OPTIMAL BRAIN SURGEON PRUNING OF NEURAL NETWORK MODELS OF MANUFACTURING PROCESSES

Bahaa Ibraheem Kazem **Ali Khudhair Mutlag**
Mechatronic Engineering
Baghdad University

ABSTRACT

In this paper, Optimal Brain Surgeon (OBS) pruning algorithm is proposed to optimize network architecture with respect to testing patterns error and overcoming the overfitting problem. Turning process is used as case study to improve the performance of the neural network-surface roughness model. Using the proposed algorithm reduced the prediction error on testing patterns from 0.6237 to 0.2854 based on the absolute percent error estimate. Also, a noticeable improvement is made in correlation coefficient from 0.8656 to 0.9807 making the network more reliable for new operating conditions.

الخلاصة

في هذا البحث طبقت خوارزمية جرّاحة الدماغ المثلي (OBS) لتشذيب النموذج المستحدث و لتَحسين معمارية الشبكةِ بالنسبة الىخطأ نماذج الاختبار ولتَحسين قدرةِ التعميمِ للشبكة بتَفادي ال overfitting لنموذجِ الشبكة العصبية. أستعملت عملية الخراطة في هذا العملِ كحالة للدراسة لتحسين موديل لشبكة عصبية لخشونة السطح حيث خفّضتْ خوارزميةُ تشذيب جرّاحةِ الدماغ المثلي خطأَ التوقع لنماذج الاختبار منْ 0.6237 إلى 0.2854 مع تحسن ملحوظ في معاملِ الإرتباطِ منْ 0.8656 إلى 0.9807 مما يجعلُ الشبكة أكثر موثوقية لحالات التشغيل الجديدة.

KEY WORDS
artificial neural networks, optimal brain surgeon, over fitting problem

INTRODUCTION

An Artificial Neural Networks (ANNs) is a common method for modeling and identification of manufacturing Processes. The use of ANNs consists of two phases training and reasoning. The performance of the neural network model during reasoning phase is largely dependent on the available number of training patterns and network size. As the cost of collecting large number of training patterns is high in most manufacturing processes and there are no hard and fast rules that control the network architecture, most neural network models of manufacturing processes suffer from overfitting problem.

The importance of applying neural networks in manufacturing processes has now been widely recognized as it results in increased productivity, improved product quality, and decreased production cost. Also, the complexity of most manufacturing processes is the obstacle to its successful modeling using the conventional methods such as stochastic techniques, partial differential equation, etc.

Therefore, ANNs were used as a operating tool because they can handle strong non-linearties, a large number of parameters, missing information, the ability to adapt themselves to changes in the production environment, and can also be used in case where no detailed information is available about the relationships among the various manufacturing parameters. Using ANNs for modeling manufacturing processes has been reported in many references. For example, [Bauer and Georges.2001] developed a neural network based monitoring system for detection of tool state (worn or sharp) of lathe machine. The system used the solid – borne sound signal in machining to identify the tool state. Sixty-five features (time, frequency and statistical features) were extracted from the signal to form the feature vectors that act as input to a neural network classifier.

The work of [Silva, 2002] and [Scheffer, 2001] integrate the sensor fusion with neural networks to develop monitoring system for tool condition in turning. Multi sensor signals (such as cutting forces, machine vibration and current of spindle drive motor) were used to identify features related to tool state. The feature vector was presented to two types of neural networks classifiers, Adaptive Resonance Theory (ART) and Self – Organizing Map (SOM). The two networks showed excellent results, the classification accuracy reached a bout 100%.

[T.Özel, 2002] and [Nadgir, 2000] have been developed neural network prediction model to predict flank wear of Cubic Boron Nitride (CBN) tools in hardened steel machining. The network was used to predict the size of flank wear correspond to the cutting speed, feed rate, depth of cut, feed force to cutting force ratio and cutting time.

[Kazem, 2004] had been developed neural network model for surface roughness prediction in turning operation of mild steel. The model showed good response to training data but unpredictable response observed when new cutting conditions are presented to the network.

Neural networks, fuzzy logic and empirical modeling methods for machining processes are compared by [Markos, 1998]. In that research, neural networks models are recommended over other models, sine neural networks able to learn from examples, map nonlinear relations and deal with noisy data efficiently.

Its important to note that many of these neural network models suffers from overfitting problem in which the model responds accurately to training patterns but gives large error when new patterns is presented to the model. Therefore, these models can not generalize its knowledge acquired during training phase due to small number of training patterns with respect to network's degrees of freedom (weights and biases).

Hence, this paper presents a novel post-training pruning method for arbitrary feedforward neural network models to optimize the model architecture and improving the performance of the model to new operating conditions (testing patterns) by discarding the non-essential weights (the weights that have the smallest saliency) via optimal brain surgeon pruning algorithm.

ARTIFICIAL NEURAL NETWORKS

Neural networks are computational models that share some of the properties of the brain. These networks consist of many simple "units" working in parallel with no central control, and learning takes place by modifying the weights between connections. The basic components of an ANN are "neurons", weights, and learning rules. In general, neural networks are utilized to establish a relationship between a set of inputs and a set of outputs. **Fig.(1)** shows the general architecture of a multilayer, (fully connected) feedforward neural network. ANNs are made up of three different types of "neurons": (1) input neurons, (2) hidden neurons, and (3) output neurons.

Inputs are provided to the input neurons, such as machine parameters, and outputs are provided to the output neurons. These outputs may be a measurement of the performance of the process, such as part measurements. The network is trained by establishing the weighted connections between the input neurons and output neurons via the hidden neurons. Weights are continuously modified until

the neural network is able to predict the outputs from the given set of inputs within an acceptable user-defined error level.
Multilayer (fully connected) networks used in modelling the relationship between surface roughness and cutting variables (cutting speed, feed rate, nose radius, and depth of cut) in turning process. The activation function used during training is the tangent sigmoid function in the hidden layer and linear activation function for the output layer.

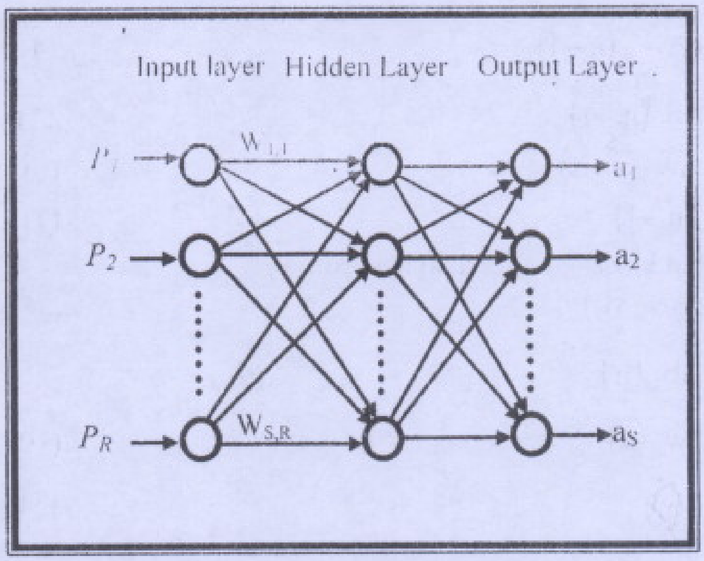

Fig.(1) Multilayer Feedforward Neural Network Architecture

The network is trained by error backpropagation algorithm. Initially, weights connecting input and hidden units and weights connecting hidden and output units are assigned a random value. The outputs of the hidden neurons are calculated by :

$$h_j = f\left(\sum_i p_i w_{ij} + b_j\right) \qquad \ldots(1)$$

Where

h_j: the actual output of hidden neuron j .
p_i: input signal of input neuron i .
w_{ij}: weight between input neuron i and hidden neuron j .
b_j: bias of hidden neuron j .
f: the tangent sigmoid activation function .

The output of the output layer is calculated by :

$$a_k = f\left(\sum_j h_j w_{jk} + b_k\right) \qquad \ldots(2)$$

Where

a_k: the actual output of output neuron k .
w_{jk}: weight between hidden neuron j and output neuron k .
b_k: bias of the output neuron k .

The error at the output layer backpropagated using the relation :

$$\delta_k = (t_k - a_k) f'\left(\sum_j h_j w_{jk} + b_k \right) \qquad \ldots(3)$$

Where

f ′: the derivative of the activation function.

t_k: the target of output neuron k.

The weights and biases are adjusted between the hidden layer and output layer (Equation 4,5) and between the input and hidden layer (Equation 6,7) where the learning rate is denoted by α and momentum factor by μ ;

$$\Delta w_{jk}(n) = \alpha\, \delta_k h_j + \mu\, \Delta w_{jk}(n-1) \qquad \ldots(4)$$

$$\Delta b_k(n) = \alpha\, \delta_k h_j + \mu\, \Delta b_k(n-1) \qquad \ldots(5)$$

$$\Delta w_{ij}(n) = \alpha\, \delta_j p_i + \mu\, \Delta w_{ij}(n-1) \qquad \ldots(6)$$

$$\Delta b_j(n) = \alpha\, \delta_j + \mu\, \Delta b_j(n-1) \qquad \ldots(7)$$

Finally, the weights and biases are updated according :

$$w_{jk}(n) = w_{jk}(n-1) + \Delta w_{jk}(n) \qquad \ldots(8)$$

$$b_k(n) = b_k(n-1) + \Delta b_k(n) \qquad \ldots(9)$$

$$w_{ij}(n) = w_{ij}(n-1) + \Delta w_{ij}(n) \qquad \ldots(10)$$

$$b_j(n) = b_j(n-1) + \Delta b_j(n) \qquad \ldots(11)$$

Once the network is sufficiently "trained", a general model is created for the relationship between inputs (cutting variables in turning process) and output (surface roughness). The user can then determine the impact that a specific set of process inputs has on a set of process outputs. For example, what affect does a change in machining parameters have on the quality of the machine part? [Demuth, 1998].

Overfitting Problem

One of the problems that occurs during neural network training is called overfitting. The error on the training set is driven to a very small value, but when new data is presented to the network the error is large. The network has memorized the training examples, but it has not learned to generalize to new situations.

When the number of Degrees Of Freedom (DOFs or parameters) of the network is too large with respect to training patterns, overfitting problem occurs. On the other hand , when the network DOFs are small, the network suffers from under fitting problem, in which the network can not learn the relationship at all.

Unfortunately, it is difficult to know beforehand how large a network should be for a specific application. Therefore, some means of network optimization most be implemented as done in this work with the OBS pruning algorithm.

The following (example) figure shows the response of a 1-20-1 neural network that has been trained to approximate a noisy sine function. The underlying sine function is shown by the dotted line, the noisy measurements are given by the '+' symbols, and the neural network response is given by the solid line. Clearly this network has overfit the data and will not generalize well. [Stich, 2000].

Optimal Brain Surgeon (OBS) Pruning Algorithm

The aim of OBS pruning algorithm is to capture the optimal network size by gradually reducing (a large trained) network's degrees of freedom. The algorithm is based on the idea of iteratively removing single degree of freedom (weight or bias) and then adjusting the remaining weights with a view to maintaining the original input-output behavior [Hasegawa, 1976] and [pricipe, 2000]. **Fig.(3)** shows the flow chart of the algorithm. One of the fundamental concerns in this pruning algorithm is how best :o select the weights or biases to be removed.

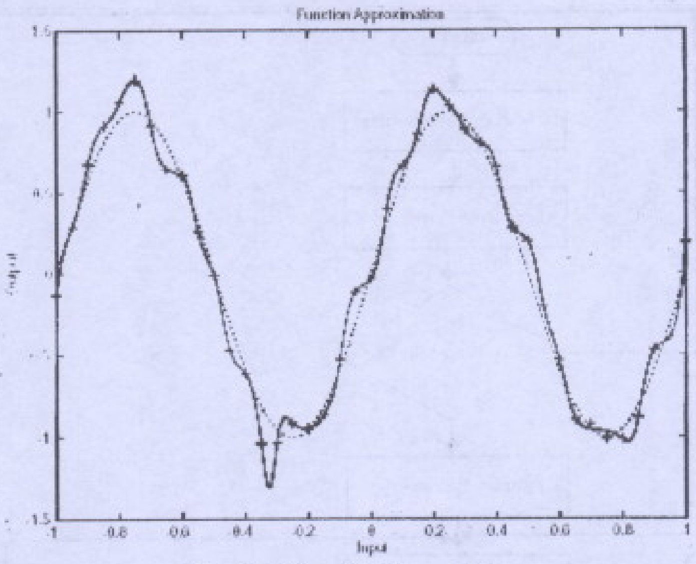

Fig.(2) Overfitting problem

One possible strategy is to make use of some relevance or sensitivity measure to quantify the contribution that individual weights or biases make in solving the network task, and then to select the less relevant weight or bias as this to be eliminated. This sensitivity measure is called "weight saliency" in which the effect of each weight on the network error is taken in account. Thus, the saliency of the weights and biases are computed and it has been shown that the Hessian matrix (**H**) with elements

$$H_{ij} = \frac{\partial^2 E}{\partial w_i \partial w_j} \qquad (12)$$

contains the required information. Where E is the network error which is defined as follow :-

$$E = \frac{1}{2Q}\sum_{i=1}^{Q}(a_i - t_i)^2 \qquad (13)$$

Where Q number of training patterns.

a_i network response to pattern i.

t_i network target to pattern i.

The problem with this computation it is nonlocal (requires information from pairs of weights). A local approximation (sometimes poor) to the Hessian can be computed by taking into consideration only the diagonal terms, which lead to the following calculation of the saliency (training error change) s_i for each weight w_i :

$$s_i = \frac{H_{ij} w_i^2}{2} \qquad (14)$$

Removing single weight or bias is not the whole story. The (new) network must be retrained up to 50 iteration using Levenberg-Marquardt training algorithm to adjust the remaining weights in order to compensate the effect of eliminated weight or bias.

Note that the iterative nature of the above procedure allows the network designer to define appropriate performance measure. So splitting the available patterns into training and testing patterns making the network performance can be measured with respect to testing patterns in an attempt to improve generalization. Moreover, after pruning the optimal (final) network need not be retrained because the updating of the weights is embedded in the pruning algorithm itself.

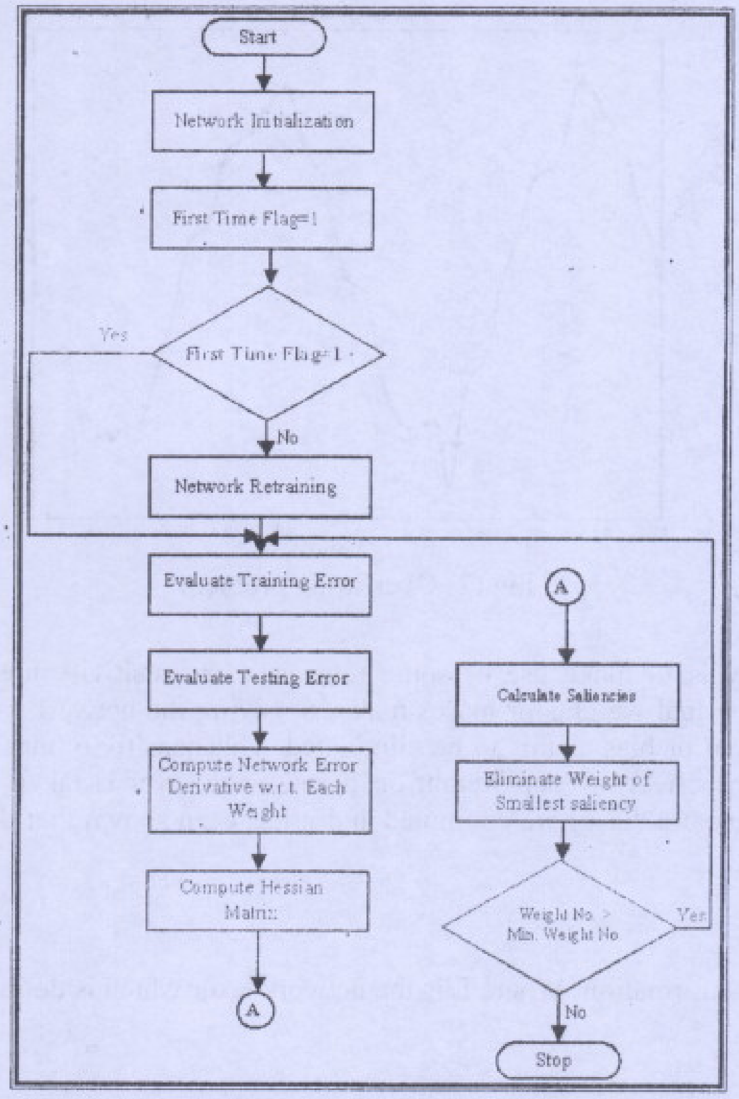

Fig. (3) Optimal Brain Surgeon Algorithm Flow Chart

Neural Network-Surface Roughness Model

One of the main objectives of this paper is to estimate a network model with high generalization ability. The neural network model for surface roughness prediction in turning process consist of four neurons in the input layer (one for each cutting parameters, cutting speed, feed rate, nose radius and depth of cut), fifteen neurons in the hidden layer, and single neuron in the output layer (corresponding to the predicted surface roughness). **Table(1)** shows the available patterns, [Hintz-Madsen, 1998] for training and testing. Ninety percent of these patterns is taken for training, while the remainder is used for testing purposes to provide a measure of the network generalization to new operating conditions. First of all, fully connected network must be trained and tested to new patterns (not contained in training patterns). The network training is done via Levenberg-Marquardt training algorithm with fifteen hidden neurons up to 61 epoch as shown in **Fig.(4)**.

Network Simulation

Testing of a neural network model requires new independent (unseen previously by the network) data set to validate the generalization capability of the network. **Fig.(5)** shows the network response to testing data set and its deviation from the target surface roughness.

The prediction accuracy for the testing patterns based on mean absolute percent error (APE) criteria:-

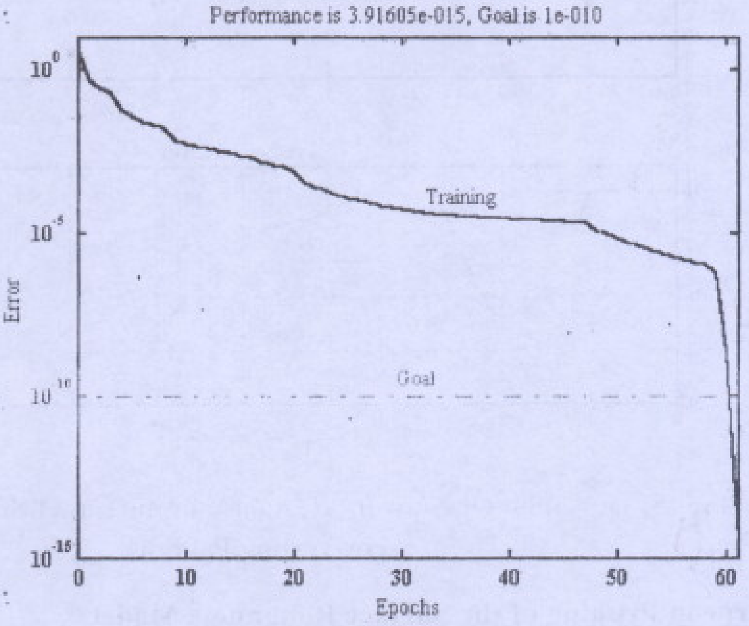

Fig.(4) Training Session

$$APE = \frac{|\text{Network Response} - \text{Target}|}{\text{Target}} \times 100\% \qquad \ldots(15)$$

is 37.6%, and the correlation coefficient 0.8656. As expected, the response of the network to testing data is poor. Since the network size is very large with respect to available training data, overtraining would be occur and the ability of the network to generalize the information acquired during training phase would be poor. Therefore, Optimal Brain Surgeon (OBS) pruning algorithm must be implemented to optimize the network architecture and to improve the generalization ability of the surface roughness model.

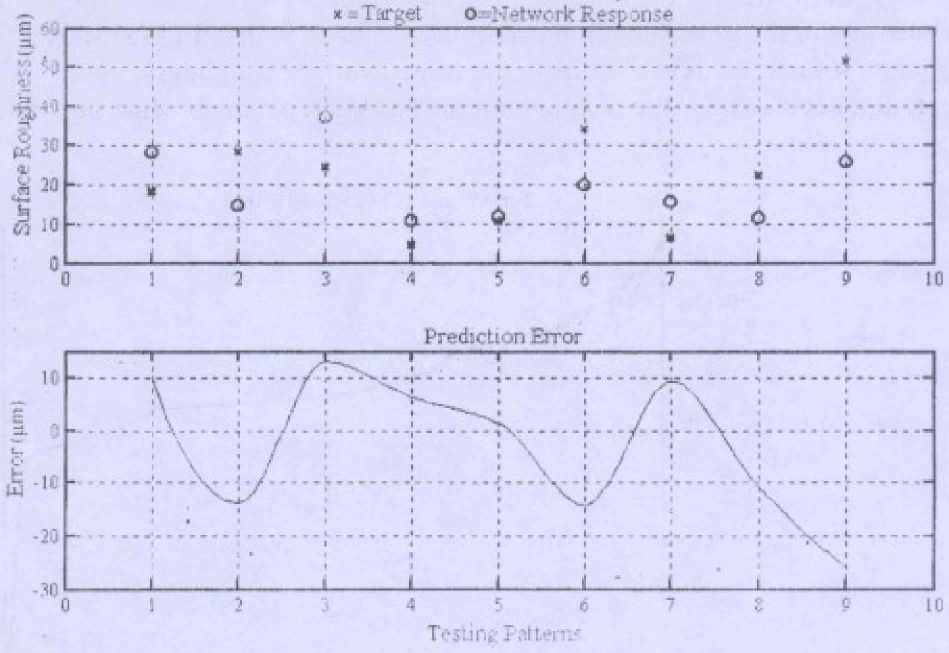

Fig.(5) Deviation Of Network Response From Target Surface Roughness For Testing Patterns.

Optimal Brain Surgeon Pruning of the Surface Roughness Model

The idea of optimal brain surgeon pruning of the surface roughness model is to find a suitable criteria to decrease the number of network parameters to optimize the network architecture and therefore, improving network performance to any new input patterns.

MATLAB based computer program was used to implement the OBS pruning algorithm. The optimization strategy is based on the minimum testing error. In other words, the optimal network architecture is the network which gives minimum testing error rather than training error. The pruning session is shown in **Fig.(6)** .

This figure is read from right to left. the errors to the right are associated with the initial network parameters (91 weights and biases). This figure gives very rich information. It can be noted that the training error is very small in the initial network architecture while the testing error is large. As pruning starts, single weight elimination is done in each times resulting in new network, this new network is retrained up to 50 iteration using Levenberg- Marqvardt training algorithm to adapt the network weights. after this, the network simulated with training and testing patterns to compute the corresponding errors. These errors are plotted versus network parameters. Then, another weight is eliminated and so on.

Fig.(7) shows that optimal network(partially connected network) architecture which gives minimum testing error the one which have 54 weights and biases .

Table(2) shows the weights and biases of the pruned network. It is important to note that pruning session must be run several times each with different initial weights and biases to avoid local minima problem in capturing optimal network architecture. The network resulted from pruning session is called "partially connected network".

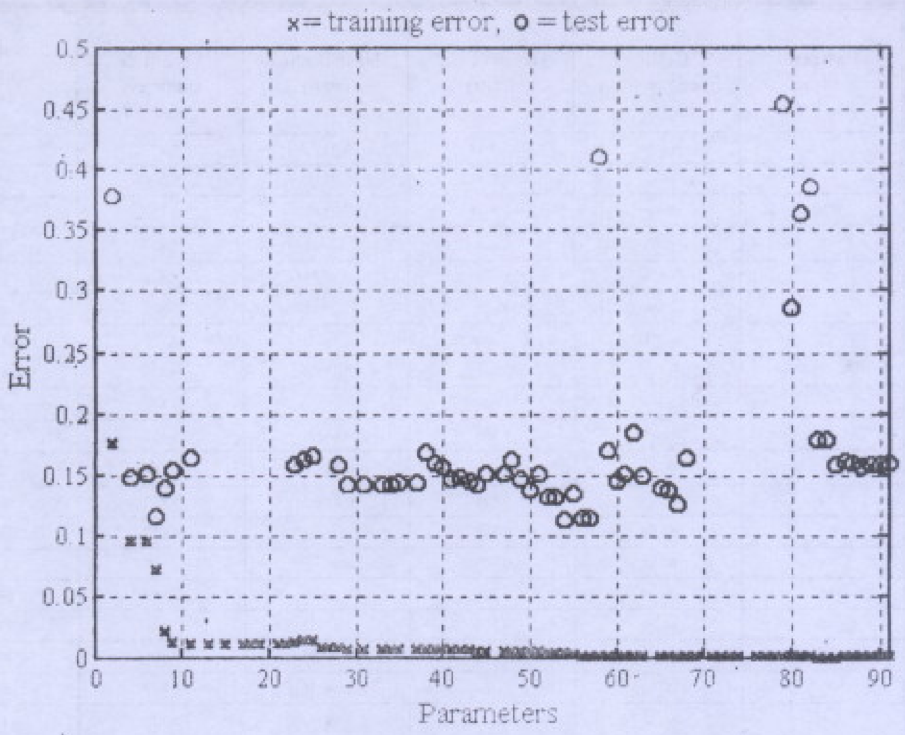

Fig.(6) Pruning Session

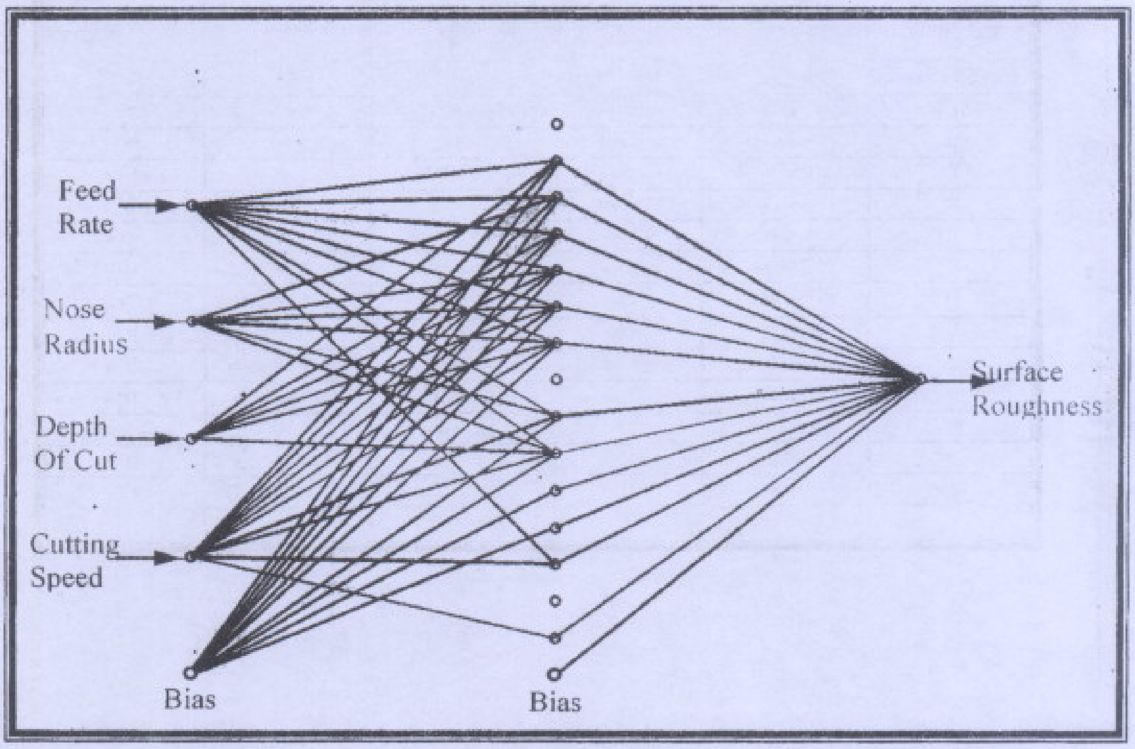

Fig.(7) Optimal Network Architecture

Table (1) Training Data Set

Pattern Number	Cutting Speed (m/min)	Depth of Cut (mm)	Nose Radius (mm)	Feed Rate (mm/rev)	Surface Roughness (µm)
1	60.96	0.3048	0.74375	0.127	28.96
2	60.96	0.3048	0.74375	0.508	39.16
3	60.96	0.3048	0.74375	0.884	94.56
4	60.96	0.3048	1.5875	0.127	17.53
5	60.96	0.3048	1.5875	0.508	39.42
6	60.96	0.3048	1.5875	0.889	40.07
7	60.96	0.3048	2.38125	0.127	13.7
8	60.96	0.3048	2.38125	0.508	41.5
9	60.96	0.3048	2.38125	0.889	44.38
10	60.96	0.508	0.74375	0.127	55.6
11	60.96	0.508	0.74375	0.508	52.16
12	60.96	0.508	0.74375	0.889	110.04
13	60.96	0.508	1.5875	0.127	31.68
14	60.96	0.508	1.5875	0.508	25.4
15	60.96	0.508	1.5875	0.889	67.77
16	60.96	0.508	2.38125	0.127	47.21
17	60.96	0.508	2.38125	0.508	30.72
18	60.96	0.508	2.38125	0.889	57.2
19	60.96	0.7112	0.79375	0.508	43.2
20	60.96	0.7112	0.79375	0.889	100.87
21	60.96	0.7112	1.5875	0.127	33.21
22	60.96	0.7112	1.5875	0.889	76.15
23	60.96	0.7112	2.38125	0.127	17.84
24	60.96	0.7112	2.38125	0.508	22.57
25	182.88	0.3048	0.79375	0.508	52.76
26	182.88	0.3048	0.79375	0.889	104.54
27	182.88	0.3048	1.5875	0.127	10.5
28	182.88	0.3048	1.5875	0.889	64.44
29	182.88	0.3048	2.38125	0.127	21.66
30	182.88	0.3048	2.38125	0.508	16.48
31	182.88	0.508	0.79375	0.127	14.6
32	182.88	0.508	0.79375	0.508	43.44
33	182.88	0.508	0.79375	0.889	110.24
34	182.88	0.508	1.5875	0.127	9.96
35	182.88	0.508	1,5875	0.508	25.08
36	182.88	0.508	1.5875	0.889	71.08
37	182.88	0.508	2.38125	0.127	17.48
38	182.88	0.508	2.38125	0.508	19.16
39	182.88	0.508	2.38125	0.889	49.12
40	182.88	0.7112	0.79375	0.127	9

Table (1)Training Data Set (cont.)

[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
41	182.88	0.7112	0.79375	0.508	35.85
42	182.88	0.7112	0.79375	0.889	99.72
43	182.88	0.7112	1.5875	0.127	8.58
44	182.88	0.7112	1.5875	0.508	17.7
45	182.88	0.7112	1.5875	0.889	61.07
46	182.88	0.7112	2.38125	0.127	10.58
47	182.88	0.7112	2.38125	0.508	17.07
48	182.88	0.7112	2.38125	0.889	44.53
49	304.8	0.3048	0.79375	0.127	6.2
50	304.8	0.3048	0.79375	0.508	32.5
51	304.8	0.3048	0.79375	0.884	41.12
52	304.8	0.3048	1.5875	0.127	3.9
53	304.8	0.3048	1.5875	0.508	9.61
54	304.8	0.3048	1.5875	0.884	54.48
55	304.8	0.3048	2.38125	0.127	5.95
56	304.8	0.3048	2.38125	0.508	10.46
57	304.8	0.3048	2.38125	0.884	34.84
58	304.8	0.508	0.79375	0.508	42.8
59	304.8	0.508	0.79375	0.884	113
60	304.8	0.508	1.5875	0.127	5.01
61	304.8	0.508	1.5875	0.884	77.4
62	304.8	0.508	2.38125	0.127	7.2
63	304.8	0.508	2.38125	0.508	13.92
64	304.8	0.7112	0.79375	0.127	5.53
65	304.8	0.7112	0.79375	0.127	37.63
66	304.8	0.7112	0.79375	0.884	101.77
67	304.8	0.7112	1.5875	0.127	4.96
68	304.8	0.7112	1.5875	0.508	17.84
69	304.8	0.7112	1.5875	0.884	62.16
70	304.8	0.7112	2.38125	0.127	5.81
71	304.8	0.7112	2.38125	0.508	11.2
72	304.8	0.7112	2.38125	0.884	38.88
73	60.96	0.7112	0.79375	0.127	18.1
74	60.96	0.7112	1.5875	0.508	28.4
75	60.96	0.7112	2.38125	0.889	24.43
76	182.88	0.3048	0.79375	0.127	4.62
77	182.88	0.3048	1.5875	0.508	22.94
78	182.88	0.3048	2.38125	0.889	34.28
79	304.8	0.508	0.79375	0.127	6.32
80	304.8	0.508	1.5875	0.508	22.32
81	304.8	0.508	2.38125	0.889	51.6

Table(2) Pruned Network Weights and Biases

Hidden Neurons	Input Layer Neurons				Hidden Layer Biases	Output Layer Weights
	cutting speed	depth of cut	nose radius	feed rate		
1	-13.2767	0	0	0	0	0.1065
2	0	0	0	0	0	0
3	8.9439	0	0	-14.2025	0	0.1625
4	0	0	0	0	1.7817	24.1591
5	0	0	0	0	-0.5379	-32.9192
6	-4.0631	8.4596	0.1556	-4.2342	-2.0494	-13.3715
7	-0.0441	0	0.412	-3.7874	2.2629	7.8577
8	0	0	0	0	0	0
9	0.0965	0.1487	-0.5796	1.5656	-1.4459	2.1772
10	-5.2061	15.9555	24.2066	-9.4347	-17.0581	0.0777
11	10.2479	-5.1649	7.4856	-25.9397	-13.1944	-0.1624
12	3.4330	-7.5909	0	3.686	1.9739	-13.8256
13	-3.7676	0	0.0785	18.5856	-5.5417	6.8187
14	-0.3303	7.7624	0	-0.8402	-1.0729	-0.5909
15	0	0	0	0	0	0
Bias of the Output Network = -38.7255						

Discussion of the Results

In this work, neural networks are used to model the relationship between surface roughness and machining variables (cutting speed, depth of cut, nose radius and feed rate) in turning process. The developed neural network model (fully connected network) suffers from overfitting problem, in which the network can not generalize its knowledge acquired during training phase to new operating conditions. Thus, the network showed poor response to new patterns (which are not seen during training) as illustrated in **Fig.(5)**.Therefore, OBS pruning algorithm is used to optimize network architecture and to improve the generalization capability of the network resulting in a partially connected network with 54 weights and biases as shown in **Fig.(7)**.

Fig.(6) shows pruning session, where the network performance in terms of training and testing error are plotted versus the number of network parameters (weights and biases). From this figure, optimal network architecture that gives minimum testing error is chosen. This figure is read from right to left and gives very important information. The error to testing patterns is very large when the size of the network is large although the training error is small. Therefore, a trade-off between training and testing error must be made in order to catch the optimal network architecture.

The developed neural network model was used to predict surface roughness for the available machining variables which show a reasonable agreements between the target and the predicted values as illustrated in **Figs.(8)**.

The pruning algorithm reduced prediction error for testing patterns from 0.6237 (for fully connected network) to 0.2854 (for partially connected network), with a noticeable improvement in correlation coefficient from 0.8656 to 0.9807 making the network more reliable for new operating conditions.

CONCLUSIONS

The OBS pruning algorithm has strong influence on the ability of the network to generalize its knowledge acquired during training phase to any new unseen operating conditions. Thus, the poor response of the fully connected network architecture to testing patterns is overcomed by pruning, making the neural network model applicable not to training data only, but to a wide and different range of machining variables for surface roughness prediction.

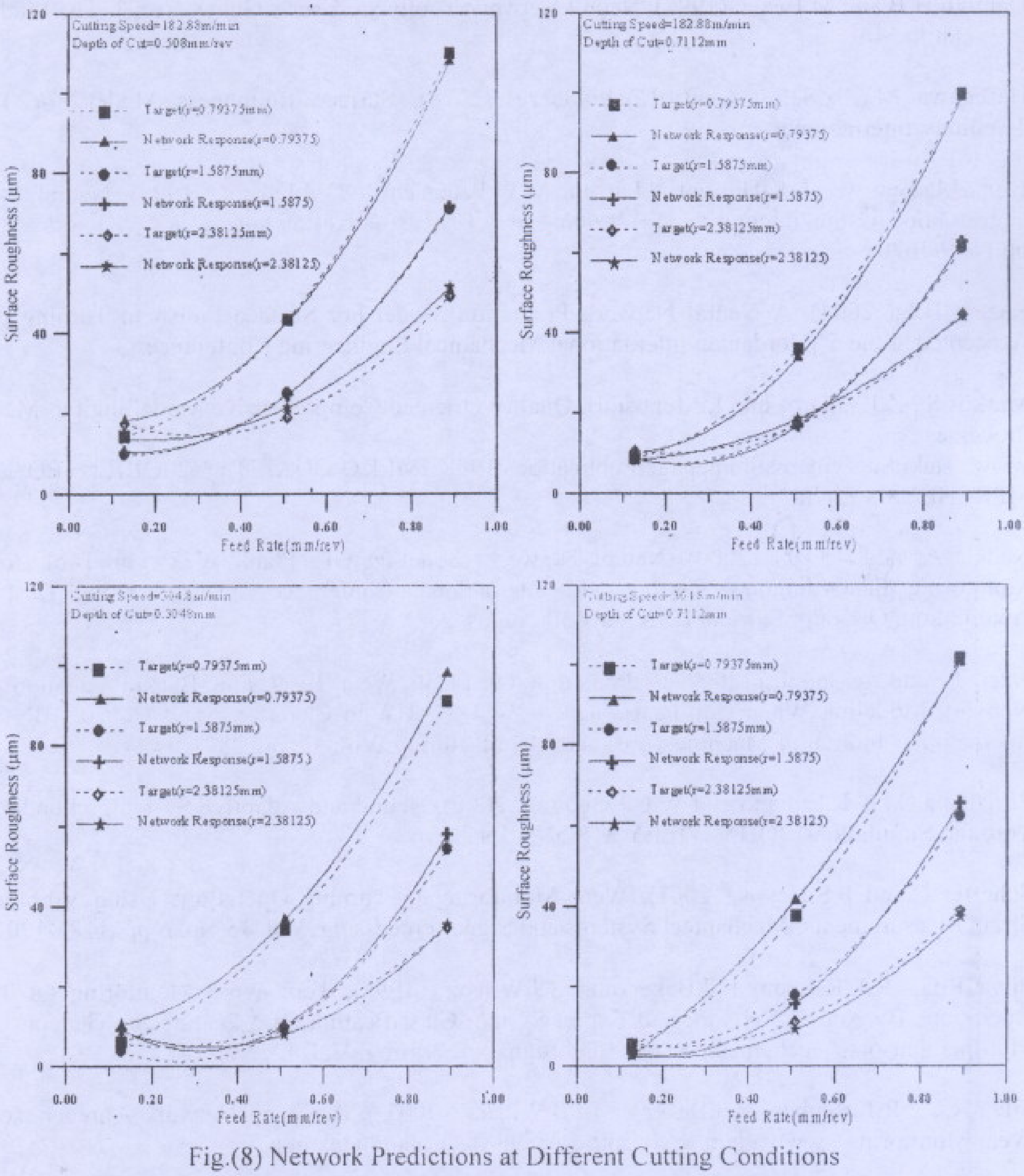

Fig.(8) Network Predictions at Different Cutting Conditions

REFERENCES

Bauer, B., D.Georges,: B.Geropp and M.Poschmann, (2002), Lathe Tool Wear Monitoring With Neural Networks, www.formikrosys.de/veroeffentlichungen.html

Demuth H.B.and M.Beala, (1998), Neural Network Toolboxa: User's Guide, Ver, 3, The Mathworks, Inc., Natik, MA.

Hasegawa M., A.Seireg, and R.A.Lindberg, (1976), Surface Roughness Model For Turning. Tripology International.

Hintz-Madsen M., L.K.Hansen, J.Larsen, M.W.Pedersen and M.Larsen,(1998), Neural Classifier Construction Using Regularization, Pruning and Test Error Estimation, Neural Networks, Vol. 11, pp.1659-1670.

Kazem B.I.,(2004), A Neural Network Prediction Model For Surface Finish In Turning process, Acccepted at the 5th Jordanian International Mechanical Engineering Conference.

Markos S., Z.J.Viharos and L.Monostori, Quality-Oriented Comprehensive Modelling Of Machining Processes . www.sztaki.hu/~viharos/homepage/Publications/1998_IMEKO/QUALITY%20ORIENTED%20COMPREHENSIVE.pdf.

Nadgir A, and T,Özel (2000), Neural Network Modeling Of Flank Wear For Tool Condition Monitoring In Orthogonal Cutting, 4th International Conference on Engineering Design and Automation, Orlando, Florida, USA, July 30-August 2.

Özel T. and A.Nadgir, (2002), Prediction Of Flank Wear By Using Backpropagation Neural Network Modeling When Cutting Hardened H-13 Steel With Chamfered and Honed CBN Tools . International Journal of Machine Tools and Manufacturing, Vol. 42, pp. 287-297.

Principe J.C., N.R.Euliano and W.C.Lefebvre,(2000), Neural and Adaptive Systems : Fundamentals Through Simulations, JOHN WILEY & SONS, INC..

Scheffer C.and P.S.Heyns,(2001), Wear Monitoring In Turning Operations Using Vibration and Strain Measurements, Mechanical Systems and Signal Processing, Vol. 15,No. 6,pp. 1185-1202.

Silva R.G., R.L.Reuben, K.J.Baker and S.J.Wilcox,(1998), Tool Wear Monitoring Of Turning Operations By Neural Network and Expert System Classification Of a Feature Set Generated From Multiple Sensors, Mechanical Systems and Signal Processing, Vol. 12, No. 2.

Silva R.G., R.L.Reuben, K.J.Baker and S.J.Wilcox,(2003), A Neural Network Apprach To Tool Wear Monitoring , www.glam.ac.uk/sot/research/MechMan/index.php.

Stich T.J., J.K.Spoerre and T.Velasco,(2000), The Application of Artificial Neural Networks To Monitoring and Control Of An Induction Hardening Process, Journal of Industrial Technology, Vol. 16, No. 1 .