Fractals, Vol. 27, No. 4 (2019) 1950080 (13 pages)

DOI: 10.1142/S0218348X19500804

World Scientific
www.worldscientific.com

COMPUTATIONAL INTELLIGENCE FOR SHOEPRINT RECOGNITION

M. E. ACEVEDO MOSQUEDA,*,§ M. A. ACEVEDO MOSQUEDA,*
R. CARREÑO AGUILERA,† F. MARTINEZ ZUÑIGA,*
D. PACHECO BAUTISTA,† M. PATIÑO ORTIZ* and WEN YU‡
*Instituto Politécnico Nacional
Escuela Superior de Ingeniería Mecánica y Eléctrica Zacatenco
México City, México
†Universidad del Istmo, Cd. Universitaria S/N
70760 Tehuantepec, Oaxaca, México
‡Instituto Politécnico Nacional
CINVESTAV, México City, México
§eacevedo@ipn.mx

Received May 27, 2018
Revised January 9, 2019
Accepted April 1, 2019
Published July 12, 2019

Abstract

Shoeprint marks present valuable information for forensic investigators to resolve a crime. These marks can be helpful to find the brand of the shoe and can make the investigation easier. In this paper, we present an associative model-based algorithm to match noisy shoeprint patterns with a brand of shoe. The shoeprints are corrupted with additive, subtractive and mixed noises. A particular case of subtractive noise are partial shoeprints such as toe, heel, left-half and right-half prints. The Morphological Associative Memories (MAMs) were applied. Both memories, *max* and *min*, recognize noisy shoeprints corrupted with 98% additive and subtractive noise, respectively, with an effectiveness of 100%. The images corrupted with mixed noise were recognized when the additive or subtractive noise applied was greater than the mixed

§Corresponding author.

noise; in this case, the recalling was around 70%, otherwise, both memories failed to recognize the shoeprints.

Keywords: Forensic Science; Computational Forensics; Computational Intelligence; Associative Models; Morphological Associative Memories; Shoeprint Recognition.

1. INTRODUCTION

Forensic Science (FS) is the application of various sciences to those criminal and civil laws that are enforced by police agencies in a criminal justice system. The main goal of FS is to analyze physical evidence and provide expert witness testimony. It justifies the validity of conclusions drawn by the forensic investigation authorities.

Forensic-related technologies[1] are potentially capable of improving the lives of many people. Computational intelligence, a recently growing field of computer science, poses a prodigious opportunity to improve FS. Computational Intelligence techniques have been widely used in the domain of computer forensics, which have been successfully used in many real-world applications for a variety of engineering problems.

Computational Intelligence is based on human intelligence, and therefore it is expected to accomplish the task equal to or even beyond human proficiency. It relies on several key paradigms, such as evolutionary algorithms, neural networks, fuzzy systems, and multi-agent systems. In this work, we applied another paradigm named Associative Memories (AMs).

1.1. Related Work

Bouridane *et al.*[2] used an iterative process to reconstruct a shoeprint image from the coefficients obtained when an unknown image is decomposed by the use of fractals. The authors had a dataset of 145 images. When they applied small rotations, the result was 88% of recognition, and with small translations, they obtained 100% of effectiveness. In this paper, we use the terms: percent of recognition and percent of effectiveness indistinctly.

In another work, a Power Spectral Density (PSD) method and Zernike Moments (ZMs) were applied to recognize shoeprints.[3] The dataset had 400 real shoeprint images. As a first experiment, the PSD method was used to obtain the similarity between images by the means of correlation coefficients. They generated 5 partial images from 200 shoeprints. The effectiveness was of 86%. In the second experiment, they generated eight rotated, translated and scaled images from an original shoeprint. They applied ZM and the recognition was around 98%.

Gabor and Radon transforms were applied to recognize shoeprints with rotations.[4] They used 200 images and generated 3 different full prints and 4 partial prints. Radon transform was applied to estimate the shoeprint direction and Gabor coefficients were used to match images. The Euclidean distance algorithm performed the matching. The maximum cumulative match score for partial prints was 100% at rank 3 which means that the image was found among the three images with a maximum score, i.e. the image can be confounded with other two images. In Ref. 5, the authors added another experiment and tested the algorithm with some levels of Gaussian white and salt and pepper noises. With a signal to noise ratio (SNR) of 15.28 dB, the recognition rate was of 68.52% with salt and pepper noise and 67.67% with Gaussian white noise.

In Ref. 6, discrete cosine transform (DCT) was applied to obtain a vector of coefficients which are presented to Fisher's linear discriminant (FLD) and principal component analysis (PCA) to reduce the vector dimensionality. The dataset was composed of 235 shoeprint images. They used the Euclidean distance for the matching. First, the test image was corrupted by Gaussian noise with mean 0 and variance 0.01 and filtered with median filter. The performance without noise is optimum when the number of coefficients (from DCT) is around 32. The algorithm is immune to noise if the noise variance is in the order of 0.01.

Li *et al.*[7] proposed a method that first constructs different scale spaces in order to detect local extrema in the underlying shoeprint images. Then, they applied the scale-invariant feature transform (SIFT) to obtain keypoints that are invariant to scale, rotation, and translation. They used cross-correlation for matching. The dataset consists of 430 full-size shoeprints. They showed that their proposal can retrieve toe prints and heel prints more effectively than left-half prints and right-half prints,

and there is no difference in the results when noise is added.

Also, Gabor transform and ZMs were applied to recognize partial shoeprints.[8] The authors first preprocessed the images manually to highlight the original image features and to remove the background. They used normalized correlation coefficient as a similarity score. The Gabor transform was applied to obtain the textures, and the ZMs to represent the shoeprint patterns. They used 1225 images for training and 104 images for testing. They obtained a 68.04% of effectiveness with a 20-rank recognition rate.

Luostarinen and Lehmussola[9] created a set of shoeprints with 367 images and tested several algorithms to recognize partial shoeprints. The algorithms were PSD, Hu's Moment Invariants, Gabor Transform, Fourier–Mellin Transform, Mahalanobis Distance Map. Local Interest Points used Random Sample Consensus (RANSAC) and Spectral Correspondence of Local Interest Points. The RANSAC algorithm showed the best results for partial prints: heel, tip, toes, inner toes, and outer toes and for rotations of 30°, 45°, and 90°.

The Secondary Positioning based on the shoe print image feature[10] is applied to positioning a shoeprint, therefore, the recognition can be easier to perform and it is not necessary to apply algorithms that are invariant to rotations.

The SIFT, PCA, and RANSAC were applied to recognize partial shoeprints.[11] SIFT was used to obtain keypoints, then, PCA was applied to reduce the set dimension of the keypoints. The matching is performed with Euclidean distance. Finally, RANSAC removes mistakes due to noise and obtains the correct matching point pairs. The authors did not mention the number of shoeprints for their experiment. The results of effectiveness were 95%, 84%, 92%, 94% for toe, heel, left-half, and right-half, respectively.

A neural code descriptor[12] was applied for automatic shoeprint retrieval, the authors obtained a cumulative match score of 88.7% at top 10.

A denoising deep belief network (DBN) was applied to extract local features from partial, noisy shoeprint images and a spatial pyramid matching (SPM) was used to obtain a local-to-global matching score.[13] The dataset has 536 shoeprint images. This algorithm achieves a cumulative match score of 65.67% at top 10 which is 5.60% higher than the second best performing method.

Zhang *et al.* applied a convolutional neural network[14] to extract features for shoeprint retrieval. They use three datasets to test their method. In the first dataset, they randomly remove pixels from images to simulate dust, in the second dataset, the images are corrupted with Gaussian noise and pixel removal simulating blood, and finally, they added more Gaussian noise and pixel removal. They found the best results at rank 5 with the following percentages of recognition: 80.3% in dataset 2, 94.3% in datasets 2 and 3, and 96.2 in datasets 1 and 2.

In this work, we propose the use of Morphological AMs (MAMs) for recognizing shoeprints. We test our algorithm with images corrupted with additive, subtractive, and mixed noises. The partial shoeprints (toe, heel, left-half, and right-half) are considered as a particular case of subtractive noise.

2. MATERIALS AND METHODS

In this section, we describe the basic concepts of AMs, the theory of MAMs and finally, the algorithm of our proposal.

2.1. Associative Memories

An AM[15] is a system which takes a codified input or pattern and produces an output which can be either a class label or another pattern. One of the main advantages of AMs is that when they are correctly designed, they can accurately recover a pattern even if it has been altered. This robustness against alterations makes them attractive for applications in which the input patterns are likely to be noisy.

Two phases comprise the design of an AM: learning and recalling. In the learning phase, the memory is trained by associating input patterns x and output patterns y. After the AM was trained, output patterns can be recalled by presenting the input patterns to the memory. This task is performed by the recalling phase.

Formally, we can say that for a k integer and positive, the corresponding association will be denoted as (x^k, y^k). The associative memory $\mathbf{M}$ is represented by a matrix whose ijth component is m_{ij}. Memory $\mathbf{M}$ is generated from an *a priori* finite set of known AMs, called the fundamental or training set associations.

Each column vector that represents input and output patterns will have n and m components, respectively, which values belong to the set of real numbers.

If μ is an index, the fundamental set is represented as $\{(x^\mu, y^\mu) \,|\, \mu = 1, 2, \ldots, p\}$ with p as the set cardinality. The patterns that form the fundamental set are called fundamental patterns.

There are two types of AMs concerning to the nature of the input and output patterns.

A memory is *Autoassociative* if it holds that $x^\mu = y^\mu \; \forall \, \mu \in \{1, 2, \ldots, p\}$, then one of the requisites is that $n = m$.

A memory is *Heteroassociative* when $\exists \, \mu \in \{1, 2, \ldots, p\}$ for which $x^\mu \neq y^\mu$. Note that there can be heteroassociative memories with $n = m$.

2.2. Morphological Associative Memories

This paper is strongly based on the work of Ritter *et al.*[16] The fundamental difference between classic AMs (Lernmatrix,[17] Correlograph,[18] Linear Associator,[19] and Hopfield[20]) and MAMs[16] lies in the operational bases of the latter, which are the morphological operations: dilation and erosion. This model broke out of the traditional mold of classic memories which use conventional operations for vectors and matrices in learning phase and sum of multiplications for recovering patterns. MAMs change products to sums and sums to maximum or minimum in both phases.

The basic computations occurring in the proposed morphological network are based on the algebraic lattice structure $(\mathbf{R}, \vee, \wedge, +)$, where the symbols $\vee$ and $\wedge$ denote the binary operations of maximum and minimum, respectively. Using the lattice structure $(\mathbf{R}, \vee, \wedge, +)$, for an $m \times p$ matrix A and a $p \times n$ matrix B with entries from $\mathbf{R}$, the matrix product $C = A \nabla B$, also called the *max* product of A and B, is defined by

$$c_{ij} = \bigvee_{k=1}^{p} a_{ik} + b_{kj} = (a_{i1} + b_{1j}) \vee \cdots \vee (a_{ip} + b_{pj}). \tag{1}$$

The *min product* of A and B induced by the lattice structure is defined in a similar fashion. Specifically, the ijth entry of $C = A \Delta B$ is given by

$$c_{ij} = \bigwedge_{k=1}^{p} a_{ik} + b_{kj} = (a_{i1} + b_{1j}) \wedge \cdots \wedge (a_{ip} + b_{pj}). \tag{2}$$

Suppose we are given a vector pair $\mathbf{x} = (x_1, x_2, \ldots, x_n)^t \in \mathbf{R}^n$ and $\mathbf{y} = (y_1, y_2, \ldots, y_m)^t \in \mathbf{R}^m$. An associative morphological memory that will recall the vector when presented the vector is given as follows:

$$W = y \nabla (-x)^t = \begin{bmatrix} y_1 - x_1 & \cdots & y_1 - x_n \\ \vdots & \ddots & \vdots \\ y_m - x_1 & \cdots & y_m - x_n \end{bmatrix}. \tag{3}$$

Since W satisfies the equation $W \nabla \mathbf{x} = \mathbf{y}$ as can be verified by the simple computation in

$$W \nabla x = \begin{bmatrix} \bigvee_{i=1}^{n} (y_1 - x_i + x_i) \\ \vdots \\ \bigvee_{i=1}^{n} (y_m - x_i + x_i) \end{bmatrix} = y. \tag{4}$$

Henceforth, let $(\mathbf{x}^1, \mathbf{y}^1), (\mathbf{x}^2, \mathbf{y}^2), \ldots, (\mathbf{x}^p, \mathbf{y}^p)$ be p vector pairs with $x^k = (x_1^k, x_2^k, \ldots, x_n^k)^t \in \mathbf{R}^n$ and $y^k = (y_1^k, y_2^k, \ldots, y_m^k)^t \in \mathbf{R}^m$ for $k = 1, 2, \ldots, p$. For a given set of pattern associations $\{(\mathbf{x}^k, \mathbf{y}^k) \,|\, k = 1, 2, \ldots, p\}$, we define a pair of associated pattern matrices (X, Y), where $X = (\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^p)$ and $Y = (\mathbf{y}^1, \mathbf{y}^2, \ldots, \mathbf{y}^p)$. Thus, X is of dimension $n \times p$ with ijth entry x_i^j and Y is of dimension $m \times p$ with ijth entry y_i^j. Since $\mathbf{y}^k \nabla \, (-\mathbf{x}^k)^t = \mathbf{y}^k \, \Delta (-\mathbf{x}^k)^t$, the notational burden is reduced by denoting these identical morphological outer vector products by $\mathbf{y}^k \cdot (-\mathbf{x}^k)^t$. With each pair of matrices (X, Y), we associate two natural morphological $m \times n$ memories M and W defined by

$$M = \bigvee_{k=1}^{p} (y^k \cdot (-x^k)^t), \tag{5}$$

$$W = \bigwedge_{k=1}^{p} (y^k \cdot (-x^k)^t). \tag{6}$$

With these definitions, we present the algorithms for the training and recalling phases.

2.2.1. *Training phase*

(1) For each p association $(\mathbf{x}^\mu, \mathbf{y}^\mu)$, the minimum product is used to build the matrix

$\mathbf{y}^{\mu}\Delta(-\mathbf{x}^{\mu})^{t}$ of dimensions $m \times n$, where the input transposed negative pattern $\mathbf{x}^{\mu}$ is defined as $(-\mathbf{x}^{\mu})^{t} = (-x_1^{\mu}, -x_2^{\mu}, \ldots, -x_n^{\mu})$.

(2) The maximum and minimum operators ($\vee$ and $\wedge$) are applied to the p matrices to obtain M and W memories as Eqs. (5) and (6) show.

2.2.2. *Recalling phase*

In this phase, the minimum or maximum product, Δ or ∇, is applied between memories $\mathbf{M}$ or $\mathbf{W}$ and input pattern $\mathbf{x}^{\omega}$, where $\omega \in \{1, 2, \ldots, p\}$, to obtain the column vector $\mathbf{y}$ of dimension m as given by the following equations:

$$\mathbf{y} = M \,\Delta\, \mathbf{x}^{\omega}, \tag{7}$$

$$\mathbf{y} = W \,\nabla\, \mathbf{x}^{\omega}. \tag{8}$$

In this work, we will apply heteroassociative memories.

2.3. Types of Noise

MAMs deal with two types of noise: additive and subtractive but cannot deal with mixed noise.

The following example illustrates these three types of noise. Suppose we have the original byte:

Original byte	10011100	156
Additive noise	11111111	255
Subtractive noise	00000000	0
Mixed noise	11100100	228

We add additive noise when we replaced zeros for ones. In the case of the subtractive noise, we replace ones for zeros, and if we add additive and subtractive noises at the same time, we have mixed noise.

That is what it happens at bit level. In an image, these types of noise would appear as shown in Fig. 1.

From Fig. 1, we observe that the suppression of the heel can be interpreted as adding a white color in that part which means applying additive noise, i.e. place values of 255 (gray level) in the heel. On the other hand, in the third image (from left to right), we observe black and white colors added to the shoeprint which is the mixed noise. The last image represents the subtractive noise because we are placing values of zero in the same part of the shoeprint, i.e. we added black color.

Fig. 1 Illustration of the types of noise. From left to right: the original image, additive noise, subtractive noise, and mixed noise.

Fig. 2 Shoeprint of the Flexi brand.

2.4. Dataset

We created a shoeprint set of 53 images from different brands: Adidas, Andrea, Flexi, Polo, World, Ozono, Pirma Brasil, Wilson, Jordan, and Nike. The images have a fixed size: 50×50 pixels in Fig. 2, we show a sample of the set that is a Flexi shoeprint.

We have five different prints for each brand and for Wilson we obtained three prints. The images were preprocessed. At the beginning, the images had three channels (color images); therefore, we had to convert them into a single channel light intensity levels as shown in the following equation:

$$f(R, G, B) = 0.299R + 0.587G + 0.114B. \tag{9}$$

Now, the images are represented with matrices whose values are in the range of 0–255. Each matrix must be converted to a vector as shown in Fig. 3, where ω is the pattern index and r and s are the rows and columns of the image, respectively.

These vectors are the input patterns, the dimension is 2500 (50×50). The dimension of the output

patterns is 11 because we have 11 brands of shoes. The form of the vectors $\mathbf{y}$ to build the *max* and *min* memories is as follows, respectively. We applied the idea of Linear Associator,[19] and we selected the value of 500 because we found that a value greater than the greatest element in the dataset results in a better performance[21]:

$$y_{\text{class1}} = \begin{pmatrix} 500 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad y_{\text{class2}} = \begin{pmatrix} 0 \\ 500 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad y_{\text{class3}} = \begin{pmatrix} 0 \\ 0 \\ 500 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix},$$

$$y_{\text{class4}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 500 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad y_{\text{class5}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 500 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad y_{\text{class6}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 500 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix},$$

$$y_{\text{class7}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 500 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \ldots, y_{\text{class11}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 500 \end{pmatrix}, \quad \bar{y}_{\text{class1}} = \begin{pmatrix} -500 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix},$$

$$\bar{y}_{\text{class2}} = \begin{pmatrix} 0 \\ -500 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad \bar{y}_{\text{class3}} = \begin{pmatrix} 0 \\ 0 \\ -500 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad \bar{y}_{\text{class4}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ -500 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix},$$

$$\bar{y}_{\text{class5}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -500 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad \bar{y}_{\text{class6}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -500 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix},$$

$$\bar{y}_{\text{class7}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -500 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \ldots, \bar{y}_{\text{class11}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -500 \end{pmatrix}.$$

We defined two different types of output vectors to build two different MAMs: *max* and *min*, respectively.

The corresponding output vectors for each input vector is

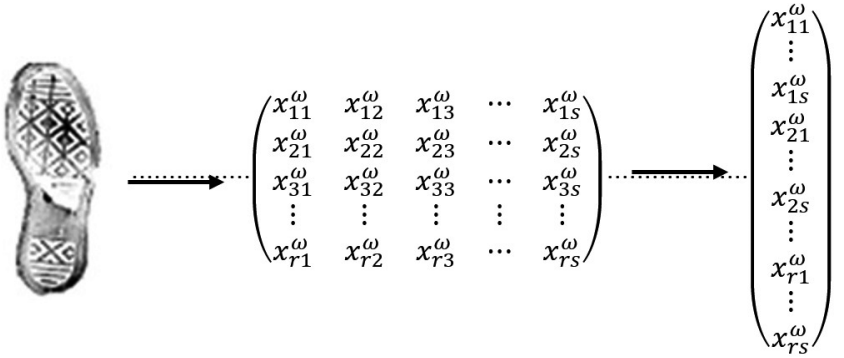

Fig. 3 The matrices that represent the images are converted to a vector to be processed by the memories.

$$\left.\begin{matrix} x^1_{\text{Adidas}} \\ x^2_{\text{Adidas}} \\ x^3_{\text{Adidas}} \\ x^4_{\text{Adidas}} \\ x^5_{\text{Adidas}} \end{matrix}\right\} \rightarrow \begin{cases} y_{\text{class1}}, \\ \bar{y}_{\text{class1}}, \end{cases} \quad \left.\begin{matrix} x^6_{\text{Andrea}} \\ x^7_{\text{Andrea}} \\ x^8_{\text{Andrea}} \\ x^9_{\text{Andrea}} \\ x^{10}_{\text{Andrea}} \end{matrix}\right\} \rightarrow \begin{cases} y_{\text{class2}}, \\ \bar{y}_{\text{class2}}, \end{cases} \quad \left.\begin{matrix} x^{11}_{\text{Flexi}} \\ x^{12}_{\text{Flexi}} \\ x^{13}_{\text{Flexi}} \\ x^{14}_{\text{Flexi}} \\ x^{15}_{\text{Flexi}} \end{matrix}\right\} \rightarrow \begin{cases} y_{\text{class3}}, \\ \bar{y}_{\text{class3}}, \end{cases} \quad \left.\begin{matrix} x^{16}_{\text{Polo}} \\ x^{17}_{\text{Polo}} \\ x^{18}_{\text{Polo}} \\ x^{19}_{\text{Polo}} \\ x^{20}_{\text{Polo}} \end{matrix}\right\} \rightarrow \begin{cases} y_{\text{class4}}, \\ \bar{y}_{\text{class4}}, \end{cases}$$

$$\left.\begin{matrix} x^{21}_{\text{World}} \\ x^{22}_{\text{World}} \\ x^{23}_{\text{World}} \\ x^{24}_{\text{World}} \\ x^{25}_{\text{World}} \end{matrix}\right\} \rightarrow \begin{cases} y_{\text{class5}}, \\ \bar{y}_{\text{class5}}, \end{cases} \quad \left.\begin{matrix} x^{26}_{\text{Ozono}} \\ x^{27}_{\text{Ozono}} \\ x^{28}_{\text{Ozono}} \\ x^{29}_{\text{Ozono}} \\ x^{30}_{\text{Ozono}} \end{matrix}\right\} \rightarrow \begin{cases} y_{\text{class6}}, \\ \bar{y}_{\text{class6}}, \end{cases} \quad \left.\begin{matrix} x^{31}_{\text{Pirma}} \\ x^{32}_{\text{Pirma}} \\ x^{33}_{\text{Pirma}} \\ x^{34}_{\text{Pirma}} \\ x^{35}_{\text{Pirma}} \end{matrix}\right\} \rightarrow \begin{cases} y_{\text{class7}}, \\ \bar{y}_{\text{class7}}, \end{cases} \quad \left.\begin{matrix} x^{36}_{\text{Brasil}} \\ x^{37}_{\text{Brasil}} \\ x^{38}_{\text{Brasil}} \\ x^{39}_{\text{Brasil}} \\ x^{40}_{\text{Brasil}} \end{matrix}\right\} \rightarrow \begin{cases} y_{\text{class8}}, \\ \bar{y}_{\text{class8}}, \end{cases}$$

$$\left.\begin{matrix} x^{41}_{\text{Jordan}} \\ x^{42}_{\text{Jordan}} \\ x^{43}_{\text{Jordan}} \\ x^{44}_{\text{Jordan}} \\ x^{45}_{\text{Jordan}} \end{matrix}\right\} \rightarrow \begin{cases} y_{\text{class9}}, \\ \bar{y}_{\text{class9}}, \end{cases} \quad \left.\begin{matrix} x^{46}_{\text{Nike}} \\ x^{47}_{\text{Nike}} \\ x^{48}_{\text{Nike}} \\ x^{49}_{\text{Nike}} \\ x^{50}_{\text{Nike}} \end{matrix}\right\} \rightarrow \begin{cases} y_{\text{class10}}, \\ \bar{y}_{\text{class10}}, \end{cases} \quad \left.\begin{matrix} x^{51}_{\text{Wilson}} \\ x^{52}_{\text{Wilson}} \\ x^{53}_{\text{Wilson}} \end{matrix}\right\} \rightarrow \begin{cases} y_{\text{class11}}, \\ \bar{y}_{\text{class11}}. \end{cases}$$

We have a main vector $\mathbf{Y}$ where the shoe brands are stored.

$$Y = \left\{ \begin{matrix} \text{Adidas} \\ \text{Andrea} \\ \text{Flexi} \\ \text{Polo} \\ \text{World} \\ \text{Ozono} \\ \text{Pirma} \\ \text{Brasil} \\ \text{Jordan} \\ \text{Nike} \\ \text{Wilson} \end{matrix} \right\}.$$

Now, an illustrative example for training and recalling phases is presented.

Suppose we have the following three pairs ($p = 3$) of vectors to be associated. The dimension of the input vectors are 15 ($n = 4$) and the dimension of output vectors is 3 ($m = 3$). These patterns will be used to build a *max*-type MAM:

$$x^1 = \begin{pmatrix} 0 \\ 255 \\ 127 \\ 48 \end{pmatrix} \rightarrow y^1 = \begin{pmatrix} 500 \\ 0 \\ 0 \end{pmatrix}, \quad x^2 = \begin{pmatrix} 112 \\ 25 \\ 201 \\ 15 \end{pmatrix} \rightarrow y^2 = \begin{pmatrix} 0 \\ 500 \\ 0 \end{pmatrix},$$

$$x^3 = \begin{pmatrix} 10 \\ 23 \\ 167 \\ 220 \end{pmatrix} \rightarrow y^3 = \begin{pmatrix} 0 \\ 0 \\ 500 \end{pmatrix}.$$

We associate the first pair of patterns.

$$y^1 \cdot (-x^1)^t = \begin{pmatrix} 500 \\ 0 \\ 0 \end{pmatrix} \cdot -(0 \quad 255 \quad 127 \quad 48)$$

$$= \begin{pmatrix} 500-0 & 500-255 & 500-127 & 500-48 \\ 0-0 & 0-255 & 0-127 & 0-48 \\ 0-0 & 0-255 & 0-127 & 0-48 \end{pmatrix},$$

$$y^1 \cdot (-x^1)^t = \begin{pmatrix} 500 & 245 & 373 & 452 \\ 0 & -255 & -127 & -48 \\ 0 & -255 & -127 & -48 \end{pmatrix}.$$

In the same way, we associate the second and the third pairs of patterns

$$y^2 \cdot (-x^2)^t = \begin{pmatrix} 0 \\ 500 \\ 0 \end{pmatrix} \cdot -(112 \quad 25 \quad 201 \quad 15)$$

$$= \begin{pmatrix} 0-112 & 0-25 & 0-201 & 0-15 \\ 500-112 & 500-25 & 500-201 & 500-15 \\ 0-112 & 0-25 & 0-201 & 0-15 \end{pmatrix},$$

$$y^2 \cdot (-x^2)^t = \begin{pmatrix} -112 & -25 & -201 & -15 \\ 388 & 475 & 299 & 485 \\ -112 & -25 & -201 & -15 \end{pmatrix},$$

$$y^3 \cdot (-x^3)^t = \begin{pmatrix} 0 \\ 0 \\ 500 \end{pmatrix} \cdot -(10 \quad 23 \quad 167 \quad 220)$$

$$= \begin{pmatrix} 0-10 & 0-23 & 0-167 & 0-220 \\ 0-10 & 0-23 & 0-167 & 0-220 \\ 500-10 & 500-23 & 500-167 & 500-220 \end{pmatrix},$$

$$y^3 \cdot (-x^3)^t = \begin{pmatrix} -10 & -23 & -167 & -220 \\ -10 & -23 & -167 & -220 \\ 490 & 477 & 333 & 280 \end{pmatrix}.$$

Now, we build the *max*-type MAM by finding the maximum element of each association as follows:

$$M = \begin{pmatrix} 500 & 245 & 373 & 452 \\ 0 & -255 & -127 & -48 \\ 0 & -255 & -127 & -48 \end{pmatrix}$$

$$\bigvee \begin{pmatrix} -112 & -25 & -201 & -15 \\ 388 & 475 & 299 & 485 \\ -112 & -25 & -201 & -15 \end{pmatrix}$$

$$\bigvee \begin{pmatrix} -10 & -23 & -167 & -220 \\ -10 & -23 & -167 & -220 \\ 490 & 477 & 333 & 280 \end{pmatrix},$$

$$M = \begin{pmatrix} 500 & 245 & 373 & 452 \\ 388 & 475 & 299 & 485 \\ 490 & 477 & 333 & 280 \end{pmatrix}.$$

In the recalling phase, we present the three input patterns to the *max* MAM:

$$M\Delta x^1 = \begin{pmatrix} 500 & 245 & 373 & 452 \\ 388 & 475 & 299 & 485 \\ 490 & 477 & 333 & 280 \end{pmatrix} \Delta \begin{pmatrix} 0 \\ 255 \\ 127 \\ 48 \end{pmatrix}$$

$$= \begin{pmatrix} (500+0) \wedge (245+255) \wedge (373+127) \wedge (452+48) \\ (388+0) \wedge (475+255) \wedge (299+127) \wedge (485+48) \\ (490+0) \wedge (477+255) \wedge (333+127) \wedge (280+48) \end{pmatrix},$$

$$M\Delta x^1 = \begin{pmatrix} 500 \wedge 500 \wedge 500 \wedge 500 \\ 388 \wedge 730 \wedge 426 \wedge 533 \\ 490 \wedge 732 \wedge 460 \wedge 328 \end{pmatrix} = \begin{pmatrix} 500 \\ 388 \\ 328 \end{pmatrix}.$$

We equate to zero each element that is not equal to 500, then which obtains

$$M\Delta x^1 = \begin{pmatrix} 500 \\ 388 \\ 328 \end{pmatrix} \quad \rightarrow \quad \begin{pmatrix} 500 \\ 0 \\ 0 \end{pmatrix} = y^1.$$

It is observed that we recalled the corresponding output pattern. Referring to our method, it can be said that input pattern x^1 corresponds to class 1 because the value of 500 is located at the first element. If we refer to the main vector $\mathbf{Y}$, and if we have just three brands of shoes, this result will indicate that we recalled Adidas brand.

Now, the second input pattern to the *max* MAM is presented:

$$M\Delta x^2 = \begin{pmatrix} 500 & 245 & 373 & 452 \\ 388 & 475 & 299 & 485 \\ 490 & 477 & 333 & 280 \end{pmatrix} \Delta \begin{pmatrix} 112 \\ 25 \\ 201 \\ 15 \end{pmatrix}$$

$$= \begin{pmatrix} (500+112) \wedge (245+25) \wedge (373+201) \wedge (452+15) \\ (388+112) \wedge (475+25) \wedge (299+201) \wedge (485+15) \\ (490+112) \wedge (477+25) \wedge (333+201) \wedge (280+15) \end{pmatrix},$$

$$M\Delta x^2 = \begin{pmatrix} 612 \wedge 270 \wedge 574 \wedge 467 \\ 500 \wedge 500 \wedge 500 \wedge 500 \\ 602 \wedge 502 \wedge 531 \wedge 295 \end{pmatrix} = \begin{pmatrix} 270 \\ 500 \\ 295 \end{pmatrix}.$$

In a similar way, the values that are different of 500 are equated to zero:

$$M\Delta x^2 = \begin{pmatrix} 270 \\ 500 \\ 295 \end{pmatrix} \rightarrow \begin{pmatrix} 0 \\ 500 \\ 0 \end{pmatrix} = y^2.$$

The recalling pattern is the output vector y^2. We could assume that input pattern x^2 corresponds to class 2.

Finally, we present x^3 to *max* MAM:

$$M\Delta x^3 = \begin{pmatrix} 500 & 245 & 373 & 452 \\ 388 & 475 & 299 & 485 \\ 490 & 477 & 333 & 280 \end{pmatrix} \Delta \begin{pmatrix} 10 \\ 23 \\ 167 \\ 220 \end{pmatrix}$$

$$= \begin{pmatrix} (500+10) \wedge (245+23) \wedge (373+167) \wedge (452+220) \\ (388+10) \wedge (475+23) \wedge (299+167) \wedge (485+220) \\ (490+10) \wedge (477+23) \wedge (333+167) \wedge (280+220) \end{pmatrix},$$

$$M\Delta x^3 = \begin{pmatrix} 510 \wedge 268 \wedge 540 \wedge 672 \\ 348 \wedge 498 \wedge 466 \wedge 705 \\ 500 \wedge 500 \wedge 500 \wedge 500 \end{pmatrix} = \begin{pmatrix} 268 \\ 348 \\ 500 \end{pmatrix}.$$

We equate to zero the values that are not equal to 500, which gives

$$M\Delta x^3 = \begin{pmatrix} 268 \\ 348 \\ 500 \end{pmatrix} \rightarrow \begin{pmatrix} 0 \\ 0 \\ 500 \end{pmatrix} = y^3.$$

Observing that the recalling vector corresponds to the third output vector, this pattern could belong to the third class because the value of 500 is in the third element.

Then, we illustrate the building of the *min* MAM with the following example. We take the latter example as a base but change the output vectors as follows:

$$x^1 = \begin{pmatrix} 0 \\ 255 \\ 127 \\ 48 \end{pmatrix} \rightarrow \bar{y}^1 = \begin{pmatrix} -500 \\ 0 \\ 0 \end{pmatrix},$$

$$x^2 = \begin{pmatrix} 112 \\ 25 \\ 201 \\ 15 \end{pmatrix} \rightarrow \bar{y}^2 = \begin{pmatrix} 0 \\ -500 \\ 0 \end{pmatrix},$$

$$x^3 = \begin{pmatrix} 10 \\ 23 \\ 167 \\ 220 \end{pmatrix} \rightarrow \bar{y}^3 = \begin{pmatrix} 0 \\ 0 \\ -500 \end{pmatrix}.$$

As before, we performed the associations of all the pairs of patterns:

$$\bar{y}^1 \cdot (-x^1)^t = \begin{pmatrix} -500 \\ 0 \\ 0 \end{pmatrix} \cdot - \begin{pmatrix} 0 & 255 & 127 & 48 \end{pmatrix}$$

$$= \begin{pmatrix} -500-0 & -500-255 & -500-127 & -500-48 \\ 0-0 & 0-255 & 0-127 & 0-48 \\ 0-0 & 0-255 & 0-127 & 0-48 \end{pmatrix},$$

$$\bar{y}^1 \cdot (-x^1)^t = \begin{pmatrix} -500 & -755 & -627 & -548 \\ 0 & -255 & -127 & -48 \\ 0 & -255 & -127 & -48 \end{pmatrix},$$

$$\bar{y}^2 \cdot \Delta(-x^2)^t = \begin{pmatrix} 0 \\ -500 \\ 0 \end{pmatrix} \cdot - \begin{pmatrix} 112 & 25 & 201 & 15 \end{pmatrix}$$

$$= \begin{pmatrix} 0-112 & 0-25 & 0-201 & 0-15 \\ -500-112 & -500-25 & -500-201 & -500-15 \\ 0-112 & 0-25 & 0-201 & 0-15 \end{pmatrix},$$

$$\bar{y}^2 \cdot \Delta(-x^2)^t = \begin{pmatrix} -112 & -25 & -201 & -15 \\ -612 & -525 & -701 & -515 \\ -112 & -25 & -201 & -15 \end{pmatrix},$$

$$\bar{y}^3 \cdot (-x^3)^t = \begin{pmatrix} 0 \\ 0 \\ -500 \end{pmatrix} \cdot - \begin{pmatrix} 10 & 23 & 167 & 220 \end{pmatrix}$$

$$= \begin{pmatrix} 0-10 & 0-23 & 0-167 & 0-220 \\ 0-10 & 0-23 & 0-167 & 0-220 \\ -500-10 & -500-23 & -500-167 & -500-220 \end{pmatrix},$$

$$\bar{y}^3 \cdot (-x^3)^t = \begin{pmatrix} -10 & -23 & -167 & -220 \\ -10 & -23 & -167 & -220 \\ -510 & -523 & -667 & -720 \end{pmatrix}.$$

With these three associations, we build the *min* MAM:

$$W = \begin{pmatrix} -500 & -755 & -627 & -548 \\ 0 & -255 & -127 & -48 \\ 0 & -255 & -127 & -48 \end{pmatrix}$$

$$\bigwedge \begin{pmatrix} -112 & -25 & -201 & -15 \\ -612 & -525 & -701 & -515 \\ -112 & -25 & -201 & -15 \end{pmatrix}$$

$$\bigwedge \begin{pmatrix} -10 & -23 & -167 & -220 \\ -10 & -23 & -167 & -220 \\ -510 & -523 & -667 & -720 \end{pmatrix},$$

$$W = \begin{pmatrix} -500 & -755 & -627 & -548 \\ -612 & -525 & -701 & -515 \\ -510 & -523 & -667 & -720 \end{pmatrix}.$$

The next step is to perform the recalling phase by presenting all the input patterns to the *min* MAM. After recalling all the patterns, we equate to zero the values that are different from -500.

$$W\nabla x^1 = \begin{pmatrix} -500 & -755 & -627 & -548 \\ -612 & -525 & -701 & -515 \\ -510 & -523 & -667 & -720 \end{pmatrix} \nabla \begin{pmatrix} 0 \\ 255 \\ 127 \\ 48 \end{pmatrix}$$

$$= \begin{pmatrix} (-500+0) \vee (-755+255) \vee (-627+127) \vee (-548+48) \\ (-612+0) \vee (-525+255) \vee (-701+127) \vee (-515+48) \\ (-510+0) \vee (-523+255) \vee (-667+127) \vee (-720+48) \end{pmatrix},$$

$$W \nabla x^1 = \begin{pmatrix} -500 \vee (-500) \vee (-500) \vee (-500) \\ -612 \vee (-270) \vee (-574) \vee (-467) \\ -510 \vee (-268) \vee (-540) \vee (-672) \end{pmatrix} = \begin{pmatrix} -500 \\ -270 \\ -268 \end{pmatrix},$$

$$W \nabla x^1 = \begin{pmatrix} -500 \\ -270 \\ -268 \end{pmatrix} \rightarrow \begin{pmatrix} -500 \\ 0 \\ 0 \end{pmatrix} = \bar{y}^1,$$

$$W \nabla x^2 = \begin{pmatrix} -500 & -755 & -627 & -548 \\ -612 & -525 & -701 & -515 \\ -510 & -523 & -667 & -720 \end{pmatrix} \nabla \begin{pmatrix} 112 \\ 25 \\ 201 \\ 15 \end{pmatrix}$$

$$= \begin{pmatrix} (-500+112) \vee (-755+25) \vee (-627+201) \vee (-548+15) \\ (-612+112) \vee (-525+25) \vee (-701+201) \vee (-515+15) \\ (-510+112) \vee (-523+25) \vee (-667+201) \vee (-720+15) \end{pmatrix},$$

$$W \nabla x^2 = \begin{pmatrix} -388 \vee (-730) \vee (-426) \vee (-533) \\ -500 \vee (-500) \vee (-500) \vee (-500) \\ -398 \vee (-498) \vee (-466) \vee (-705) \end{pmatrix} = \begin{pmatrix} -388 \\ -500 \\ -398 \end{pmatrix},$$

$$W \nabla x^2 = \begin{pmatrix} -388 \\ -500 \\ -398 \end{pmatrix} \rightarrow \begin{pmatrix} 0 \\ -500 \\ 0 \end{pmatrix} = \bar{y}^2,$$

$$W \nabla x^3 = \begin{pmatrix} -500 & -755 & -627 & -548 \\ -612 & -525 & -701 & -515 \\ -510 & -523 & -667 & -720 \end{pmatrix} \nabla \begin{pmatrix} 10 \\ 23 \\ 167 \\ 220 \end{pmatrix}$$

$$= \begin{pmatrix} (-500+10) \vee (-755+23) \vee (-627+167) \vee (-548+220) \\ (-612+10) \vee (-525+23) \vee (-701+167) \vee (-515+220) \\ (-510+10) \vee (-523+23) \vee (-667+167) \vee (-720+220) \end{pmatrix},$$

$$W \nabla x^3 = \begin{pmatrix} -490 \vee (-732) \vee (-460) \vee (-328) \\ -602 \vee (-502) \vee (-534) \vee (-295) \\ -500 \vee (-500) \vee (-500) \vee (-500) \end{pmatrix} = \begin{pmatrix} -328 \\ -295 \\ -500 \end{pmatrix},$$

$$W \nabla x^3 = \begin{pmatrix} -328 \\ -295 \\ -500 \end{pmatrix} \rightarrow \begin{pmatrix} 0 \\ 0 \\ -500 \end{pmatrix} = \bar{y}^3.$$

From the results, it can be observed that we correctly recalled the corresponding output patterns. If we were applying our proposal, the input pattern x^1 would belong to the class 1, x^2 would correspond to class 2, and the third input pattern would belong to class 3.

3. RESULTS

We used an HP-Omen 17-w00la laptop with a processor Intel i7 quad core, and the system was implemented with the programming language Visual Studio 2013 C#.

The first experiment consisted in training both memories, *max* and *min*, with 53 images. Then, we presented all images to the memories which correctly classified all shoeprints. Therefore, the effectiveness was 100%, consequently, the memory did not have a forgetting factor, meaning that our memory recalled all the patterns with which it was trained.

The second experiment consisted in applying additive, subtractive, and mixed noises to the images. The goal of applying noise was to simulate shoe wear or spots in the shoe sole.

We started with a 20% of additive and subtractive noises to all images and presented the *max* and *min* MAMs. Then we applied 50%, 90%, 95%, 98%, and 99% of both types of noise: additive and subtractive. Figure 4 shows the different levels of subtractive noise applied to an Adidas shoeprint.

The process for adding the noise was as follows. We randomly selected I number of coordinates of the images by the means of the follwing equation:

$$I = \frac{w * h * \text{noise}}{100}, \tag{10}$$

where w is the image weight, h is the image height and noise is the percentage of the noise to add.

If we add additive noise, we placed a value of 255 in the selected coordinate. For subtractive noise, we placed a 0. In the case of mixed noise, the value (0 or 255) is randomly selected.

In Table 1, we show the results of effectiveness of classification from the experiment in Fig. 4.

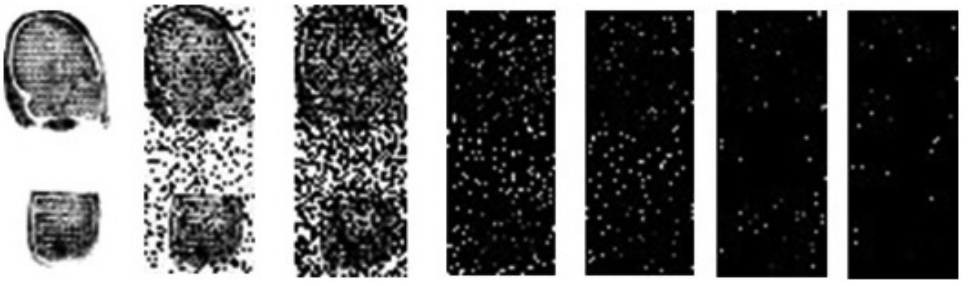

Fig. 4 From left to right: the original image, image with 20%, 50%, 90%, 95%, 98%, and 99% of subtractive noise.

Table 1 Results of Effectiveness of Classification When Different Levels of Subtractive Noise Were Applied to 53 Images.

Subtractive Noise (%)	Effectiveness of Classification (%)	
	max MAM	min MAM
20	0	100
50	0	100
90	0	100
95	0	100
98	0	100
99	0	0

From Table 1, as was expected, we observe that *max* memory cannot deal with subtractive noise. On the other hand, *min* MAM correctly recalled or classified all the noisy images even when they had 98% noise.

In Fig. 5, an example of noisy images with additive noise is shown. The noise levels are 20%, 50%, 90%, 95%, 98%, and 99%.

Table 2 shows the respective effectiveness when the different levels of additive noise were applied.

Max MAM recalled the 53 images even with 98% additive noise while *min* MAM had zero effectiveness with all levels of noise.

Some papers present the results from recalling partial shoeprints as toe, heel, left-half, and right-half. The above results confirm that our proposal is able to recognize partial shoeprints too because a partial pattern represents additive noise.

In Fig. 6, we show the original Adidas shoeprint and the noisy version of this image. It was added with 10%, 20%, 50% 80%, 90%, and 95% of mixed noise.

We know that *max* and *min* MAM can deal with additive and subtractive noises, respectively, but none can deal with mixed noise. We tried to solve this problem by adding subtractive and additive noises to those images corrupted with mixed noise. The percentage of adding noise (subtractive and additive) was greater than the percentage of mixed noise. For example, a corrupted image with 10% mixed noise was added with 30% subtractive and additive noises. The goal of this action is to counteract the effect of the mixed noise and to just have subtractive noise or additive noises. In this way, an MAM can deal with that noisy image.

Table 2 Results of Effectiveness of Classification When Different Levels of Additive Noise Were Applied to 53 Images.

Additive Noise (%)	Effectiveness of Classification (%)	
	max MAM	min MAM
20	100	0
50	100	0
90	100	0
95	100	0
98	100	0
99	0	0

Fig. 6 On the left, we can observe the original image and on the right, we show the noisy image with mixed noise.

The results are shown in Tables 3 and 4.

Both Tables 3 and 4 show that when the added noise (additive or subtractive) is greater than the mixed noise, the percentage of classification is

Fig. 5 From left to right: the original image, image with 20%, 50%, 90%, 95%, 98%, and 99% of additive noise.

Table 3 Results of Effectiveness of Classification When Different Levels of Mixed Noise Were Applied to 53 Images and the Images Were Corrupted with Additive Noise.

Mixed Noise (%)	Additive Noise (%)	Effectiveness of Classification (%)	
		min MAM	max MAM
10	30	0	72
20	40	0	71
50	70	0	72
80	80	0	0
90	90	0	0
95	95	0	1

Table 4 Results of Effectiveness of Classification When Different Levels of Mixed Noise Were Applied to 53 Images and the Images Were Corrupted with Subtractive Noise.

Mixed Noise (%)	Subtractive Noise (%)	Effectiveness of Classification (%)	
		min MAM	max MAM
10	30	70	0
20	40	73	0
50	70	74	0
80	80	1	0
90	90	1	0
95	95	0	0

around 70%, but when they are equal, none of the memories can recognize the patterns.

In Table 5, we show a comparison among other algorithms. We must highlight that these works did not use the same dataset. Some of them report their results based on the cumulative match score which is the probability of a match estimated by determining the proportion of times during trials of the system a database pattern appeared in the first n% of the sorted patterns and is from the same pattern category as the reference image. In other words, accuracy of a matching system could be taken to mean that for a given query image, the most similar image in the database is retrieved as the top ranking (rank 1).

From Table 5, we can observe that one advantage of our method is that it can deal with big amounts of additive and subtractive noises, but the performance is very poor as the images are corrupted with mixed noise.

4. DISCUSSION

We applied MAMs (*max* and *min*) to recognize the shoeprints. Both memories could recall 53 images that were trained. We simulated the wear and tear of shoes by adding additive, subtractive, and mixed noises.

When we added additive noise, *max* MAM recognized all images even when the level of noise was of 98%. On the other hand, *min* MAM handled high levels of subtractive noise. The maximum level of subtractive noise was 98%.

Clearly, we observed that none of the memories could recall any of the images when more than 98% of additive or subtractive noise was added.

In the case of mixed noise, the memories recognized the images with a 70% of effectiveness when

Table 5 Comparisons Among Algorithms Working with Partial Print Images: Toe, Heel, Left-Half and Right-Half.

Method	Num Images	Partial Prints	Type of Noise	% of Recognition
Gabor and Radon transforms	200 subjects with 3 different full prints and 4 partial prints	Toe and heel	None	100 at rank 2 for heel and at rank 3 for toe
SIFT	430	Toe, heel, left-half, and right-half	None	98 rank 2 for heel and toe and 87 for right and left-halves
PCA–SIFT	—	Toe, heel, left-half, and right-half	None	95 for toe, 84 for heel, 92 for left-half, and 94 for right-half
MAM	53	Toe, heel, left-half, and right-half	Additive and subtractive	100 for all the partial prints when the images are corrupted until 98%

the percentage of noise (additive and subtractive) is greater than mixed noise.

Both memories, *min* and *max*, showed to be an adequate tool to recognize shoeprints. In the particular case, when the images are corrupted with subtractive noise, we assure that our proposal can recall partial images as toe, heel, left-half, and right-half because they represent special cases of subtractive noise.

It should be highlighted that if MAM memories could recognize the original images they surely will recognize noisy versions added with even 98% of additive and subtractive noises.

REFERENCES

1. P. S. Fajri, L. Pratiwi, A. Abraham and M. A. Kamilah, Computational intelligence in digital forensics: Forensic investigation and applications, *Stud. Comput. Intell.* **555** (2014) 1–16.
2. A. Bouridane, A. Alexander, M. Nibouche and D. Crookes, Application of fractals to the detection and classification of shoeprints methods, in *Int. Conf. Image Processing* (IEEE, New York, 2000), pp. 474–477.
3. R. Xiao and P. Shi, Computerized matching of shoeprints based on sole pattern, in *Int. Workshop of Computational Forensics* (Springer, Berlin, 2008), pp. 96–104.
4. M. P. Deshmukh and M. P. Pradeep, Automatic shoeprint matching system for crime scene investigation, *Int. J. Comput. Sci. Commun. Technol.* **2**(1) (2009) 281–287.
5. M. P. Pradeep and J. V. Kulkarni, Rotation and intensity invariant shoeprint matching using Gabor transform with application to forensic science, *Pattern Recognit.* **42** (2009) 1308–1317.
6. S. Rathinavel and S. Arumugam, Full shoe print recognition based on pass band DCT and partial shoe print identification using overlapped block method for degraded images, *Int. J. Comput. Appl.* **26**(8) (2011) 16–21.
7. Z. Li, C. Wei, Y. Li and T. Sun, Research of shoeprint image stream retrieval algorithm with scale-invariance feature transform, in *2nd Int. Conf. Multimedia Technology* (*ICMT2011*) (IEEE, New York, 2011), pp. 5488–5491.
8. X. Kong, C. Yang and Z. Zheng, A novel method for shoeprint recognition in crime scenes, in *Chinese Conf. Biometric Recognition* (Springer, Cham, 2014), pp. 498–505.
9. T. Luostarinen and A. Lehmussola, Measuring the accuracy of automatic shoeprint recognition methods, *J. Forensic Sci.* **59**(6) (2014) 1627–1634.
10. Y. Li and H. X. Wang, A secondary positioning algorithm of the shoe print, in *Int. Conf. Network and Information Systems for Computers* (IEEE, New York, 2015), pp. 560–563.
11. Y. Dong, Matching method of partial shoeprint images based on PCA-SIFT algorithm, *Int. J. Eng. Res. Sci.* **2**(10) (2016) 167–171.
12. J. Cui, X. Zhao and D. Li, An automatic shoeprint retrieval method using neural codes for commercial shoeprint scanners, *Commun. Comput. Inf. Sci.* **772** (2017) 158–169.
13. J. Cui, X. Zhao, N. Liu, S. Morgachev and D. Li, Robust shoeprint retrieval method based on local-to-global feature matching for real crime scenes, *J. Forensic Sci.* **64**(2) (2019) 422–430.
14. Y. Zhang, H. Fu, E. Dellandréa and L. Chen, Adapting convolutional neural networks on the shoeprint retrieval for forensic use, in *Chinese Conf. Biometric Recognition* (Springer, Cham, 2017), pp. 520–527.
15. A. Ferreira, C. Yáñez-Márquez, M. Aldape and I. López, Evolutionary approach to feature selection with associative models, *Res. Comput. Sci.* **78** (2014) 111–122.
16. G. X. Ritter, P. Sussner and J. L. Diaz de León, Morphological associative memories, *IEEE Trans. Neural Netw.* **9**(2) (1998) 281–293.
17. K. Steinbuch, Die lernmatrix, *Kybernetik* **1**(1) (1961) 36–45.
18. D. Willshaw, O. Buneman and H. Longuet-Higgins, Non-holographic associative memory, *Nature* **222** (1969) 960–962.
19. J. A. Anderson, A simple neural network generating an interactive memory. *Math. Biosci.* **14** (1972) 197–220.
20. J. J. Hopfield, Neural networks and physical systems with emergent collective computational abilities, *Proc. Natl. Acad. Sci.* **79** (1982) 2554–2558.
21. E. Acevedo, A. Martínez, A. Acevedo and F. Martínez, A novel text encryption algorithm, *Res. Comput. Sci.* **68** (2013) 91–101.