Signal Processing for Multimedia
J.S. Byrnes (Ed.)
IOS Press, 1999

On Applying Fast and Efficient Methods in Pattern Recognition

Basil G. Mertzios and Dimitris A. Karras

Automatic Control Systems Laboratory
Department of Electrical and Computer Engineering
Democritus University of Thrace
67100 Xanthi, Hellas

ABSTRACT. This chapter presents a new approach in pattern recognition which relies on using fast and efficient techniques in the implementation of basic pattern recognition procedures. More specifically, concerning the pattern preprocessing and feature extraction stage, a novel method is employed in the design of the relevant algorithms. Namely, the Image Block Representation methodology is used in the construction of efficient algorithms for quantitatively analyzing and processing patterns, like 2-D moment computation for improved feature extraction. It is based on exploiting the properties of simple operators applied to pattern representation and manipulation. Second, concerning the pattern classification stage, a novel methodology is presented for designing improved pattern classifiers based on the concept of effectively training feedforward neural networks using constrained optimization techniques. The two methods, developed in the present research effort as new architectural principles for designing improved pattern recognition systems, are evaluated through extensive experimentation demonstrating reliability and enhanced performance in several real world problems.

1. Introduction

Many pattern recognition methodologies and design techniques have been developed over the years and new approaches continue to emerge. These tools apply to the different stages of a pattern recognition system with the goal to improve its performance. Therefore, a multitude of rival methodologies exists in the literature for designing feature extraction and selection processes as well as for designing the classification stage of such systems [1]. The primary concern in the application of these methodologies is the requirement for enhanced pattern recognition performance. Computational complexity issues are not usually taken into account.

The goal of this chapter is to demonstrate the importance of two new concepts as design principles in the development of fast and efficient pattern recognition systems. The first is the Image Block Representation approach, which can be involved in the feature extraction stage of such a system for the fast implementation of pattern preprocessing techniques. The second is a Constrained Optimization based approach for efficiently training feedforward neural networks of the MultiLayer Perceptron (MLP) type, which can be involved in the classification stage of such a pattern recognition system. These principles lead to effective pattern recognition procedures of reduced order in terms of computational complexity. Both concepts are involved in the design of an improved Optical Character Recognition (OCR) system. In the next paragraphs a brief introduction of these principles is presented. As we have mentioned the first one deals with the problem of effective image representation. The most common image representation format is a two-dimensional (2-D) array, each element of which has

the brightness value of the corresponding pixel. For a binary image these values are 0 or 1. In a serial machine, only one pixel is to be processed at a time, by using the 2-D array representation. Many research efforts have considered the problem of selecting an image representation suitable for concurrent processing in a serial machine. The need for such approaches arises from the fact that an image contains a great amount of information, thus rendering the processing a difficult and slow task. Existing approaches to image representation aim to provide machine perception of images in pieces larger than a pixel and are separated in two categories: boundary based methods and region based methods. Such approaches include quadtree representations [2], chain code representations [3], contour control point models [4], autoregressive models [5], the interval coding representation [6] and block implementation techniques [7–9]. One common objective of the above methods is the representation of an image in a more suitable form for a specific operation. This chapter presents a new advantageous representation for binary images, which is called Image Block Representation (IBR) and constitutes an efficient tool for image processing and analysis techniques [10, 11]. Using the block represented binary images, real-time computation of 2-D statistical moments is achieved through analytical formulae. The computational complexity of the proposed technique is O(L), where (L-1, L-1) is the order of the 2-D moments to be computed. Various sets of 2-D statistical moments constitute a well-known image analysis and pattern recognition tool [12–21]. Therefore, IBR can be employed in the feature extraction stage of a pattern recognition system for the fast calculation of the 2-D moments and other important features of an input pattern like the critical points, as explained in a following section. Critical point and statistical moment computations for 2-D patterns using IBR have been precisely involved as the feature extraction procedures of an improved OCR system presented in a next section.

Concerning the previously mentioned second design principle, which is related to the classification stage of a pattern recognition system, a constrained optimization procedure is involved for fast and accurate training of MLPs. MLPs have been hailed in recent years for their potential and ability to provide efficient solutions to classification, function approximation, control and optimization problems. In this context, the application of these networks to interesting technical problems—often requiring processing of numerous and diverse data—places stringent requirements on learning algorithms in terms of speed, scalability properties and generalization capabilities. In order to meet these requirements, it will probably be necessary to incorporate in an optimal way various kinds of knowledge about efficient neural network learning into our training algorithms. This knowledge can be problem specific, or can represent more general statements about the formation of proper internal representations, the efficient pruning of weights or the optimal handling of learning acceleration techniques, such as the use of momentum. In terms of these ideas, while the pioneering Back Propagation (BP) algorithm [42, 43] for training MLPs is a theoretically well established, simple and mathematically elegant method it lacks the capability of manipulating different kinds of knowledge effectively. Algorithms which have been proposed so far with the aim of exploiting further knowledge about learning in MLP(e.g. enhancing the role of internal representations during learning) [52–54] lack the elegance and clarity in objectives and methodology of BP. With these considerations in mind and following a program of constructing learning algorithms which preserve the mathematical rigor and clarity in objectives of BP and incorporate different forms of knowledge in MLP in the form of well defined constrained optimization tasks, it has been illustrated that they provide in effect improved learning speed and scalability properties for MLPs [31, 32, 38–40]. In the present chapter we also demonstrate the enhanced generalization capabilities which can be acquired by MLPs trained within the proposed constrained optimization framework.

The chapter is organized as follows. In the next section the IBR principles are illustrated and the fast computation of 2-D statistical moments along with the computation of the critical points of a 2-D pattern is demonstrated. Next, the MLP training procedure involving a

constrained optimization framework is presented. In the experimental section of the chapter, the IBR methodology is evaluated first and, afterwards, the performance of an OCR system incorporating both suggested design principles is investigated. The promising results obtained by utilizing the proposed methodologies as well as their prospects are discussed in the conclusion.

2. Image Block Representation (IBR)

A bilevel digital image is represented by a binary 2-D array. Without loss of generality, we suppose that the object pixels are assigned to level 1 and the background pixels to level 0. Due to this representation there are rectangular areas of object value 1 in each image. These rectangles, which are called blocks in the terminology of this chapter, have their edges parallel to the image axes and contain an integer number of image pixels. At the extreme case, the minimum rectangular area of the image is one pixel.

Consider a set that contains as members all the non-overlapping blocks of a specific binary image, in such a way that no other block can be extracted from the image (or equivalently each pixel with object level belongs to only one block). It is always feasible to represent a binary image with a set of all the non-overlapping blocks with object level, and this representation is called Image Block Representation (IBR). According to the above discussion, two useful definitions concerning IBR are formulated:

- *Definition 1*. Block is called a rectangular area of the image, with edges parallel to the image axes, that contains pixels of the same value.
- *Definition 2*. A binary image is called block represented if it is represented as a set of blocks with object level, and if each pixel of the image with object value belongs to one and only one block.

According to Definitions 1 and 2, it is concluded that the IBR is an information lossless representation. Given a specific binary image, different sets of different blocks can be formed. Actually, the non-unique block representation does not have any implications on the implementation of any operation on a block represented image. The IBR concept leads to a simple and fast algorithm, which requires just one pass of the image and a simple bookkeeping process. In fact, considering a binary image $f(x, y), x = 0, 1, \ldots, N_1 - 1, y = 0, 1, ..., N_2 - 1$, the block extraction process requires a pass from each line y of the image. In this pass all object level intervals are extracted and compared with the previous extracted blocks. In the following, an IBR algorithm is given.

Algorithm 1: Image Block Representation.

1. Consider each line y of the image f and find the object level intervals in line y.
2. Compare intervals and blocks that have pixels in line y-1.
3. If an interval does not match with any block, this is the beginning of a new block.
4. If a block matches with an interval, the end of the block is in the line y.

As a result of the application of the above algorithm, we obtain a set of all the rectangular areas with level 1 that form the object. A block represented image is denoted as:

$$f(x, y) = (b_i : i = 0, 1, \ldots, k - 1) \tag{1}$$

where k is the number of the blocks. Each block is described by four integers, the coordinates of the upper left and lower right corner in vertical and horizontal axes. The block extraction process is implemented easily with low computational complexity, since it is a pixel checking process without numerical operations. Fig. 1, illustrates the blocks that represent an image of the character B. The optimum representation is characterized by the minimum possible number of blocks. Different IBR algorithms, which may result in smaller number of blocks at the cost of the increased processing time, may be implemented. Specifically, the algorithm

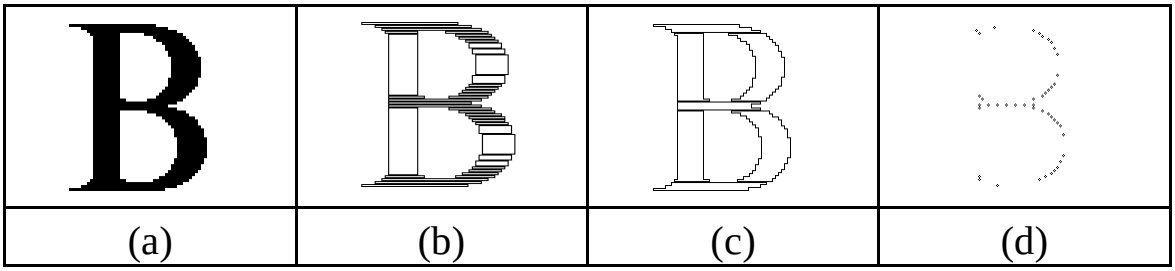

FIGURE 1. (a) Image of the character B. (b) The extracted blocks. (c) The groups of blocks. (d) The critical points.

for finding the maximal empty rectangle [28–30] in an area may be applied recursively: at each stage the points of this rectangle are labelled as background points, in order to apply the algorithm recursively to the remaining points. The time complexity of this method is O(n log n), where n is the number of the points of the area at each recursion. Therefore, the selection of the optimum representation implies an additional computational cost, which may compensate the achieved savings due to the minimum number of blocks.

2.1. *Computation of Moments Based on IBR*

In pattern recognition applications, a small set of lower order moments is used to discriminate among different patterns. The most common moments are the geometrical moments, the central moments, the normalized central moments and the moment invariants [18, 19]. Other sets of moments are the Zernike moments and the Legendre moments (which are based on the theory of orthogonal polynomials) [20, 22], and the complex moments [21]. One main difficulty concerning the use of moments as features in image analysis applications is the implied high computational time. A number of approaches that reduce the computational time concerning calculation of moments have appeared [4], [24–26]. In [24–26], the problem has been reduced from 2-D to 1-D using Green's theorem; this approach reduces the complexity from $O(N^2)$ to O(N), since only the boundary pixels are considered and the length P of the boundary is linearly related to $\sqrt{A}$, where A is the object area. In [4] control point models based on least-square normalized B-splines are used for the representation of the object boundary, where the complexity of the moments computation is related to the shape model order and independent of the scale. The computational cost of this method is comparable to the cost of the proposed method but, mainly due to the deviations of the boundary representation model, the moment values are significantly affected. In [27] the computation formula of each central moment has been considered as an impulse response of a filter, which is then transformed to the z-domain and the transfer function of the corresponding digital filter is obtained. This latter approach is also inferior to the block based computation, since it is dependent on the image size and its computational complexity for the calculation of the 16 central moments up to the order (4,4) of an image with $N \times N$ points is $4N^2 + 16N + 80$ additions and only 32 multiplications or power calculations

2.1.1. *Geometrical moments.* Consider a binary digital image f(x,y), with N_1 pixels in the horizontal axis and N_2 pixels in the vertical axis. The 2-D geometrical moments of order (p,q) of the image are defined by the relation:

$$m_{pq} = \sum_{x=0}^{N_1-1} \sum_{y=0}^{N_2-1} x^p y^q f(x,y), p, q = 0, 1, 2, \ldots \tag{2}$$

Since the background level is 0, only the pixels with level 1 are taken into account in the computation of the moments. Thus, the 2-D geometrical moments of order (p,q) of the image $f(x,y)$ are defined by the relation:

$$m_{pq} = \sum_x \sum_y x^p y^q f(x,y), \forall x,y, f(x,y) = 1 \tag{3}$$

Specifically, if the image $f(x,y)$ is represented by k blocks, as it is described in (1), all the image pixels with level 1 belong to the k image blocks and therefore (3) may be rewritten:

$$m_{pq} = \sum_{i=0}^{k-1} m_{pq}^{b_i} = \sum_{i=0}^{k-1} \sum_{x=x_{1,b_i}}^{x_{2,b_i}} \sum_{y=y_{1,b_i}}^{y_{2,b_i}} x^p y^q \tag{4}$$

where x_{1,b_i}, x_{2,b_i} and y_{1,b_i}, y_{2,b_i} are the coordinates of the block b_i with respect to the horizontal axis and to the vertical axis, respectively. In (4), if the rectangular form appearing within the blocks is taken into account, then the geometrical moments of one block b, with coordinates x_{1b}, x_{2b}, y_{1b}, y_{2b}, are given by

$$\begin{aligned} m_{pq}^b &= \sum_{x=x_{1,b}}^{x_{2,b}} \sum_{y=y_{1,b}}^{y_{2,b}} x^p y^q = x_{1b}^p \sum_{y=y_{1,b}}^{y_{2,b}} y^q + (x_{1b}+1)^p \sum_{y=y_{1,b}}^{y_{2,b}} y^q + \ldots + x_{2b}^p \sum_{y=y_{1,b}}^{y_{2,b}} y^q \\ &= \left[\sum_{x=x_{1,b}}^{x_{2,b}} x^p \right] \left[\sum_{y=y_{1,b}}^{y_{2,b}} y^q \right] \end{aligned} \tag{5}$$

Using the rectangular form appearing within the block, the computational effort, which is characterized by the complexity $O(N^2)$ for the calculation of moments using (2), is reduced to O(N) for the calculation of moments using (5). For the computation of (5), it is adequate to calculate the following summations of the powers of x and y :

$$S_{x_{1b},x_{2b}}^p = \sum_{x=x_{1b}}^{x_{2b}} x^p, S_{y_{1b},y_{2b}}^q = \sum_{y=y_{1b}}^{y_{2b}} y^q, \; x,y,p,q \in Z \tag{6}$$

Moreover, taking into account the known formulae:

$$\begin{aligned} S_{1,n}^1 &= \frac{n(n+1)}{2} \\ S_{1,n}^2 &= \frac{n(n+1)(2n+1)}{6} \\ S_{1,n}^3 &= \frac{n^2(n+1)^2}{4} \\ S_{1,n}^4 &= \frac{n(n+1)(2n+1)(3n^2+3n+1}{30} \end{aligned} \tag{7}$$

and in the general case for sums of powers greater than 4, the formula

$$\begin{aligned} &\binom{m+1}{1} S_{1,n}^1 + \binom{m+1}{2} S_{1,n}^2 + \ldots \\ &\ldots + \binom{m+1}{m} S_{1,n}^m = (n+1)^{m+1} - (n+1) \end{aligned} \tag{8}$$

where $m, n \in Z$ and

$$\begin{pmatrix} i \\ j \end{pmatrix} = \frac{i!}{j!(i-j)!}$$

are the number of combinations of i objects taken j at a time, it is concluded that the summation $S^p_{x_1,x_2}$ can be directly calculated by the following analytical formulae:

$$\begin{aligned}
S^1_{x_{1b},x_{2b}} &= S^1_{1,x_{2b}} - S^1_{1,x_{1b}-1} = \frac{x_{2b}(x_{2b}+1) - x_{1b}(x_{1b}-1)}{2} \\
S^2_{x_{1b},x_{2b}} &= S^2_{1,x_{2b}} - S^2_{1,x_{1b}-1} = \frac{x_{2b}(x_{2b}+1)(2x_{2b}+1) - x_{1b}(x_{1b}-1)(2x_{1b}-1)}{6} \\
S^3_{x_{1b},x_{2b}} &= S^3_{1,x_{2b}} - S^3_{1,x_{1b}-1} = \frac{x^2_{2b}(x_{2b}+1)^2 - x^2_{1b}(x_{1b}-1)^2}{4} \\
S^4_{x_{1b},x_{2b}} &= S^4_{1,x_{2b}} - S^4_{1,x_{1b}-1} \\
&= \frac{x_{2b}(x_{2b}+1)(2x_{2b}+1)(3x^2_{2b}+3x_{2b}-1)}{30} \\
&\quad - \frac{x_{1b}(x_{1b}-1)(2x_{1b}-1)(3x^2_{1b}+3x_{1b}-1)}{30}
\end{aligned}$$

$$\begin{aligned}
S^p_{x_{1b},x_{2b}} &= \frac{(x_{2b}+1)^{p+1} - x^{p+1}_{1b} - (x_{2b}-x_{1b}+1) - \begin{pmatrix} p+1 \\ 1 \end{pmatrix} S^1_{x_{1b},x_{2b}}}{p+1} \\
&\quad - \frac{\begin{pmatrix} p+1 \\ 2 \end{pmatrix} S^2_{x_{1b},x_{2b}} - \cdots - \begin{pmatrix} p+1 \\ p-1 \end{pmatrix} S^{p-1}_{x_{1b},x_{2b}}}{p+1} \\
&\quad \text{for all } p \in Z^+
\end{aligned} \tag{9}$$

The summation $S^q_{y_{1b},y_{2b}}$ is computed in a similar manner. Fast computation of the 2-D geometrical moments of one block, according to (5), is achieved using the above simple and analytical formulae.

According to (4), the 2-D geometrical moments of the whole image are computed as the summation of the 2-D geometrical moments of all the individual blocks of the binary image.

2.1.2. *Central moments.* The 2-D central moments of an image f(x,y) are invariant under image translation and they are defined as

$$\mu_{pq} = \sum_{x=0}^{N_1-1} \sum_{y=0}^{N_2-1} (x-\overline{x})^p (y-\overline{y})^q f(x,y) \tag{10}$$

where $\overline{x} = \frac{m_{10}}{m_{00}}, \overline{y} = \frac{m_{01}}{m_{00}}$ are the coordinates of the centroid. Since all the image pixels with level 1 belong to the k image blocks, (10) may be rewritten as

$$\mu_{pq} = \sum_{i=0}^{k-1} \mu^{b_i}_{pq} = \sum_{i=0}^{k-1} \sum_{x=x_{1,b_i}}^{x_{2,b_i}} \sum_{y=y_{1,b_i}}^{y_{2,b_i}} (x-\overline{x})^p (y-\overline{y})^q \tag{11}$$

where $x_{1,b_i}, x_{2,b_i}, y_{1,b_i}, y_{2,b_i}$ are the coordinates of the block. The coordinates of the centroid in equation (11), refer to the center of the gravity of the whole image and not to the centroid of each block. The computation of the geometrical moments m_{00}, m_{10}, m_{01} using (5) and (9)

ensures the fast computation of the centroid of the image. In (11), if the rectangular form appearing within the blocks is taken into account, then the central moments of one block b, with coordinates $x_{1b}, x_{2b}, y_{1b}, y_{2b}$, are given by

$$\mu_{pq}^{b} = \sum_{x=x_{1b}}^{x_{2b}} \sum_{y=y_{1b}}^{y_{2b}} (x-\overline{x})^p (y-\overline{y})^q = \left[\sum_{x=x_{1b}}^{x_{2b}} (x-\overline{x})^p \right] \left[\sum_{y=y_{1b}}^{y_{2b}} (y-\overline{y})^q \right] \tag{12}$$

The complexity is reduced from $O(N^2)$ to O(N). For the computation of (12), it is adequate to calculate the two summations of the product. Using the mathematical identity:

$$(c-d)^m = c^m - \begin{pmatrix} m \\ 1 \end{pmatrix} c^{m-1}d + \begin{pmatrix} m \\ 2 \end{pmatrix} c^{m-2}d^2 - \ldots + (-1)^m d^m \tag{13}$$

it is concluded that

$$\begin{aligned}
\sum_{x=1}^{x_{2b}} (x-\overline{x}) &= S^1_{1,x_{2b}} - x_{2b}\overline{x} \\
\sum_{x=1}^{x_{2b}} (x-\overline{x})^2 &= S^2_{1,x_{2b}} - 2\overline{x}S^1_{1,x_{2b}} + x_{2b}\overline{x}^2 \\
\sum_{x=1}^{x_{2b}} (x-\overline{x})^3 &= S^3_{1,x_{2b}} + 3\overline{x}^2 S^1_{1,x_{2b}} - x_{2b}\overline{x}^3 \\
\sum_{x=1}^{x_{2b}} (x-\overline{x})^4 &= S^4_{1,x_{2b}} - 4\overline{x}S^2_{1,x_{2b}} + 6\overline{x}^2 S^2_{1,x_{2b}}
\end{aligned} \tag{14}$$

and

$$\begin{aligned}
\sum_{x=x_{1b}}^{x_{2b}} (x-\overline{x}) &= \sum_{x=1}^{x_{2b}} (x-\overline{x}) - \sum_{x=1}^{x_{1b}-1} (x-\overline{x}) \\
&= S^1_{x_{1b},x_{2b}} - (x_{2b}-x_{1b}+1)\overline{x} \\
\sum_{x=x_{1b}}^{x_{2b}} (x-\overline{x})^2 &= \sum_{x=1}^{x_{2b}} (x-\overline{x})^2 - \sum_{x=1}^{x_{1b}-1} (x-\overline{x})^2 \\
&= S^2_{x_{1b},x_{2b}} - 2\overline{x}S^1_{x_{1b},x_{2b}} + (x_{2b}p - x_{1b}+1)\overline{x}^2 \\
\sum_{x=x_{1b}}^{x_{2b}} (x-\overline{x})^3 &= \sum_{x=1}^{x_{2b}} (x-\overline{x})^3 - \sum_{x=1}^{x_{1b}-1} (x-\overline{x})^3 \\
&= S^3_{x_{1b},x_{2b}} - 3\overline{x}S^2_{x_{1b},x_{2b}} + 3\overline{x}^2 S^1_{x_{1b},x_{2b}} + (x_{2b}-x_{1b}+1)\overline{x}^3 \\
\sum_{x=x_{1b}}^{x_{2b}} (x-\overline{x})^p &= S^p_{x_{1b},x_{2b}} - \overline{x}\begin{pmatrix} p \\ 1 \end{pmatrix} S^{p-1}_{x_{1b},x_{2b}} + \ldots + (-1)^p (x_{2b}-x_{1b}+1)\overline{x}^p \\
&\qquad \forall p \in Z^+
\end{aligned} \tag{15}$$

where $S^p_{x_{1b},x_{2b}}$ have been calculated from (9). The above analytical formulae (15) are used for the fast computation of the factor $\sum_{x=x_{1b}}^{x_{2b}} (x-\overline{x})^p$ of the central moments (12) of the block b. The factor $\sum_{y=y_{1b}}^{y_{2b}} (y-\overline{y})^q$ appearing in (12) is calculated in a similar manner. The fast computation of the central moments of each block, using the proposed method, ensures the fast computation of the central moments of the whole image, according to (11).

2.1.3. *Normalized central moments and moment invariants.* The 2-D normalized central moments of an image f(x,y) are defined as $n_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}}$ where $\gamma = \frac{p+q}{2} + 1, \;\; p+q = 2, 3, \ldots$ and μ_{pq} are the corresponding central moments of the image. The central moments are required for the computation of the normalized central moments.

A set of seven moments, which are invariant to translation, rotation and scaling factors and called moments invariants [18, 19], is derived from the normalized central moments. Therefore, the fast computation of the central moments ensures the fast computation of the normalized central moments and of the moments invariants.

2.2. *Computational Complexity in Moment Estimation Using IBR*

It is clear from the IBR algorithm that block extraction is a pixel checking process without involving any numerical operations, and requires only one pass from each point of the image. Therefore, IBR is fast and the required time is proportional to the image size. However, in pattern recognition applications, IBR is applied to images of separated objects rather than to the whole image. In a number of such applications it results that computation time for block extraction is much less than the time for image file reading and image segmentation.

Consider a binary image that contains one rectangular block with level 1. For simplicity and without loss of generality, assume a square block with $M \times M$ points. In the sequel we estimate the computational complexity required for the computation of the geometrical moments of order up to $(L-1, L-1)$. The analysis concerning the computational complexity of other sets of moments may be given in a similar manner. It is seen from (2) that direct computation of one geometrical moment requires M^2 power computations, M^2 multiplications and M^2 additions. For the computation of L^2 moments, L^2M^2 power computations, L^2M^2 multiplications and L^2M^2 additions are required.

Consider (5), which exploits the rectangular form appearing within the block. For the computation of the factor $S^p_{x_{1b},x_{2b}}$, LM power calculations and LM additions are required. The same number of operations are required for the computation of the factor $S^q_{y_{1b},y_{2b}}$. Therefore, for the computation of L^2 geometrical moments using (5), $2LM$ power calculations, L^2 multiplications and $2LM$ additions are required for one block. Now consider the analytical formula (9), where the same binomial coefficients appear for any specific geometrical moment of each block of the image; therefore the corresponding computational effort is reduced by the number of blocks. Moreover, the factorials for the determination of the binomial coefficients require least effort, *i.e.*, one multiplication for the calculation of each $m!$ in terms of $(m-1)!$. There are two alternative approaches for the execution of the power calculations in (9). In the first one, 2 power calculations are considered. Alternatively, the calculation of x^p_{1b} and $(x_{2b}+1)^{p+1}$ may be seen as 2 multiplications in terms of x^p_{1b} and $(x_{2b}+1)^{p+1}$ respectively. In the following analysis the first approach is used. The factors $S^i_{x_{1b},x_{2b}}$, i = 1,2,...,p-1, that have been computed previously and their values stored, are used for the computation of the $S^p_{x_{1b},x_{2b}}$. Thus the computation of the $S^p_{x_{1b},x_{2b}}$ from equation (9) requires 2 power calculations, p multiplications and p additions. The computation of the sum of $S^q_{y_{1b},y_{2b}}$ requires 2 power calculations, q multiplications and q additions. The calculation of all the L^2 moments requires 4L power calculations, $2L^2 - L$ multiplications and $L^2 - L$ additions.

Table 1 demonstrates the above results. The complexity is reduced from 2-D format (2) to 1-D using image block representation and (5). Moreover, the complexity is independent of the size when the analytical formula (9) is used. The required number of power calculations, multiplications and additions for the computation of the geometrical moments up to the order (4,4) of a block with $M \times M$ pixels, where M varies from 1 to 100, is shown in Figure 2.

LEMMA 2.1. *Assuming that the complexity of raising a number to a power is the same as one multiplication, the computation of (5) and (9) requires* $L^2 + 2LM$ *and* $3L^2 + 2L$

Operations Number	Direct Computation from equation (2)	Computation from equation (5)	Computation from equation (9)
power calculations	L^2M^2	2 L M	4 L
multiplications	L^2M^2	L^2	$2L^2 - L$
additions	L^2M^2	2 L M	$L^2 - L$

TABLE 1. The required number of operations for the computation of geometrical moments up to the order (L-1,L-1), of one block with $M \times M$ pixels.

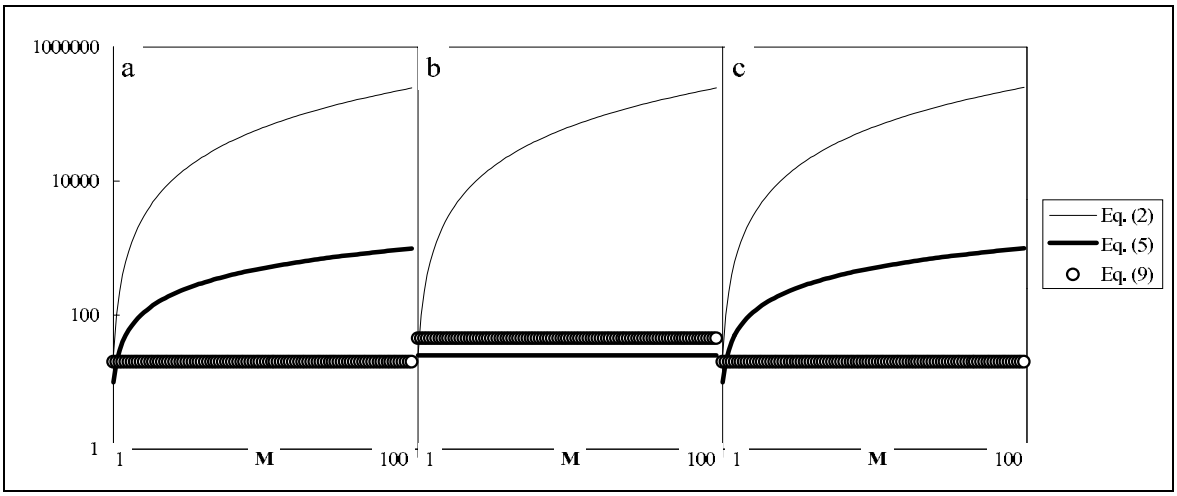

FIGURE 2. Number of operations for the geometrical moments computation up to the order $(L-1, L-1)$, of a $M \times M$ block, from equations (2), (5) and (9) with $L = 5$ and $M = 1, 2, \ldots, 100$. (a) Number of power calculations. (b) Number of multiplications. (c) Number of additions.

multiplications respectively. Comparing the above number of multiplications, (5) has less computational complexity than (9) when

$$L^2 + 2LM \leq 2L^2 + 3L \Longrightarrow M \leq \frac{L+3}{2} \tag{16}$$

However, in typical pattern recognition applications the moments usually are calculated up to the order (4,4). The higher order moments are not used since they are very sensitive to noise. From (16), it becomes clear that for $L = 5$, if an edge of a block contains less than 4 pixels it is faster to compute the sum of powers of the variable that corresponds to that edge directly using (5) instead of (9).

Consider the worst case of an $N \times N$ chessboard image with $N^2/2$ blocks. Since now $M = 1$, according to the criterion of Lemma 2.1, the computation using (5) requires LN^2 power calculations, $L^2N^2/2$ multiplications and LN^2 additions. Using (2), $L^2N^2/2$ power calculations, $L^2N^2/2$ multiplications and $L^2N^2/2$ additions are required. Thus, IBR for the computation of moments is still computationally attractive in comparison with the use of (2).

2.3. *Critical Points Extraction Involving IBR*

An object normalization procedure is first executed in order to facilitate rotation invariant descriptions of the objects. Specifically, the maximal axis of the object is found and the whole object is rotated in such a way that the maximal axis has a vertical position and the upper half of the image object contains most of the object's mass. At this point, it is necessary to give the following definitions :

1. *Group* is an ordered set of connected blocks, in such a way that all its intermediate blocks are connected with two other blocks, while the first and last blocks are connected with only one block.
2. *Junction point* is a point that it is connected with two other points.
3. *End point* is a point that it is connected with only one other point.
4. *Tree point* is the point that it is connected with more than two other points.
5. *Critical point* is a junction or an end or a tree point.

In this chapter, a fast non-iterative critical points detection method for block represented binary images is presented. The method has low computational complexity, extracts only critical points and to a degree appears to be immune to locality problems. This is achieved by the use of a suitable neighborhood in each case. Specifically, groups of connected blocks are formed. Each group is terminated when an adjacent block does not exist for its continuation, or when two or more blocks exist for the continuation of the group.

Each group defines a local neighborhood and all the necessary processing takes place in this neighborhood. Using a few simple rules for the processing, the groups are checked and labeled by certain categories:

1. *Vertical Elongated groups*. The absolute value of the angle of these groups with the horizontal axis is usually greater than 30°. The width of each block of a vertical elongated group should not exceed a threshold value. The connections among the blocks result in junction points, which belong to the thinned line that results from the group. For each pair of connected blocks, one junction point (the central point of the common line segment of two connected blocks) is extracted. For each block we check if the distance among its junction points and its extremities (*i.e.*, the central points of the edges of the small dimension) of the block exceeds a threshold value.
2. *Horizontal Elongated groups*. The absolute value of the angle of these groups with the horizontal axis is smaller than 30^{0}. The width of a horizontal elongated group is significantly greater than its height and also its height appears to have small variation. For the extraction of the junction points the algorithm starts from the left end of a horizontal elongated group and moves to the right with constant width steps. At each step a junction point is extracted at the middle of the height of the group at this vertical position.
3. *Angle groups*. The angle groups are connected with two other groups that lie on the same vertical or horizontal side of the angle group. The width and the height of an angle group are usually small. An angle group should not be connected to a noisy group. If a group has been labeled an angle group and it is connected with a noisy group, then the label "angle" is replaced by the horizontal elongated label or the vertical elongated label. Three junction points are extracted from an angle group, for the formulation of an angle.
4. *Noisy groups*. The noisy groups have width and height less than a threshold and they are connected to only one group, which is not an angle group. In most cases, the noisy groups are connected from the left or right side to vertical elongated groups or from the top or bottom side to horizontal elongated groups. In these cases the extraction of junction points from the noisy groups is not acceptable, otherwise a noisy end point would be created. The noisy groups are branches of the object that have small height and width. Usually, junction points are extracted from the noisy groups if and only if the noisy group is connected at the ends of an elongated group.

Fig. 1 demonstrates (a) an image of the character B, (b) the extracted blocks, (c) the groups of blocks and (d) the critical points.

3. The Constrained Optimization Framework for Efficiently Training MLPs

To fill in the gap between theory and real world pattern recognition applications, it is imperative to study the performance of MLP learning algorithms in large scale problems and networks. For instance, OCR tasks are good examples of such large-scale problems. Usually it is not only possible, but also essential, to include a large dataset of different examples of each character in the training set, if acceptable recognition rates are to be achieved. Moreover, the relatively large number of categories and input features calls for networks with a large number of weights. Under these circumstances, stringent requirements are placed on learning algorithms in terms of speed, scalability properties and generalization capabilities. The proposed constrained optimization based MLP training framework has been designed to meet these needs.

In a series of papers [31, 32, 38–41] the first author has proposed that a successful way of improving these properties of MLP learning algorithms is to incorporate different forms of knowledge about learning in MLP in the form of well-defined constrained optimization tasks. We have introduced a framework of basic requirements for incorporating knowledge in MLP learning algorithms [38]. This framework was used as an engine for developing different Algorithms for Learning Efficiently using Constrained Optimization techniques (ALECO). The second in the series of these algorithms (ALECO-2) was based on the incorporation of knowledge about optimal use of momentum acceleration techniques. This algorithm was tested using several different benchmark training tasks (encoder, multiplexer, counter and XOR problems) and its performance was compared to that of several different MLP training algorithms (on-line and off-line back propagation (BP) [42, 43], Quickprop [44] and Delta-bar-Delta [45]) [31, 40]. As regards large-scale problems (multiplexers and encoders with up to 2048 input patterns and 4360 weights), which are of interest in this work, the following results were obtained:

- From all algorithms tried, only on-line BP and ALECO-2 exhibited good convergence ability in large-scale problems. By contrast, off-line BP, Quickprop and Delta-bar-Delta exhibited difficulties in achieving convergence in all large-scale tasks tried.
- Concerning learning speed, ALECO-2 was found to clearly outperform its closest rival (on-line BP) in all large-scale tasks. Moreover, ALECO-2 showed a relatively small standard deviation in the distribution of epochs needed to successfully complete a task, thus exhibiting reliability of performance as regards learning speed.

These results illustrate the excellent capabilities of ALECO-2 as regards learning speed and scalability properties and make it an excellent candidate for training MLPs to solve large-scale pattern recognition problems such as the OCR tasks studied here. This chapter presents our first opportunity to assess the generalization ability of ALECO-2 using extensive multi-font character datasets and compare it to the ability of on-line BP, whose good performance in large-scale problems is well known [46]. The patterns which are the input vectors of these MLPs employed in the suggested OCR system come from the feature extraction stage developed in the previous section and are based on the IBR principle.

3.1. *Derivation of ALECO-2*

Augmenting the BP algorithm with momentum is inherently heuristic in nature, although attempts have been made to invest it with theoretical background [47, 48]. Thus, the mathematical rigor of gradient descent—where much information is available in the form of convergence theorems [49, 50]—is compromised; in return, it is expected that bigger weight steps can be achieved by filtering out high frequency variations of the error surface in the weight space [42]. ALECO-2 is based on the idea of obtaining *optimal* weight steps by optimizing, at each epoch of the learning algorithm, the Euclidean distance between the current and previous epoch weights [31]. In this way, improved learning speed is achieved.

Consider a multilayered feedforward MLP with one layer of input, M layers of hidden and one layer of output units. The units in each layer receive input from all units in the previous layer. We denote the unit outputs and synaptic weights respectively by $O_{jp}^{(m)}$ and $w_{ij}^{(m)}$. The superscript (m) labels a layer within the structure of the MLP ($m = 0$ for the input layer, $m = 1, 2, \ldots, M$ for the hidden layers, $m = M + 1$ for the output layer), i and j denote units in layers $(m-1)$ and (m) respectively and p labels the input patterns.

The training procedure in ALECO-2 solves, *for each epoch*, the following problem: First, change the cost function

$$E = \frac{1}{2}\sum_{jp} \varepsilon_{jp}, \;\; \varepsilon_{jp} = \left(T_{jp} - O_{jp}^{(M+1)}\right)^2 \tag{17}$$

by a specified negative amount δE. After a sufficient number of epochs, the accumulated changes to the cost function should suffice to achieve the desired input-output relation. Second, simultaneously maximize the squared Euclidean distance

$$\Phi = \sum_{ijm} \left(w_{ij}^{(m)} - W_{ij}^{(m)}\right)^2 \tag{18}$$

between the weight vectors $\boldsymbol{w}$ at the present epoch and $\boldsymbol{W}$ at the immediately preceding epoch, in order to achieve optimal weight steps. This problem is solved in an elegant way by a straightforward generalization of the optimal control method introduced by Bryson and Denham [51].

ALECO-2 is an iterative procedure, whereby the synaptic weights $w_{ij}^{(m)}$ are changed by small amounts $dw_{ij}^{(m)}$ at each iteration so that the quadratic form

$$\sum_{ijm} dw_{ij}^{(m)} \cdot dw_{ij}^{(m)}$$

takes on a prespecified value $(\delta P)^2$. Thus, at each epoch, the search for an optimum new point in the weight space is restricted to a small hypersphere centered at the point defined by the current weight vector. If δP is small enough, the changes in E and Φ induced by changes in the weights can be approximated by the first differentials dE and $d\Phi$. The problem then amounts to determining, for given values of δP and δE, the values of $dw_{ij}^{(m)}$, so that the maximum value of $d\Phi$ is attained.

Maximization of $d\Phi$ is attempted with respect to $w_{ij}^{(m)}$ and $O_{jp}^{(m)}$. In the language of non-linear programming, the synaptic weights correspond to decision variables and the unit outputs correspond to state (solution) variables. These quantities must satisfy the state equations, *i.e.*, the constraints describing the network architecture

$$f_{jp}^{(m)}(\boldsymbol{O}, \boldsymbol{w}) = g\left(\sum_i w_{ij}^{(m)} O_{ip}^{(m-1)}\right) - O_{jp}^{(m)} = 0, \;\; m = 1, \ldots, M+1 \tag{19}$$

where g is the logistic function $g(x) = 1/(1 + \exp(-x))$. Biases are treated as weights emanating from units with constant, pattern-independent activation equal to one. Apart from the state equations, the following conditions should be satisfied at each epoch of the algorithm

$$dE - \delta E = 0, \;\; \sum_{ijm} dw_{ij}^{(m)} \cdot dw_{ij}^{(m)} - (\delta P)^2 = 0 \tag{20}$$

To maximize $d\Phi$, suitable Lagrange multipliers $\lambda_E^{jp(m)}$, $\lambda_\Phi^{jp(m)}$ of the $f_{jp}^{(m)}$ are introduced to take account of the architectural constraints. Two further multipliers λ_1 and λ_2 are also

needed to take account of the respective terminal conditions 20. Requiring that $d\Phi$ be maximum ($d^2\Phi = 0$, $d^3\Phi < 0$), we are led to the following equations for the Lagrange multipliers

$$\begin{aligned} \lambda_E^{jp(M+1)} &= O_{jp}^{(M+1)} - T_{jp}^{(M+1)} \\ \lambda_E^{ip(m)} &= \sum_j \lambda_E^{jp(m+1)} w_{ij}^{(m+1)} O_{jp}^{(m+1)} \left(1 - O_{jp}^{(m+1)}\right) \\ & \qquad 1 \le m \le M \end{aligned} \tag{21}$$

and

$$\begin{aligned} \lambda_\Phi^{jp(m)} &= 0 \ \ (1 \le m \le M+1) \\ \lambda_2 &= \frac{1}{2}\left[\frac{I_{EE}(\delta P)^2 - (\delta E)^2}{I_{\Phi\Phi} I_{EE} - I_{E\Phi}^2}\right]^{-1/2} \\ \lambda_1 &= (I_{E\Phi} - 2\lambda_2 \delta E)/I_{EE} \end{aligned} \tag{22}$$

where

$$I_{\Phi\Phi} = \sum_{ijm} \left(F_{ijm}\right)^2, \ \ I_{EE} = \sum_{ijm} \left(J_{ijm}\right)^2, \ \ I_{E\Phi} = \sum_{ijm} J_{ijm} F_{ijm} \tag{23}$$

with

$$F_{ijm} = 2\left(w_{ij}^{(m)} - W_{ij}^{(m)}\right), \quad J_{ijm} = \sum_p \lambda_E^{jp(m)} O_{jp}^{(m)} \left(1 - O_{jp}^{(m)}\right) O_{ip}^{(m-1)} \tag{24}$$

Moreover, the following updating rule is obtained for the weights:

$$\begin{aligned} dw_{ij}^{(m)} &= \frac{1}{2\lambda_2}(F_{ijm} - \lambda_1 J_{ijm}) \\ &= \left[\frac{I_{EE}(\delta P)^2 - (\delta E)^2}{I_{\Phi\Phi} I_{EE} - I_{E\Phi}^2}\right]^{1/2} \left(F_{ijm} - \frac{I_{E\Phi}}{I_{EE}} J_{ijm}\right) + \frac{J_{ijm}}{I_{EE}} \delta E \end{aligned} \tag{25}$$

The detailed calculations are given in [31]. Note the bound $|\delta E| \le \delta P \sqrt{I_{EE}}$ imposed on the value of δE by equation 26. We always use a value $\delta E = -\xi \delta P \sqrt{I_{EE}}$ where ξ is a constant between 0 and 1. Thus δP and ξ are the only free parameters of the algorithm which can be tuned to obtain optimal performance. It is shown in [31] that this guarantees convergence to global or local minima of the cost function for small enough δP.

4. Experimental Results

In the experimental study of this research effort we first evaluate the IBR methodology alone. Then we design an OCR system for evaluating both the IBR and ALECO-2 frameworks.

4.1. *Evaluating the IBR approach*

Concerning the IBR approach we consider the test images of Fig. 3. In Table 2 the number of pixels with object level, the number of rows with object pixels, the number of blocks extracted from these images using Algorithm 1, the required storage for 2-D images and the required storage for block represented images are shown. It can been seen that storage of blocks requires less space in comparison with the required storage for 2-D images. In images with a high entropy value, like images of text where a significant number of small blocks appears, the required time for the computation of the moments is reduced, using image block representation, by a factor between 10 and 50. In images with large areas of object level, like images of industrial parts, aircraft, ships *etc.*, the time reduction factor is much greater. The computation time of moments up to the order (4, 4), for the set of the test images of Fig. 3, using the four different methods described earlier, is given in this Section. These methods are (i) regular computation from (2), (ii) IBR and use of (5), (iii) IBR with the analytical formula

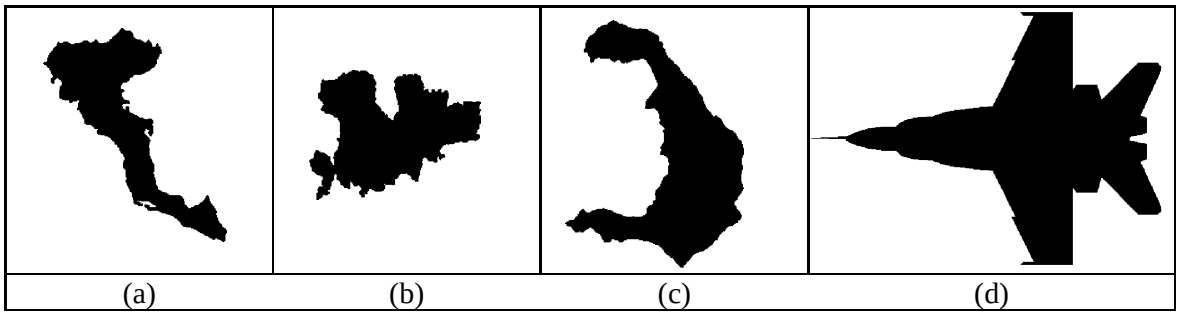

FIGURE 3. A set of test images. (a) Image of the island Corfu of 512x512 pixels. (b) Image of the island Mikonos of 512x512 pixels. (c) Image of the island Santorini of 512x512 pixels. (d) Aircraft image of 512x697 pixels.

Image	Pixels with object level	Rows with object pixels	Number of blocks	Storage for the 2-D image (bytes)	Storage for blocks (bytes)
Image of the island Corfu	41605	411	250	32768	2000
Image of the island Santorini	63203	474	257	32768	2056
Image of an aircraft	118831	494	397	44608	3176

TABLE 2. The number of pixels with object level, the number of rows with object pixels, the number of blocks, the required storage for 2-D images and the required storage for the block represented images for the set of test images of Figure 3.

(9) and (iv) IBR with the criterion provided by Lemma 2.1. The geometrical moments of the test images of Fig. 3 have been computed up to the order (4,4) and the results show that the use of IBR and (5) results in a reduction of computation time by a factor of 20. The use of the analytical formula (9) decreases the computation time by a factor of 200. Using the criterion provided by Lemma 2.1, the computation time is decreased by a factor of hundreds or thousands, since in most of the extracted blocks one edge has width less than 4 points.

4.2. *Setting up the OCR System*

Concerning the evaluation of the suggested pattern recognition design principles, extensive experiments were conducted to test their efficiency on specific OCR tasks involving typeset Greek characters. These experiments involve different combinations of features—Convolutional features and Critical Points-based features using IBR—and classifiers—Minimum Distance (MDC), on-line BP and ALECO-2. The spatial resolution of characters in all these experiments is 64 X 64 pixels.

Regarding the first kind of features which are compared in the suggested OCR system, namely, convolutional features, the pattern representation scheme was designed so as to simultaneously achieve:

- Filtering out of the noise in the pattern images by the successive application of a low pass filtering procedure.
- Efficient image encoding that takes into account that neighboring pixels are highly correlated [33] and forms a compressed pattern image representation of reduced dimensionality by removing redundant information. This is very important for the neural

classifiers in the next stage of the proposed system, since it is well known that improved generalization performance is dependent on the number of free parameters of the model [34, 35].

These criteria led directly to the employment of the Laplacian pyramid theory [33]. A one level Gaussian pyramid was used with the following features:

- Two generating kernels (Gaussian masks), were used (a 5-by-5 mask and a 4-by-4 mask). The pattern of weights $h(m,n)$ of the generating kernels was chosen subject to the constraints of normalization and equal contribution for the different levels [36], as well as equality for all the weights at a given level. This last constraint was selected for its simplicity instead of the separability one [33] and our experiments validated its efficiency.
- In order to achieve higher dimensionality reduction rates, we did not apply the convolution scheme required in [33]. Instead of performing the normal convolution operation as in [33, 37], a different, more complex, convolution scheme of the original rastered image g_0 was employed. The resulting pattern image is derived as follows: We define $g_{l1} = h_5 \bigoplus g_0$ and $g_{l2} = h_4 \bigoplus g_0$ as the convolved images obtained by convolving g_0 (the original pattern) with the generating kernels h_5 (5-by-5) and h_4 (4-by-4) respectively. This scheme takes into account different correlations of the neighboring image pixels. The final pattern image g_l, which is the input of the classifier, is obtained by keeping the most important terms of g_{l1} and g_{l2}. It has found been experimentally that by utilizing the first 41 such terms we achieve the best performance of our OCR system in the experiments described next.

Regarding the second kind of features involved in the suggested OCR system, namely, the Critical Points extracted using the IBR methodology previously depicted, it could be mentioned that the pattern vector had 256 components. This number was selected to correspond to the maximum number of critical points extracted from any one of the training images.

All experiments were carried out using a software platform developed on a 266MHz Pentium Windows-95 PC. Using a menu driven environment we can select an already implemented method for every stage of the recognition process (preprocessing, segmentation, normalization, feature extraction and classification) and test the recognition rate of our OCR system. The platform can process several pre-recognized characters from many fonts for automatic recognition rate extraction. Moreover, it can use several training sets and test their performance on any TIFF image which contains typewritten characters. An attractive feature of this OCR platform, which substantially facilitated the experimental part of this work, is that the user can easily create character bases from TIFF character images. Character sets can then be created containing user-specified parts of these character bases, to be used as training or recognition sets in the classifying procedure.

4.3. *Training and Test Character Sets*

For reasons of fair comparison, all experiments were conducted using the same character sets for training and recognition. Both natural and artificial sets were used. The natural sets were produced using images of plain text of more than 1500 Greek characters. The artificial sets were made from images containing 10 versions of 32 different characters (lower-case Greek letters) for a total of 320 characters. We used 6 different character fonts both in the artificial and the natural sets: Arial, Arc, Times, Bold Arial, Bold Arc and Bold Times. We also used 2 different contrast regulations in the scanning software for the character sets, which provides 2 different character thicknesses. The scanning resolution was stable for all the experiments at 300 dpi though the size of the characters varied from set to set. We chose

- the junction of Arial, Arc and Times artificial character bases (total of 960 characters) as our training set. More specifically, the training set consisted of the following:

Arc set We used an HP scanner at 300 dpi resolution with the default contrast regulation in the scanning software in order to binarize a text printed from an HP printer with 320 characters (32 classes with 10 prototypes each) from MS Windows Arc font. We produced a 731K TIFF image from which we derived the Arc set.

Arial set The same procedure as above was followed using the MS Windows Arial font.

Times set The same procedure as above was followed, but using the MS Windows Times font.

- Various artificial and natural character sets were used for recognition (testing) purposes. In particular, 3 natural sets of Arial and Times fonts (total of 4647 characters) and 3 artificial sets of Bold Arial, Bold Arc and Bold Times fonts (total of 960 characters) were used as test sets. These recognition sets have the following specifications:

  text1 set The scanner, printer and scanning software contrast regulation were different from those used in the training phase. We had 1554 characters in a plain text from MS Windows Arial font at 300 dpi resolution in a 629K TIFF file.

  text2 set We had different scanner, printer and contrast regulation from those used in the training phase. The character set consisted of 1508 characters in a plain text from MS Windows Times font at 300 dpi resolution in a 612K TIFF file.

  text3 set The scanner and printer were different from those used at the training phase, but the contrast regulation was the same as in the training phase. We had 1585 characters in a plain text from MS Windows Arial font at 300 dpi resolution in a 616K file.

  barc set The scanner, printer and scanning software contrast regulation were those used in the training phase. We had 320 characters (32 classes with 10 prototypes each) from MS Windows Bold Arc font at 300 dpi resolution in a 731K file.

  barial set The same as with barc set with characters from MS Windows Bold Arial font.

  btimes set The same as with barc set with characters from MS Windows Bold Times font.

4.4. *Types of Experiments Conducted*

The following architectures combining different features and classifiers were implemented:

1. A minimum distance classifier was trained using either the 41 convolutional weighted mask features $x_{l,i}^{k,p}$ or the 256 IBR critical points as inputs. Various distance metrics were used and the best recognition accuracy results, presented in Table 3, were achieved using a fourth power metric

$$D(k) = \left[\sum_i (x_{l,i}^{k,p} - m_{l,i}^{k})^4 \right]^{1/4}$$

2. A fully connected feedforward network with two layers of weights and a 41-40-32 architecture was trained using on-line BP as the training algorithm with a learning rate of 0.4 and a momentum acceleration factor of 0.5. For each character belonging to a category k, the convolutional features $x_{l,i}^{k,p}$ were used as the network input. The desired output of all output nodes was 0, except for the k-th node, whose desired output was 1. A set of characters distinct from the training and test sets was used as a validation set: The final weights used for testing were those for which recognition accuracy in the validation set during training had reached its maximum value. Second, a similar MLP with 256-50-32 architecture (with respect to the other parameters being precisely the same) has been involved in the case of IBR Critical Points-based feature extraction.

	MDC-Conv	MDC-IBR	On-line BP (Conv.)	On-line BP (IBR)	ALECO-2 (Conv.)	ALECO-2 (IBR)
text1	97.94	98.13	99.16	99.09	99.74	*99.94*
text2	93.77	95.89	98.67	99.07	99.07	*99.64*
text3	99.37	99.56	99.56	99.75	99.87	*99.95*
barc	85.31	85.94	92.18	95.31	96.69	*99.74*
barial	91.25	92.50	97.18	96.88	96.98	*98.23*
btimes	86.88	86.31	93.75	95.31	97.81	*98.45*

TABLE 3. Classification accuracy results for the characters in our six different test sets using various feature-classifier combinations. Each column corresponds to a described experiment.

3. Two MLPs with the same architecture, inputs and desired outputs as in case 2 were trained using ALECO-2 as the learning algorithm. The parameter values $\delta P = 0.5$ and $\xi = 0.5$ were used.

4.5. *Results*

Our results regarding the recognition accuracy of different feature extraction-classification combinations are summarized in Table 3.

¿From these results the following conclusions can be drawn:

- The use of IBR-based features helps improve recognition accuracy results. This is evident in nearly all the results obtained using either MDC or MLP-based classifiers.
- ALECO-2 emerges as a powerful neural network training algorithm for large-scale OCR tasks. In our experiments, the excellent speed and scalability properties of ALECO-2 were confirmed: Convergence of the training procedure for experiment no. 5 in a network with 2992 weights was achieved in just 150 epochs (using Fahlman's 0.4-0.6 criterion [44]). Moreover, our results show attractive generalization ability properties: Compared with on-line BP, ALECO-2 achieved better recognition rates, including substantial improvements in the barc and btimes test sets. The good generalization ability of ALECO-2 can probably be attributed to the fact that the cost function is changed monotonically and gradually [31], without the abrupt jumps sometimes involved in learning algorithms which incorporate heuristics in their formulation (including on-line BP). Note that, in the same spirit of constrained learning, it is possible to augment ALECO-2 with weight elimination techniques [34], which will hopefully further improve its generalization ability without adverse effect on its learning speed.

5. Conclusions and a discussion of the future trends

In this chapter two main methodologies have been presented and evaluated for the design of fast and efficient pattern recognition systems. Namely, the IBR approach for image representation as well as a constrained optimization based framework for training MLP pattern classifiers.

First, the image block representation idea and the associated algorithm were presented. Owing to the nature of the digital image, only rectangular areas with the same level are present. IBR uses these rectangular similarities and offers advantages in image handling and computational cost. IBR also provides a perception about rectangular image regions larger than a pixel. 2-D moments is a classical image analysis tool, and the use of block represented binary images dramatically decreases the computation effort. The complexity of the algorithm for computation of moments in block represented images is independent of

image size. Using the IBR scheme for computation of moments a rate of 35 to 50 frames/sec with 512x512 images is achieved. The real-time moments computation in block represented binary images is useful in motion detection, moving object recognition, target identification and tracking, and robot vision applications. Other image processing and analysis tasks can be also performed on block represented images, but this is a topic for future research. The extension of the proposed method to gray level images is straightforward. Each block is represented by five integers: the coordinates of the upper left and lower right corners and its gray level value. For the moments computation, it suffices to calculate the moments of the corresponding binary block and to multiply them by the gray level value of the block, since all pixels of the block have the same gray level value.

Second, the constrained optimization-based training of MLPs has been illustrated to be a fruitful idea for designing fast and efficient pattern classifiers in terms of convergence, scalability properties and generalization performance. It is based on the incorporation of knowledge in the learning process of MLPs, in terms of additional functions to be optimized during the learning process, but in a decomposing manner. More successful forms of knowledge functions as well as new ways for incorporating knowledge in the MLP training procedure are under investigation by the authors, in order to facilitate the design of fast and effective pattern recognition systems.

References

[1] L. N. Kanal, "Automatic Pattern Recognition," in *Pattern Recognition and Image Processing in Physics*, (R. A. Vaughan, ed.), NATO-ASI, Dundee, August 1990: IOP Publishing, 1991.

[2] H. Samet, "The quadtree and related hierarchical data structures," *Computing Survey*, vol. 16, no. 2, pp. 187–260, 1984.

[3] H. Freeman, "Computer processing of line drawings," *ACM Computing Surveys*, vol. 6, pp. 57–97, 1974.

[4] D. W. Paglieroni and A. K. Jain, "Control point transforms for shape representation and measurement," *Computer Vision, Graphics and Image Processing*, vol. 42, pp. 87–111, 1988.

[5] R. L. Kashyap and R. Chellappa, "Stochastic models for closed boundary analysis: Representation and reconstruction," *IEEE Trans. Information Theory*, vol. 27, no. 5, pp. 627–637, 1981.

[6] J. Piper, "Efficient implementation of skeletonisation using interval coding," *Pattern Recognition Letters*, vol. 3, pp. 389–397, 1985.

[7] B. G. Mertzios, "Block realization of 2-D IIR digital filters," *Signal Processing*, vol. 7, no. 2, pp. 135–149, 1984.

[8] B. G. Mertzios, "Fast block implementation of two-dimensional recursive digital filters via VLSI array processors," *Archiv für Elektronik und Übertragungstechnik (AEU)*, vol. 43, no. 1, pp. 55–58, 1990.

[9] X. Liu and A. Fettweis, "Multidimensional digital filtering by using parallel algorithms based on diagonal processing," *Multidimensional Systems and Signal Processing*, vol. 1, pp. 51–66, 1990.

[10] I. M. Spiliotis and B. G. Mertzios, "Real-time computation of statistical moments on binary images using block representation," in *Proceedings of the 4th International Workshop on Time-Varying Image Processing and Moving Object Recognition*, Florence, Italy, June 10-11, pp. 27–34, 1993.

[11] I. M. Spiliotis, D. A. Mitzias and B. G. Mertzios, "A skeleton-based hierarchical system for learning and recognition," in *Proceedings of the International Symposium on the Mathematical Theory of Networks and Systems, (MTNS 93)*, Regensburg, Germany, August 2-6, pp. 873–878, 1993.

[12] S. A. Dudani, K. J. Breeding and R. B. McGhee, "Aircraft identification by moment invariants," *IEEE Trans. Computers*, vol. C-26, pp. 39–45, 1977.

[13] G. L. Cash and M. Hatamian, "Optical character recognition by the method of moments," *Computer Vision, Graphics and Image Processing*, vol. 39, pp. 291–310, 1987.

[14] K. Tsirikolias and B. G. Mertzios, "Statistical pattern recognition using efficient 2-D moments with applications to character recognition," *Pattern Recognition*, vol. 26, no. 6, pp. 877–882, 1993.

[15] F. A. Sadjaji and E. L. Hall, "Three-dimensional moments invariants," *IEEE Trans. Pattern Anal. and Machine Intel.*, vol. PAMI-2, no. 2, pp. 127–136, 1980.

[16] A. P. Reeves, M. L. Aleey and O. R. Mitchell, "A moment based two-dimensional edge operator," in *Proc. IEEE Conf. Comput. Vision, Pattern Recogn.*, pp. 114–120, 1983.

[17] A. Khotanzad and Y. H. Hong, "Invariant image recognition by Zernike moments," *IEEE Trans. Pattern Anal. and Machine Intel.*, vol. 12, no. 5, pp. 489–497, 1990.

[18] M. K. Hu, "Pattern recognition by moments invariants," *Proceedings IRE*, vol. 49, 1961.

[19] M. K. Hu, "Visual pattern recognition by moment invariants," *IRE Trans. Inform. Theory*, vol. IT-8, pp. 179–187, 1962.
[20] M. R. Teague, "Image analysis via the general theory of moments," *Journal Opt. Soc. America*, vol. 70, pp. 920–930, 1980.
[21] Y. S. Abu-Mostafa and D. Psaltis, "Image normalization by complex moments," *IEEE Trans. Pattern Anal. and Machine Intel.*, vol. PAMI-7, no. 1, pp. 46–55, 1985.
[22] C. H. Teh and R. T. Chin, "On image analysis by the methods of moments," *IEEE Trans. Pattern Anal. and Machine Intel.*, vol. 10, no. 4, pp. 496–513, 1980.
[23] A. Papoulis, "Probability, Random Variables, and Stochastic Processes," McGraw-Hill, New York, 1965.
[24] G. Y. Tang, "A discrete version of Green's theorem," *IEEE Trans. Pattern Anal. and Machine Intel.*, vol. PAMI-4, no. 3, 1982.
[25] X. Y. Jiang and H. Bunke, "Simple and fast computation of moments," *Pattern Recognition*, vol. 24, no. 8, pp. 801–806, 1991.
[26] B. C. Li and J. Shen, "Fast computation of moments invariants," *Pattern Recognition*, vol. 24, no. 8, pp. 807–813, 1991.
[27] M. Hatamian, "A real-time two-dimensional moment generating algorithm and its single chip implementation," *IEEE Trans. Acoustics, Speech and Signal Processing*, vol. ASSP-34, no. 3, pp. 546–553, 1986.
[28] H. S. Baird, S. E. Jones and S. J. Fortune, "Image segmentation by shape-directed covers," in *Proc. 10th Intern. Conf. on Pattern Recognition*, Atlantic City, NJ, USA, June 16-21, pp. 820–825,1990.
[29] M. McKenna, J. O'Rourje and S. Suri, "Finding the largest empty rectangle in an orthogonal polygon," in Proc. of the 23rd Annual Allerton Conf. on Communication, Control and Computing, Urbana-Campaign, Illinois, USA, pp. 486–495, 1985.
[30] M. Orlowski, "A new algorithm for the largest empty rectangle problem," *Algorithmica*, vol. 5, no. 1, pp. 65–73, 1990.
[31] S. J. Perantonis and D. A. Karras, "An efficient constrained learning algorithm with momentum acceleration," *Neural Networks*, vol. 8, no. 2, pp. 237–249, 1995.
[32] D. A. Karras and S. J. Perantonis, "Comparison of learning algorithms for feedforward networks in large scale networks and problems," 1993. Presented at *IJCNN'93*,(Japan).
[33] P. J. Burt and E. H. Adelson, "The Laplacian pyramid as a compact image code," *IEEE Transactions on Communications*, vol. 31, no. 4, pp. 532–540, 1983.
[34] A. S. Weigend, D. E. Rumelhart, and B. A. Huberman, "Generalization by weight elimination with application to forecasting," in *Advances in Neural Information Processing Systems*, pp. 875–882, 1991.
[35] J. Moody, "Note on generalization, regularization, and architecture selection in nonlinear learning systems," in *Neural Networks for Signal Processing*, (B. H. Juang, S. Y. Kung, and C. A. Kamm, eds.), Piscataway, NJ: IEEE press, 1991.
[36] P. J. Burt, "Fast filter transforms for image processing," *Computer Graphics and Image Processing*, vol. 16, pp. 20–51, 1981.
[37] D. A. Karras, S. J. Varoufakis, and G. Carayannis, "A neural network for character recognition using a preprocessing Gaussian pyramid encoding block," *Neural Network World*, pp. 347–352, 1992.
[38] D. A. Karras and S. Perantonis, "An Efficient constrained training algorithm for feedforward networks," *IEEE Trans. on Neural Networks*, vol. 6, no. 6, 1995.
[39] D. A. Karras, S. J. Perantonis, and S. J. Varoufakis, "Constrained learning: a new approach to pattern classification using feedforward networks," in *Proceedings of WCNN'93*, (Portland, Oregon, USA), pp. IV235–IV238, 1993.
[40] S. J. Perantonis and D. A. Karras, "A fast constrained learning algorithm based on the construction of suitable internal representations," 1993. Presented at *IJCNN'93*, (Japan).
[41] S. J. Varoufakis, S. J. Perantonis, and D. A. Karras, "A family of efficient learning algorithms for feedforward networks based on constrained optimization techniques," in *Proceedings of NEURONET'93*, (Praga, Czech Republic), 1993.
[42] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, (D. E. Rumelhart and J. L. McClelland, eds.), ch. 8, pp. 318–362, MIT Press, Cambridge, MA, 1986.
[43] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back propagating errors," *Nature*, vol. 323, pp. 533–536, 1986.
[44] S. E. Fahlman, "Faster learning variations on back propagation: an empirical study," in *Proceedings of the 1988 Connectionist Models Summer School*, (San Mateo), pp. 38–51, Morgan Kaufmann, 1988.
[45] R. A. Jacobs, "Increased rates of convergence through learning rate adaptation," *Neural Networks*, vol. 1, pp. 295–307, 1988.

[46] F. Fogelman Soulie, "Neural network architectures and algorithms: a perspective," in *Artificial Neural Networks*, (T. Kohonen, K. Makisara, O. Simula, and J. Kangas, eds.), pp. 605–615, Elsevier Science Publishers B.V. (North Holland), 1991.
[47] A. Sato, "An analytical study of the momentum term in a back propagation algorithm," in *Artificial Neural Networks*, (T. Kohonen, K. Makisara, O. Simula, and J. Kangas, eds.), pp. 617–622, Elsevier Science Publishers B.V. (North Holland), 1991.
[48] M. Hagiwara, "Theoretical derivation of momentum term in back propagation," in *Proceedings of the IEEE International Conference on Neural Networks*, (Baltimore), pp. I682–I686, 1992.
[49] A. A. Goldstein, "On steepest descent," *J. SIAM Contr. Ser. A*, vol. 3, no. 1, pp. 147–151, 1965.
[50] R. Fletcher, *Practical Methods of Optimization*. Vol. 1, New York: Wiley, 1980.
[51] A. E. Bryson and W. F. Denham, "A steepest-ascent method for solving optimum programming problems," *Journal of Applied Mechanics*, vol. 29, pp. 247–257, 1962.
[52] R. Rohwer. The 'moving targets' training algorithm. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems*, pages 558–565, Morgan Kaufmann, San Mateo, 1990.
[53] J. Hertz, A. Krogh, and R. G. Palmer. *Introduction to the Theory of Neural Computation*. Addison-Wesley, 1991.
[54] T. Grossman, R. Meir, and E. Domany. Learning by choice of internal representations. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems*, pages 73–80, Morgan Kaufmann, San Mateo, 1990.