ISSN: 2278-0181
Vol. 1 Issue 5, July - 2012

# Convergence of Recurrent Neural Networks Using Partially Trained ANN

A. V. Nigalye*
*National Institute of Technology, Suratkal, Karnataka, India, Phone: 09890011961*

U. J. Amonkar
*Goa College of Engineering, Goa, India*

S. S. Rao
*National Institute of Technology, Suratkal, Karnataka, India*

M. A. Herbert
*National Institute of Technology, Suratkal, Karnataka, India*

## Abstract

*Artificial Neural Networks are known for non linear mappings of complex systems. However these are static mapping tools in the sense that the knowledge update is based on static data provided for training the network. Simple recurrent neural networks (SRN) such as the one proposed by Elman have the capacity of dynamic learning, but are found to possess severely hampered learning capabilities due to convergence problems. A novel way of overcoming the problem of convergence is proposed through this paper by using a Hybrid Recurrent Neural Network modelled from an Artificial Neural Network (ANN) possessing similar architecture.*

**Keywords:** Elman networks (ENs), extended Elman networks, backpropagation, RNN convergence, hybrid RNN (HRNN).

## 1. Introduction:

Artificial Neural Networks (ANN) is a field of machine learning which in a way represents, to a large extent, the human style of learning. The study of ANN is inspired by the working principles of the human brain and by the way in which a human brain is able to process a large data by way of parallel processing and is able to retrieve the data at will [1]. Fundamentally, an ANN network does not need any knowledge of the process that it is trying to model, as it learns on the basis of the examples or the experimental data being supplied to it during training [2, 3]. It is exactly because of this advantage that a tool such as ANN is preferred over other prediction tools, such as statistical or numerical methods [3].

Artificial Neural Networks are thus mathematical models representing the gathering and processing data in a way similar to the human brain [4-7]. The neural network's ability to carry out the computations relating the inputs and outputs is inspired by the massive parallel and distributed processing of biological neurons. Neural Networks (NN) are able to perform complex mappings of input and output elements. When properly trained, ANN nicely generalizes the input output relationship by understanding the underlying relationship functions between input and output parameters, even from a limited data set. There are a variety of ANN architectures available, but the literature [8, 9] suggests that the ones which are being widely used are the supervised learning Feed Forward Neural Networks (FFNN). In FFNN the output of each neuron in a layer is passed on to each neuron in the subsequent layer, through connections called as synaptic weights. The knowledge of mapping is stored in the network in the form of these weights. The network is fed with the input vectors and the target output vectors one by one. The network calculates the output from the input data and in case of supervised learning this is compared with the target output. The error in the output is used to update the weights.

There are a lot of learning algorithms proposed in the studies on Neural Networks, but the one which is used most widely is the back propagation algorithm [1, 4, 8, 9, 10]. When the error is propagated backwards in the network, after each set of inputs is presented to the network, this algorithm uses the gradient descent approach in minimizing the Mean Squared Error (MSE) on the MSE weight planes and adjusts the connection weights accordingly, thus making the network learn. The entire set of inputs is presented (epoch) to the network again and again till the MSE reduces to some predetermined value. The training time and the number of epochs required to train the network depend upon, the number of hidden layers in the MLP, the number of neurons in each layer, learning rate parameter and momentum factor. There is no authentic information as to determine the number of hidden layers required for formulating a FFNN. One hidden layer is good enough to map most of the input output relationships, but more complex mappings are better achieved with two hidden layers [11]. Similarly there is no formula or relationship to fix the number of neurons in each hidden layer and this is done by trial and error [4, 11].

In general a FFNN would look like the network shown in Fig: 1. The inputs from $i^{th}$ layer ($x_i$) are fed to the neurons in the first hidden ($j^{th}$) layer. Each neuron in this layer receives the inputs from each neuron in the input layer through a weighted connection ($w_{ji}$). In the neuron the weighted sum of inputs $\Sigma w_{ji}x_i$ is calculated. The activation function to be used has to be continuous so that back propagation algorithm can be used. The reason of using a activation function is to limit the output of the neurons within a pre set range. The activation function most often used is the sigmoid function which is continuous, monotonic non decreasing and nonlinear which is as follow

$$y = \frac{1}{1+e^{-x}} \qquad (1)$$

In the recent past quite a few Recurrent Neural Network (RNN) architectures have been studied [12-16]. Recurrent networks are neural networks with one or more feedback loops, in which the loops may be local or global. RNN can be divided into two broad categories depending on whether the states of the network are guaranteed and observable or not. Observable state is one in which the state of the network can be derived by observing only the inputs and outputs [12]. A model which falls into this class was proposed by Narendra and Parthasarathy [17] and had time delayed outputs as well as inputs fed to a Multi Layer Perceptron (MLP) which computed the output using the recent state dynamics. However, network having hidden dynamic states are not observable [12]. Single layered and multi layered recurrent networks are being extensively studied in recent times. A typical single layered RNN was the one proposed by Elman in 1990 [14]. In this network, the hidden layer is copied in a virtual or context layer and the feedback is given back to the same layer along with the next set of inputs in the next time step as seen in Fig: 2. The Elman network can be extended for a multilayered network with the temporal context layer providing feedback at each subsequent time step. Such a network is shown in Fig: 3. The convergence of a Simple Elman Recurrent Neural Network (SRN) has been established. The computational power of Elman networks is as good as that of finite state machines (FSM) [18]. In addition any network having additional layers between input and output layer than Elman network, possesses the same FSM power. The convergence of RNN has been active subject of research in machine learning. An extended back propagation algorithm for Elman networks reported a better convergence, faster training and better generalization [19]. In this algorithm, use is made of adaptive learning scheme coupled with adaptive dead zone to improve convergence speed.

In this paper we try to develop a novel way of improving the convergence of Elman (SRN) using the borrowed weights of a partially trained FFNN into an Elman network with single hidden layer or an extended Elman network having more than one hidden and context layer. The paper further highlights the fact that the recurrent neural network so formulated performs the task of predictions of outputs from a given set of inputs comparable to that performed by the fully trained FFNN, from which the weights were borrowed to formulate the RNN, together with better convergence.

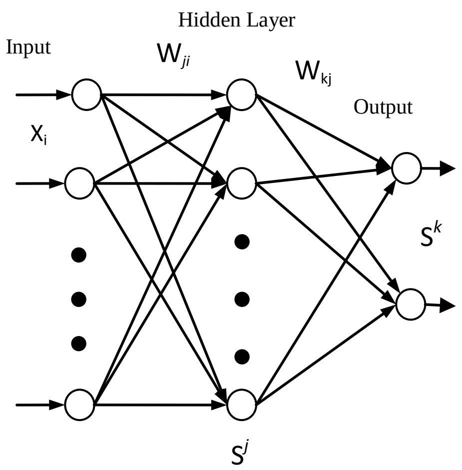


**Fig1. Schematic diagram of FFNN**

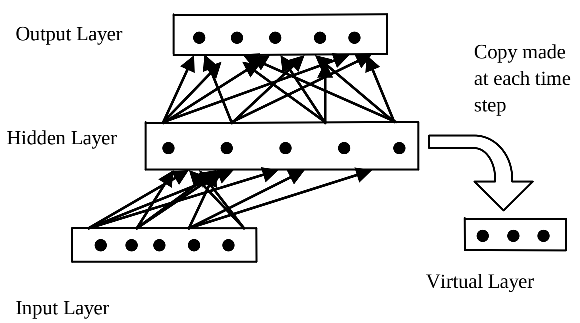


**Fig2. A simple RNN**

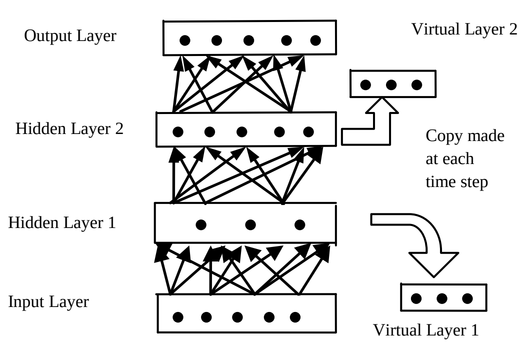


**Fig3. An extended simple RNN**

## 2. Data Generation:

The data for the ANN and proposed RNN has been taken from the work of M. A. Herbert [20]. The work dealt with study of microstructural details and mechanical properties of Al-4.5Cu-5$TiB_2$ composite when rolled from mushy state in as cast and in pre hot rolled condition. The data is presented in Table: 1.

Two ANNs have been trained. The first ANN predicts small and large grain sizes (μm) from four inputs namely, Material type (as cast or pre hot rolled), % thickness reduction during rolling, % liquid volume fraction which determines the mushy state condition during rolling initiation and % $TiB_2$ (wt %) in the composite. ANNs were formulated using different values of learning rate parameter (η) and momentum factor (α). The FFNNs with architectures 4-7-4-2 and 4-9-9-2 with η = 0.85 and α = 0.65 were found to converge to a MSE of 0.000362612 after 15 lakh epochs and 0.00017 after 277356 epochs respectively.

The second ANN predicts hardness of the composite ($H_v$) as output from five parameters provided as inputs. The input parameters were Material type, % thickness reduction, % liquid volume fraction, large grain size and small grain size. The ANN with 5-5-3-1 architecture and 5-9-6-1 architectures were trained with η = 0.85 and α = 0.65 and converged nicely to a MSE of $9.75*10^{-5}$ after 325000 epochs and a MSE of $9.8774*10^{-5-}$ after 345000 epochs.

## 3. RNN Modeling:

Elman Simple Recurrent Network (SRN) was modeled for Grain sizes as well as hardness predictions. The SRN with two context layers were tried with two hidden layer and with different combinations of number of neurons in each layer for different combinations of η and α. The networks failed to converge for a variety of combinations mentioned above. The SRN with Elman architecture uses a context layer that contains the same number of neurons as that in the hidden layer. The output of each hidden neuron which is being copied in the context layer will contain neurons with exciting as well as inhibiting signals. These neurons then pass on the signals through weighted connections to each neuron in the current hidden layer in the next time step along with the signals these neurons in the current hidden layer receive from the neurons in the previous layer. Due to this, the previously excited neuron or the neuron which otherwise would have received a consistent excited signal from previous layer neurons may get inhibiting signals from the context layer neurons or vice versa. This probably, does not allow the network to progressively move along the path of negative gradient on the MSE weight plane. Such a phenomenon is likely to cause a oscillatory profile on the MSE synaptic weight plane, which is exactly witnessed while training the simple Elman recurrent network.

**Table1. Experimental data of Al-4.5Cu Alloy and Al-4.5Cu-5TiB$_2$ composite rolled from mushy state in as cast and pre hot rolled condition [21].**

| Specimen Descriptions | Liquid Volume Fraction | As cast Al-4.5Cu-5TiB$_2$ Composite samples subjected to mushy state rolling | | Pre hot rolled Composite samples subjected to mushy state rolling | | Grain sizes of Al-4.5Cu alloy samples subjected to mushy state rolling | |
|---|---|---|---|---|---|---|---|
| | | Grain size ( µm ) | | Grain size ( µm) | | Grain size ( µm) | |
| | | Large | Small | Large | Small | Large | Small |
| As cast | | 50 ± 8 | | 52 ± 15 | 28 ± 9 | 44 ± 6 | |
| 2.5% thickness reduction | $f_1 \sim 0.1$ | 62 ± 14 | 27 ± 12 | 43 ± 16 | 27 ± 13 | | |
| | $f_1 \sim 0.2$ | 58 ± 18 | 33 ± 11 | 42 ± 18 | 26 ± 11 | | |
| | $f_1 \sim 0.3$ | 66 ± 15 | 37 ± 10 | 47 ± 20 | 25 ± 11 | 329 ± 204 | 158 ± 91 |
| 5% thickness reduction | $f_1 \sim 0.1$ | 54 ± 16 | 25 ± 9 | 42 ± 16 | 26 ± 11 | | |
| | $f_1 \sim 0.2$ | 51 ± 11 | 31 ± 10 | 41 ± 15 | 25 ± 12 | | |
| | $f_1 \sim 0.3$ | 55 ± 14 | 32 ± 10 | 46 ± 17 | 24 ± 11 | 363 ± 225 | 157 ± 86 |
| 7.5% thickness reduction | $f_1 \sim 0.1$ | 62 ± 20 | 32 ± 13 | 40 ± 15 | 26 ± 10 | | |
| | $f_1 \sim 0.2$ | 48 ± 19 | 26 ± 12 | 39 ± 15 | 25 ± 11 | | |
| | $f_1 \sim 0.3$ | 53 ± 18 | 27 ± 13 | 45 ± 17 | 24 ± 09 | 383 ± 222 | 158 ± 98 |
| 10% thickness reduction | $f_1 \sim 0.1$ | 49 ± 17 | 29 ± 11 | 47 ± 18 | 32 ± 13 | 351 ± 218 | 194 ± 96 |
| | $f_1 \sim 0.2$ | 47 ± 14 | 30 ± 12 | 43 ± 16 | 25 ± 11 | 325 ± 217 | 176 ± 103 |
| | $f_1 \sim 0.3$ | 54 ± 12 | 26 ± 11 | 45 ± 16 | 27 ± 12 | 357± 217 | 155 ± 87 |

In order to overcome this shortcoming, various strategies discussed hereunder were tried;

a. The networks with single hidden layer were trained for the same architectures mentioned earlier with different combinations of $\eta$ and $\alpha$. During training, it was observed that the network does not converge. Initially during training it learns nicely. But as the training progresses, the network starts oscillating randomly. Further it is observed that the oscillations decrease and the network stops learning and MSE reaches a value much higher than the pre set value which in our case is selected as 0.0001, thus indicating that the network has not been able to map the input output pattern.

b. The network model was slightly changed now, with the neurons in the hidden layer giving feedback only to itself. The networks were modeled for each of the case with similar architectures discussed at (a) above with different combinations of η and α. It was observed that the networks still oscillate during training and fail to converge to the pre-set value, but a better convergence is seen as indicated by slightly lower MSE indicating that learning capability of the network has slightly improved. But the convergence obtained is nowhere near the pre-set value of MSE. Hence, this strategy, though could not be discarded totally, was found to be ineffective.

c. In the modified model stated at (b) above, the weight vectors of a partially trained ANN with similar architecture were borrowed. The ANN network is partially trained till a steep negative gradient is identified on the MSE weights plane indicating the downward movement of MSE. The SNN is then modeled with similar architecture as that of partially trained ANN. The weight vectors of the SNN from input layer to hidden layer 1, hidden layer1 to hidden layer 2 and from hidden layer 2 to output layer are replaced by the corresponding weight vectors of partially trained ANN. The biases for different layers except the context layers of the SNN are also replaced by the corresponding biases from the partially trained ANN. The weight vectors from the context layer neurons to hidden layer neurons (each neuron to itself) is taken as a zero vector (unbiased) and so also the biases to these neurons in context layer are kept as zero. Once the architecture is finalized this way, the network is trained. Upon training with the same values of η and α as that used for partially trained ANN, the SNN so formulated is found to converge excellently. The convergences of SNNs so modeled are found to perform better as compared to the parent ANNs from which these have been modeled. The performance of the SNNs modeled from the parent ANNs is demonstrated using the following cases. We have named this SNN as Hybrid RNN (HRNN).

### 3.1. Modeling of Hybrid RNN for Grain size prediction:

#### 3.1.1 HRNN with 4-7-4-2 architecture:

The Elman RNN network selected has 4 input neurons, 7 neurons in hidden layer 1, 4 in hidden layer2 and 2 output neurons. The network was trained with learning rate parameter as 0.85 and momentum factor 0.65. Initially the network is trained as an ANN network. Table: 2 gives the details of the error in prediction in terms of MSE existing at various stages of network training. The ANN network converges to a MSE of 0.000362612 after 15 lakh epochs. To obtain the HRNN the ANN is now trained till 50000 epochs. The weights of this ANN are then borrowed in the input weights file for RNN training. The network is found to oscillate after it reaches a MSE of 0.00205. This happens, probably due to the fact that the ANN training up to 50000 epochs has not provided sufficient gradient descent on the MSE weights plane for the Hybrid RNN to further travel in the direction of negative gradient.

Further to this, the ANN was trained up to 100000 epochs and the weights were borrowed in the input weights file for RNN training. The results were better than the first case, but the convergence was found to be slow. After training for150000 epochs, the MSE is found to be 0.0013. Hence in the next step the ANN is trained for 500000 epochs and subsequently, the weights are borrowed in the input file for RNN training. It was found that after training for around 227000 epochs, the network converged satisfactorily.

#### 3.1.2 HRNN with 4-9-9-2 architecture:

Here the network is selected with 4 input neurons, 9 neurons each in second and third layer and 2 output neurons. Here the number of input patterns is taken as 240, just to emphasize that the number of input vectors has no great bearing on the convergence of Hybrid RNN. The ANN network is first trained until 50000 epochs and then hybrid RNN is constructed by borrowing weights of trained ANN. The network is further trained for 85000 epochs as it gave the same MSE as that of parent ANN when trained to 277356 epochs.

**Table 2: Variation of MSE with number of epochs**

| Sr. | Number of | MSE |
|---|---|---|
| 1 | 1 | 0.329771 |
| 2 | 100000 | 0.00467185 |
| 3 | 200000 | 0.002215 |
| 4 | 300000 | 0.00153321 |
| 5 | 400000 | 0.0014147 |
| 6 | 500000 | 0.00129209 |
| 7 | 600000 | 0.00105979 |
| 8 | 700000 | 0.000684544 |
| 9 | 800000 | 0.00056784 |
| 10 | 1500000 | 0.000362612 |

### 3.1.2 Modeling of Hybrid RNN for Hardness prediction:

In a manner similar to the Hybrid RNN formulated for Grain size prediction, Hybrid RNN is constructed for Hardness as will be discussed in forthcoming sections.

#### 3.2.1 HRNN with 5-5-3-1 architecture:

Initially, the ANN for hardness prediction was trained with architecture of 5 input neurons, 5 and 3 neurons in hidden layer1 and 2 respectively and one output neuron. The network is trained to achieve a MSE of $09.75*10^{-5}$ after 325000 epochs. The Hybrid RNN was constructed after training the above ANN for 50000 epochs in the same fashion as discussed earlier for grain size prediction. After around 200000 epochs, the hybrid RNN gave the same MSE as parent ANN.

#### 3.2.2 HRNN with 5-9-6-1 architecture:

Hybrid RNN is constructed from ANN having architecture as input layer 1: 5 neurons, Layer 2: 9 neurons, Layer 3: 6 neurons, Output: 1 neuron. The Hybrid RNN was formulated after just 5000 epochs, when a steep downward trend was observed in MSE at a rapid pace. The Hybrid RNN gave a MSE of $9.8774*10^{-5-}$ after 124000 epochs while to obtain the same value of MSE 345000 epochs were required for training the ANN.

## 4. Results and Discussions:

### 4.1. Grain Size Predictions:

#### 4.1.1 HRNN with 4-7-4-2 architecture:

The results of the predictions and the comparison between the ANN network after 15 lakh epochs and HRNN after 2.24 lakh epochs is present in Table: 3 below.

It is seen that the maximum error is |0.58| % at 5% thickness reduction with 10% liquid volume fraction in the as cast composite for small grain size, while for large grain size it is also at the same location. However in the majority of the cases, the error with hybrid RNN is within |0.5|%. The HRNN is modelled after borrowing the weights from partially trained ANN after 5 lakh epochs. Further, the HRNN converged nicely to a MSE of 0.000362612. Thus to achieve the same degree of convergence HRNN has consumed 776000 lesser epochs as compared to that of parent ANN, thus giving a saving of more than 50% of computational time. Furthermore, the error in predictions too is quite insignificant in comparison with the parent ANN predictions for the same data.

#### 4.1.2 HRNN with 4-9-9-2 architecture:

The comparison of predictions between the parent ANN and Hybrid RNN is given in Table: 4 below for a MSE of 0.0001762 achieved by parent ANN after being trained using 277356 epochs. The HRNN was trained with 240 patterns to emphasize that the data set available for training has no much bearing on the implementation of the model. The total number of epochs of HRNN coupled with partially trained ANN works out to 135000 epochs, thus giving a saving in computation time in excess of 50%. It can be seen that the error in estimation with hybrid RNN with respect to Parent ANN is within 6%, while in majority of the cases the error is within 1%.

## 4.2 HRNN for Hardness predictions:

In a manner similar to the Hybrid RNN formulated for Grain size prediction, Hybrid RNN is constructed for Hardness predictions.

### 4.2.1 HRNN with 5-5-3-1 architecture:

Initially, the ANN for hardness prediction was trained with architecture of 5 input neurons, 5 and 3 neurons in hidden layer1 and 2 respectively and one output neuron. The network was trained to achieve a MSE of $09.75*10^{-5}$ after 325000 epochs. . The Hybrid RNN was then constructed after training the above parent ANN for 50000 epochs in the same fashion as discussed earlier for grain size prediction. After around 200000 epochs, the hybrid RNN gave the same MSE. The comparison of the predictions done by the parent ANN and the HRNN for hardness prediction with 5-5-3-1 architecture is presented in Table: 5. It can be seen that the error in estimation lies between |2.5%|, while the computational time is saved by 23%. It can also be seen that in majority of the cases, the error is within|1%|. Fig: 4 shows the graph of comparison of variation of hardness predicted by ANN and Hybrid RNN for as cast Al-4.5Cu-5$TiB_2$ composite when rolled from mushy state with various liquid volume fractions at the initiation of rolling with 5-5-3-1 architecture. It can be seen from the graph that the plots for predictions with ANN and that with Hybrid RNN follow each other very closely. Maximum deviation is observed in case of predictions at 30% liquid volume fraction in the vicinity of 5% thickness reduction. Fig: 5 shows the plot of comparison of variation of hardness as predicted by ANN and Hybrid RNN with 5-5-3-1 architecture when Al-4.5Cu-5$TiB_2$ composite in pre hot rolled condition is rolled from mushy state at various thickness reductions. The rolling is initiated at mushy state corresponding to 10%, 20% and 30% liquid volume fractions respectively. It can be seen from Fig: 5 that the plots for predictions with ANN and hybrid RNN are almost identical indicating that the learning has been adequate and that both the networks have generalized quite nicely.

### 4.2.2 HRNN with 5-9-6-1 architecture:

Hybrid RNN is constructed from ANN having architecture as 5 neurons in input layer, 9 neurons in Layer 2, 6 neurons in Layer 3, 1 neuron in Output layer for hardness . The network was formulated after just 5000 epochs, when the down trend was observed in MSE at a rapid pace. The Hybrid RNN gave a MSE of $9.8774*10^{-5}$ after 124000 epochs while to obtain the same MSE, 345000 epochs of ANN were required. Table: 6 gives the relative performance in prediction of hardness with HRNN with the parent

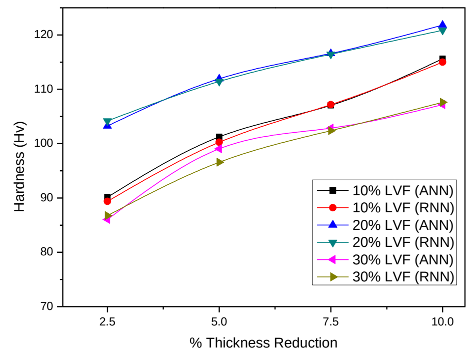


**Fig4: Plot showing comparison of ANN and Hybrid RNN for hardness prediction when as cast Al-4.5Cu-5$TiB_2$ composite is rolled from mushy state with various thickness reductions**

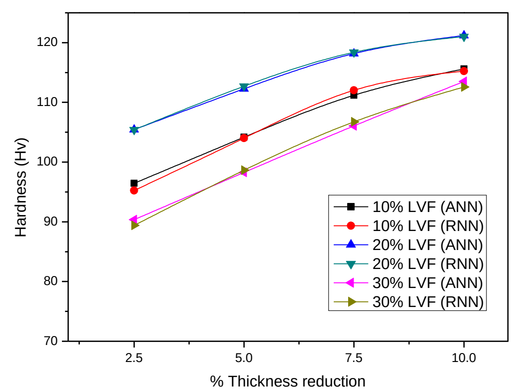


**Fig5: Plot showing comparison of ANN and Hybrid RNN for hardness prediction when pre hot rolled Al -4.5Cu-5$TiB_2$ composite is rolled from mushy state with various thickness reductions**

**Table3. Comparison of grain sizes by ANN Hybrid RNN after 2.24 lakh epochs of HRNN and 15 lakh epochs of ANN**

| | Input | | | output | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Grain Sizes | | | | | |
| | | | | large | | % Difference over ANN | Small | | % Difference over ANN |
| Sr. No. | Mat / Process | % TR* | %LVF** | ANN | RNN | | ANN | RNN | |
| 1 | 1 | 2.5 | 10 | 61.218 | 61.101 | 0.190139 | 28.124 | 28.037 | 0.307209 |
| 2 | 1 | 2.5 | 20 | 58.339 | 58.323 | 0.026569 | 33.370 | 33.364 | 0.018879 |
| 3 | 1 | 2.5 | 30 | 66.012 | 65.976 | 0.054232 | 36.223 | 36.2 | 0.064047 |
| 4 | 1 | 5 | 10 | 54.532 | 54.235 | 0.54335 | 23.870 | 23.732 | 0.578546 |
| 5 | 1 | 5 | 20 | 50.537 | 50.525 | 0.023745 | 28.842 | 28.841 | 0.004854 |
| 6 | 1 | 5 | 30 | 55.643 | 56.666 | -1.83923 | 29.867 | 29.86 | 0.024776 |
| 7 | 1 | 7.5 | 10 | 61.727 | 61.623 | 0.168968 | 31.654 | 31.586 | 0.21482 |
| 8 | 1 | 7.5 | 20 | 47.586 | 47.582 | 0.007986 | 26.494 | 26.497 | -0.01057 |
| 9 | 1 | 7.5 | 30 | 53.171 | 53.139 | 0.060559 | 27.758 | 27.751 | 0.022696 |
| 10 | 1 | 10 | 10 | 48.363 | 48.361 | 0.004135 | 31.307 | 31.306 | 0.002236 |
| 11 | 1 | 10 | 20 | 48.280 | 48.275 | 0.010149 | 28.316 | 28.318 | -0.00671 |
| 12 | 1 | 10 | 30 | 52.639 | 52.609 | 0.057751 | 27.881 | 27.875 | 0.020085 |
| 13 | 2 | 2.5 | 10 | 43.255 | 43.218 | 0.085769 | 27.893 | 27.880 | 0.047323 |
| 14 | 2 | 2.5 | 20 | 43.189 | 43.194 | -0.01343 | 27.267 | 27.271 | -0.0154 |
| 15 | 2 | 2.5 | 30 | 47.313 | 47.305 | 0.018177 | 26.775 | 26.776 | -0.00486 |
| 16 | 2 | 5 | 10 | 41.217 | 41.203 | 0.032511 | 26.380 | 26.377 | 0.013267 |
| 17 | 2 | 5 | 20 | 41.050 | 41.059 | -0.02168 | 25.334 | 25.340 | -0.02566 |
| 18 | 2 | 5 | 30 | 44.999 | 44.990 | 0.021111 | 25.377 | 25.380 | -0.01025 |
| 19 | 2 | 7.5 | 10 | 41.790 | 41.793 | -0.00885 | 26.301 | 26.305 | -0.01369 |
| 20 | 2 | 7.5 | 20 | 39.705 | 39.716 | -0.0267 | 24.067 | 24.075 | -0.03282 |
| 21 | 2 | 7.5 | 30 | 44.711 | 44.701 | 0.022366 | 24.925 | 24.928 | -0.01163 |
| 22 | 2 | 10 | 10 | 46.251 | 46.255 | -0.00757 | 30.492 | 30.494 | -0.00394 |
| 23 | 2 | 10 | 20 | 42.351 | 42.359 | -0.01983 | 26.462 | 26.468 | -0.02381 |
| 24 | 2 | 10 | 30 | 45.599 | 45.588 | 0.025658 | 25.617 | 25.619 | -0.00781 |

% TR* - % Thickness Reduction                % LVF** - % liquid Volume Fraction

**Table: 4 Comparison of grain size predicted by ANN & hybrid RNN after MSE = 0.0001762**

| | Input | | | Output | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Grain Sizes | | | | | |
| | | | | large | | % Difference over ANN | Small | | % Difference over ANN |
| Sr. No. | Mat / Process | % TR* | %LVF** | ANN | RNN | | ANN | RNN | |
| 1 | 1 | 2.5 | 10 | 54.1875 | 54.2067 | -0.03543 | 29.5234 | 29.484 | 0.133453 |
| 2 | 1 | 2.5 | 20 | 52.6612 | 52.6909 | -0.0564 | 34.1046 | 34.0894 | 0.044569 |
| 3 | 1 | 2.5 | 30 | 59.9039 | 59.9517 | -0.07979 | 37.5049 | 37.7475 | -0.64685 |
| 4 | 1 | 5 | 10 | 51.05 | 51.0677 | -0.03467 | 29.0427 | 29.0233 | 0.066798 |
| 5 | 1 | 5 | 20 | 46.2683 | 45.6567 | 1.321855 | 30.6739 | 30.6757 | -0.00587 |
| 6 | 1 | 5 | 30 | 52.2429 | 52.2859 | -0.08231 | 33.6145 | 31.3427 | 6.758393 |
| 7 | 1 | 7.5 | 10 | 49.4019 | 49.4101 | -0.0166 | 29.4733 | 29.4526 | 0.070233 |
| 8 | 1 | 7.5 | 20 | 43.8056 | 43.8131 | -0.01712 | 29.4299 | 29.4262 | 0.012572 |
| 9 | 1 | 7.5 | 30 | 49.2255 | 49.2655 | -0.08126 | 28.0999 | 28.3094 | -0.74555 |
| 10 | 1 | 10 | 10 | 49.0084 | 49.0092 | -0.00163 | 30.5889 | 30.5638 | 0.082056 |
| 11 | 1 | 10 | 20 | 43.7229 | 43.7293 | -0.01464 | 29.9121 | 29.9042 | 0.026411 |
| 12 | 1 | 10 | 30 | 49.1137 | 49.4527 | -0.69024 | 27.3891 | 27.5708 | -0.6634 |
| 13 | 2 | 2.5 | 10 | 44.8487 | 44.8163 | 0.072243 | 24.5998 | 24.5705 | 0.119107 |
| 14 | 2 | 2.5 | 20 | 42.8446 | 42.8538 | -0.02147 | 26.5484 | 26.5529 | -0.01695 |
| 15 | 2 | 2.5 | 30 | 48.2753 | 48.3038 | -0.05904 | 26.7442 | 26.9351 | -0.7138 |
| 16 | 2 | 5 | 10 | 43.5371 | 43.5299 | 0.016538 | 24.9919 | 24.9872 | 0.018806 |
| 17 | 2 | 5 | 20 | 39.5439 | 39.5377 | 0.015679 | 25.133 | 25.1385 | -0.02188 |
| 18 | 2 | 5 | 30 | 43.6984 | 43.7249 | -0.06064 | 24.7239 | 24.8992 | -0.70903 |
| 19 | 2 | 7.5 | 10 | 43.6721 | 43.6584 | 0.03137 | 26.2624 | 26.2512 | 0.042647 |
| 20 | 2 | 7.5 | 20 | 38.8598 | 38.8544 | 0.013896 | 25.3776 | 25.3831 | -0.02167 |
| 21 | 2 | 7.5 | 30 | 42.8648 | 42.8592 | 0.013064 | 25.0421 | 25.0383 | 0.015174 |
| 22 | 2 | 10 | 10 | 44.9272 | 44.9116 | 0.034723 | 28.1909 | 28.1709 | 0.070945 |
| 23 | 2 | 10 | 20 | 40.3236 | 40.3147 | 0.022071 | 26.947 | 26.9414 | 0.020782 |
| 24 | 2 | 10 | 30 | 44.3036 | 44.3185 | -0.03363 | 26.6612 | 26.7723 | -0.41671 |

% TR* - % Thickness Reduction    % LVF** - % liquid Volume Fraction

ANN with 5-9-6-1 architecture with same values of $\eta = 0.85$ and $\alpha = 0.65$. We see that the error in estimation lies within |4%|, while the computation time using Hybrid RNN is saved by around 80%.

### 4.3 Statistical Testing:

In the statistical analysis carried out, three types of tests were conducted which are listed as under:

1. To test equality of two means by using the two sample student_t test.

ISSN: 2278-0181
Vol. 1 Issue 5, July - 2012

**Table5. Comparison of ANN & hybrid RNN for hardness prediction using 5-5-3-1 architecture**

| | Input | | | | | Output | | % DIFFERENCE OVER ANN VALUE |
|---|---|---|---|---|---|---|---|---|
| Sr. No. | Materia l/ | % TR* | %LVF** | Grain Sizes | | Hardness | | |
| | | | | large | Small | ANN | RNN | |
| 1 | 1 | 0 | 0 | 50 | 50 | 78.0613 | 77.9555 | 0.135535 |
| 2 | 1 | 2.5 | 10 | 62 | 27 | 90.1529 | 89.3799 | 0.857432 |
| 3 | 1 | 2.5 | 20 | 58 | 33 | 103.241 | 104.161 | -0.89112 |
| 4 | 1 | 2.5 | 30 | 66 | 37 | 86.037 | 86.7449 | -0.82279 |
| 5 | 1 | 5 | 10 | 54 | 25 | 101.248 | 100.237 | 0.998538 |
| 6 | 1 | 5 | 20 | 51 | 31 | 111.916 | 111.436 | 0.428893 |
| 7 | 1 | 5 | 30 | 55 | 32 | 99.031 | 96.5667 | 2.488413 |
| 8 | 1 | 7.5 | 10 | 62 | 32 | 107.057 | 107.183 | -0.11769 |
| 9 | 1 | 7.5 | 20 | 48 | 26 | 116.601 | 116.431 | 0.145796 |
| 10 | 1 | 7.5 | 30 | 53 | 27 | 102.854 | 102.386 | 0.455014 |
| 11 | 1 | 10 | 10 | 49 | 29 | 115.604 | 114.982 | 0.538044 |
| 12 | 1 | 10 | 20 | 47 | 30 | 121.807 | 120.859 | 0.77828 |
| 13 | 1 | 10 | 30 | 54 | 26 | 107.18 | 107.595 | -0.3872 |
| 14 | 2 | 0 | 0 | 52 | 28 | 84.6917 | 84.7368 | -0.05325 |
| 15 | 2 | 2.5 | 10 | 43 | 27 | 96.4489 | 95.235 | 1.258594 |
| 16 | 2 | 2.5 | 20 | 42 | 26 | 105.44 | 105.405 | 0.033194 |
| 17 | 2 | 2.5 | 30 | 47 | 25 | 90.347 | 89.418 | 1.028258 |
| 18 | 2 | 5 | 10 | 42 | 26 | 104.18 | 104.019 | 0.15454 |
| 19 | 2 | 5 | 20 | 41 | 25 | 112.271 | 112.693 | -0.37588 |
| 20 | 2 | 5 | 30 | 46 | 24 | 98.266 | 98.6478 | -0.38854 |
| 21 | 2 | 7.5 | 10 | 40 | 26 | 111.18 | 112.016 | -0.75193 |
| 22 | 2 | 7.5 | 20 | 39 | 25 | 118.164 | 118.393 | -0.1938 |
| 23 | 2 | 7.5 | 30 | 45 | 24 | 106.062 | 106.752 | -0.65056 |
| 24 | 2 | 10 | 10 | 47 | 32 | 115.232 | 115.346 | -0.09893 |
| 25 | 2 | 10 | 20 | 43 | 25 | 121.191 | 121.019 | 0.141925 |
| 26 | 2 | 10 | 30 | 45 | 27 | 113.495 | 112.567 | 0.817657 |

% TR* - % Thickness Reduction % LVF** - % liquid Volume Fraction

**Table6. Comparison of ANN & hybrid RNN for hardness prediction using 5-9-6-1 architecture**

| | Input | | | | | Output | | % DIFFERENCE OVER ANN VALUE |
|---|---|---|---|---|---|---|---|---|
| Sr. No. | Mat / Process | % TR* | %LVF** | Grain Sizes | | Hardness | | |
| | | | | large | Small | ANN | RNN | |
| 1 | 1 | 0 | 0 | 50 | 50 | 67.9705 | 69.231 | -1.85448 |
| 2 | 1 | 2.5 | 10 | 62 | 27 | 89.7614 | 93.22 | -3.8531 |
| 3 | 1 | 2.5 | 20 | 58 | 33 | 103.16 | 101.879 | 1.24176 |
| 4 | 1 | 2.5 | 30 | 66 | 37 | 86.0228 | 87.2178 | -1.38917 |
| 5 | 1 | 5 | 10 | 54 | 25 | 99.6521 | 101.178 | -1.53123 |
| 6 | 1 | 5 | 20 | 51 | 31 | 111.04 | 112.874 | -1.65166 |
| 7 | 1 | 5 | 30 | 55 | 32 | 96.483 | 96.8224 | -0.35177 |
| 8 | 1 | 7.5 | 10 | 62 | 32 | 107.059 | 106.62 | 0.410054 |
| 9 | 1 | 7.5 | 20 | 48 | 26 | 115.966 | 116.751 | -0.67692 |
| 10 | 1 | 7.5 | 30 | 53 | 27 | 102.586 | 102.287 | 0.291463 |
| 11 | 1 | 10 | 10 | 49 | 29 | 115.246 | 117.323 | -1.80223 |
| 12 | 1 | 10 | 20 | 47 | 30 | 120.33 | 119.63 | 0.581734 |
| 13 | 1 | 10 | 30 | 54 | 26 | 107.765 | 108.413 | -0.60131 |
| 14 | 2 | 0 | 0 | 52 | 28 | 57.2824 | 58.824 | -2.69123 |
| 15 | 2 | 2.5 | 10 | 43 | 27 | 95.179 | 98.8946 | -3.9038 |
| 16 | 2 | 2.5 | 20 | 42 | 26 | 106.801 | 105.267 | 1.436316 |
| 17 | 2 | 2.5 | 30 | 47 | 25 | 90.3766 | 91.543 | -1.2906 |
| 18 | 2 | 5 | 10 | 42 | 26 | 103.904 | 104.088 | -0.17709 |
| 19 | 2 | 5 | 20 | 41 | 25 | 113.233 | 112.989 | 0.215485 |
| 20 | 2 | 5 | 30 | 46 | 24 | 98.7331 | 97.684 | 1.062562 |
| 21 | 2 | 7.5 | 10 | 40 | 26 | 111.772 | 110.56 | 1.08435 |
| 22 | 2 | 7.5 | 20 | 39 | 25 | 118.619 | 117.936 | 0.575793 |
| 23 | 2 | 7.5 | 30 | 45 | 24 | 106.473 | 105.934 | 0.506232 |
| 24 | 2 | 10 | 10 | 47 | 32 | 115.947 | 115.17 | 0.670134 |
| 25 | 2 | 10 | 20 | 43 | 25 | 120.866 | 119.493 | 1.135969 |
| 26 | 2 | 10 | 30 | 45 | 27 | 112.755 | 114.811 | -1.82342 |

% TR* - % Thickness Reduction        % LVF** - % liquid Volume Fraction

2. To test if the given population has standard normal distribution using one sample Kolmogorov – Smirnov test.
3. Testing if the two populations belong to the same continuous distribution using two samples Kolmogorov – Smirnov test.

K-s test for single sample and two samples were used to test the normality of the distribution of errors in prediction of grain sizes (Large and Small grain sizes) and hardness as predicted by ANN and HRNN with different architectures mentioned at 3.1 and 3.2 above. The errors were calculated between the predicted values obtained from ANN and actual values and also between the predicted values obtained from HRNN and actual values. A third type of error was calculated between the values predicted by HRNN and ANN. Further to this, two sample student_t – test was performed for testing the equality of means of populations representing the errors between ANN and HRNN predictions for large grain size, small grain size and hardness. To be able to analyze the data more meaningfully, the population of errors was divided into errors in predictions for as cast Al-4.5Cu-5$TiB_2$ composite and pre hot rolled Al-4.5Cu-5$TiB_2$ composite. For the purpose of statistical analysis of the performance of HRNN with the parent ANN, the following terminology of errors is defined.

a. Error (a): Error in prediction of large grain size and small grain size by ANN with 4-7-4-2 architecture over the target values.
b. Error (b); Error in prediction of large grain size and small grain size by HRNN with 4-7-4-2 architecture over the target values.
c. Error(c); Error in prediction of large grain size and small grain size by ANN with 4-9-9-2 architecture over the target values.
d. Error (d); Error in prediction of large grain size and small grain size by HRNN with 4-9-9-2 architecture over the target values.
e. Error (e); Error in prediction of hardness by ANN with 5-5-3-1 architecture over the target values.
f. Error (e); Error in prediction of hardness by HRNN with 5-5-3-1 architecture over the target values.
g. Error (e); Error in prediction of hardness by ANN with 5-9-6-1 architecture over the target values.
h. Error (e); Error in prediction of hardness by HRNN with 5-9-6-1 architecture over the target values.

#### 4.3.1 Testing of equality of means:

Table: 7 gives the results of the two sample student_t test performed on the population deduced from the predictions of HRNN and ANN with different architectures. It can be seen that the means of the HRNN prediction populations are comparable to the mean values of populations obtained from ANN predictions for similar architectures. Furthermore, the standard deviation values also are comparable for similar architectures of ANN and HRNN.

In testing of hypothesis, the strength of the conclusion is decided by level of significance α. The popular value of α is 0.05. The decision about the test is based on the value of the test statistics obtained from the sample and the benchmark value of appropriate test statistics obtained from the tables, using α and degrees of freedom (which can always be obtained from the size of sample).

However, while reporting the conclusion, the value of test statistic obtained is never reported. As a result, closeness of this test value obtained from the sample and the bench mark value of appropriate test statistic also does not get reported.

This difficulty is overcome when p value of test is indicated. The p value indicates the probability of obtaining a test statistic as extreme as the one actually observed, assuming that the null hypothesis is true.

#### 4.3.2 Testing errors for standard normal distribution :

Testing of the error distribution of ANN and HRNN predictions over target values of large grain sizes, small grain sizes and hardness was carried out using one sample Kolmogorov – Smirnov test. For this purpose, the various errors as defined in section 4.3 were considered. The results of the test are tabulated in Table 8. It can be seen that with a confidence of 5% ($\alpha = 0.05$), used for the test, all the error distributions i.e. Error (a) to Error (h) are accepted as standard normal distributions. Table 8 gives the p values for all the error distributions. $H_0$ indicates that the error population has normal distribution, while $H_1$ indicates that the error population does not have normal distribution.

#### 4.3.3 Testing for equality of continuous distributions using Two Sample Kolmogorov – Smirnov test.

Here the distribution of errors between ANN predicted values for grain sizes using 4-7-4-2 and 4-9-9-2 architectures over target values of grain sizes were checked with their counterparts predicted by HRNN with similar architectures, for equality. The same test was carried out to check the equality of similar error distributions obtained using HRNN and ANN for hardness using 5-5-3-1 and 5-9-6-1 architectures. The results of this test are presented in Table 9. $H_0$ indicates that the two error distributions under test are equal, while $H_1$ indicates that they are not equal.

**TABLE7. Results of two sample student_t test**

| Architecture | | Large grain size / hardness (HRNN) **Population 1** | | Small grain size/hardness (ANN) **Population 2** | |
|---|---|---|---|---|---|
| | | As cast | Pre hot rolled | As cast | Pre hot rolled |
| 4-7-4-2 | Mean: | -0.0334 | 0.0034 | -0.0136 | 0.0047 |
| | Stdv: | 0.0401 | 0.0068 | 0.0244 | 0.0026 |
| 4-9-9-2 | Mean: | -0.0201 | -0.0134 | -0.0053 | 0.0052 |
| | Stdv: | 0.0099 | 0.0073 | 0.0022 | 0.0015 |
| 5-5-3-1 | Mean: | 0.8248 | | | 1.2176 |
| | Stdv: | 2.8327 | | | 2.0481 |
| 5-9-6-1 | Mean: | -1.9638 | | | -2.6684 |
| | Stdv: | 2.3999 | | | 2.1505 |

**TABLE8. Results of one sample Kolmogorov – Smirnov test.**

| Test No. | α | Errors | p value | | Conclusion |
|---|---|---|---|---|---|
| | | | Large grain size / hardness | Small grain size | |
| A | 0.05 | Error(a) | 0.7366 | 0.8174 | $H_0$: Accepted |
| B | 0.05 | Error(b) | 0.8174 | 0.0940 | $H_0$: Accepted |
| C | 0.05 | Error(c) | 0.1879 | 0.1860 | $H_0$: Accepted |
| D | 0.05 | Error(d) | 0.1983 | 0.1901 | $H_0$: Accepted |
| E | 0.05 | Error(e) | 0.5502 | | $H_0$: Accepted |
| F | 0.05 | Error(f) | 0.2993 | | $H_0$: Accepted |
| G | 0.05 | Error(g) | 0.3258 | | $H_0$: Accepted |
| H | 0.05 | Error(h) | 0.1499 | | $H_0$: Accepted |

**TABLE9. Results of two sample Kolmogorov – Smirnov test.**

| Test No. | Population of errors tested | α | p value | | Conclusion |
|---|---|---|---|---|---|
| | | | Large grain size / hardness | Small grain size | |
| A | Error(a) v/s Error(b) | 0.05 | 0.9999 | 0.1094 | $H_0$: Accepted |
| B | Error(a) v/s Error(b) | 0.05 | 0.9999 | 0.9999 | $H_0$: Accepted |
| C | Error(a) v/s Error(b) | 0.05 | 0.2581 | | $H_0$: Accepted |
| D | Error(a) v/s Error(b) | 0.05 | 0.8922 | | $H_0$: Accepted |

## 5. Conclusions:

1. It is seen that the Simple Elman Recurrent Network does not necessarily converge for all the applications, as is seen in the present case. The SRN modeled for the predictions of grain sizes and hardness failed to converge despite all possible architectures being tried with various combinations of learning rate parameter (η) and momentum factor (α).
2. The slight modification in the network architecture with the neurons in the hidden layers giving feed back to itself resulted in

slightly better convergence, but the learning was not good enough to do correct mappings.

3. A Hybrid Recurrent Network constructed by borrowing weights form a partially trained FFNN with similar architecture is found to excellently converge and is able to predict the outputs comparable with those predicted by the parent FFNN / ANN. The Network training time is drastically reduced by employing such Hybrid Recurrent Neural Networks as have been demonstrated by the four cases considered.
4. The test on normality of the errors justifies the stability of the model. Secondly, the test on the two means confirms that the HRNN and the parent FFNN are not significantly different in behavior. Likewise, the distribution of errors obtained with parent FFNN and HRNN also confirm that they have equivalent performance capabilities. This equivalent performance coupled with the reduced learning time provides a strong potential for use of HRNN in real time process control applications.

## 6. References:


1. J. Larkiola, P. Myllykoski, A. Korhonen, L. Cser, “The role of neural Networks in the optimization of rolling process”, Jour. Of Material Procss. Tech., 80-81 (1998), pp. 16-23.
2. J. Kusiak, R. Kuziaak, “Modelling of Microstructure and mechanical properties of steel using the artificial neural network”, Jour. Of Material Procss. Tech., 127, (2002), pp. 115-121.
3. S. Mandal,P. Sivprasad, S. Venugopal ,K. Murthy, “Artificial neural network modeling to evaluate and predict the deformation behavior of stainless steel type AISI 304L during hot torsion”, Applied Sci. Computing, 9 (2009), pp. 237-244.
4. N. Reddy, A. Prasad Rao, M. Chakraborty, B. Murty, “Prediction of grain size of Al-7Si alloy by neural networks”, Material science and Engineering, A 391 (2005), pp. 131-140.
5. Zurada J. M., Introduction to artificial neural Systems, PWS Publishing Company, Boston, 1992.
6. R. P. Lippman, “An Introduction to Computing with neural nets”, IEEE ASSp. Magazine, (1987), pp. 4-22.
7. K. Hornik, H. white, “Multilayer feedforward networks are universal approximators”, Neural Networks, 2(1989), pp. 359-366.
8. S. Roberts, J. Kusiak, Y. Liu, A. Forcellese, P. withers, “Prediction of damage evolution in forged aluminium metal matrix composites using a neural network”, Jour. Of Material Procss. Tech., 80-81 (1998), pp. 507-512.
9. M. Haque, K. sudhakar, “ANN back-propagation prediction model for fracture toughness in microalloy steel”, International Journal of Fatigue, 24 (2002), pp. 1003-1010.
10. N. Altinkok, R. Koker, “Modelling the prediction of tensile and density properties in particle reinforced metal matrix composites by using neural networks”, Materials and Design, 27 (2006), pp. 625-631.
11. N.S. Reddy, Y. H. Lee , C. H. Park, S. C. Lee, “Prediction of flow stress in Ti-6Al-4V alloy with an equiaxed $\alpha+\beta$ microstructure by artificial neural networks”, Mater. Sci. and Engg. A (2008), pp. 276-282.
12. C. Giles, C. Miller et.al., “Learning and extracting finite state automats with second - order recurrent neural networks”, Neural Computation, 4 (1992), pp. 393-405.
13. G. Tesauro, D. Touretzky, T. Leen, “An Experimental Comparison of Recurrent Neural Networks”, MIT Press (1995), pp.697-704.
14. J. Elman, “Finding structure in time”, Cognitive Science, 14 (1990), pp. 179-211.
15. P. Frasconi, M. Gori, S. Goda, “Local Feedback Multi-layered Networks”, Neural Computation, MIT Press 4 (1992), pp. 120-130.
16. K. Lang, A. Waibel, G. Hinton, “A time delay neural network architecture for isolated word recognition”, Neural Networks, 3 (1990), pp. 23-44.
17. K. Narendra, K. Parthasarathy, “Identification and control of dynamical systems using neural networks. IEEE Transactions on Neural Networks”, IEEE Transactions on Neural Networks, 1 (1990), pp. 4-27.
18. S. Kremer, “On the computational power of Elman – style recurrent networks”, IEEE Transactions on Neural Networks, 6(4) (1995), pp. 1000-1004.
19. Q. Song, Y. Soh, L. Zhao, “A robust extended Elman backpropagation algorithm”, Proceed. Of Inter. Joint Conference on Neural Networks, Atlanta, Georgia, USA, June 14-19 (1990), pp. 2971-2977.
20. M. A. Herbert, “Some studies on the mushy state rolling of Al-4.5Cu alloy based in-situ composites reinforced with TiB 2 or TiC particles”, Ph. D. Thesis, IIT Kharagpur, India, 2008.

# LIST OF FIGURES