CONVERGENCE ANALYSIS OF CASCADE ERROR PROJECTION-AN EFFICIENT LEARNING ALGORITHM FOR HARDWARE IMPLEMENTATION

Tuan A. Duong
Center for Space Microelectronics Technology
Jet Propulsion Laboratory, California Institute of Technology
Pasadena, CA 91109

Abstract:

In this paper, we present a mathematical foundation, including a convergence analysis, for cascading architecture neural network. Our analysis also shows that the convergence of the cascade architecture neural network is assured because it satisfies Liapunov criteria, in an added hidden unit domain rather than in the time domain. From this analysis, a mathematical foundation for the cascade correlation learning algorithm can be found. Furthermore, it becomes apparent that the cascade correlation scheme is a special case from mathematical analysis in which an efficient hardware learning algorithm called Cascade Error Projection(CEP) is proposed.

The CEP provides efficient learning in hardware and it is faster to train, because part of the weights are deterministically obtained, and the learning of the remaining weights from the inputs to the hidden unit is performed as a single-layer perceptron learning with previously determined weights kept frozen. In addition, one can start out with zero weight values (rather than random finite weight values) when the learning of each layer is commenced. Further, unlike cascade correlation algorithm (where a pool of candidate hidden units is added), only a single hidden unit is added at a time. Therefore, the simplicity in hardware implementation is also achieved.

Finally, 5- to 8-bit parity and chaotic time series prediction problems are investigated; the simulation results demonstrate that 4-bit or more weight quantization is sufficient for learning neural network using CEP. In addition, it is demonstrated that this technique is able to compensate for less bit weight resolution by incorporating additional hidden units. However, generation result may suffer somewhat with lower bit weight quantization.

I-Introduction

Many ill-defined problems in areas such as pattern recognition, classification, vision, and speech recognition require a practical solution. Typically, these problems are too complex to be solved by a linear technique thus non-linear methods, such as neural network methods are used. Usually the practical value of a neural network method is closely related to the paradigm used to train the neural network. Currently, there are several neuromophic learning paradigms reported in literature [1-13] which are widely used. The majority of them are supervised learning techniques. The Error Backpropagation (EBP)[8] learning algorithm is one of the most popular supervised learning technique. In the real world applications, EBP often suffers convergence problems [11]. Recently, a learning technique called "cascade correlation" (CC)[11,14] has shown encouraging results. This method appears to be fast and reliable in learning, but thus for only empirical studies of its convergence properties have been provided. A mathematical foundation for this algorithm is urgently needed so that from this a convergence analysis can be developed.

Such an analysis is herein provided for a learning algorithm, called cascade error projection (CEP), of which cascade correlation is a special case. CEP is a simple learning method using a one-layer perceptron approach followed by a deterministic calculation for another layer. This simple procedure offers a very fast, reliable, and implementable learning algorithm in hardware. The architecture for CEP is given in Figure 1.

Shaded squares and circles indicate frozen weights; squares indicate calculated weights, and circles indicate learned weights. The analysis is based only on the set of weights that is connected to the new hidden unit *(n+1)*. In this case, only the blank squares and circles must be determined in order to decrease the energy level.

In the following sections of this paper an analysis of the structure and a learning technique is presented. Next, a difference energy function ΔE between layers n and

(n+1) is introduced. This function contains two sets of variables: (1) the set of weights between the input (including previously expanded inputs) and the current hidden unit, namely W_{ih}; (2) the set of weights between the current hidden unit and the output unit, namely W_{ho}. These two sets of variables are treated sequentially (not simultaneously). First, the difference energy function is maximized with respect to W_{ho} thus obtaining $\max_{Who}(\Delta E)$. Note that, the $\max_{Who}(\Delta E)$ is also a function of W_{ih}. We will show that there exists a solution set W_{ih}^{*}, obtained from an affine space which guarantees that the network reduces (or at least maintains constant) the present energy level when the new hidden unit is added. Thus, we can conclude that the network converges in the Liapunov's sense as new units are added. From this we propose that the solution which is obtained in a non-linear space by learning techniques such as gradient descent, conjugate gradient, correlation, covariance or Newton's second order may be suitable. The problems that are used to simulate the CEP are 5- to 8-bit parity problems.

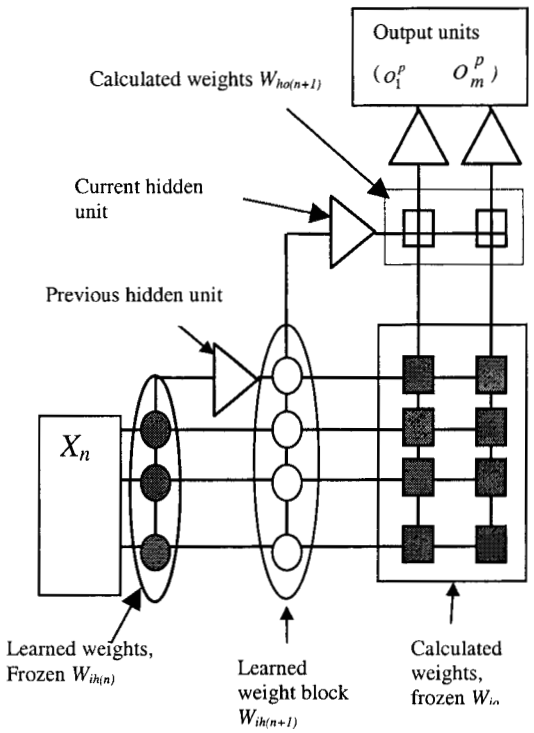

Figure 1. The architecture of cascade error projection includes inputs, hidden units, and output units. The shaded circles or squares indicate the learned or calculated weight set which were computed and frozen. A circle indicates that perceptron learning has been applied to obtain the weight, and a square indicates that the weight set is deterministically calculated.

II STRUCTURE OF CASCADE ERROR PROJECTION:

We start this section with a definition which will help to define the general structure of our neural network.

Definition:

For any k ∈ N, A^k is the set of all affine functions from $\Re^k$ to $\Re$, that is, the set of all functions of the form $A(X) = W^T X + b$ where W and X are vectors in $\Re^k$, and $b \in \Re$ is a scalar.

In this paper, X will correspond to input of the network and W corresponds to the weight set which will vary with the dimension of the required cascade network. We start with the neural network in Figure 1 where we assume that the network contains n hidden units. We also assume that the learning cannot be further improved; that is, the energy level cannot be further reduced. At this point, the new hidden unit *(n+1)* is added to the network and we choose the new weights to further reduce the energy level.

Let Ξ be the input space and $\Xi \subset [-1,1]^N$, Ψ be an output space and $\Psi \subset [-1,1]^m$, and Ω be a hidden output space and $\Omega \subset [-1,1]^q$. Thus, $\Xi x \Omega$ forms the input space of the newly added hidden unit which is $[-1,1]^{N+q}$ where N is the dimension of the input space, q is the dimension of the expanded input space ($N+q$ is the dimension of the total input space to hidden unit $n+1$), and m is the dimension of the output space. Let us define

$$f_h : [-1,1]^{N+q} \times \Re^{N+q} \longrightarrow [-1,1]$$
$$f_o : [-1,1]^{N+q+1} \times \Re^{N+q+1} \longrightarrow \Psi$$

Where $\Re^{N+q}$ is the weight space of $N+q$ dimensional real elements and similarly for $\Re^{N+q+1}$. Finally, the components f_h and f_o are sigmoidal transfer functions which are defined by:

$$f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$$

Other notation we use are defined as follows:

$\varepsilon_o^p = t_o^p - o_o^p(n)$ denotes the error between output element o and training pattern p with target t and actual output $o(n)$ where n indicates that the output has n hidden units in the network.

$f'^p_o(n)$ denotes the output transfer function derivative with respect to input of the output element o and the training pattern p.

$f_h^p(n+1)$ denotes the transfer function of hidden unit n+1 and training pattern p.

X^p denotes the input pattern p and $|X|$ denotes the Euclidean length of vector X.

1. *CONTINUOUS WEIGHT SPACE:*

In continuous weight space, weight component and weight updating can be any real number with unlimited bit weight quantization. In this section, there are no constraints of weight space.

Theorem 1: In cascade architecture, the maximum energy reduction between hidden unit n and $(n+1)$ with respect to w_{ho} is

$$\sum_{p=1}^{P}\sum_{o=1}^{m} \varepsilon_o^p f'^p_o f_h^p(n+1)$$

where the energy function of the network is defined as

$$E=\sum_{p=1}^{P} E^p = \sum_{p=1}^{P}\sum_{o=1}^{m}(t_o^p - o_o^p)^2 = \sum_{p=1}^{P}\sum_{o=1}^{m}(\varepsilon_o^p)^2$$

Proof:

Let t_o^p be the target output of unit o given input pattern p, and let the actual output of unit o be given by:

$$o_o^p = f((X_i^p)^T W_{io} + \sum_{j=1}^{n+1} f_h^p(j) w_{ho}(j))$$

with

$$X_i^p = \begin{bmatrix} 1 \\ x_1^p \\ . \\ . \\ x_N^p \end{bmatrix} \quad ; \; X_h^p(n+1) = \begin{bmatrix} f_h^p(1) \\ . \\ . \\ f_h^p(n+1) \end{bmatrix};$$

X_i^p (dimension $(N+1)$x1) denotes the original input vector of pattern p, and $X_h^p(n+1)$ (dimension $(n+1)$x1) denotes an expanded input vector with $(n+1)$ hidden units,
and let

$$i^p(n+1) = \begin{bmatrix} X_i^p \\ X_h^p(n+1) \end{bmatrix}$$

then

$$f((i^p(j))^T W_{ih}(j)) = f_h^p(j+1)$$

where $f_h^p(j+1)$ denotes the output of hidden unit $j+1$ with the input pattern p.

Let $E(n)$ and $E(n+1)$ be the energy level of the network with n and $n+1$ hidden units, respectively. The desire in learning is to reduce the energy from $E(n)$ to $E(n+1)$ as much as possible (ignoring the overlearning phenomenon). The ideal case would be

$$\max\{E(n) - E(n+1)\} = \max \Delta E$$

Then, we have

$$\Delta E = -\sum_{o=1}^{m} w_{ho}^2 \sum_{p=1}^{P}[f'^p_o f_h^p(n+1)]^2 + \sum_{o=1}^{m} 2w_{ho} \sum_{p=1}^{P}[\varepsilon_o^p f'^p_o f_h^p(n+1)] \quad (1)$$

The sufficient condition for equation (1) to be maximum with respect to w_{ho} is

$$\max_{w_{ho}}(\Delta E) = \sum_{p=1}^{P}\sum_{o=1}^{m}\{\varepsilon_o^p f'^p_o f_h^p(n+1)\} \quad (2)$$

$$\text{where } w_{ho} = \frac{\sum_{p=1}^{P} \varepsilon_o^p f'^p_o f_h^p(n+1)}{\sum_{p=1}^{P}[f'^p_o f_h^p(n+1)]^2}$$

Theorem 2: The projection of the error surface onto the output of a new hidden unit always guarantees that there exists a weight subspace of the learning weight space, which can be obtained from the affine space. These cascading sequential subspaces ensure that the network converges in the Liapunov sense.

Proof:

Let

$$\Gamma = \begin{bmatrix} \frac{1}{m}\sum_{o=1}^{m} f'^{1}_{o}\{t^1_o - o^1_o\} \\ \cdots\cdots\cdots\cdots \\ \cdots\cdots\cdots\cdots \\ \cdots\cdots\cdots\cdots \\ \frac{1}{m}\sum_{o=1}^{m} f'^{P}_{o}\{t^P_o - o^P_o\} \end{bmatrix} = \begin{bmatrix} \varphi_1 \\ \cdots \\ \cdots \\ \cdots \\ \varphi_P \end{bmatrix}$$

Then, $\Gamma \in \Psi$. and

$$F_h(n+1) = \begin{bmatrix} f^1_h(n+1) \\ \cdots\cdots\cdots \\ \cdots\cdots\cdots \\ \cdots\cdots\cdots \\ f^P_h(n+1) \end{bmatrix}$$

We can rewrite equation (2) in a matrix form as follows:

$$\Delta E = m\Gamma^T F_h(n+1)$$

Now let

$$F_h(n+1) = \Gamma \qquad (3)$$

but

$$F_n(n+1) = F(IW_{ih}(n+1)) \qquad (4)$$

with

$$I = \begin{bmatrix} (i^1(n))^T \\ \cdots\cdots\cdots \\ \cdots\cdots\cdots \\ \cdots\cdots\cdots \\ (i^P(n))^T \end{bmatrix}$$

From (3) and (4), we let $W^*_{ih}(n+1)$ be a solution in affine space; then we have

$$IW_{ih}(n+1) = F_h^{-1}(\Gamma) \qquad (5)$$

Finally, the solution is

$$W^*_{ih}(n+1) = I^+ F_h^{-1}(\Gamma)$$

where I^+ is the pseudo-inverse of I.

The existence of $W^*_{ih}(n+1)$ depends on the non-zero column matrix $I^+ F_h^{-1}(\Gamma)$, and the rank of I is at least 1 because of the non-linear combination of all previous dimensions (i=1,n). At the same time, the error surface still exists (if it is zero, then the energy is already zero). Therefore, the existence of $W^*_{ih}(n+1)$ is always guaranteed. As shown, the existence in affine space is demonstrated; however, we are interested in a non-linear space.

For (3) and (5), we apply the mean value theorem:

$$\Gamma - F^*_h(n+1) = F'_h(C)\{F^{-1}(\Gamma) - IW^*_{ih}(n+1)\}$$

with

$$F'_h(C) = \begin{bmatrix} f'_h(c^1)0\ldots0 \\ 0f'_h(c^2)\ldots0 \\ \cdots\cdots\cdots\cdots \\ 0\ldots0f'_h(c^P) \end{bmatrix} \quad \text{and}$$

$$c^p \in (\ (i^p)^T W^*_{ih}(n+1)\ ,\ f^{-1}\{\frac{1}{m}\sum_{o=1}^{m} f'^{p}_{o}\,\varepsilon^p_o\}\ )$$

Note that the dimension of $F'_h(C)$ is PxP.

Now let

$$\Gamma^* = F_h(IW^*_{ih}(n+1)) = \begin{bmatrix} \varphi^*_1 \\ . \\ . \\ . \\ \varphi^*_P \end{bmatrix}$$

In other words, we have

$$\Gamma - \Gamma^* = F'_h(C)\{F^{-1}(\Gamma) - F^{-1}(\Gamma^*)\}$$

We see that

$$\left|\Gamma - \Gamma^*\right|^2 = \left|F'_h(C)\{F_h^{-1}(\Gamma) - F'_h(\Gamma^*)\}\right|^2 \qquad (6)$$

Expanding equation (6), we obtain

$$\left|\Gamma - \Gamma^*\right|^2 = \left|F'_h(C)F_h^{-1}(\Gamma)\right|^2 + \left|F'_h(C)F_h^{-1}(\Gamma^*)\right|^2 - 2\{F'_h(C)F_h^{-1}(\Gamma)\}^T\{F'_h(C)F_h^{-1}(\Gamma^*)\} \qquad (7)$$

Now let

$$S = \left|\Gamma\right|^2 - \left|F'_h(C)F_h^{-1}(\Gamma)\right|^2 - \left|F'_h(C)F_h^{-1}(\Gamma^*)\right|^2 + 2\{F'_h(C)F_h^{-1}(\Gamma)\}^T\{F'_h(C)F_h^{-1}(\Gamma^*)\} \qquad (8)$$

and $S \geq 0$ (9)

where the inequality (9) is proved in Appendix B

From (8) and (9), we see that:

$$-2\Gamma^T\Gamma^* + \left|\Gamma^*\right|^2 \leq 0$$

In other words,

$$\Gamma^T F_h^*(n+1) \geq \frac{\left|F_h^*(n+1)\right|^2}{2}$$

Also we have

$$\Gamma^T F_h^*(n+1) \leq \frac{\left|\Gamma\right|^2 + \left|F_h^*(n+1)\right|^2}{2}$$

Finally, we have

$$\frac{\left|F_h^*(n+1)\right|^2}{2} \leq \Gamma^T F_h^*(n+1) \leq \frac{\left|\Gamma\right|^2 + \left|F_h^*(n+1)\right|^2}{2} \quad (10)$$

We should note that $\frac{\left|F_h^*(n+1)\right|^2}{2} > 0$, because *the rank of I is at least 1* (it is noted that the output of hidden unit n is a non-linear combination of the all the previous outputs of hidden units and the original inputs and this output of hidden unit n ensures the rank of I to be at least 1). However, the inverse of the sigmoidal function is used to obtain $W_{ih}^*(n+1)$, so it is possible to encounter the null space in affine space. Therefore, the precise inequality is

$$\frac{\left|F_h^*(n+1)\right|^2}{2} \geq 0$$

From (10), there exists at least one solution obtained by the pseudo-inverse technique in affine space. This solution also indicates that the lower bound of reduction in energy that can be obtained by the hidden unit (n+1). Therefore, in non-linear space it can be shown there always exists a solution space when the error surface is projected to the new hidden unit for learning and the lower bound energy reduction is $\frac{m\left|F_h^*(n+1)\right|^2}{2}$. To obtain the maximum energy reduction, a straight forward approach is to obtain the closest match between $\left|F_h(n+1)\right|$ and Γ, one can use gradient descent, maximum correlation, covariance, Newton's second order, or conjugate gradient techniques to obtain this.

Finally, $\Delta E(n) \leq 0$,

with $\Delta E(n) = E(n+1) - E(n)$.

In conclusion, we have shown that there exists a weight set $W_{ih}^*(n+1)$ obtained by the pseudo-inverse technique, and this weight set guarantees a reduction of the energy or at worst the same energy with the addition of the hidden unit (n+1). From the network viewpoint, the energy decreases or remains the same when the number of hidden units increases; therefore the network converges (in the Liapunov sense).

2. *DISCRETE WEIGHT SPACE:*

From the hardware implementation viewpoint, the weight space has discrete and finite precision. The conversion from continuous learning space into the discrete space must be done for hardware implementation. As shown in [14,17,19], the weight space plays very important role in learning convergence. In this section, we want to extend the learning theory further so that the convergence network is guaranteed in discrete and finite precision weight (limited weight quantization) space. Typically, the conversion from a real number from continuous space into a finite precision number is done by two techniques: round-off and truncation [15]. The theory below will provide us the effect of CEP learning in finite precision weight space especially the network will be able to converge in a limited weight space when the round-off technique is applied.

Theorem 3: In cascading architecture, the convergence of the network that is achieved in a high weight quantization space can also be obtained in limited weight quantization space (B is weight bit quantization available) such that:
$-\beta 2^{-(B+1)} \leq \delta \leq \beta 2^{-(B+1)}$ *and* $\Theta\{\delta\} = 0$;
The network with the limited weight quantization space converges in mean square sense which is

$$\Theta\{\left|\tilde{F}_h^*(n+1)\right|^2\} \leq 2\Theta\{\Gamma^T \tilde{F}_h^*(n+1)\}$$

Θ is a statistical mean operator.
β is a dynamical coefficient.
$\tilde{F}$ is a transfer function of inner product of input and discrete limited weight vectors.

Proof:

As proved above for theorem 2, the network converges in Liapunov's sense. In this section we want to show further that the CEP learning approach has the capability to learn in the discrete limited quantization space. The requirement for this learning capability is the use of the dynamical step size that can be obtained from the previous energy level. In Figure 1, with a new hidden unit *(n+1)*, the output *o* for pattern input *p* can be expressed as:

$$o_o^p(n+1) = f(net_o^p + w_{ho} f_h^p(n+1))$$

In equation (2), w_{ho} is calculated; hence, it has very little effect on the learning capability, and it is ignored. However, the main focus of this study in the learning capability of the network is the determination of W_{ih}. It can be expressed as follows:

$$W_{ih} = \tilde{W}_{ih} + \delta$$

and

$$\frac{-stepsize(n+1)}{2} \le \delta < \frac{stepsize(n+1)}{2}$$

with $stepsize(n+1) = \beta 2^{-B}$

where $\tilde{W}_{ih}$ is a weight vector in discrete limited weight space, and δ is a noise vector that may come from the round-off technique.

We have

$$f_h^p(n+1) = f_h(i^p(\tilde{W}_{ih}(n+1) + \delta)) \quad (12)$$

If δ is sufficiently small, equation (12) can be written as:

$$f_h^p(n+1) = f_h(i^p \tilde{W}_{ih}) + f'_h(i^p \tilde{W}_{ih}) i^p \delta$$

Let $\upsilon^p = f_h^p(n+1) - \tilde{f}_h^p(n+1)$ be an error between the hidden output with infinite weight resolution and the hidden output with limited weight resolution of hidden unit (n+1). Then,

$$\upsilon^p = f'_h(i^p \tilde{W}_{ih}) i^p \delta$$

From the previous proof (Eqn 10), we obtain

$$\frac{\left|F_h^*(n+1)\right|^2}{2} \le \Gamma^T F_h^*(n+1) \quad (13)$$

Let $f'_h(i^p \tilde{W}_{ih}) = \tilde{f}'^p_h$

$$Y = \begin{bmatrix} \upsilon^1 \\ . \\ . \\ . \\ \upsilon^P \end{bmatrix} = \begin{bmatrix} \tilde{f}'^1_h i^1 \delta \\ ..... \\ ..... \\ ..... \\ \tilde{f}'^P_h i^P \delta \end{bmatrix}$$

From (13), we can rewrite

$$\left|\tilde{F}_h^*(n+1) + \mathrm{Y}\right|^2 \le 2\Gamma^T(\tilde{F}_h^*(n+1) + \mathrm{Y})$$

Expanded, we have

$$\left\|\tilde{F}_h^*(n+1)\right\|^2 + 2Y^T \tilde{F}_h^*(n+1) + \left\|\mathrm{Y}\right\|^2 \le 2\{\Gamma^T \tilde{F}_h^*(n+1) + \Gamma^T \mathrm{Y}\} \quad (14)$$

Let us introduce the *statistical mean operator* Θ. In the process of obtaining the weight set $\tilde{W}_{ih}$, the learning is repeatedly applied. This technique can be viewed as a statistical mean process, then (14) becomes

$$\Theta\{\left\|\tilde{F}_h^*(n+1)\right\|^2\} + 2\Theta\{Y^T \tilde{F}_h^*(n+1)\} + \Theta\{\left\|\mathrm{Y}\right\|^2 \le 2\Theta\{\Gamma^T \tilde{F}_h^*(n+1) + \Gamma^T \mathrm{Y}\} \quad (15)$$

But, Y is independent to Γ and $F_h^*(n+1)$.

In the round-off technique, Y can be considered as white noise and $\Theta(Y) = 0$. Then inequality (15) becomes

$$\Theta\{\left\|\tilde{F}_h^*(n+1)\right\|^2 \le 2\Theta\{\Gamma^T \tilde{F}_h^*(n+1)\} \quad (16)$$

The result of inequality (16) guarantees the learning capability of the network if $F_h^*(n+1)$ is not zero, but it does not ensure the same achievement of energy level as does the infinite weight resolution. As analysis, inequality (16) only guarantees that the learning in limited weight quantization can be done, given the assumption $\delta << W_{ih}$. The remaining question is how small δ can be compared to W_{ih}, or how can we obtain information about δ through known information? We can observe that the smaller δ is, the closer the reduction between energies in limited weight quantization and infinite weight resolution is.

Stepsize

The conversion between the continuous and the limited weight quantization weight space requires the scaling factor known as stepsize. With the fixed weight quantization levels (2^B levels, and B is bit quantization), this stepsize is proportional to the energy reduction level (ignoring the non linear factor which is come from non linear transfer function). The summarization can be described as follows:

$\tilde{W}_{ih}(n+1) \propto stepsize(n+1)$

, *and*

$\tilde{W}_{ih}(n+1) \propto F_h(n+1)$ (Roughly estimated and ignored the non linear factor)

$F_h(n+1) \propto \Delta E(n) \propto E(n)$

then, $stepsize(n+1) \propto E(n)$

Therefore, $stepsize(n+1) = \alpha E(n)$ with α *constant* (see Table I below)

As it is shown, the weight set $W_{ih}(n+1)$ can be obtained directly from affine space by using

the pseudo-inverse technique, which has been thoroughly studied [16]. However, in our present approach, we are interested in a non-linear solution space in which the solution weight set can be obtained directly from a learning technique using analog/digital hardware. This learning approach will offer a better solution from both the theoretical and implementable point of view. First, the solution which is obtained in non-linear space has compactness of the network and smoothness of the transformation because the data distribution is always in the non-linear domain. Second, it is hard to solve a singular-valued decomposition problem in a linear hardware network, even though the solution is deterministically defined, but the cost of the complicated hardware required by the network may exceed the available resources.

III SIMULATION

a) The Cascade Error Projection Learning Algorithm Procedure

1. Start with the network which has input and output neurons. With the given input and output patterns and hyperbolic transfer function, one can determine the set of weights between input and output by using pseudo-inverse or perceptron learning. The weight set W_{io} is thus obtained and frozen.

2. Add a new hidden unit with a zero weight set for each unit. In each loop (contains an epoch) an input-output pattern is picked up randomly in the epoch (no pattern repeated until every pattern in the epoch is picked). Use the perceptron learning technique of equation (a) to train $W_{ih}(n+1)$ for 100 epoch iterations.

3. Stop the perceptron training. Calculate the weights $W_{ho}(n+1)$ between the current hidden unit and the output units from equation (2).

Cross-validate the network. If the criteria is satisfied, then stop training, and test the network. Otherwise, go to step 2 above until the number of hidden units is more than 20; then give up and quit!

b) *Conversion technique*

The updating weight Δw is converted into the available weight quantization which is Δw^*. The conversion can be summarized as follows:

$$stepsize(n) = \alpha E(n-1) \quad \text{with}$$

α constant

Round-off technique:

$$\Delta w^*_{jh}(n) = \begin{cases} stepsize(n) * \text{int}(\dfrac{\Delta w_{jh}}{stepsize(n)} + 0.5) \\ stepsize(n) * \text{int}(\dfrac{\Delta w_{jh}(n)}{stepsize(n)} - 0.5) \\ 0 \end{cases}$$

$$if \quad (\frac{w_{jh}(n)}{stepsize(n)} + \text{int}(\frac{\Delta w_{jh}(n)}{stepsize(n)} + 0.5)) \le 2^B \; and \quad \Delta w_{jh}(n) > 0$$

$$if \quad (\frac{w_{jh}(n)}{stepsize(n)} + \text{int}(\frac{\Delta w_{jh}(n)}{stepsize(n)} - 0.5)) \le -2^B \; and \quad \Delta w_{jh}(n) < 0$$

Otherwise

c) Parameters

The learning rate η is used and is set to decrease linearly as: $\eta_{new} = \eta_{old} - .01 * \eta_0$, where η_0 = initial learning rate.

For our simulation, the parameter values of Table I are

Table I: *Values of initial learning rate and α used in simulation for different parity problems and bit-resolution of synpases.*

	5-bit parity	6-bit parity	7-bit parity	8-bit parity
64-bit W	η_0=1.0 α=N/A	η_0=1.0 α=N/A	η_0=0.4 α=N/A	η_0=0.4 α=N/A
3-bit W	η_0=1.0; α=.0025	η_0=1.0; α=.0167	η_0=1.0; α=.0087	η_0=1.0; α=.0041
4-bit Ws	η_0=1.0; α=.0016	η_0=1.0; α=.0109	η_0=1.0; α=.0082	η_0=1.0; α=.0041
5-bit W	η_0=1.0; α=.0016	η_0=1.0; α=.0108	η_0=1.0; α=.0081	η_0=1.0; α=.0042
6-bit Ws	η_0=1.0; α=.0016	η_0=1.0; α=.0108	η_0=1.0; α=.0081	η_0=1.0; α=.0042

d) Problems:

Parities problems:

The problems that are solved in this paper are 5- to 8-bit parity problems (1) with no limited weight quantization (The weight resolution is the same as the floating point machine which is

about 32-bit for floating point or 64-bit for double precision); and, (2) with limited weight quantization from 3-to 6-bits. The details of simulation can be found else where[17-18]

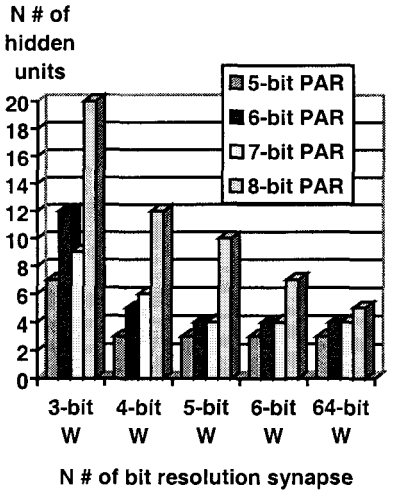

Figure 2: The chart shows CEP learning capability for 5- to 8-bit parity problems using round-off technique. x axis represents limited weight quantization (3-6 and 64-bit) and y axis shows the resulting number of hidden units (limited to 20). Each hidden unit has 100 epoch iterations. As shown, the lager number of hidden units compensate for the lower weight resolution.

Figure 2 refers to simulation results with round-off technique. Even with 3-bit weight resolution the network is able to learn 5- to 7-bit parity problems with no error within the 20 hidden units limit. For weight quantization of 4-bit or more, the network reliably demonstrates the capability of learning from 5- to 8-bit parity problems.

Chaotic Time Series Problem:

The data in this problem represents chaos and is never repeated. However, this data between past, present, and future are correlated in high order. To validate the capability of CEP as shown in theory, we use CEP learning technique under constraints of limited weight quantization (4-, 6-, and 64-bit weight resolution) to capture the high order correlation of this problem.

In this experiment, we use x_i, x_{i+1}, x_{i+2}, x_{i+3} and the target is x_{i+4} . The number of training data is 351 and test data is 651 and no cross validating data is applied in this phase.

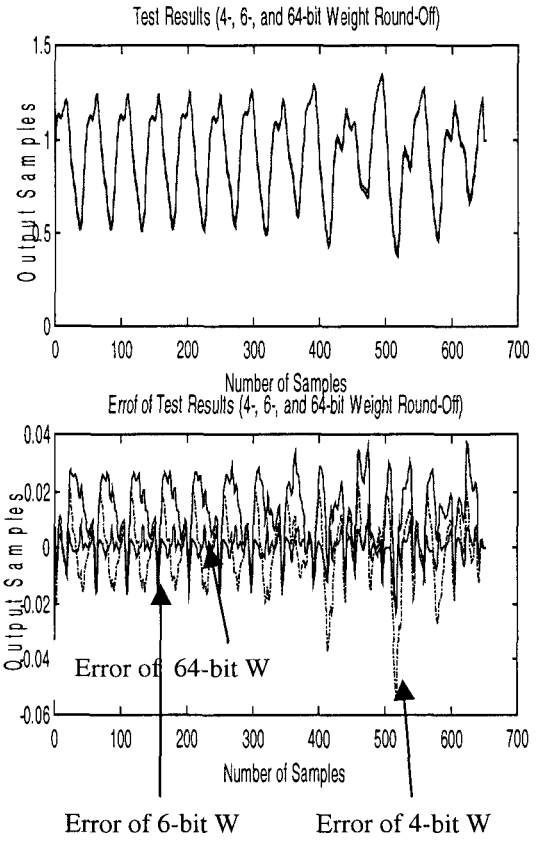

Figure 3: Simulation Results of CEP for chaotic time series prediction problem. Top trace contains four curves: ideal data, 64-bit, 6-bit and 4-bit prediction results. Bottom trace contains : errors between ideal data and 64-bit, 6-bit, and 4-bit generalization data.

The results in Figure 3 show that the error between ideal data and prediction with 64-bit weight learning network is within +/-0.01 and is similar to white noise, whereas, 6-bit error is more harmonic than 4-bit error prediction. These results can be interpreted to infer that the more bit weight quantization is available for learning the better and smoother the transform would be. In addition, the better and smoother transformation will help network to interpolate for predictions.

IV. Conclusions

In this paper, we have shown that CEP is feasible for both a software- and a hardware-based learning algorithm. From this analysis, the way CC works can be understood in depth. Moreover, the theoretical analysis provides us with the general framework of the learning architecture, and the particular learning algorithm can be independently studied for its suitability in a given application associated with some constraints for each problem. (For example, in the hardware approach, CEP is most advantageous, and for software, Covariant or Newton's second order method is more advantageous).

Acknowledgments:

The research described herein was performed by the Center for Space Microelectronics Technology, Jet Propulsion Laboratory, California Institute of Technology and was jointly sponsored by the Ballistic Missile Defense Organization (BMDO), and the National Aeronautics and Space Administration (NASA). The authors would like to thank Prof. A. Stubberud, Drs T. Daud, and A. Thakoor for useful discussions.

References:

1. Albus, J. S. 1971, "A theory of cerebellar function," *Mathematical Biosciences* vol. 10, pp. 25-61.
2. Cohen, M. and Grossberg, S. 1983, "Absolute stability of global pattern formation and parallel memmory stage by competitive neural networks," *IEEE Trans. Systems, Man, Cybernetics*, vol. SMC-13, pp. 815-826.
3. Hopfield, J. J. 1982, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Natl. Acad. Sci. USA*, vol. 79, pp. 2554-2558.
4. Jackson. I. R., 1988, "Convergence properties of radial basis functions", Constructive Approximation, vol. 4, 243-264.
5. Kohonen, T. 1989, Self-Organization and Associative Memory, *Springer-Verlag*, Berlin Heidelberg.
6. Kosko, B. 1988, "Bidirectional associative memories," *IEEE Trans. Systems, Man and Cybernetics*, vol. 18, N# 1, pp. 49-60.
7. Rosenblatt, F. 1958, "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychology Review*, vol. 65, pp. 386-408.
8. Rumelhart, D. E., and McClelland, J. L. 1986, *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Vol. 1: Foundation*. MIT Press, Cambridge, MA.
9. Widrow, B. 1962, "Generalization and information storage in networks of ADALINE neurons," Ed: G.T. Yovitt, "Self-Organizing Systems," *Spartan Books*, Washington DC.
10. Duong, T.A et al., 1996, "Learning in neural networks: VLSI implementation strategies," In: Fuzzy logic and Neural Network Handbook, Ed: C.H. Chen, McGraw-Hill.
11. Fahlman, S. E., Lebiere, C. 1990, "The Cascade Correlation learning architecture," in *Advances in Neural Information Processing Systems II*, Ed: D. Touretzky, Morgan Kaufmann, San Mateo, CA, pp. 524-532.
12. Fukushima, K., Miyake, S. 1982, "Neocognitron: A new algorithm for pattern recognition telerant of deformations and shifts in position," *Pattern Recognition*, 15 (6), pp. 455-469.
13. Hinton, G. F., Sejnowski, T. J., and Ackley, D.H. 1984, "Boltzmann machines: Constrain satisfaction networks that learn," *CMU Technical Report* # CMU-CS-84-119, Canergie Mellon University, Pittsburgh PA.
14. Hoehfeld, M. and Fahlman, S. 1992, "Learning with limited numerical precision using the cascade-correlation algorithm," *IEEE Trans. Neural Networks*, vol.3, No. 4, pp 602-611.
15. Oppenheim, A.V. and Schafer, R.W., *Discrete-Time Signal Processing*, Prentice Hall Signal Processing Series, 1989.
16. Haykin, S. 1991 *Adaptive Filter Theory,* Prentice-Hall, Inc. 2nd Ed
17. Duong, T. A. 1995, "Cascade Error Projection-An efficient hardware learning algorithm," Proceeding Int'l IEEE/ICNN in Perth, Western Australia, vol. 1, pp. 175-178.
18. Duong, T. A. et al., 1996, "Cascade Error Projection-A New Learning Algorithm," Proceeding Int'l IEEE/ICNN in Washington D.C., vol. 1, pp. 229-234.
19. P. W. Hollis, J.S. Harper, and J.J. Paulos, "The effects of Precision Constraints in a Backpropagation learning Network," *Neural Computation*, vol. 2, pp. 363-373, 1990.

Appendix A:

The energy function of the network can be defined as

$$E = \sum_{p=1}^{P} E^p = \sum_{p=1}^{P}\sum_{o=1}^{m}(t_o^p - o_o^p)^2$$

Assume that the network currently has n hidden units, and the energy no longer improves with any search techniques (gradient descent search, or exhausted search, etc.). The new hidden unit is now added to the network. The expected result is

$E(n+1) < E(n)$

This is equivalent to

$$\sum_{p=1}^{P}\sum_{o=1}^{m}\{t_o^p - f(net_o^p + w_{ho}f_h^p(n+1))\}^2 \le \sum_{p=1}^{P}\sum_{o=1}^{m}\{t_o^p - f(net_o^p)\}^2$$

with $o_o^p = f(net_o^p)$

Expanding and rearranging, we have

$$\sum_{p=1}^{P}\sum_{o=1}^{m}\{[f(net_o^p + w_{ho}f_h^p(n+1)) - f(net_o^p)][f(net_o^p + w_{ho}f_h^p(n+1)) + f(net_o^p) - 2t_o^p]\} \le 0 \quad \text{(i)}$$

Assume that $w_{ho}f_h^p(n+1)$ is small so that

$$f\{net_o^p + w_{ho}f_h^p(n+1)\} \approx f(net_o^p) + f'(net_o^p)w_{ho}f_h^p(n+1) \quad \text{(ii)}$$

From (i) and (ii), it can be shown that

$$\sum_{p=1}^{P}\sum_{o=1}^{m}\{w_{ho}f'^p_o f_h^p(n+1) [w_{ho}f'^p_o f_h^p(n+1) - 2(t_o^p - o_o^p)]\} \le 0$$

with $f'(net_o^p) = f'^p_o$

or

$$\sum_{o=1}^{m}\{w_{ho}^2\sum_{p=1}^{P}[f'^p_o f_h^p(n+1)]^2 - 2w_{ho}\sum_{p=1}^{P}[f'^p_o f_h^p(n+1)\varepsilon_o^p]\} \le 0$$

Appendix B:

From equation (8), it is rewritten:

$$S = \sum_{i=1}^{P}[\varphi_i^2 - f'^2_h(c^i)\{f_h^{-1}(\varphi_i) - f_h^{-1}(\varphi_i^*)\}^2]$$

or

$$S = \sum_{i=1}^{P}[\{f'_h(d^i)f_h^{-1}(\varphi_i)\}^2 - f'^2_h(c^i)\{f_h^{-1}(\varphi_i) - f_h^{-1}(\varphi_i^*)\}^2]$$

In this proof, the output of the network belongs to [-1,1]. However, it is easier to conduct this proof through Taylor's expansion, the desired outputs of the network are scaled down further by some scaling factor. By scaling the desired outputs, the quality of the network remains the same as before. By doing in so, the proof can be achieved tremendously simple.

Let $\varphi_{max} = max\{\varphi_i\}$ $i=\{1...P\}$

and $\rho_{max} = f^{-1}(\varphi_{max})$

$$\Gamma = \begin{bmatrix} f(\frac{f^{-1}(\varphi_1)}{\rho_{max}}) \\ \cdots\cdots\cdots \\ \cdots\cdots\cdots \\ \cdots\cdots\cdots \\ f(\frac{f^{-1}(\varphi_P)}{\rho_{max}}) \end{bmatrix} = \begin{bmatrix} \phi_1 \\ . \\ . \\ . \\ \phi_P \end{bmatrix}, \quad \text{and}$$

$$\Gamma^* = F_h(IW_{ih}^*(n+1)) = \begin{bmatrix} \phi_1^* \\ . \\ . \\ . \\ \phi_P^* \end{bmatrix}$$

with

$$\phi_i = \frac{e^{\gamma_i} - e^{-\gamma_i}}{e^{\gamma_i} + e^{-\gamma_i}}$$

ϕ_i and ϕ_i^* is expanding around zero and can be obtained as follows:

$$\phi_i = -\gamma_i + \frac{1}{3}\gamma_i^3 + \xi$$

$$\phi_i^* = -\gamma_i^* + \frac{1}{3}(\gamma_i^*)^3 + \xi^*$$

with $\xi, \xi^* \approx 0$

γ_i^* is a component i^{th} of $f_h^{-1}(\phi_i^*)$ which is a pseudo-inverse solution of element $f_h^{-1}(\phi_i)$

From equation (8), it can be reduced

$$S = |\Gamma|^2 - |\Gamma - \Gamma^*|^2 = \sum_{i=1}^{P}\phi_i^2 - (\phi_i - \phi_i^*)^2$$

Since $(\gamma_i)^3$ and $(\gamma_i^*)^3$ are very small to compare with γ_i and γ_i^* respectively, and $F^{-1}(\Gamma^*)$ and $\{F^{-1}(\Gamma) - F^{-1}(\Gamma^*)\}$ are orthogonal vectors [15]. S can be simplified as below

$$S = \sum_{i=1}^{P} \gamma_i^2 - (\gamma_i - \gamma_i^*)^2 = \sum_{i=1}^{P} (\gamma_i^*)^2$$

Therefore, $S \geq 0$