To appear in *The Proceedings of the IEEE World Congress on Computational Intelligence.* June 26 to July 2, 1994.

Back-Propagation Without Weight Transport

John F. Kolen and Jordan B. Pollack
Laboratory for AI Research
Department of Computer and Information Sciences
The Ohio State University
Columbus, OH 43210
kolen-j@cis.ohio-state.edu and pollack@cis.ohio-state.edu

Abstract

In back-propagation (Rumelhart et al, 1985) connection weights are used to both compute node activations and error gradients for hidden units. Grossberg (1987) has argued that the dual use of the same synaptic connections ("weight transport") constitutes a bidirectional flow of information through synapses, which is biologically implausable. In this paper we formally and empirically demonstrate the feasibility of an architecture equivalent to back-propagation, but without the assumption of weight transport. Through coordinated training with weight decay, a reciprocal layer of weights evolves into a copy of the forward connections and acts as the conduit for backward flowing corrective information. Examination of the networks trained with dual weights suggests that functional synchronization, and not weight synchronization, is crucial to the operation of back-propagation methods.

Introduction

Back-propagation (Rumelhart et al, 1985) is a popular gradient descent method for tuning the behavior of connectionist networks. When interpreted as neurons and synapses, the nodes and links of back-propagation appear biologically implausable: private synaptic weights must be communicated to another part of the system. Grossberg (1987, p. 50) argues that "such a physical transport of weights has no plausible physical interpretation" within the known biology of neural systems. By modulating error signal distribution to previous layers with *backward weights* and synchronizing forward and backward weights with weight decay, our learning system avoids the biological implausibility of weight transport. We have tested this system in a variety of training situations including training deep (number of hidden layers greater than one) networks to learn exclusive-or, the 8-3-8 decoder problem, the Nettalk dictionary training set, the two spiral problem, a recursive autoassociative memory (Pollack, 1990), and training a sequential cascaded network to learn the seven Tomita languages (Pollack, 1992). Finally, we examine several networks generated by the dual weight learning system and discover that the weights modulating the backward flow of error-correcting signals do not necessarily need to be equal with the forward weights. What is important is that these weights provide the same functionality of the forward weights for the activations which pass through the network, and not all possible activation patterns.

Forward and Backward Weights

Back-propagation requires a forward pass through the network to generate the network's output response followed by a backward pass to calculate the weight gradient for error minimization. As illustrated in Figure 1, the weight values used during the forward pass must be communicated to the error distribution mechanism and such a mechanism is infeasible under a neural interpretation of weights as synapses. One possible solution to the weight transport problem provides both processes with copies of the weights and involves only the transmission of weight changes to them. Both (Parker, 1985) and (Zipser and Rumelhart, 1990) have considered networks which maintain two weight sets: a $w_{i,j}$ for the forward pass and a $w_{i,j}'$ for the backward pass. These pairs of corresponding weights, however, were started and kept equivalent at all times, *i. e.* $w_{i,j} = w_{i,j}'$. Unfortunately, this merely pushes the weight transport problem backward in time to network genesis, from an ontogenetic correctness issue to a morphogenetic one! What developmental mechanism guarantees equality between two corresponding weights?

In this paper, we first show that it is possible to relax the equality restriction and consider only the asymtopic case, *i.e.* for all i, j, $\lim_{t \to \infty} |w_{i,j}(t) - w_{i,j}'(t)| = 0$. We show that besides its other uses (Hinton & Sejnowski, 1986; Hinton, 1987; Krogh and Hertz, 1992; MacKay, 1992; Moody, 1992), weight decay can synchronize each weight pair, as outlined in Figure 2. Weight decay is defined simply as $w_{i,j}(t+1) = \gamma w_{i,j}(t) + \Delta w_{i,j}(t)$, where γ is the decay constant. In physical systems, such as the brain or VLSI chips, maintaining stable analog values requires elaborate mechanisms. By adopting weight decay, designers may bypass these complex feats of engineering and exploit the natural dynamics of their medium.

The following argument demonstrates that the difference between corresponding weights in the new system will

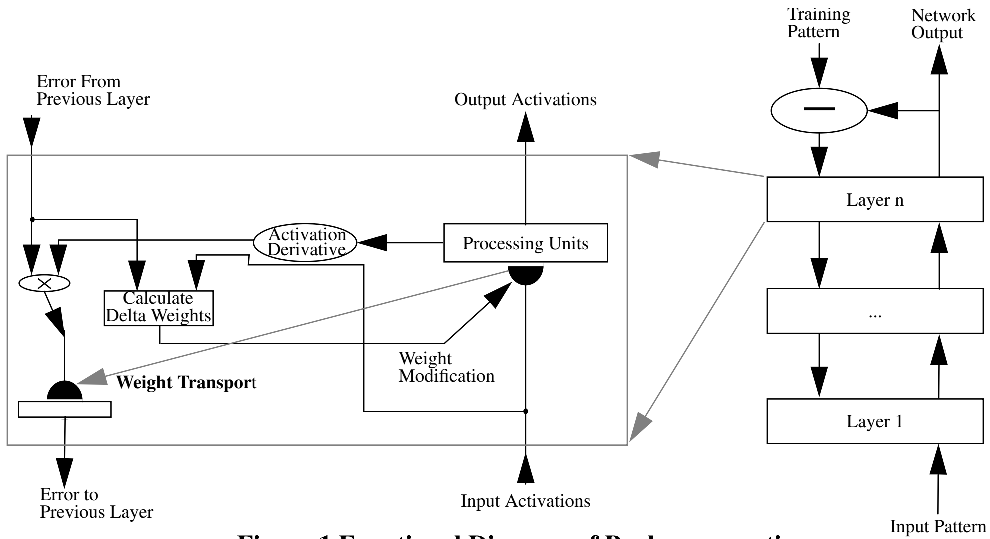

Figure 1 Functional Diagram of Back-propagation

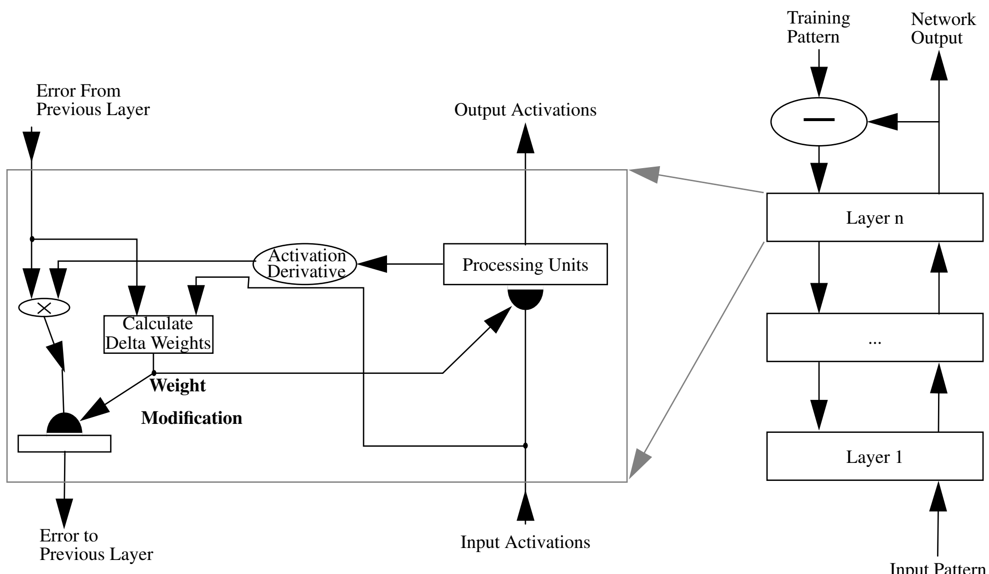

Figure 2 Modified Dual Weight Architecture

evolve to zero. Let $z(t) \in \Re$ be the weight change at some time t, $x(0)$ and $y(0) \in \Re$ be the initial values for the two corresponding weights, and $0 < \gamma < 1$. At time $t+1$, $x(t+1) = \gamma x(t) + z(t)$ and $y(t+1) = \gamma y(t) + z(t)$. Thus for all $t > 0$,

$$0 \le |x(t+1) - y(t+1)| = |\gamma x(t) + z(t) - (\gamma y(t) + z(t))| = |\gamma x(t) - \gamma y(t)| = \gamma |x(t) - y(t)| = \gamma^{t+1} |x(0) - y(0)|$$

Since $\lim_{t \to \infty} \gamma^{t+1} = 0$ when $0 < \gamma < 1$ and $|x(0) - y(0)|$ is finite, we conclude $\lim_{t \to \infty} |x(t) - y(t)| = 0$. The effects of the initial conditions wash out over time, replaced by the most recent modification signals. While we can reduce the effects of different initial forward and backward weights on learning, the total elimination of initial condition effects from back-propagation resulting from multiple possible solutions is impossible (Kolen & Pollack, 1990).

We have empirically verified this architecture through a series of experiments. Figure 3 shows the effects of training a 2 input, 2 hidden unit, 1 output network on the exclusive-or problem[1]. The initial forward and backward weights for this network were selected uniformly from the range $(-0.5, 0.5)$. Error gradients and weight deltas were calculated with on set of weights for the forward pass and another set for the backward passes using a learning rate η of 1.0 and momentum rate α of 0.9. Weight changes were applied at the end of each training epoch to both sets of weights and they decayed at a rate of 0.999 per epoch. The first graph plots the sum of error squared over the 130 presentations of the

1. Error tolerance was 0.5, i.e. performance was measured by a decision threshold at 0.5.

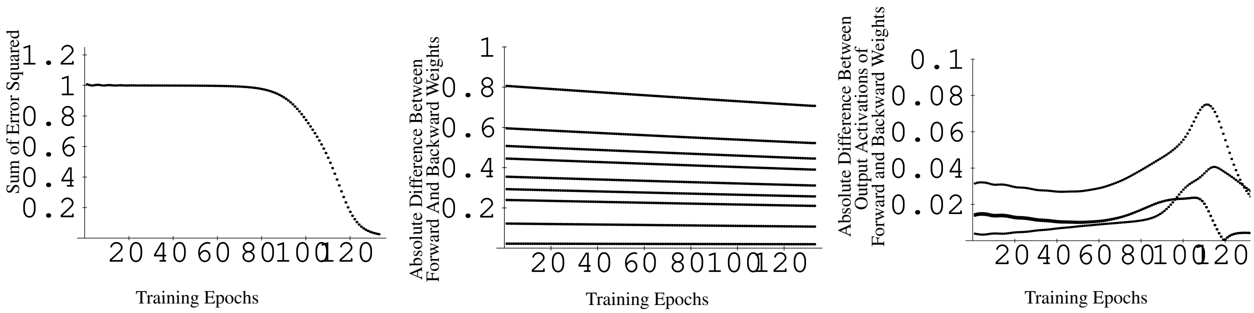

Figure 3 Sample Exclusive Or Results.

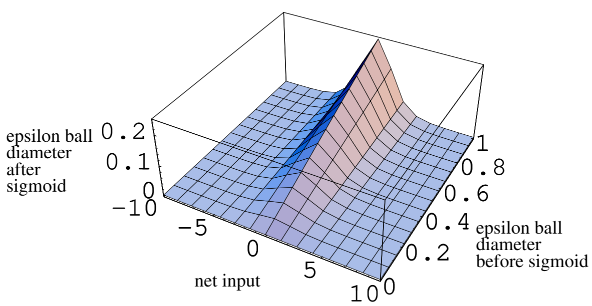

Figure 4 The effective difference between neighboring input levels.

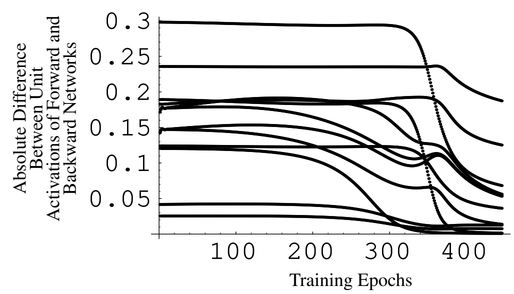

Figure 5 Dual Weight Learning Performace Without Decay

training data. The second graph reports upon the absolute difference between corresponding forward and backward weights in the network during the 130 epochs of training. Note that the weight differences monotonically decrease as predicted above.

Functional Synchronization

We have shown that enforced equality forward and backward weights is unnecessary for the proper operation of the back-propagation learning system and that the mechanisms of weight modification and decay can externally synchronize the two weight systems. In this section, we demonstrate that weight synchronization is not necessary either. What is important is the functional synchronization of the forward and backward weight systems.

Curiously, in almost all cases the forward and backward weights of each network never totally synchronize; the differences between weights never reach zero. By plotting the differences between each corresponding pair of weights, we see that the network learned exclusive-or without the equality of the two weight sets (Figure 3.) The right most graph of Figure 3 supports one explanation: the difference between squashed versions (i.e. passed through the sigmoid) of the weights better gauges the weight synchronization than mere distance. This relationship is made explicit in the plot of Figure 7 which plots the input to the sigmoid squashing function and the diameter of the epsilon ball centered at that level of input versus the resulting absolute activation difference between two points in that epsilon ball. A difference of 0.25 has less effect between 6.25 and 6.00 as opposed to -0.125 and 0.125. The former pair maps to 0.998073 and 0.997527 and the later pair to 0.468791 and 0.531209.

The dual weight back-propagation networks acquired associative mappings despite large (> 0.25) differences between forward and backward weights, suggesting that learning without weight decay may be possible. We ran the exclusive-or example again, with this time with no weight decay, and the results summarized in Figure 5 clearly demonstrated that weight synchronization is unnecessary for learning in a dual weight system. The first plot in Figure 5 reports the standard sum of squared error measurement of network performance. The second plot consists of the differences between unit activations (net input after sigmoid) for units using forward weights verses backward weights. This suggested that the *functional difference* between weight systems is minimized independent of weight synchronization. Let a_j be the activation of unit j in the forward pass network. Now replace each weight in the forward pass network with its counterpart from the backward weights, calling the resulting activations a_j'. For each input pattern, our goal is $a_j = a_j'$, or rather $\lim_{t \to \infty} |a_j(t) - a_j'(t)| = 0$.

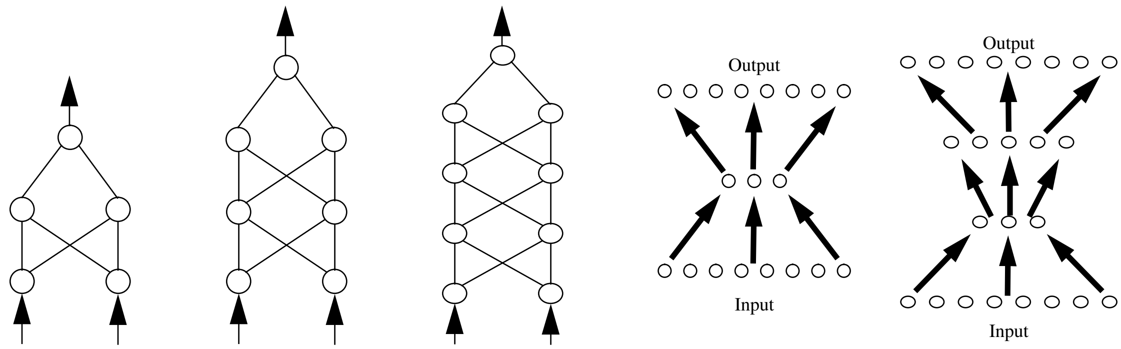

Figure 6 Three architectures for the exclusive or problem and two architectures for the 8-3-8 encoder problem.

Let $w_k(t)$ be the weight from unit k to unit j at time t and $x_k(t)$ be the delta weight calculated at time t for w_k. Assume for all k, $w_k(0) \neq w_k'(0)$ and both $w_k(0)$ and $w_k'(0)$ are selected at random from uniformly distributed interval $(-\rho, \rho)$. Let g be the sigmoidal squashing function. For time t,

$$w_k(t) = w_k(0) + \sum_{i=1}^{t} x_k(i).$$

Thus

$$a_j(t) = g(\sum_{k=1}^{n} w_k(t) a_k(t)) = g(\sum_{k=1}^{n} a_k(t) \left(w_k(0) + \sum_{i=1}^{t} x_k(i) \right)) = g(\sum_{k=1}^{n} a_k(t) w_k(0) + \sum_{k=1}^{n} \sum_{i=1}^{t} x_k(i) a_k(t))$$

Which implies

$$a_j'(t) = g(\sum_{k=1}^{n} a_k(t) w_k'(0) + \sum_{k=1}^{n} \sum_{i=1}^{t} x_k(i) a_k(t))$$

Let $0 < \varepsilon$. If the condition $\left| \sum_{k=1}^{n} w_k(0) \right| < \varepsilon$ holds when the weights are initialized and since $0 < a_j(t) < 1$ is true for all j and t, we can conclude that $|a_j(t) - a_j'(t)| < \varepsilon$ since $|g(x + \Delta x) - g(x)| < \Delta x$ when g is the standard sigmoid function.

Enough "random" error corrections will entrain the functionality of the two weight systems with respect to the learning task. The weight matrices need only agree on activation patterns calculated from input patterns within the training set, other input patterns have no influence and need not be considered.

More Empirical Results

Similar results have been recorded on a number of other tasks. To date, these tasks include training deep (number of hidden layers greater than one) networks to learn exclusive-or, the 8-3-8 decoder problem, the Nettalk dictionary training set, the two spiral problem, a recursive autoassociative memory (Pollack, 1990), and training a sequential cascaded network to learn the seven Tomita languages (Pollack, 1992). Extensive simulations were carried out on different initial weight values because small differences in weight space can cause dramatic differences in learning times (Kolen & Pollack, 1991). For each training set and network architecture, the a full experimental design was used to test the effects of dual weights, weight decay, and initial weight range. Weight decay consisted of two factors, no decay and a decay of 0.999. Inital weight range was assigned from seven values, 0.1, 0.3, 0.5, 0.7, 1.0, 2.0, and 3.0. This parameter specified the maximum absolute value of an initial weight when selected from a uniform distribution.

The first task we will examine is the problem of learning exclusive or. While some may question this task's utility in evaluating learning mechanisms, we believe that problem lies in the combination of learning mechanism and network architecture rather than the input-output mapping itself. One could construct a network biased in such a manner to prevent the learning of a very simple mapping, or the immediate acquistion of an apparently difficult task. Rather than test the dual weight mechanism on a simple network with a single hidden layer of nodes, we tested back propagation's ability to load the exclusive or mapping from examples into three different networks. Figure 6 shows three feedforward networks with one, two, and three hidden layers used in this experiment. For each cell in the design, one hundred different initial networks were trained with each combination of factors. The within subject design was maintained, except between network architectures. Each trial was run for up to ten thousand epochs. Training was terminated if the network

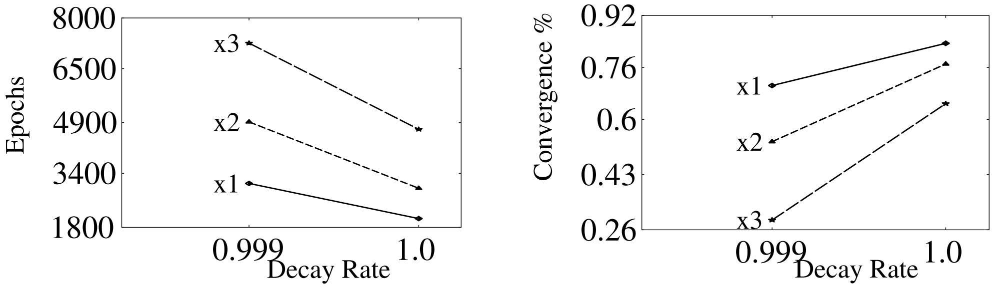

Figure 7 Interaction between depth of network and weight decay in the exclusive or learning task.

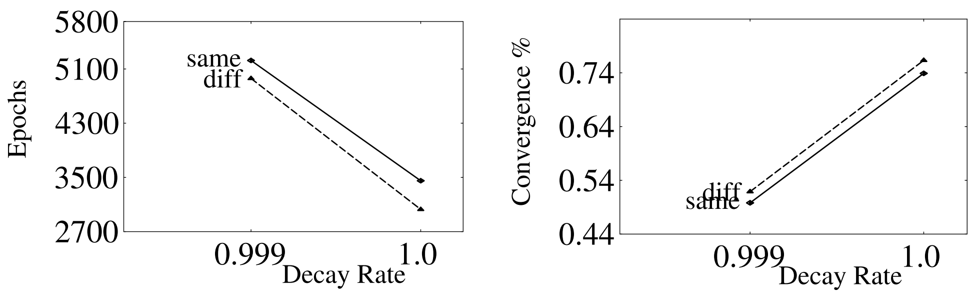

Figure 8 The lack of interaction between weight implementation and decay in learning exclusive or.

could separate the training set (tolerance of 0.5) or if time ran out, in which case the condition was recorded as 10001 epochs. The data was collected and analyzed from two perspectives. First, an ANOVA was performed on the number of epochs to convergence. The data was then processed to convert converged trials to 1's and nonconverged trials to 0's. The resulting data was then subjected to another ANOVA to examine the effects and interactions of the factors on the probabilities of convergence.

We were able to draw several conclusions from the analysis. First, there was a significant difference in the number of epochs to convergence between dual weight and same weight systems. The dual weight systems had a mean convergence time of 3994 epochs, compared to the 4338 epochs for network which used the same weights for forward and backward propagation of signals ($p < 0.001$). The difference, however, was not significant ($p = 0.071$) for the percentage of t-convergent networks: dual weights, 0.6412; same weights, 0.6186. The same relationship was observed across the different architectures. This suggests that a dual weight implementation of back-propagation does not effect the overall dynamics of the learning system, but provides initial noise which facilitates learning.

We initially believed that weight decay may accelerate the learning process in a dual weight implementation of back-propagation since weight decay would eventually erase any initial difference between the two weight sets. The data collected in this experiment does not support this hypothesis, however. A significant difference ($p < 0.001$) was seen between decay ($\overline{\text{epochs}}$=5093, $\overline{\text{conv}}$=0.5086) and no decay ($\overline{\text{epochs}}$=3240, $\overline{\text{conv}}$=0.7512). This effect was strengthened by the depth of the network, as illustrated in Figure 7; networks with more hidden layers showed a significant ($p < 0.001$) decrease in performance while employing weight decay. Yet there was no interaction between the weight implementation and decay, and Figure 8 clearly shows how weight decay hinders both dual and same weight systems while learning exclusive or under a variety of architectures.

The same experiment was performed with the one-in-eight encoder. Two architectures were employed (Figure 6): 8-3-8 network and 8-3-5-8 network. In nearly every trial, the network under consideration could learn the encoder task within 10000 epochs. An exception occured during trials involving the two hidden layer network and an initial weight range of ± 3.0. One same weight case and four dual weight cases, out of one hundred trials, failed to converge under these conditions. As in the exclusive or case, there is a significant difference between weight implementations. Networks using the same weights converged, on average, in 705 epochs for same weights and 431 epochs for the dual weight case. This difference is statistically significant ($p < 0.001$). Although there was no interaction between the use of dual weights and weight decay in the case of learning exclusive or, such an interaction shows up for learning the encoder task (Figure 9).

Conclusion

We have introduced a back-propagation architecture which apparently resolves Grossberg's major criticism of this

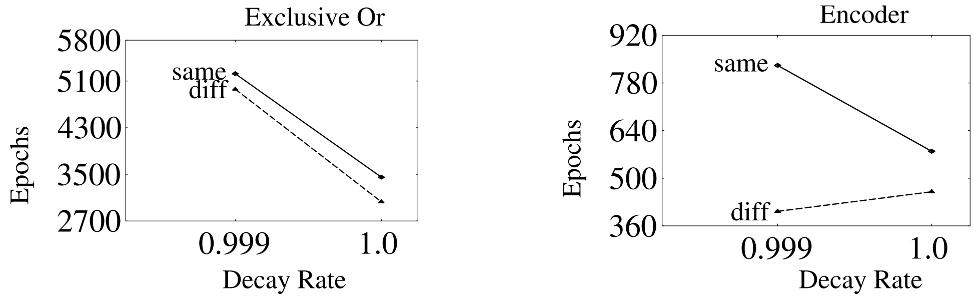

Figure 9 Interaction between learning task, dual weights, and weight decay.

learning technique, namely the problem of weight transport. Through the creation of two separate weight subsystems, the forward weights and the backward weights, we eliminated the need to transmit values from the behavioral mechanism to the learning mechanism since weight decay washes away any initial differences. In addition to providing a counter example to his complaint, we demonstrated that the disagreement was misguided. What is important is that the forward and backward weight systems have functional similarities. The assumption of small differences between initial weights and the sigmoid activation function allows the accumulation of error signals to produce two weight sets that are functionally similar. We have made no claims of improved "convergence", we simply claim that some of the internal mechanisms of back-propagation may not exist at the creation of the device and can emerge over time. The payback associated with the separation of forward and backward pass weights is two-fold. First, it may legitimize back-propagation as a plausible model of a biological learning process. Second, it may simplify the design of physical implementations neural network learning mechanisms.

The same can be said for the replacement of the mathematical operations associated with back-propagation, or other learning methods. We are currently applying the same methodology to other aspects of back-propagation in which we replace fundamental mathematical operations like the derivative of the sigmoid with qualitatively similar functions. Initial results indicate that performance may degrade in terms of learning time and convergence probability, such a design strategy may lead to very efficient implementations.

Acknowledgements

This work was supported by the Office of Naval Research grant number N00014-89-J1200.

References

Fahlman, S. E. (1988). Faster-learning variations on back-propagation: An empirical study. In D. S. Touretzky, G. E. Hinton & T. J. Sejnowski(eds.), *Proceedings of the 1988 Connectionist Models Summer School.* San Mateo, CA: Morgan Kaufman.

Grossberg, S. (1987). Competitive learning: From interactive activation to adaptive resonance. *Cognitive Science. 11*, 23-63.

Hinton, G. E. (1987). Learning translation invariant recognition in a massively parallel network. In G. Goos and J. Hartmanis (Eds.) *PARLE: Parallel Architectures and Languages Europe. Lecture Notes in Computer Science*. Berlin:Springer-Verlag.

Hinton, G. E., & Sejnowski, T. J., (1986) Learning in Boltzman machines. In D. E. Rumelhart, J. L. McClelland, and the PDP Research Group (Eds.) *Parallel Distributed Processing.* Volume 1. Cambridge, MA: MIT Press.

Kolen, J. F., & Pollack, J. B., (1990) Back propagation is sensitive to initial conditions. *Complex Systems. 4*(3), 269-280.

Krogh, A., & Hertz, J. A., (1992). A simple weight decay can improve generalization. In J. E. Moody, S. J. Hanson, and R. P. Lippmann (Eds) *Advances in Neural Information Processing Systems: 4*. San Mateo, CA:Morgan Kaufmann.

Lang, K. J., & Witbrock, M. J. (1988). Learning to tell two spirals apart. In *Proceedings of the 1988 Connectionist Models Summer School.* Carnegie Mellon University. June 17-26, 1988.

Moody, J., (1992). Generalization, weight decay, and architecture selection for nonlinear learning systems. In J. E. Moody, S. J. Hanson, and R. P. Lippmann (Eds) *Advances in Neural Information Processing Systems: 4*. San Mateo, CA:Morgan Kaufmann.

MacKay, D. (1992). A practical bayesian framework for backprop networks.In J. E. Moody, S. J. Hanson, and R. P. Lippmann (Eds) *Advances in Neural Information Processing Systems: 4*. San Mateo, CA:Morgan Kaufmann.

Parker, D. B. (1985). Learning-Logic. Technical Report TR-47. Center for Computational Research in Economics and Management Science. MIT. Cambridge, MA.

Rumelhart D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature. 323*, 533-536.

Zipser, D., & D. E. Rumelhart. (1990). Neurological significance of new learning models. In E. L. Schwartz (Ed.)*, Computational Neuroscience.* Cambridge, MA: MIT Press.