Neural Networks, Vol. 5, pp. 843–852, 1992
Printed in the USA. All rights reserved.

0893-6080/92 $5.00 + .00

ORIGINAL CONTRIBUTION

Optical Character Recognition by a Neural Network

MICHAEL SABOURIN

Bell-Northern Research

AMAR MITICHE

INRS-Télécommunications

(*Received* 8 *March* 1991; *revised and accepted* 16 *January* 1992)

Abstract—*An optical character recognition (OCR) system, which uses a multilayer perceptron (MLP) neural network classifier, is described. The neural network classifier has the advantage of being fast (highly parallel), easily trainable, and capable of creating arbitrary partitions of the input feature space. Issues in design of the neural network that we examine include the selection of input features, the choice of network learning and momentum parameters, and the selection of training patterns. We also provide a detailed analysis of the learning parameters to provide insight into the MLP, and to suggest a mechanism to automatically tune these parameters. An OCR neural network classifier was trained to recognize characters from a large number of fonts, thereby approaching an omnifont environment. Samples were selected from over* 200 *fonts and* 50 *typical office documents, for a total of* 110,000 *training patterns. In order to evaluate the performance of the MLP classifier, a comparison is made with a high performance dynamic contour warping (DCW) classifier. The base recognition rate on the test set is* 96.7% *for the neural network classifier, compared to* 95.9% *for the DCW classifier.*

Keywords—OCR, Multilayer perceptron, Learning parameters, Momentum, Hierarchical classifier.

1. INTRODUCTION

Optical character recognition (OCR) is used to convert text in printed documents into machine readable ASCII codes for the purpose of storage or further processing (Baird & Thompson, 1987; Cash & Hatamian, 1987; Kahan, Pavlidis, & Baird, 1987; Suen, Berthold, & Mori, 1980; Tappert, 1982). A typical OCR system consists of preprocessing, feature extraction, classification, and context processing (Figure 2). Preprocessing partitions the image into isolated objects, and then segments the document into text and graphic regions. Feature extraction is the process of computing high-level information about individual characters in order to facilitate classification. The classifier then identifies and labels the unknown character. Context processing is used to correct errors in the text by utilizing language statistics, error matrices, and spelling rules.

Currently, an OCR system based on a dynamic contour warping (DCW) classifier (Duffie, 1985), special handlers and postprocessor, is capable of successfully classifying 98% of characters,[1] which is comparable to the best available OCR systems (Sinha, Prasada, Houle, Sabourin, in press). The DCW classification rate (before special handlers and postprocessing) is 95%. Performance shortcomings in the DCW classifier arise from inherent limitations of minimum distance classifiers, such as the convex shape of decision regions. In this implementation, there is also a maximum distance threshold for each model[2] which further constrains the shape of the decision boundaries (Figure 1). For this reason, each class requires several models in order to sufficiently represent character variations in the feature space.

In this work, a multilayer perceptron (MLP) (Lippman, 1987; Minsky & Papert, 1988) will be used as

Acknowledgment: We wish to thank Bell Northern Research for supporting this project and for use of computer resources. This work is also supported in part by NSERC grant #OGP0004234.

Requests for reprints should be sent to Amar Mitiche, INRS-Télécommunications, 3 Place du Commerce, Ile des Soeurs, Verdun, Que., Canada, H3E 1H6.

[1] Results from a test set of 25 multicolumn, complex, multifont documents of normal quality scanned at 200 dots per inch facsimile were 98.0% for the DCW based system, and 98.2% for a tested commercially available system.

[2] A model is a representative of a class. In this case, it is the center of a cluster determined during training.

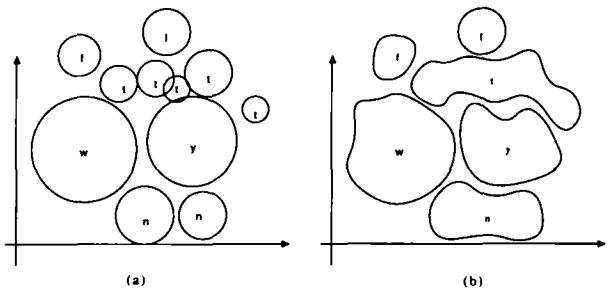

FIGURE 1. Partitioning a multidimensional feature space. (a) Dynamic contour warping classifier. (b) Multilayer perceptron classifier. Note that the convexity requirements of (a) require several models for each character class, while neural networks are capable of partitioning the feature space into nonconvex regions.

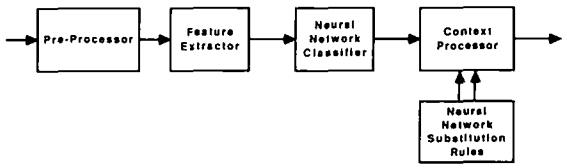

FIGURE 2. Neural network OCR system. A document is scanned at 200 dots per inch. Connected objects are separated from graphics. Features are extracted from characters, and the neural network classifier labels these characters based on their features. Finally, the labelled text is corrected for language context using substitution rules determined during training.

the classifier in an OCR system. The motivation for using an MLP classifier is: (a) arbitrary partitioning of the feature space should produce higher recognition rates by covering the feature space more effectively than a minimum distance classifier, (b) estimated speed of classification, via specialized hardware, is an order of magnitude faster than warping, (c) simple training procedure does not require a separate clustering stage, and (d) training determines the relative significance of various input features (i.e., reduce the dependence on heuristics of the classification procedure).

The objective of this work is to examine the problem of using neural networks for OCR, and in this context, to (a) determine the most suitable set of input features for use in character classification, (b) mathematically analyze learning and momentum parameters in order to select (problem specific) optimal values for these parameters, (c) propose a novel training strategy, (d) determine a methodology concerning the presentation of input patterns to the network, and (e) determine the layout of the most suitable character classifier.

In Section 2, a neural network based OCR system is presented. A review of neural network training using back propagation, as well as a new analysis of learning and momentum parameters, is presented in Section 3. A methodology for training neural networks is also given. Training and test results are presented in Section 4. Conclusions are presented in Section 5.

2. NEURAL NETWORK BASED OCR

An MLP neural network classifier which utilizes character features is presented in Figure 2. A document is scanned via facsimile at 200 dots per inch. Characters are isolated using contour following, which produces a chain code (shape) description and the coordinates of the character. A text/graphics discriminator is used to identify text structures, such as paragraphs, strings, captions, and so on. Isolated characters are processed to extract features. Features are useful in classification because when properly selected they tend to be less sensitive to character variations (due to font differences or scanning effects) than the original bit map of the character. Then, the features are input to the neural network, which computes an ASCII label for the character.[3] The output of the classifier is input to a context post processor (Sinha & Prasada, 1987, 1988) which corrects possible classification errors using substitution rules, English language context, and a spell checker.

In this classification system, the primary feature is the shape of the object, as represented by the tangent field of its contour. The tangent field is derived by smoothing the chain code description and then uniformly sampling the contour to 64 points. Smoothing reduces noise influences, and uniform sampling makes this feature scale invariant. The angle between adjacent samples of the smoothed contour are encoded as a vector, which forms the input to the neural network.

Other character features, such as coreline classification (the relative position of a character on a line), and genus (the number of white spaces in a character), are used to prescreen the data in order to reduce computational time. The coreline feature is computed using algorithms which utilize character statistics. The genus is computed by comparing the subsuming rectangle coordinates of white and black objects. The screening features are encoded using a simple numeric code.

3. ANALYSIS OF LEARNING PARAMETERS

In this section an analysis of the learning parameters of the back propagation algorithm is presented. Training an MLP involves presenting the neural network with sample patterns in order to determine network parameters. The back propagation technique is a learning rule used to adapt the network parameters (weights) based on the error between the target and actual network output. The rate at which learning takes place is controlled by selecting the gain and momentum pa-

[3] The output with the highest activation level is selected as the character class. It may also be of interest to select the second highest output activation as an alternate interpretation.

rameters. No rule for selecting optimal values for these parameters currently exists, although specific values for the gain and momentum terms are sometimes suggested in the literature. A new analysis of these terms (given in the second subsection) will suggest a mechanism for automatically selecting these learning parameters, as well as offering new insight into the training of multilayer perceptrons.

3.1. Some Implications of Back Propagation

Training a neural network involves presenting labelled patterns repeatedly to the network in order to determine interconnection weights. The back propagation algorithm (Lippman, 1987; Rumelhart & McClelland, 1986) is an error driven parameter estimation algorithm. The objective is to minimize the output squared error function by adjusting interconnection weights and node thresholds.

Consider a fully-connected multilayer neural network (Figs. 3 and 4) with input pattern, I, and corresponding output, O, given by

$$O(i) = f\left(\sum_j w_{n-1}(i,j) X_{n-1}(j) \right), \tag{1}$$

where,

$$X_k(i) = f\left(\sum_j w_{k-1}(i,j) X_{k-1}(j) \right), \tag{2}$$

is the output at node i layer k, and f is the cell nonlinearity, chosen to be $f(x) = 1/(1 + e^{\theta - x})$, where θ $\theta_k(i)$ is the node threshold. The nonlinearity plays a significant role: it allows for complex functionality of a network, for otherwise the MLP would be equivalent to a matrix multiplication.[4]

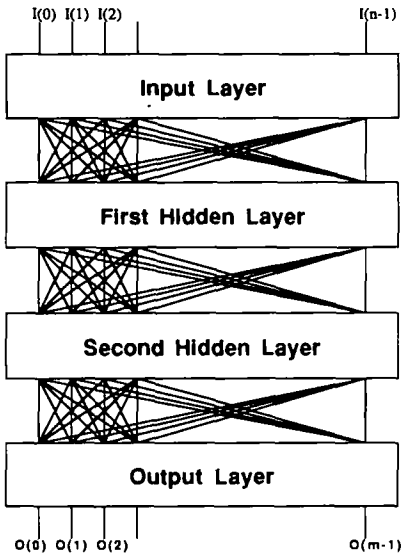

FIGURE 3. Multilayer perceptron architecture. The input pattern, $I = [I(0), I(1), \ldots, I(n-1)]$ excites the network producing output vector $O = [O(0), O(1), \ldots, O(m-1)]$. In pattern classifiers, it is desirable to have each output element, $O(i)$, correspond to a single output class.

The objective of training is to determine the weight parameters, $\{w_{ij}\}$, which minimize a cost functional. The desired *pattern recognition objective* of minimizing the number of misclassifications is substituted with the *mathematical objective* of minimizing the squared error at the network output. For these objectives to be equivalent, it is necessary that as the squared error decreases, the number of classification errors also decreases. In general, this condition is only guaranteed to be satisfied (a) for parameter values very close to the optimal configuration, or (b) for an energy function which contains no local minima.

Interconnection weights are adjusted by adding a correction term proportional to the gradient (slope) of the error function. The error at the output layer is propagated towards using the generalized delta rule (Rumelhart & McClelland, 1986, chapter 8),

$$w_l^{t+1}(i,j) = w_l^t(i,j) + \eta \delta_l^t(j) X_{l-1}^t(i), \tag{3}$$

where η is the gain or learning rate, t is the iteration index, $X_{l-1}^t(i)$ is the output of node i at either layer $l-1$ or the network input, and $\delta_l^t(j)$ is the correction term. For small values of η, it can be shown that back propagation is equivalent to gradient descent methods (Rumelhart & McClelland, 1986, p. 324). The value of the correction term δ_l is given by

$$\delta_l^t(j) = X_l^t(j)(1 - X_l^t(j)) \sum_k \delta_{l+1}^t(k) w_l^t(j,k), \tag{4}$$

for node j in the hidden layer and

$$\delta_{N-1}^t(j) = O(j)(1 - O(j))(T(j) - O(j)), \tag{5}$$

for node j in the output layer, where $T(j)$ is the target output, and N is the number of layers in the network. The node thresholds, θ, are adapted in a similar manner.

In order to improve convergence speed, a momentum term, α, is often added to the weight update equations, i.e.,

$$w^{t+1}(i,j) = w^t(i,j) + \eta \delta^t(j) X^t(i) + \alpha(w^t(i,j) - w^{t-1}(i,j)), \tag{6}$$

where $0 < \alpha < 1$ for stability reasons. To date, no theoretical framework exists for selecting optimal values of α and η. In the literature (for instance, Rumelhart & McClelland, 1986, p. 330), it is often advised to decrease these parameters according to an annealing schedule. More speed up in training may be possible by adding more terms to the regression (e.g., terms of

[4] A multilayer perceptron can be viewed as the concatenation of several single-layer perceptrons. The nonlinear logistic function, $f(x)$, allows complex functionality within the network. If the nonlinearity were replaced by a linearity, multilayer networks would be equivalent to a single layer network, and would only be able to solve linearly separable classification problems.

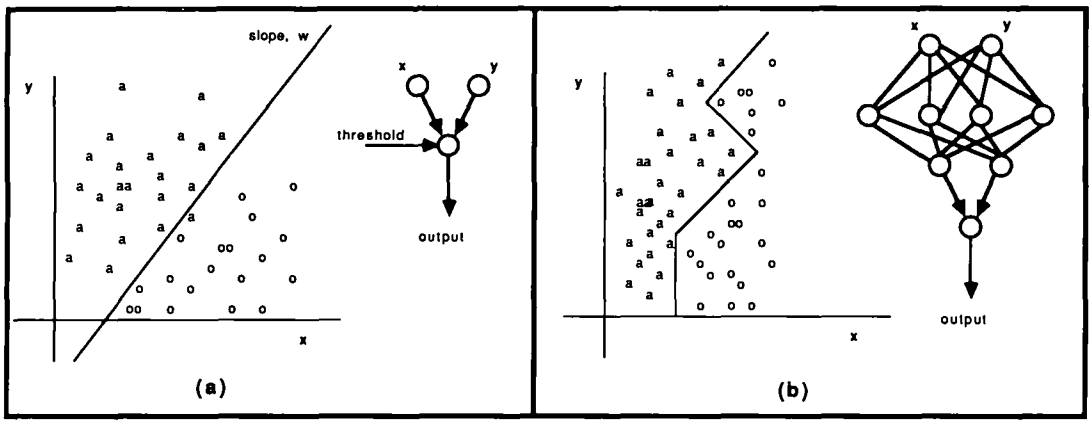

FIGURE 4. Partitioning the feature space. (a) A two input, one output single layer perceptron. Input feature pairs, represented as points in x-y coordinates, are presented to the network. The network learns interconnection weights, w_x and w_y. The line, $w_x x + w_y y = \theta$, partitions the space into two classes. (b) Multilayer perceptron is used to partition the space into nonconvex decision regions. The partition is specified by the union and intersection of single-layer perceptron decision regions.

the form $\beta_k w^{t-k}$), although establishing strategies for estimating β_k terms may prove difficult.

3.1.1. *Discussion.* The function of training a (single layer) perceptron may be viewed as partitioning the multidimensional input pattern (feature) space by manifolds (lines, planes, hyperplanes) determined by equations of the form

$$\sum_i w(i,j)I(i) = \theta(j). \tag{7}$$

For a multilayer perceptron, the boundaries of the decision region are (roughly) determined by taking the union and intersection of equations of the above form (see Figure 4). This idea has significant implications (Lippman, 1987, pp. 13–14). The number of layers and the number of nodes per layer affects the shape of the decision regions (e.g., limits the complexity of the partition). For instance, a necessary condition for nonconvex decision regions is that a network consists of at least two hidden layers.

3.2. Analysis

We now present an analysis of the MLP training parameters (rate and momentum), and a novel mechanism for selecting optimal training parameters.

Many results regarding neural network performance, and the choice of training parameters η and α, have been presented (Roberts & Agarwal, 1990; Rumelhart & McClelland, 1986, pp. 330). The selection of these parameters tend to be adhoc, and no theory for selecting optimal values exists. The objective of the following analysis is to relate the choice of training parameters to characteristics of the training data, and to suggest a mechanism for selecting optimal values.

Recall the back propagation learning algorithm with momentum:

$$w^{t+1}(i,j) = w^t(i,j) + \eta\delta^t(j)X^t(i) + \alpha(w^t(i,j) - w^{t-1}(i,j)). \tag{8}$$

Consider the change of variables $v^t = w^t(i,j) - w^{t-1}(i,j)$, and $\mu^t = \delta^t(j)X^t(i)$. Note that μ^t is the error term and v^t is the correction term at iteration t. Equation (8) is rewritten as

$$v^{t+1} = \eta\mu^t + \alpha v^t. \tag{9}$$

The discrete Fourier transform[5] of eqn (9) is

$$zV(z) = \eta\mu(z) + \alpha V(z), \tag{10}$$

or

$$V(z) = \frac{\eta}{z - \alpha}\mu(z), \tag{11}$$

where $z = e^{i\phi}$ is the frequency parameter, and ϕ is the discrete frequency.

The relationship between the error term, μ, and the correction term, v, is given by the transfer function

$$H(e^{i\phi}) = \left|\frac{V(e^{i\phi})}{\mu(e^{i\phi})}\right|^2 = \frac{\eta^2}{1 - 2\alpha\cos\phi + \alpha^2}, \tag{12}$$

where $\phi \in [-\pi, \pi]$ is the discrete frequency. For large values of α, the transfer function is a lowpass filter. For small values of α, the transfer function is allpass (see Figure 5).

Training an MLP would be rather simple, except for the fact that the error term, μ, has a time varying frequency characteristic. In practice, μ generally decreases in magnitude as training progresses. This is due to the fact that as training progresses the weight and threshold parameters better represent the process being

[5] For a signal $x(t)$, we define the z-transform as $X(z) = \sum x(n)z^{-n}$. Set $z = e^{i\phi}$ to get the Fourier transform. Note that the z-transform of $x(t+1)$ is $zX(z)$.

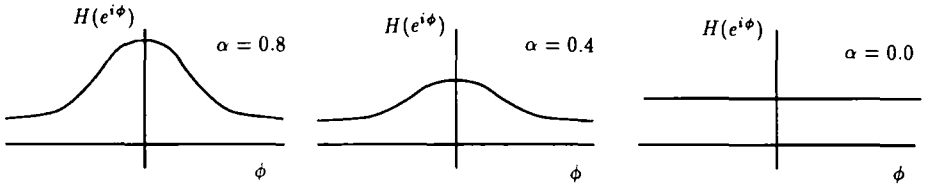

FIGURE 5. Transfer function (gain) between error and correction terms for various values of momentum, α.

modelled; consequently the modelling error, μ, will decrease. As well, the predictable component of the error term will also decrease. Ultimately, if the training is successful, the error will become decorrelated (Figure 6).

The error term, μ, is a nonlinear function of the (input) training data. For a single layered perceptron,

$$\begin{aligned}\mu &= \delta(j)X(i)\\ &= O(j)(1 - O(j))(T(j) - O(j))I(i),\end{aligned} \tag{13}$$

where $O(i) = f(\sum_{j=1}^{N_I} w(i, j)I(j))$. In other words,

$$\begin{aligned}\mu &= g(O(j))I(i)\\ &= g(I(0), \ldots, I(N_I))I(i),\end{aligned} \tag{14}$$

where $g = O(1 - O)(T - O)$ is a nonlinear function. The explicit analysis of this system is rather complicated, even if we assume that the activation function, f, is linear.

Although we are not able to explicitly solve for optimal learning parameters, we have the ability to analyse the error terms explicitly. It is possible, for instance, to determine the power spectrum of μ by taking the discrete Fourier transform of a large enough sample. Then, matching this data with the appropriate transfer function, we can compute the optimal momentum coefficient, α, for that node (see Figure 7). This is analogous to a matched filter approach (Haykin, 1978): the optimal transfer of information through a system is achieved when the frequency characteristics of the signal and the filter are matched.

3.2.1. *Example.* Consider an MLP whose output error characteristic is lowpass. Initially, training should only respond to "coarse" differences in the data, and so a large value of momentum is initially selected. As training progresses, the error spectrum flattens out, and the significance of high frequency components increases. Hence, a smaller term of α is required. Training first concentrates on learning coarse differences in the data (low frequency) and later concentrates on finer differences in the data (high frequency).

3.2.2. *Discussion.* We have applied this technique on simple networks and have observed the benefits. Similar benefits are achieved using a schedule of decreasing the momentum term as the squared output error decreases.

The learning rate, η, is the gain term in eqn (12). In terms of the frequency characteristics, increasing this term corresponds to an amplification of the correction term. A large value of η will cause a large correction to terms w. This would suggest choosing as large a value of η that is stable, except that (a) back propagation loses its correspondence to gradient descent for large η, (2) large gain terms on gradient descent methods can lead to overshooting the global minimum (much like an underdamped linear system), and (c) the dependency of the error term on the gain is nonlinear, and a large value of gain generally results in unstable training.

4. EXPERIMENTS IN OPTICAL CHARACTER RECOGNITION

A neural network was designed to classify characters based on shape information. The hierarchical neural network in Figure 8 was used.[6] Based on the characters' screening features (computed via software), a specialized shape subnetwork is selected. The tangents excite this network, producing an ASCII label.

A hierarchical classifier architecture was selected in order to facilitate recognition, produce higher classification rates, speed up classification, and minimize the number of confusion classes. Experience shows that it is simpler to train several small classifiers than to train one large classifier. This is due to the simpler partitioning required for classifiers distinguishing a few classes. Also, several characters (cC, pP, oO, il . . .) have similar shapes but different screening features, making them difficult to distinguish using only shape information.

4.1. Selection of Training Samples

In our experiments, we have observed the following: (a) It is better to present the training samples in a random order than to present the data in a structured order. This ensures the maximum excitation of the nodes in a network by reducing the correlation in the training samples, which can have the effect of impeding (counteracting) training. (b) It is better to present an equal number of training patterns from each class than an unequal number. A disproportionate number of training patterns from one class will pull the decision boundaries in a manner emphasizing the most frequent

[6] For comparison purposes, a hierarchical classifier was also used in the DCW classifier.

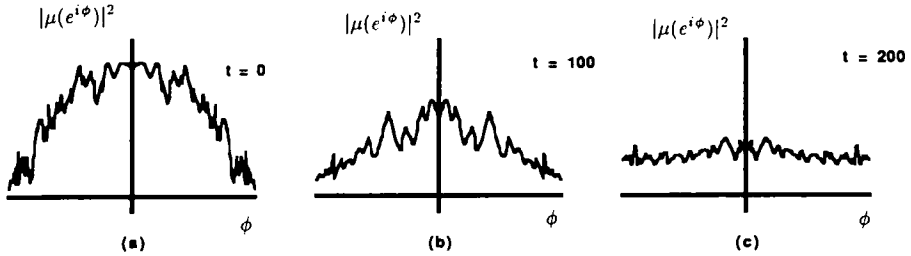

FIGURE 6. Error decorrelation. As training progresses, the network parameters become tuned in such a manner as to reduce the predictable portion of the error. Training reduces the amplitude of the error, and the error distribution becomes more uniform.

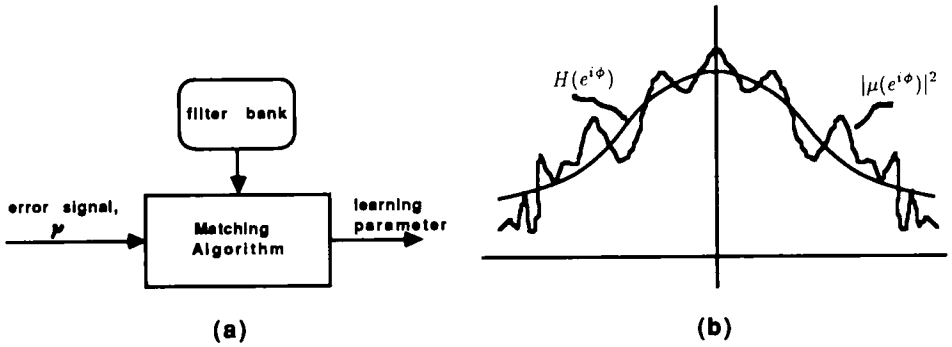

FIGURE 7. Determining momentum term via matching algorithm. The maximum transfer of information through a filter is achieved by matching the filter frequency characteristics with that of the error signal. (a) The spectral characteristics of the propagated error, v, is used to select the most similar filter from a bank, and the corresponding learning and momentum terms. (b) The selected filter is superimposed over the error signal spectrum.

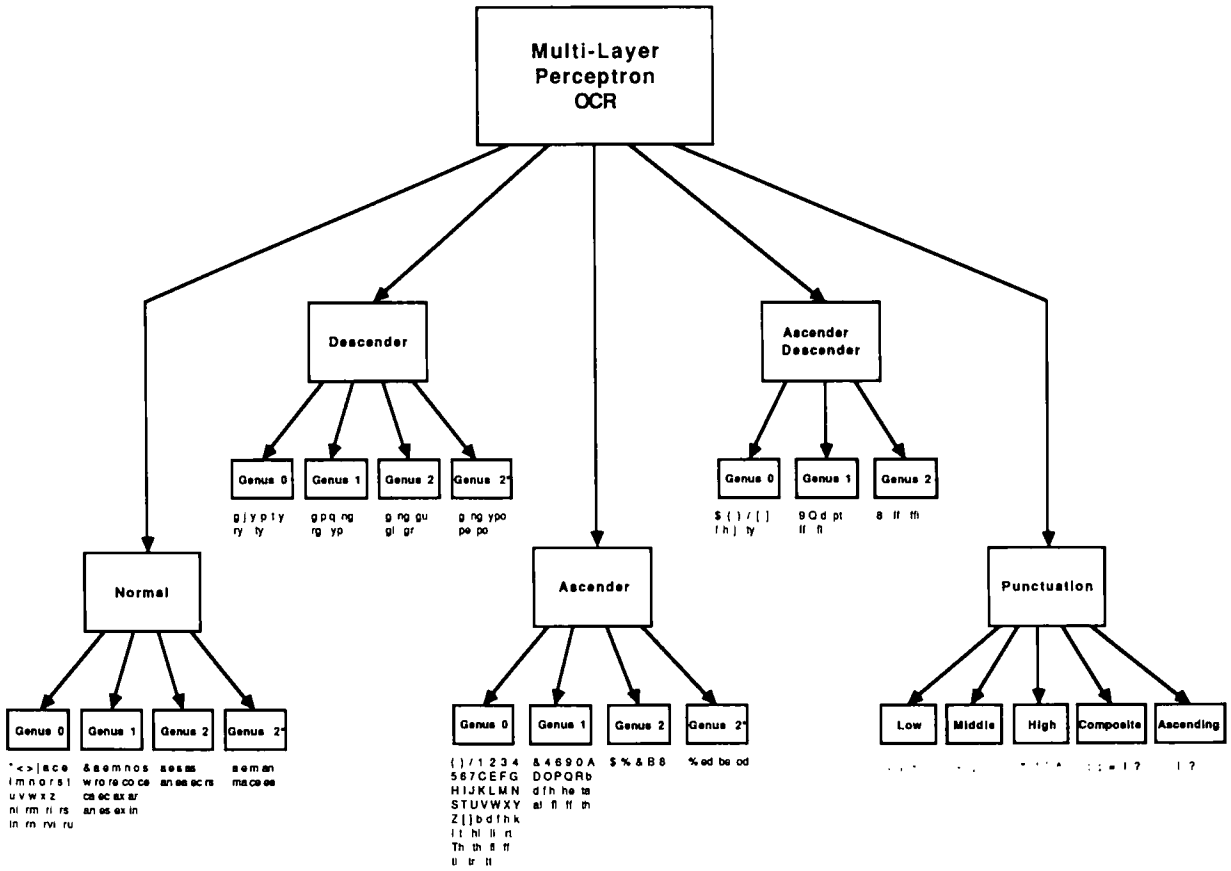

FIGURE 8. Hierarchical neural network architecture. The neural network classifier is divided into 20 subnetworks, each specialized to recognizing a portion of the ASCII character set. The division is based on reliable screening features: the coreline class (position of a character with respect to baselines) and genus (the number of white spaces in a character).

TABLE 1
Preliminary Training Results for Hierarchical Multilayer Perceptron. Column 1 Lists the Network Name (Screening Features), Column 2 is the Constituent Character Classes, Column 3 is the Network Node Configuration, Column 4 is the Base Recognition Rate, Column 5 is the Modified Recognition Rate (Adjusted for Similar Shape Misclassifications), and Column 6 Lists Confusion Classes (Similar Shaped Objects)

		Training Set			
Network	Entries	Configuration	Base Rate	Modified Rate	Confusion Classes
Ascender Genus 0	CEFGHIJKLMNSTUV WXYZfhklt123457	64:96:60:30	94.55	97.64	Il1, S5 kK
Ascender Genus 1	ADOPQRbd0469	64:128:36:12	98.01	98.81	0O
Ascender Genus 2	B8	64:64:32:2	99.04	99.04	
Normal Genus 0	cimnrstuvwxz	64:64:36:12	99.14	99.14	
Normal Genus 1	aeo	64:64:23:3	99.23	99.23	
Descender Genus 0	jy	64:64:32:2	100.0	100.0	
Descender Genus 1	gpq	64:64:30:3	99.07	99.07	
Punctuation	'"- .,_!?:;=	32:76:26:13	100.0	100.0	
Total	ASCII		97.57	99.08	

classes, crippling the training of the minority class. Presenting a class in a nonuniform manner generally increases the amount of training needed, in some cases to infinity. (c) It is best to present as many different samples as possible, so that the training patterns represent the pattern variations within a class. Insufficient variation in training samples will produce boundaries that will not separate the feature space. Sufficient variation will produce boundaries which better partition the feature space, minimizing the probability of a rare variation in the input data pattern being misclassified.

4.2. Preliminary Training Results

Based on the preceding comments, it was decided to create a character database by selecting over 200 different samples for each alpha-numeric class, for a total of 12,000 training patterns. Each sample was selected from a different font in order to maximize the variation in patterns within a class. The font styles were divided between Roman (serif) and Lineale (sans serif), and several examples of italic, and bold fonts were selected. This large number of fonts approaches an omnifont OCR environment. During training, the patterns were presented to the neural network at random to ensure maximum variation.

The selection mechanism for learning parameters described in the previous section was not utilized due to lack of available signal processing resources required. Instead, the momentum and gain parameters were both initially set to 0.5 and slowly annealed. As such, the networks were quickly trained (after a few epochs) to a performance of approximately 90%, with the remainder of training taking hundreds of epochs before convergence.[7] This is due to the size and variation of shapes within the training set, and the fine tuning required in finding the optimal manifolds which form the partition between character classes. If, after a thousand epochs, the network had converged to a performance less than 95%, then a new network was created with a larger number of nodes (i.e., the current network did not have the capacity to partitioning the pattern space).

A table of preliminary training results is given in Table 1. The overall recognition rate on the training set is 99%. Each leaf of the hierarchical classifier (see Figure 8) corresponds to a specialist neural network. Note that for the preliminary training set, eight screening possibilities occur. The specialist networks are listed along with their screening features, character classes, network configurations (number of neurons in each layer), recognition rate during training, and a modified recognition rate which excludes confusion errors between identical shapes (O0, Il1, kK, etc.). For example, the network "Descender Genus 1" is used to recognize characters "p," "q," and "g." This network consists of 64 input nodes, 64 first layer nodes, 30 second layer nodes, and 3 output nodes for a total of 161 neurons, 6,106 interconnection weights, and 97 node thresholds. This network was trained to a recognition rate of 99.1%, with corresponding mean squared output error of 0.04 units.

Along with the optimal network weights and thresholds, training also produces an error matrix. Error matrices are useful in developing substitution rules in order to correct classification errors (post processor). A sample error matrix is given in the Appendix.

4.2.1. *Discussion.* Theoretically, it is possible to design a neural network which would recognize 100% of the training samples. This is not always possible in practice and is not always desirable! Nonperfect training occurs due to (a) insufficient number of nodes in network (suboptimal configuration), (b) nonseparable classes,

[7] Using a Motorola 88100 based RISC computer, the amount of time required for training the entire classifier is approximately 2 CPU weeks.

TABLE 2
Test Results: Multilayer Perceptron Versus Dynamic Contour Warping. The Initial MLP Classifier was Trained Using Samples from 200 Fonts, While the Final MLP Classifier was Trained Using Samples from an Expanded Set of 100,000 Samples (Section 4.4)

	Test Set		
	Recognition Rates		
Document	Warping	MLP (initial)	MLP (final)
Test 1	92.2	91.7	94.2
Test 2	98.5	98.5	98.8
Test 3	97.8	96.0	97.2
Test 4	97.5	95.0	96.5
Test 5	93.3	95.0	95.5
Average	95.9	95.2	96.7

(c) labelling errors in the training set, and (d) insufficient training period.

Perfect training may be possible if computation constraints were nonexistent (i.e., allow for a large number of nodes and a long training period). However, since we train on representatives from each class (and not the totality), then one function of the network is to generalize from the data (interpolation). A large network may perform perfectly on the training set but may not be able to interpolate as well as a smaller network.

4.3. Preliminary Test Results

The neural network classifier was tested on a set of five typical documents of normal quality. Note that the post processor was disabled in order to compare directly the performance of the classification techniques. The test documents were multifont, and were not used in the training of either OCR system. The test results are presented in Table 2. The (initial) recognition rate was 95.2% for neural networks as compared to 95.9% for DCW.

The preliminary recognition performance of the neural network was less than that of the DCW classifier. This is not unexpected since the DCW classifier was trained using a larger number of examples per class (total of 100,000 characters). As well, several mutilated characters (such as blurred "h," "e," or broken "b," "h") are included as DCW templates, along with common ligatures (such as *ff*, *fi*, *ffi*, and also *rt*, *ru*, *tt*, etc.).

The neural network, in general, had difficulty in classifying the letter "t." This difficulty probably arises from the similarity in the features (tangents) between the letter "t" and "l" and "f." In the classifier presented here, all classes were treated equally. Apparently, it is necessary to provide special training to help distinguish the letter "t" when using a shape classifier. The DCW

TABLE 3
Training Results for Hierarchical Multilayer Perceptron

	Training Set					
Network	Entries	# Samples	Configuration	Base Rate	Modified Rate	Confusions
Ascender Genus 0	( )/1234567?CEFGHIJ KLMNSTUVWXYZ[ ]bdfhiklt hi li rt Th th fi ff ti tr tt	9054	64:128:100:50	95.05	98.35	Il1 kK 2Z 5S
Ascender Genus 1	&0469ADOPQRbdfh he ta al fl ff th	2640	64:128:128:19	96.51	97.92	OO
Ascender Genus 2	$%&B8	447	64:64:25:5	97.99	97.99	
Ascender Genus 2*	% ed be od	30	64:40:20:4	100.0	100.0	
Normal Genus 0	*〈 〉aceimnorstuvwxz ni rm ri rs in rn rvi ru	3694	64:128:81:27	97.13	97.13	
Normal Genus 1	&aemnosw ro re co ce ca ec ax ar an es ex in	1597	64:64:40:20	99.23	99.23	
Normal Genus 2	aes as an ea ec rs	1113	64:32:16:8	99.19	99.19	
Normal Genus 2*	aem an ma ce ee	215	64:64:35:7	100.0	100.0	
Descender Genus 0	gjpty ry ty	566	64:64:28:7	99.29	99.29	
Descender Genus 1	gpq ng rg yp	700	64:48:24:6	99.42	99.42	
Descender Genus 2	g ng gu gi gr	19	64:40:20:5	100.0	100.0	
Descender Genus 2*	g ng ypo pe po	50	64:40:20:5	100.0	100.0	
Ascender-Descender Genus 0	$( )/[ ]fhj ty	272	64:64:44:11	100.0	100.0	
Ascender-Descender Genus 1	9Qd pt ff fl	37	64:48:24:6	100.0	100.0	
Ascender-Descender Genus 2	8 ff ffi	4	64:36:18:3	100.0	100.0	
Low punctuation	. , -	72	64:64:30:3	100.0	100.0	
Middle punctuation	- ,	27	64:48:24:3	100.0	100.0	
High punctuation	‘’ ‘ ’ ?	59	64:64:32:5	94.92	94.92	
Composite punctuation	:;=?!	102	64:64:40:5	98.03	98.03	
Ascending punctuation	?!	14	64:36:12:2	100.0	100.0	
Total	ASCII	20712		97.01	98.63	

classifier overcame this by including many models for this shape. Further training of the NN classifier will clearly require a disproportionately large number of samples for the letter "t."

The MLP does not learn "neighborhoods" implicitly. Training must provide sufficient samples to the network to determine correctly the decision boundaries. Without appropriate training, the MLP does not always generalize well. This has been observed, for example, in (Robillard, 1990).

4.4. Enhancements

The hierarchical OCR neural network was **retrained** using 100,000 samples from a set of 50 documents of varying quality. Ligatures and topological mutilations were included as training patterns. Each subnetwork was expanded in size (number of output classes) to include new ligature and mutilation classes. As well, several new screening possibilities arose which were not present in the preliminary training: a total of 20 specialist networks are required. Note that a genus value of "2*" will refer to a character with two white spaces which are horizontally aligned.

The training results are presented in Table 3. The hierarchical network had a combined training rate of 98.63%. Although the recognition rate on the training set is less than the initial training (Section 4.2), the training data is more representative of the class variations which arise from document noise and other effects.

On the test set (Table 2), a recognition rate of 96.7% was observed. This is significantly better than the conventional classifier based on dynamic contour warping. Note that the integrity of the test set was **not** compromised during the enhanced training.

5. CONCLUSIONS

This paper had two main objectives: first, to present a new analysis of learning parameters of a multilayer perceptron, and second, to develop an OCR system using a MLP classifier.

The analysis of MLP learning parameters shows us that selection of optimal values of the momentum term are intimately related to the frequency characteristic of the error term in the back propagation algorithm. As training progresses, the frequency characteristic changes, and hence so should the value of the momentum term. In fact, training the network has the effect of decorrelating the error term, and hence making the output error spectrum more uniform.

An OCR system was developed using a hierarchical MLP structure. The classifier uses shape information (tangent field) to classify characters. On a test set of five unseen documents, the MLP classifier was capable of recognizing 96.7% of characters, compared to 95.9% for an OCR system based on contour warping. This performance is considered significant.

Directions for future research include investigating (a) the use of moments (Cash & Hatamian, 1987) to provide additional feature information, (b) associative memories (Kohonen feature maps (Kohonen, 1988)) for clustering, and (c) the applicability of these methods to handprinted OCR.

REFERENCES

Baird, H. S., & Thompson, K. (1987). Reading chess. *Proceedings of the IEEE Computer Society Workshop on Computer Vision*, 277–282.

Cash, G. L., & Hatamian, M. (1987). Optical character recognition by the method of moments. *Computer Vision, Graphics and Image Processing*, **39**, 291–310.

Duffie, P. K. (1985). *Contour elastic matching for omni-font character recognition*. unpublished M.Eng. Thesis, McGill University, Montreal, Canada.

Haykin, S. (1978). *Communication systems*. New York: John Wiley & Sons.

Kahan, S., Pavlidis, T., & Baird, H. S. (1987). On the recognition of printed characters of any font and size. *IEEE Pattern Analysis and Machine Intelligence*, **9**, 274–287.

Kohonen, T. (1988). Self-organization and associative memory (2nd ed.). Berlin: Springer-Verlag.

Lippman, R. P. (1987). An introduction to computing with neural networks. *IEEE Acoustics, Speech, and Signal Processing Magazine*, 4–22.

Minsky, M. L., & Papert, S. A. (1988). *Perceptrons. An introduction to computational geometry*. Cambridge, MA: MIT Press.

Roberts, G., & Agarwal, V. (1990). Course notes: VLSI methods for neural networks. Department Electrical Engineering, McGill University, Montreal, Canada.

Robillard, S. (1990). Contribution à l'étude des mémoires associatives. unpublished Masters Thesis, INRS-Telecommunications, Montreal, Quebec.

Rumelhart, D. E., & McClelland, J. L. (1986). *Parallel distributed processing* (vol I and II). Cambridge, MA: MIT Press.

Sinha, R. M. K., & Prasada, B. (May 1987). *An integrated omni-font text recognition system* (Rapport Tech. INRS-Telecommunications, No. 87-21).

Sinha, R. M. K., & Prasada, B. (1988). Visual text recognition through contextual processing. *Pattern Recognition*, **21**, 463–479.

Sinha, R. M. K., Prasada, B., Houle, G., & Sabourin, M. (in press). Hybrid contextual text recognition with string matching. *IEEE Pattern Analysis and Machine Intelligence*.

Suen, C. Y., Berthod, M., & Mori, S. (1980). Automatic recognition of handprinted characters—The state of the art. *Proceedings of the IEEE*, **68**, 469–487.

Tappert, C. C. (1982). Cursive script recognition by elastic matching. *IBM Journal of Research*, **10**, 765–771.

APPENDIX

Training Error Matrix

In this Appendix, a sample error matrix (Ascender, Genus 1) of the hierarchical neural network classifier is presented (Table 4). Error matrices are useful in determining substitution rules to be used during context postprocessing.

To read information from this table, note that the classes indicated on the top row represent the output of the neural network classifier, while the classes indicated in the first column represent the truth

TABLE 4
Error Matrix for Specialist Classifier Ascender Genus 1. The "Truth Label" Refer to the Label Assigned to the Training Pattern, While the "OCR Label" is the Machine Generated Category Assigned to the Pattern. As an Example, Consider the Entries of Truth Label "A": 208 Patterns are Correctly Labelled as "A", While 10 Patterns were Labelled as "R," and 1 Pattern was Labelled as "b."

	OCR Label											
Truth Label	0	4	6	9	A	D	O	P	Q	R	b	d
0	163				1		22			1		
4		3		1								
6			176		1				1		1	
9		1		179		1						
A					208					10	1	
D				1		199				2		
O	19	1					170					
P								204		1		
Q	4		2				1		184		1	
R				1	3					201		
b			1			1	1				218	
d				1		2			1			205

labels. For example, the entry intersected by the row "A" and the column "R" has a value, 10. This is the number of characters classified as "R" which are in truth "A."

Fortunately, many classification errors are resolvable using a post-processor, which has special "number" handlers, spell checkers, and string matching mechanisms.