International Journal of Modern Physics C
Vol. 29, No. 8 (2018) 1850071 (29 pages)

DOI: 10.1142/S0129183118500717

Enhancing LBP by preprocessing via anisotropic diffusion

Mariane Barros Neiva*

Institute of Mathematics and Computer Science
University of São Paulo, USP, Avenida
Trabalhador São-Carlense, 400
13566-590, São Carlos, SP, Brazil
marianeneiva@usp.br

Patrick Guidotti

Department of Mathematics, University of California Irvine
340 Rowland Hall, Irvine, CA 92697, USA
gpatrick@math.uci.edu

Odemir Martinez Bruno

São Carlos Institute of Physics, University of São Paulo
São Carlos – SP, PO Box 369, 13560-970, Brazil

Scientific Computing Group – http://scg.ifsc.usp.br
bruno@ifsc.usp.br

Received 24 March 2018
Accepted 8 July 2018
Published 3 August 2018

The main goal of this paper is to study the addition of a new preprocessing step in order to improve local feature descriptors and texture classification. The preprocessing is implemented by using transformations which help highlight salient features that play a significant role in texture recognition. We evaluate and compare four different competing methods: three different anisotropic diffusion methods including the classical anisotropic Perona–Malik diffusion and two subsequent regularizations of it and the application of a Gaussian kernel, which is the classical multiscale approach in texture analysis. The combination of the transformed images and the original ones are analyzed. The results show that the use of the preprocessing step does lead to an improvement in texture recognition.

Keywords: Texture classification; anisotropic diffusion; image processing; texture enhancement.

PACS Nos.: 11.25.Hf, 123.1K.

1. Introduction

Computer vision has become an important tool for industrial, scientific and entertainment applications. The use of image and video processing in order to automate

*Corresponding author.

activities that used to require significant amount of time and effort is the main reason for this and justifies continued research towards better and better algorithms. An important task in vision is pattern recognition. It can be performed by using different characteristics of an image such as shape or color. Recently, the use of texture as an image feature proved to be a viable alternative.[22,18,10,6] While a precise definition of texture is not agreed upon, humans naturally perceive it and, in fact, make use of it to recognize objects. It is therefore important to study its efficacy in image recognition.

Over the years, many scientists have described methods to extract features from texture. These can be divided into six main categories: statistical, stochastic, structural, spectral, complexity- and agent-based methods.[26] Each one analyzes the image in a different way either by considering local patterns, Fourier spectrum, fractal dimension or other relevant quantities. However, more can be done. Usually, texture extraction methods are applied to the original dataset. If the image is not good and salient characteristics are not pronounced, texture analysis methods may not work as desired.

In the latter case, one possibility is to enhance the image prior to attempting a classification by texture. However, except in specific situations such as face recognition,[11,33] disease detection,[31] or other studies focused on a specialized application, little attention has been paid to studying the influence of smoothing on texture classification. In this direction, in Ref. 25 images are smoothed by the use of Gabor filters yielding encouraging classification results. In Ref. 28, the authors evaluate a series of preprocessing methods to improve co-occurrence-based descriptors. Like our paper, Ref. 9 uses Gaussian filters to process images in order to improve local descriptors. However, in this case, the filter is applied only to the training set and new test samples with blurred texture can or cannot be correctly classified. Anisotropic diffusions are not considered.

This paper aims at filling this gap by applying preprocessing methods to the original dataset and, subsequently, performing texture extraction by evaluating LBP and its extensions, methods that usually give good results in texture classification. Three databases are tested: Brodatz, Vistex and UspTex. While, here, the technique is applied to classification, the approach can be utilized in combination with any other task that requires texture features, such as segmentation, synthesis, detection and prediction.

As mentioned before, the core of the proposed method is the application of a preprocessing step. Inspired by physics, smoothing methods are an extension of the heat equation. The latter describes the distribution of heat under a certain region as a function of time t. Similarly, smoothing methods and the distribution of intensities can be understood as temperatures over a region consisting of different materials (represented by edges, intra-regions, among other aspects in a scene). The usual isotropic diffusion is directly linked to the well-known heat equation (Eq. (1)), where heat spreads uniformly over the surface. However, replacing the constant c by a function (related to the gradient, for instance), we can effectively control the amount of diffusion applied according to the material u (or region in the texture,

in this case).

$$\frac{\partial u}{\partial t} = c\Delta^2 u. \tag{1}$$

Therefore, different modalities of diffusion were chosen for this task in order to check whether a multiscale technique, based on taking the output of the method at a series of times (scales), improves feature extraction. The multiscale nature of vision appears in the cortex area where receptive fields separate bars, corners, lines and edges of an image.[23] Usually, multiscale resolution is achieved by a Gaussian kernel with different variances (i.e. values of σ). When a Gaussian filter is used, however, edges are altered, which are typically a very important feature of an image. Gaussian filtering amounts to applying isotropic diffusion which is insensitive to direction and hence to geometry. This problem can be overcome by the use of nonlinear anisotropic diffusion. Edges are implicitly located by the method, which then prevents their blurring, effectively confining the effect of diffusion within areas separated by the detected edges.

Beside isotropic diffusion, the paper also uses three different anisotropic diffusion methods: Perona *et al.* anisotropic diffusion,[30] and two regularizations of it proposed by Guidotti *et al.*: a forward–backward regularization (FBR) of Perona–Malik[13] and another using fractional derivatives (NL).[12] Experiments are performed with all methods and comparisons are presented. Without any prior hypothesis on whether isotropic or anisotropic methods are the best, the goal is to determine what preprocessing method enhances which characteristics analyzed by feature extraction.

Recently, deep learning networks have been used for many machine learning tasks such as image recognition. This is an important achievement for computer science but it does not solve all the problems and does not fit all situations. Many applications such as medical disease detection and biological analysis require public authorization from patients or the acquisition of images is manual and can be slow. Therefore, the amount of images is not suitable for deep learning networks since the number of weights that need to be calibrated is too high as compared to the training set.

Also, in some applications, it is important to understand the features being analyzed. In the neural network approach, these are implicit and not available to the developer and researcher. These are some of the reasons why handcrafted feature extraction procedures are still needed to create a meaningful representations of the texture. With this in mind, this paper has the goal of enhancing one of the most widely known and well-posed descriptors, the Local Binary Pattern method, proposed in 2002 by Ojala *et al.*[29] LBP, has become well known as a descriptor in computer vision due its effectiveness and simplicity. Even though it was proposed almost 16 years ago, the method is still being studied, applied and used as an inspiration in the development of other feature extraction methods.[2,7,20,24,32]

This paper uses these local pattern extractors and attempts to enhance them by using preprocessed images. The main contributions of this paper are the following:

- Analyze the influence of the application of four diffusion methods including the classical isotropic and the Perona–Malik method to enhance feature extraction by six different local extractors.
- Study the influence of the iteration parameter in each dataset to analyze the limitations of enhancement a diffusion method can deliver.
- Compare the proposed framework to the traditional approach where no preprocessing method is used.

The paper is structured as follows: Section 2 presents the texture description methods used: LBP, CLBP, LBPV, LBPFH, LTP and CSLBP, as well as briefly introduces the four diffusion methods. Section 3 details the proposed approach combining preprocessing with the descriptor methods along a classifier. Section 4 shows the results for texture classification using four datasets and, finally, Sec. 5 draws the main conclusions of our work.

2. Background

Different feature extraction methods will be used in order to test the possible advantages of applying diffusion methods prior to feature extraction. These descriptors along with the four preprocessing methods are described in this section.

2.1. *Feature extraction methods*

2.1.1. *Local binary pattern*

The core of all the methods described below is the analysis of pixels' neighborhood. The basic technique used to capture local features is the so-called Local Binary Pattern. In this method, all the pixels are used once as the main g_c, the pixel in the center of the window, and the sum of the differences to its P neighbors chosen within a radius r is used as a pattern. It is denoted by $\mathrm{LBP}_{P,r}$ and is shown in Eq. (2):

$$\mathrm{LBP}_{P,r}(I_{x_{g_c}}, I_{y_{g_c}}) = \sum_{p=0}^{P-1} s(g_p - g_c)\, 2^p, \quad s(x) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}. \tag{2}$$

As we want to create a 1D representation of the texture for classification, the LBP feature descriptor computes the histogram of the image output in Eq. (2) following the equation below:

$$H(k) = \sum_{i=1}^{M} \sum_{j=1}^{N} \mathrm{bool}(\mathrm{LBP}_{P,r}(i,j), k), \tag{3}$$

where $k \in [0, K]$, K is the maximum gray scale of the image, and

$$\text{bool}(x, y) = \begin{cases} 1, & x = y, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

The basic approach uses a 3×3 window, i.e. $P = 8$, $r = 1$. Therefore, the histogram, and, consequently, the feature vector, has $2^8 = 256$ entries. The method decodes the quantities of borders, corners and smoothed regions in the patterns histogram but can be sensitive to rotations. Some variations of the classical LBP overcome this problem, such as the one below.

2.1.2. *Local binary pattern variance*

In order to provide rotation invariant features, Ojala *et al.*[29] proposed a method that combines all basic LBP patterns into a single pattern if they only differ by a rotation. This is done by counting the number of bits that change from 0 to 1 or from 1 to 0. These patterns are computed based on a metric that checks uniformity:

$$U(\text{LBP}_{P,r}) = \sum_{p=1}^{P-1} |s(g_{p+1} - g_c) - s(g_p - g_c)|. \tag{5}$$

Therefore, uniform rotation invariant LBP is computed by

$$\text{LBP}_{P,r}^{riu2}(i, j) = \begin{cases} \displaystyle\sum_{p=0}^{P-1} s(g_p - g_c), & \text{if } U(\text{LBP}_{P,r}(i, j)) < 2, \\ P + 1, & \text{otherwise.} \end{cases} \tag{6}$$

Then, in order to add contrast information to $\text{LBP}_{P,r}^{riu2}$, Guo *et al.*[17] proposed the use of variance in combination with the uniform rotation invariant descriptor. It is computed by

$$\text{VAR}_{P,r} = \frac{1}{P} \sum_{p=0}^{P-1} \left[g_p - \left(\frac{1}{P} \sum_{p=0}^{P-1} g_p \right) \right]^2 \tag{7}$$

In practice, variance is used to compute the strength of each pattern $k \in [0, K]$ in the $\text{LBP}_{P,r}^{riu2}$ of an image I of size $M \times N$:

$$\text{LBPV}_{P,r}(k) = \sum_{i=1}^{M} \sum_{j=1}^{N} \omega(\text{LBP}_{P,r}^{riu2}(i, j), k), k \in [0, K], \tag{8}$$

$$\omega(\text{LBP}_{P,r}^{riu2}, k) = \begin{cases} \text{VAR}_{P,r}(i, j), & \text{if } \text{LBP}_{P,r}^{riu2} = k \\ 0, & \text{otherwise} \end{cases}, \tag{9}$$

finally, LBPV is used as feature vector for the image I.

2.2. *Complete local binary pattern*

One of the best extensions of classic LBP to date is the Complete Local Binary Pattern descriptor. Proposed by Guo *et al.*,[16] the technique improves the original method by adding information of sign and magnitude of the local patterns.

The sign and magnitude are computed from matrix LSDMT, where each pixel p is calculated as:

$$\mathrm{LSDMT}_p = g_c - g_p, \tag{10}$$

where, similarly as in LBP, g_c and g_p are the intensity values of the central pixel of the evaluated window and the value of the pixel at position p, respectively. Two matrices are computed from LSDMT: CLBP_S and CLBP_M. The first contains the sign of the entries of the original matrix, while the second is formed based on the absolute values.

Finally, three different quantities are used to generate the features: the image itself (CLBP_C), CLBP_S that represents the same information contained in LBP, and CLBP_M. The latter is a matrix of decimals and is given by

$$\mathrm{CLBP}_M^{P,r} = \sum_{p=0}^{P-1} t(|g_p - g_c|, c)2^p, \quad t(x,c) = \begin{cases} 1, & x \geq c \\ 0, & x < c \end{cases}, \tag{11}$$

where c is a threshold computed by the mean value of the $|\mathrm{LSDMT}_P|$ matrix and CSLBP_C is given by

$$\mathrm{CSLBP}_C = t(g_c, c). \tag{12}$$

Finally, a histogram computed by the concatenation of the three matrices above is used as feature vector in this method. Results show that the descriptor yields high recognition rates with the addition of sign and magnitude information of local patterns. Like LBP, we choose $P = 8$ and $r = 1$ in experiments.

2.3. *Local binary pattern histogram Fourier features*

This method has the advantage of being locally and globally invariant with respect to rotation,[1] a feature that differs from the previous methods cited above. It first computes the non-invariant LBP histograms and from them, the method calculates the global rotationally invariant features from the basic approach. Like the classical LBP, the method uses a 3×3 window around the central pixel to extract binary features but reduces the histogram by combining patterns that only differ by rotation into the same bin. Therefore, patterns such as 110 and 011 are identified by the method. The paper states that, if the original image is rotated by a certain degree $\alpha\left(\alpha = a\,\frac{360}{P}\right)$ where P is the number of pixels evaluated in each window, the histogram is also cyclically shifted. To reach global invariance, the Discrete Fourier Transform is computed as:

$$H(n,u) = \sum_{p=0}^{P-1} h(\mathrm{LBP}_{P,r}^{riu2}) e^{-i2\pi ur/P}, \tag{13}$$

where $\mathrm{LBP}_{P,r}^{riu2}$, defined in Eq. (6), is obtained by separation of the pattern in uniform and nonuniform categories. The uniform patterns are those with an up to three transitions 0–1 or 1–0 in the binary pattern, totaling 58 possibilities. The nonuniform patterns are reduced to one bin. Therefore, the feature vector is reduced from 256 to 59 bins, as shown in Ref. 17. Also, $\mathrm{h}(\mathrm{LBP}_{P,r}^{riu2})$ is the histogram value for the uniform pattern $\mathrm{LBP}_{P,r}^{riu2}$, n is related to the number of 1's in the pattern computed by $U(\mathrm{LBP}_{P,r})$ in Eq. (5), and $u \in [0, P-1]$. The original article shows that features in spectrum domain are invariant to cyclic features and therefore rotationally invariant.

2.4. *Local ternary pattern*

In this forth descriptor, the pattern considering a central pixel is not represented with two bits but three different values[33]: $\{-1, 0, 1\}$. In order to extract features around a certain value g_c with a small tolerance b and a neighboring pixel value g_p, thresholding is done as follows:

$$\begin{cases} 1, & \text{if } g_p > g_c + b, \\ 0, & \text{if } g_p > g_c - b \text{ and } g_p < g_c + b, \\ -1, & \text{if } g_p < g_c - b. \end{cases} \tag{14}$$

When computing the corresponding histogram, one notices that the number of features is very high. Therefore, in order to minimize the final vector size, two matrices are created collecting the negative values of the LTP image (lower patterns) and another one gathering the positive ones (upper patterns). Therefore, the final feature vector is the concatenation of the histograms of the lower and upper patterns. The information of the signal now includes direction analysis into the descriptor and can enhance the representation of LBP.

2.5. *Center-symmetric local binary pattern*

An additional local feature descriptor combines the power of LBP and the well-known SIFT[19] technique. The method first performs noise removal in each analyzed window. The filter is able to preserve edges while removing noise, improving the feature extraction power of LBP.

After filtering, a pattern is extracted for each pixel in the image similarly to what occurs in classic LBP. However, in this case, features are extracted comparing center-symmetric pairs of pixels, i.e. comparing pixels that are in the opposite position according to a radius around the central pixel. Thus, while LBP uses a 3×3 window to produce 256 features, CSLBP computes a feature vector of only 16 values. Additionally, the threshold used is set to a value $T(= 0.01$ in experiments), in order to improve the outcome in flat image regions:

$$\mathrm{CSLBP}(x, y) = \sum_{p=0}^{P/2-1} s(g_p - g_{p+(P/2)}) 2^p, \tag{15}$$

$$s(x) = \begin{cases} 1, & x > T, \\ 0, & \text{otherwise}, \end{cases} \tag{16}$$

where g_p and $g_{p+(P/2)}$ are the symmetric pairs. In the experiments a grid of size 4×4 is used obtaining a final histogram of 256 features.

2.6. *Diffusion methods*

The key methods analyzed in this paper are diffusion algorithms and their influence on texture recognition. Linear and nonlinear, isotropic and anisotropic diffusions are used to create a new set of transformed images at different scales to be used as input for the descriptors of Sec. 2.5. For reader's convenience, a brief description of each of them is given in the following section.

2.6.1. *Isotropic diffusion*

The simplest and oldest form of multiscale resolution is obtained by applying a Gaussian kernel to the texture with varying smoothing parameter σ. A 2D Gaussian filter is calculated by convolution with

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}. \tag{17}$$

For such an isotropic diffusion, smoothing is applied equally in all regions of the image and in all directions. Consequently, information from edges and from smooth regions get mixed up and shifted as the scale parameter σ is increased. High frequency areas such as edges and corners are, however, an important feature of texture and this information is lost in the multiscale pyramid obtained by this filtering procedure.

2.6.2. *Perona–Malik's anisotropic diffusion*

Perona *et al.*[30] were among the first to propose and to develop nonlinear anisotropic diffusion methods. Until then a multiscale resolution was obtained by convolving the image intensity function with a family of Gaussian filters with different variances (σ values) leading to a sequence of images of varying levels of detail as explained above. The higher the σ, the coarser the details. Convolution with a Gaussian is given by

$$I(x, y, t) = I_0(x, y) * G(x, y; t), \tag{18}$$

which can be thought the solution to the heat equation:

$$I_t = \nabla^2 I = I_{xx} + I_{yy}, \tag{19}$$

with initial datum $I_0(x, y)$ and, in this case, $t = \sigma^2$.

As mentioned before, using a Gaussian kernel to generate different levels of details, however, leads to the disappearance of edges, which are a source of important information on an image.

A multiscale pyramid can be visualized by means of a tree, where, at one level, the shape of the tree can be recognized, and, at the next level, branches and leaves are added. Eventually, increasing the resolution level, all details including leaves' veins are present. Regardless of resolution, edges are present and well defined and, as such, should be preserved for visual tasks.

This was one of the reasons which motivated Perona *et al.*[30] to propose the anisotropic diffusion given by

$$I_t = \operatorname{div}(c(x, y, t)\nabla I). \tag{20}$$

The goal was then to find a function c that would result in the output image satisfying three main requirements

Causality. No fake details should be generated in coarser resolutions.
Immediate Location. At each resolution, edges should be sharp and semantically meaningful to the actual resolution.
Piecewise Smoothing. Blurring should privilege intra-regions over inter-regions.

If c is constant, ∇c vanishes and the diffusion is linear and isotropic. As intra-region blurring is to be privileged over inter-region blurring, c should have a lower value at inter-region boundaries. Thus, in Ref. 30, c is related to the gradient of the image itself, which is used as an edge detector

$$c(x, y, t) = g(|\nabla I(x, y, t)|). \tag{21}$$

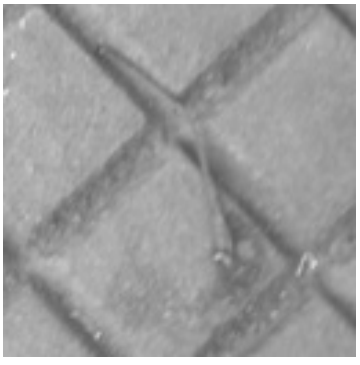
The function g is taken to be related to the inverse of the gradient square and controls the amount of diffusion applied in the region. Since, at inter-region boundaries, the gradient is large, blurring is lower. One of the functions proposed in Ref. 30 is given by

$$g(\nabla I) = \frac{1}{1 + \left(\frac{\|\nabla I\|}{\kappa}\right)^2}, \tag{22}$$

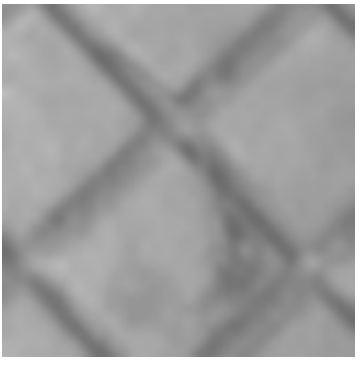
and will be used in the sequel. The parameter κ is user-defined and represents the threshold value beyond which sharp transitions are considered edges and are hence preserved. According to the authors, the chosen form of g privileges large regions. Figure 1 compares linear diffusion to the nonlinear, anisotropic diffusion in Ref. 30 applied in images of the Usptex dataset.[8] It is clear that edges are better maintained by the latter. While fake details, such as staircasing, are produced by this type of anisotropic diffusion, blur is effectively confined to regions between edges, delivering on at least two of the three properties above.

2.6.3. *Forward–Backward regularization diffusion*

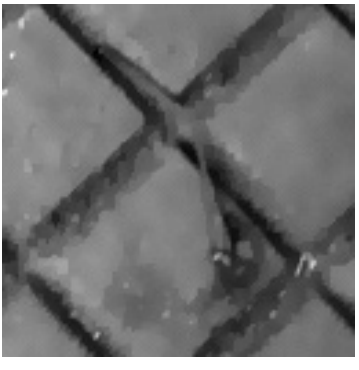
In Ref. 15, it is noted that, while Perona *et al.*[30] method is an interesting and effective model, it is not without shortcomings due to the generation of artificial edges,



(a) Original Image



(b) Gaussian Diffusion – $\sigma = 2.0$



(c) Perona–Malik Diffusion – 15th iteration

Fig. 1. Comparison between classic Gaussian diffusion and classic nonlinear diffusion. Images smoothed by Gaussian filter looses definition mainly in the edges. It can be useful for noise reduction but, if not used carefully, important features of the texture can be missed.

a phenomenon known as staircasing. Therefore, the authors propose a regularization characterized by two parameters $p \in (1, \infty)$ and $\delta > 0$ with the aim of avoiding staircasing without completely discarding the edge sharpening power of the original Perona–Malik model. The equation proposed reads as follows:

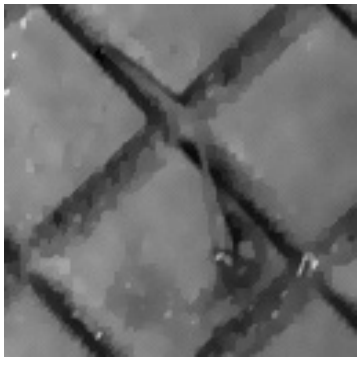
$$I_t = \nabla \cdot \left(\left[\frac{1}{1 + K^2 |\nabla I|^2} + \delta |\nabla I|^{p-2} \right] \nabla I \right). \tag{23}$$

As this model is a regularization of PM, the parameter K plays the same role as in Eq. (22), but is eventually overwhelmed by the δ term for very large gradients (this is precisely what prevents staircasing). According to the experiments in Ref. 15, best results are obtained for a parameter p which is close (but not equal) to 1. The addition of these two parameters allows the control of desired gradient growth and also effectively bounds it to a maximal size. This is due to the forward–backward nature of the equations and the fact that the backward regime is confined to

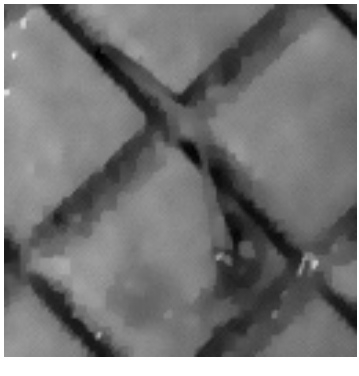
$[1 < |\nabla I| < M(K, \delta, p)]$, where M is a function of the given variables (see Ref. 13). As a consequence, staircasing is replaced by (micro-)ramping, the steepness of which is controlled by the values of p and δ. The right choice of the parameters is essential. For $p = 1$, for instance, the backward regime is unbounded, leading to staircasing. An experiment using $p = 2$ is shown in Ref. 13. Reference 15 shows good results when employing Eq. (23) for denoising and deblurring and it will be tested here as a feature enhancer for texture classification.

Figure 2 shows the comparison between the three approaches (applied on images from dataset Usptex[8]) explained so far. The visual difference between Gaussian and anisotropic methods is evident. However, comparing both anisotropic methods, it is hard to see the micro-ramping versus staircasing while it does exist as can be seen in Ref. 15.

(a) Gaussian Diffusion – $\sigma = 2.0$



(b) Perona–Malik Diffusion – 15th iteration



(c) Forward–Backward Regularization Diffusion – 15th iteration

Fig. 2. Comparison between classic Gaussian diffusion, classic nonlinear diffusion (PM) and a regularization of PM. The differences between images produced from isotropic and anisotropic diffusion are easier to notice. While isotropic diffusion smooths the whole image equally, anisotropic methods can distinguish between edges and smooth regions. However, there are still differences between the anisotropic diffusions. Using the same parameter values ($K = 1$ and $t = 15$), it can be observed that intra-regions are less noisy in Forward–Backward Regularization diffusion image (Fig. (c)) as compared to the output of PM.

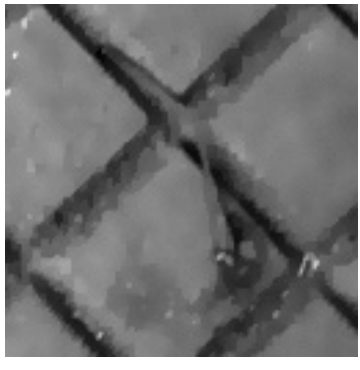
2.6.4. *Nonlocal anisotropic diffusion*

In Ref. 12, another approach is taken to regularize Perona–Malik and thus, effectively avoid staircasing without reducing Perona–Malik intrinsic sharpening feature. The trick is performed by the use of a fractional derivatives in the edge-detector

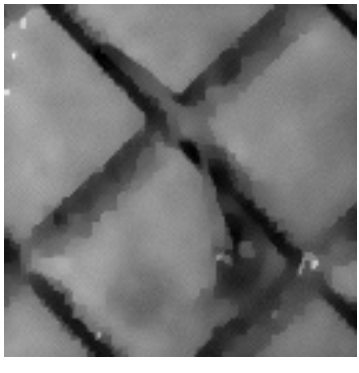
$$I_t = \nabla\left(\frac{1}{1 + K^2|\nabla^{1-\varepsilon} I|^2}\nabla I\right), \quad \text{where } \varepsilon \in (0, 1). \tag{24}$$

The main advantages of Eq. (24) are its mathematical well-posedness, on the one hand, and its simultaneous tendency to promote intra-region smoothing while solidly preserving edges for a long time. Latter is due to the fact that, as stated in Guidotti *et al.,*[12] "...piecewise constant functions or characteristic functions of smooth sets in higher dimension can be shown to be equilibria for the evolution."

(a) Gaussian Diffusion – $\sigma = 2.0$



(b) Perona–Malik Diffusion – 15th iteration



(c) NonLocal Anisotropic Diffusion – 15th iteration

Fig. 3. Comparison between classic Gaussian diffusion, classic nonlinear diffusion and the non local diffusion. The same phenomena occur as in Fig. 2, when the output of anisotropic diffusion are compared. The nonlocal anisotropic is also a regularization of the Perona–Malik algorithm and therefore intra-regions are shown more smoothed due the reduction of the staircasing artifact.

If $\varepsilon = 0$, the equation reduces to Perona–Malik. However, positive values of ε require the calculation of fractional derivatives. This can be done by means of the Fourier transform considering periodic boundary conditions as follows:

$$|\nabla^{1-\varepsilon} I| = \mathcal{F}^{-1}(\mathrm{diag}[2\pi|k|^{-\varepsilon}]\mathcal{F}(|\nabla I|)), \tag{25}$$

where

$$\mathcal{F}(I)(k) = \int_{\Omega} e^{-2\pi i k \cdot x} I(x) dx, \quad k \in \mathbb{Z}^2, \tag{26}$$

and $\mathrm{diag}[(m_k)_{k\in\mathbb{Z}^2}]$ denotes matrix multiplication with diagonal entries given by the sequence $(m_k)_{k\in\mathbb{Z}^2}$.

This approach is capable of partially capturing nonlocal information due to the use of the "wider support" fractional derivatives and thus enhancing edge detection robustness in the presence of noise. This well-posed anisotropic diffusion can be effectively used for denoising purposes due to its ability to differentiate between high frequency global features (that really belong to the image) and local ones (noise). Further explanations and proofs are found in Ref. 12.

Figure 3 shows results very comparable to what Fig. 2 demonstrates, where differences between both anisotropic methods are very subtle, while compared to Gaussian diffusion, the difference is high in terms of edge preservation. All images are processed from Usptex dataset.[8]

3. Proposed Method

While typically used for deblurring and denoising purposes, Gaussian filters, Perona–Malik and the enhanced counterparts (23) and (24) can also be used as a basic tool to obtain multiresolutions of an image. For these algorithms, in fact, coarser and coarser images are created over time. We will take advantage of this augmented dataset to improve information to represent a single texture.

Motivated by human vision where dots, corners and lines are separated in the cortex for object recognition,[21] a multiscale resolution that can distinguish different features of the image and use them separately can deliver a better and easier input to feature extraction methods. At different iterations (denoted by the parameter it), different features appear, and the one that better represents the dataset will hopefully give a higher texture classification rate.

The method works as follows:

(1) First, the original dataset images are preprocessed with one of the diffusion methods presented above. This will create a pyramid with 150 ($1 \leq it \leq 150$) new images for each one in the dataset with different levels of details.
(2) All images in the new dataset (original and smoothed ones) are submitted to a feature extraction method which will transform the 2D matrix of pixels to a 1D representation of the texture.

(3) At each time, the concatenation of the feature vector produced by one iteration $v(I_i t)$ of a diffusion method and the representation of the original image v(I) are used to describe the texture.

Anisotropic methods are believed to have the advantage over isotropic ones to smooth while keeping important structures of the images such as edges and T-junctions. This advantage will be tested for feature extraction. Beyond just adding a preprocessing method to feature extraction, this paper has also the goal to compare the four different methods described above and analyze which one most enhances each type of texture features. A diagram of the approach is presented in Fig. 4, where all steps of the proposed method are represented. First, images are subjected to one of the diffusion methods studied (Fig. 4(b)). This will result in a set of different images at different iterations for each one of the original images in the dataset. Then, all the smoothed images and the original texture are submitted to one of the local extractors and generate 151 unidimensional vectors (Fig. 4(c)). Finally, we pick one of the described preprocessed images, $v_i^{\bar{i}}t$, at iteration it, and combine it with the original image vector, $v_0^{\bar{i}}$. The final feature vector of each texture in the dataset is then given by $\bar{v^i} = [v_0^{\bar{i}}, v_i^{\bar{i}}t]$ (Fig. 4(d)). The final feature vectors $\bar{v^i}$ of each image i are evaluated

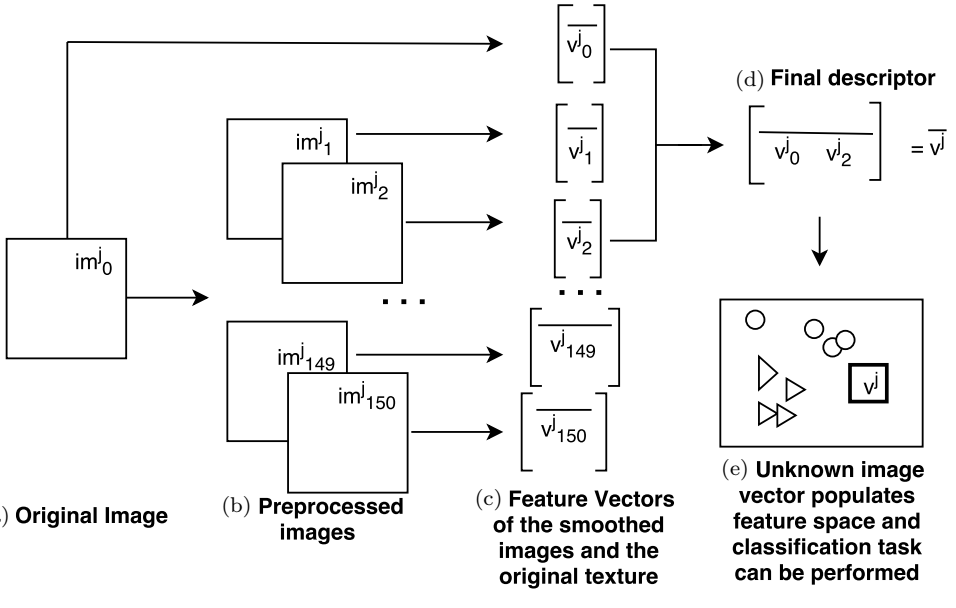


Fig. 4. Diagram of the proposed method. (a) The original image i is preprocessed by one of the diffusion methods previously described to obtain a multiscale resolution (b). Then, (c) all the images obtained in the preprocessing step (in the experiments 150 iterations are output) and the original texture i are described by one of the chosen feature extractor methods. The final vector (d) is composed by one of the feature vectors obtained in the preprocessing step (in the example iteration 2 is chosen) and the descriptor of the original image. Finally (e), the vector $\bar{v^i}$ of image i populates the feature space and classification can be performed according to the machine learning algorithm scheme chosen.

by a classifier to retrieve a corrected classification rate of recognition. In order to evaluate the approach, all possible combinations of preprocessing method × diffusion method × classifier (Fig. 4(e)) are tested. As each preprocessing method output 150 smoothed images for a single texture (all feature vectors $\bar{v^i} = [\bar{v_0^i}, \bar{v_i^i}t]$ with $1 \leq it \leq 150$) it means that, for each combination mentioned above, 150 CCR and the classification of the dataset with no preprocessing are output.

In the study, six feature extractions are analyzed chosen mainly by the importance of local binary pattern descriptor in the literature. The good results obtained by this simple technique caught the interest of researchers and lead to attempts at extending the method. This paper also offers a different approach in order to improve the classical LBP, as well as some extensions of it. For the recognition step, K-Nearest Neighbor and Naive Bayes will be applied as a simple yet powerful classification model. Also, although SVM have been widely used for the community, Ref. 3 shows that in many cases, the simple KNN, $k = 1$ performs pretty well for classification.

4. Experimental Results

4.1. *Datasets*

Three texture datasets are used to evaluate the approach:

- **Brodatz Dataset:**[a] A well known, widely used dataset to test texture related algorithms. The set contains 1110 images from 111 different classes. Images are grayscale and of size 200 × 200. Brodatz is characterized by a lack of illumination, scale, and viewpoint variation.[5] As a well-known database of textures, it should be tested anyway.
- **Vistex Dataset:**[b] The Vistex dataset contains a total of 864 RGB color images, 16 in each class.[27] It contains variations in illumination, conditions, and scale. Images are converted to grayscale using only the luminance component and have size of 128 × 128.
- **UspTex Dataset:**[c] The dataset contains 12 color images per class for a total of 2292 images of size 128 × 128.[4] Each class set is generated by taking nonoverlapping windows of an original 512 × 384 image. All images are converted to grayscale maintaining only image luminance and removing hue and saturation information prior to preprocessing.

4.2. *Performance evaluation*

For simplicity and efficiency, KNN ($k = 1$) and Naive Bayes are used. A cross validation (10-fold) was performed to get a more reliable classification rates. For each

[a]Images from Brodatz dataset available at: http://multibandtexture.recherche.usherbrooke.ca.
[b]Vistex dataset available at: http://vismod.media.mit.edu/.
[c]Usptex dataset available at: http://scg.ifsc.usp.br.

dataset and preprocessing method, 150 new images in different scales were generated ($1 \leq it \leq 150$).

For isotropic diffusion, σ starts at 0.5 and is incremented by 0.5 in each iteration ($\sigma = 0.5 * it$). For the first anisotropic method, the implementation was done following Ref. 30 with g as in (22). For FBR (Sec. 2.6.3), parameters were chosen to be $\delta = 0.1$, $p = 1.1$ and finally, for nonlocal anisotropic diffusion presented in Sec. 2.6.4, ε was set to 0.1. The time derivation for convolution was set to 0.25 and $K = 1$ for all methods. It is important to notice that it is not possible to strictly compare the values of iteration for different diffusion methods, since they have inner parameters and their own characteristics. However, the higher the iteration is, the coarser the image becomes, a phenomenon that occurs in all smoothing methods. In this paper, we are willing to analyze what the limit of time is, for which each preprocessing method can enhance intrinsic features capture d by the descriptors and which one is the best iteration. However, further experiments could be done evaluate the influence of the parameter set of each diffusion algorithm.

Each scale is evaluated separately in order to analyze which one better improves feature extraction (Fig. 4). We combine the original image feature vector and the smoothed image vector from iteration it to use as input for the KNN classification space. As datasets have different characteristics, each method works in a different way and the it scale that yields the best improvement in feature extraction is different.

Analyzing the results for each dataset, the easiest, Brodatz, shows the best performance when FBR is combined with the CLBP descriptor. It can be noted that any combination shown in Tables 1 and 2 improves the correct classification rate (CCR) when compared to the traditional approach. The largest benefit is observed for CSLBP with a gain of 5.94%. In general, FBR is the method that most improves the classification rate for this dataset for both classifications.

The second dataset, Vistex, yields the best classification rate combining FBR and CLBP (KNN, $k = 1$). Similarly to Brodatz, images are better characterized when FBR is applied even though all combinations result in enhanced texture classification. The method with the highest benefit in this case is LBPHF with an improvement of up to 11.23%.

Finally, for Usptex, the improvement in texture recognition obtained with the addition of images preprocessed by FBR method was 6.33% for the same CLBP method. This set was the one that was most benefited from the addition of the processed images. The difference between CCR output by the addition of a diffusion and the CCR output by any traditional approach, descriptors applied on the original images, are higher compared to other datasets. The average improvement is of 10.25% with a lowest gain of around 6%, when considering all descriptors tested in this experiment.

Figures 5–10 show that the use of a Gaussian kernel, the classical preprocessing approach, resulted in a CCR gain only for small iterations, low values of σ, such as shown in Figs. 6 and 7. This is due the fact that edges disappear at higher scales,

Table 1. Best results for KNN ($k = 1$) of all combinations considering the 150 iterations. In the table, only the CCR (correct classification rate) of the iteration that combined with the original image gave the highest result is shown. Also, along with the CCR, the standard deviation value, AUC (area under ROC curve), F-measure and precision are also presented. It can be noted that anisotropic diffusion yields better results compared with no preprocessing method or isotropic diffusion in most cases.

	KNN, $k = 1$														
	Brodatz					Vistex					Usptex				
Method	CCR (std)	Best it	AUC	F-measure	Precision	CCR (std)	Best it	AUC	F-measure	Precision	CCR (std)	Best it	AUC	F-measure	Precision
LBP	93.96 (1.70)	—	0.9696	0.9381	0.9440	94.21 (2.74)	—	0.9693	0.9402	0.9442	73.65 (2.47)	—	0.8657	0.7338	0.7531
G+LBP	95.59 (1.19)	4	0.9782	0.9562	0.9615	97.69 (1.90)	10	0.9793	0.9648	0.9661	79.01 (1.58)	5	0.8910	0.7796	0.7974
PM+LBP	**97.66 (0.77)**	22	0.9886	0.9773	0.9804	98.61 (1.75)	73	0.9891	0.9813	0.9830	82.72 (1.69)	31	0.9092	0.8156	0.8298
FBR+LBP	96.21 (0.83)	20	0.9855	0.9706	0.9739	**98.73 (1.36)**	82	0.9902	0.9837	0.9850	**84.55 (2.29)**	33	0.9203	0.8379	0.8508
NL+LBP	96.76 (1.77)	124	0.9814	0.9625	0.9687	95.60 (2.21)	130	0.9772	0.9544	0.9571	77.84 (1.57)	140	0.8841	0.7665	0.7812
CLBP	96.21 (0.99)	—	0.9882	0.9759	0.9783	98.96 (0.98)	—	0.9962	0.9931	0.9938	83.16 (2.10)	—	0.9262	0.8493	0.8638
G+CLBP	97.21 (0.99)	2	0.9891	0.9780	0.9818	98.96 (1.43)	2	0.9946	0.9908	0.9914	85.21 (2.59)	1	0.9304	0.8574	0.8701
PM+CLBP	97.84 (0.97)	134	0.9877	0.9752	0.9796	99.54 (0.98)	14	0.9962	0.9931	0.9934	85.57 (1.81)	127	0.9394	0.8774	0.8859
FBR+CLBP	**97.93 (1.06)**	48	0.9891	0.9779	0.9809	**99.88 (0.78)**	149	0.9960	0.9919	0.9923	**89.44 (1.39)**	53	0.9488	0.8948	0.9021
NL+CLBP	97.84 (0.77)	50	0.9882	0.9762	0.9797	99.42 (1.46)	10	0.9948	0.9919	0.9923	85.86 (2.36)	145	0.9328	0.8608	0.8734
LBPV	94.23(1.76)	—	0.9709	0.9410	0.9487	92.82 (3.14)	—	0.9709	0.9410	0.9487	73.30 (3.62)	—	0.8669	0.7305	0.7502
G+LBPV	95.41 (1.50)	4	0.9768	0.9538	0.9583	**94.56 (3.16)**	3	0.9782	0.9553	0.9580	75.57 (2.62)	10	0.8721	0.7481	0.7678
PM+LBPV	95.68 (1.06)	43	0.9746	0.9483	0.9552	94.44 (4.59)	37	0.9728	0.9448	0.9523	79.14 (2.66)	10	0.8928	0.7869	0.8031
FBR+LBPV	**96.22 (1.44)**	64	0.9777	0.9551	0.9595	**94.56 (4.01)**	22	0.9759	0.9503	0.9576	**79.71 (2.65)**	36	0.8930	0.7875	0.8075
NL+LBPV	95.23 (1.31)	105	0.9732	0.9460	0.9526	93.75 (3.95)	60	0.9718	0.9437	0.9505	79.06 (3.00)	107	0.8907	0.7780	0.7957

Table 1. (*Continued*)

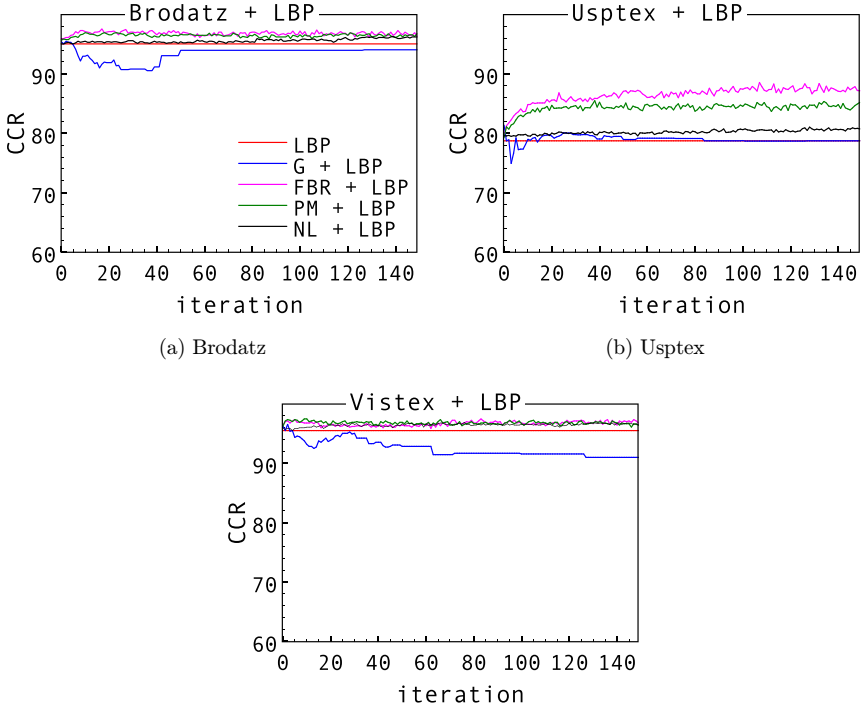
	KNN, $k = 1$														
	Brodatz					Vistex					Usptex				
Method	CCR (std)	Best it	AUC	F-measure	Precision	CCR (std)	Best it	AUC	F-measure	Precision	CCR (std)	Best it	AUC	F-measure	Precision
LBPHF	93.33 (1.86)	—	0.9664	0.9326	0.9373	82.29 (3.41)	—	0.9065	0.8179	0.8243	56.11 (1.50)	—	0.9065	0.8179	0.8243
G+LBPHF	94.95 (1.14)	6	0.9700	0.9383	0.9435	90.39 (2.54)	10	0.9411	0.8822	0.8825	66.36 (2.11)	5	0.9344	0.8654	0.8714
PM+LBPHF	96.31 (0.79)	38	0.9823	0.9646	0.9692	92.82 (3.37)	138	0.9573	0.9140	0.9204	69.42 (2.99)	24	0.9561	0.9142	0.9188
FBR+LBPHF	**96.40 (1.42)**	13	0.9800	0.9594	0.9635	**93.52 (2.19)**	53	0.9689	0.9361	0.9401	**74.21 (2.18)**	110	0.9696	0.9401	0.9461
NL+LBPHF	94.95 (1.52)	148	0.9737	0.9464	0.9523	87.27 (1.99)	150	0.9377	0.8757	0.8785	64.83 (2.90)	120	0.9323	0.8652	0.8705
LTP	93.96 (1.70)	—	0.9696	0.9381	0.9440	94.21 (2.74)	—	0.9693	0.9402	0.9442	73.65 (2.47)	—	0.8657	0.7338	0.7531
G+LTP	95.59 (1.19)	4	0.9782	0.9562	0.9615	97.69 (1.90)	10	0.9793	0.9648	0.9661	79.01 (1.58)	5	0.8910	0.7796	0.7974
PM+LTP	**97.66 (0.77)**	22	0.9886	0.9773	0.9804	98.61 (1.75)	73	0.9891	0.9813	0.9830	82.72 (1.69)	31	0.9092	0.8156	0.8298
FBR+LTP	96.94 (1.37)	8	0.9859	0.9711	0.9732	**98.73 (1.36)**	82	0.9902	0.9837	0.9850	**84.55 (2.29)**	33	0.9203	0.8379	0.8508
NL+LTP	96.76 (1.77)	124	0.9814	0.9625	0.9687	95.60 (2.21)	130	0.9772	0.9544	0.9571	77.84 (1.57)	140	0.8841	0.7665	0.7812
CSLBP	86.49 (2.33)	—	0.8657	0.7338	0.7531	82.87 (4.76)	—	0.9071	0.8239	0.8354	56.50 (3.59)	—	0.7835	0.5610	0.5761
G+CSLBP	91.89 (2.13)	4	0.8841	0.7665	0.7812	**93.63 (2.51)**	7	0.9621	0.9324	0.9388	**69.46 (2.09)**	6	0.8456	0.6857	0.7022
PM+CSLBP	**92.43 (1.27)**	95	0.9591	0.9154	0.9205	91.44 (3.00)	93	0.9502	0.9018	0.9090	68.98 (2.90)	140	0.8387	0.6729	0.6921
FBR+CSLBP	**92.43 (1.35)**	65	0.9605	0.9179	0.9255	92.82 (3.10)	62	0.9587	0.9192	0.9292	67.84 (2.77)	44	0.8418	0.6786	0.7009
NL+CSLBP	90.63 (1.91)	142	0.9482	0.8934	0.8984	86.00 (4.74)	119	0.9242	0.8524	0.8626	62.52 (2.41)	147	0.8025	0.6048	0.6215

Table 2. In the table, only the CCR (correct classification rate) of the iteration (iterations range from 1 to 150) is shown that, combined with the original image, gave the highest result, considering the combination dataset, preprocessing method, descriptor and Naive Bayes. Also, along with the CCR, the standard deviation value, AUC (area under ROC curve), F-measure and precision are also shown. Again, the best results are observed that FBR diffusion yields the best results compared to methods with no preprocessing or to other diffusion methods in most cases.

	Naive Bayes														
	Brodatz					Vistex					Usptex				
Method	CCR (std)	Best it	AUC	F-measure	Precision	CCR (std)	Best it	AUC	F-measure	Precision	CCR (std)	Best it	AUC	F-measure	Precision
LBP	95.05 (1.60)	—	0.9978	0.9521	0.9564	95.49 (3.27)	—	0.9988	0.9552	0.9567	78.71 (1.44)	—	0.9918	0.7984	0.8207
G+LBP	95.50 (1.82)	3	0.9945	0.9471	0.9538	96.53 (1.32)	3	0.9977	0.9630	0.9643	81.33 (2.12)	1	0.9899	0.8309	0.8485
PM+LBP	96.94 (1.20)	11	0.9957	0.9708	0.9742	**97.45 (1.82)**	10	0.9990	0.9729	0.9745	85.47 (1.63)	39	0.9874	0.8415	0.8562
FBR+LBP	**97.57 (1.80)**	18	0.9967	0.9708	0.9749	97.45 (1.20)	84	0.9972	0.9673	0.9684	**88.53 (2.47)**	108	0.9881	0.8738	0.8832
NL+LBP	96.31 (1.63)	140	0.9956	0.9628	0.9672	96.88 (3.08)	77	0.9988	0.9641	0.9649	81.02 (1.47)	129	0.9872	0.8090	0.8305
LBPV	91.26 (2.72)	—	0.9964	0.9150	0.9234	79.75 (4.33)	—	0.9848	0.7944	0.8046	60.60 (1.76)	—	0.9592	0.6023	0.6383
G+LBPV	**93.96 (2.34)**	4	0.9962	0.9348	0.9430	**90.39 (3.02)**	4	0.9945	0.9129	0.9204	68.41 (3.04)	1	0.9749	0.6936	0.7202
PM+LBPV	93.51 (2.76)	76	0.9945	0.9350	0.9422	86.81 (4.10)	1	0.9914	0.8708	0.8816	73.12 (2.87)	11	0.9812	0.7413	0.7709
FBR+LBPV	93.06 (2.55)	27	0.9946	0.9324	0.9401	86.92 (3.98)	1	0.9914	0.8722	0.8825	73.21 (2.73)	7	0.9824	0.7458	0.7726
NL+LBPV	92.25 (2.86)	2	0.9950	0.9256	0.9322	87.27 (3.98)	48	0.9915	0.8765	0.8856	**73.65 (2.89)**	57	0.9827	0.7430	0.7645
CLBP	96.67 (1.35)	—	0.9983	0.9751	0.9793	97.34 (1.50)	—	0.9999	0.9802	0.9811	84.03 (2.17)	—	0.9936	0.8642	0.8801
G+CLBP	96.85 (1.48)	1	0.9953	0.9714	0.9753	97.80 (1.06)	3	0.9991	0.9780	0.9795	85.86 (2.14)	1	0.9917	0.8760	0.8912
PM+CLBP	97.84 (1.20)	40	0.9972	0.9758	0.9794	98.15 (1.74)	55	0.9999	0.9804	0.9814	89.66 (2.31)	35	0.9926	0.9010	0.9106
FBR+CLBP	**97.93 (1.14)**	73	0.9975	0.9773	0.9801	**99.42 (1.35)**	150	0.9998	0.9805	0.9817	**90.36 (2.24)**	115	0.9879	0.8983	0.9086
NL+CLBP	97.66 (0.87)	140	0.9967	0.9769	0.9806	98.50 (2.46)	3	0.9998	0.9804	0.9815	87.09 (2.35)	138	0.9927	0.8785	0.8919

Table 2. (*Continued*)

	Naive Bayes														
	Brodatz					Vistex					Usptex				
Method	CCR (std)	Best it	AUC	F-measure	Precision	CCR (std)	Best it	AUC	F-measure	Precision	CCR (std)	Best it	AUC	F-measure	Precision
LBPHF	94.32(1.65)	—	0.9983	0.9446	0.9509	94.33 (2.09)	—	0.9991	0.9429	0.9442	73.34 (2.33)	—	0.9867	0.7385	0.7590
G+LBPHF	94.95 (1.46)	3	0.9978	0.9491	0.9549	96.41 (1.90)	5	0.9976	0.9432	0.9471	79.41 (2.19)	1	0.9919	0.8032	0.8170
PM+LBPHF	96.58 (1.41)	148	0.9993	0.9617	0.9643	96.99 (2.39)	73	0.9992	0.9597	0.9633	84.16 (1.65)	124	0.9928	0.8273	0.8412
FBR+LBPHF	**96.67 (2.08)**	66	0.9993	0.9644	0.9669	**97.22 (1.57)**	150	0.9977	0.9601	0.9638	**84.64 (2.27)**	68	0.9921	0.8310	0.8456
NL+LBPHF	95.14 (1.41)	145	0.9977	0.9500	0.9552	95.14 (1.81)	5	0.9991	0.9508	0.9525	78.14 (2.34)	136	0.9908	0.7799	0.7982
LTP	95.05(1.60)	—	0.9961	0.9521	0.9564	95.49 (3.27)	—	0.9987	0.9552	0.9567	78.71 (1.44)	—	0.9895	0.7984	0.8207
G+LTP	95.59 (1.77)	3	0.9884	0.9485	0.9551	96.53 (1.28)	3	0.9949	0.9641	0.9654	81.28 (2.02)	1	0.9777	0.8302	0.8480
PM+LTP	97.03 (1.20)	11	0.9940	0.9709	0.9744	**97.45 (1.82)**	10	0.9979	0.9729	0.9745	85.43 (2.23)	135	0.9745	0.8466	0.8609
FBR+LTP	**97.39 (1.66)**	18	0.9935	0.9699	0.9745	**97.45 (1.20)**	84	0.9955	0.9674	0.9684	**88.74 (2.39)**	108	0.9782	0.8777	0.8878
NL+LTP	96.31 (1.63)	140	0.9926	0.9628	0.9672	96.88 (3.08)	77	0.9966	0.9641	0.9649	81.02 (1.29)	129	0.9731	0.8085	0.8298
CSLBP	86.40(2.27)	—	0.9941	0.8688	0.8832	81.37 (3.45)	—	0.9937	0.8108	0.8135	58.16 (2.82)	—	0.9802	0.5789	0.5869
G+CSLBP	90.45 (1.99)	5	0.9953	0.8942	0.9061	91.09 (3.14)	6	0.9960	0.8926	0.8979	68.24 (3.18)	6	0.9816	0.6245	0.6588
PM+CSLBP	91.89 (1.80)	130	0.9972	0.9209	0.9314	90.16 (2.71)	148	0.9976	0.8996	0.9031	74.83 (1.83)	146	0.9926	0.7352	0.7483
FBR+CSLBP	**92.07 (2.17)**	83	0.9966	0.9171	0.9290	**91.55 (2.55)**	86	0.9975	0.9117	0.9159	**76.18 (2.55)**	142	0.9911	0.7512	0.7688
NL+CSLBP	88.92 (1.69)	134	0.9957	0.8857	0.8977	85.19 (3.33)	146	0.9954	0.8478	0.8509	64.05 (3.15)	122	0.9846	0.6251	0.6359

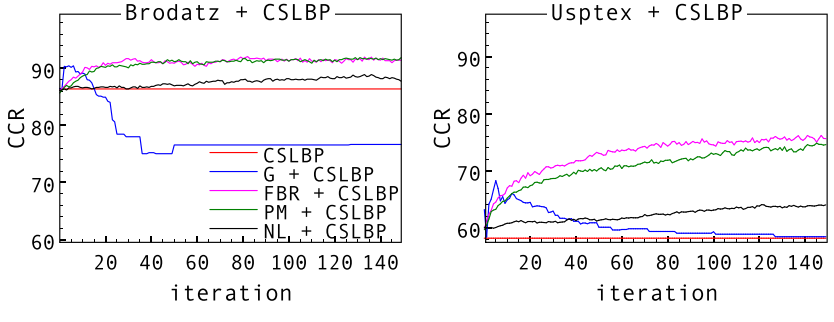


(a) Brodatz (b) Usptex

(c) Vistex

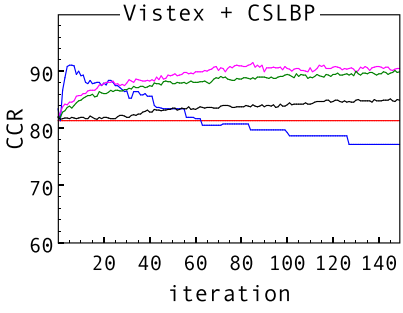
Fig. 5. (Color online) Results of each iteration when preprocessing methods are combined with LBP (Naive Bayes).

an inherent property of the method. In Figs. 8–10 almost no iteration (excluding few cases in Vistex and Usptex) results in an improvement compared to the traditional approach. In the case of LBP, LBPV, LTP and CSLBP, all 150 iterations of anisotropic image processing improve texture extraction and increase the recognition



(a) Brodatz (b) Usptex

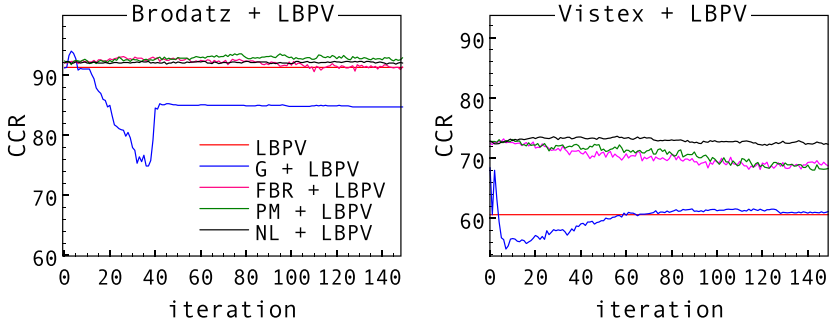
Fig. 6. (Color online) Results of each iteration when preprocessing methods are combined with CSLBP (Naive Bayes).



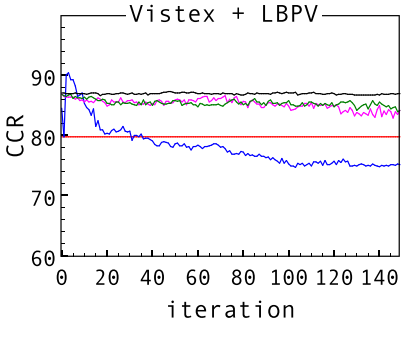
(c) Vistex

Fig. 6. (*Continued*)

rate. However, for LBPHF some iterations output a negative result for Brodatz and Vistex. In addition, graphs show a high similarity among preprocessing methods PM and FBR. However, as stated in Ref. 14, the classic anisotropic diffusion results in images with evident staircasing effect and, eventually, lower recognition rates.

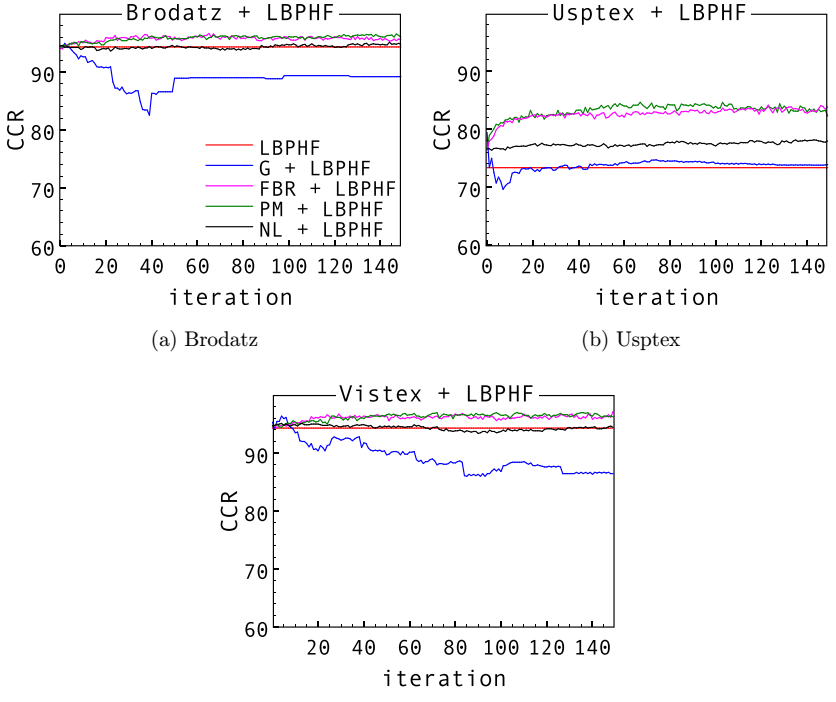


(a) (b)



(c)

Fig. 7. (Color online) Results of each iteration when preprocessing methods are combined with LBPV (Naive Bayes).



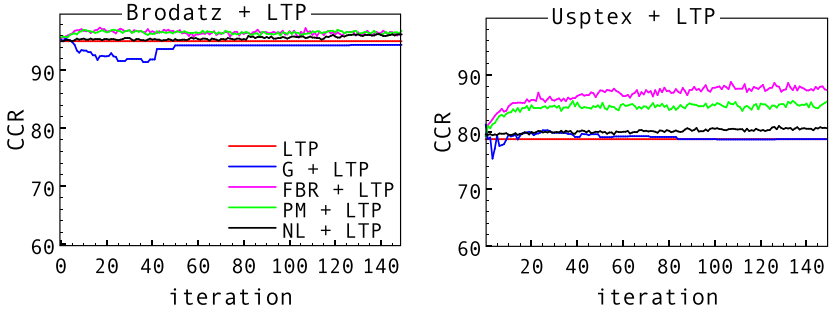
(a) Brodatz (b) Usptex

(c) Vistex

Fig. 8. (Color online) Results of each iteration when preprocessing methods are combined with LBPHF (Naive Bayes).

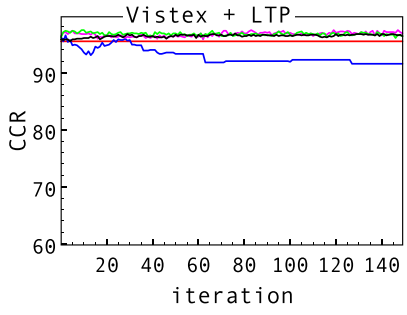
4.3. *Parameter evaluation*

As noticed in the previous tables, while all iterations were tested, we only present the best results for each combination set = {dataset, diffusion method, feature extractor, classifier}. Therefore, one can think that the benefit is only because



(a) Brodatz (b) Usptex

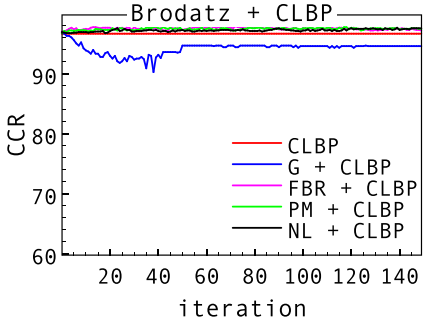
Fig. 9. (Color online) Results of each iteration when preprocessing methods are combined with LTP (Naive Bayes).



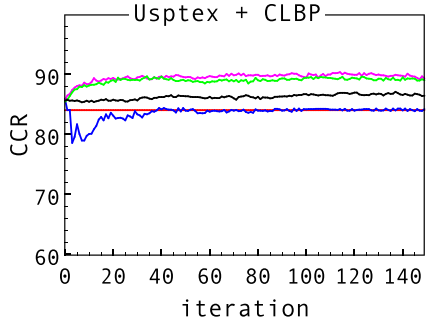
(c) Vistex

Fig. 9. (*Continued*)

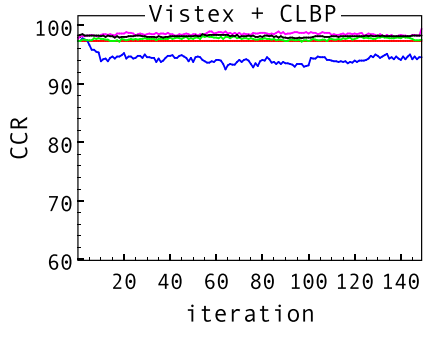
choosing the iteration that was set exactly to improve the most improved result. However, it is possible to check in Figs. 5–8, and 10 that, especially for anisotropic diffusion, the most smoothed images in any level of details enhance intrinsic features of the texture.



(a) Brodatz



(b) Usptex



(c) Vistex

Fig. 10. (Color online) Results of each iteration when preprocessing methods are combined with CLBP (Naive Bayes).

Table 3. As the previous tables show the best result for each combination, we analyzed the output for the average iteration considering the three datasets, the diffusion method and the feature extractor. For instance, in Table 1, the best iterations for G + LBP (KNN, $k = 1$) is the set $= \{4, 10, 5\}$. Therefore, we evaluate the three dataset using iteration 6 for this combination. Results show that even the average iteration helps the descriptor to better represent the texture and like what was noted in the previous tables, FBR was the best diffusion method among the combinations evaluated.

	KNN, $k = 1$				Naive Bayes			
	Average iteration	Brodatz	Vistex	Usptex	Average iteration	Brodatz	Vistex	Usptex
LBP	—	93.96 (1.70)	94.21 (2.74)	73.65 (2.47)	—	95.05 (1.60)	95.49 (3.27)	78.71 (1.44)
G+LBP	6	95.41 (1.16)	97.22 (1.98)	78.31 (1.35)	2	95.05 (1.60)	95.82 (2.92)	80.50 (1.34)
PM+LBP	42	**96.40 (1.53)**	**97.45 (1.32)**	81.37 (2.31)	20	96.13 (1.95)	96.64 (2.66)	83.16 (1.88)
FBR+LBP	52	**96.31 (1.23)**	**97.45 (1.62)**	**84.53 (2.02)**	70	**96.67 (1.47)**	**96.75 (2.56)**	**87.83 (2.46)**
NL+LBP	131	95.77 (1.53)	95.95 (2.34)	77.75 (2.32)	115	96.31 (1.31)	96.64 (2.29)	80.32 (1.56)
CLBP	—	97.21 (0.99)	98.96 (0.98)	83.16 (2.10)	—	96.67 (1.35)	97.34 (1.50)	84.03 (2.17)
G+CLBP	2	97.75 (1.36)	99.19 (0.95)	85.19 (2.23)	2	97.21 (1.61)	97.38 (1.47)	88.57 (1.61)
PM+CLBP	92	97.93 (1.20)	**99.30 (0.81)**	88.49 (1.64)	43	97.75 (1.42)	**98.61 (1.20)**	89.75 (1.36)
FBR+CLBP	83	**98.11 (1.16)**	99.19 (0.78)	**89.33 (2.29)**	113	97.75 (1.55)	98.38 (1.36)	**90.13 (1.78)**
NL+CLBP	68	97.30 (0.98)	97.11 (1.16)	85.87 (2.20)	94	**98.11 (1.31)**	98.37 (1.92)	88.70 (1.30)
LBPV	—	94.23 (1.76)	92.82 (3.14)	73.30 (3.62)	—	91.26 (2.72)	79.75 (4.33)	60.60 (1.76)
G+LBPV	6	**95.32 (1.63)**	93.49 (1.76)	75.57 (3.10)	3	**93.60 (2.27)**	**90.09 (1.82)**	**77.57 (2.45)**
PM+LBPV	30	94.77 (1.40)	**93.75 (2.21)**	79.08 (2.97)	29	91.08 (2.86)	85.99 (3.99)	71.51 (2.95)
FBR+LBPV	41	94.95 (0.97)	**93.75 (1.65)**	**79.50 (3.04)**	12	90.90 (3.13)	86.10 (3.90)	71.90 (2.84)
NL+LBPV	91	94.77 (1.58)	93.68 (1.65)	78.18 (3.10)	36	91.26 (2.44)	86.45 (3.74)	72.82 (3.22)

Table 3. (*Continued*)

	KNN, $k = 1$				Naive Bayes			
	Average iteration	Brodatz	Vistex	Usptex	Average iteration	Brodatz	Vistex	Usptex
LBPHF	—	93.33 (1.86)	82.29 (3.41)	56.11 (1.50)	—	94.32 (1.65)	94.33 (2.09)	73.34 (2.33)
G+LBPHF	7	94.50 (1.31)	90.20 (2.93)	68.15 (2.65)	3	**94.86 (1.75)**	94.37 (2.26)	79.62 (3.23)
PM+LBPHF	67	95.59 (1.37)	88.89 (2.62)	63.91 (2.00)	115	94.59 (1.47)	**95.14 (2.18)**	84.42 (2.42)
FBR+LBPHF	59	95.59 (1.67)	**92.13 (2.03)**	**69.36 (3.05)**	95	**94.86 (2.33)**	**95.13 (2.11)**	**85.60 (2.30)**
NL+LBPHF	140	**95.77 (1.35)**	88.42 (1.75)	63.92 (2.47)	95	**94.86 (2.13)**	93.98 (2.18)	78.88 (2.30)
LTP	—	93.96 (1.70)	94.21 (2.74)	73.65 (2.47)	—	95.05 (1.60)	95.49 (3.27)	78.71 (1.44)
G+ LTP	6	95.41 (1.16)	97.22 (1.98)	78.31 (1.35)	2	95.05 (1.60)	95.82 (2.92)	80.50 (1.34)
PM+LTP	42	96.40 (1.53)	97.45 (1.32)	81.37 (2.31)	52	96.58 (1.98)	96.52 (1.82)	84.51 (2.63)
FBR+LTP	41	**96.76 (1.06)**	**97.80 (1.27)**	**84.52 (1.37)**	70	**96.67 (1.47)**	**96.87 (2.26)**	**88.00 (2.43)**
NL+LTP	131	95.77 (1.53)	95.95 (2.34)	77.75 (2.32)	115	96.31 (1.31)	96.64 (2.29)	80.37 (1.56)
CSLBP	—	86.49 (2.33)	82.87 (4.76)	56.50 (3.59)	—	86.40 (2.27)	81.37 (3.45)	58.16 (2.82)
G+CSLBP	6	**91.89 (2.29)**	**93.07 (1.64)**	**69.36 (2.32)**	6	90.45 (2.09)	90.96 (4.17)	69.50 (3.07)
PM+CSLBP	109	90.54 (1.81)	90.27 (3.38)	66.01 (2.82)	141	90.00 (2.23)	87.14 (2.96)	68.89 (2.37)
FBR+CSLBP	57	90.72 (2.21)	90.62 (2.16)	66.06 (3.02)	104	**91.26 (2.85)**	**91.26 (2.85)**	**73.52 (1.83)**
NL+CSLBP	136	88.47 (2.75)	84.02 (4.75)	58.68 (2.80)	134	88.38 (2.30)	83.90 (3.22)	63.83 (3.24)

Furthermore, in order to quantify this analysis, we took the average iteration, $\hat{it}$, for each combination of diffusion method plus feature extractor algorithm for a given classifier and evaluated the images using $\bar{v^i} = [\bar{v_0^i}, \bar{v_{\hat{it}}^i}]$. Results are shown in Table 3. Note that the results with the addition of a preprocessed image are better in any dataset.

For Vistex, for instance, the best result achieved using an average iteration is 99.19% against 99.80% considering the best iteration with FBR + CLBP. Usptex, a challenging dataset, exhibited its best results diverging in only 0.13% regarding both (best and average approaches), an insignificant amount considering the standard deviation.

Another observation is that, in general, FBR is the most stable method to enhance features in the study. In all tables, it was the method that, in general, presented the highest gain in terms of CCR comparing the result with and without the preprocessing step. However, Gaussian filtering, the isotropic diffusion, appears as the most instable filter. This is due to its tendency to shift and distort edges and homogeneous regions as the iteration count grows. The result of Perona–Malik method, PM, is also very similar to what occurs for FBR, which is good even though it is not a mathematically well-posed method.

In summary, the addition of the preprocessing method, especially by anisotropic diffusion, have been proved as a good approach to improve texture recognition even when the iteration parameter is not set specifically for a dataset. The use of an average iteration as an additional information to the image enhanced the intrinsic features of the image and higher CCRs were output.

5. Conclusion

This paper evaluates the influence of different diffusion methods, three of which are nonlinear and anisotropic, and one of which is linear and isotropic, as a preprocessing tool generating extended feature vectors for known texture classifiers. The goal is to verify whether the use of such a preprocessing step prior to feature extraction can enhance image characteristics and consequently increase texture recognition. Six texture analysis algorithms were tested (LBP, LBPV, CLBP, LBPHF, LTP and CSLBP) in combination with two different classifiers (KNN with $k = 1$ and Naive Bayes).

The experiments demonstrate that the use of any of these preprocessing methods can indeed improve texture extraction and pattern recognition. In particular, the combination of FBR and CLBP delivers the highest gain across all datasets. We also analyzed the stability with respect to parameter choices and, Figs. 5–10 as well as Table 3 indicate that anisotropic diffusion methods are very constant in terms of results regarding the preprocessed image used but, for Gaussian filtering, the iteration count cannot be very high due to edge distortion and loss of important image features.

LBP and its extended methods (cited in Sec. 2.1) analyze the contrast of texture and the use of anisotropic diffusion proves helpful in improving these features while removing unnecessary artifacts, which is very useful for pattern recognition. As for the question of which methods to use in practice, Tables 1 and 2 help in identifying a good candidate for the feature extraction methods considered in this paper. The area under ROC curve, F-measure and precision do not give us much information since the first is very close to 1 for almost all experiments, while precision and F-measure are very similar to the CCR results. More in general, however, if one chooses a feature extraction not considered here, the tip would be to first apply linear isotropic diffusion with small number of scales and check the results.

Acknowledgments

The authors acknowledge support from CNPq (National Counsel of Technological and Scientific Development) (Grant Nos. 132409/2014-3, 307797/2014-7 and 484312/2013-8), FAPESP (Grant No. 14/08026-1) and CAPES (Coordination for the Improvement of Higher Education Personnel) (Grant No. 9056169/D).

References

1. T. Ahonen, J. Matas, C. He and M. Pietikäinen, Rotation invariant image description with local binary pattern histogram fourier features, in *Scandinavian Conference on Image Analysis* (Springer, 2009), pp. 61–70.
2. H. Ajmal, S. Rehman, F. Hussain, M. Abbas, A. Khan, R. Young and M. S. Alam, Comparative study of local binary pattern and its shifted variant for osteoporosis identification, in *Pattern Recognition and Tracking XXIX*, Vol. 10649 (International Society for Optics and Photonics, 2018), p. 1064908.
3. D. R. Amancio, C. H. Comin, D. Casanova, G. Travieso, O. M. Bruno, F. A. Rodrigues and L. da F. Costa, A systematic comparison of supervised classifiers, *PloS One* **9**, e94137 (2014).
4. A. R. Backes, D. Casanova and O. M. Bruno, *Pattern Recognit.* **45**, 1984 (2012).
5. P. Brodatz, *Textures: A Photographic Album for Artists and Designers* (Dover Publications, New York, 1966).
6. A. F. Costa, G. Humpire-Mamani and A. J. M. Traina, An efficient algorithm for fractal analysis of textures, in *25th SIBGRAPI Conf. Graphics, Patterns and Images (SIBGRAPI)*, (IEEE, 2012), pp. 39–46.
7. G. Deep, L. Kaur and S. Gupta, *Comput. Methods Biomech. Biomed. Eng. Imaging Vis.* **6**, 155 (2018).
8. J. B. Florindo and O. M. Bruno, *Physica A, Stat. Mech. Appl.* **391**, 4909 (2012).
9. M. Gadermayr and A. Uhl, *EURASIP J. Image Video Process.* **2016**, 14 (2016).
10. O. García-Olalla, E. Alegre, L. Fernández-Robles and V. González-Castro, Local oriented statistics information booster (losib) for texture classification, in *2014 22nd Int. Conf. Pattern Recognition (ICPR)*, (IEEE, 2014), pp. 1114–1119.
11. R. Gross and V. Brajovic, An image preprocessing algorithm for illumination invariant face recognition, in *AVBPA*, Vol. 3 (Springer, 2003), pp. 10–18.

12. P. Guidotti, A new nonlocal nonlinear diffusion of image processing, *J. Differ. Equ.* **246**, 4731 (2009).
13. P. Guidotti, *J. Differ. Equ.* **252**, 3226 (2012).
14. P. Guidotti, *Adv. Stud. Pure Math.* **99**, 20XX (2014).
15. P. Guidotti, Y. Kim and J. Lambers, *SIAM J. Imaging Sci.* **6**, 1416 (2013).
16. Z. Guo, L. Zhang and D. Zhang, *IEEE Trans. Image Process.* **19**, 1657 (2010).
17. Z. Guo, L. Zhang and D. Zhang, *Pattern Recognit.* **43**, 706 (2010).
18. R. M. Haralick, *Proc. IEEE*, **67**, 786 (1979).
19. M. Heikkilä, M. Pietikäinen and C. Schmid, Description of interest regions with center-symmetric local binary patterns, in *Computer Vision, Graphics and Image Processing* (Springer, 2006), pp. 58–69.
20. S. Jia, B. Deng, J. Zhu, X. Jia and Q. Li, *IEEE Trans. Geosci. Remote Sens.* **56**, 749 (2018).
21. B. Julesz, *Sci. Am.* **232**, 34 (1975).
22. J. K. Kim and H. W. Park, *Med. Imaging, IEEE Trans.* **18**, 231 (1999).
23. J. J. Koenderink and A. J. van Doorn, *Biol. Cybern.* **55**, 367 (1987).
24. L. I. Kuncheva and J. H. V. Constance, *Pattern Recognit. Lett.* **111**, 36 (2018).
25. B. B. Machado, W. N. Gonçalves and O. M. Bruno, Enhancing the texture attribute with partial differential equations: A case of study with gabor filters, in *ACIVS* (Springer, 2011), pp. 337–348.
26. A. Materka, M. Strzelecki *et al.*, *Texture analysis Methods–a Review*, Technical university of lodz, institute of electronics, COST B11 report, Brussels, pp. 9–11, (1998).
27. MIT, Mit vistex texture database, (2011).
28. L. Nanni, S. Brahnam, S. Ghidoni and E. Menegatti, *Expert Syst. Appl.* **42**, 8989 (2015).
29. T. Ojala, M. Pietikäinen and T. Mäenpää, *IEEE Trans. Pattern Anal. Mach. Intell.* **24**, 971 (2002).
30. P. Perona, T. Shiota and J. Malik, Anisotropic diffusion, in *Geometry-Driven Diffusion in Computer Vision* (Springer, 1994), pp. 73–92.
31. N. Petrick, H.-P. Chan, D. Wei, B. Sahiner, M. A. Helvie and D. D. Adler, *Med. Phys.* **23**, 1685 (1996).
32. O. Simon, R. Yacoub, S. Jain, J. E. Tomaszewski and P. Sarder, *Sci. Rep.* **8**, 2032 (2018).
33. X. Tan and B. Triggs, *IEEE Trans. Image Process.* **19**, 1635 (2010).