Rare Met. (2014) 33(3):249–257
DOI 10.1007/s12598-013-0182-2

RARE METALS
www.editorialmanager.com/rmet

Artificial neural network approach for prediction of stress–strain curve of near β titanium alloy

Srinivasu Gangi Setti*, R. N. Rao

Received: 4 September 2012 / Revised: 9 March 2013 / Accepted: 15 July 2013 / Published online: 31 December 2013

Abstract In the present study, artificial neural network (ANN) approach was used to predict the stress–strain curve of near beta titanium alloy as a function of volume fractions of α and β. This approach is to develop the best possible combination or neural network (NN) to predict the stress–strain curve. In order to achieve this, three different NN architectures (feed-forward back-propagation network, cascade-forward back-propagation network, and layer recurrent network), three different transfer functions (purelin, Log-Sigmoid, and Tan-Sigmoid), number of hidden layers (1 and 2), number of neurons in the hidden layer(s), and different training algorithms were employed. ANN training modules, the load in terms of strain, and volume fraction of α are the inputs and the stress as an output. ANN system was trained using the prepared training set (α, 16 % α, 40 % α, and β stress–strain curves). After training process, test data were used to check system accuracy. It is observed that feed-forward back-propagation network is the fastest, and Log-Sigmoid transfer function is giving the best results. Finally, layer recurrent NN with a single hidden layer consists of 11 neurons, and Log-Sigmoid transfer function using trainlm as training algorithm is giving good result, and average relative error is 1.27 ± 1.45 %. In two hidden layers, layer recurrent NN consists of 7 neurons in each hidden layer with trainrp as the training algorithm having the transfer function of Log-Sigmoid which gives better results. As a result, the NN is founded successful for the prediction of stress–strain curve of near β titanium alloy.

Keywords Artificial neural network (ANN); Stress–strain curve; Titanium alloys

1 Introduction

The increasing range of application of titanium alloys is due to their very good combinations of high strength to weight ratio, low density, exceptional corrosion-resisting properties, etc. They are heat treatable and hot or cold deformable [1, 2] and has gained more and more applications in many fields [3–5]. Over the past decades, many new titanium-based materials have been developed for different ranges of application. When titanium alloys are subjected to different heat treatments, they will produce different microstructures (volume fraction of α, grain size and shape of α). With different percentages of α and β phases, several metastable microstructures can be formed. Thus, it is formed with the integration of several physical properties, functional performances, and processing properties [6]. It is well known [7–9] that the mechanical properties of titanium alloys depend essentially on the characteristics of the microstructure. However, because of different percentages of α and β phases, a lot of funds and time will be wasted by experiments. In order to reduce these numerical methods, soft computing techniques are adopted to get the required properties. In order to achieve this, some of the investigators used finite element method (FEM) to get the stress–strain curve based on the stress–strain curve of the individual phases [10–13].

Recently, with the developments of artificial intelligence, researchers have a great deal of attention to the solution of nonlinear problems in mechanical properties of alloys. Sha and Edwards [14] explained the use of artificial neural networks (ANN) in materials science-based

S. Gangi Setti*, R. N. Rao
Mechanical Engineering, National Institute of Technology, Warangal 506004, India
e-mail: nivassetti@gmail.com

research. Many of the researchers carried the work in the field of mechanical properties predicted by using ANN [15–18]. The main aim of this work was to design a model of ANN for prediction of the stress–strain curve of near beta titanium alloy as a function of volume fractions of α and β.

2 Artificial neural network (ANN)

ANN is a mathematical model consisting of a number of interconnected processing elements organized into layers; the geometry and functionality of which were likened to that of the human brain. Typically neural networks (NN) are adjusted or trained, so that a particular input leads to a specific target output [19, 20]. The training of NN is based on a comparison of the output and the target, until the network output matches the target. In general, many such input/target pairs are used to train a network. That means for reliable training and performance of any NN, we need an appropriate database. Using such a database, we can train NN to perform complex functions.

Commonly, NN modeling follows these steps: database collection; analysis and preprocessing of the data; training of the neural network—this includes the choice of architecture, training functions, training algorithms, and parameters of network; test of trained network; using trained NN for simulation and prediction.

In general, the network has one input layer, one hidden layer, and one output layer. The input layer consists of independent variables. In the present study, they are strain and the volume fraction of in the microstructure. Information from the input layer is then processed in the course of one hidden layer; following which output vector (stress) is computed in the final (output) layer. A schematic description of the layers used in the present study is given in Fig. 1. In developing an ANN model, the available dataset is divided into two sets; one is to be used for training of the network, and the remaining is to be used to verify the utility and capability of the network. For example, the architecture of ANN becomes 2-$[10]_1$-1, where 2 corresponds to the input values, 10 the number of hidden layer neurons and 1 the output (stress). In order to relieve the training difficulty and balance the importance of each parameter during the training process, the examinational data were normalized.

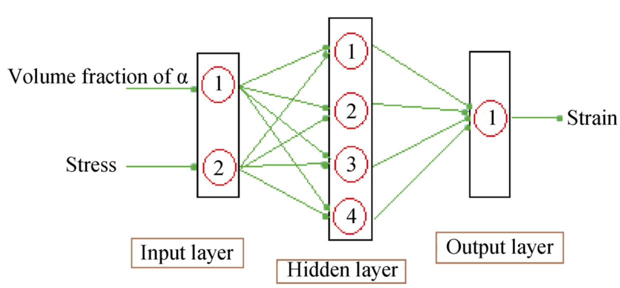

Fig. 1 Schematic representation of an artificial neural network

It is recommended that the data be normalized between slightly offset values such as 0.1 and 0.9. One way to scale input and output variables in interval [0.1, 0.9] is as

$$P_n = 0.1 + (0.9 - 0.1) \times (P - P_{min})/(P_{max} - P_{min}) \quad (1)$$

where P_n is the normalized value of P; P_{max} and P_{min} are the maximum and minimum values of P, respectively. After the neural network was trained, tested, and simulated, it is necessary for the simulating data to be un-normalized corresponded with normalization. The unnormalized method is as

$$P = (P_n - 0.1) \times (P_{max} - P_{min})/(0.9 - 0.1) + P_{min} \quad (2)$$

where P is the unnormalized value of P_n.

3 Methodology

3.1 Experimental data used for training

Initially, the alloy was developed in the vacuum arc remelting furnace. The chemical composition is shown in Table 1. Then the ingot was processed thermo-mechanically (forging and rolling); finally solution was treated at different temperatures in order to get different volume fractions of α and β. The blanks were cut from the work piece, then tested in ultimate tensile strength in order to get the stress–strain curve according to ASTM standard E-99.

In order to train the network, four stress–strain curves are used, in which three stress–strain curves (β, 84 % β and 60 % β) of Ti–10 V–4.5Fe–3Al were measured experimentally, and stress–strain curve of α was obtained from Ref [10]. Figure 2 shows the stress–strain curves which are used in this work.

3.2 Working platform

MATLAB platform was used to train and test the ANN. Among the various kinds of ANN approaches, the multilayer perceptron architecture with back-propagation learning algorithm becomes the most popular in

Table 1 Chemical composition of Ti–10V–4.5Fe–3Al alloy(wt%)

V	Al	Fe	O	N	P	C	S	Ti
9.6–10.0	3.0–3.2	4–5	0.11	0.009	0.01	0.02	0.001	Bal.

engineering applications; hence, two back-propagation (feed-forward back-propagation network and cascade-forward back-propagation network) and layer recurrent networks were used in the present work. The architecture of these three networks is shown in Fig. 3. For the transfer (or activation) function, different choices are possible [20]; the step and sign transfer functions are often used for classification and pattern recognition tasks; the sigmoid function transforms the input, which can have any value (also plus or minus infinity), into a value in the range between 0 and 1; the linear activation function provides an output equal to the neuron weighted input: it is often used for linear approximation problems. One of the most common choices for the transfer function is the step function and the sigmoid. Figure 4 shows the transfer functions and their corresponding equations. The mean square error (MSE) is considered as a measurement criterion for a training set. Table 2 shows the different parameters in the simulation. Table 3 shows the neural networks combinations with different types of neural network and transfer functions.

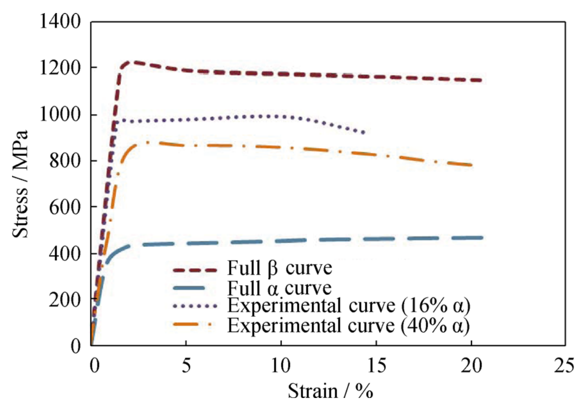

Fig. 2 Stress-strain curves of Ti–10V–4.5Fe–3Al at different volume fractions

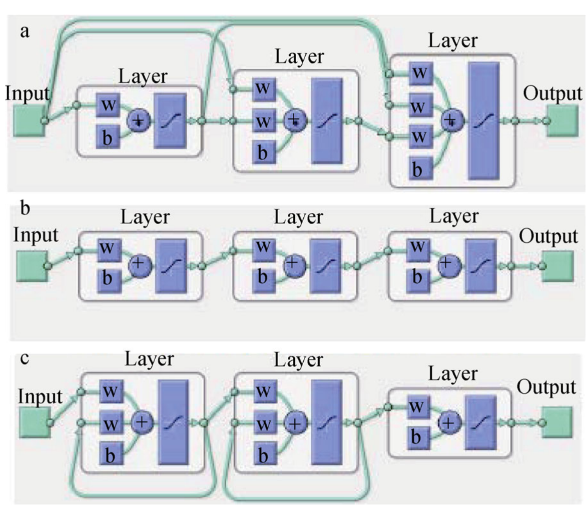

Fig. 3 Architecture of neural networks: **a** cascade-forward back-propagation network, **b** feed-forward back-propagation network, and **c** layer recurrent network

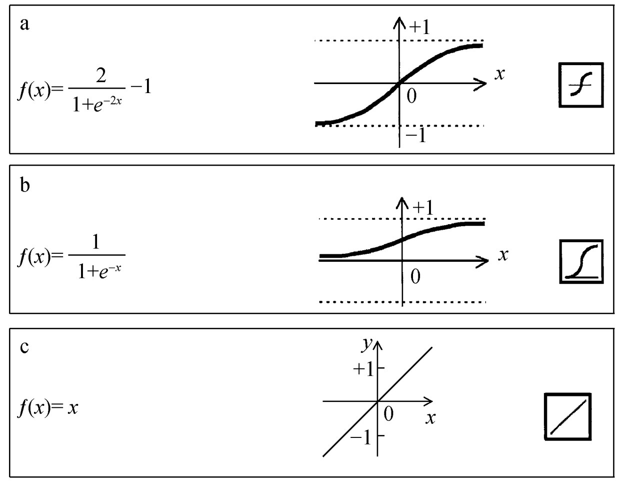

Fig. 4 Transfer functions: **a** Tan-Sigmoid transfer function, **b** Log-Sigmoid transfer function, and **c** Purelin transfer function

Table 2 List of parameters used in neural networking

ANN parameters
Network type: feed-forward back-propagation network, cascade-forward back-propagation network and layer recurrent network
Number of hidden layers: 1 and 2
Number of neurons: 10 (1st hidden layer) and 1 (2nd hidden layer)
Transfer function: tan-Sigmoid, Log-Sigmoid and Purelin
Training algorithm: trainlm
Best performance: MSE

4 Results and discussion

During the training period, the MSE decreases with the number of iterations increasing. After training, the stress–strain curves of 18 % and 24 % volume fraction of α curves are calculated and compared with the experimental results, presented in Table 4. The formula for MSE is as follows:

$$\mathrm{MSE} = \sum_{i=1}^{n} \left[\hat{Y} - Y\right]^2 \tag{3}$$

where i is the number of iterations from 1 to n; Y is predicted value and $\hat{Y}$ is true value.

Figure 5 shows the average relative errors for different neural networks. From Fig. 5 it is observed that for Networks 1, 3, 7, 9, 10, 12, 17 and 18, the average relative error is less than 10 %, but for Networks 1, 9, 12, 17 and

Table 3 List of neural networks (different neural networks and transfer functions)

Network Nos.	Network details	Network type	Transfer function
1	2-$[10]_1$-1	Cascade-forward back propagation	Tan-Sigmoid
2	2-$[10]_1$-1	Feed-forward back propagation	Tan-Sigmoid
3	2-$[10]_1$-1	Layer recurrent	Tan-Sigmoid
4	2-$[10]_1$-1	Cascade-forward back propagation	Purelin
5	2-$[10]_1$-1	Feed-forward back propagation	Purelin
6	2-$[10]_1$-1	Layer recurrent	Purelin
7	2-$[10]_1$-1	Cascade-forward back propagation	Log-Sigmoid
8	2-$[10]_1$-1	Feed-forward back propagation	Log-Sigmoid
9	2-$[10]_1$-1	Layer recurrent	Log-Sigmoid
10	2-$[10]_1$-$[1]_2$-1	Cascade-forward back propagation	Tan-Sigmoid
11	2-$[10]_1$-$[1]_2$-1	Feed-forward back propagation	Tan-Sigmoid
12	2-$[10]_1$-$[1]_2$-1	Layer recurrent	Tan-Sigmoid
13	2-$[10]_1$-$[1]_2$-1	Cascade-forward back propagation	Purelin
14	2-$[10]_1$-$[1]_2$-1	Feed-forward back propagation	Purelin
15	2-$[10]_1$-$[1]_2$-1	Layer recurrent	Purelin
16	2-$[10]_1$-$[1]_2$-1	Cascade-forward back propagation	Log-Sigmoid
17	2-$[10]_1$-$[1]_2$-1	Feed-forward back propagation	Log-Sigmoid
18	2-$[10]_1$-$[1]_2$-1	Layer recurrent	Log-Sigmoid

18, the standard deviation between the data points is within 5 %. Generally, triple layered networks give the best accuracy as compared with the two layer networks, in the present investigation, most cases obey the same. Tan-Sigmoid and Log-Sigmoid transfer functions are giving the better results as compared with Purelin transfer function in two and three layer neural networks. The average relative error is calculated by the following equation.

$$\text{Average relative error} = \left[\sum_{t=1}^{n}\left\{\frac{(\text{Difference between actual and predicted}) \times 100}{\text{Actual}}\right\}\right]\Big/ n \tag{4}$$

Table 4 Network training results

Network Nos.	Iterations	Time/s	Best performance (MSE)
1	236	5	0.0002
2	61	1	0.0007
3	42	23	0.0005
4	16	1	0.0615
5	9	1	0.0545
6	8	5	0.0602
7	21	1	0.0614
8	72	1	0.0006
9	60	33	0.0005
10	116	4	0.0003
11	44	1	0.0006
12	190	237	0.0004
13	1,000	30	0.0200
14	44	1	0.0004
15	46	71	0.0459
16	52	1	0.0010
17	80	2	0.0004
18	105	131	0.0004

Table 5 contains average relative error and standard deviation (SD) between errors. From above observations (Table 5), Network 18 is selected as the best network based on the average relative error. Thus, triple layer–layer recurrent neural with Logisg transfer function gives the best result. Network 18 is 2-$[10]_1$-$[1]_2$-1. The training plot, performance plot, and regression plot in the training, testing, validation, and combined are shown in Figs. 6, 7, and 8, respectively, for the Network 18, and also the comparison between the experimental calculated curve and ANN

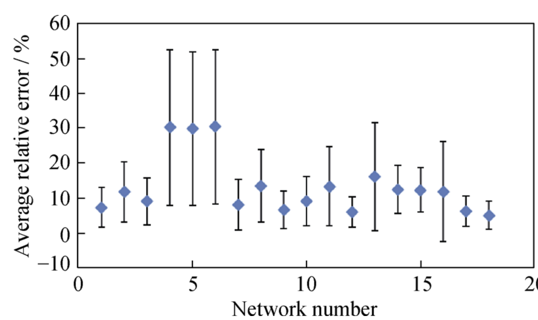

Fig. 5 Error variation plot

Table 5 Errors compared with experimental results

Network Nos.	Average relative error/%	SD
1	7.4	5.7
2	11.8	8.7
3	9.1	6.7
4	30.3	22.3
5	30.0	22.1
6	30.5	22.2
7	8.1	7.3
8	13.5	10.4
9	6.7	5.4
10	9.2	7.1
11	13.4	11.3
12	6.1	4.3
13	16.2	15.5
14	12.5	6.9
15	12.4	6.4
16	11.9	14.4
17	6.3	4.3
18	5.1	4.0

model predicted curve for the titanium alloy at two volume fractions of α, i.e., 18 % and 24 %, is shown in Fig. 9.

From Table 6, it is observed that all single hidden layer (SHL) ANNs with Tan-Sigmoid and Purelin transfer functions give medium and high average relative errors and standard deviation, respectively, as compared with double hidden layer (DHL) ANN networks. Layer recurrent NN with Logsig transfer function gives consistent results as compared with other combinations (Avg. relative error and SD are low).

Layer recurrent NN with Logsig transfer function was used for the further analysis. In the training, increased number of neurons (5–25) in the hidden layer was used to define the output accurately. The number of hidden layers is also increased to two hidden layers; in each hidden layer, the number of neurons is varied. After training the network successfully, it was tested by using the known data. Statistical methods were used to compare the results produced by the network. Table 7 shows the list of parameters. Table 8 shows the analysis of results.

The networks from 1 to 10 consist of only one hidden layer, in which the number of hidden neurons varies from 5 to 25. The MSE is very less, i.e., in the range of 0.0002–0.0007 for the networks 1–10 (Fig. 10). The average relative error for each network is also shown in Fig. 11. The Network 7 gives the best results (Fig. 11). It is clearly seen in Table 7 that the time taken for the simulation is nearly 1,220 s, and the MSE is also 0.0002.

The networks from 11 to 17 consist of two hidden layers. The number of neurons in the hidden is varied, and the

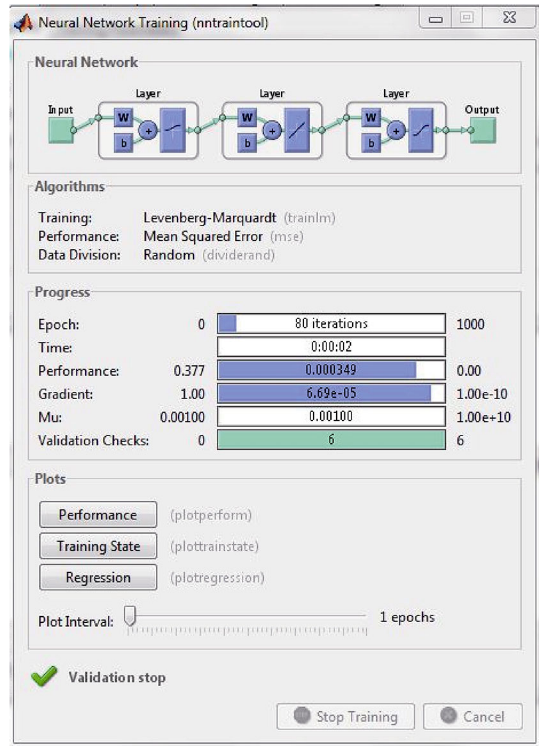

Fig. 6 NN training tool for Network 17

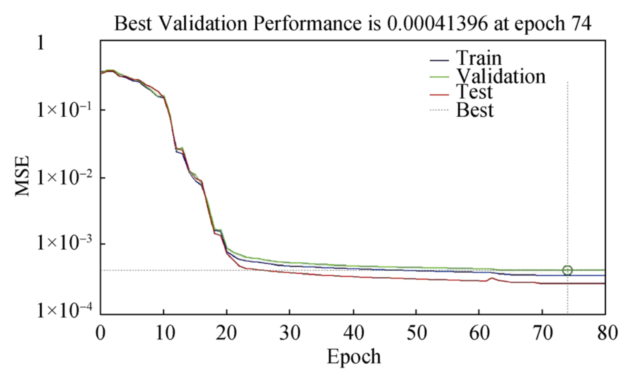

Fig. 7 Performance plot for Network 17

results are shown in Table 8. Networks 13, 14, and 16 are predicting the stress–strain curve with an average relative error nearly 3 %. From these networks, network number 16 is selected as the best network for the prediction of the stress–strain curve. Figures 12 and 13 show the comparison between the predicted and measured stress–strain curves of Ti alloy at 18 % and 24 % of α. These plots represent the comparison between the measured and ANN calculated curves with the use of Network 7.

Another attempt is that the training algorithm was changed in the Network 16 (i.e., double layer–layer recurrent neural network) in order to get the further improvement in predicting the output result. Table 8 shows

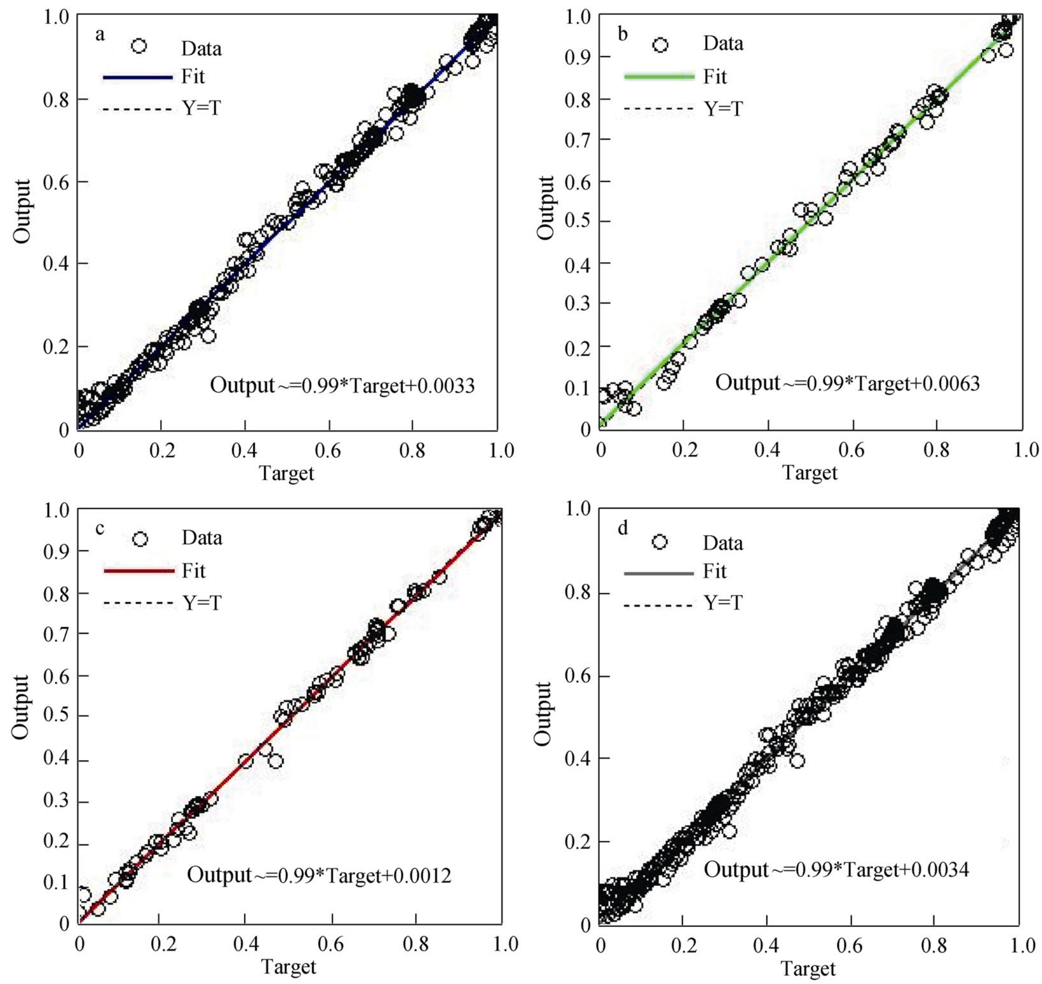

Fig. 8 Regression plot for Network 17: **a** training, $R = 0.99816$; **b** validation, $R = 0.99781$; **c** test, $R = 0.99848$; **d** all, $R = 0.99813$

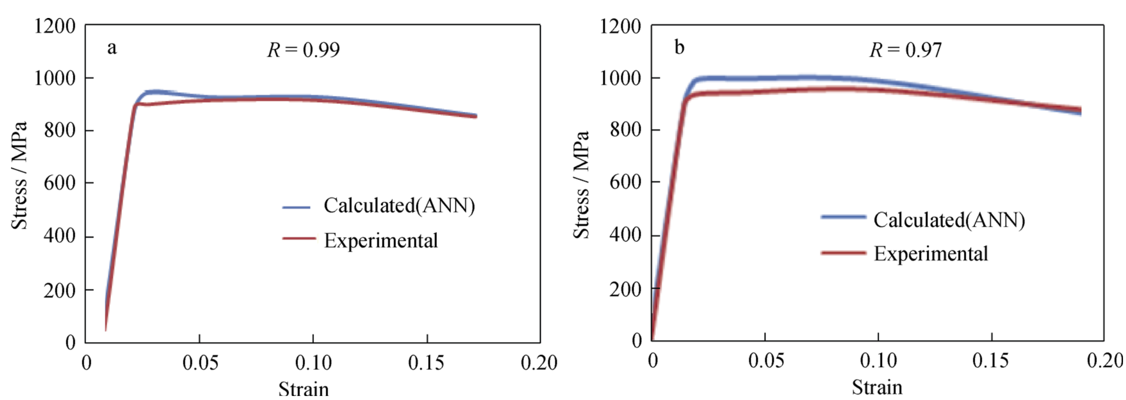

Fig. 9 Experimental versus predicted curve **a** 18 % and **b** 24 % volume fraction of α

the parameters used in the network models. Table 9 shows the ANN training results with different training algorithms.

Table 10 represents the list of other parameters used in neural networks. Figure 14 shows the comparison of average relative errors with its standard deviation values for different training algorithms. It is observed from Fig. 14 that along with trainlm another two training algorithms are giving good results. Those are traincgp and trainrp. The MSE value is very small for these three algorithms as compared with others. Figure 15 shows the

Table 6 Overall comparison

Network type		Tan-Sigmoid		Purelin		Log-Sigmoid	
		SHL	DHL	SHL	DHL	SHL	DHL
Cascade-forward back propagation	ARE/%	7.4	9.2	30.3	16.2	8.1	11.9
	SD	5.7	7.1	22.3	15.5	7.3	14.4
Feed-forward back propagation	ARE/%	11.8	13.4	30.0	12.5	13.5	6.3
	SD	8.7	11.3	22.1	6.9	10.4	4.3
Layer recurrent	ARE/%	9.1	6.1	30.5	12.4	6.7	5.1
	SD	6.7	4.3	22.2	6.4	5.4	4.0

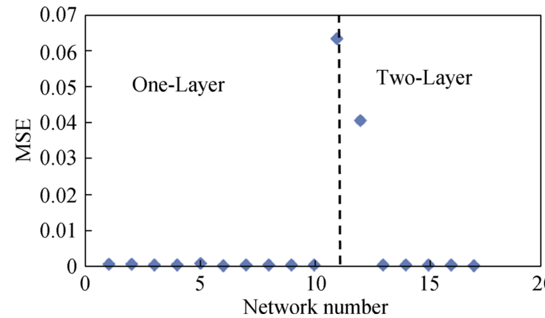

Fig. 10 Graph between mean square error versus network number

Table 7 Network parameters used for determination of optimum number of neurons

ANN parameters
Network type: layer recurrent network
Number of hidden layers: 1 and 2
Number of neurons: 5–25 (1st hidden layer) and 1–7 (2nd hidden layer)
Transfer function: Log-Sigmoid
Training algorithm: trainlm
Best performance: mean square error (MSE)

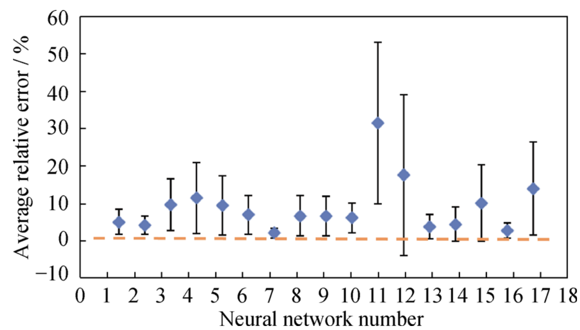

Fig. 11 Graph between average error versus network number

Table 8 Analysis of results

Nos.	Hidden neurons	Average relative error/%	SD	Epochs	Time/s	MSE
One hidden layer						
1	5	4.38	3.49	40	21	0.0006
2	6	3.52	2.59	41	22	0.0005
3	7	9.24	7.26	212	116	0.0003
4	8	11.01	9.89	197	112	0.0003
5	9	8.92	8.25	60	35	0.0008
6	10	6.70	5.40	60	33	0.0005
7	11	1.27	1.45	220	1221	0.0002
8	15	6.10	5.59	55	73	0.0004
9	20	6.10	5.52	74	151	0.0004
10	25	5.52	4.08	174	527	0.0003
Two hidden layers						
11	1…1	31.84	22.39	23	27	0.0632
12	1…5	17.37	22.41	53	67	0.0403
13	5…5	3.08	3.38	104	194	0.0004
14	7…5	3.71	4.76	246	346	0.0002
15	5…7	9.62	10.67	215	299	0.0003
16	7…7	2.06	2.10	53	71	0.0004
17	5…10	13.63	12.95	491	690	0.0001

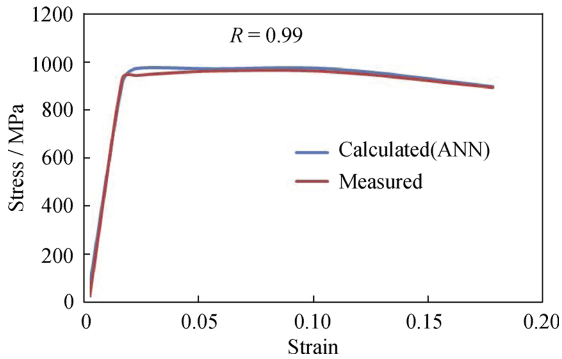

Fig. 12 Comparison between calculated and measured stress–strain curves of titanium alloy at 18 % volume fraction of α

Table 10 Network parameters used for determination of better training algorithm

ANN parameters
Network type: layer recurrent network
Number of hidden layers: 2
Number of neurons: 7 (1st hidden layer) and 7 (2nd hidden layer)
Transfer function: Log-Sigmoid
Training algorithm: BFGS, Conjugate, Traincfg, Traincgp, Traingd, Traingdm, Traingda, Traingdx, Trainoss, Trainr, Trainrp, Trainlm
Best performance: MSE

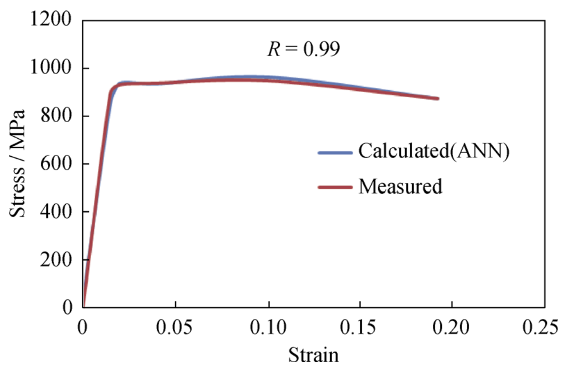

Fig. 13 Comparison between calculated and measured stress–strain curves of titanium alloy at 24 % volume fraction of α

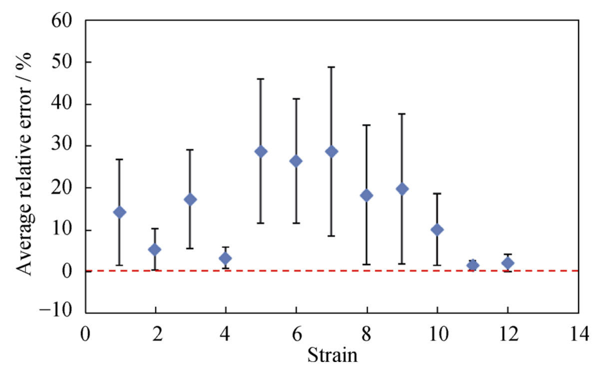

Fig. 14 Average error versus NN with different training algorithms

comparison of ANN predicted stress–strain curve with its corresponding experimental and FEM predicted stress–strain curves at two different volume fractions of α. Another interesting observation is that trainlm gives the acceptable results in less time as compared with other two training algorithms.

From these observations, it is observed that the best network is a layer recurrent neural network having two hidden layers consisting of 7 neurons in each, and the training algorithm is trainrp, and the transfer function is Log-Sigmoid. Figure 15 shows the comparison between the predicted and calculated stress–strain curves at two

Table 9 Training results on NN with different training algorithms

Nos.	Training algorithm	Average relative error/%	SD	Epochs	Time/s	MSE
1	BFGS QN	14.05	12.57	56	77	0.0151
2	Conjugate	5.21	4.84	108	180	0.0023
3	Traincfg	17.15	11.68	44	64	0.0413
4	Traincgp	3.22	2.50	182	273	0.0008
5	Traingd	28.52	17.07	1,000	685	0.0581
6	Traingdm	26.26	14.74	1,000	691	0.0812
7	Traingda	28.39	20.04	124	84	0.0603
8	Traingdx	18.16	16.60	156	102	0.0264
9	Trainoss	19.55	17.81	34	44	0.0269
10	Trainr	9.91	8.47	100	125	0.0980
11	Trainrp	1.47	1.12	391	283	0.0006
12	Trainlm	2.06	2.10	53	71	0.0004

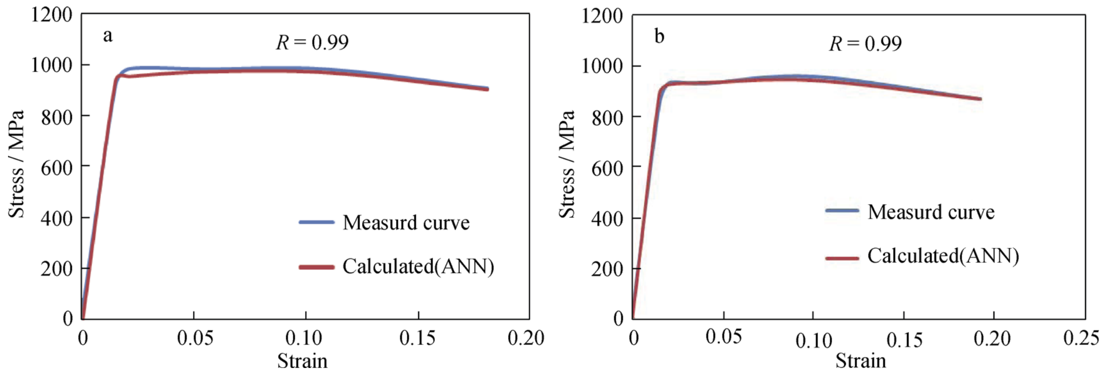

Fig. 15 Comparison between experimental and predicted curves at **a** 18 % and **b** 24 % volume fraction of α

different volume fractions of α (18 % and 24 %) using ANN Network 11 from Table 9.

5 Conclusion

The generalization ability is its main quality indicator of a neural network, and also another ability to predict accurately the output of unseen test data. Levenberg–Marquardt training algorithm with feed-forward back-propagation neural network is one of the fastest computational methods for predicting stress–strain curve with an acceptable accuracy (average relative error <(8 ± 2) %). The Log-Sigmoid transfer function is giving good results as compared with the Tan-Sigmoid and purelin. The layer recurrent neural network is accurately predicted with a trainlm as the training function consists of a SHL having 7 hidden neurons with a Log-Sigmoid transfer function (average relative error well within (1 ± 0.5) %). In two hidden layer neural network, layer recurrent neural network consists of 7 neurons in each hidden layer, and the training algorithm is trainrp with Log-Sigmoid transfer function (average relative error well within (2 ± 1) %).

Acknowledgments The experimentations are conducted in DMRL Laboratory Hyderabad. Dr. T.K. Nandy was helped a lot for the experimentation work. We also thank to all persons in TAG group.

References

[1] Li WP. The appliance of titanium alloy and its development. Light Met. 2002;5(53):25.
[2] Li L, Sun JK, Meng XJ. The appliance of titanium alloy and its development. Titanium Ind Prog. 2004;21(5):19.
[3] Schutz RW, Watkins HB. Recent developments in titanium alloy application in the energy industry. Mater Sci Eng A. 1998; 243(1–2):305.
[4] Yamada Makoto. An overview on the development of titanium alloys for non-aerospace application in Japan. Mater Sci Eng A. 1996;213(1–2):8.
[5] Rack HJ, Qazi JI. Titanium alloys for biomedical applications. Mate Sci Eng C. 2006;26(8):1269.
[6] Yao ZK, Guo HZ, Su ZW. The influences of thermodynamics parameter on (α+β) duplex titanium recrystallization percentage and static behavior. Rare Met Mater Eng. 2000;29(2):340.
[7] Boyer R, Welsch G, Collings EW. Materials Properties Handbook: Titanium Alloys. Materials Park: ASM International; 1994. 35.
[8] Wood RA, Favor RJ. Titanium Alloys Handbook. 3rd ed. Metals and Ceramics Information Center, vol. 24. Columbus: Batelle; 1972. 24.
[9] Murray JL. Phase Diagrams of Binary Titanium Alloys. Materials Park: ASM International; 1987. 24.
[10] Jindrich J, Sreeramamurthy A, Harold M. Calculations of stress–strain curve and stress and strain distributions for an α-βTi–8Mn alloy. Mater Sci Eng. 1978;34(3):203.
[11] Liao B, Zhang CL, Wu J, Cai DY, Zhao CM, Ren XJ, Yang QX. Numerical simulation of the stress–strain curve of duplex weathering steel. Mater Des. 2008;29(2):562.
[12] Dong HF, Jing L, Zhang Y, Joongkeun Park, Yang QX. Numerical simulation on the microstress and microstrain of low Si–Mn–Nb dual-phase steel. Int J Miner Metall Mater. 2010;17(2):173.
[13] Srinivasu G, Rao RN, Nandy TK, Gupta DK. Finite element modelling of α particle size on the stress strain curve of near beta Ti alloy. Mater Des. 2013;46(1):8.
[14] Sha W, Edwards KL. The use of artificial neural networks in materials science based research. Mater Des. 2007;28(6):1747.
[15] Ozerdem MS, Kolukisa S. Artificial neural network approach to predict mechanical properties of hot rolled, nonresulfurized, AISI 10xx series carbon steel bars. J Mater Process Technol. 2008;199(1–3):437.
[16] Malinov S, Sha W. Application of artificial neural networks for modelling correlations in titanium alloys. Mater Sci Eng A. 2004;365(1–2):202.
[17] Zhu YC, Zeng WD, Sun Y, Feng F, Zhou YG. Artificial neural network approach to predict the flow stress in the isothermal compression of as-cast TC21 titanium alloy. Comput Mater Sci. 2011;50(5):1785.
[18] Haykin S. Neural Networks: a Comprehensive Foundation. Upper Saddle River: Prentice-Hall; 1999. 157.
[19] Lippman RP. An introduction to computing with neural nets. IEEE ASSP Mag. 1987;4(2):4.
[20] Negnevitsky M. Artificial Intelligence: a Guide to Intelligent Systems, vol. 452. Harlow: Addison Wesley; 2002. 452.