A NEW TRAINING RULE FOR OPTICAL RECOGNITION OF BINARY CHARACTER IMAGES BY SPATIAL CORRELATION

Chanchal Chatterjee* and Vwani Roychowdhury
Purdue University
School of Electrical Engineering, West Lafayette, IN 47907

Abstract *- In this study we shall explore a new method for automatically training and recognizing patterns such as characters or symbols in an image. The research is based upon the commercially proven recognition technique of Spatial Correlation, whose major drawback is the tedious process of training each character, while taking into account the variations in print from sample to sample. The research attempts to completely automate the training process by a* ***new learning rule*** *in feedforward neural networks, to create an "optimal" representation of each character from a representative set of character images. The research presents bounds of the learning constant, and proofs of convergence of the proposed algorithm. The method significantly enhances existing commercial OCR systems that are based on Spatial Correlation.*

I. Introduction

In this paper we shall explore a new method for automatically training and recognizing patterns such as symbols or characters in an image, by machine vision systems. In recent years, due to large production volumes, high cost of manual inspection, and stringent quality control, many industries are searching for efficient and robust Optical Character Recognition (OCR) systems. These industries include the following: (1) pharmaceuticals, (2) food and packaging, (3) semiconductors, (4) automotive, (5) document processing (libraries, check processing, etc.), and (6) mail service (US Postal Service, United Parcel Service, etc.). Other industries include the US military, federal and state governments, and coupon manufacturers.

In recent years, the Pharmaceuticals/Food and Packaging industries have shown considerable interest in OCR. A recent ruling[3] by the Food and Drug Administration (FDA) pertaining to the Current Good Manufacturing Practice regulations for human and veterinary drug products has revised certain labeling control provisions to reduce the frequency of drug product recalls due to mislabeling. Many Food and Drug companies are now requiring the inspection of lot, expiration date, and product codes on product labels and packages. Due to high production speeds (up to 1800 parts per minute in Food industry), and the volume of information printed on these labels, the OCR technology is necessary for almost all applications related to print inspection.

Besides pharmaceuticals, the document processing industry that includes the US military, federal and state governments, libraries and, office information systems have contributed extensively to research in OCR Document analysis research continues to pursue more intelligent handling, better compression-especially through faster OCR. Other industries such as the semiconductor industry needs OCR for automated inspection of characters printed on wafers, IC chips, circuit boards and many others. The automotive industry needs OCR for inspection of stamped or indented characters marked on parts or on the vehicle identification number (VIN) tags. Proper inspection of these characters assure better quality, management, and tracking of products.

A common method for OCR in the above industries, and particularly the pharmaceutical/food and packaging industries, is the method of Spatial Correlation[4,9,10], which evolved from the idea of correlation of points as defined in a two dimensional space. An immediate advantage of this technique is its capability to locate all occurrences of a symbol at the same time regardless of the number present. Furthermore, the method is completely font trainable. However, the drawbacks of this approach are: (1) its dependence on robust training of each character, which is usually performed manually with user-interactive aids; and (2) its capability to accommodate only a limited set of fonts (limited by computer memory) for simultaneous recognition.

In this paper, we shall explore a **new learning algorithm** in neural networks to automatically train characters. The training involves the "optimal" selection of correlation points for each character from a representative set of character images. This method also considers the statistical variations of images of the same character, while exploring differences from other characters. A successful solution of this problem will significantly enhance the state of the art.

*Mr. Chatterjee is partially funded by Phoenix Software Development Company at Sterling Heights, Michigan 48314.

0-7803-1901-X/94 $4.00 ©1994 IEEE

Most research in this area are based on the methods of mathematical morphology[1,2,4,7,9]; most notable of them are by Wilson[9,10] and Haralick[4]. Wilson approached the correlation method from a generalized formalism of mathematical morphology called *matrix morphology*[9]. Moreover, he has introduced a method for automatically training structuring elements for morphological operations on character images. This technique, based on Hebbian learning, is useful to recognize some poorly marked characters on noisy surfaces such as semiconductor wafers.

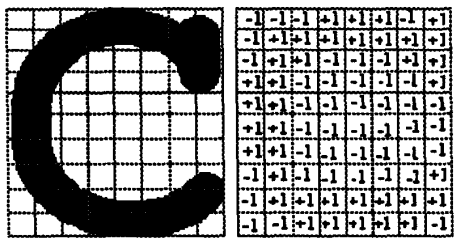

FIG. 1. CHARACTER WITH EXCITATORY AND INHIBITORY CORRELATION POINTS.

The key step in the correlation approach to OCR is the proper selection of a set of correlation points (see Fig. 1) that represent the character and its background. The correlation points representing the character are called *"excitatory"* points, while those representing its background are called *"inhibitory"* points.

In order to create an optimal selection of excitatory and inhibitory correlation points, as well as a set of weights for these points, we have presented a new learning rule in feedforward neural networks. Many standard learning rules such as the perceptron learning rule[5,6,11] can be used to create such weights. However, the proposed rule has the following features:

(1) The proposed learning technique creates a set of *weights* for the excitatory and inhibitory correlation points within pre-specified bounds. The perceptron learning rule does not produce weights within such a range.

(2) The proposed rule trains an "optimal" set of excitatory and inhibitory correlation points for each character from a *representative set of character images*. In contrast, state of the art methods for OCR[2,9] use the Hebbian learning rule[5,9,11] to train morphological structuring elements to recognize each character.

(3) The method automatically computes a *percentage correlation level* for each character that determines the threshold correlation necessary to successfully recognize each character.

Section 2 describes the proposed algorithm. Section 3 contains proofs of convergence of the proposed learning rule and bounds of the learning rate. Section 4 contains experimental results on the proposed method.

II. Proposed Approach

This section explains the new training rule used to optimally select a set of excitatory and inhibitory correlation points, their weights, and the threshold correlation for each character. The new rule is designed for supervised learning in feedforward neural networks. In order to apply our learning rule, we have a set of representative images of characters with variations in print. Fig. 2 shows an example of variations in prints for characters printed by the same ink jet printer on different parts.

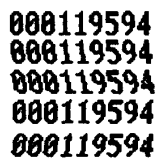

FIG. 2. VARIATIONS IN CHARACTER PRINTS FROM SAMPLE TO SAMPLE.

In this research, we shall consider only binary images of characters with symmetric pixel values of +1 and -1. Furthermore, we shall consider all characters with same scale, i.e. we shall only train the correlation points for characters of a given size (within tolerance).

The OCR system is designed with the feedforward neural model where every character is represented by a node in the first layer of a network (see Fig. 3). Although this research has been applied to multilayer feedforward networks, we shall only consider the first layer of the model in this paper. The OCR system consists of two parts: (1) a training step in which the network is trained with a representative set of character images by the new supervised learning method; and (2) an execution step in which the trained network is applied to binary character images for recognition.

Fig. 3 shows a character node for the first layer of the neural model. In the first layer, there is one such node for each character. The node receives a set of inputs $X=(x_1,...,x_n)^T$ ($x_k \in [-1,+1]$, k=1...n) from the binary character image. Let us assume that there are N characters to be trained. For the i^{th} character (i=1...N), the input vector X is weighted by weight vector $W(i)=(w_1(i)..,w_n(i))^T$ and summed to produce an output

$W^T(i)X$. A weight $w_0(i)$ with input x_0=-1 is used to threshold the summed output $W^T(i)X$ which is then sent through a threshold logic unit (TLU) to determine the final output o(i). The threshold weight $w_0(i)$ is the *correlation level*. The TLU is the sgn(·) function defined as sgn(x)={+1 for x>0, and -1 for x<0}.

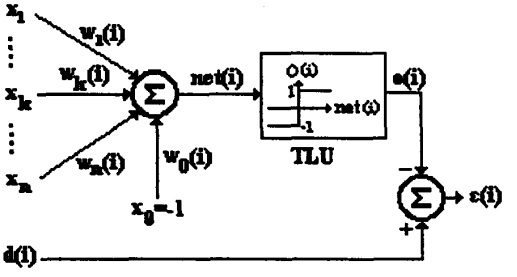

FIG 3. TRAINING NODE FOR CHARACTER i.

In the training step, a set of M training images are used to compute the following: (1) weight vector W(i), and (2) threshold weight $w_0(i)$ for character i, i=1...N. Assuming the characters are indexed by i (i=1,...,N), and the images are indexed by j (j=1,...,M), the new training rule for the i^{th} character is:

$$W(i,j+1) = W(i,j) + \eta_1(X(j)-d(i,j)W(i,j))(d(i,j)-o(i,j)),$$
$$i=1,...,N,\ j=1,...,M \qquad (1)$$

Here η_1 is the learning rate. Input vector X(j) and weight vector W(i,j) are defined as:
$X(j) = (x_1(j),...,x_n(j))^T$, and $W(i,j) = (w_1(i,j),...,w_n(i,j))^T$.
The desired output d(i,j)=+1 when input vector X(j) belongs to character i, and -1 when input X(j) does not belong to character i (i=1...N). The output o(i,j) of the network is: $o(i,j) = sgn(W^T(i,j)X(j)-w_0(i,j)$.

The threshold weight $w_0(i,j)$ is also computed by the same adaptive algorithm (1) in which the input x(j)=-1.

$$w_0(i,j+1) = w_0(i,j) - \eta_2(-1-d(i,j)w_0(i,j))(d(i,j)-o(i,j)). \quad (2)$$

Here $w_0(i,j)$ is the threshold weight for the i^{th} character with the j^{th} training image.

The algorithm for the training rule is as follows:
For the i^{th} character trained (i=1...N):

(1) Start from the first image $X(j_1)$, $1 \le j_1 \le M$, of i^{th} character, where $X(j_1)$ is used as the first estimate of weight vector W(i,1). Compute the first estimate of threshold weight $w_0(i,1)=W^T(i,1)X(j_1)$.
(2) For the next character image $X(j_2)$, $1 \le j_2 \le M$, compute the network output $o(i,j) = sgn(W^T(i,j)X(j)-w_0(i,j))$.
(3) Update the weight vector W(i,j) by the learning rule (1). Also update threshold weight w_0 by (2).
(4) Repeat steps 2-3 for all training images X(j), j=1...M, of characters.

The algorithm simultaneously computes weights W(i), and the threshold weight $w_0(i)$ (correlation level) for character i. Furthermore, the algorithm trains character i in the context of other characters. The process is repeated for all characters. In our learning rule, we have considered two different learning rates η_1 and η_2. In practice, we shall use only one value of learning rate $\eta=\eta_1=\eta_2$.

Note that the above algorithm differs from the perceptron learning rule whereby, the input to our network is X(j)-d(i,j)W(i,j) instead of X(j) in the perceptron rule. If the input X(j) is from the current training character i (i.e. d(i,j)=+1), the network input is X(j)-W(i,j). Thus, the new training rule will only consider those inputs that are different from the previous weights. This method reduces the differences between the current weight W(i,j) and the current input X(j) while maintaining their similarities. If the input X(j) is not from the current training character i (i.e. d(i,j)=-1), the network input in X(j)+W(i,j). In this case, the new training rule reduces the similarities between X(j) and W(i,j) while maintaining their differences. This property and others are discussed later in the next Section.

III. Convergence of the New Learning Rule

This section discusses the convergence properties of the new learning rule (1), and bounds of the learning rate η. Some properties of the weight vector W are also discussed.

Lemma 1: A sufficient condition for learning rule (1) to be stable is $0<\eta<1$, under which condition, the optimum weight vector $W^*(i)$ for character i (i=1...N) is $E[\varepsilon(i,j)X(j)]/E[d(i,j)\varepsilon(i,j)]$ where $\varepsilon(i,j)=d(i,j)-o(i,j)$.

We shall assume that X(j), d(i,j) and $\varepsilon(i,j)$ are jointly stationary. If we further assume that X(j) is statistically independent over j, then W(i,j) is independent of X(j), d(i,j) and $\varepsilon(i,j)$.

With these assumptions, (1) can be:
$W(i,j+1) = (1-\eta d(i,j)\varepsilon(i,j))W(i,j) + \eta\varepsilon(i,j)X(j)$.
The expected value (E()) of the above expression is:

$$E[W(i,j+1)] = (1-\eta a(i))E[W(i,j)] + \eta P(i), \qquad (3)$$

where $a(i)=E[d(i,j)\varepsilon(i,j)]$, and $P(i)=E[\varepsilon(i,j)X(j)]$.

The difference equation (3) can be generalized to the j^{th} iteration as follows:

$$E[W(i,j+1)]=(1-\eta a(i))^j E[W(i,1)] + P(i)/a(i). \qquad (4)$$

The above equation (4) is stable for $|1-\eta a(i)|<1$ which

yields $0<\eta<2/a(i)$. Since $d(i,j)$ and $\varepsilon(i,j)$ have the same sign, $a(i)>0$. The maximum value of $a(i)=2$ since the possible values of $\varepsilon(i,j)=0,+2$ or -2. Thus, the sufficient condition for stability of (1) is $0<\eta<1$.

Furthermore, as $j\rightarrow\infty$, from (4) we obtain:

$$W^*(i) = \lim(j\rightarrow\infty)\, E[W(i,j))] = P(i)/a(i). \quad (5)$$

Note that the above optimum solution (5) for W(i) is different from the mean E[W(i,j)] of the weight vectors. Further note that substituting (5) into (3) we obtain:

$$E[W(i,j+1)]-W^*(i) = (1-\eta a(i))(E[W(i,j)]-W^*(i)). \quad (6)$$

Adopting the symbol $V(i,j)=E[W(i,j)]-W^*(i)$, we obtain the following linear difference equation from (6):

$$V(i,j+1) = (1-\eta a(i))\, V(i,j).$$

Proof of Convergence of Learning Rule (1):

From (1) we obtain:

$$\Delta W(i,j) = W(i,j+1)-W(i,j)$$
$$= -\eta d(i,j)\varepsilon(i,j)W(i,j)+\eta\varepsilon(i,j)X(j)$$

Computing the expected values:

$E[\Delta W(i,j)] = -\eta a(i)E[W(i,j)] + \eta P(i)$. Using (5),

$\lim(j\rightarrow\infty)\, E[\Delta W(i,j)] = -\eta a(i)W^*(i) + \eta P(i) = 0$.

Therefore, the weight increments eventually diminish to 0, and the algorithm converges to the optimum weight $W^*(i)$ if each character pattern is linearly separable from the remaining character patterns.

However, as observed in our experimental results (Sec. 3), and pointed out by the discussion below, considerably more stringent conditions on η than those given in Lemma 1 are required to ensure the convergence of the weight vector W(i).

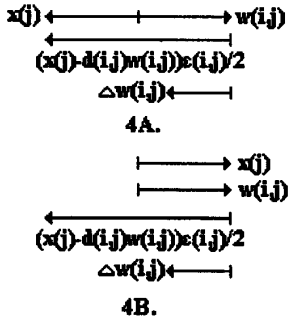

FIG. 4 $\Delta W(i,j)$ FOR (A) $d(i,j)=+1$, $\varepsilon(i,j)=+2$, AND (B) $d(i,j)=-1$, $\varepsilon(i,j)=-2$.

Fig. 4a shows an example where input $x(j)=-1$ while $w(i,j)=+1$ such that input x(j) is from character i (i.e. $d(i,j)=+1$). If output $o(i,j)=-1$, then error $\varepsilon(i,j)=+2$. Thus, $(x(j)-d(i,j)w(i,j))\varepsilon(i,j)=-4$. If $\eta=1$, $\Delta w(i,j)=-4$ and $w(i,j+1)=-3$ which overshoots the input x(j) by -2. On the other hand, if $\eta=½$, $\Delta w(i,j)=-2$ and $w(i,j+1)=x(j)= -1$. Thus $\eta=½$ adjusts the old weight $w(i,j)=+1$ such that the new weight $w(i,j+1)$ matches the current input x(j) exactly. Therefore $\eta=½$ appears to be a more stable choice than $\eta=1$.

Fig. 4b shows an example for $d(i,j)=-1$, i.e. when input x(j) is not from character i. Error $\varepsilon(i,j)=-2$, and assuming $x(j)=w(i,j)=+1$, $(x(j)-d(i,j)w(i,j))\varepsilon(i,j)=-4$. If $\eta=1$, $\Delta w(i,j)=-4$ and $w(i,j+1)=-3$ which overshoots -x(j) by -2. On the other hand, if $\eta=½$ $\Delta w(i,j)=-2$, and $w(i,j+1)=-x(j)=-1$. Since, $\eta=½$ compensates the weight $w(i,j)$ such that the new weight $w(i,j+1)$ completely matches (for $d(i,j)=+1$) or mis-matches (for $d(i,j)=-1$) input x(j), a more appropriate choice of η seems to be $<½$. The analysis below corroborates this observation.

Lemma 2: A sufficient condition for the convergence of the learning algorithm (1) is $0<\eta<1/3$.

Computing the norm of (1), we obtain:

$$|W(i,j+1)|^2 = (1-\eta d(i,j)\varepsilon(i,j))^2\, |W(i,j)|^2 + \eta^2\varepsilon(i,j)^2N +$$
$$2\eta\varepsilon(i,j)(1-\eta d(i,j)\varepsilon(i,j))W^T(i,j)X(j). \quad (7)$$

Computing the expected value of (7), we obtain:

$$E[|W(i,j+1)|^2] = (1+\eta^2\xi(i)-2\eta a(i))E[|W(i,j)|^2] +$$
$$\eta^2\varepsilon(i,j)^2N - 2\eta^2b(i)E[W^T(i,j)X(j)]. \quad (8)$$

Here $\xi(i)=E[\varepsilon(i,j)^2]$, $b(i)=E[d(i,j)\varepsilon(i,j)^2]$, and a(i) is as defined in (3). Assuming that $E[W^T(i,j)X(j)] \geq E[|W(i,j)|^2]$, we can re-arrange (8) as follows:

$$E[|W(i,j+1)|^2] = (\eta^2(\xi(i)-2b(i))-2\eta a(i)+1)E[|W(i,j)|^2] +$$
$$\eta^2\xi(i)N. \quad (9)$$

The above difference equation converges under the condition: $|\eta^2(\xi(i)-2b(i))-2\eta a(i)+1| < 1$, which yields $0<\eta<2a(i)/(\xi(i)-2b(i))$. A sufficient condition for the above bounds can be obtained for $a(i)=2$, $\xi(i)=4$, $b(i)=-4$, as $0<\eta<1/3$.

Lemma 3: For $\eta<½$, every component of weight vector W(i,j) in the new learning algorithm is bounded by the range of each component of input vector X(j), j=1...M.

Since d(i,j) and o(i,j) have the values {+1 or -1}, error $\varepsilon(i,j)$ for each node can be {0, +2, or -2}. For these three values of $\varepsilon(i,j)$, every component w(i,j) of weight vector W(i,j) in learning rule (1) is:

$w(i,j+1) = w(i,j)$ for $\varepsilon(i,j)= 0$, (10a)

$w(i,j+1) = w(i,j)(1-2\eta) + 2\eta x(j)$ for $\varepsilon(i,j)=+2$, (10b)

$w(i,j+1) = w(i,j)(1-2\eta) - 2\eta x(j)$ for $\varepsilon(i,j)= -2$. (10c)

Since $w(i,1)=x(1)$, the maximum value of $w(i,j+1)$ in (10b) is +1 for $w(i,j)=+1$ and $x(j)=+1$, and the

minimum value is -1 for w(i,j)=-1 and x(j)=-1. Similarly, the maximum value of w(i,j+1) in (10c) is +1 for w(i,j)=+1 and x(j)=-1, and the minimum value is -1 for w(i,j)=-1 and x(j)=+1. Thus the range [-1,+1] of w(i,j) is same as the range of x(j), j=1...M.

IV. Experimental Results

The proposed neural learning algorithm is applied on a set of 162 character images with 10 different numeric characters (characters 0-9). A representative sample is shown in Fig. 2. These characters are taught by the learning rule (1) for different values of learning rate η. A new set of 162 characters marked by the same ink jet printer is used to test the algorithm. The results for these tests are described below. Note that the threshold weight w_0 is expressed in percentage of the maximum possible weighted sum $W^T(i,j)X(j)$. Since we have a grid of 20X20 pixels for each character, the maximum possible weighted sum $W^T(i,j)X(j)$=400.

Table 1. Number of iterations required to compute weight vector W and threshold weight w_0 for η=0.01.

Char.	Iters.	w_0 (%)
0	4	40.96
1	9	57.85
2	16	72.31
3	26	59.04
4	29	55.55
5	6	45.34
6	26	59.04
7	9	83.33
8	4	49.18
9	4	57.85

Table 2. Number of iterations required to compute weight vector W and threshold weight w_0 for η=0.02.

Char.	Iters.	w_0 (%)
0	2	38.89
1	5	56.36
2	8	72.07
3	13	58.72
4	15	54.09
5	3	44.06
6	13	58.72
7	5	81.49
8	2	47.83
9	2	56.36

Table 3. Number of iterations required to compute weight vector W and threshold weight w_0 for η=0.05.

Char.	Iters.	w_0 (%)
0	1	34.36
1	2	53.03
2	3	72.83
3	5	58.95
4	6	53.03
5	2	34.36
6	5	58.95
7	2	80.95
8	2	38.20
9	1	53.03

Table 4. Number of iterations required to compute weight vector W and threshold weight w_0 for η=0.30.

Char.	Iters.	w_0 (%)
0	**	**
1	1	39.85
2	2	25.41
3	**	**
4	1	39.85
5	**	**
6	1	39.85
7	**	**
8	**	**
9	**	**

It is clear from Tables 1-5 that for η<0.3, the proposed learning rule (1) reached successful convergence. The number of iterations required for convergence decreased linearly with increase in values of η. Furthermore, the computed threshold weights w_0 and the weight vectors W are very similar (within quantization error) for different values of η. However, for η=0.3, the proposed rule failed to converge for characters 0, 3, 5, 7, 8, and 9. Thus, the proposed algorithm can be used successfully within a range of η <0.3 which is within Lemmas 1-2. The optimum selection of η within this range depends on the experimental data. We recommend the choice of η =0.05 for the proposed application.

Fig. 4a below shows a binary representation of character "9" (printed sideways). This character image forms one input X(j) for training rule (1). After applying the training rule for all 162 characters in the data base (with η=0.05), we obtained the weight vector W, as represented in an image form in Figure 4b below. Threshold weight w_0=53.03%.

-1	-1	-1	-1	-1	-1	1	1	1	-1
-1	-1	-1	-1	-1	1	1	1	1	-1
-1	1	-1	-1	-1	1	1	1	1	1
-1	1	1	-1	-1	1	-1	-1	-1	1
-1	1	1	1	-1	1	-1	-1	-1	1
-1	-1	1	1	1	1	-1	-1	-1	1
-1	-1	-1	1	1	1	-1	-1	-1	1
-1	-1	-1	-1	1	1	1	-1	1	1
-1	-1	-1	-1	-1	1	1	1	1	1
-1	-1	-1	-1	-1	-1	-1	1	1	-1

5A.

-1.0	-1.0	-1.0	-1.0	-1.0	0.6	0.6	0.5	-1.0
-1.0	-1.0	-1.0	-1.0	0.6	1.0	1.0	1.0	-0.8
0.6	-1.0	-1.0	-0.6	0.6	0.9	0.7	1.0	0.8
0.6	0.6	-1.0	-0.7	1.0	-0.6	-1.0	-0.1	1.0
1.0	0.6	0.6	-0.7	1.0	-1.0	-1.0	-1.0	1.0
-0.6	1.0	0.8	0.6	1.0	-1.0	-1.0	-1.0	1.0
-0.8	-0.6	1.0	1.0	1.0	-1.0	-1.0	-0.8	1.0
-1.0	-0.8	-0.6	1.0	1.0	0.6	-1.0	1.0	1.0
-1.0	-1.0	-1.0	-0.8	1.0	1.0	0.8	1.0	0.7
-1.0	-1.0	-1.0	-1.0	-1.0	-0.1	1.0	1.0	-1.0

5B

FIG. 5 AN EXAMPLE BINARY IMAGE (5A) AND WEIGHT VECTOR (5B) FOR CHARACTER "9" (PRINTED SIDEWAYS) WITH η=0.05.

V. Conclusion

In this paper, we have proposed a new supervised learning rule in feedforward neural networks. We have shown the utility of this rule in the training of binary character images for OCR. This research continues to explore the training of gray scale characters and also the implementation of this rule in multilayer networks. The general implementation is useful in cases where the character patterns are not linearly separable.

References

[1] V. Chandra and R. Sudhakar, "Recent Devel. in Artificial Neural Network Based Character Recognition: A Performance Study", *Proc. IEEE Southeastcon*, Knoxville, TN, pp. 633-637.

[2] C. Chatterjee, "Rotation and Position Invariant Optical Character Recognition", *U.S. Patent* serial number 07/710, 840.

[3] *Federal Register*, Vol. 58, No. 147, August 3, 1993, Rules and Regulations, pp. 41348-41354.

[4] R.M. Haralick et al., "Image Analysis using Mathematical Morphology", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 9, No. 4, pp. 532-550, 1988.

[5] J. Hertz, A. Krogh, R.G. Palmer, *Introduction to the Theory of Neural Computation*, New York: Addison-Wesley Publishing Co., 1991.

[6] P. Peretto, *An Introduction to the Modeling of Neural Networks*, Cambridge University Press, Cambridge, U.K. 1992.

[7] J. Serra, *Image Analysis and Mathematical Morphology*, Academic Press, London, 1982.

[8] B. Widrow and S.D. Stearns, *Adaptive Signal Processing*, Prentice-Hall, New Jersey, 1985.

[9] S.S. Wilson, "Theory of Matrix Morphology", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Sept. 1992.

[10] S.S. Wilson, "Teaching network connectivity using simulated annealing on a massively parallel processor", *Proceedings IEEE*, Vol. 79, No. 4, pp. 559-566, 1991.

[11] J.M. Zurada, *Introduction to Artificial Neural Systems*, West Publishing Company, St. Paul, MN, 1992.