Pattern Decomposition and Associative Processing Applied to Object Identification

Benjamín Cruz, Humberto Sossa and Ricardo Barrón
Centro de Investigación en Computación – IPN
Av. Juan de dios Batís, Esq. Miguel Othón de Mendizábal
Ciudad de México, 07738, México
E-mails: benjamincruz@sagitario.cic.ipn.mx, hsossa@cic.ipn.mx, rbarron@cic.ipn.mx

Abstract. Pattern identification in the presence of noise is a main problem in pattern recognition. An essential characteristic of the noise acting on a pattern is its local nature. If a pattern is thus separated into enough sub-patterns, only few of them will be somehow affected, others will remain intact. In this note we propose a simple methodology that takes into account this property. A pattern is identified if enough of its sub-patterns are also identified. Since several patterns can share some of the sub-patterns, final decision is accomplished by means of a voting mechanism. Before deciding if a sub-pattern belongs to a pattern, sub-pattern identification in the presence of noise is done by an associative memory. Numerical and real examples are given to show the effectiveness of the proposal.

1 Introduction

A main problem in pattern recognition is pattern identification in the presence of noise. In real situations usually patterns appear distorted by noise and must be identified despite of this. One approach usually used to identify a pattern from a distorted version of it, is by means of an associative memory by which we reconstruct the distorted pattern. Associative memories have been used for pattern recovering for many years [1-13]. Usually, complete unaltered patterns are first used to build a chosen memory model. Trained memory models are next used to recover a given pattern, given a possibly distorted version it. This allows pattern identification.

One main feature of the noise affecting a pattern is its locality, i.e. the pattern is affected somehow at specific parts or locations; other parts remain unchanged. In this paper we take advantage of this situation and exploit it in two ways. In the one hand, we decompose the pattern into a set of sub-patterns. In the other hand, we make use of an associative memory specially designed to filter the noise affecting the patterns' sub-patterns. The resulting sets of sub-patterns are first used to build a bank of associative memories. During pattern recall a possibly distorted version of a pattern is first decomposed into its sub-patterns. Each sub-pattern is presented to its corresponding memory for noise cleaning. The cleaned sub-pattern is then used to index into a table for the set patterns sharing it. A simple but efficient voting mechanism allows to finally deciding the index of the corresponding pattern.

Lots of models of associative memories have been emerged in the last 40 years, starting with the *Lermatrix* of Steinbuch [1], then the *Linear Associator of* Anderson [2] and Kohonen [3], and the well-known model proposed by Hopfield in 1982, the *Hopfield Memory* [5]. For their operation, all of these models use the same algebraic structure. In the 90's appeared the so-called Morphological Associative Memories

(MAMS) [6] and [7]. While the classic memories found their operation on multiplications and additions, the MAMS do it in the *min* and *max* operations used in Mathematical Morphology. Several of these models, especially the morphological models are very efficient to recall patterns corrupted either with additive noise or subtractive noise. To overcome this problem, MAMS memories and theirs variants make use of the so-called kernel approach [6]. Kernels for MAMS are however difficult to find [9]. Additionally, if new patterns have to be added to the learning set the kernels need to be recomputed again. In [11], the authors describe a memory model able to handle mixed noise by means of the well-known median operator. Median operation is however time consuming as know. In this paper we show how by decomposing a pattern into its sub-patterns, we can avoid the use of kernels and the median operator. We give numerical and realistic examples where the effectiveness of the proposal is tested.

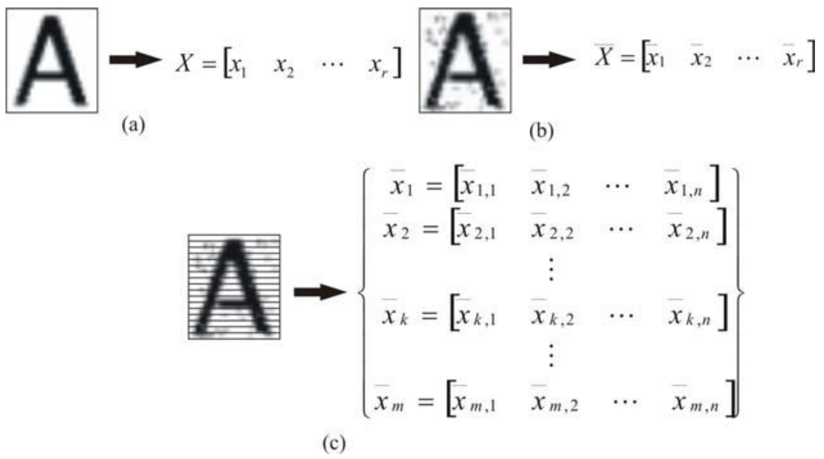
2 Basics About Associative Memories

An associative memory as defined in [13] is an input-output system able to associate an input pattern with an output pattern as follow: $a \rightarrow \boxed{\mathbf{M}} \rightarrow b$, with a and b, respectively the input and output patterns vectors. Each input vector forms an association with its corresponding output vector. An association between input pattern a and output pattern b is denoted by (a,b). For k a positive integer, the corresponding association will be (a^k,b^k). Associative memory **M** is represented by a matrix whose *ij*-th component is m_{ij} [2]. **M** is generated from a finite a priori set of known associations, known as *fundamental set of association or simply fundamental set* (FS) [13]. If k is an index, this FS is represented as: $\{(a^k,b^k), k = 1,p\}$, with p the cardinality of the set. The patterns integrating a FS are called *fundamental patterns* [13]. The nature of the FS provides an important judgment for associative classification. If for $k = 1,p$, it holds that $(a^k = b^k)$, then that memory is auto-associative, otherwise it is hetero-associative [13].

Fundamental patterns could be distorted with noise. A distorted version of a pattern a will be denoted as $\tilde{a}$. If when presenting to an associative memory **M** a fundamental pattern, **M** responds with the correct pattern, we say that **M** presents perfect recall. If for all patterns of a given FS, perfect recall is obtained, **M** is said to present perfect recall.

3 Idea of Solution

As already mentioned, the proposal is based on the locality of the noise affecting the pattern, i.e. when the object is decomposed into several parts, some of them will appear more or less affected by noise, some others will not. From these less altered and the unaltered parts is that the whole object is identified. For example in Figure 1(a) we have an image of an object for which a numerical representation (a pattern) has been obtained. In Figure 1(b), we have the same image but distorted with some noise, this of course affects also its numerical representation. Finally, in Figure 1(c) it is shown the same pattern but decomposed into several parts. By obtaining these set of parts (sub-patterns), as can be appreciated some sub-patterns appear altered, some others no. From these unaltered sub-patterns is that the object can be identified.



$X = [x_1 \ x_2 \ \cdots \ x_r]$
(a)
$\bar{X} = [\bar{x}_1 \ \bar{x}_2 \ \cdots \ \bar{x}_r]$
(b)
$\bar{x}_1 = [\bar{x}_{1,1} \ \bar{x}_{1,2} \ \cdots \ \bar{x}_{1,n}]$
$\bar{x}_2 = [\bar{x}_{2,1} \ \bar{x}_{2,2} \ \cdots \ \bar{x}_{2,n}]$
$\bar{x}_k = [\bar{x}_{k,1} \ \bar{x}_{k,2} \ \cdots \ \bar{x}_{k,n}]$
$\bar{x}_m = [\bar{x}_{m,1} \ \bar{x}_{m,2} \ \cdots \ \bar{x}_{m,n}]$
(c)

Figure 1. (a) An object and its numerical representation. (b) The same object altered by noise and its corresponding numerical representation altered also by the noise. (c) The corresponding pattern decomposed into parts (sub-patterns).

4 Basic Definitions

The steps composing the proposed methodology to recognize an object from its parts (sub-patterns) are explained next. For this let us have the following definitions:

Definition 2.1. Let B a pattern of an object O obtained somehow (for example as an image-vector by the standard row scanning method or a feature vector). A sub-pattern b of object O is a pattern obtained as B but from a part of O.

We have already mentioned that to get the sub-patterns of an object it is necessary to divide this object into parts and obtain their corresponding patterns. We have thus the following definition:

Definition 2.2. Let a set of m sub-patterns of an object, represented as row vectors of dimension n denoted by $b_k, k = 1, m$. The matrix B of dimensions $m \times n$ containing all of these patterns as its rows is called *base pattern*.

Definition 2.3. The set of all q matrices B^k is called the *base set of matrices* or simply the *base set*, and it is represented as: $\{B^k \mid k = 1, q\}$, with q the number of patterns or objects.

In what follows, for notation purposes b_k^i represents the k-th sub-pattern of the i-th object, with $k, k = 1, m$ and $i, i = 1, q$.

5 The Methodology

The proposal is composed of two main stages: 1) memory training and 2) object recognition. During training the chosen associative memories are built. Also the so-called voting matrix is build. During testing, the objects' sub-patterns are presented to the already trained memories for identification porpoises.

5.1 Learning phase

This phase is composed of two main steps as follows:

Step 1: For each k, take the b_k^i and build associative memory M^k. We can select any among the existing different models (see section 6).

Step 2: Taking into account that several objects can share a given sub-pattern, we build a matrix V (voting matrix) of dimensions $q \times m$, with *q* the number of objects and *m* the number of sub-patterns. Matrix V tells us exactly which sub-pattern is in which object. First row of V is reserved for first object, second row for the second object, and so on. To build V, we first fill it of 0's, i.e. $v_{i,j} = 0, i = 1, q; j = 1, m$. We then convert each sub-pattern b_k^i of each base pattern B^i to a decimal equivalent number, and assign this number to component $v_{i,k}$ of V. This would mean that sub-pattern b_k^i belongs to base pattern B^i. This completes the learning stage.

5.2 Recalling phase

We have two cases. First case is related with the recalling of a pattern of the FSP, second case, in the contrary, is focused on the recalling of a pattern of the same FSP but from a distorted version of it.

Case 1: Recalling a pattern of the FSP. For each base pattern B^k of the FSP:

Step 1: We begin by building a voting vector and filling it by 0's as follows $Z = \left(z_1 = 0 \quad z_2 = 0 \quad \cdots \quad z_q = 0\right)$.

Step 2: Now, for a given base pattern B^i (it can be anyone of them), for each of its sub-patterns b_k^i, we operate it with the corresponding associative memory $M^k, k = 1, m$, convert it to its decimal equivalent, let say d. We then look for all the appearances of d in matrix V at column k, and update the corresponding component z_i of vector Z as follows: For $i = 1, q$ do $z_i = z_i + 1 \; if \; v_{i,k} = d$. We repeat this process for each sub-pattern of B^i.

Step 3: We finally get the index of the corresponding pattern as:

$$j = \arg\max_i (z_i), i = 1, q \qquad \textbf{(1)}$$

The whole process is repeated for each B^i.

Case 2: Recalling a pattern of the FSP from a distorted version of it. Given a distorted version $\overline{B}^i$ of one the patterns of the FSP:

Step 1: Again, we begin by defining $Z = (z_1 = 0 \quad z_2 = 0 \quad \cdots \quad z_q = 0)$.

Step 2: For each sub-pattern $\overline{b}_k^i$ of $\overline{B}^i$, we operate it with the corresponding associative memory $M^k, k = 1, m$, convert it to its decimal equivalent, let say d. We then look for all the appearances of d in matrix V at column k, and update the corresponding component z_i of vector Z as follows: For $i = 1, q$ do $z_i = z_i + 1 \; if \; v_{i,k} = d$. We repeat this process for each base pattern.

Step 3: Get the index of the corresponding pattern as:

$$j = \arg\max_i (z_i), i = 1, q \qquad \textbf{(2)}$$

5.3 Variations

Instead of using the rows of binary patterns to define the base vectors we can use their columns or diagonals and follow the same procedure to learn and recall patterns. The idea is to decompose the pattern into sub-patterns for recalling.

6 Numerical Examples

In this section we provide some numerical examples to better understand the functioning of the proposal. In the next section we give some real examples where we test the effectiveness of the proposal.

Example 6.1. Let be the following FSP, representing the five vowels of the Latin Alphabet (A, E, I, O and U; 1 is for the information, 0 is for background):

$$B^1=\begin{pmatrix}0&1&0\\1&0&1\\1&0&1\\1&1&1\\1&0&1\end{pmatrix}, B^2=\begin{pmatrix}1&1&1\\1&0&0\\1&1&0\\1&0&0\\1&1&1\end{pmatrix}, B^3=\begin{pmatrix}1&1&1\\0&1&0\\0&1&0\\0&1&0\\1&1&1\end{pmatrix}, B^4=\begin{pmatrix}1&1&1\\1&0&1\\1&0&1\\1&0&1\\1&1&1\end{pmatrix}, B^5=\begin{pmatrix}1&0&1\\1&0&1\\1&0&1\\1&0&1\\1&1&1\end{pmatrix}$$

Learning phase:

Step 1: Memory construction. We can use any associative memory. Let us use W associative memory reported in [6] useful to handle with subtractive noise. Just to remember, *W* memories make use of arithmetic subtraction between elements and min ($\wedge$) operator for memory building. For pattern recall they use arithmetic addition and the max ($\vee$) operator. For the details refer to [6]. Because each pattern is composed of five sub-patterns, and each of these sub-patterns is of size 3, we have then the next five memories:

$$W^1=\begin{pmatrix}0&-1&0\\-1&0&-1\\0&-1&0\end{pmatrix}, W^2=\begin{pmatrix}0&-1&0\\-1&0&-1\\-1&-1&0\end{pmatrix}, W^3=\begin{pmatrix}0&-1&0\\-1&0&-1\\-1&-1&0\end{pmatrix}, W^4=\begin{pmatrix}0&-1&0\\-1&0&-1\\-1&-1&0\end{pmatrix}, W^5=\begin{pmatrix}0&0&0\\-1&0&-1\\0&0&0\end{pmatrix}.$$

For the details of how W^1 to W^5 were obtained, the interested reader is refereed to [6].

Step 2: Construction of matrix V. As explained in section 5.1, we proceed with each row of V:

For pattern B^1 and first sub-pattern $b_1^1=\begin{pmatrix}0&1&0\end{pmatrix}$, $d=2$, thus $v_{11}=2$. For pattern B^1 and second base vector $b_2^1=\begin{pmatrix}1&0&1\end{pmatrix}$, $d=5$, thus $v_{12}=5$. If we continue we this procedure for the remaining base patterns of B^1 and the sub-patterns of base patterns B^2, B^3, B^4 and B^5:

$$V=\begin{pmatrix}2&5&5&7&5\\7&4&7&4&7\\7&2&2&2&7\\7&5&5&5&7\\5&5&5&5&7\end{pmatrix}.$$

This ends the learning stage.

Recalling phase:

Example 6.2. Recalling a pattern of the FSP. Let us take the first fundamental pattern B^1 of example 6.1. Let us proceed:

Step 1: As discussed in section 5.2: $Z = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \end{pmatrix}$.

Step 2: For pattern recall a W memory uses arithmetic addition between components and the max ($\vee$) operator. For the details, refer to [6]. Now for each b_k^1 of B^1, we have:

For $b_1^1 = \begin{pmatrix} 0 & 1 & 0 \end{pmatrix}$:
$$W^1 \wedge b_1^1 = \begin{pmatrix} 0 & -1 & 0 \\ -1 & 0 & -1 \\ 0 & -1 & 0 \end{pmatrix} \wedge \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} (0+0)\vee(-1+1)\vee(0+0) \\ (-1+0)\vee(0+1)\vee(-1+0) \\ (0+0)\vee(-1+1)\vee(0+0) \end{pmatrix} = \begin{pmatrix} 0\vee 0\vee 0 \\ -1\vee 1\vee -1 \\ 0\vee 0\vee 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}.$$

$010 = 2_{10}$. We look for the appearances of 2 now into first column of V. As can be seen it appears only in the element $v_{11} = 2$ of V, so: $Z = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \end{pmatrix}$.

For $b_2^1 = \begin{pmatrix} 1 & 0 & 1 \end{pmatrix}$:
$$W^2 \wedge b_2^1 = \begin{pmatrix} 0 & -1 & 0 \\ -1 & 0 & -1 \\ -1 & -1 & 0 \end{pmatrix} \wedge \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} (0+1)\vee(-1+0)\vee(0+1) \\ (-1+1)\vee(0+0)\vee(-1+1) \\ (-1+1)\vee(-1+0)\vee(0+1) \end{pmatrix} = \begin{pmatrix} 1\vee -1\vee 1 \\ 0\vee 0\vee 0 \\ 0\vee -1\vee 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}.$$

$101 = 5_{10}$. We look for the appearances of 5 now into second column of V. As can be seen it appears in the first, fourth and fifth positions of V, so: $Z = \begin{pmatrix} 2 & 0 & 0 & 1 & 1 \end{pmatrix}$.

If we continue this way, it can be easily shown that $Z = \begin{pmatrix} 5 & 0 & 0 & 2 & 2 \end{pmatrix}$.

Step 3: We finally get the index of the corresponding pattern as: $j = \arg\max_i (5,0,0,2,2) = 1$. Thus the desired pattern is pattern B^1, that is the pattern we were looking for.

Example 5.3. Recalling a pattern of the FSP given a distorted version of it. Let us now take the following distorting version of fundamental pattern B^1:

$$\overline{B}^1 = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{pmatrix}.$$

You can observe that in this case sub-patterns b_1^1 and b_2^1 appear distorted. The other three are not altered. Proceeding as before:

Step 1: As discussed in section 5.2: $Z = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \end{pmatrix}$.

Step 2: Now for each b_k^1 of B^1, we have:

For $\bar{b}_1^1 = \begin{pmatrix} 0 & 0 & 0 \end{pmatrix}$: $W^1 \wedge \bar{b}_1^1 = \begin{pmatrix} 0 & -1 & 0 \\ -1 & 0 & -1 \\ 0 & -1 & 0 \end{pmatrix} \wedge \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} (0+0)\vee(-1+0)\vee(0+0) \\ (-1+0)\vee(0+0)\vee(-1+0) \\ (0+0)\vee(-1+0)\vee(0+0) \end{pmatrix} = \begin{pmatrix} 0\vee -1\vee 0 \\ -1\vee 0\vee -1 \\ 0\vee -1\vee 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$.

$000 = 0_{10}$. We look for the appearances of 0 now into first column of V. As can be seen, it does not match any element of V, so: $Z = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \end{pmatrix}$.

For
$b_2^1 = \begin{pmatrix} 1 & 1 & 0 \end{pmatrix}$: $W^2 \wedge b_2^1 = \begin{pmatrix} 0 & -1 & 0 \\ -1 & 0 & -1 \\ -1 & -1 & 0 \end{pmatrix} \wedge \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} (0+1)\vee(-1+1)\vee(0+0) \\ (-1+1)\vee(0+1)\vee(-1+0) \\ (-1+1)\vee(-1+1)\vee(0+0) \end{pmatrix} = \begin{pmatrix} 1\vee 0\vee 0 \\ 0\vee 1\vee -1 \\ 0\vee 0\vee 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$.

$110 = 6_{10}$. We look for the appearances of 6 into second column of V. As can be seen, again it does not match any element of V, so: $Z = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 \end{pmatrix}$.

If we continue this way, it can be easily shown that $Z = \begin{pmatrix} 3 & 0 & 0 & 1 & 1 \end{pmatrix}$.

Step 3: We finally get the index of the corresponding pattern as: $j = \arg\max_i (3,0,0,1,1) = 1$. Again the recalled index coincides with the index of the desired pattern.

7 Results with Real Patterns

In this section the proposed methodology is tested with more realistic patterns. For this, we make use of the twelve patterns shown in Figure 2. Tests were performed with four associative memories: morphological associative memories **M** and **W** [6], and $\alpha\beta$ associative memories **M** and **W** [8]. Just to remember, **M** memories are good for additive noise and **W** memories are good for subtractive noise. For the details about the operation of both memories, the interested readers is refereed to [6] and [8].

Figure 3 shows the recalling results. As you can appreciate, 100 percent of perfect recall was obtained with **min** (**W**) memories morphological and $\alpha\beta$ from 5 to 15% of noise. From then on, the performance falls little by little. However as can be seen W memories show a better performance than **M** memories.

8 Conclusions and Ongoing Research

In this note we have described a simple but effective methodology for the recalling of patterns distorted by mixed noise. Instead of adopting the kernel method used in [6], or the median recently proposed in [11], we prefer to decompose each pattern into a set of sub-patterns. This way we can take advantage of the locality of affecting noise. During memory construction, sub-patterns sets are first used to build a set of memories. Next, during pattern recall a given pattern, possibly distorted by noise is also decomposed into its set of patterns. Each sub-pattern is operated by its corresponding memory. The filtered result is then used to recover the pattern to which the sub-pattern belongs. Finally, the index of the pattern is recovered by simple voting mechanism.

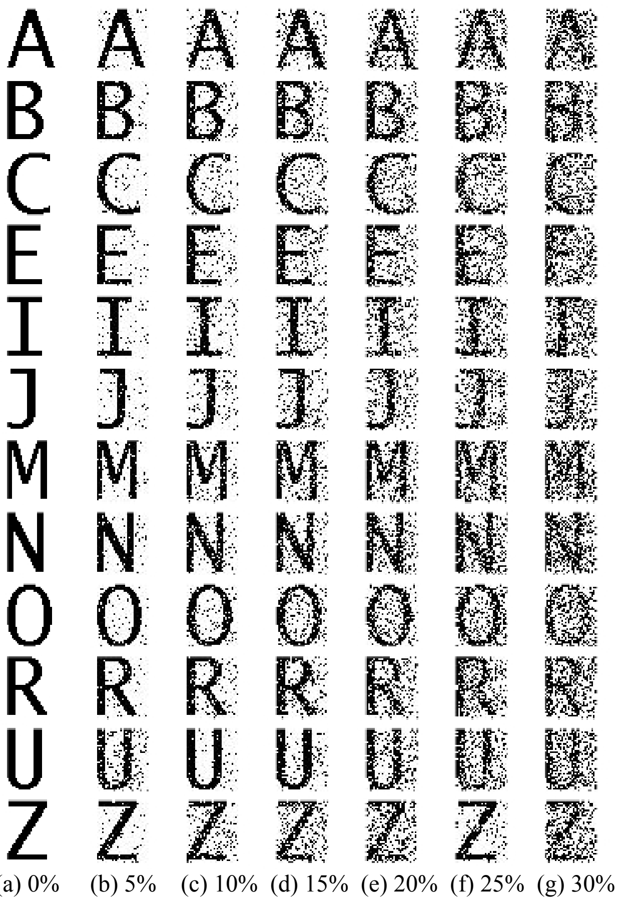


Figure 2. (a) Patterns used to test the proposal. Their size is 31x37 pixels. For testing these patterns were distorted by mixed noise at percentages of: (b) 5%, (c) 10%, (d) 15%, (e) 20%, (f) 25% and (g) 30% percent. One version of each letter is used.

Nowadays, we are looking for the formal propositions (Lemmas and Theorems and Corollaries) that specify the conditions under which the proposed methodology can be used to perfectly recover a given pattern from a distorted version of it. We are also looking for more real problems where the proposal can find applicability.

Acknowledgements. This work was economically supported by CGPI-IPN under grants 20050156 and CONACYT by means of grant 46805. H. Sossa specially thanks COTEPABE-IPN, CONACYT (Dirección de Asuntos Internacionales) and DAAD (Deutscher Akademischer Austauschdienst) for the economical support granted during research stay at Friedrich-Schiller University, Jena, Germany.

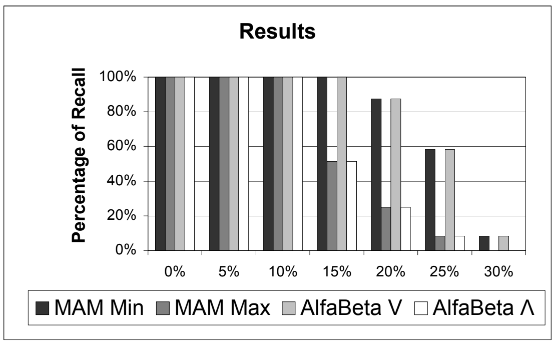


Figure 3. Recalling results obtained by applying the proposed methodology to the set of patterns shown in Fig. 2.

References

1. K. Steinbuch (1961). die Lernmatrix, Kyberneitk C-1,1,26-45.
2. J. Anderson (1972). A simple neural network generating an interactive memory, Mathematical Biosciences C-14, 197-220.
3. T. Kohonen (1972). Correlation matrix memories, IEEE Transactions Computers C-21, 4, 444-445.
4. G. Palm (1982). Neural Assemblies. An alternative approach to artificial intelligence. Studies of the brain function. Springer Verlag.
5. J. Hopfield (1982). Neural Network and Physicals systems with emergent collective computational abilities. Proceedings of the National Academy of Sciences, C-79, 2554-2558.
6. G. X. Ritter qt al. (1998). Morphological associative memories, IEEE Transactions on Neural Networks, C-9,281-293.
7. G. X. Ritter et al. (1999). Morphological bi-directional associative memories, Neural Networks, 12:851-867. .
8. C. Yáñez (2002). Associative Memories based on Order Relations and Binary Operators (In Spanish), PhD Thesis, Center for Computing Research-IPN.
9. G. X. Ritter et al. (2003), Reconstruction of patterns from noisy inputs using morphological associative memories. International Journal of Mathematical Imaging and Vision, 19(2), pp. 95-111.
10. H. Sossa et al. (2004). Extended Associative Memories for Recalling Gray Level Patterns. Lecture Notes on Computer Science 3287. Springer Verlag. Pp. 187-194.
11. H. Sossa et al. (2004). New Associative Memories to Recall Real-Valued Patterns. LNCS 3287. Springer Verlag. Pp. 195-202.
12. H. Sossa et al. (2005). Associative gray-level pattern processing using binary decomposition and a-b memories. Neural Processing Letters 22:85-111.
13. M. H. Hassouon (1993). Associated neural memories. Theory and implementation. Oxford University Press.