MORPHOLOGICAL/RANK NEURAL NETWORKS AND THEIR ADAPTIVE OPTIMAL DESIGN FOR IMAGE PROCESSING

Lúcio F. C. Pessoa and Petros Maragos

School of Electrical and Computer Engineering
Georgia Institute of Technology, Atlanta, GA 30332-0250
lpessoa@ece.gatech.edu

ABSTRACT

In this paper we formulate a general class of neural network based filters, where each node is a morphological/rank operation. This type of system is computationally efficient since no multiplications are necessary. The introduction of such networks is partially motivated from observations that internal structures of a neuron can generate logic operations. An efficient adaptive optimal design procedure is proposed for these networks, based on the back-propagation algorithm. The procedure is optimal under the LMS criterion. Finally, experimental results are illustrated in problems of noise cancellation, encouraging the use of such class of systems and its training algorithm as important tools for nonlinear signal and image processing.

1. INTRODUCTION

Adaptive filters and (artificial) neural networks (NNs) are closely related, and their adaptation/training can be studied under the same framework [1]. In this sense, a NN-filter is a multilayer feed-forward network with only one output, where its training process corresponds to the filter design. The usefulness of NN-filters can be efficiently investigated due to the existence of the back-propagation algorithm [2], which is their usual design procedure, and represents a generalization of the LMS algorithm for feed-forward networks. In this way, the filter design is viewed as a problem of unconstrained optimization that is iteratively solved by the method of steepest descent.

The structure of each node in a NN-filter represents the essence of the system, so that some effort should be directed to consider appropriate node structures. The most common structure of a node is a linear combiner followed by a nonlinearity of the logistic type. This structure was initially defined to model the input-output characteristic of a *neuron*, *i.e.*, the fundamental cell in the brain. More recently, however, a thorough study of the neuron indicated its enormous complexity, resulting in the conclusion that the neuron itself is a complex network. Furthermore, based on computer simulations, Shepherd and Brayton [3] observed that some internal interactions in a neuron can support logic operations, including AND, OR and AND-NOT gates, so that a collection of neurons can be able to compute arbitrarily complex logic functions. Using morphological/rank operations, an alternative node structure can be defined to incorporate these internal properties of a neuron.

Morphological systems [4] and their related rank operations [5] are nonlinear methods widely used in image processing and analysis, even though very few ideas exist for their optimal design. Furthermore, morphological networks [6, 7] represent a simple way to design systems based on morphological/rank operations. Nevertheless, due to the non-differentiability of such operations, the design procedures for those networks are not yet very efficient. To overcome this problem, a general class of nonlinear systems, called *morphological/rank neural networks*, is formulated and efficiently designed in this paper. Some of the new improvements are: (i) easier way to estimate the gradients of the output with respect to the structuring elements; (ii) simple design choice to adapt the rank parameters via an auxiliary real variable; (iii) definition of a smoothed rank function to improve the numerical robustness of the adaptive procedure; and (iv) systematic design methodology based on the back-propagation algorithm.

This work was partially supported by the US National Science Foundation under grant MIP-94-21677, and by CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico), Brasília, Brazil, through a Doctoral Fellowship under grant 200.846/92-2.

2. MORPHOLOGICAL/RANK NEURAL NETWORKS

A *m*orphological/*r*ank *n*eural *n*etwork (MRNN) is a NN-filter where each node is a morphological/rank operation. It is formally defined by the following set of recursive equations

$$\begin{aligned} \underline{y}^{(l)} &= (y_1^{(l)}, y_2^{(l)}, \cdots, y_{N_l}^{(l)}) \ , \ l = 1, 2, \cdots, L \ , \\ y_m^{(l)} &\equiv \mathcal{R}_{r_m^{(l)}}(\underline{y}^{(l-1)} + \underline{a}_m^{(l)}) \ , \ m = 1, 2, \cdots, N_l \ , \end{aligned} \tag{1}$$

where l is the layer number, and N_l is the number of nodes in layer l ($N_L = 1$). The function $\mathcal{R}_r(\underline{t})$ is the *r-th rank function* of the vector $\underline{t}$. It is evaluated by sorting the elements of $\underline{t}$ in decreasing order, and taking the r-th ordered element as the output. The vectors $\underline{a}_m^{(l)}$ represent structuring elements, and the rank parameters $r_m^{(l)}$ control the order of the operations. The structure of the l-th layer is illustrated in Figure 1. Observe that, for each node, the morphological operations of dilation ($r = 1$) and erosion ($r = N_l$) can be

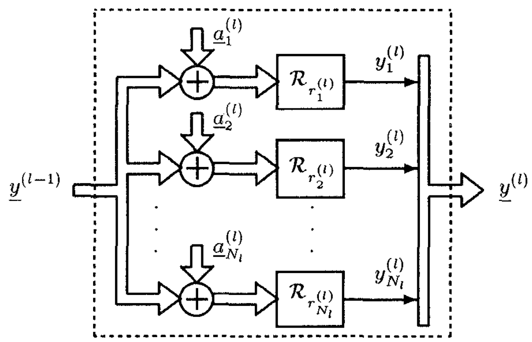


Figure 1: Structure of the l-th layer in a MRNN.

obtained as special cases. Furthermore,

$$\begin{array}{lll} \underline{y}^{(0)} = \underline{x} = (x_1, x_2, \cdots, x_{N_0}) & \text{(input)} \\ \underline{y}^{(L)} = y_1^{(L)} = y & \text{(output)} \end{array} \tag{2}$$

Since we are going to design MRNNs using an adaptive algorithm, derivatives of rank functions will be needed. However, these functions are *not* differentiable in the common sense, so that some design alternative is required. Our approach is to implicitly express a rank function $z = \mathcal{R}_r(\underline{t})$ as an inner product $\underline{c} \cdot \underline{t}'$, and use the vector $\underline{c}$ as a design choice to estimate $\partial z / \partial \underline{t}$. This vector $\underline{c} = \underline{c}(\underline{t}, r)$, with $\underline{t} \in \mathbb{R}^n$ and $r \in \{1, 2, \cdots, n\}$, will be called *rank indicator vector* and defined by

$$\underline{c} \equiv \frac{Q(z\underline{1} - \underline{t})}{Q(z\underline{1} - \underline{t}) \cdot \underline{1}'} \ , \ z = \mathcal{R}_r(\underline{t}) \ , \tag{3}$$

where $\underline{1} = (1, 1, \cdots, 1)$,

$$Q(\underline{v}) = (q(v_1), q(v_2), \cdots, q(v_n)) \ , \tag{4}$$

and $q(v)$, $v \in \mathbb{R}$, is the *unit sample function*, defined by

$$q(v) \equiv \left\{ \begin{array}{ll} 1 & , \text{ if } v = 0 \\ 0 & , \text{ otherwise} \end{array} \right. \tag{5}$$

The rank indicator vector marks the locations in $\underline{t}$ where the z value occurs and has unit area ($\underline{c} \cdot \underline{1}' = 1$).

In practice, for increased numerical robustness, the unit sample function can be approximated by bell-shaped functions, $q_\sigma(v)$, such as

$$\operatorname{sech}^2(\frac{v}{2\sigma}) \quad \text{or} \quad \exp[-\frac{1}{2}(v/\sigma)^2] \ .$$

Likewise, we define

$$Q_\sigma(\underline{v}) = (q_\sigma(v_1), q_\sigma(v_2), \cdots, q_\sigma(v_n)) \tag{6}$$

and

$$\underline{c}_\sigma = \frac{Q_\sigma(z\underline{1} - \underline{t})}{Q_\sigma(z\underline{1} - \underline{t}) \cdot \underline{1}'} \ , \ z = \mathcal{R}_r(\underline{t}) \ , \tag{7}$$

resulting in the *smoothed r-th rank function*

$$\mathcal{R}_{r,\sigma}(\underline{t}) \equiv \underline{c}_\sigma \cdot \underline{t}' \ . \tag{8}$$

Observe that for $\sigma \approx 0$ then $\mathcal{R}_{r,\sigma}(\underline{t}) \approx \mathcal{R}_r(\underline{t})$. This approach represents the basis for the adaptive design methodology presented in the following section.

3. ADAPTIVE DESIGN

The framework for adaptive design is connected to adaptive filtering, where the design is viewed as a learning process, and the filter parameters are iteratively adapted. In this way, the training goal is to achieve a set of parameters $\underline{a}_m^{(l)}$ and $r_m^{(l)}$, $m = 1, 2, \cdots, N_l$, $l = 1, 2, \cdots, L$, such that some design requirement is met.

A possible approach for adaptive optimization of morphological/rank operations was introduced in Salembier [8]. Yang and Maragos [9] applied the approach in [8] to design min-max classifiers, which represent a special case of our proposed system. Following some of the ideas by Salembier [8], in this paper we propose a steepest descent method to optimally design MRNNs, under the LMS criterion, using the back-propagation algorithm.

The structuring elements $\underline{a}_m^{(l)}$ are used directly in the training equations. However, instead of directly using the ranks $r_m^{(l)}$, indirect real variables $\rho_m^{(l)}$ are properly defined, and used in the adaptation algorithm. The relation between $\rho_m^{(l)}$ and the output $y_m^{(l)}$ is defined later by an appropriate design choice.

In this way, the weight vectors used in the filter design are then defined by

$$\underline{w}_m^{(l)} \equiv \left(\underline{a}_m^{(l)} \ , \ \rho_m^{(l)}\right) \ , \tag{9}$$

and using an objective function based on the LMS criterion, then the general learning algorithm is of the form

$$\begin{array}{c} \underline{w}_m^{(l)}(i+1) = \underline{w}_m^{(l)}(i) + \mu \ e(i) \ \underline{u}_m^{(l)}(i) \ , \\ m = 1, 2, \cdots, N_l \ ; \ l = 1, 2, \cdots, L \ , \end{array} \tag{10}$$

where $e(i) = d(i) - y(i)$ is the difference between the desired output $d(i)$ and the actual filter output $y(i)$ for the training sample i; $\mu > 0$ is the step size, that controls the tradeoff between stability and speed of convergence; and

$$\underline{u}_m^{(l)} = \frac{\partial y}{\partial \underline{w}_m^{(l)}} = \left(\frac{\partial y}{\partial \underline{a}_m^{(l)}} \ , \ \frac{\partial y}{\partial \rho_m^{(l)}} \right) . \tag{11}$$

In addition, each node can be designed individually, leading to a better control during the training process.

If we define the matrices $W^{(l)}$ and $U^{(l)}$ by

$$W^{(l)} = \begin{pmatrix} \underline{w}_1^{(l)} \\ \underline{w}_2^{(l)} \\ \vdots \\ \underline{w}_{N_l}^{(l)} \end{pmatrix} \ , \ U^{(l)} = \begin{pmatrix} \underline{u}_1^{(l)} \\ \underline{u}_2^{(l)} \\ \vdots \\ \underline{u}_{N_l}^{(l)} \end{pmatrix} \ , \tag{12}$$

then, based on the back-propagation algorithm, the general algorithm (10) can be written as

$$\begin{array}{c} W^{(l)}(i+1) = W^{(l)}(i) + \mu \ e(i) \ U^{(l)}(i) \ , \\ l = 1, 2, \cdots, L \ , \end{array} \tag{13}$$

where

$$U^{(l)} = \mathrm{diag}(\underline{\delta}^{(l)}) \cdot \Gamma^{(l)} \ , \tag{14}$$

$$\underline{\delta}^{(l)} = \begin{cases} 1 & , \ l = L \\ \underline{\delta}^{(l+1)} \cdot \Theta^{(l+1)} & , \ \text{otherwise} \end{cases} \tag{15}$$

The matrices $\Gamma^{(l)}$ and $\Theta^{(l)}$ in (14) and (15) are defined by

$$\Gamma^{(l)} = \begin{pmatrix} \underline{\gamma}_1^{(l)} \\ \underline{\gamma}_2^{(l)} \\ \vdots \\ \underline{\gamma}_{N_l}^{(l)} \end{pmatrix} , \ \Theta^{(l)} = \begin{pmatrix} \underline{\theta}_1^{(l)} \\ \underline{\theta}_2^{(l)} \\ \vdots \\ \underline{\theta}_{N_l}^{(l)} \end{pmatrix} , \tag{16}$$

where

$$\underline{\theta}_m^{(l)} = \frac{\partial y_m^{(l)}}{\partial \underline{y}^{(l-1)}} \ , \tag{17}$$

$$\underline{\gamma}_m^{(l)} = \frac{\partial y_m^{(l)}}{\partial \underline{w}_m^{(l)}} = \left(\frac{\partial y_m^{(l)}}{\partial \underline{a}_m^{(l)}} \ , \ \frac{\partial y_m^{(l)}}{\partial \rho_m^{(l)}} \right) \ . \tag{18}$$

Using our design choice, from (1) and (3) we have that $\partial y_m^{(l)} / \partial \underline{y}^{(l-1)} = \partial y_m^{(l)} / \partial \underline{a}_m^{(l)} = \underline{c}_m^{(l)}$, where

$$\underline{c}_m^{(l)} \equiv \frac{Q(y_m^{(l)} \underline{1} - \underline{t}_m^{(l)})}{Q(y_m^{(l)} \underline{1} - \underline{t}_m^{(l)}) \cdot \underline{1}'} \ , \ \underline{t}_m^{(l)} = \underline{y}^{(l-1)} + \underline{a}_m^{(l)} \ . \tag{19}$$

Furthermore, based on heuristic arguments, we can appropriately define $\partial y_m^{(l)} / \partial \rho_m^{(l)} = s_m^{(l)}$. If all the elements of $\underline{t}_m^{(l)} = \underline{y}^{(l-1)} + \underline{a}_m^{(l)}$ are identical, then the rank $r_m^{(l)}$ does not play any role, so that $s_m^{(l)} = 0$ whenever this happens. On the other hand, if only one element of $\underline{t}_m^{(l)}$ is equal to $y_m^{(l)}$, then variations in the rank $r_m^{(l)}$ can drastically modify the output $y_m^{(l)}$; it means that $s_m^{(l)}$ should be maximum in this case. A possible simple choice for $s_m^{(l)}$ is defined as

$$s_m^{(l)} \equiv 1 - \frac{1}{N_{l-1}} Q(y_m^{(l)} \underline{1} - \underline{t}_m^{(l)}) \cdot \underline{1}' \ . \tag{20}$$

Finally, the one-to-one correspondence between $r_m^{(l)}$ and $\rho_m^{(l)}$ is defined [1] by

$$r_m^{(l)} \equiv \left\lfloor N_{l-1} - \frac{N_{l-1} - 1}{1 + \exp(-\rho_m^{(l)})} + 0.5 \right\rfloor \ . \tag{21}$$

For example, if $\rho_m^{(l)} \to -\infty$, then $r_m^{(l)} \to N_{l-1}$, corresponding to a minimum operation; if $\rho_m^{(l)} \to \infty$, then $r_m^{(l)} \to 1$, corresponding to a maximum operation; if $\rho_m^{(l)} = 0$, then $r_m^{(l)} = \left\lfloor \frac{N_{l-1}+1}{2} + 0.5 \right\rfloor$, corresponding to a median operation. Observe that the actual implementation of (19) and (20) is based on (6). The resulting algorithm is then illustrated in the section that follows.

[1] $\lfloor \cdot \rfloor$ denotes the usual truncation operation, so that $\lfloor \cdot + 0.5 \rfloor$ is the usual rounding operation.

4. APPLICATION IN IMAGE PROCESSING

Problems of image noise cancellation using the MRNN system are presented next. We illustrate the training algorithm in the situations of supervised and unsupervised designs. In the supervised design, we know the original noiseless image and use it as reference; hence, the desired signal is the original (noiseless) signal. In the unsepervised design, however, we do not know the original image and only have the noisy image; hence, the desired signal is the input (noisy) signal itself. In the latter case, the filter is employed as a predictor, where the MRNN estimates the original signal.

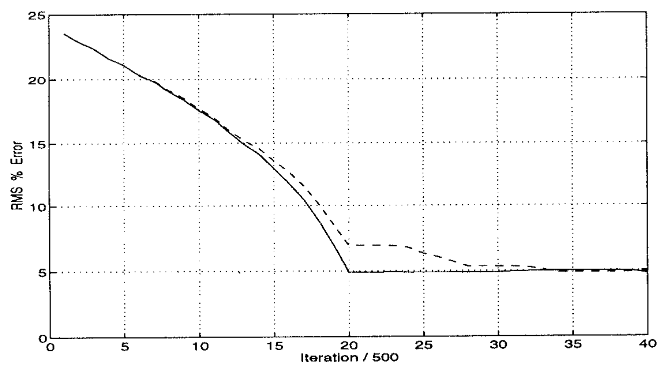
As proposed by Salembier [8], the images will be scanned twice during the training process, following a zig zag path from top to bottom, and then from bottom to top. To define the input vector $\underline{x}$ at each pixel location, a square (3x3) neighborhood centered around it is defined such that $\underline{x}$ is the corresponding square matrix transformed to a vector column-by-column. The structuring elements of the first layer should be interpreted the same way.

The MRNN system used in this example is a two-layer neural network, where the first layer is composed by three nodes – maximum, median and minimum, all with a symmetric 3x3 flat structuring element, and the second layer is a rank operation whose parameters $\underline{a}_1^{(2)}$ and $r_1^{(2)}$ are optimized. Figure 2 shows results of noise cancellation in an image corrupted by 20% *salt and pepper noise* (SNR=–2.07dB). Starting with a flat structuring element $\underline{a}_1^{(2)}(0) = (0,0,0)$ and a median operation $r_1^{(2)}(0) = 2$ ($\rho_1^{(2)}(0) = 0$), the restored images after the training processes are illustrated in Figure 2-(c) (supervised, $\mu = 10^{-3}$) and Figure 2-(d) (unsupervised, $\mu = 10^{-4}$). The final results for the supervised design were $\underline{a}_1^{(2)} = (-5.47, -0.42, 5.01)$ and $r_1^{(2)} = 2$, generating a RMS percentage error of 4.77% (SNR=11.98dB); for the unsupervised design, the final results were $\underline{a}_1^{(2)} = (-0.01, -4.75, 3.39)$ and $r_1^{(2)} = 2$, generating a RMS percentage error of 4.87% (SNR=11.80dB). Although the final structuring elements were different, the resulting SNRs were essentially the same. Figure 2-(e) shows the RMS percentage error during the training process for both cases. Notice that, for the supervised design, the first pass through the image accomplishes the noise cancellation task, since the RMS percentage error stays approximately the same during the second pass.

5. CONCLUSIONS

A general class of neural network based filters was introduced in this paper, and denoted morphological/rank neural networks (MRNNs). Based on simple design choices to estimate required gradients, we then provided an efficient and systematic design procedure using the back-propagation algorithm. The proposed procedure was finally illustrated in problems of noise cancellation, resulting in good performance. This procedure can be easily extended to networks with more than one output. Although a simple filtering task was performed using MRNNs, our preliminary results are encouraging, suggesting the potential of this class of systems and its training algorithm as important tools for nonlinear signal and image processing.

(a) (b) (c) (d)



(e)

Figure 2: (a) Original image. (b) Original image corrupted by 20% salt and pepper noise. (c) Restored image with supervised design. (d) Restored image with unsupervised design. (e) RMS percentage error after every 500 iterations, for step sizes $\mu = 10^{-3}$ (supervised, solid line) and $\mu = 10^{-4}$ (unsupervised, dashed line), with $\sigma = 0.01$ and $q_\sigma(v) = \mathrm{sech}^2(\frac{v}{2\sigma})$ in (6).

ACKNOWLEDGMENTS

The authors thank Dinei A. F. Florêncio, for his helpful suggestion about smoothed rank functions, and discussions during the beginning of this work; and Alexandros Potamianos, for his thorough reading of the manuscripts and suggestions.

6. REFERENCES

[1] S. Marcos, O. Macchi, C. Vignat, G. Dreyfus, L. Personnaz and P. Roussel-Ragot, "A unified framework for gradient algorithms used for filter adaptation and neural network training," *Intl. Journal of Circuit Theory and Applications*, vol. 20, pp. 159–200, 1992.

[2] D. E. Rumelhart, G. E. Hinton and R. J. Willians, "Learning internal representations by error propagation," in D. E. Rumelhart and J. L. McClelland, Eds., *Parallel Distributed Processing: Explorations in the Microstructure of Cognition. Volume 1: Foundations.* MIT Press, Cambridge, MA, 1986. Chapter 8, pp. 318–362.

[3] G. M. Shepherd and R. K. Brayton, "Logic operations are properties of computer-simulated interactions between excitable dentritic spines," *Neuroscience*, vol. 21, no. 1, pp. 151–165, 1987.

[4] J. Serra, *Image Analysis and Mathematical Morphology.* Vol. 1, Academic Press, 1982.

[5] P. Maragos, "A representation theory for morphological image and signal processing," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 11, pp. 586–599, June 1989.

[6] S. S. Wilson, "Morphological Networks," *Proc. of SPIE Visual Commun. and Image Processing IV*, vol. 1199, pp. 483–493, 1989.

[7] J. L. Davidson and F. Hummer, "Morphological neural networks: an introduction with applications," *Circuits, Systems and Signal Processing*, vol. 12, no. 2, pp. 177–210, 1993.

[8] P. Salembier, "Adaptive rank order based filters," *Signal Processing*, vol. 27, pp. 1–25, Apr. 1992.

[9] P. Yang and P. Maragos, "Morphological systems for character image processing and recognition," *Proc. of IEEE ICASSP*, Minneapolis, MN, April 1993.