MADALINE RULE II: A Training Algorithm for Neural Networks

By

Capt. Rodney Winter, USAF **Prof. Bernard Widrow**

Dept. of Electrical Engineering
Stanford University

ABSTRACT

A new algorithm for training muti-layer fully connected feed-forward networks of ADALINE neurons has been developed. Such networks cannot be trained by the popular back-propagation algorithm since the ADALINE processing element uses the nondifferentiable signum function for its nonlinearity. The algorithm is called MRII for MADALINE RULE II.

Previously, MRII sucessfully trained the adaptive "descrambler" portion of a neural network system used for translation invariant pattern recognition [1]. Since then, studies of the algorithm's convergence rates and its ability to produce generalizations have been made. These were conducted by training networks with MRII to emulate fixed networks.

A fixed network, acting as a teacher, provides desired responses for the net being trained. MRII trains the adaptive net to emulate the input-output mapping of the teacher. This training is conducted using only a small number of the patterns available in the input space. Once trained, the adaptive net's responses to patterns it has not been trained on are compared to the fixed net's responses to see if the adaptive net has truly generalized. MRII has demonstrated its ability to produce useful generalizations when training the adaptive net on as little as one percent of the input space patterns.

This paper will present the principles and experimental details of the MRII algorithm. Typical learning curves will show the algorithm's efficient use of training data. Architectures that take advantage of MRII's quick learning to produce useful generalizations will be presented.

I The Network

The MRII algorithm is used to train feed-forward layered networks of the type in Figure 1. The processing elements, labelled "AD" in the figure, are ADALINEs. "ADALINE" derives from "adaptive linear neuron."[2]

The details of an ADALINE are shown in Figure 2. The input pattern vector, augmented by a fixed bias input x_0, is represented by $X = [x_0, x_1, \ldots, x_n]^T$. This input vector is weighted by $W = [w_0, w_1, \ldots, w_n]^T$ to form the analog response $y = X^T W = W^T X$. The weight w_0 is the bias or threshold weight. The absolute value of the analog response is referred to as the "confidence level" of the ADALINE. The analog response y is passed through a hard-limiting quantizer to provide the ADALINE's binary response $q = sgn(y)$, where sgn() is +1 for nonnegative argument and -1 otherwise.

The ADALINE acts as a binary classifier, splitting its input pattern space into two regions. For the case of a two dimensional input pattern the decision boundary is described by:

$$y = X^T W = w_0 + w_1 x_1 + w_2 x_2 = 0 \,.$$

In the input pattern space this describes a line:

$$x_2 = -\frac{w_1}{w_2} x_1 - \frac{w_0}{w_2} \,.$$

With appropriate weights this line could be as shown in Figure 3. Input patterns on one side of the separating line are classified +1 while those on the other side are classified -1. In general, the ADALINE splits its input

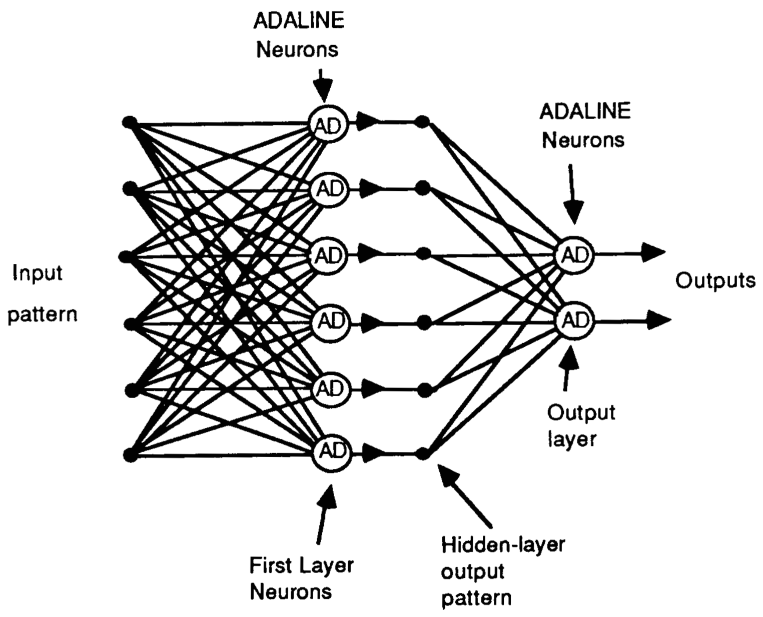

Figure 1: Layered feed-forward ADALINE network.

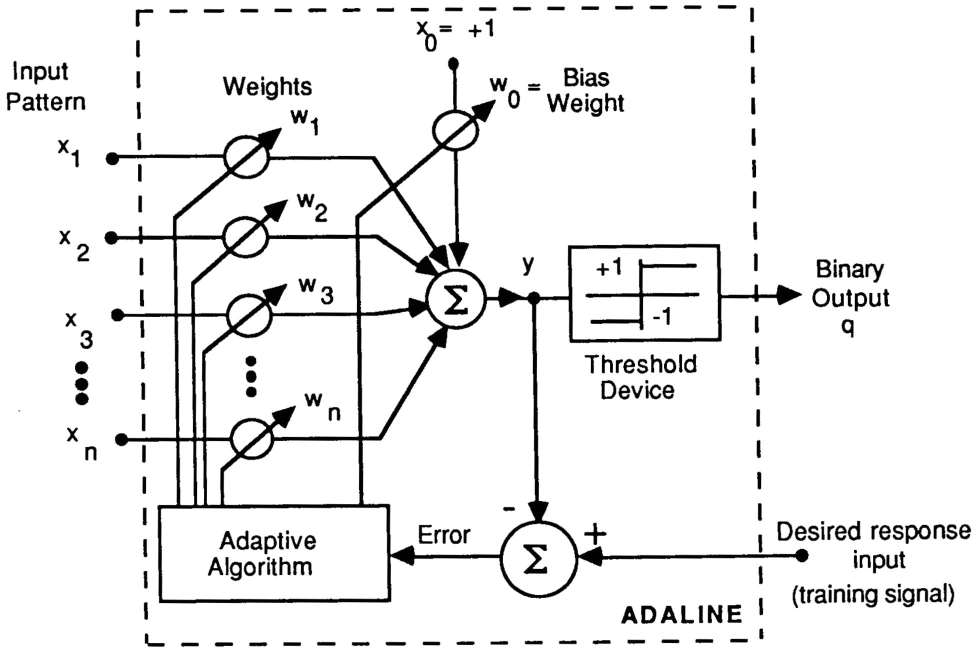

Figure 2: Adaptive linear neuron (ADALINE)

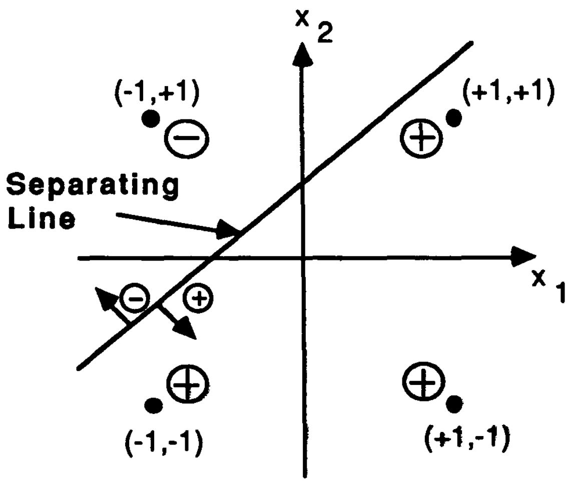

Figure 3: Separating line in pattern space.

pattern space with a hyperplane. Thus, a single ADALINE is capable of realizing only pattern classifications that are linearly separable [3].

II Principles of Multilayer Adaptation

MRII is an extension of the MADALINE (many ADALINEs) adaptation rules developed by Ridgway [4]. Ridgway's networks were composed of a layer of ADALINEs whose outputs then fed into a single fixed logic element. The fixed logic element was typically an "or" element or a majority vote taker element. These logic elements could be realized using an ADALINE with appropriate weights. Thus, Ridgway had a two-layer feed-forward neural net in 1962. The first layer had an arbitrary number of adaptive ADALINE processing elements, but the output layer was constrained to having a single nonadaptive output unit.

MRII removes these restrictions. The output layer may have as many adaptive units as desired. More than two adaptive layers are also possible. This greatly increases the power of the network to realize arbitrary input-output relationships presented to it.

Ridgway's adaptation procedure incorporated the principle of *minimal disturbance*. When the network was presented an input and it responded correctly, no adaptation was done. When a correction to the network was needed, the network was disturbed as little as possible. Consider a network of Ridgway's where the output element was a majority vote taker. When the response was wrong, it is obvious that one or more of the first-layer ADALINEs that were giving a wrong vote needed to be changed. The minimal disturbance principle says to change the ADALINE(s) whose confidence level is closest to zero. The magnitude of the weight change vector needed to cause this ADALINE to reverse its output is smallest. Overall then, the weights in the network are minimally disturbed. Thus, new patterns are accommodated with least likelihood of disturbing the solution for the patterns the network has already been trained on. Given that the patterns are presented acyclically, Ridgway showed these rules would lead to a network solution if one existed.

Current work with networks having several adaptive output elements encounters special challenges. The goal is to adaptively arrive at a set of weights that will map input patterns to desired output patterns for the training set. In doing this with a two-layer network, the first-layer ADALINEs map the inputs into what are commonly referred to as hidden-layer output patterns (refer to Figure 1). These hidden-layer output patterns are then binary classified by each output unit to provide one bit of the network's response. If an output unit gives a wrong response, one has two choices to correct the situation.

The easiest fix is to adjust the weights of the output unit to provide the correct response. This changes the separation of the hidden-layer output pattern space provided by this ADALINE. While this will work for any given pattern, it may not work on the entire training set. The hidden-layer output pattern set that corresponds to the input training patterns may not be linearly separable in the required way for that output unit. Moreover, this hidden-layer output pattern set must be separable by *each* output unit in *its* required way. The choice of the hidden-layer output pattern set is crucial.

The second way to correct an output error then is to change the hidden-layer output pattern to one that provides a correct response by the output ADALINE. The hidden-layer output pattern set is changed by adapting the first-layer ADALINEs. Of course, changing the hidden-layer output pattern may make an output unit that was previously correct now respond incorrectly.

MRII provides a systematic way to arrive at a solution. If the network responds correctly to an input, do nothing. Go on to the next pattern. When an error does occur, make a correction that least disturbs the network as a whole. These principles are easy to follow, and experience indicates they are workable. MRII has been developed experimentally, guided by these principles. The details of the algorithm and the manner in which minimal disturbance is employed follow.

III The MRII Algorithm

When output errors occur in a two-layer network, perform adaptation at the first layer of ADALINEs. By the minimal disturbance principle, select the first-layer ADALINE whose confidence level is closest to zero and reverse its output. This is called a "trial adaptation." Now check the number of output errors. If the number of output units with errors has been reduced, accept the trial change in the weights. If the number of output errors is not reduced, reject the trial adaptation by returning the ADALINE's weight vector to its previous value. If output errors remain, trial adapt the first-layer ADALINE whose confidence level is next closest to zero. Accept or reject the trial change based on whether or not the number of output errors is reduced. Continue in this fashion until all output errors are corrected. Restart the procedure any time a trial adaptation is accepted because a trial change that was rejected previously may be accepted, after the acceptance of another trial adaptation.

It is possible that the procedure will exhaust all trials involving a single ADALINE without reducing the output errors to zero. In such a case, try pairwise trial adaptations. That is, reverse the outputs of the two ADALINEs with confidence levels closest to zero. If this trial is rejected, return the output of the lowest confidence ADALINE to its previous value and reverse the output of the third least confident ADALINE, etc. Again, if an adaptation trial is accepted, restart the procedure starting with single trials. If pairwise trials do not correct all the output errors, then 3-wise, 4-wise, etc., trials are made. Anytime the output errors are reduced to zero, go on to the next pattern and proceed as before.

The minimal disturbance principle suggests that not all ADALINEs in the first layer should be considered for trial adaptations, since, some of them could have relatively large confidence values. The amount of weight change needed to reverse such an ADALINE's output would be large. The more any given ADALINE's weights are changed, the greater becomes the probability that changes in the hidden-layer responses to other input patterns will occur. This is contrary to the minimal disturbance principle. Therefore, consider for trial adaptation only those first layer ADALINEs whose confidence level is low.

The above procedure could cause a radical change in the mapping from input patterns to hidden-layer output patterns. The result will be that many of the training patterns that were correctly responded to before a given change took place could now be wrong. Proceeding ahead, adaptation will tend to restore the previous weights when these now errant patterns are presented again. Unfortunately, a cycle can develop where the same first-layer ADALINEs are adapted back and forth to accommodate different subsets of the training set. A way to break such cycles is to weight the confidence level of the ADALINEs by the number of times they have participated in successful trial adaptations. This will cause the order of consideration for trial adaptation to change. This forces other first-layer ADALINEs to assume responsibility for correct pattern mapping. It is also important to present the patterns acyclically. Random presentation order of the training patterns helps to prevent cycles of adaptation from occurring.

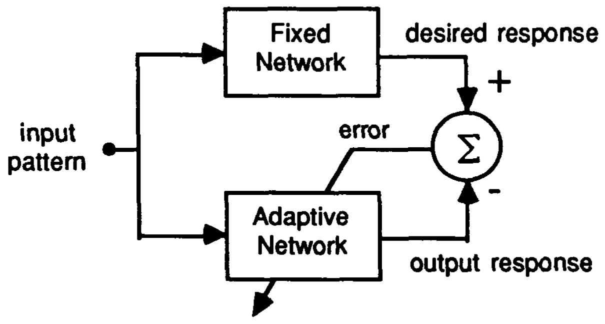

Figure 4: Training an adaptive net to emulate a fixed net.

Sometimes no acceptable change or combination of changes in the first layer will correct all the output errors. This indicates the output-layer ADALINEs are not providing the proper separations of the hidden-layer output patterns and should be adjusted. In such cases, adapt the output units to provide the network's desired response.

Occasionally, training performance by MRII will stagnate. The adaptations that occur are characterized by frequent changes being made to output ADALINEs and few if any adaptations being accepted for first-layer units. The network has settled into a local minimum. There are no first layer adaptations consistent with the minimal disturbance principle that will correct the output errors. The failure of the output ADALINEs to converge to a solution indicate the hidden-layer output patterns are not separable. In such a situation, the minimal disturbance principle must be abandoned. A change in the first-layer ADALINEs must be made to change, perhaps drastically, the mapping of the hidden layer.

The procedure above applies to two-layer networks. The procedure generalizes to networks having more layers. Beginning at the first layer, make trial adaptations consistent with the minimal disturbance principle. Accept those that reduce errors at the output. After doing as well as can be done on the first layer, proceed to the second layer and repeat the trial adaptation procedure. From there, go to the third layer, etc.

The above desciption of the MRII algorithm does not include all the details needed to write computer simulation code. The details are not insignificant and are being refined on a continuing basis. The authors will provide the details of the algorithm to interested people upon request.

IV Simulation Results

A key issue affecting the utility of neural networks is their ability to generalize. Generalization occurs when a neural net can respond in a desired way to inputs it has not been specifically trained on. The anticipated applications of neural networks include those where it is impractical to precompute and store all possible inputs and desired responses to these inputs. To be useful, neural nets must be able, with a limited number of examples, to discern the underlying relationship between inputs and desired responses and encode this relationship in its weights. For many applications this learning need not be perfect. Few things in the world are perfect, and real input data is usually imperfect. For those applications requiring a perfect match between inputs and desired responses, neural nets may not be the answer. The best neural net known to man, the human brain, makes mistakes. With these notions in mind, MRII has been tested for its ability to learn and generalize.

A neural net is not expected to do well on a problem that it cannot solve due to limitations of its architecture. For this reason, a neural net was used as the source of training data for networks trained by MRII. The idea is shown in Figure 4. Input patterns are fed to a fixed net with known architecture and simultaneously to the network being trained. The fixed net's response is used as the desired response for training the adaptive net. The number of output elements for both nets will be the same. The number of

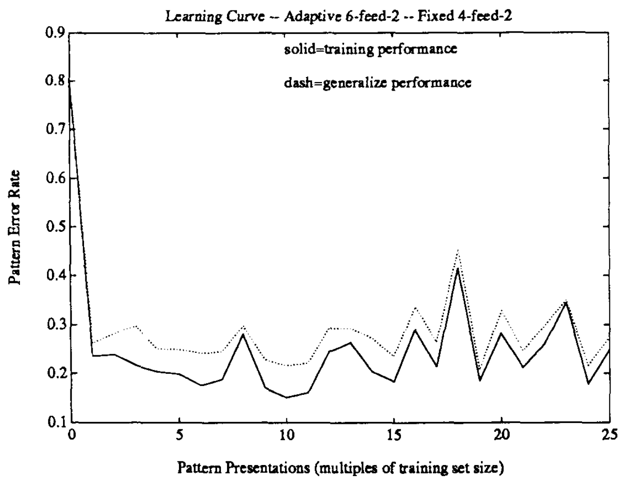

Figure 5: Typical learning curve for the MRII algorithm.

first-layer ADALINEs need not be the same. The effect of using the same number, and sometimes a greater number of first-layer elements in the adaptive net was investigated.

How well is the adaptive net able to learn the training set? How well will the net respond to patterns it has not been trained on? How does the size of the training set affect these performances? What effect does the number of first-layer elements have on these issues? Some preliminary results towards answering these questions follow.

The simulation studies reported here all involved a 16-bit input pattern. The total number of possible input patterns was 65,536. Training sets of 650 and 1500 patterns picked at random from this total were used. Following training, the network was then tested on 6500 unknown patterns picked randomly. The network architectures varied. If a particular network had n first-layer elements and m output elements, this is referred to as an n-feed-m network. The weights in the fixed net were chosen at random. This provided an arbitrary mapping between inputs and outputs, perhaps more arbitrary than for most real applications. In any event, a perfect solution existed, i.e., the weights in the fixed net.

As one might expect, the performance of the adaptive net as training progresses is nonmonotonic. The input pattern to hidden-layer output pattern map is changing frequently as well as are the separations being made by the output units. If arbitrary stops are made during training and the performance of the network against the entire training set checked, a wide range of performance can be observed. Figure 5 is a typical plot of the network performance on the training set and on the generalization test set as a function of the number of patterns presented. Usually, generalization was checked only after training was completed. Here it is seen that generalization performance tracks training performance very well. This suggests that one can "catch" the network at a point of good training performance and be confident that its generalization performance will be nearly as good. The algorithm rarely achieved perfect performance on the training set, but occasionally came quite close.

Table 1 summarizes some typical experimental results. The type of problem investigated can be characterized by the architectures of the fixed and adaptive networks. The first entry in the table shows the result of using a 2-feed-1 adaptive net to emulate a fixed net with the same architecture. The experiment was conducted 25 times as indicated by the "runs" entry, each run starting from a different set of initial conditions in the adaptive net. Error rate on the training set was checked regularly. The run was stopped when a set training error-rate goal was achieved. For those runs never achieving the training goal, training was terminated after a maximum number of presentations, "max pres", were made. The table lists the number

For any given problem there exists a minimum adaptive net architecture that in principle allows a perfect solution. The data, while inconclusive, suggests that chosing a neural network architecture close to this minimum is not necessary. MRII shows a counter-intuitive tolerance for maintaining performance even when over-architectured. Of particular interest is the fact that generalization performance continues to track training performance.

V Summary

Guided by the principle of minimal disturbance, the MRII algorithm has been developed experimentally. The primary motivation for devoloping MRII was to find a way to train multilayer ADALINE networks. It is believed ADALINEs are more easily constructed using existing digital VLSI technology than many of the networks proposed that rely on analog values passing between network layers. The preliminary results obtained to date indicate MRII has interesting and promising properties.

The close tracking that generalization makes with training performance is especially interesting. This property will allow termination of the algorithm based on recent training performance being good. The other interesting property is MRII's apparent stability of performance across changes of architecture in the adaptive network when emulating a given fixed net. This indicates MRII should perform robustly in real applications where the actual complexity of the problem is unknown.

Being a product of experimentation, MRII has not received rigorous mathematical analysis. The properties that MRII displays are somewhat counter-intuitive. An attempt to analyze the algorithm's dynamics and establish the observed properties mathematically is being made. Refinement of the algorithm's details continues. Training and generalization performances in excess of 90% will be needed for many applications. It is believed such performances will be obtained consisitently with further refinement.

References

[1] B. Widrow, R. G. Winter, and R. A. Baxter, "Learning phenomena in layered neural networks," in *IEEE First Annual International Conference on Neural Networks*, 1987.

[2] B. Widrow, "Generalization and information storage in networks of adaline 'neurons'," in *Self Organizing Systems 1962*, (M. C. Yovitz, G. T. Jacobi, and G. D. Goldstein, eds.), pp. 435–461, Washington, DC: Spartan Books, 1962.

[3] P. M. Lewis II and C. L. Coates, *Threshold Logic*. New York: John Wiley and Sons, 1967.

[4] W. C. Ridgway III, *An adaptive logic system with generalizing properties*. PhD thesis, Stanford Electronics Labs. Rep. 1556-1, Stanford University, Stanford, CA, Apr. 1962.