Nonlinear Hebbian Rule: A Statistical Interpretation

Agus Sudjianto
Mohamad H. Hassoun
Computation and Neural Networks Laboratory
Department of Electrical and Computer Engineering
Wayne State University
Detroit, MI 48202

Abstract

Recently, the extension of Hebbian learning to nonlinear units has received increased attention. Some successful applications of this learning rule have been reported as well; however, a fundamental understanding of the capability of this learning rule is still lacking. In this paper, we pursue a better understanding of what the network is actually doing by exploring the statistical characteristics of the criterion function and interpreting the nonlinear unit as a probability integral transformation. To improve the capability of the nonlinear units, data preprocessing is suggested. A better data preprocessing leads to the development of a two-layer network which consists of linear units in the first layer and nonlinear units in the second layer. The linear units capture and filter the linear aspect of the data and the nonlinear units discover the nonlinear effects, such as clustering and other general nonlinear associations among the variables. Several potential applications are demonstrated through the simulation results given throughout this paper.

1. Introduction

The objective of unsupervised learning is to discover features or regularities in a set of training data without teacher signals. Often, a learning rule is designed to transform a set of unlabeled vector data $\{\mathbf{x}^i \in R^n;\ i = 1, 2, ..., m\}$ into another (lower) dimensional set to gain insight and understanding about important features in the data without biasing the results by imposing preconceived structures. If the information in the data is adequately captured by linear relations among the variables, a learning rule may be developed based on covariance (correlation) among the variables. For example, one-layer linear feedforward neural networks employing Hebbian-type learning have been proposed to project data onto the directions of maximum variances, known as principal components (Oja, 1982; see also Hassoun et al., 1993). However, certain topological relationships among the variables may not be adequately captured by such linear relations (Softky and Kammen, 1991; Taylor and Coombes, 1993). Examples of problems which require the discernment of topological relationships in the input data are to handle this class of problems, commonly found in pattern recognition, clustering, and filtering applications. Oja et al. (1991) extended the Hebbian learning to nonlinear units. This extension has drawn increased attention (Softky and Kammen, 1991; Shapiro and Prugel-Bennett, 1992; Taylor and Coombes, 1993; Oja and Karhunen, 1993) and applications (Oja et. al., 1991; Karhunen and Joutsensalo, 1993; Oja and Karhunen, 1993); however, it remains theoretically unclear what the network is actually doing.

In this paper, we explore the statistical characteristics of this learning rule using the probability integral interpretation. The results reveal a better understanding of the processing capabilities of the network and its potential applications. Our findings provide analytical justification to the previously reported applications of using nonlinear units for extracting sinusoidal signals buried in noise. As suspected by previous works, we show that the nonlinear units are indeed capable of capturing nonlinear relationships among the input variables. For some applications, such as clustering, data preprocessing is necessary to either scale the data or remove linear structures in the input data. This preprocessing suggests the use of a two-layer network which is capable of extracting both linear and nonlinear structures in the data.

2. Hebbian Learning in Nonlinear Units

As mentioned above, Oja et al. (1991) generalized the stable form of Hebb's learning rule to nonlinear units. One of their learning rules for a single unit has the following form

$$\Delta \mathbf{w} = \eta_k \left[\mathbf{I} - \mathbf{w}\mathbf{w}^T\right] L(y)\mathbf{x} \qquad (1)$$

where:

η_k : learning rate at step k.
$\mathbf{x}$: input pattern.
$\mathbf{w}$: weight vector.

0-7803-1901-X/94 $4.00 ©1994 IEEE

$y = \mathbf{w}^T \mathbf{x}$: projection of the input patterns onto the weight vector.
$L(y)$: learning signal.

The $L(y)\,\mathbf{x}$ term is the realization of a nonlinear version of Hebb's rule and the $\mathbf{w}\,\mathbf{w}^T L(y)\,\mathbf{x}$ term serves as the stabilizer of Hebb's rule. The learning rule in (1) may be considered as a constrained gradient ascent algorithm by letting

$$\nabla J = L(y)\,\mathbf{x} \tag{2}$$

Depending upon the nonlinearity involved in the learning signal, a criterion function for (1) may be synthesized. For example, if the learning signal is assumed as $L(y) = \tanh(y)\,[1 - \tanh^2(y)]$, then the appropriate criterion function is $J = \tanh^2(y)$. Noticing that if the output of the nonlinear unit is $z = \tanh(y)$, the learning rule in (1) may be perceived as a gradient algorithm to solve an optimization (maximization) problem with the objective function being the variance of the output (Oja et al., 1991). Consider the outputs of a unit, z, and assume that they are governed by a probability density function $p(z)$. Here, the criterion function maximized by Hebb's rule may be written as

$$J = E[z^2] = \int_{-\infty}^{\infty} z^2\, p(z)\, dz \tag{3}$$

subject to $\mathbf{w}^T \mathbf{w} = 1$

Applying constrained gradient ascent [as in (1)] with J as in (3) and using the chain rule leads to the following learning rule

$$\Delta \mathbf{w} = \eta_k \left(\mathbf{I} - \mathbf{w}\,\mathbf{w}^T\right) \mathrm{E}\left[z \frac{dz}{dy} \mathbf{x} \right] \tag{4}$$

where the constant term has been absorbed into η_k. The stochastic approximation version of (4) may be written as

$$\Delta \mathbf{w} = \eta_k \left(\mathbf{I} - \mathbf{w}\,\mathbf{w}^T\right) z \frac{dz}{dy} \mathbf{x} \tag{5}$$

Note that the learning signal in (5) is given by $L(y) = z\frac{dz}{dy}$. If z is a linear function of y, say $z = y$, then $\frac{dz}{dy} = 1$ and the learning rule is reduced to the simple Hebbian rule for a linear unit.

For the case of linear units, the objective function in (3) has an obvious interpretation, i.e., projection of the input patterns in the directions of maximum variances, known as the principal components. The same criterion function applied to nonlinear units, on the other hand, does not have an apparent meaning. Note that both linear and nonlinear learning rules are seeking a set of weight parameters such that the outputs of the unit have the largest variance. The nonlinear unit, however, constraints the output to remain within a bounded range, e.g. $z = \tanh(y)$ limits the output within [-1, 1]. The restriction of the nonlinear unit outputs significantly distinguishes nonlinear units from their linear counterparts and dramatically affects the mechanism of variance maximization. For linear units, the learning rule pushes the weight vector to favor components of the input vectors (or their linear combinations) having the largest variance regardless of the shape of the distribution function, $p(z)$. On the other hand, due to the restriction of the nonlinear outputs, nonlinear learning will produce outputs with a U-shape distribution in order to maximize the output variance. Thus, the variance maximization objective in the criterion function drives the learning rule to prefer a weight vector $\mathbf{w}$ such that the output z is distributed near the extremes -1 and 1. As an example, suppose z follows a standard beta distribution (the beta distribution is chosen because it has various shapes depending on its parameters)

$$p(z) = \frac{1}{B(p,q)} z^{p-1} (1-z)^{q-1} \qquad 0 \le z \le 1 \tag{6}$$

where $B(p, q)$ is the beta function

$$B(p,q) = \int_0^1 t^{p-1} (1-t)^{q-1}\, dt \tag{7}$$

The mean and variance of this standard beta distribution (Johnson and Kotz, 1970) are given as

$$\mathrm{E}[z] = \frac{p}{p+q} \tag{8}$$

$$\mathrm{var}(z) = \frac{p+q}{(p+1)^2 (p+q+1)} \tag{9}$$

Consider three beta distributions, as shown in Figure 1, with the following parameters: (1) $p = q = 3$, (2) $p = q = 1$, and (3) $p = q = 0.5$. The mean of each of the

3 distributions is equal to 0.5, where the variance are equal to 0.0536, 0.1667, and 0.2222, respectively.

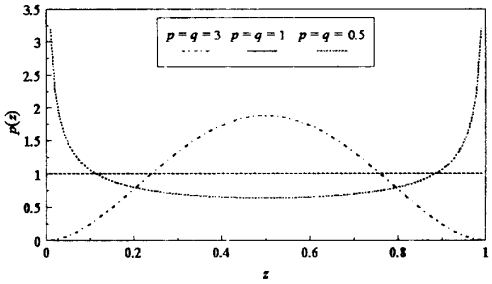

Figure 1. Beta distribution with the following parameters: (1) $p = q = 3$, (2) $p = q = 1$, and (3) $p = q = 0.5$.

Note that the U-shape distribution, $p = q = 0.5$, has the largest variance compare to the others. The criterion function in (3) forces the learning rules in (4) and (5) to prefer this U-shape distribution.

3. Probability Integral Transformation and Nonlinear Units

The nonlinear units, in addition to restricting the output to within a certain range, also transform the probability density of the weighted sum $y = \mathbf{w}^T \mathbf{x}$ into other probability density functions. The probability density function of the output, z, is determined by the probability density function of the input vector, the weight vector, and the shape of the activation function. In the following discussion we show how the shapes of activation functions govern the behavior of the learning rule.

Consider a single unit with the following output relation

$$z = 2\,\phi(y) - 1 \qquad (10)$$

where $\phi(y)$ is a nonlinear function bounded within [0, 1]. Clearly, z takes on values in the interval $-1 \le z \le 1$. If $\phi(y)$ is chosen as a monotonic increasing function on [0, 1], then $\phi(y)$ may be interpreted as a cumulative (probability integral) distribution function. Hence, the output of a nonlinear unit in (10) may be perceived as a linear projection transformation (with projection vector $\mathbf{w}$) of a multivariate random variable $\mathbf{x}$ into a univariate random variable y and followed by a probability integral transformation of y into z. If y is randomly distributed with probability density function $p(y)$ and cumulative distribution function $\phi(y)$, then the probability integral transformation $\phi(y)$ of y results in a uniform distribution within the interval [0, 1] (Rohatgi, 1984). For example, consider the case of y having a standard normal distribution given by,

$$p(y) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{y^2}{2}\right) \qquad (11)$$

the probability integral transformation of y

$$\phi(y) = \int_{-\infty}^{y} p(y)\,dy$$

$$= \int_{-\infty}^{y} \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{y^2}{2}\right) dy \qquad (12)$$

results in z being uniformly distributed in [-1, 1]. On the other hand, if y is not normally distributed, then z will not follow a uniform distribution.

Consider input vectors $\mathbf{x} \in R^n$ where each component is generated by a different probability distribution. The restriction on the nonlinear output causes the learning rule (4) and (5), in order to maximize the output variance, to seek a weight vector, $\mathbf{w}$, such that $p(z)$ has a U-shape distribution (deviates away from a uniform distribution). The learning rules will suppress the components of $\mathbf{x}$ (or their linear combinations) that follow a normal distribution (because they generate a uniformly distributed z) and intensify those which depart from normality and produce U-shape distribution of z. The nonlinear activation function (the probability integral transformation) together with $\mathbf{w}$ play the role of shaping the distribution of z. On the contrary, the linear neuron does not involve a probability integral transformation to change the structure of output distributions. Hence, whatever linear combination of components of $\mathbf{x}$ that produces the largest variance is preferable by the linear learning. Shapiro and Prugel-Bennett (1992) also observed the important role of the shape of the activation function.

To illustrate the significance of the type of activation function used, consider input vectors consisting of 2 components, a pure sinusoidal $x_1 = \sin(t)$; where t is uniformly distributed within $[0, 2\pi]$ and a gaussian random component x_2 with zero mean and unit variance. The probability density and the cumulative distribution functions of x_2 is represented by Equations (11) and (12), respectively. The probability distribution function of x_1 can be easily shown (Rohatgi, 1984) to be

$$g(x_1)=\frac{1}{\pi\sqrt{1-x_1^2}} \qquad (13)$$

This distribution is known as the *arc-sine* distribution (Johnson and Kotz, 1970), and the corresponding cumulative distribution function is given by

$$\phi(x_1)=\frac{1}{\pi}\sin^{-1}(x_1)+0.5 \qquad (14)$$

The shape of the probability density functions of x_1 and x_2 are shown in Figure 2a. Accordingly, the activation functions using (12) and (14) are shown in Figure 2b.

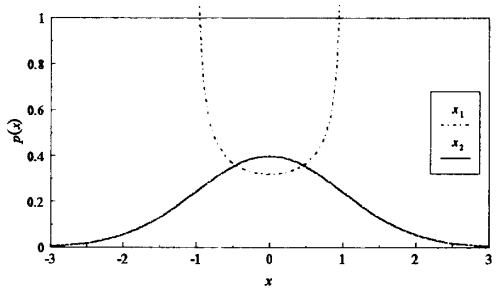

Figure 2a. Probability density function of x_1 from Equation (13) and x_2 from Equation (11).

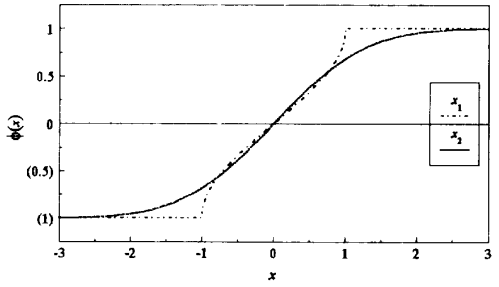

Figure 2b. Activation functions constructed from probability density functions of x_1 and x_2 from Equation (14) and (12), respectively.

Depending on which component of **x** is interesting, an activation function can be selected to enhance (or suppress) either x_1 or x_2. If (12) is chosen as the activation function then x_1 and x_2 produces z with a U-shape distribution and a uniform distribution within [-1, 1], respectively. Similarly, the opposite effect occurs with the selection of (14) as the activation function. To demonstrate this behavior, a set of 100 training samples is generated and a nonlinear unit is trained using the learning rule in (5) with a constant learning rate η = 0.01. Suppose we wish to extract x_1 from the input signal. In this case, x_2 may be suppressed by transforming it into a uniform distribution, and x_1 may be intensified into a U-shape distribution using the probability integral transformation in (12). To avoid the integration calculation, an approximation to the cumulative normal distribution may be obtained by using a cumulative logistic distribution (Johnson and Kotz, 1970), z = tanh(βy). For simplicity we select a logistic distribution with β = 1. For a better approximation, β may be adjusted such that the difference from cumulative normal distribution is less than one percent (see Johnson and Kotz, 1970). Since the two components of each input vector have different scales (variances), a preprocessing calculation is necessary using their standard deviations, $x'_i=\frac{x_i}{\sigma_{x_i}}$, where σ_{x_i} is the standard deviation of x_i. The result of the training is shown in Figure 3a.

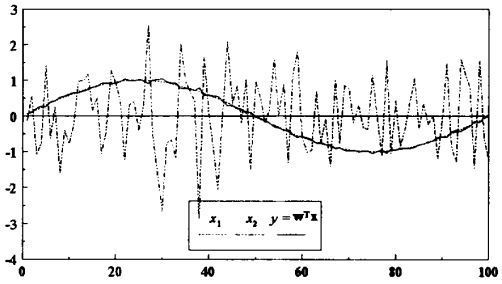

Figure 3a. The result of nonlinear unit training using the activation function in (12) to extract the sinusoidal signal.

Similarly, to extract x_2 we can use a probability integral transformation in (14). Figure 3b shows that the nonlinear unit is capable of extracting the gaussian component.

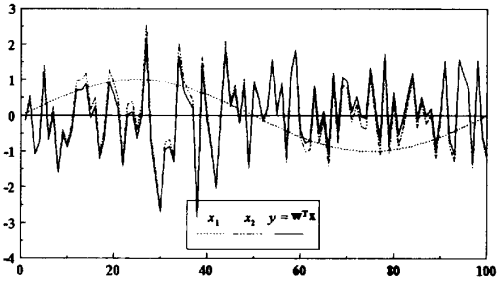

Figure 3b. The result of nonlinear unit training using activation function in (14) to extract gaussian signal.

This result agrees with some early applications for sinusoidal signal extraction (Oja et al., 1991; Karhunen and Joutsensalo, 1992; Oja and Karhunen, 1993). In these applications, the inputs to the network are L successive lags of samples $\mathbf{x}(k) = [x(k), x(k+1), \ldots, x(k+L-1)]^T$. Sigmoidal type functions were chosen for the activation functions. Similar to the previous examples, the sigmoidal signal drives the learning rule to find projection vectors that suppress the gaussian or near gaussian signals and enhance the U-shape distributed signals, i.e. the sinusoidal signals. Note that either the input data or the activation functions need to be scaled to produce the desired results.

4. Clustering Applications and Linear PCA Preprocessing

The fact that Hebbian learning in a saturating nonlinear unit tends to yield an output clustered at saturation extremes leads to a natural application for data clustering. To demonstrate the capability of nonlinear units for data clustering, forty samples of 2-dimensional data are generated as shown in Figure 4. The data are generated from uniform distributions with $1 \le x_2 \le 2$ for both classes and $0 \le x_1 \le 0.5$ and $0.5 \le x_1 \le 1$ for class I and class II, respectively.

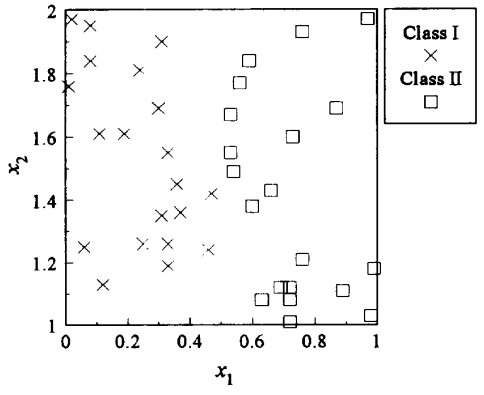

Figure 4. Forty samples of 2-dimensional data from 2 classes.

The data represent two class populations. In this problem we are interested in finding a projection that indicates a clear separation between the two classes. We should not use the class labeling in the training, but it is of course interesting to compare that labeling with any structure found by Hebbian learning in a nonlinear unit. To produce more than 1 cluster, the learning should move away from a projection that produce a unimodal (or gaussian-type) pattern by applying a cumulative normal distribution as the activation function. Data clustering often has little to do with any linear (covariance) structure in the data. By preprocessing the data with a linear PCA network this linear structure may be removed. The idea is to carry out a linear transformation--rotation, location, and scale change--to eliminate all the location, scale and correlational structures. This preprocessing leads to a two-layer network architecture (see Figure 5) where the first layer is a linear PCA network, e.g. Sanger's network (Sanger, 1989), which removes linear structure in the data and the second layer is a nonlinear unit which is responsible for extracting nonlinear structures. The variance scaling in between the two layers is essential to assure that the inputs to the nonlinear unit has the same scale. The linear layer and the variance scaling implement the following transformation:

$$\mathbf{v} = \mathbf{S}^{-\frac{1}{2}} \mathbf{U}^T (\mathbf{x} - \mathrm{E}[\mathbf{x}]) \qquad (15)$$

where $\mathbf{U}$ is a matrix composed of eigenvectors of the covariance matrix of the input vectors, and $\mathbf{S}$ is the corresponding diagonal matrix of eigenvalues.

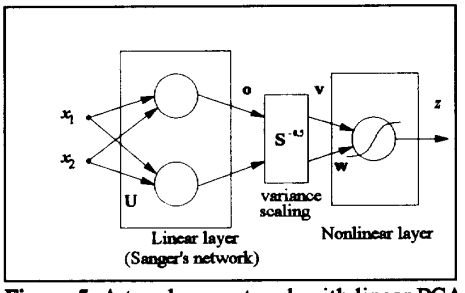

Figure 5. A two-layer network with linear PCA units in the hidden layer and a nonlinear PCA unit in the output layer.

The learning rule in (5), with a constant learning rate $\eta = 0.01$, is used to train the nonlinear unit. Again, for a fast computation an activation function $z = \tanh(y)$ is utilized to approximate the cumulative normal distribution function. The initial value of the weight vector, $\mathbf{w}$, of the nonlinear unit is set equal to the weight vector of the first principal component of the linear PCA to provide nonlinear learning with a good starting point. The projection plots of the first and second principal components $\left(o_i = \mathbf{u}_i^T \mathbf{x}; i = 1,2\right)$ of the linear PCA outputs are shown in Figure 6a and 6b, respectively. Notice that linear PCA is distracted by components with large variances and fails to provide genuine clusters.

Figure 6c shows the plot of the outputs of the nonlinear unit, $z = \tanh(\mathbf{w}^T \mathbf{v})$. The nonlinear unit is capable of learning a projection vector, **w**, that provides good data clustering. For comparison, Figure 6d shows the plot of the outputs of the nonlinear unit with variance scaling but without the linear PCA preprocessing, $z = \tanh(\mathbf{w}^T \mathbf{x}')$; $x'_i = \frac{x_i}{\sigma_i}$. The result demonstrates the necessity of having the linear network removes all linear structure before feeding the data into nonlinear units.

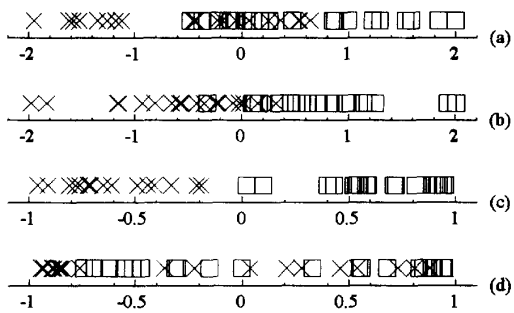

Figure 6. Data clustering using projections to (a) the first principal component of the linear PCA network, (b) the second principal component of the linear PCA network, (c) the first principal component of the nonlinear unit with a linear PCA preprocessing, and (d) the first principal component of the nonlinear unit without a linear PCA preprocessing.

5. Conclusion

We have demonstrated the behavior of nonlinear units by employing a probability integral transformation interpretation of the unit activation function. This approach leads to a better understanding of the learning capabilities of the nonlinear units. Two-layer networks with linear units in the hidden layer are proposed to enhance the capability of nonlinear units. The hidden layer extracts all linear structure in the input data and allows the nonlinear units to exclusively manipulate the nonlinear information. The approach also suggests and justifies potential applications of Hebbian learning in nonlinear units for signal extraction and data clustering.

Acknowledgments

The authors would like to thank Paul Watta for his critical reading of this manuscript.

References

Hassoun, M. H., Sudjianto, A., and Mortazavian, H. (1993), "On the Stability of Hebb's Rule," submitted to *Neural Networks*.

Johnson, N. L., and Kotz, S. (1970), *Distribution in Statistics: Continuous Univariate Distributions-2*, John Wiley, New York.

Karhunen, J. and Joutsensalo, J. (1992), "Nonlinear Hebbian Algorithm for Sinusoidal Frequency Estimation," *Artificial Neural Networks, 2 (Proc. ICANN-92, Brighton, UK)* Aleksander, I. and Taylor, J. (eds.), North-Holland, 1099-1102.

Oja, E. (1982), "A Simplified Neuron Model as a Principal Component Analyzer," *Journal of Mathematical Biology*, **15**, 267-273.

Oja, E. and Karhunen, J. (1993), "Nonlinear PCA: Algorithms and Applications," *Report A18 Helsinki University of Technology*, Espoo, Finland.

Oja, E., Ogawa, H., and Wangviwattana, J. (1991), "Learning in Nonlinear Constrained Hebbian Networks," *Artificial Neural Networks (Proc. ICANN-91, Espoo, Finland)*, Kohonen, T., Makisara, K., Simula, O., and Kangas, J. (eds.), North-Holland, 385-390.

Rohatgi, V. K. (1984), *Statistical Inference*, John Wiley & Sons, New York.

Sanger, D. T. (1989), "Optimal Unsupervised Learning in a Single-layer Linear Feed-forward Neural Network," *Neural Networks*, **2**, 459-473.

Shapiro, J. L. and Prugel-Bennett, A. (1992), "Unsupervised Hebbian Learning and the Shape of the Neuron Activation Function," *Artificial Neural Network*, **2** *(Proc. ICANN-92, Brighton, UK)*, Aleksander, I. and Taylor, J. (eds.), North-Holland, 1099-1102.

Softky, W. R. and Kammen, D. M. (1991), "Correlations in High Dimensional or Asymmetric Data Sets: Hebbian Neuronal Processing," *Neural Networks*, **4**, 337-347

Taylor, J. G. and Coombes, S. (1993), "Learning Higher Order Correlations," *Neural Networks*, **6**, 423-427.