Analysis of MultiLayer Neural Networks with Direct and Cross Forward Connection

Stanisław Płaczek and Bijaya Adhikari
stanislaw.placzek@wp.pl, bijaya.adhikari1991@gmail.com

Abstract

Artificial Neural Networks are of much interest for many practical reasons. As of today, they are widely implemented. Of many possible ANNs, the most widely used one is the back-propagation model with direct connection. In this model the input layer is fed with input data and each subsequent layers are fed with the output of preceding layer. This model can be extended by feeding the input data to each layer. This article argues that this new model, named Cross Forward Connection, is optimal than the widely used Direct Connection.

1 Introduction

Artificial Neural Networks are implemented as universal approximator function with multidimensional variables. The function can be represented as:

$$Y = F(X) \tag{1}$$

where:

- X-input vector
- Y-output vector

Selecting a network to solve a specific problem is a tedious task. Decision regarding following thing must be made prior to attempting a solution.

- Structure of Neural Network, number of hidden layers and number of neurons in each layer. Conventionally, the size of input and output layers are defined by dimension of X and Y vectors respectively.
- Structure of individual neurons encompassing activation function, which takes requirement of learning algorithm into account.
- Data transfer methods between layers
- Optimization criteria and type of learning algorithm

Structure of Network can be defined in arbitrary ways to accomplish complex tasks. The structure plays vital role in determining the functionality of ANN. This paper will compare and contrast two multilayer network structures.

- Direct Connection: This structure consists of at-least one hidden layer. Data is fed from preceeding layer to succeeding one.

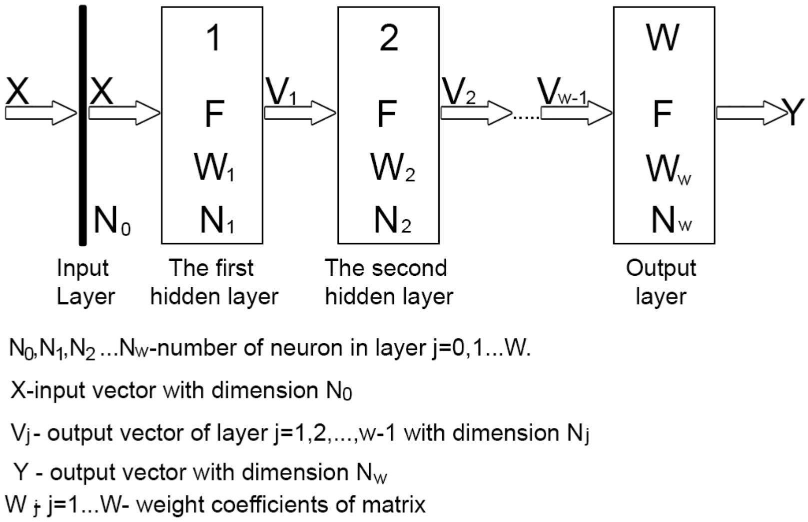

Figure 1: Structure of Direct Connection ANN

- Cross Forward Connection. In this structure, the input signal is passed on to every layer in the network. Therefore, a layer j=1,2,3.....W, where W is the output layer, has two inputs : vector X and vector V_{j-1}, output of preceding layer.

Structure of Cross Forward Connection is simpler than that of Direct Connection, in terms of neuron distribution in hidden layers. Learning time too is shorter for Cross Forward Connection . In later part of the paper, we will analyze a particular optimization problem for ANN where total number of neurons, N, and number of layers , W, are given. Our target is to maximize the total number of subspaces which are created by neurons of every hidden layers. We will solve this complex problem with respect to the relation between dimensionality of feature space, N_0, and number of neurons in all hidden layers, N_i. This problem can be divided into two sub-problems.

- $N_i \leq N_0$ – linear optimization problem,
- $Ni > N_0$ – non-linear optimization problem.

Where: i= 1,2,3,. . . . . . W-1.
We can solve linear target function using linear-programming method. The non-linear task, with linear constrains, can be solved using Kuhn- Tucker conditions. We have solved both sub-problems and discussed different ANN structures for each case and have presented them as examples. In conclusion section, we have summarized our results and have given recommendation for different ANN structures.

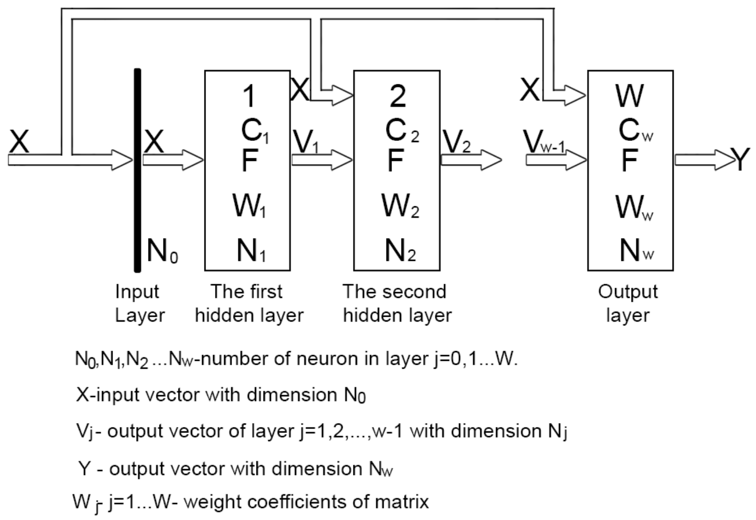

Figure 2: Structure of Cross Forward Connection ANN

2 Criteria of ANN structure selection

The threshold function for the each neuron is defined as follows:

$$g(x) = \begin{cases} 1, & \text{if } x > 0 \\ -1, & \text{if } x \leq 0 \end{cases} \tag{2}$$

We say that the network in Fig. 3 has structure 2-3-1. Where:

- N_0=2; number of neurons in input layer.
- N_1=3; number of neurons in hidden layer.
- N_2=1; number of neurons in output layer.

Signal transfer from input layer to output layer in this structure can be represented in the following way.

$$U = W_1 \cdot X \tag{3}$$

$$V = F_1(U) \tag{4}$$

$$E = W_2 \cdot V + C_2 \cdot X \tag{5}$$

$$Y = F_2(E) \tag{6}$$

Where,

- $X[0 : N_0]$ -input signal

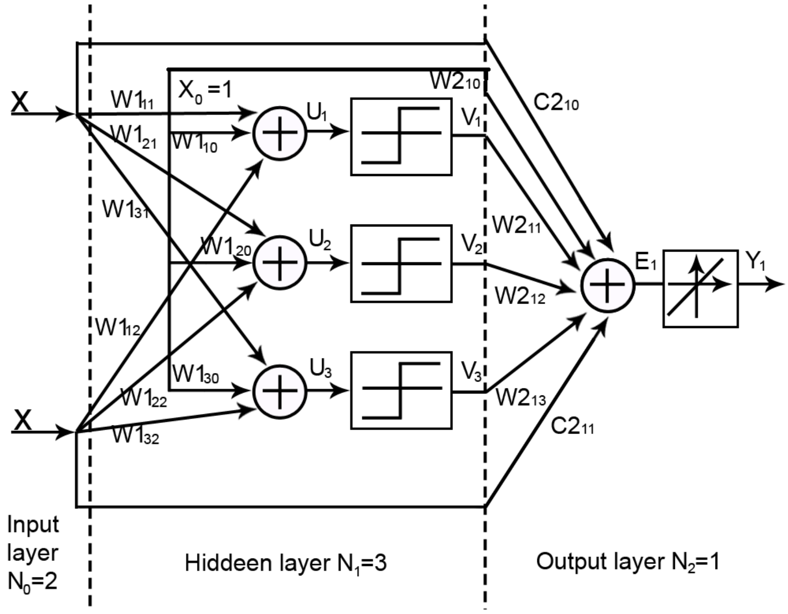

Figure 3: Two Layer ANN with Cross Forward Connection

- $W_1[1{:}N_1;0{:}N_0]$ - weight coefficients matrix of hidden layer
- $U[1{:}N_1]$-analog signal of hidden layer
- $V[1{:}N_1]$-output signal of hidden layer
- $W_2[1{:}N_2;0{:}N_1]$ - weight coefficients matrix of output layer
- $E[1{:}N_2]$-analog signal of output layer
- $Y[1{:}N_2]$-output signal of output layer
- $C_2[1 : N_2; 0 : N_0]$ -weight coefficients matrix of Cross connection

This network will be used for pattern recognition after being trained by teacher data.

The architecture of ANN in Fig.3 could be represented using hyper-spaces. Lets imagine a hyperspace having dimension of the number of neurons in the input layer. The first hidden layer, depicted in equation (3) and (4), divides feature space, X, into subspaces.

Two dimensional feature space is divided into seven sub-spaces. These sub-spaces correspond to internal structure of input data.

The function Φ (p,q) gives the maximum number of p dimensional sub-spaces formed when original p dimensional hyper-space is divided by q number of $p-1$ dimensional hyper-planes. The function has following recursive form.[3]

$$\Phi(p, q) = \Phi(p, q-1) + \Phi(p-1, q-1) \tag{7}$$

By definition of $\Phi(p, q)$, it is clear that

$$\Phi(p, 1) = 2 \tag{8}$$

q \ p	1	2	3	4	5	6	7	8	9	10
1	2	2	2	2	2	2	2	2	2	2
2	3	4	4	4	4	4	4	4	4	4
3	4	7	8	8	8	8	8	8	8	8
4	5	11	15	16	16	16	16	16	16	16
5	6	16	26	31	32	32	32	32	32	32
6	7	22	42	57	63	64	64	64	64	64
7	8	29	64	99	120	127	128	128	128	128
8	9	37	93	163	219	247	255	256	256	256
9	10	46	130	256	382	466	502	511	512	512
10	11	56	176	386	638	848	968	1013	1023	1024

Table 1: Number of sub spaces formed by division of p dimensional input vector by q neurons present in the first hidden layer

and

$$\Phi(1, q) = q + 1 \tag{9}$$

In context of Neural Networks, q is the number of neurons in the first hidden layer, N_i, and p is dimension of input vector, N_0.

Now, re-writing (7), we get:

$$\Phi(p, q) = \Phi(p, 1) + \sum_{k=1}^{q-1} \Phi(p-1, k) \tag{10}$$

Solving recursion (10), we get :

$$\Phi(p, q) = C_{q-1}^{p} + 2 \sum_{k=0}^{p-1} \cdot C_{q-1}^{k} \tag{11}$$

where,

$$C_n^k = \frac{n!}{k! \cdot (n-k)!} \quad \text{and } C_n^k = 0, \text{if } k > n. \tag{12}$$

In the equations above:

- p-dimension of input vector.
- q- number of neurons in hidden layer

Let us consider an example, for a network having three neurons in first hidden layer and input vector of dimension 2. From (11), We get $\Phi(2,3)=7$.

The number of subspaces formed due to division of the neurons in input layer by the neurons in the first hidden layer depends solely on the number of neurons. The table presented above shows number of subspaces for different values of p and q.

Coming back to the structure of Cross-Forward Connection, according to Fig.3, input signals to the second hidden layer can be divided into two subsets:

- input received from the output of previous layer- vector V
- raw input received - vector X

All input signals are multiplied by the adjustable weights of associated neurons i.e. matrices W_2 and C_2 respectively.

For ANN presented in Fig.3, we can write:

$$e_k = \sum_{i=1}^{N_1} W_{2k,i} \cdot V_i + \sum_{j=0}^{N_0} C_{2k,j} \cdot X_j \tag{13}$$

And, finally,

For e_k=0,

$$\sum_{j=0}^{N_0} C_{2k,j} \cdot X_j = -\sum_{i=1}^{N_1} W_{2k,i} \cdot V_i \tag{14}$$

The input space, X, in (14) represents bundles of hyper planes shifted by some vectors. The number of hyper-planes depend on V_i. For two dimension space, the second layer of ANN is composed of four parallel lines formed by all possible combination of values of V_i and V_j i.e.,0,0; 0,1; 1,0; 1,1.

Every subspace which is formed by the hidden layer is further divided into two smaller sub-spaces by output neuron. For N_0 dimensional input space and N_1 number of neurons in the first hidden layer, the maximum number of sub-spaces is given by:

$$\Psi(N_0, 2) = \Phi(N_0, N_1) \cdot \Phi(N_0, N_2) \tag{15}$$

For, W>2 ,number of sub-spaces is:

$$\Psi(N_0, W) = \prod_{i=1}^{W} \Phi(N_0, N_i) \tag{16}$$

The number of subspaces of initial feature space in Fig 3 is:

$$\Psi(2, 2) = \Phi(2, 3) \cdot \Phi(2, 1) = 7 * 2 = 14$$

For example, to divide input space into 14 subspaces, we require 3 neurons in the first hidden layer and 1 in output layer. Whereas, we need 5 neurons in the first hidden layer and 1 neuron in output layer to obtain the same number of subspaces in the standard direct connection.

3 Structure Optimization of Cross Forward Connection Network

ANN structure optimization is very complicated problem and can be solved in different ways. Experience has taught us that ANN with 1 or 2 hidden layer can solve most of the practical problems. The problem of ANN optimization structure can be described as :

- maximizing number of subspaces, $\Psi(N_0, W)$.

when total number of neurons, N, and number of layers, W, are given.

3.1 Calculation of the number of neurons for ANN with one hidden layer

For ANN with one hidden layer, the number of input neurons,N_0,is defined by the input vector X and is known as a priori. The number of output neurons N_2 is given by the output vector structure, Y, known as task definition. We can calculate the number of neurons in the hidden layer N_1 using equation 16. According to the optimization criteria and formula 16, the total number of subspaces for ANN with one hidden layer is given by:

$$\Psi(N_0, W) = \Psi(N_0, 2) = \Phi(N_0, N_1) \cdot \Phi(N_0, N_2) \tag{17}$$

Finally we can calculate number of neurons in the hidden layer N_1.

3.2 Optimization for more than one hidden layer

For ANN with 2 or more hidden layers, optimization is more complicated. We assume that:

- the number of layers W is given and,
- total number of neurons N is given.

N can be calculated using:

$$N = \sum_{i=1}^{W-1} N_i = N_1 + N_2 + N_3 + ... + N_{W-1} \tag{18}$$

In practice, we have to calculate neuron's distribution between $\{1 : W - 1\}$ layers. To find neuron's distribution, we have to maximize the number of subspaces according to the equation 19 with 20 as constraint.

$$\Psi(N_0, W-1)^{opt} = \max_{N_1, N_2 ... N_{W-1}} \prod_{i=1}^{W-1} \Phi_i(N_0, N_i) \tag{19}$$

$$N = \sum_{i=1}^{W-1} N_i = N_1 + N_2 + N_3 + ... + N_{W-1} \tag{20}$$

From 11 and 19,

$$\begin{gathered} \Phi(N_0, N_i) = C_{N_i-1}^{N_0} + 2 \sum_{k=0}^{N_0-1} \cdot C_{N_i-1}^{k} \\ for\ i\ \epsilon\ [1; W-1] \end{gathered} \tag{21}$$

Please note that:

$$\begin{aligned} C_{N_i-1}^{N_0} &= 0 \\ when\ N_i - 1 - N_0 &< 0 \\ N_i &\leq N_0 \end{aligned} \tag{22}$$

Taking 19, 20, 21, and 22 into account, our optimization task can be written as:

$$\Psi(N_0, W-1)^{opt} = \max_{N_1, N_2 \ldots N_{W-1}} \left\{ \prod_{i=1}^{W-1} [C_{N_i-1}^{N_0} + 2 \sum_{k=0}^{N_0-1} C_{N_i-1}^{k}] \right\} \tag{23}$$

with constraints

$$N = \sum_{i=1}^{W-1} N_i \tag{24}$$

$$C_{N_i-1}^{N_0} = 0 \ \ for \ \ N_i \leq N_0 \tag{25}$$

$$C_{N_i-1}^{k} = 0 \ \ for \ \ N_i \leq k \tag{26}$$

The optimization problem in 23 is non-linear and solution space can be divided into :

1. For all hidden layers $N_i \leq N_0$ and $N_i \leq k$ — linear task
2. For all hidden layers $N_i > N_0$ and $N_i > k$ — non-linear task

Set of hidden layers can be divided into two subspaces:

- $S1 = \{N_1, N_2, N_3, ..., N_j\}$ where $j \leq W-1$.For S_1, $N_i \leq N_0$ and $Ni \leq k$
- $S2 = \{N_{j+1}, N_{j+2}, N_{j+3}, ..., N_{W-1}\}$.For S_2, $N_i > N_0$ and $Ni > k$

Where W = number of layers and W-1 = number of hidden layers. This is a mixed structure, for which final solution can be found using mixture of both methods from point 1 and 2. Other cases not foreseen under S1/S2 split can be equivalently reduced to this split. However, we have not taken this case into account in this paper.

3.3 Neuron distribution in the hidden layers, where neurons' number for all hidden layers is less or equal than initial feature space

In this case, we have

$$N_i \leq N_0 \ for \ i \, \epsilon \{ 1; W-1\} \tag{27}$$

So, the total number of subspaces is defined by

$$\Phi(N_0, N_i) = \frac{(N_i-1)!}{N_0!(N_i-1-N_0)!} + 2 \cdot \sum_{k=0}^{N_0-1} \frac{(N_i-1)!}{k!(N_i-1-k)!} \tag{28}$$

or,

$$\Phi(N_0, N_i) = 0 + 2 \cdot 2^{N_i-1} = 2^{N_i} \tag{29}$$

Our optimization target can be written as,

$$\Psi(N_0, W-1)^{opt} = \max_{N_i \,\epsilon\, [1,W-1]} \left\{ \prod_{i=1}^{W-1} 2^{N_i} \right\} = \max_{N_i \,\epsilon\, [1,W-1]} \left\{ 2^{\sum_{i=1}^{W-1} N_i} \right\}$$
$$for\ N = \sum_{i=1}^{W-1} N_i$$
$$N_i \leq N_0\ and\ N_i, N_0 \geq 0 \tag{30}$$

Equation 30 is monotonously increasing and can be written as

$$\Psi(N_0, W-1)^{opt} = \max_{N_i \,\epsilon\, [1,W-1]} \left\{ \sum_{i=1}^{W-1} N_i \right\}$$
$$For\ N = \sum_{i=1}^{W-1} N_i \tag{31}$$
$$N_i \leq N_0\ and\ N_i, N_0 \geq 0$$

Under the given number of layers, total number of neurons have to satisfy following constraints

$$N_i \leq N_0\ and\ N \leq (W-1)N_0 \tag{32}$$

Example:
For ANN with $N_0 = 3, N_1 \leq 3, N_2 \leq 3, N_3 = 1$, $W = 3$, find optimum neurons distribution between two hidden layers N_1, N_2.

It is known that for output layer $N_3 = 1$ and therefore we will only consider two hidden layer for optimization process. For all N_i, where $i = 1, 2$ and $N_i \leq N_0$, using 35 we can write:

$$\begin{aligned}
&N \leq (W-1) \cdot N_0 = (3-1) \cdot 3 = 6 \\
&\text{Taking } N_0 = 3 \text{ using 31 we achieve,} \\
&\Psi(N_0, W-1) = \Psi(3, 2) = max\{N_1 + N_2\} \\
&\text{and constraints} \\
&N_1 \leq 3 \\
&N_2 \leq 3 \\
&we\,use\ N_1 + N_2 = 4 < 6
\end{aligned} \tag{33}$$

To solve this optimization task, we can use linear programming methods or use figure 5.

Using only discrete values of N_1, N_2 for N=4, we can find three solutions $(N_1, N_2) = \{(1, 3), (2, 2), (3, 1)\}$

The following equations indicate the number of subspaces for different number of neurons.

$$\begin{aligned}
\Phi(N_0, N_1) &= \Phi(3, 1) = 2^1 = 2 \\
\Phi(N_0, N_1) &= \Phi(3, 2) = 2^2 = 4 \\
\Phi(N_0, N_1) &= \Phi(3, 3) = 2^3 = 8
\end{aligned} \tag{34}$$

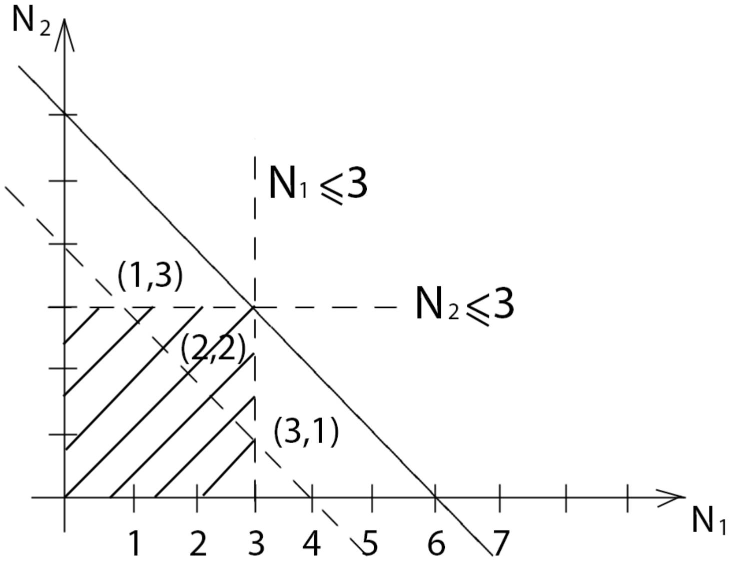

Figure 4: Graphical solution of linear programming when total number of neurons, N=6 and N=4

N_0	N_1	N_2	$\Phi(N_0, N_1)$	$\Phi(N_0, N_2)$	$\Psi(N_0, W-1)$
3	1	3	2	8	16
3	2	2	4	4	16
3	3	1	8	2	16

Table 2: Solution of linear programming for N=4

Finally, we have three optimal solutions with three different ANN structure. Every structure generates 16 subspaces and are equivalent. Please refer to Table 2.

In conclusion, we can say that for every given total number of neurons, N, we may have many possible optimal neurons distribution between layers. Optimal number of subspaces in the initial feature space has the same value of Ψ for all distributions.

3.4 Neurons distribution in the hidden layers, where neurons number for all hidden layers is greater than initial feature space

Let's assume number of layers, W=3. It implies that we have only two hidden layers. According formula 21,

$$
\begin{aligned}
\Phi(N_0, N_i) =& C_{N_i-1}^{N_0} + 2\sum_{k=0}^{N_0-1} C_{N_i-1}^{k} \\
& \text{for i } \epsilon \ [1 : W-1] \ \textit{and} \ N_i > N_0 \\
& \text{For whole ANN, total number of subspaces is given by} \\
\Psi(N_0, W-1) =& \Psi(N_0, 2) = \Phi_1(N_0, N_1) \cdot \Phi_2(N_0, N_2) \\
& \text{and } N_1 + N_2 = N \\
& \text{ so, } \ N_1 + N_2 > 2N_0
\end{aligned} \tag{35}
$$

Taking all assumptions into account we can write,

$$
\begin{aligned}
&\Phi(N_0, N_1) = C_{N_i-1}^{N_0} + 2 \cdot (C_{N_i-1}^{0} + C_{N_i-1}^{1} + ... + C_{N_i-1}^{N_0-1}) \ for \ N_0 < N_i \\
&\Phi(N_0, N_1) < C_{N_i-1}^{N_0} + 2 \cdot 2^{N_i-1} < 2^{N_i}
\end{aligned} \tag{36}
$$

In this situation we do not know how many subpaces there are for $\Phi(N_0, N_1)$. To find neurons distribution between the hidden layers we should know relations between N_0, N_i and N.

Example:
For N_0=3, W=3 N=8, and N=10, N=12 find neuron distribution in the layers, were $N_i > 3$. We should maximize the quality criterion

$$
\Psi(N_0, W-1)^{OPT} = \max_{N_1, N_2 .... N_{W-1}} \prod_{i=1}^{W_1} \left[C_{N_i-1}^{N_0} + 2 \cdot \sum_{k=0}^{N_0-1} C_{N_i-1}^{k} \right] \tag{37}
$$

For example,

$$
\Psi(3, 2)^{OPT} = \max_{N_1, N_2} \prod_{i=1}^{2} \left[C_{N_i-1}^{3} + 2 \cdot \sum_{k=0}^{2} C_{N_i-1}^{k} \right] \tag{38}
$$

After simple algebraic operations, we achieve

$$
\begin{aligned}
&\Psi(3, 2)^{OPT} = \max_{N_1, N_2} \left\{ \frac{N_1^3 + 5N_1 + 6}{6} \cdot \frac{N_2^3 + sN_2 + 6}{6} \right\} \\
&N_1 > 3 \\
&N_2 > 3 \\
&N_1 + N_2 = 8 > 6
\end{aligned} \tag{39}
$$

We solve the equation using Kuhn-Tucker conditions. Taking 39 into account. we can write the following Lagrange equation

$$
\begin{aligned}
L = \frac{N_1^3 + 5N_1 + 6}{6} \cdot \frac{N_2^3 + 5N_2 + 6}{6} & - \lambda_1 \cdot (N_1 - 4) - \lambda_2 \cdot (N_2 - 4) - \lambda_3 \cdot (N_1 + N_2 - 8) \\
& N_1 - 4 \geq 0 \\
& N_2 - 4 \geq 0 \\
& N_1 + N_2 - 8 = 0
\end{aligned} \tag{40}
$$

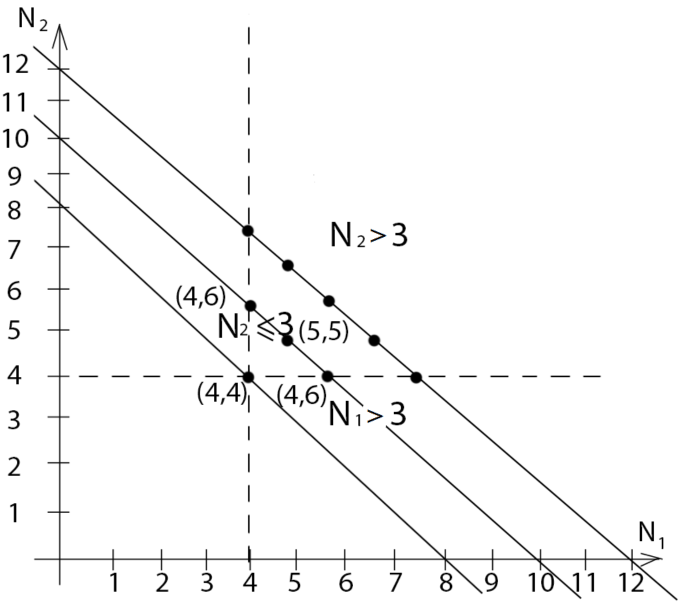

Figure 5: Graphical solution of Kuhn Tucker conditions. Line $N + N_1 + N_2$ is a solving line with one or more solutions. Only one point is max. Figure shows three solution lines for $N_1 + N_2 = 8$, $N_1 + N_2 = 10$, $N_1 + N_2 = 12$

N	$N_1 > 3$	$N_2 > 3$	$\Phi(3, 21)$	*Solution*
8	4	4	225	max
9	5	4	390	max
	4	5	390	max
10	6	4	630	
	5	5	676	max
	4	6	630	
11	4	7	960	
	5	6	1092	max
	6	5	1092	max
	7	4	960	
12	4	8	1395	
	5	7	1664	
	6	6	1774	max
	7	5	1664	
	8	4	1395	

Table 3: Solution for non-linear Kuhn Tucker conditions for total number of neurons, N=8–12

Epochs	10	50	100	500	1000	5000	10000	50000
$\sum \epsilon^2$ for Direct Connection	12.40415	9.10857	8.58351	8.48001	8.38696	8.260625	8.14166	8.0152
$\sum \epsilon^2$ for Cross Forward	2.22719	0.33131	0.12325	0.02912	0.00808	0.00148	0.00076	0.00014

Table 4: Comparison for Direct Connection and Cross Forward Connection with $N_0 = 3$, $N_1 = 1$,$N_W = 2$

Epochs	10	50	100	500	1000	5000	10000	50000
$\sum \epsilon^2$ for Direct Connection	6.91134	0.28018	0.11306	0.01864	0.00542	0.000092	0.000052	0.00009
$\sum \epsilon^2$ for Cross Forward	1.02033	0.12252	0.10.064224	0.01945	0.00441	0.000823	0.000381	0.00007

Table 5: Comparison for Direct Connection and Cross Forward Connection with $N_0 = 3$, $N_1 = 4$,$N_W = 2$

4 Conclusion

For most practical purposes, ANNs with one hidden layer are sufficient. Learning Algorithms for the networks are time consuming and depend on number of layers and number of neurons in each layer. The running time of learning algorithm has dependency, greater than linear, on the number of neurons. Hence, the running time increases faster than the total number of neurons.

Cross Forward connection provides us an opportunity to decrease the number of neurons and thus, the running time of learning algorithm.

We implemented both Direct Connection Neural Networks and Cross Forward Neural Networks with one hidden layer and used them for pattern recognition.

Our implementation required three input neurons and two output neurons. We varied the number of neurons in hidden layer and trained both networks for limited number of epochs and noted the sum of squared errors of each output neurons. The procedure was repeated 20 times and the average sum of square of errors were recorded. Data for two cases are presented in tables 4 and 5.

Tables 4 and 5 demonstrate that for the given number of neurons in the hidden layer, Cross-Forward Connection are optimal. If we closely examine the error term in table 4 for Direct Connection and the same in table 5 for Cross Forward Connection we will notice that they are fairly comparable. It demonstrates that Cross Connection Structure with one neuron in hidden layer is almost as good as Direct Connection with four neurons in hidden layer. Thus, Cross-Forward connection reduce the required number of neurons in ANNs.

Additionally, we presented optimization criteria for Cross Forward Connection and solved two different problems. For linear optimization , where $N_i \leq N_0$, for i=1,2,... W-1, we achieved multiple equivalent ANN structures with the same number of total subspaces $\Psi(N_0, W-1)$. This means that for given total number of neurons ,N, and number of layers W , there are multiple equivalent ANN structures (Table 2). In practice, this ANN structures can be used for tasks with very big dimensionality of input vector X (initial feature space). For nonlinear optimization task, where $N_i > N_0$ for i=1,2,3...... W-1, the target function is nonlinear with linear constraints. There could be one or more optimum solutions. Final solution depends on dimensionality of feature space N_0 and relation between N, N_i and W. In our example, for ANN with $N_0 = 3$, W=3, and N=8,9,10,11,12,. . . .. we achieved one optimum solution for even N_0s and two solutions for odd N_0s (Table 3).

References

[1] Stanisaw Osowski, Sieci Neuronowe do Przetwarzania Informacji. Oficyna Wydawnicza Politechniki Warszawskiej, Warszawa 2006.

[2] S. Osowski, Sieci neuronowe w ujeciu algorytmicznym.WNT, Warszawa 1996.

[3] O.B.Lapunow, On Possibility of Circuit Synthesis of Diverse Elements, Mathematical Institut of B.A. Steklova, 1958.

[4] Toshinori Munakate, Fundationals of the New Artificial Intelligence. Second Edition, Springer 2008.

[5] Colin Fyle, Artificial Neural networks and Information Theory, Departmeeent of Ciomputing and information Systems, The University of Paisley, 2000.

[6] Joarder Kamruzzaman, Rezaul Begg, Artificial Neural Networks in Finance and Manufacturing, Idea Group Publishing, 2006.

[7] A. Mariciak, J. Korbicz, J. Kus, Wstepne przetwarzanie danych, Sieci Nuronowe tom 6, Akademicka Oficyna Wydawnicza EXIT 2000.

[8] A. Marciniak, J. Korbicz, Neuronowe sieci modularne, Sieci Nuronowe tom 6, Akademicka Oficyna Wydawnicza EXIT 2000.

[9] Z. Mikrut, R. Tadeusiewicz, Sieci neuronowe w przetwarzaniu i rozpoznawaniu obrazow, Sieci Nuronowe tom 6, Akademicka Oficyna Wydawnicza EXIT 2000.

[10] L. Rutkowski, Metody i techniki sztucznej inteligencji, Wydawnictwo Naukowe PWN, warszawa 2006.

[11] Juan R. Rabunal, Julian Dorado, Artificial Neural Networks in Real-Life Applications, Idea Group Publishing 2006.