231.0776	669.6775	2210.341	1278.869	67.90647	1653.272	149.1165	513.2609	2.457507	2275.834	685.2535
1.2	1300	3.113943	2311.12	252.6851	784.2073	847.064	2249.871	280.7684	698.7357	1350.833	927.6365	1203.701	2861.563	1748.832	2405.735	2016.32	1748.677
1.25	1100	3.041393	1934.459	115.3853	91.80241	684.8366	2519.031	1480.038	1135.793	299.8875	3449.571	2751.692	1743.097	562.2067	433.6211	564.4038	916.9013
1.4	2100	3.322219	2367.875	1588.1	1580.645	668.404	734.1837	1167.439	2056.696	685.9154	1508.95	2171.141	291.2733	495.5917	1453.744	1232.157	1339.131
1.5	2400	3.380211	179.5317	2108.654	4056.187	2938.147	2228.458	1550.702	2915.048	1524.174	1170.919	3153.577	3988.044	2732.894	2624.064	2166.427	1851.427
1.85	3500	3.544068	2331.181	2094.929	3360.598	3502.278	321.7786	6648.269	1122.447	956.3706	1573.724	5510.335	4099.74	3671.549	2092.376	5078.472	1912.134
2	3200	3.50515	2179.974	2151.781	1206.5	1441.767	2299.849	4040.351	3703.689	1075.931	5842.295	2618.631	4140.822	455.0763	1804.143	2579.163	2689.524
2.5	2000	3.30103	1820.135	1863.869	1033.577	2877.661	2647.504	2503.026	1950.449	1017.624	1089.345	3021.618	1932.836	1673.849	1896.108	1413.993	1472.677
2.65	1400	3.146128	1322.993	1155.791	481.0714	1329.989	301.4845	1499.303	947.2226	996.2874	1167.358	951.5132	996.0246	1719.903	1837.465	100.0904	1734.902
2.7	1600	3.20412	2111.636	1263.381	1852.989	1573.252	1273.165	1802.277	1730.801	1221.733	1776.739	870.1986	357.5106	1042.286	1639.163	1271.542	2152.586
2.75	2500	3.39794	1718.704	1867.834	3898.387	3576.159	2622.574	2117.314	1829.887	2403.521	1388.543	1671.975	713.8865	2116.238	2608.121	3356.242	1932.681
2.95	1500	3.176091	1091.414	1613.953	1097.433	1064.552	2097.393	1667.817	1719.425	1102.956	821.2556	1532.741	934.7578	1092.519	2588.983	1191.671	1474.103
3.2	2050	3.311754	1649.292	2397.994	1411.917	2285.048	952.9641	2231.991	1357.23	2405.325	2196.15	2693.541	1701.899	1515.99	1751.214	1951.313	1585.397
3.45	1950	3.290035	1877.016	1170.825	1191.597	2132.165	1584.478	1631.801	1693.564	1101.366	2117.511	2028.297	2426.776	871.2014	1534.664	1783.579	1544.452
3.55	3450	3.537819	2357.518	1627.909	1255.466	3972.743	4178.564	2591.31	4397.389	3047.233	3034.686	2019.865	3149.876	3383.038	2662.047	2456.332	3060.811
3.65	3100	3.491362	4511.395	3186.813	2971.945	3336.87	3540.6	1984.605	2959.396	1730.328	3046.629	2148.949	3033.899	2028.189	2189.248	3078.106	3484.558
3.7	1950	3.290035	1894.716	1394.753	1247.748	1319.582	2022.233	1719.043	2472.914	2410.725	2349.523	1509.388	1772.738	2237.086	1772.973	2323.73	2600.418
4.1	3400	3.531479	3740.603	3113.421	1959.528	2943.849	2201.207	2540.867	2918.208	3690.436	2728.701	4244.432	2825.903	2742.118	2565.268	2009.156	1273.739
4.25	2800	3.447158	3627.407	2586.823	3149.07	3332.157	3119.263	3264.755	3157.141	3669.095	2325.008	2959.961	2500.974	2206.876	2570.193	2505.907	2663.992
4.52	1750	3.243038	1485.29	1536.65	2224.048	975.5948	1356.798	1344.738	2190.394	1150.684	964.9939	698.3155	1153.63	1277.79	1599.893	1975.166	1717.94
4.7	3200	3.50515	3596.263	2648.801	3269.433	1989.179	3760.688	1240.873	2886.774	2554.498	637.3272	3352.914	3114.61	3023.607	2420.496	2273.682	3820.571

Figure25. Screenshot for testing data of Scenario 4

Training and Testing result for Scenario 2

The topology 20-7-2 has the best prediction performance compared to other topologies in Scenario 2. Some snapshots are given during training and testing for the actual and network estimated Shape and Scale parameters, and their percentage error.

Table 3: Percentage error during training of 500-20 data set

Sl No	Actual output		Network output		Average Error	
	Scale Log(Θ)	Shape (β)	Scale Log(Θ)	Shape (β)	Scale Log(Θ)	Shape (β)
1	2.477	0.5	2.697	0.8303	0.088	.6606
2	2.398	0.52	2.648	1.488	.1045	1.862
3	2.544	0.55	2.654	0.5000	.0433	.0908
4	2.580	0.57	2.681	0.5368	.0394	.0580
5	2.602	0.60	2.791	.5478	.0728	.0869
6	2.778	0.62	2.815	.5361	.0132	.1352
7	2.628	0.65	2.749	0.5193	0.0462	0.2009
8	2.653	0.7	2.780	0.6701	.0481	.0426
9	2.778	.75	2.803	1.073	.0090	.4307
10	2.845	0.8	2.834	1.082	.0036	.3533
11	3.000	0.82	3.013	.6170	.0044	.2474
12	2.813	0.85	2.947	.6068	.0477	.2860
13	2.903	0.9	2.815	.5319	.0302	.4089
14	2.954	0.94	2.817	0.5380	.0464	.4275
15	3.079	0.92	3.327	.5002	.0805	.4562
16	2.929	.95	2.803	1.058	.0428	.1143
17	2.954	1	2.998	0.9296	.0149	.0703
18	2.690	1.1	2.739	.7593	.0182	.3096
19	2.740	1.12	2.791	.7601	.0187	.3212
20	2.778	1.14	2.747	1.565	.0109	.3728
21	2.813	1.16	2.825	.5260	.0046	.5465
22	3.000	1.18	3.046	.8137	.0156	.3104
23	3.079	1.2	2.926	.5293	.0494	.5588
24	2.978	1.24	2.883	.6143	.0315	.5045
25	3.041	1.26	2.877	.6995	.0539	.4447
26	3.230	1.28	3.378	.5216	.0457	.5924
27	3.130	1.3	3.061	.7448	.0218	.4270
28	3.146	1.32	3.172	.5462	.0083	.5861
29	3.190	1.34	2.935	.5110	.0797	.6186
30	3.312	1.35	3.244	1.087	.0204	.1948
31	3.114	1.36	3.253	.8703	.0448	.3600
32	3.204	1.38	3.255	.7397	.0161	.463
33	3.279	1.40	3.219	.5200	.0181	.6285
34	3.301	1.42	3.279	1.583	.0066	.1151

	Actual output		Network Output		Average Error	
	Scale log(Θ)	Shape (β)	Scale log(Θ)	Shape (β)	Scale log(Θ)	Shape (β)
35	3.312	1.44	3.276	1.130	.0106	.2149
36	3.3222	1.45	3.509	.6672	.0565	.5398
37	3.380	1.46	3.357	.9385	.0067	.3571
38	3.267	1.48	3.347	.5061	.0244	.6579
39	3.371	1.5	3.418	.5701	.0139	.6199
40	3.230	1.52	3.273	1.405	.0132	.0755
41	3.217	1.54	3.252	1.081	.0109	.2976
42	3.230	1.55	3.331	.5065	.0311	.6731
43	3.301	1.56	3.352	.8753	.0156	.4388
44	3.279	1.58	3.291	1.483	.0038	.0610
45	3.389	1.60	3.361	1.495	.0080	.0653
46	3.477	1.61	3.215	.7119	.0752	.5577
47	3.462	1.63	3.453	1.951	.0025	.1974
48	3.447	1.62	3.350	2.156	.0281	.3310
49	3.602	1.64	3.597	1.795	.0013	.0951
50	3.544	1.65	3.465	3.651	.0222	1.213
51	3.477	1.66	3.468	.5461	.0023	.6709
52	3.462	1.68	3.500	.5289	.0110	.6851
53	3.431	1.7	3.353	1.664	.0228	.0207
54	3.407	1.71	3.361	.5246	.0133	.6931
55	3.255	1.72	3.344	2.137	.0273	.2424
56	3.505	1.74	3.363	.5234	.0403	.6991
57	3.531	1.75	3.451	1.712	.0227	.0214
58	3.470	1.76	3.463	0.5045	.0017	.7133
59	3.477	1.78	3.506	.5533	.0083	.6880
60	3.491	1.80	3.512	1.651	.0060	.0822
61	3.613	1.82	3.492	1.926	.0334	.0582
62	3.580	1.84	3.550	2.200	.0082	.1960
63	3.556	1.85	3.408	1.030	.0416	.4430
64	3.556	1.86	3.485	1.9119	.0199	.0279
65	3.498	1.90	3.500	.5144	.0006	0.7292
66	3.352	1.92	3.433	1.504	.0242	.2166
67	3.538	1.94	3.469	.5018	.0194	.7413
68	3.519	1.95	3.424	.8091	.0268	.5850
69	3.389	1.96	3.412	1.813	.0069	.0749
70	3.447	1.98	3.463	1.708	.0048	.1372
71	3.477	2	3.404	1.809	.0209	.0953
72	3.447	2.11	3.427	.5914	.0057	.7197

Table 4: Percentage error during testing of 50-20 data set

Sl No	Actual output		Network output		Percentage Error	
	Scale log(Θ)	Shape (β)	Scale log(Θ)	Shape (β)	Scale log(Θ)	Shape (β)
1	2.74	0.75	2.739	.8935	0.0002	.1913
2	2.875	0.80	3.012	.6773	.0478	.1533
3	2.954	0.9	2.907	.6562	.0157	.2707
4	2.929	.95	2.812	1.320	.0399	.3897
5	3.113	1.2	3.031	.5003	.0264	.5830
6	3.041	1.25	2.950	.9308	.0298	.2553
7	3.322	1.4	3.477	.9563	.0467	.3169
8	3.380	1.5	3.387	1.172	.0022	.2181
9	3.544	1.85	3.551	4.213	.0020	1.277
10	3.505	2	3.500	0.6327	.0013	.6836
11	3.301	2.5	3.301	3.162	.0002	.2649
12	3.146	2.65	3.128	3.958	.0056	.4938
13	3.204	2.7	3.170	2.869	.0105	.0629
14	3.397	2.75	3.396	.9446	.0002	.6564
15	3.176	2.95	3.134	1.744	.0130	.3984
16	3.311	3.2	3.298	2.593	.0040	.1895
17	3.290	3.45	3.334	2.537	.0134	.2645
18	3.537	3.55	3.428	1.404	.0308	.6043
19	3.491	3.65	3.503	5.927	.0033	.6238
20	3.290	3.7	3.315	2.852	.0077	.2290
21	3.531	4.1	3.534	3.890	.0009	.0509
22	3.447	4.25	3.434	4.411	.0038	.0378
23	3.243	4.52	3.240	4.897	.0007	.0835
24	3.505	4.7	3.521	4.199	.0047	.1065
25	3.406	4.85	3.395	3.396	.0031	.2996

	Actual output		Network output		Percentage error	
	Scale log(Θ)	Shape (β)	Scale log(Θ)	Shape (β)	Scale log(Θ)	Shape (β)
26	3.505	5.15	3.474	4.901	.0087	.0482
27	3.290	5.3	3.283	6.015	.0019	0.1349
28	3.371	5.45	3.422	6.536	.0153	.1993
29	3.322	5.65	3.309	5.390	.0038	.0458
30	3.406	5.75	3.422	5.323	.0046	.0742
31	3.511	6.15	3.480	6.536	.0089	.0629
32	3.491	6.35	3.488	6.793	.0008	.0698
33	3.371	6.5	3.395	7.729	.0071	.1891
34	3.311	6.75	3.329	7.892	.0054	.1692
35	3.498	6.95	3.501	7.647	.0009	.1003
36	3.380	7	3.390	6.823	.0029	.0252
37	3.423	7.25	3.462	7.734	.0114	.0667
38	3.361	7.35	3.413	8.459	.0153	.1510
39	3.585	7.6	3.580	8.611	.0014	.1331
40	3.389	7.8	3.427	8.846	.0113	.1341
41	3.562	8.1	3.566	8.248	.0011	.0183
42	3.676	8.35	3.649	9.455	.0073	.1324
43	3.406	8.52	3.423	8.265	.0050	.0298
44	3.332	8.7	3.345	8.259	.0038	.0506
45	3.414	8.76	3.418	7.192	.0011	.1789
46	3.462	9.25	3.493	8.142	.0089	.1197
47	3.332	9.5	3.354	8.068	.0067	.1506
48	3.389	9.8	3.391	8.115	.0007	.1719
49	3.648	9.9	3.630	9.572	.0047	.0330
50	3.423	9.95	3.439	9.337	.0048	.0615

References

[1] Ebeling, C. E. (2004). *An introduction to reliability and maintainability engineering*. Tata McGraw-Hill Education.

[2] Nwobi, F. N., & Ugomma, C. A. (2014). A comparison of methods for the estimation of Weibull distribution parameters. *Metodoloski zvezki*, *11*(1), 65.

[3] Watkins, A. J. (1996). On maximum likelihood estimation for the two parameter Weibull distribution. *Microelectronics Reliability*, *36*(5), 595-603.

[4] Flygare, M. E., Austin, J. A., & Buckwalter, R. M. (1985). Maximum likelihood estimation for the 2-parameter Weibull distribution based on interval-data. *IEEE transactions on reliability*, *34*(1), 57-59.

[5] Zhang, L. F., Xie, M., & Tang, L. C. (2007). A study of two estimation approaches for parameters of Weibull distribution based on WPP. *Reliability Engineering & System Safety*, *92*(3), 360-368.

[6] Bütikofer, L., Stawarczyk, B., & Roos, M. (2015). Two regression methods for estimation of a two-parameter Weibull distribution for reliability of dental materials. *dental materials*, *31*(2), e33-e50.

[7] Murthy, K. S. R., & Rahi, O. P. (2014, December). Estimation of Weibull parameters using graphical method for wind energy applications. In *Power Systems Conference (NPSC), 2014 Eighteenth National* (pp. 1-6). IEEE.

[8] Indhurani, L., & Subburaj, R. (2015). An artificial neural network approach to software reliability growth modelling. *Procedia Computer Science*, *57*, 695-702.

[9] Liu, M. C., Kuo, W., & Sastri, T. (1995). An exploratory study of a neural network approach for reliability data analysis. *Quality and Reliability Engineering International*, *11*(2), 107-112.

[10]Alsina, E. F., Cabri, G., & Regattieri, A. (2016). A neural network approach to find the cumulative failure distribution: modeling and experimental evidence. *Quality and Reliability Engineering International*, *32*(2), 567-579.

[11]Smith, K. A. (1999). *Introduction to neural networks and data mining for business applications*. Eruditions Publishing.

[12]Haykin, S. (1999). *Neural networks: a comprehensive foundation*. Prentice Hall PTR.

[13] A quick introduction to neural network. https://ujjwalkarn.mequick-intro-neural-networks/. Web accessed on 2019/01/04.

[14] Singh, Y., & Chauhan, A. S. (2009). NEURAL NETWORKS IN DATA MINING. *Journal of Theoretical & Applied Information Technology*, *5*(1).

[15] Introduction. https://cimss.ssec.wisc.edu/wxwise/class/aos340/spr00/whatismatlab.htm. Web accessed on 2019/03/12.

[16] Neural Network Toolbox. https://www.spsc.tugraz.at/system/files/nnt_intro.pdf. Web accessed on 2019/01/07.

[17] Neural Network Toolbox. http://matlab.izmiran.ru/help/toolbox/nnet/backpr26.html.

Web accessed on 2019/01/12

[18] Mathworks. https://www.mathworks.com/help/deeplearning/ref/mapminmax.html. Web accessed on 2019/01/13.