Research Article

Neural Architectures for Correlated Noise Removal in Image Processing

Cătălina Cocianu and Alexandru Stan

Computer Science Department, Bucharest University of Economics, 010552 Bucharest, Romania

Correspondence should be addressed to Cătălina Cocianu; ccocianu@ase.ro

Received 21 January 2016; Accepted 24 March 2016

Academic Editor: Marco Perez-Cisneros

Copyright © 2016 C. Cocianu and A. Stan. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The paper proposes a new method that combines the decorrelation and shrinkage techniques to neural network-based approaches for noise removal purposes. The images are represented as sequences of equal sized blocks, each block being distorted by a stationary statistical correlated noise. Some significant amount of the induced noise in the blocks is removed in a preprocessing step, using a decorrelation method combined with a standard shrinkage-based technique. The preprocessing step provides for each initial image a sequence of blocks that are further compressed at a certain rate, each component of the resulting sequence being supplied as inputs to a feed-forward neural architecture $F_X \rightarrow F_H \rightarrow F_Y$. The local memories of the neurons of the layers F_H and F_Y are generated through a supervised learning process based on the compressed versions of blocks of the same index value supplied as inputs and the compressed versions of them resulting as the mean of their preprocessed versions. Finally, using the standard decompression technique, the sequence of the decompressed blocks is the cleaned representation of the initial image. The performance of the proposed method is evaluated by a long series of tests, the results being very encouraging as compared to similar developments for noise removal purposes.

1. Introduction

There have been proposed a long series of digital image manipulation techniques, general and special tailored ones for different particular purposes. Digital image processing involves procedures including the acquisition and codification of images in digital files and the transmission of the resulting digital files of some communication channels, usually affected by noise [1, 2]. Consequently, a significant part of digital image procedures are devoted to noise removal and image reconstruction, most of them being developed in the framework represented by the assumptions that the superimposed noise is uncorrelated and normally distributed [3, 4]. Our approach is somehow different, keeping the assumption about normality but relaxing the constraint that the superimposed noise affects neighbor image pixels in a correlated way.

There are two basic mathematical characterizations of images, deterministic and statistical. In deterministic image

representation, the image pixels are defined in terms of a certain function, possibly unknown, while, in statistical image representation, the images are specified in probabilistic terms as means, covariances, and higher degree moments [5–7]. In the past years, a series of techniques have been developed in order to involve neural architectures in image compression and denoising processes [8–13].

A neural network is a massively parallel-distributed processor made up of simple processing units, which has a natural propensity for storing experiential knowledge and making it available for use [14]. The neural networks methodology is of biological inspiration, a neural network resembling the biological brain in two respects; on one hand the knowledge is acquired by the network for its environment through a learning process, and on the other hand the interneuron connection strengths are used to store the acquired knowledge.

The “shrinkage” is a method for reducing the uncorrelated Gaussian noise affecting additively a signal image by soft

thresholding applied to the sparse components [15–17]. Its use in neural network-based approach is intuitively explained by the fact that when only a few of the neurons are simultaneously active, it makes sense to assume that the activities of neurons with small absolute values correspond to noise; therefore they should be set to zero, and only the neurons whose absolute values of their activities are relatively large contain relevant information about the signal.

Recently, a series of correlated noise removal techniques have been reported. Some approaches focus on estimating spatial correlation characteristics of noise for a given image either when noise type and statistics like variance are known [18] or in case the noise variance and spatial spectrum have to be estimated [19] and then use a DCT-based method for noise removal. Wavelet-based approaches mainly include noise prewhitening technique followed by the wavelet-based thresholding [20], additive stationary correlated noise removal by modeling the noise-free coefficients using a multivariate Gaussian Scale Mixture [21], and image denoising using HMM in the wavelet domain based on the concept of signal of interest [22, 23]. Since the sparsity of signals can be exploited for noise removal purpose when different representations are used (Fourier, wavelet, principal components, independent components, etc.), a series of results concerning this property could be of interest in image denoising [24–26] and artifact (noise) removal in magnetic resonance imaging [27].

The outline of the paper is as follows. The general model of image transmission through a noisy corrupted channel is described in Section 2. Each image is transmitted several times as a sequence of equal sized blocks, each block being disturbed by a correlated Gaussian noise whose statistical properties are not known. All variants of each block are submitted to a sequence of transforms that decorrelate, shrink, and average the pixel values.

A special tailored family of feed-forward single-hidden-layer neural networks is described in Section 3, their memories being generated using a supervised learning algorithm of gradient descent type.

A suitable methodology aiming to implement a noise removal method on neural network for image processing purposes is then described in the fourth section of the paper. The proposed methodology was applied to process images from different standard databases, the conclusions experimentally derived from the tests performed on two standard databases, the former containing images of human faces and the latter containing images of landscapes being reported in the next section.

The final section of the paper contains a series of conclusive remarks.

2. Image Preprocessing Based on Decorrelation and Shrinkage Techniques

We assume that the images are transmitted through a noisy channel, each image I being transmitted as a sequence of m d -dimensional blocks, $B_1, B_2, \dots, B_m, I = (B_1, B_2, \dots, B_m)$, and we denote by $\tilde{I} = (\tilde{B}_1, \tilde{B}_2, \dots, \tilde{B}_m)$ the received image.

A working assumption of our model is that the noise modeled by the d -dimensional random vectors $\eta_i, 1 \leq i \leq m$, affects the blocks in a similar way, where $\eta_1, \eta_2, \dots, \eta_m$ are independent identically distributed; $\eta_i \sim N(0, \Sigma), 1 \leq i \leq m$.

In case N images, $I^1, I^2, \dots, I^N$, are transmitted sequentially through the channel we denote by $(\tilde{I}^i = (\tilde{B}_1^i, \tilde{B}_2^i, \dots, \tilde{B}_m^i), 1 \leq i \leq N)$ the sequence of received disturbed variants. In our model, we adopt the additional working assumption that, for each $1 \leq j \leq m$, $(\tilde{B}_j^1, \tilde{B}_j^2, \dots, \tilde{B}_j^N)$ is a realization of a d -dimensional random vector $\tilde{B}_j^0 = B_j^0 + \eta_j$, where B_j^0 is a random vector of mean μ_j^0 and covariance matrix Σ_j^0 , and that B_j^0 and η_j are independent; therefore the covariance matrix of $\tilde{B}_j^0$ is $\tilde{\Sigma}_j^0 = \Sigma_j^0 + \Sigma$. The working assumptions included in our model seem to be quite realistic according to the currently used information transmission frameworks. According to the second working assumption, for each index value j , the sequence of blocks $(B_j^1, B_j^2, \dots, B_j^N)$ could represent fragments of possibly different images taken at the counterpart positions, as, for instance, in case of face images the areas of eyes or mouths and so on. Therefore the assumption that each B_j^0 is a random vector corresponds to a model for each particular block, the parameters μ_j^0 and Σ_j^0 expressing the variability existing in the sequence of images $I^1, I^2, \dots, I^N$ at the level of j th block.

On one hand, the maximum likelihood estimates (MLE) of the parameters μ_j^0 and $\tilde{\Sigma}_j^0$ are given by

$$\hat{\mu}_j^0 = \frac{1}{N} \sum_{i=1}^N \tilde{B}_j^i, \quad (1)$$

$$\hat{\Sigma}_j^0 = \frac{1}{N-1} \sum_{i=1}^N (\tilde{B}_j^i - \hat{\mu}_j^0)(\tilde{B}_j^i - \hat{\mu}_j^0)^T, \quad (2)$$

respectively. On the other hand, the values of the parameters μ_j^0 and Σ_j^0 are also unknown and moreover it is quite inconvenient to estimate them before the transmission of the sequence of images is over.

The covariance matrix corresponding to the noise component can be estimated before the transmission is performed by different methods, as, for instance, the white wall method; therefore, without loss of generality, the matrix Σ can be assumed to be known; therefore, $\tilde{\Sigma}_j^0 = \hat{\Sigma}_j^0 - \Sigma$ can be taken as an estimate of Σ_j^0 .

Also, in case each sequence $(\tilde{B}_j^1, \tilde{B}_j^2, \dots, \tilde{B}_j^N)$ is processed separately, we can assume that the data are centered; that is, $\hat{\mu}_j^0 = 0, 1 \leq j \leq m$.

Consequently, the available information in developing a denoising procedure is represented by the sequences $(\tilde{B}_j^1, \tilde{B}_j^2, \dots, \tilde{B}_j^N)$, the estimates $\tilde{\Sigma}_j^0, 1 \leq j \leq m$, and Σ .

In our work we consider the following shrinkage type denoising method.

For each $1 \leq j \leq m$, we denote by A_j a matrix that diagonalizes simultaneously the matrices $\tilde{\Sigma}_j^0$ and Σ .

According to the celebrated W theorem [28, 29], the columns of A_j are eigenvectors of $(\hat{\Sigma}_j^0)^{-1}\Sigma$ and the following equations hold:

$$(A_j)^T \hat{\Sigma}_j^0 A_j = I_d, \quad (3)$$

$$(A_j)^T \Sigma A_j = \Lambda^j = \text{diag}(\lambda_1^j, \lambda_2^j, \dots, \lambda_d^j), \quad (4)$$

where $\lambda_1^j, \lambda_2^j, \dots, \lambda_d^j$ are the eigenvalues of the matrix $(\hat{\Sigma}_j^0)^{-1}\Sigma$. Note that although $(\hat{\Sigma}_j^0)^{-1}\Sigma$ is not a symmetric matrix, its eigenvalues are proved to be real positive numbers [29].

Let $\tilde{C}_j^i$, $1 \leq i \leq N$, be the random vectors:

$$\tilde{C}_j^i = (A_j)^T \tilde{B}_j^i = (A_j)^T B_j^i + (A_j)^T \eta_j. \quad (5)$$

Note that the linear transform of matrix $(A_j)^T$ allows obtaining the representation $\tilde{C}_j^i$, where the most amount of noise is contained in the second term. Moreover, since

$$\text{Cov}\left((A_j)^T \eta_j, ((A_j)^T \eta_j)^T\right) = (A_j)^T \Sigma A_j = \Lambda^j, \quad (6)$$

the linear transform of matrix $(A_j)^T$ decorrelates the noise components.

Let $\tilde{D}_j^i$, $1 \leq i \leq N$, be the sequence of variants of $\tilde{C}_j^i$ using the code shrinkage method [16], where each entry p , $1 \leq p \leq d$, of $\tilde{D}_j^i$ is

$$\tilde{D}_j^i(p) = \text{sgn}(\tilde{C}_j^i(p)) \max\{0, |\tilde{C}_j^i(p)| - \sqrt{2\lambda_p^j}\}. \quad (7)$$

Then $\tilde{D}_j^i$ is a variant of $\tilde{C}_j^i$ where the noise distributed $N(0, \Lambda^j)$ is partially removed. Since $\tilde{B}_j^i = ((A_j)^T)^{-1} \tilde{C}_j^i$ a variant of $\tilde{B}_j^i$ where the noise was partially removed can be taken as

$$\hat{B}_j^i = ((A_j)^T)^{-1} \tilde{D}_j^i. \quad (8)$$

Obviously, from (3) we get $((A_j)^T)^{-1} = \hat{\Sigma}_j^0 A_j$; that is,

$$\hat{B}_j^i = \hat{\Sigma}_j^0 A_j \tilde{D}_j^i. \quad (9)$$

Note that, although the eigenvalues of $(\hat{\Sigma}_j^0)^{-1}\Sigma$ are theoretically guaranteed to be positive numbers, in real world applications frequently arise situations when this matrix is ill

conditioned. In order to overpass this difficulty, in our tests we implemented the code shrinkage method using

$$\tilde{D}_j^i(p) = \begin{cases} \text{sgn}(\tilde{C}_j^i(p)) \max\{0, |\tilde{C}_j^i(p)| - \sqrt{2\lambda_p^j}\}, & \lambda_p^j > \varepsilon, \\ \tilde{C}_j^i(p), & \text{otherwise,} \end{cases} \quad (10)$$

where ε is a conventionally selected positive threshold value. Also, instead of (8) we use

$$\hat{B}_j^i = ((A_j)^T)^+ \tilde{D}_j^i, \quad (11)$$

where $((A_j)^T)^+$ is the generalized inverse (Penrose pseudoinverse) of $(A_j)^T$ [30].

In our approach we assumed the source of noise (namely, the communication channel used to transmit the image) can be observed. This hypothesis is frequently used in image restoration techniques [26]. In preprocessing and training stages, undisturbed original versions of the images transmitted are not available; instead, a series of perturbed versions are available and also through white wall technique noise component characteristics may be estimated. Working hypothesis includes the fact that images come from a common probability repartition (maybe a mixture); that is, they share the same statistical characteristics. This hypothesis is frequently used when sets of images are captured and processed [16]. The purpose of this method is, on one hand, to eliminate correlated noise, and, on the other hand, to eliminate the noise from new images transmitted through a communication channel, when they come from the same probability distribution as the images in the initially observed set.

3. Neural Networks Based Approach to Image Denoising

The aim of this section is to present an image denoising method in the framework described in the previous section implemented on a family of standard feed-forward neural architectures $\text{NN}_j : (F_X)_j \rightarrow (F_H)_j \rightarrow (F_Y)_j$, $1 \leq j \leq m$, working in parallel.

Let us assume that $\tilde{I} = (\tilde{B}_1, \tilde{B}_2, \dots, \tilde{B}_m)$ is the noisy received version of the image $I = (B_1, B_2, \dots, B_m)$ transmitted through the channel. The training process of the architectures NN_j , $1 \leq j \leq m$, is organized such that the resulting memories encode the associations of the type (*input block, sample mean*), the purpose being the noise removal according to the method presented in the previous section.

In order to reduce in some extent the computational complexity, a preprocessing step aiming dimensionality reduction is required. In our work we use L_2 -PCA method to compress the blocks. Since the particular positions of the blocks correspond to different models, their compressed versions could be of different sizes. Indeed, according to (2), the estimates of the autocorrelation matrices $\hat{\Sigma}_j^0 + \hat{\mu}_j^0(\hat{\mu}_j^0)^T$, $1 \leq j \leq m$, are different for different values of the index j ;

therefore, the numbers of the most significant directions are different for different values of index j ; that is, the sizes of the compressed variants of blocks are, in general, different. Consequently, the sizes of $(F_X)_j$ and $(F_Y)_j$ depend on j , these sizes resulting in the preprocessing step by applying the L_2 -PCA method [31, 32].

The hidden neurons influence the error on the nodes to which their output is connected. The use of too many hidden neurons could cause the so-called overfitting effect which means the overestimate of the complexity corresponding to the target problem. Maybe the most unpleasant consequence is that this way the generalization capability is decreased; therefore, the capacity of prediction is degraded too. On the other hand, at least in image processing, the use of fewer hidden neurons implies that less information extracted from the inputs is processed and consequently less accuracy should be expected. Consequently, the determining of the right size of the hidden layer results as a trade-off between accuracy and generalization capacity.

There have been proposed several expressions to compute the number of neurons in the hidden layers [33, 34]. Denoting by $|\cdot|$ the number of elements of the argument, the sizes of the hidden layers $(F_H)_j$ can be computed many ways, some of the most frequent expressions being [34]

$$|(F_H)_j| = 2 \left\lceil \sqrt{(|(F_Y)_j| + 2) |(F_X)_j|} \right\rceil, \quad (12a)$$

$$|(F_H)_j| = \left\lceil \sqrt{(|(F_Y)_j| + 2) |(F_X)_j|} + 2 \sqrt{\frac{|(F_X)_j|}{(|(F_Y)_j| + 2)}} \right\rceil. \quad (12b)$$

The aim of the training is that, for each value of the index j to obtain on the output on the layer $(F_Y)_j$, a compressed cleaned version of the input applied to the layer $(F_X)_j$, the output being computed according to the method presented in the previous section.

According to the approach described in the previous section, all blocks of the same index say j are processed by the same compression method yielding to compressed variants, the size of compressed variants being the same for all these blocks. The compressed variants corresponding to the blocks of index j are next fed as inputs to j th neural architecture. Consequently, the denoising process of an image consisting of m blocks is implemented on a family of m neural architectures operating in parallel $(NN_j, 1 \leq j \leq m)$, where $NN_j : (F_X)_j \rightarrow (F_H)_j \rightarrow (F_Y)_j$; the sequence of denoised variants resulted as outputs of the layers $(F_Y)_j$ being next decompressed. The cleaned variant of each input image is taken as the sequence of the decompressed cleaned variants of its blocks.

The preprocessing step producing the compressed variants fed as input blocks is described as follows. For each index value j , the sequence of compressed versions of the blocks $(\tilde{B}_j^1, \tilde{B}_j^2, \dots, \tilde{B}_j^N)$ denoted by $(\widehat{CB}_j^1, \widehat{CB}_j^2, \dots, \widehat{CB}_j^N)$ is

$$\widehat{CB}_j^i = W_j^T \tilde{B}_j^i, \quad i = 1, \dots, N, \quad (13)$$

where the columns of the matrix W_j are the most significant unit eigenvectors of $\tilde{S}_j^0 + \tilde{\mu}_j^0(\tilde{\mu}_j^0)^T$. The most significant unit eigenvectors of $\tilde{S}_j^0 + \tilde{\mu}_j^0(\tilde{\mu}_j^0)^T$ are computed as follows. Let $\theta_1^{(j)} \geq \theta_2^{(j)} \geq \dots \geq \theta_d^{(j)}$ be the eigenvalues of $\tilde{S}_j^0 + \tilde{\mu}_j^0(\tilde{\mu}_j^0)^T$ and $\varepsilon_1 \in (0, 1)$ a conventionally selected threshold value. If t is the smallest value such that (14) holds, then the columns of W_j are unit eigenvectors of $\tilde{S}_j^0 + \tilde{\mu}_j^0(\tilde{\mu}_j^0)^T$ corresponding to the largest t eigenvalues:

$$\frac{1}{\sum_{k=1}^d \theta_k^{(j)}} \sum_{k=t+1}^d \theta_k^{(j)} < \varepsilon_1; \quad (14)$$

therefore, $|(F_X)_j| = t$.

Assuming that the sequence of blocks $(\tilde{B}_j^1, \dots, \tilde{B}_j^N)$ are cleaned versions of $(\tilde{B}_j^1, \tilde{B}_j^2, \dots, \tilde{B}_j^N)$ computed according to (11), we denote by $(\widehat{CB}_j^1, \dots, \widehat{CB}_j^N)$ their compressed variants:

$$\widehat{CB}_j^i = V_j^T \tilde{B}_j^i, \quad i = 1, \dots, N, \quad (15)$$

where the columns of the matrix V_j are the most significant unit eigenvectors of the autocorrelation matrix $(1/N) \sum_{i=1}^N \tilde{B}_j^i (\tilde{B}_j^i)^T$. The most significant eigenvectors of $(1/N) \sum_{i=1}^N \tilde{B}_j^i (\tilde{B}_j^i)^T$ are computed in a similar way as in the compression step applied to input blocks using possibly a different threshold value $\varepsilon_2 \in (0, 1)$. Note that, in tests, the threshold values $\varepsilon_1, \varepsilon_2$ are experimentally tuned to the particular sequence of images.

To summarize, the preprocessing scheme consists of applying L_2 -PCA method to both noisy sequence of blocks $(\tilde{B}_j^1, \tilde{B}_j^2, \dots, \tilde{B}_j^N)$ and their cleaned versions $(\tilde{B}_j^1, \dots, \tilde{B}_j^N)$ causing the sequence of inputs to be applied to the input layer $(F_X)_j$ and to their compressed cleaned versions $(\widehat{CB}_j^1, \dots, \widehat{CB}_j^N)$:

$$\begin{aligned} (\tilde{B}_j^1, \tilde{B}_j^2, \dots, \tilde{B}_j^N) &\xrightarrow{W_j^T} (\widehat{CB}_j^1, \widehat{CB}_j^2, \dots, \widehat{CB}_j^N) \\ &\rightarrow (F_X)_j. \end{aligned} \quad (16)$$

The aim of the training is to produce on each output layer the sequence $(\widehat{CB}_j^1, \dots, \widehat{CB}_j^N)$, the decompressed versions of its blocks being $(V_j \widehat{CB}_j^1, \dots, V_j \widehat{CB}_j^N)$:

$$\begin{aligned} (F_Y)_j &\rightarrow (\widehat{CB}_j^1, \dots, \widehat{CB}_j^N) \\ &\xrightarrow{V_j} (V_j \widehat{CB}_j^1, \dots, V_j \widehat{CB}_j^N); \end{aligned} \quad (17)$$

therefore, the blocks of $(V_j \widehat{CB}_j^1, \dots, V_j \widehat{CB}_j^N)$ are denoised versions of $(\tilde{B}_j^1, \tilde{B}_j^2, \dots, \tilde{B}_j^N)$, respectively.

The training of each neural architecture NN_j is of supervised type using a gradient descent approach, the local memories of $(F_H)_j$ and $(F_Y)_j$ being determined using the Levenberg-Marquardt variant of the backpropagation learning algorithm (LM-BP algorithm) [35].

We organized the training process for the m neural networks by transmitting through the channel each available image several times, say p times; the reason of doing that is that this way better estimates of the covariance matrices Σ_j^0 , $1 \leq j \leq m$, of the proposed stochastic models are expected to be obtained.

Consequently, the whole available data is the collection $(\tilde{I}^{i,l} = (\tilde{B}_1^{i,l}, \tilde{B}_2^{i,l}, \dots, \tilde{B}_m^{i,l}), 1 \leq i \leq N, 1 \leq l \leq p)$; therefore, for each index value j , the inputs applied to the j th neural network are the sequence $(\widetilde{CB}_j^{1,1}, \widetilde{CB}_j^{1,2}, \dots, \widetilde{CB}_j^{1,p}, \dots, \widetilde{CB}_j^{N,1}, \widetilde{CB}_j^{N,2}, \dots, \widetilde{CB}_j^{N,p})$ of compressed versions of the blocks $(\tilde{B}_j^{1,1}, \tilde{B}_j^{1,2}, \dots, \tilde{B}_j^{1,p}, \dots, \tilde{B}_j^{N,1}, \dots, \tilde{B}_j^{N,p})$:

$$\widetilde{CB}_j^{i,l} = W_j^T \tilde{B}_j^{i,l}, \quad i = 1, \dots, N, \quad l = 1, \dots, p. \quad (18)$$

The linear compression filter W_j is a matrix whose columns are the most significant unit eigenvectors of the matrix $(1/Np) \sum_{i=1}^N \sum_{l=1}^p \tilde{B}_j^{i,l} (\tilde{B}_j^{i,l})^T$.

Let $(\hat{B}_j^{1,1}, \dots, \hat{B}_j^{1,p}, \dots, \hat{B}_j^{N,1}, \dots, \hat{B}_j^{N,p})$ be the sequence of the cleaned variants of $(\tilde{B}_j^{1,1}, \tilde{B}_j^{1,2}, \dots, \tilde{B}_j^{1,p}, \dots, \tilde{B}_j^{N,1}, \dots, \tilde{B}_j^{N,p})$ computed using (11) and, for each $1 \leq i \leq N$, let M_j^i be the sample mean of cleaned blocks $(\hat{B}_j^{i,1}, \dots, \hat{B}_j^{i,p})$:

$$M_j^i = \frac{1}{p} \sum_{l=1}^p \hat{B}_j^{i,l}. \quad (19)$$

We denote by V_j a linear compression filter whose columns are the most significant unit eigenvectors of the matrix $(1/N) \sum_{i=1}^N M_j^i (M_j^i)^T$ computed in a similar way as (15) using a threshold value $\varepsilon_2 \in (0, 1)$ and let $CM_j^i = V_j^T M_j^i$.

The learning process for each neural architecture NN_j , $1 \leq j \leq m$, is developed to encode the associations $(\widetilde{CB}_j^{k,1}, \dots, \widetilde{CB}_j^{k,p}) \rightarrow CM_j^k$, $1 \leq k \leq N$. The reason of using the means M_j^k , $1 \leq k \leq N$, and their corresponding compressed versions instead of the associations $(\widetilde{CB}_j^{k,1}, \dots, \widetilde{CB}_j^{k,p}) \rightarrow (\widetilde{CB}_j^{k,1}, \dots, \widetilde{CB}_j^{k,p})$, $1 \leq k \leq N$, resides in the fact that taking the means and their compressed versions some amount of noise is expected to be removed, for each value of the index j ; that is, the compressed versions of the means are expected to be better cleaned variants of the compressed blocks.

Summarizing, the memory of each neural architecture NN_j is computed by the Levenberg-Marquardt algorithm applied to the input/output sequence $\{(\widetilde{CB}_j^{1,1}, CM_j^1), \dots, (\widetilde{CB}_j^{1,p}, CM_j^1), \dots, (\widetilde{CB}_j^{N,1}, CM_j^N), \dots, (\widetilde{CB}_j^{N,p}, CM_j^N)\}$, $1 \leq j \leq m$.

Once the training phase is over, the family of NN_j 's is used to remove the noise from a noisy version of an image $\tilde{I} = (\tilde{B}_1, \tilde{B}_2, \dots, \tilde{B}_m)$ received through the channel according to the following scheme. Let $I = (B_1, B_2, \dots, B_m)$ be the initial image transmitted through the channel and $\tilde{I} = (\tilde{B}_1, \tilde{B}_2, \dots, \tilde{B}_m)$ the received noisy version.

Step 1. Compress each block $\tilde{B}_j$ of $\tilde{I}$ using the filter W_j and get its compressed version $\widetilde{CB}_j$; that is, $(\widetilde{CB}_1, \widetilde{CB}_2, \dots, \widetilde{CB}_m)$ is a dynamically block-compressed version of $\tilde{I}$.

Step 2. Apply $(\widetilde{CB}_1, \widetilde{CB}_2, \dots, \widetilde{CB}_m)$ as inputs to the architectures NN_j 's, $\widetilde{CB}_j$ applied as input to the layer $(F_X)_j$, $1 \leq j \leq m$, and get the outputs RB_j 's.

Step 3. Decompress each block RB_j using the decompression filter V_j , $1 \leq j \leq m$.

Step 4. Get $\hat{I} = (V_1 \cdot RB_1, \dots, V_m \cdot RB_m)$ the cleaned version of $\tilde{I}$.

4. Description of the Methodology Applied in the Implementations of the Proposed Method on Neural Architectures

The aim of this section is to describe the methodology followed in implementing the neural network-based noise removal method for image processing purposes. The proposed methodology was applied to process images from different standard databases, the conclusions experimentally derived from the tests performed on two standard databases, the former containing images of human faces and the latter containing images of landscapes being reported in the next section.

We performed the experiments according to the following methodology.

(1) The quality of the a certain test image $T = (t(x, y))$ versus a reference image $R = (r(x, y))$ of the same size (n_x, n_y) is evaluated in terms of the Signal-to-Noise Ratio (SNR), Peak Signal-to-Noise Ratio (PSNR), Root Mean Squared Signal-to-Noise Ratio (SNR_RMS) indicators [36], and the Structural Similarity Metric (SSIM) [37], where

$$\begin{aligned} \text{SNR}(R, T) &= 10 * \log_{10} \left[\frac{\sum_{x=1}^{n_x} \sum_{y=1}^{n_y} (r(x, y))^2}{\sum_{x=1}^{n_x} \sum_{y=1}^{n_y} (r(x, y) - t(x, y))^2} \right], \\ \text{PSNR}(R, T) &= 10 \\ &* \log_{10} \left[\frac{\max(r(x, y))^2}{(1/(n_x * n_y)) \sum_{x=1}^{n_x} \sum_{y=1}^{n_y} (r(x, y) - t(x, y))^2} \right], \quad (20) \\ \text{SNR_RMS}(R, T) &= \sqrt{\frac{\sum_{x=1}^{n_x} \sum_{y=1}^{n_y} (r(x, y))^2}{\sum_{x=1}^{n_x} \sum_{y=1}^{n_y} (r(x, y) - t(x, y))^2}}. \end{aligned}$$

Let x and y be spatial patches extracted from the images R and T , respectively. The two patches correspond to the same

spatial window of the images R and T . The original standard SSIM value computed for the patches x and y is defined by

$$\text{SSIM}(x, y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1} \cdot \frac{2\sigma_{xy} + C_2}{\sigma_x^2 + \sigma_y^2 + C_2}, \quad (21)$$

where μ_x denotes the mean value of x , σ_x is the standard deviation of x , and σ_{xy} represents the cross-correlation of the mean shifted patches $x - \mu_x$ and $y - \mu_y$. The constants C_1 and C_2 are small positive numbers included to avoid instability when either $\mu_x^2 + \mu_y^2$ or $\sigma_x^2 + \sigma_y^2$ is very close to zero, respectively. The overall SSIM index for the images R and T is computed as the mean value of the SSIM measures computed for all pairs of patches x and y of R and T , respectively.

(2) The size of the blocks and the model of noise in transmitting data through the channel are selected for each database. The size of the blocks is established by taking into account the size of the available images in order to assure reasonable complexity to the noise removal process. In our tests the size of input blocks is about 150 and the sizes of images are 135×100 in case of the database containing images of human faces and 154×154 in case of the database containing images of landscapes. We assumed that the components of the noise η induced by the channel are possibly correlated; in our tests, the noise model is of Gaussian type, $\eta \sim N(0, \Sigma)$, where Σ is a symmetric positive defined matrix.

(3) The compression thresholds $\varepsilon_1, \varepsilon_2$ in (14) and (15) are established in order to assure some desired accuracy. In our tests we used $\varepsilon_1 = c_1 * 10^{-4}$, $\varepsilon_2 = c_2 * 10^{-4}$, where c_1, c_2 are positive constants. The reason for selecting different magnitude orders of these thresholds stems from the fact that ε_1 is used in compressing noise affected images, while ε_2 is used for compressing noise cleaned images [32]. The sizes of the input and output layers $(F_X)_j, (F_Y)_j$ of the neural network NN_j result in terms of the established values of ε_1 and ε_2 accordingly.

(4) The quality evaluation of the preprocessing step consisting in noise cleaning data is performed in terms of the indicators (20) and (21), by comparing the initial data $I = (B_1, B_2, \dots, B_m)$ against the noisy transmitted images $\tilde{I} = (\tilde{B}_1, \tilde{B}_2, \dots, \tilde{B}_m)$ through the channel and $I = (B_1, B_2, \dots, B_m)$ against their corresponding cleaned versions $\hat{I} = (V_1 \cdot RB_1, \dots, V_m \cdot RB_m)$, respectively.

(5) In order to implement the noise removal method on a family of neural networks $\text{NN}_j : (F_X)_j \rightarrow (F_H)_j \rightarrow (F_Y)_j$, $1 \leq j \leq m$, the sizes of the input $(F_X)_j$ and the output layers $(F_Y)_j$ are determined by L_2 -PCA compression/decompression method and the established values of $\varepsilon_1, \varepsilon_2$. The sizes of the layers $(F_H)_j$ are determined as approximations of the recommended values cited in the published literature (12a) and (12b). In order to assure a reasonable tractability of the data, in our tests we were forced to use a less number of neurons than it is recommended, on the hidden layers $(F_H)_j$.

For fixed values of $\varepsilon_1, \varepsilon_2$, the use of the recommended number of neurons as in (12a) and (12b) usually yields to

either the impossibility of implementing the learning process or to too lengthy training processes. Therefore, in such case we are forced to reconsider the values of $\varepsilon_1, \varepsilon_2$ by increasing them, therefore decreasing the numbers of neurons on the input and the output layers and consequently the number of neurons on the hidden layers too. Obviously, by reconsidering this way the values of $\varepsilon_1, \varepsilon_2$, inherently imply that some larger amount of information about data is lost. The effects of losing information are manifold, one of them being that the cleaned versions resulted from decompressing the outputs of NN_j 's yield to poorer approximation $\hat{I}$ of the initial image I .

This way we arrive at the conclusion that, in practice, we have to solve a trade-off problem between the magnitude of the compression rates and the number of neurons on the hidden layers $(F_H)_j$'s. In order to solve this trade-off, in our tests we used smaller numbers of neurons than recommended on the hidden layers and developed a comparative analysis on the quality of the resulting cleaned images.

(6) The activation functions of the neurons belonging to the hidden and output layers can be selected from very large family. In our tests, we considered the logistic type to model the activation functions of the neurons belonging to the hidden layers and the unit functions to model the outputs of the neurons belonging to the output layers. Also, the learning process involved the task of splitting the available data into training, validation, and test data. In our tests the sizes of the subcollections were 80%, 10%, and 10%, respectively.

(7) The evaluation of the overall quality of the noise removal process implemented on the set of neural networks, as previously described, is performed in terms of the indicators (20) and (21), on one hand by comparing the initial data $I = (B_1, B_2, \dots, B_m)$ to the noisy transmitted images $\tilde{I} = (\tilde{B}_1, \tilde{B}_2, \dots, \tilde{B}_m)$ through the channel and on the other hand by comparing $I = (B_1, B_2, \dots, B_m)$ to their cleaned versions $\hat{I} = (V_1 \cdot RB_1, \dots, V_m \cdot RB_m)$.

(8) The comparative analysis between the performances corresponding to the decorrelation and shrinkage method and its implementation on neural networks is developed in terms of the indicators (20) and (21).

5. Experimentally Derived Conclusions on the Performance of the Proposed Method

In this section we present the results in evaluating both the quality of the proposed decorrelation and shrinkage method and the power of the neural network-based approach in simulating it for noise removal purposes. The tests were performed in a similar way on two standard databases, the former, referred to as Senthil, containing images of 5 human faces and 16 images for each person [38] and the latter containing 42 images of landscapes [39]. In case of the Senthil database, the preprocessing step used 75 images; for each human face 15 of its available versions are being used. The tests performed in order to evaluate the quality of the trained family of neural networks used the rest of 5 images, one for each person. In case of the database containing images of landscapes, we identified three types of quite similar images,

TABLE 1

The maximum number of epochs	The minimum value of the performance (Jacobian computation)	The maximum validation failures	The minimum performance gradient	The initial/maximum μ factor (in the LM adaptive learning rule)
1000	0	5	10^{-5}	$10^{-3}/10^{10}$

and we used 13 images of each type in the training process, the tests being performed on the rest of three ones.

The sizes of hidden layers were set to smaller values than recommended by (12a) and (12b). For instance, when $\varepsilon_1 \approx 10^{-4}$ and $\varepsilon_2 \approx 10^{-4}$ the resulting sizes of the layers $|(F_X)_j|$ and $|(F_Y)_j|$ are about 115 and 30, respectively, the recommended sizes of the layers $|(F_H)_j|$ being about 65.

The results of a long series of tests pointed out that one can use hidden layers of smaller sizes than recommended without decreasing dramatically the accuracy. For instance, in this work, we used only half of recommended sizes; that is,

$$|(F_H)_j| = \left\lfloor \frac{\sqrt{(|(F_Y)_j| + 2)|(|F_X)_j|} + 2\sqrt{|(F_X)_j| / (|(F_Y)_j| + 2)}}{2} \right\rfloor. \quad (22)$$

In our test, the memory of each neural architecture is computed by the LM-BP algorithm, often the fastest variant of the backpropagation algorithm and one of the most commonly used in supervised learning. The available data was split into training set, validation set, and test set, the sizes of the subcollections being 80%, 10%, and 10%, respectively. The main parameters of the LM-BP training process are specified in Table 1.

In order to experimentally establish the quality of the proposed method, a comparative analysis against three of the most used and suitable algorithms for correlated noise removal, namely, BM3D (block-matching and 3D filtering [25]), NLMF (Nonlocal Means Noise Filtering [40]), and ProbShrink (correlated noise removal algorithm using nondecimated wavelet transform and generalized Laplacian [22]), was conducted. The reported results include both quantitative and qualitative comparisons.

In the following, we summarize some of our results.

(a) The quality evaluation of the preprocessing step in terms of the indicators (20) and (21) is as follows:

- (1) In Figure 1(a), a sample of five face images belonging to the Senthil database is presented, their cleaned versions resulted from applying the decorrelation and shrinkage method being shown in Figure 1(e), where each image was transmitted 30 times through the channel. In Figures 1(b), 1(c), and 1(d) the restored versions resulting from applying the NLMF algorithm, ProbShrink algorithm, and BM3D method, respectively, are depicted. Table 2 contains the

values of the indicators (20) and (21) corresponding to these five pairs of noisy-cleaned versions of these images.

Note that, on average, the best results were obtained when our method was used.

- (2) A sample of three images of landscapes, one from each class, is presented in Figure 2(a) together with their cleaned versions resulting from applying the decorrelation and shrinkage method shown in Figure 2(e), where each image was transmitted 30 times through the channel. In Figure 2(b) the restored variants using NLMF algorithm are exhibited, while in Figure 2(c) the restored variants using ProbShrink method are shown. The cleaned version using the BM3D algorithm is presented in Figure 2(d). The values of the indicators (20) and (21) are given in Table 3.

Note that, on average, the best results were obtained when our method was used.

- (b) As it was previously described, the images resulting from the preprocessing step are used in the supervised training of the family of neural networks. Once the training process is over, the family of neural networks are used to remove noise from new unseen-yet images. Obviously, it is impossible to guarantee that the new test images share the same statistical properties with the images used during the training process, the unique criterion being that they are visually enough similar. In order to take into account this constraint, we split each of these two databases containing similar images into training and testing subsets, the sizes being 75/5 for Senthil dataset and 39/3 for the second database.

- (1) The test images from the Senthil database and their versions resulting from applying the preprocessing step are shown in Figures 3(a) and 3(b), respectively. Their cleaned versions computed by the resulting family of trained neural networks are shown in Figure 3(f), while their restored versions when NLMF algorithm, ProbShrink method, and BM3D method are used are presented in Figures 3(c), 3(d), and 3(e), respectively. In terms of the indicators (20) and (21), the results are summarized in Table 4. Note that in this case our method, ProbShrink algorithm, and BM3D method produce similar results, according to both SNR measure and SSIM metric.

FIGURE 1

(2) Similar tests were performed on the database containing images of landscapes. The tests were performed on three new images shown in Figure 4(a), the results of the preprocessing step being given in Figure 4(b). The cleaned versions computed by the resulting family of trained neural networks are shown in Figure 4(f) and the clean versions given by the BM3D algorithm are presented in Figure 4(e). The results obtained when the NLMF algorithm and ProbShrink method are used are displayed in Figures 4(c)

and 4(d), respectively. The numerical evaluation in terms of the indicators (20) and (21) is summarized in Table 5.

Note that, in this case, the BM3D algorithm proved to smooth the results too much. Also, the images obtained when NLMF algorithm was used are of poor visual quality. The ProbShrink algorithm performed better than BM3D and NLMF, but, on average, the best results were obtained when our method was used.

TABLE 2

	SNR-RMS (the mean value)	SNR (the mean value)	Mean Peak SNR (the mean value)	SSIM (the mean value)
Noisy images versus original images	7.3128	17.2193	26.3217	0.5838
Cleaned images versus original images (NLMF)	12.1367	21.6369	30.8944	0.8377
Cleaned images versus original images (ProbShrink)	15.3787	23.7071	33.1043	0.8782
Cleaned images versus original images (BM3D)	15.0573	23.5242	33.3160	0.8867
Cleaned images versus original images (the proposed method)	19.5967	25.8067	35.0112	0.9163

TABLE 3

	SNR-RMS (the mean value)	SNR (the mean value)	Mean Peak SNR (the mean value)	SSIM (the mean value)
Noisy images versus original images	7.1013	17.0083	24.6427	0.6868
Cleaned images versus original images (NLMF)	7.5050	17.4992	25.8548	0.6503
Cleaned images versus original images (ProbShrink)	9.6534	19.7858	27.7324	0.8002
Cleaned images versus original images (BM3D)	8.7304	18.8084	27.6967	0.7389
Cleaned images versus original images (the proposed method)	20.8267	26.2779	34.0215	0.9341

FIGURE 2

FIGURE 3

TABLE 4

	SNR-RMS/new image	SNR/new image	Mean Peak SNR/new image	SSIM/new image
Noisy images versus original images	7.1352	17.0681	26.3356	0.5700
	7.7979	17.8395	26.4804	0.5693
	6.7452	16.5800	26.2165	0.5733
	9.0387	19.1221	26.4122	0.6410
	6.1524	15.7809	26.5486	0.5513
Cleaned images (using the preprocessing step) versus original images	8.4056	18.4914	27.8057	0.6328
	9.6450	19.6860	28.3873	0.6411
	8.3065	18.3883	28.0880	0.6392
	9.5696	19.6179	26.9517	0.6643
	7.4780	17.4758	28.2541	0.6308
Cleaned images versus original images (NLMF)	12.1067	21.6605	31.0955	0.8474
	13.6918	22.7292	31.5051	0.8448
	11.9228	21.5275	31.3547	0.8424
	14.5670	23.2674	30.6666	0.8588
	10.8248	20.6884	31.6597	0.8619
Cleaned images versus original images (ProbShrink)	15.6521	23.8914	33.2490	0.8834
	15.5453	23.8320	32.5356	0.8648
	14.6064	23.2909	33.0326	0.8705
	17.5242	24.8728	32.2178	0.8836
	13.2653	22.1486	33.5365	0.8812
Cleaned images versus original images (BM3D)	14.8333	23.4248	33.34	0.8824
	16.3388	24.2644	33.59	0.8854
	14.7108	23.3527	33.68	0.8991
	17.7965	25.0067	33.00	0.8525
	13.4046	22.4632	33.63	0.9074
Cleaned images (using NN's) versus original images (the proposed method)	13.7982	22.7964	32.2178	0.8850
	16.9576	24.5873	33.0786	0.8781
	16.5712	24.3871	33.8409	0.8968
	12.6325	22.0298	29.6259	0.8454
	13.4488	22.5737	33.6427	0.8939

TABLE 5

	SNR-RMS/new image	SNR/new image	Mean Peak SNR/new image	SSIM/new image
Noisy images versus original images	6.7603	16.5994	24.6606	0.6564
	6.5259	16.2928	24.6103	0.6796
	7.7468	17.7824	24.7497	0.6157
Cleaned images versus original images	6.8194	16.6749	24.7947	0.6590
	7.8383	17.8845	26.2810	0.7203
	7.3444	17.3191	24.3228	0.6990
Cleaned images versus original images (NLMF)	7.7084	17.7393	26.0425	0.6773
	6.9041	16.7822	25.3903	0.6025
	8.2267	18.3045	25.4729	0.6830
Cleaned images versus original images (ProbShrink)	9.5398	19.6813	27.7907	0.8060
	9.0598	19.1423	27.6171	0.7732
	10.0007	20.0039	27.5228	0.8218
Cleaned images versus original images (BM3D)	9.0589	19.1415	27.95	0.7265
	7.8517	17.8993	26.99	0.6969
	9.7064	19.7412	27.89	0.7688
Cleaned images (using NN's) versus original images (the proposed method)	9.4599	19.5177	27.6939	0.8090
	13.1507	22.3790	30.8164	0.8770
	10.0606	20.0525	27.1269	0.8412

FIGURE 4

6. Conclusive Remarks and Suggestions for Further Work

The proposed method combines the decorrelation and shrinkage techniques to neural network-based approaches for noise removal purposes. The images are assumed to be transmitted as sequences of blocks of equal sizes, each block being distorted by a stationary statistical correlated noise and some amount of the noise being partially removed using the method that combines noise decorrelation and standard shrinkage technique. The preprocessing step provides, for each initial image, a sequence of blocks that are further PCA-compressed at a certain rate, each component of the resulting sequence being supplied as inputs to a feed-forward neural architecture $F_X \rightarrow F_H \rightarrow F_Y$. Therefore, each indexed block is processed by a neural network corresponding to that index value. The local memories of the neurons of the layers F_H and F_Y are generated through a supervised learning process based on the compressed versions of blocks of the same index value supplied as inputs and the compressed versions of them resulting as the mean of their preprocessed versions. Finally, using the standard PCA-decompression technique, the sequence of the decompressed blocks is the cleaned representation of the initial image. The performance of the proposed method is evaluated by a long series of tests, the results being very encouraging as compared to similar developments for noise removal purposes. The evaluation of the amount of the noise removed is done in terms of some of the most frequently used similarity indicators, SNR, SNR-RMS, Peak SNR, and SSIM.

The results produced by applying the proposed method were compared to those produced by applying three of the most widely used algorithms for eliminating correlated noise. NLMF algorithm consistently produces weaker results than the proposed method. Using ProbShrink or BM3D, the results are similar to or weaker than those yielded by the proposed method, in both quality and quantity.

The long series of tests proved good results of the above-described methodology entailing the hope that further and possibly more sophisticated extensions can be expected to improve it. Among several possible extensions, some work is still in progress concerning the use of different output functions for the hidden and output neurons and the use of more hidden layers in the neural architectures. Also, some other compression techniques combined with new techniques for feature extraction as well as the use of other learning schemes to generate the local memories of the neurons are expected to allow the removal of a larger amount of noise.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

A major contribution to the research work reported in this paper belongs to Mrs. Luminita State, a Professor and a Ph.D. Distinguished Member of the Romanian academic

community; Professor Luminita State passed away in January 2016. The authors will always remember her amazing spirit, as well as her brilliant mind. May God rest her soul in peace!

References

- [1] W. Pratt, *Digital Image Processing*, Wiley-Interscience, Hoboken, NJ, USA, 4th edition, 2007.
- [2] E. López-Rubio, "Restoration of images corrupted by Gaussian and uniform impulsive noise," *Pattern Recognition*, vol. 43, no. 5, pp. 1835–1846, 2010.
- [3] L. State, C. Cocianu, C. Săraru, and P. Vlamos, "New approaches in image compression and noise removal," in *Proceedings of the 1st International Conference on Advances in Satellite and Space Communications (SPACOMM '09)*, pp. 96–101, IEEE, Colmar, France, July 2009.
- [4] Z. H. Shamsi and D.-G. Kim, "Multiscale hybrid nonlocal means filtering using modified similarity measure," *Mathematical Problems in Engineering*, vol. 2015, Article ID 318341, 17 pages, 2015.
- [5] P. Fieguth, *Statistical Image Processing and Multidimensional Modeling*, Springer, New York, NY, USA, 2011.
- [6] L. Tan and J. Jiang, *Digital Signal Processing. Fundamentals and Applications*, Academic Press, Elsevier, 2nd edition, 2013.
- [7] S. Kay, *Fundamentals of Statistical Signal Processing, Volume III: Practical Algorithm Development*, Prentice Hall, New York, NY, USA, 2013.
- [8] M. Egmont-Petersen, D. de Ridder, and H. Handels, "Image processing with neural networks—a review," *Pattern Recognition*, vol. 35, no. 10, pp. 2279–2301, 2002.
- [9] F. Hussain and J. Jeong, "Efficient deep neural network for digital image compression employing rectified linear neurons," *Journal of Sensors*, vol. 2016, Article ID 3184840, 7 pages, 2016.
- [10] A. J. Hussain, D. Al-Jumeily, N. Radi, and P. Lisboa, "Hybrid neural network predictive-wavelet image compression system," *Neurocomputing*, vol. 151, no. 3, pp. 975–984, 2015.
- [11] S. Bhattacharyya, P. Pal, and S. Bhowmick, "Binary image denoising using a quantum multilayer self organizing neural network," *Applied Soft Computing Journal*, vol. 24, pp. 717–729, 2014.
- [12] Y. Li, J. Lu, L. Wang, and Y. Takashi, "Denoising by using multineural networks for medical X-ray imaging applications," *Neurocomputing*, vol. 72, no. 13–15, pp. 2884–2891, 2009.
- [13] I. Turkmen, "The ANN based detector to remove random-valued impulse noise in images," *Journal of Visual Communication and Image Representation*, vol. 34, pp. 28–36, 2016.
- [14] S. Haykin, *Neural Networks A Comprehensive Foundation*, Prentice Hall, 1999.
- [15] Y. Wu, B. H. Tracey, P. Natarajan, and J. P. Noonan, "Fast block-wise SURE shrinkage for image denoising," *Signal Processing*, vol. 103, pp. 45–59, 2014.
- [16] A. Hyvarinen, J. Karhunen, and E. Oja, *Independent Component Analysis*, John Wiley & Sons, New York, NY, USA, 2001.
- [17] L. Shang, D.-S. Huang, C.-H. Zheng, and Z.-L. Sun, "Noise removal using a novel non-negative sparse coding shrinkage technique," *Neurocomputing*, vol. 69, no. 7–9, pp. 874–877, 2006.
- [18] N. N. Popomarenko, V. V. Lukin, A. A. Zelensky, J. T. Astola, and J. T. Astola, "Adaptive DCT-based filtering of images corrupted by spatially correlated noise," in *Image Processing: Algorithms and Systems VI*, vol. 6812 of *Proceedings of SPIE*, San Jose, Calif, USA, January 2008.
- [19] N. N. Popomarenko, V. V. Lukin, K. O. Egiazarian, and J. T. Astola, "A method for blind estimation of spatially correlated noise characteristics," in *Image Processing: Algorithms and Systems VIII*, vol. 7532 of *Proceedings of SPIE*, February 2010.
- [20] I. M. Johnstone and B. W. Silverman, "Wavelet threshold estimators for data with correlated noise," *Journal of the Royal Statistical Society. Series B. Methodological*, vol. 59, no. 2, pp. 319–351, 1997.
- [21] J. Portilla, V. Strela, M. J. Wainwright, and E. P. Simoncelli, "Image denoising using scale mixtures of Gaussians in the wavelet domain," *IEEE Transactions on Image Processing*, vol. 12, no. 11, pp. 1338–1351, 2003.
- [22] A. Pižurica and W. Philips, "Estimating the probability of the presence of a signal of interest in multiresolution single- and multiband image denoising," *IEEE Transactions on Image Processing*, vol. 15, no. 3, pp. 654–665, 2006.
- [23] B. Goossens, Q. Luong, A. Pizurica, and W. Philips, "An improved non-local denoising algorithm," in *Proceedings of the International Workshop on Local and Non-Local Approximation in Image Processing (LNLA '08)*, pp. 143–156, Lausanne, Switzerland, August 2008.
- [24] M. Jansen, *Noise Reduction by Wavelet Thresholding*, Springer, Berlin, Germany, 2001.
- [25] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-D transform-domain collaborative filtering," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2080–2095, 2007.
- [26] M. P. S. Chawla, "PCA and ICA processing methods for removal of artifacts and noise in electrocardiograms: a survey and comparison," *Applied Soft Computing Journal*, vol. 11, no. 2, pp. 2216–2226, 2011.
- [27] L. Griffanti, G. Salimi-Khorshidi, C. F. Beckmann et al., "ICA-based artefact removal and accelerated fMRI acquisition for improved resting state network imaging," *NeuroImage*, vol. 95, pp. 232–247, 2014.
- [28] P. Common and C. Jutten, *Handbook of Blind Source Separation: Independent Component Analysis and Applications*, Academic Press, Elsevier, 2010.
- [29] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Academic Press, 2nd edition, 1990.
- [30] J. E. Gentle, *Matrix Algebra. Theory, Computations, and Applications in Statistics*, Springer Texts in Statistics, Springer, New York, NY, USA, 2007.
- [31] I. T. Jolliffe, *Principal Component Analysis*, Springer Series in Statistics, Springer, Berlin, Germany, 2nd edition, 2002.
- [32] C. Cocianu, L. State, and P. Vlamos, "Neural implementation of a class of PCA learning algorithms," *Economic Computation and Economic Cybernetics Studies and Research* no. 3/2009, 2009.
- [33] K. Gnana Sheela and S. N. Deepa, "Review on methods to fix number of hidden neurons in neural networks," *Mathematical Problems in Engineering*, vol. 2013, Article ID 425740, 11 pages, 2013.
- [34] G.-B. Huang, "Learning capability and storage capacity of two-hidden-layer feedforward networks," *IEEE Transactions on Neural Networks*, vol. 14, no. 2, pp. 274–281, 2003.
- [35] G. A. F. Seber and C. J. Wild, *Nonlinear Regression*, John Wiley & Sons, New York, NY, USA, 2003.
- [36] R. Gonzales and R. Woods, *Digital Image Processing*, Prentice Hall, 5th edition, 2008.

- [37] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.
- [38] <http://www.geocities.ws/senthilirtt/Senthil%20Face%20Database%20Version1>.
- [39] <http://sipi.usc.edu/database/database.php?volume=sequences>.
- [40] A. Buades, B. Coll, and J.-M. Morel, "A non-local algorithm for image denoising," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05)*, vol. 2, pp. 60–65, IEEE, San Diego, Calif, USA, June 2005.

Hindawi

Submit your manuscripts at
<http://www.hindawi.com>