Pattern Analysis and Applications
https://doi.org/10.1007/s10044-018-0729-9

THEORETICAL ADVANCES

Automatic computing of number of clusters for color image segmentation employing fuzzy *c*-means by extracting chromaticity features of colors

Farid García-Lamont[1] · Jair Cervantes[1] · Asdrúbal López-Chau[2] · Arturo Yee-Rendón[3]

Received: 11 November 2016 / Accepted: 18 July 2018

Abstract
In this paper we introduce a method for color image segmentation by computing automatically the number of clusters the data, pixels, are divided into using fuzzy *c*-means. In several works the number of clusters is defined by the user. In other ones the number of clusters is computed by obtaining the number of dominant colors, which is determined with unsupervised neural networks (NN) trained with the image's colors; the number of dominant colors is defined by the number of the most activated neurons. The drawbacks with this approach are as follows: (1) The NN must be trained every time a new image is given and (2) despite employing different color spaces, the intensity data of colors are used, so the undesired effects of non-uniform illumination may affect computing the number of dominant colors. Our proposal consists in processing the images with an unsupervised NN trained previously with chromaticity samples of different colors; the number of the neurons with the highest activation occurrences defines the number of clusters the image is segmented. By training the NN with chromatic data of colors it can be employed to process any image without training it again, and our approach is, to some extent, robust to non-uniform illumination. We perform experiments with the images of the Berkeley segmentation database, using competitive NN and self-organizing maps; we compute and compare the quantitative evaluation of the segmented images obtained with related works using the probabilistic random index and variation of information metrics.

Keywords Competitive neural networks · Color classification · Image segmentation · Color spaces

1 Introduction

Image segmentation by color features has been employed in different areas such as medicine [1, 2], food analysis [3, 4], among others [5–9]. Segmentation of color images has been addressed using different methods and techniques; several related works treat this issue as a clustering problem because the feature of each pixel corresponds to a pattern and the combination of the pixels, segment, corresponds to a cluster [10, 11]. Several of the previous works based on clustering employ FCM [12–15], unsupervised NN [16–20] or a combination of both techniques [21–23].

FCM has been employed successfully for color image segmentation; however, this technique requires a priori knowledge of the number of clusters the data, pixels, are grouped; usually, the number of clusters is defined by the user [12, 13, 24]. By using unsupervised NN, they are trained to identify specific colors; that is, they are trained only with the colors of the image; when the NN is already trained the pixels of the image are grouped by the NN. But

✉ Farid García-Lamont
fgarcial@uaemex.mx

Jair Cervantes
jcervantesc@uaemex.mx

Asdrúbal López-Chau
alchau@uaemex.mx

Arturo Yee-Rendón
arturo.yee@uas.edu.mx

[1] Centro Universitario UAEM Texcoco, Universidad Autónoma del Estado de México, Av. Jardín Zumpango s/n, Fraccionamiento El Tejocote, CP 56259 Texcoco-Estado de México, Mexico

[2] Centro Universitario UAEM Zumpango, Universidad Autónoma del Estado de México, Camino viejo a Jilotzingo continuación Calle Rayón, CP 55600 Zumpango-Estado de México, Mexico

[3] Facultad de Informática, Universidad Autónoma de Sinaloa, Josefa Ortiz de Domínguez s/n, Cd. Universitaria, CP 80013 Culiacán-Sinaloa, Mexico

Published online: 25 July 2018

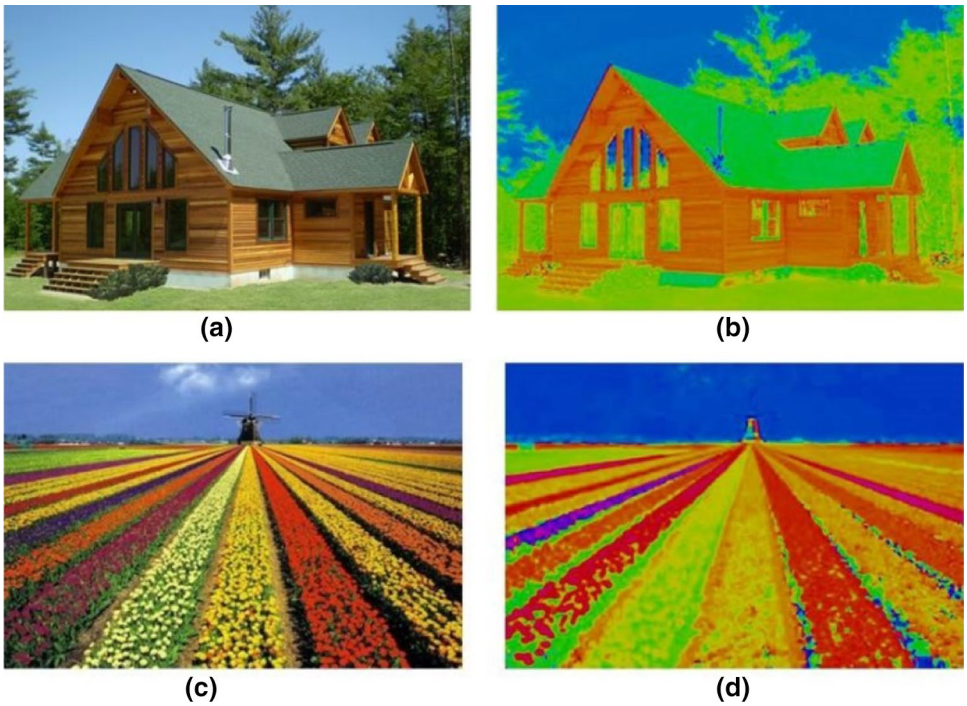

Fig. 1 Original images (**a**) and (**c**); **b** and **d** images obtained after normalizing the intensity of the pixels (color figure online)

using NN involves they must be trained every time a new image is given and it may be time-consuming [16–18].

Because FCM require a priori knowledge of the number of clusters the data are grouped; some works employ unsupervised NN to compute automatically the number of clusters. The proposals that use this approach, essentially, work as follows: A NN is trained with the colors of the image to segment; the image is processed by the NN; in a histogram the activation occurrences of each neuron are collected; and the number of clusters is defined by computing the number of peaks of the histogram [20–23]. The drawback with this approach is, again, the NN must be trained every time a new image is given. On the other hand, though different color spaces are employed, such as HSV, YIQ, L*a*b*, the intensity data of colors are employed; it may be difficult to compute the number of dominant colors for images with non-uniform illumination.

In this paper we present a method to compute automatically the number of clusters the pixels are grouped using FCM by obtaining the number of dominant colors of the image. Our proposal is inspired on the human perception of color; humans can recognize the different sections or parts of an image or scene by the chromatic features of colors, without using intensity data, to some extent, because humans recognize the colors mainly by the chromaticity then by the intensity [25, 26]. For instance, the images (a) and (c) in Fig. 1 are the original images and the corresponding images (b) and (d) are obtained after normalizing the intensity of the pixels. Though all the pixels of images (b) and (d) have the same intensity it is possible to appreciate the different sections of the images by the color features, because the chromaticity is not modified. Thus, in order to obtain a precise segmentation the hue of each segment must be homogenized because, as shown in images (b) and (d) in Fig. 1, within the sections there are different colors with similar hues.

Hence, by employing the chromatic data of the colors within the scene, the number of dominant colors of the image can be obtained; then, the number of clusters the data are grouped is also defined. On the other hand, it is important to remark that humans do not need to learn the colors every time they need to identify a color; they just employ the knowledge acquired previously. So, instead of training a NN every time an image is given to compute the number of dominant colors, we train a NN to recognize different colors by the chromatic features.

Therefore, the contribution of this paper is an approach to compute the number of clusters the pixels of a color image are grouped using FCM. An unsupervised NN is trained with chromaticity samples of different colors; it is important to mention that the NN is trained just once and it can be employed to process any image without training it again every time a new image is given. The image is processed by the NN using the chromatic data of the colors which are extracted by mapping, previously, the image to the HSV color space. The activation occurrences of each neuron of the NN are collected in a histogram; the number of neurons with the highest activation occurrences, peaks of the histogram, defines the number of clusters; and the image is segmented using the FCM with the number of

clusters obtained, where the pixels are represented in the RGB space.

The paper is divided as follows: A review of related works is presented in Sect. 2; in Sect. 3 we introduce our proposed approach; the experiments performed are shown in Sect. 4; in Sect. 5 is performed a discussion; the paper closes with conclusions and future work in Sect. 6.

2 Related works

The segmentation of color images is an issue addressed in different ways, where the cluster-based methods are often employed. Next, we show the papers found in the state of the art on color image segmentation.

Liu et al. [27] propose a segmentation method of mixture models of multivariate Chebyshev orthogonal polynomials. This model is derived from the Fourier analysis, tensor product theory and the nonparametric mixture models of multivariate orthogonal polynomials. The mean integrated squared error is used to estimate the smoothing parameter for every model. The estimation of the number of density mixture components is solved employing the stochastic nonparametric expectation–maximization algorithm, so as to compute the orthogonal polynomial coefficient and weight of each model.

Sag and Cunkas [28] introduce a multiobjective optimization algorithm; the segmentation is addressed as a clustering problem by grouping the image features, where the multiobjective optimization algorithm is combined with seeded region growing. The main features of an image are color, texture and gradient magnitudes, which are measured by using the local homogeneity, Gabor filter and color spaces. The seeded region growing employs the extracted feature vector to classify the pixels spatially. The optimization algorithm determines the coordinates of the seed points and similarity difference of each region by optimizing a set of cluster validity indices so as to improve the quality of segmentation. The segmentation is completed by merging small and similar regions.

Ong et al. [16] present for color image segmentation a two-stage hierarchical NN based on SOMs. The first stage of the network uses a two-dimensional feature map which captures the dominant colors of an image. The second stage employs a one-dimensional feature map to control the number of color clusters that is used for segmentation.

Salah et al. [29] propose a multiregion graph cut image partitioning via kernel mapping of the image data. The data of the image are transformed by the kernel function, so that the piecewise constant model of the graph cut becomes applicable; an objective function contains an original data term to evaluate the deviation of the transformed data, within each segmentation region, from the piecewise constant model. A common kernel function is employed; the energy minimization consists in iterating image partitioning by graph cut iterations.

Mújica-Vargas et al. [12] present two improved FCM clustering algorithms with spatial constraints for color image segmentation. In order to obtain spatial data of the pixels the rank M-type and L-estimators are used. With these estimators the local data of every color component in the RGB model are incorporated; the proposed approach is applied in the chromatic subspace in the IJK color space in order to overcome some limitations related to RGB model. Such estimators are involved in the FCM algorithm to provide robustness for the proposed segmentation techniques.

Tan and Isa [30] present an approach based on histogram thresholding; this approach can be applied in pattern recognition, particularly for color image segmentation. The approach employs the histogram thresholding technique to obtain all possible uniform regions in the color image. The compactness of the clusters forming the uniform regions is improved with FCM.

Huang et al. [31] propose a fuzzy inference system designed by neuro-adaptive learning techniques; from a given image, the proposed system can reveal the probability of being a special color for each pixel through the image. The intensity of every pixel shows this probability in the gray-level output image. After selecting a threshold value, a binary image is computed, which can be used as a mask to segment desired color in the input image.

Guo and Sengur [14] apply neutrosophic set which studies the origin, nature and scope of neutralities. A directional α-mean operation is proposed to reduce the set indeterminacy; the FCM algorithm is improved by integrating with neutrosophic set and employed to segment the color image. The membership computation and the clustering termination are redefined accordingly.

Khan and Jaffar [21] addressed the segmentation of color images as a clustering problem and a fixed length genetic algorithm. An objective function is proposed to evaluate the quality of the segmentation and the fitness of a chromosome. A self-organizing map is used to determine the number of segments in order to set the length of a chromosome automatically. The initialization of the population is performed with an opposition-based strategy.

Khan et al. [23] apply a spatial fuzzy genetic algorithm for segmentation of color images; the performance of the algorithm is influenced by the number of clusters and the initialization of the cluster centers. These factors are overcome using a progressive technique based on self-organizing maps to find the optimal number of clusters automatically. The cluster centers are set with the weights of the neurons represented in the histogram peaks.

Yang et al. [32] address the segmentation of color images as a problem of clustering texture features as multivariate

mixed data. The distribution of the texture features is modeled using a mixture of Gaussian distribution. The mixture distribution is segmented with an agglomerative clustering algorithm derived from a lossy data compression approach; the algorithm employs either 2D texture filter banks or simple fixed-size windows to obtain texture features.

Nock and Nielsen [33] present an approach for image segmentation by region merging following a particular order in the choice of regions. The blend of algorithmics and statistics limits the segmentation error from both the qualitative and quantitative standpoints. The approach is approximated in linear time and space, leading to fast segmentation.

Wang and Dong [11] propose the multilevel low-rank approximation-based spectral clustering method to segment high-resolution images. The proposed method is a graph-theoretic approach, which finds natural groups in a given data set. It approximates the multilevel low-rank matrix, the approximations to the affinity matrix and its subspace, as well as those for the Laplacian matrix and the Laplacian subspace, gains computational spacial efficiency.

Mignotte [34] introduces a segmentation approach based on a Markov random field fusion model which combines several segmentation results associated with simple clustering methods. The fusion model is based on the probabilistic rand measure for comparing one segmentation result to one or more manual segmentations of the same image. This nonparametric measure allows to derive an appealing fusion model of label fields expressed as a Gibbs distribution. This Gibbs energy model encodes the binary constraints set given by the segmentation results to be fused.

Rashedi and Nezamabadi-por [35] propose an algorithm based on the theory of gravity called "stochastic feature-based gravitational image segmentation." The proposed algorithm employs color, texture and spatial data to partition the image. The algorithm is equipped with an operator called "escape" that is inspired by the concept of escape velocity in physics. A stochastic characteristic is incorporated into the algorithm which gives it the ability to search the image for finding the fittest pixels that are suitable for merging.

Mignotte [36] estimates a segmentation map into regions from a boundary representation. The author defines a non-stationary model, MRF model, with long-range pairwise interactions whose potentials are estimated from the probability of the presence of an edge at each pair of pixels. The paper shows that an efficient and interesting strategy to complex region-based segmentation models consists in averaging soft contour maps and using the MRF reconstruction model to achieve an accurate segmentation map into regions.

Huang et al. [24] introduce a clustering algorithm which maintains coherence of data in feature space; the algorithm works under the paradigm of clustering-then-labeling. Applied on the L*a*b* color space, the image is segmented by setting each pixel with its corresponding cluster. The algorithm is based on the theory of minimum description length, which is an effective approach to select automatically the parameters for the proposed segmentation method.

Nadernejad and Sharifzadeh [13] propose an algorithm where bilateral filtering is employed as a kernel function to form a pixonal image. The bilateral filtering is a preprocessing step that eliminates unnecessary details of the image and results in a few numbers of pixons. Later, the computed pixonal image is segmented using FCM.

Most of the reviewed works employ cluster-based methods; as mentioned before, the drawback with these methods is that the number of clusters must be defined a priori. Other works use unsupervised NN, but the NN employed are trained every time a novel image is given. That is, a NN trained with the colors of a given image cannot always recognize all the colors of a different image; hence, the NN must be trained with the colors of the new image.

Our proposal consists in training an unsupervised NN with chromaticity samples of different colors; the segmentation of a given image is obtained by processing the colors of such image by the already trained NN; and the number of clusters the image is segmented, using FCM, is obtained by computing the number of the most activated neurons of the NN. The benefits of our proposal are as follows: (1) It is not necessary to define a priori the number of clusters; (2) the way we train the NN can be employed to process any image without training it again; and (3) the color recognition is, to some extent, robust to non-uniform illumination. In the following section we introduce in detail our proposal for image segmentation.

3 Proposed approach

We have stated before that humans recognize colors mainly by the chromatic features and then by the intensity [25, 26]. For instance, if the reader is asked to name the colors of the squares (a) and (b) in Fig. 2, it is almost sure that he/she would answer "green"; note that square (a) is brighter than square (b), but the chromaticity does not change; however, we can state that squares (a) and (b) are the same color but with different intensities. Now, if the reader is asked to name the colors of the squares (c) and (d) in Fig. 2, it is almost sure he/she would answer "red and pink", respectively. In this example, the intensity is the same in both squares; the chromaticity difference between both the squares is small, but we can notice that the colors of squares (c) and (d) are not the same, though both squares have the same intensity.

It is important to remark, on the one hand, humans are able to recognize the different regions within a scene by the chromaticity features, as we claim in Sect. 1. On the other hand, humans do not need to learn the colors every time

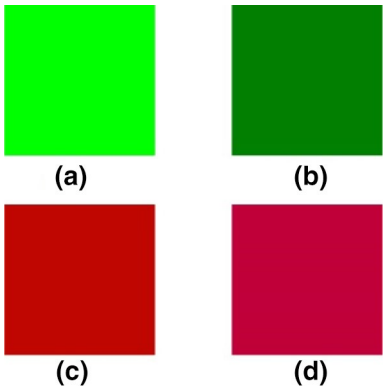

Fig. 2 Color of squares (**a**) and (**b**) with the same chromaticity but with different intensities; color of squares (**c**) and (**d**) with different chromaticities but with the same intensity (color figure online)

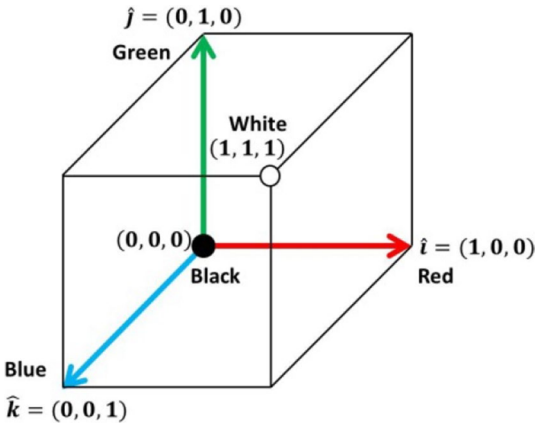

Fig. 3 RGB color space (color figure online)

they need to recognize a given color; they just employ the knowledge acquired previously.

Our proposal consists in emulating the human perception of color, by processing only the chromatic features of colors in order to compute the number of dominant colors within an image and then to segment the image using FCM, where the number of clusters the data are divided is defined by the number of dominant colors. The number of dominant colors is computed using an unsupervised NN trained with chromaticity samples of different colors; the NN processes the chromaticity of each image's pixel; the number of the most often activated neurons defines the number of the dominant colors; therefore, with this number of clusters the data are grouped employing FCM. The steps of our proposal are as follows:

1. Train an unsupervised NN with chromaticity samples of different colors.
2. Map the given image to the HSV space and extract the chromaticity of each pixel's color of the given image.
3. The chromaticity of each pixel is processed by the NN trained previously.
4. Collect in a histogram the activation occurrences of each neuron.
5. Obtain the number of clusters by computing the number of peaks of the histogram.
6. Update the number of clusters by comparing the chromaticity of the neurons with the highest activation occurrences.
7. Segment the image with the number of clusters obtained in step 6 using FCM, where the colors of the pixels are represented in the RGB space.

The number of clusters is updated because there may be neurons with similar chromaticity and occurrence number; that is, the colors these neurons recognize are almost the same, so they belong to the same section and they must be grouped in the same cluster.

3.1 Chromaticity extraction

The RGB color space is based on a Cartesian coordinate system where colors are vectors that extend from the origin, where black is located in the origin and white in the opposite corner to the origin, see Fig. 3.

The color of a pixel p is written as a linear combination of the basis vectors, red, green and blue [25]:

$$\phi_p = r_p\hat{i} + g_p\hat{j} + b_p\hat{k}, \tag{1}$$

where r_p, g_p and b_p are the red, green and blue components, respectively, and the range of every component is $[0, 255] \subset \mathbb{R}$. The orientation and magnitude of a color vector define the chromaticity and the intensity of the color, respectively [25]. This color space faces two problems for color processing: (1) It is sensible to non-uniform illumination and (2) the colors cannot be compared accurately using the Euclidean distance [16, 25].

The HSV (hue, saturation and value) color space has been employed to process color images because the intensity is decoupled from the chromaticity [25, 26]; therefore, the color recognition is more robust before non-uniform illumination. It is also claimed the representation of colors in this spaces emulates the human perception of color [26]. Figure 4 shows the cone shape of the HSV space.

The color of a pixel p is written as [25]:

$$\varphi_p = [h_p, s_p, v_p], \tag{2}$$

where h_p, s_p and v_p are the hue, saturation and value components, respectively. The hue is the chromaticity, the saturation is the distance to the glow axis of black–white, and

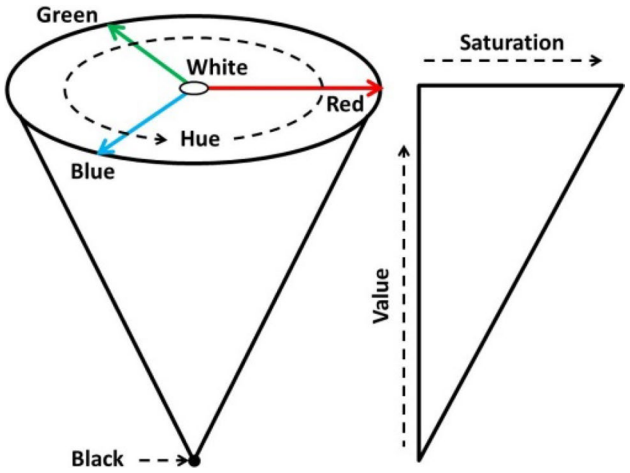

Fig. 4 HSV color space (color figure online)

value is the intensity. The real ranges of hue, saturation and value are $[0, 2\pi]$, $[0, 1]$ and $[0, 255]$, respectively.

The chromaticity is modeled as a vector because of the case when the hue is almost 0 or 2π. Consider squares (c) and (d) shown in Fig. 2; their hue values are $\pi/100$ and $19\pi/10$, respectively. Numerically, both values are very different, but the chromaticities of both squares are similar; if the chromaticity of both squares is classified only by the scalar values, the chromaticity would be recognized as if they were very different. The problem is overcome as follows: Let φ_p be the color of a pixel in the HSV space represented as in Eq. (2), the chromaticity is modeled as follows:

$$\psi_p = \left[\cos h_p, \sin h_p\right]. \tag{3}$$

It is important to mention that the color difference cannot be measured using the Euclidean distance in the HSV space. But with our chromaticity characterization proposal it is possible to employ the Euclidean distance to compute the chromaticity difference between colors, because the chromaticity is represented as unit-length vectors whose orientation defines the chromatic data.

3.2 Neural network architecture and training

The idea of our proposal is that each neuron of the NN is trained to recognize a specific color; when the NN is already trained, each neuron is activated only by the color it learnt to recognize, or a similar one. Because we are interested in computing the number of times the neurons are activated we employ competitive NN (CNN) and self-organizing maps (SOMs). These kinds of unsupervised NN are based on finding the index of the winning neuron before external stimuli; the neurons of the NN are trained such that they compete with

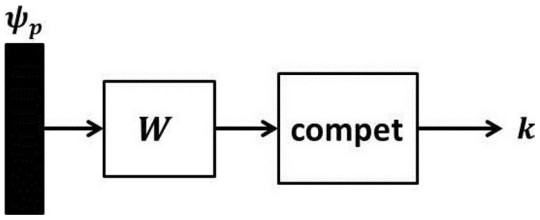

Fig. 5 Architecture of CNN and SOM (color figure online)

each other for the *right* to provide the output associated with an input vector.

The architecture of both the CNN and SOM is shown in Fig. 5, where ψ_p is the external stimuli, $W = \left[w_1, \ldots, w_n\right]^{\mathrm{T}}$ is a matrix whose row vectors are the weight vectors of the neurons, and *compet* is the transfer function that finds the index of the winning neuron.

The Euclidean distance is employed to measure the match between neuron w_k and the external pattern ψ_p. The neuron k whose weight vector w_k is the closest to ψ_p is declared the winner, that is:

$$\|w_k - \psi_p\| = \min_i ||w_i - \psi_p||. \tag{4}$$

For instance, in image (a) of Fig. 6 is shown a NN with 9 neurons and their respective weight vectors. The NN is fed with the vector ψ_p that represents the chromaticity of a color; the neurons are excited, using Eq. (4), by the input vector; the transfer function computes the index of the winning neuron; thus, the output of the NN is the number or index of the winner neuron.

Image (b) of Fig. 6 shows the same NN but already trained, where the color of each neuron is the color that each neuron learnt to recognize. The NN receives the external stimuli represented by the vector ψ_p; in this example, a green-like color is fed, then all the neurons are excited, and the ninth neuron is the winner neuron because the color of neuron 9 is the most similar to the color represented by ψ_p. In other words, $\|w_9 - \psi_p\| < \|w_i - \psi_p\|, \ \forall i = 1, \ldots, 8$.

The difference between CNN and SOM lies in the training and the neuron array. In CNN, only the weight vector of the winning neuron is updated, while in SOM, where the neurons are set in a specific array, the weight vector of the winning neuron and the weight vectors of the neighbor neurons are also updated. A key advantage of SOM over CNN is topology preservation of data. The weight vectors of the neurons are updated with the Kohonen learning rule [37], see Eq. (5).

$$w_k^* = (1 - \alpha)w_k + \alpha\psi_p, \tag{5}$$

where w_k is the weight vector of the winning neuron k, ψ_p is the external stimuli, $0 < \alpha < 1$ is the learning rate, and w_k^* is the updated weight vector of the winning neuron k.

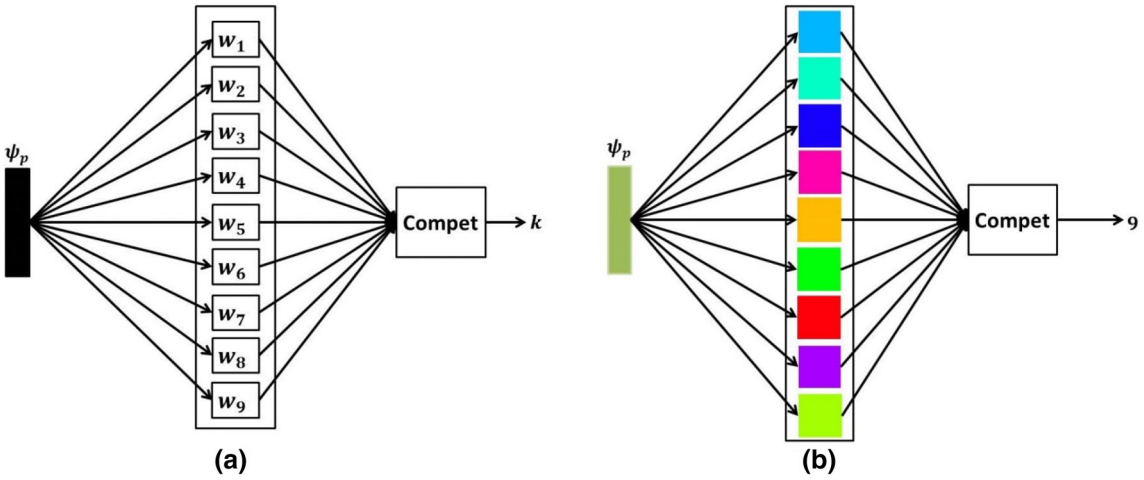

Fig. 6 Example of a NN excited by the external stimuli $\boldsymbol{\psi}_p$ (color figure online)

Because of the fuzzy nature of color, it is not possible to recognize all the colors of the spectrum; hence, the color spectrum is discretized by dividing the spectrum in a finite number of colors. The NN employed for experiments are trained with the elements of the set Ψ built with chromaticity samples as follows:

$$\Psi = \left\{ \psi_k = \left[\cos\theta_k, \sin\theta_k\right] | \theta_k = \frac{\pi}{128} k \,:\, k = 0, 1, \ldots, 255 \right\}. \tag{6}$$

The number of colors the NN can recognize depends on the number of neurons the NN has; in the experimental stage section, we perform experiments with several NN with different sizes in order to determine an adequate size of the NN.

3.3 Computing the number of clusters

Obtaining the number of clusters or sections of the image involves performing the following operations to each pixel of the image:

1. The color vector $\phi = \left[r, g, b\right]$ represented in the RGB space is mapped to the HSV space obtaining the vector $\varphi = [h, s, v]$.
2. Verify if the color of the pixel is black by comparing its intensity with respect to a threshold value. That is, if $v \leq \delta_v$ then the occurrence for black is collected in the histogram and set $\phi = [0, 0, 0]$; go to step 6.
3. If the color is not black then it is verified if the color of the pixel is white by comparing its saturation with respect to a threshold value. In other words, if $s \leq \delta_s$ then the occurrence for white is collected in the histogram and set $\phi = [255, 255, 255]$; go to step 6.
4. If $v > \delta_v$ and $s > \delta_s$ then the color is a chromaticity and modeled as Eq. (3).
5. The vector ψ is computed and processed by the NN; the index of the winning neuron is collected in the histogram.
6. End.

Here δ_s and δ_v are the thresholds for saturation and value, respectively. Because the NN are trained with chromaticity samples, they cannot recognize either white or black because both colors are not chromaticities. Black can be regarded as a low intensity color, that is, when $v \approx 0$; analogously, white can be regarded as a low saturated color, i.e., when $s \approx 0$.

There are no specific values to decide exactly when a color is black or white. Experimentally, we found suitable values for thresholds can be obtained with $\delta_s = \mu_s - \sigma_s$ and $\delta_v = \mu_v - \sigma_v$, where μ_s and μ_v are the mean of the saturation and intensity values of the image, respectively, and σ_s and σ_v are the standard deviation of the saturation and intensity values of the image, respectively.

The number of clusters is defined by the number of bins of the histogram greater than zero. The number of bins of the histogram is equal to the number of neurons plus two; the last two bins correspond to the occurrences of white and black colors. In the histogram the activation occurrences of the neurons which are activated a few times are also collected; it implies that there are irrelevant small parts within the image which are considered they form important parts of the image.

Therefore, there must be selected just the bins whose number is large enough to represent a significant section within the image. Thus, let P be the set of indexes of bins, of the normalized histogram, whose number is greater than or equal to the threshold δ_H:

$$P = \{k \in \mathbb{N} | H(k) \geq \delta_H\}, \tag{7}$$

where $H(k)$ is the value of the normalized histogram at bin k. Therefore, the total of clusters c is the number of elements of the set P; in other words:

$$c = |P|. \tag{8}$$

3.4 Updating the number of clusters

As stated before, there may be neurons with almost the same activation occurrences and with similar chromaticity. Thus, there is a section within the image where the pixels have a common color that activates two neurons several times; therefore, such section of the image is segmented into two parts.

Because the colors are almost the same it means they must be grouped in the same cluster. Thus, the colors of the neurons with the highest activation occurrences are compared by computing the orientation difference between the weight vectors of the neurons as follows:

$$\Delta\theta_{i,j} = \cos^{-1}\left(\frac{\mathbf{w}_i \cdot \mathbf{w}_j^{\mathrm{T}}}{\|\mathbf{w}_i\|\|\mathbf{w}_j\|}\right), \quad \forall i,j \in N, i \neq j, \tag{9}$$

where $\Delta\theta_{i,j}$ is the orientation difference between the weight vectors $\mathbf{w}_i$ and $\mathbf{w}_j$ of neurons i and j, respectively; the set $N = P \cap \{k \in \mathbb{N} | 1 \leq k \leq n\}$, where n is the number of neurons of the NN.

If the orientation difference is smaller than the threshold value δ_θ then the number of clusters must be reduced. That is:

$$c_a = \begin{cases} c_a + +, & \Delta\theta_{i,j} \leq \delta_\theta \\ c_a, & \Delta\theta_{i,j} > \delta_\theta, \end{cases} \tag{10}$$

where c_a is the number of similar colors. The final number of clusters c_t is:

$$c_t = c - c_a. \tag{11}$$

Finally, the image is segmented using the FCM with c_t clusters. Note that the colors are represented in the HSV space so as to compute the number of clusters, while for segmentation, using the FCM, the colors are represented in the RGB space.

3.5 Fuzzy *c*-means clustering algorithm

FCM is a data clustering algorithm in which each data point belongs to a cluster to a degree specified by a membership grade. This gives the flexibility to express that data points can belong to more than one cluster [38]. Let $\Theta = \{\theta_1, \ldots, \theta_n\}$ be the set of feature data, and let c be the number of clusters. Then $U_f = (u_{i,j})$ is called a fuzzy cluster partition of Θ if $\sum_{i=1}^{n} u_{i,j} > 0$, $\forall j \in \{1, \ldots, c\}$ and $\sum_{j=1}^{c} u_{i,j} = 1$, $\forall i \in \{1, \ldots, n\}$ hold. A fuzzy cluster model of Θ into c clusters is defined to be optimal when it minimizes the following objective function:

$$J_f(\Theta; U_f, c) = \sum_{i=1}^{n}\sum_{j=1}^{c} u_{i,j}^m \|\theta_i - c_j\|^2, \tag{12}$$

where $m > 1$ is a weighting exponent called the fuzzifier and $\|\theta_i - c_j\|^2$ is the square of the Euclidean distance from feature vector θ_i to the center of the class c_j. The objective function J_f is alternately optimized using the parameters $u_{i,j}$ and c_j by setting the derivate of J_f with respect to the parameters equal to zero, considering the established constraint above. The resulting equations for the two iterative steps forming the FCM algorithm are given as follows:

$$u_{i,j} = \frac{\|\theta_i - c_j\|^{-\frac{2}{m-1}}}{\sum_{k=1}^{c} \|\theta_i - c_k\|^{-\frac{2}{m-1}}} \tag{13}$$

$$c_j = \frac{\sum_{i=1}^{n} u_{i,j}^m \theta_i}{\sum_{i=1}^{n} u_{i,j}^m}. \tag{14}$$

4 Experimental stage

We test our approach on images taken from the Berkeley segmentation database[1] (BSD), which contains 300 color images of size 481×321 (or 321×481) pixels. The BSD is becoming the standard benchmark to test and evaluate quantitatively algorithms of color image segmentation [39]; hence, we employ the BSD so as to compare the performance of our proposal with related works.

The number of colors a NN can recognize depends on its number of neurons, as we state in Sect. 3.2. Thus, in Sects. 4.1 and 4.2 experiments using CNN and SOM are performed, respectively, with different sizes; the images are segmented with the FCM ordinary algorithm presented in Sect. 3.5. In order to show the appearances of the images obtained with our approach, we select 20 images from the BSD, see Fig. 7.

The images obtained, after processing the images of Fig. 7, are shown in Sects. 4.1 and 4.2, depending on the kind of NN employed and their size; there is also shown the number of clusters computed for each image of Fig. 7.

4.1 Experiments with competitive neural networks

There are performed experiments with three CNN with different sizes so as to determine an adequate size; the first,

[1] http://www.eecs.berkeley.edu/Research/Projects/CS/vision/bsds/.

Fig. 7 Example images employed for experiments taken from the BSD (color figure online)

second and third CNN with 9, 16 and 25 neurons, respectively. From now, we refer to the 9-, 16- and 25-neuron CNN as CNN9, CNN16 and CNN25, respectively. There is employed the threshold value $\delta_H = 0.001$; the threshold values employed to update the number of clusters are $\delta_\theta = 2\pi/9$, $\delta_\theta = \pi/6$ and $\delta_\theta = \pi/9$, for the CNN9, CNN16 and CNN25, respectively.

Because the more neurons the NN have, the closer the neurons' weighting vectors are, the smaller the threshold δ_θ should be. In other words, the smaller the size of the NN is, the larger the threshold value is. The images obtained, by processing the images of Fig. 7, using the three CNN to compute the number of sections, are shown in Figs. 8, 9 and 10, respectively.

Tables 1, 2 and 3 show the number of clusters c and the final number of clusters c_t computed for each image, using the CNN9, CNN15 and CNN25, respectively.

The appearances of the images obtained using the three CNN are very similar because the number of clusters computed is equal or almost the same using any of the CNN with the cluster number updating mechanism proposed in Sect. 3.4. For instance, in image 35010 there are computed 5 clusters using the CNN9, while there are computed 4 clusters employing the CNN16 and CNN25 although with the

Fig. 8 Images obtained by processing the images of Fig. 7 using CNN9 (color figure online)

CNN16 there are computed 5 clusters, but this number is updated to 4 clusters.

In images 118035, 160067 and 210088 there are computed the same numbers of clusters using the three CNN. The images obtained have the same segments, but not always they keep the hues of the original image; for example, in image 210088 the fish does not keep the red hue, but it is segmented from the other parts of the image.

In image 35010 there are computed 5 clusters with the CNN9, while 4 clusters are computed employing the CNN16 and CNN25; the images obtained are essentially the same, except the one obtained with the CNN9 where some parts of the leaves of the background have two different kinds of green hue. With the CNN9 and CNN25 there are computed 6 clusters for image 67079, while 5 clusters are computed using the CNN16. The appearances of the segmented images are very similar; some parts of the building shown in the image, obtained using the CNN16, have different hues than that obtained employing the other NN.

In image 108073, the appearance of the image obtained with the CNN16 is different with respect to the images obtained using the CNN9 and CNN25, where the reflection of the tiger on the water is slightly different. In image 113044 the horses are not segmented uniformly with a unique hue, but in the three images obtained the horses are segmented with the same colors; the segmentation difference can be appreciated in the background.

In image 374067, the segmentation of the green area corresponding to the grass image obtained using the CNN16 is different to the ones obtained with the other CNN. The images obtained, by processing the image 198023, using the CNN9 and CNN16 are the same; in the image obtained using the CNN25, part of the hair is segmented from the head, while with the other CNN both parts are segmented as if they were the same object.

The segmentation differences between the images obtained, by processing the image 86000, can be appreciated in the right side of the building. In image 241004, the

Fig. 9 Images obtained by processing the images of Fig. 7 using CNN16 (color figure online)

images obtained with the CNN16 and CNN25 have the same segmented parts with the same colors; in the resulting image using the CNN9, the area corresponding to grass is segmented in green but also some parts of the rocks.

There are images where, though different numbers of clusters are computed, the resulting segmented images are almost the same. For instance, the segmented images obtained from image 296059 using the CNN9 and CNN16 are the same because in both there are computed 6 clusters. While in the image obtained with the CNN25 there are computed 8 clusters, in the resulting image the sky is segmented with two kinds of blue, and the segmentation of the elephants' bodies is different from the segmentation obtained with the other CNN.

In the images 232038 and 36046 there are computed 6 clusters using the CNN9 and CNN25, and 5 clusters with the CNN16. The segmented images are almost the same; the appearances of the segmented images obtained are very similar in the respective three images obtained. In image 35070 there are computed 4 clusters employing the CNN16 and CNN25, and 5 clusters with the CNN; the segmentation differences between the obtained images lies in the background. In image 97010 there are computed 8 clusters using the CNN9 and CNN25, and 6 clusters with the CNN16; it is easy to appreciate in the image obtained with the CNN16 the sky is segmented uniformly, while in the other images the same area is segmented in two colors.

In image 157032 there are computed 7, 6 and 8 clusters using the CNN9, CNN16 and CNN25, respectively. The segmentation difference is more notable in the image obtained employing the CNN25, where the areas corresponding to the table and the water are different from the resulting images using the other CNN. In the image 201080 there are computed 6, 5 and 8 clusters employing the CNN9, CNN16 and CNN25, respectively; the segmentation between the images obtained with the CNN9 and CNN16 is very similar. The segmented image using the CNN25 is different from the other resulting

Fig. 10 Images obtained by processing the images of Fig. 7 using CNN25 (color figure online)

Table 1 Number of clusters c and final number of clusters c_t computed by processing the images of Fig. 7 using CNN9

Image	c	c_t	Image	c	c_t
35010	5	5	108073	6	6
374067	8	7	210088	8	7
241004	6	5	232038	7	6
118035	7	7	35070	6	5
157032	8	7	201080	6	6
67079	7	6	113044	6	6
198023	7	6	86000	11	8
296059	6	6	36046	7	6
3063	7	6	97010	9	8
160067	4	4	124084	5	5

Table 2 Number of clusters c and final number of clusters c_t computed by processing the images of Fig. 7 using CNN16

Image	c	c_t	Image	c	c_t
35010	5	4	108073	7	5
374067	8	6	210088	9	7
241004	5	4	232038	7	5
118035	8	7	35070	5	4
157032	8	6	201080	7	5
67079	7	5	113044	7	6
198023	7	6	86000	9	7
296059	7	6	36046	7	5
3063	8	6	97010	8	6
160067	6	4	124084	6	5

images, where though the sky is still segmented into two parts, the shapes are not the same as the segmentation obtained with the other CNN; also, the other difference is the segmentation of the building's roof. In image 124084 there are computed 5 and 6 clusters employing the CNN9 and CNN16, and CNN25, respectively. The segmented images obtained using the CNN9 and CNN16 are almost the same; the segmentation difference between the resulting image with the CNN25 and the other ones lies on the flowers' petals.

Table 3 Number of clusters c and final number of clusters c_t computed by processing the images of Fig. 7 using CNN25

Image	c	c_t	Image	c	c_t
35010	4	4	108073	6	6
374067	7	7	210088	7	7
241004	4	4	232038	6	6
118035	7	7	35070	4	4
157032	8	8	201080	8	8
67079	6	6	113044	7	7
198023	8	8	86000	9	9
296059	8	8	36046	6	6
3063	7	7	97010	8	8
160067	4	4	124084	6	6

4.2 Experiments with self-organizing maps

Analogously to the experiments using the CNN, in this section the experiments are performed with three SOMs with 9, 16 and 25 neurons set in 3×3, 4×4 and 5×5 arrays, respectively. From now, we refer to the 3×3-, 4×4- and 5×5-neuron SOMs as SOM3×3, SOM4×4 and SOM5×5, respectively. The values employed for the parameters δ_H and δ_θ are the same values used for the experiments performed with the CNN. That is: $\delta_\theta = 2\pi/9$, $\delta_\theta = \pi/6$ and $\delta_\theta = \pi/9$ for the SOM3×3, SOM4×4 and SOM5×5, respectively, and $\delta_H = 0.001$.

The images obtained, by processing the images of Fig. 7, using the SOM3×3, SOM4×4 and SOM5×5 to compute the number of sections, are shown in Figs. 11, 12 and 13, respectively. Tables 4, 5 and 6 show the number of clusters c and the final number of clusters c_t computed for each image, using the SOM3×3, SOM4×4 and SOM5×5, respectively.

The segmented images obtained with the three SOMs are very alike because the number of clusters computed is almost the same using any of the three SOMs. Notice that the larger the SOM is, the higher the number of activated neurons is; therefore, the number of clusters computed is large. However, the number of clusters is reduced with the

Fig. 11 Images obtained by processing the images of Fig. 7 using SOM3×3 (color figure online)

Fig. 12 Images obtained by processing the images of Fig. 7 using SOM4×4 (color figure online)

updating mechanism proposed in Sect. 3.4, and the number of clusters obtained is almost the same as the number of clusters computed using the CNN.

Only with the image 201080 there are obtained the same numbers of clusters using the three SOMs. In image 35010 there are computed 4 clusters using the SOM4×4 and SOM5×5, while 5 clusters are computed employing the SOM3×3. The appearances of the segmented images are very similar in the three images, but in the image obtained with the SOM3×3, some parts of the leaves of the background have two kinds of green hue; in the other resulting images the leaves have only one kind of green hue.

Six clusters are computed for the image 67079 using the SOM4×4 and SOM5×5, and 5 clusters are computed employing the SOM3×3. The segmented images are alike; the segmentation difference can be appreciated in the right part of the image obtained with the SOM3×3, where some pixels that correspond to the sky area are segmented with the color of the building.

In image 108073 five clusters are computed using the SOM3×3 and SOM4×4 and 6 clusters using the SOM5×5. The difference in segmentation lies on the reflection of the tiger on the water, which is different in the three images. Five clusters are computed for the image 113044 using the SOM3×3 and SOM5×5, and 4 clusters using the SOM4×4; the horses are segmented in the same fashion in all the resulting images, but the color of the image's background obtained with the SOM4×4 is more homogeneous than that obtained with the other SOM.

Six clusters are computed, for image 374067, using the SOM3×3 and SOM4×4, and 7 clusters are computed with the SOM5×5. The images obtained, using the SOM3×3 and SOM4×4, are the same; in the image obtained with the SOM5×5, the segmentation of the grass is more homogeneous than the other resulting images. In image 198023 seven clusters are computed using the SOM3×3, and 5 clusters using the SOM4×4 and SOM5×5; it is remarkable in the images obtained with

Fig. 13 Images obtained by processing the images of Fig. 7 using SOM5×5 (color figure online)

Table 4 Number of clusters c and final number of clusters c_t computed by processing the images of Fig. 7 using SOM3×3

Image	c	c_t	Image	c	c_t
35010	5	5	108073	6	5
374067	8	6	210088	7	5
241004	5	4	232038	8	6
118035	7	6	35070	5	4
157032	9	7	201080	7	6
67079	6	5	113044	6	5
198023	8	7	86000	9	7
296059	7	6	36046	8	6
3063	8	6	97010	9	7
160067	5	4	124084	5	5

Table 5 Number of clusters c and final number of clusters c_t computed by processing the images of Fig. 7 using SOM4×4

Image	c	c_t	Image	c	c_t
35010	7	4	108073	8	5
374067	11	6	210088	10	5
241004	6	4	232038	10	6
118035	8	5	35070	7	4
157032	11	6	201080	9	6
67079	10	6	113044	7	4
198023	8	5	86000	13	9
296059	8	5	36046	7	5
3063	10	7	97010	12	8
160067	7	4	124084	6	3

the SOM4×4 and SOM5×5 that the blue fringes of the sweater are segmented in gray color. A plausible explanation is, because there are computed 5 clusters, the data are divided into a smaller number of groups, and the center of the blue color group is located in the gray color location within the RGB space.

Table 6 Number of clusters c and final number of clusters c_t computed by processing the images of Fig. 7 using SOM5×5

Image	c	c_t	Image	c	c_t
35010	7	4	108073	10	6
374067	15	7	210088	14	7
241004	8	5	232038	14	7
118035	11	5	35070	9	5
157032	17	6	201080	9	6
67079	11	6	113044	10	5
198023	12	5	86000	19	8
296059	12	7	36046	14	7
3063	13	7	97010	19	7
160067	8	5	124084	9	4

In image 210088 five clusters are computed using the SOM3×3 and SOM4×4 and 7 clusters using the SOM5×5. The images obtained are very similar, despite the difference in the number of clusters; the difference between the segmentations is not easy to appreciate, but the segmentation of the plants' tips of the background is different in the three images. In image 86000 seven, nine and eight clusters are computed using the SOM3×3, SOM4×4 and SOM5×5, respectively; the images obtained are almost the same, the segmentation differences can be appreciated in the right part of the building, in which several parts are segmented in green instead of pink-like hue.

For image 241004, four clusters are computed employing the SOM3×3 and SOM4×4, and 5 clusters using the SOM5×5. The images obtained with the SOM3×3 and SOM4×4 are segmented in the same fashion, but in the image obtained with the SOM5×5, the area corresponding to the grass is segmented in green, while in the other images the same area is segmented with the same color of one of the background's mountains. In image 296059 there are computed 6, 5 and 7 clusters using the SOM3×3, SOM4×4 and SOM5×5, respectively. The segmentation differences can be appreciated in the sky, the elephants and the pastureland; the segmentation of the sky is the same in the image obtained with the SOM4×4 and SOM5×5; the segmentation of the elephants is almost the same using the images obtained with the SOM3×3 and SOM4×4; the segmentation of the pastureland is very alike in the images obtained with the SOM4×4 and SOM5×5.

In image 232038 there are computed 6 clusters using the SOM3×3 and SOM4×4, and seven clusters employing the SOM5×5. The appearance of the images obtained with the SOM3×3 and SOM4×4 is the same, but the segmentation using the SOM5×5 is different, and it can be appreciated that the sky is segmented in two parts, while in the same part of the other resulting images, sky area is segmented as an only one part. In image 36046 there are computed 6, 5 and 7 clusters using the SOM3×3, SOM4×4 and SOM5×5, respectively. In the resulting image obtained using the SOM4×4, the grass area is segmented homogeneously, but the sky area is segmented in two parts; the same happens in the image obtained with the SOM5×5, but the segmentation of the grass area is not homogeneous, it resembles the segmentation, of the same area, of the image obtained with the SOM3×3, but in this image the sky is segmented homogeneously.

For image 118035, five clusters are computed employing the SOM4×4 and SOM5×5, and 6 clusters using the SOM3×3. The segmented images obtained using the SOM4×4 and SOM5×5 are the same; the segmented image using the SOM3×3 is different in the sky area, which is segmented in two parts, while in the other resulting images the same sky area is segmented homogeneously. In image 3063 there are computed 6 clusters using the SOM3×3 and seven clusters employing the SOM4×4 and SOM5×5. The segmented images obtained using the SOM4×4 and SOM5×5 are the same; the resulting image obtained using the SOM3×3 is slightly different, from the other segmented images, because there are some parts of the airplane segmented in black.

In image 35070 there are computed 4 clusters using the SOM3×3 and SOM4×4, and 5 clusters employing the SOM5×5. The images obtained with the SOM3×3 and SOM4×4 are the same, the image obtained using the SOM5×5 is a little different in the background segmentation. In image 97010 there are computed 7 clusters using the SOM3×3 and SOM5×5, and 8 clusters employing the SOM4×4. The segmentation differences between the images obtained are not easy to appreciate; a few differences can be observed in the straw package and the color of the building.

For image 157032, six clusters are computed using the SOM4×4 and SOM5×5, and seven clusters employing the SOM3×3. The appearance of the resulting images using the SOM4×4 and SOM5×5 are the same; the image segmentation obtained with the SOM3×3 is different from the segmentation obtained with the other SOM. In the image obtained with the SOM3×3 the table is segmented in three colors, while in the other images the table is segmented in two colors; the segmentation of the area corresponding to the water is different in the image using the SOM3×3 from the segmentation of the same area using the other SOM.

In image 160067 there are computed 4 clusters using the SOM3×3 and SOM4×4, and 5 clusters employing the SOM5×5. The segmentation difference between the images obtained using the three SOM can be appreciated in the areas corresponding to the water and the ground, not in the animal. Notice that despite there are computed 4 clusters employing the SOM3×3 and SOM4×4 the resulting segmented images are slightly different; a plausible explanation

is because of the initial values of the clusters' centers when FCM are employed.

For image 124084, five, three and four clusters are computed using the SOM3×3, SOM4×4 and SOM5×5, respectively. It is easy to appreciate the segmentation differences of the images obtained for each SOM. In the image obtained with the SOM3×3 the flowers' petals are segmented with some parts in green, and the background's leaves are segmented in two kinds of green hue. In the image obtained with the SOM4×4 the flowers' petals and the background's leaves are segmented homogeneously with their respective colors, but the parts in yellow are not segmented. In the resulting image using the SOM5×5 the flowers' petals and the background's leaves are segmented homogeneously with their respective colors, but several parts in yellow are successfully segmented.

5 Discussion

We have shown that it is possible to compute the number of dominant colors of the images with our proposal, where only the chromatic features of colors are employed. There are three issues we consider important to discuss: (1) comparison of the performance of our proposal before previous works; (2) adequate size of the NN; and (3) segmentation performance of our approach when the RGB color vectors are normalized before they are processed by the FCM. These three issues are addressed in the following subsections.

Regarding the third issue, the results obtained so far are achieved by processing the color vectors represented in the RGB space. As we claim before, in the RGB space the intensity of the colors influences the clustering of the color vectors. Hence, the images of Fig. 7 are processed again with the number of clusters computed with all the NN employed, where the color vectors are normalized before they are processed by the FCM; thus, all the colors have the same intensity and they are grouped by the chromatic features.

5.1 Quantitative evaluation of the neural networks

The evaluation of color image segmentation algorithms has been subjective, but recently different metrics and defined ground truth images have been proposed in order to evaluate quantitatively the performance of segmentation algorithms of color images [10, 39]. The BSD is becoming the standard benchmark to compute the performance of color image segmentation algorithms. As mentioned before, the BSD is a database of natural images; for each of these images, the database provides between 4 and 9 human segmentations in the form of label maps which are employed as benchmark, ground truth images, to test quantitatively the performance of color image segmentation algorithms [39].

Several metrics have been proposed, but apparently there have not been already defined absolute metrics to evaluate the algorithms. We have observed in different papers [11, 12, 24, 34] that the probabilistic rand index (PRI) and variation of information (VOI) are becoming the standard metrics. Thus, we employ these metrics to evaluate our algorithm.

The PRI compares the image obtained from the tested algorithm to a set of manually segmented images. Let $\{I_1, \ldots, I_m\}$ and S be the ground truth set and the segmentation provided by the tested algorithm, respectively. $L_i^{I_k}$ is the label of pixel x_i in the kth manually segmented image, and L_i^S is the label of pixel x_i in the tested segmentation. The PRI is computed with [12]:

$$\mathrm{PRI}(S, I_k) = \frac{2}{n(n-1)} \sum_{i,j,i<j} \left(p_{i,j}^{c_{i,j}} \left(1 - p_{i,j}\right)^{1-c_{i,j}} \right), \tag{13}$$

where n is the number of pixels, $c_{i,j}$ is a Boolean function: $c_{i,j} = 1 \quad \text{if} \quad L_i^S = L_j^S \quad \text{and} \quad c_{i,j} = 0 \quad \text{otherwise}$; $p_{i,j} = \sum_{k=1}^{m} T(i,j,k)/m$ is the expected value of the Bernoulli distribution for the pixel pair, where $T(i,j,k) = 1$ if $L_i^{I_k} = L_j^{I_k}$ and $T(i,j,k) = 0$ otherwise. The PRI is in the range $[0, 1]$ where high values indicate a large similarity between the segmented images and the ground truth.

The VOI index measures the sum of loss of information and the gain between two clusters belonging to the lattice of possible partitions in the following way [40]:

$$\mathrm{VOI}(S, I_k) = H(S) + H(I_k) - 2F(S, I_k), \tag{14}$$

where H is the entropy $-\sum_{i=1}^{c} \frac{n_i}{n} \log \frac{n_i}{n}$, with n_i being the number of points belonging to the ith cluster and c being the number of clusters, and F is the mutual information between two clusters defined as:

$$F(S, I_k) = \sum_{i=1}^{c_S} \sum_{j=1}^{c_{I_k}} P\left(S^i, I_k^j\right) \log \frac{P\left(S^i, I_k^j\right)}{P\left(S^i\right) P\left(I_k^j\right)}, \tag{15}$$

where c_S and c_{I_k} are the number of clusters of S and I_k, respectively; $P\left(S^i, I_k^j\right)$ is the joint probability distribution function of clusters i and j of images S and I_k, respectively; and $P\left(S^i\right)$ and $P\left(I_k^j\right)$ are the probability density functions of clusters i and j of images S and I_k, respectively. The range of VOI is $[0, \infty)$, where the smaller the VOI value is, the closer the segmentation obtained and the ground truth are.

All the images of the BSD are processed with our proposal; the segmented images obtained with all the NN are evaluated with metrics PRI and VOI. Table 1 shows the average values of the quantitative evaluation of the segmented images.

Table 7 Average quantitative evaluation of the segmented images, by processing all the images of the BSD using our proposal

NN	PRI	VOI
CNN9	0.7221	2.2910
CNN16	0.7221	2.3283
CNN25	0.7220	2.3128
SOM3×3	0.7169	2.2307
SOM4×4	**0.7258**	**2.0968**
SOM5×5	0.7224	2.2655

Bold represents the proposed methods that obtained the highest scores

Table 8 Comparison of quantitative evaluation of values reported in related works

References	VOI	PRI
Mújica-Vargas et al. [12]	**0.8730**	0.8640
Guo and Sengur [14]	–	0.7720
Ilea and Whelan [20]	–	0.8000
Khan and Jaffar [21]	1.9239	0.8332
Khan et al. [23]	0.9219	**0.8961**
Liu et al. [27]	2.2730	0.7390
Salah et al. [29]	2.4091	0.7650
Tan and Isa [30]	2.2500	0.7280
Huang et al. [31]	1.9510	0.7850
Yang et al. [32]	2.2035	0.7627
Nock and Nielsen [33]	2.0551	0.7681
Mignotte and Helou [41]	2.0100	0.8000
Mignotte [42]	1.8800	0.8100
Our proposal: SOM4×4	*2.0968*	*0.7258*

Bold represents the proposed methods that obtained the highest scores

Italic represents the highest values obtained with our proposal

As given in Table 7, the average values of the segmentation evaluation are similar between all the NN employed; the explanation is that a similar number of clusters are computed with the different NN we employ. The highest value is obtained with the SOM4×4.

Table 8 shows the average values of quantitative evaluations of the segmented images using the proposals of previous works, where we compare the highest performance obtained with our approach, using the SOM4×4. References [20, 27, 29, 31–33, 41, 42] employ other methods than FCM; however, it lets us compare the performance of our approach with respect to other techniques. We claim our approach is competitive, to some extent, because the quantitative values obtained with our approach are close to the values reported in the related works.

It is important to mention the principal contribution of the references [12, 14, 21, 23, 30] focuses mainly on developing modified versions of the FCM algorithm, but not on determining automatically the number of clusters; in references [12, 14], the number of clusters is set by the user.

In [20] is presented a method to compute the number of clusters; a 4×4-neuron SOM is trained with the colors of the image, where the number of the dominant colors, number of clusters, is defined by computing the number of the most activated neurons, where the color vectors are grouped with the k-means algorithm. As we mentioned before, using the colors of the current image leads the SOM to be trained every time a new image is given; with our approach, the NN we employ are trained just one time, then they can be used to process any given image without training it again. The method proposed in [20], for computing the number of clusters, is also employed in [21–23]; in such references, there are employed different color spaces to represent and obtain the dominant colors, such as L*a*b*, YIQ and HSV, but they include intensity data of colors, so it may lead to obtaining the undesired effects of the RGB space before non-uniform illumination.

For our experiments we use the ordinary FCM algorithm, presented in Sect. 3.5; we consider that if our approach is employed for the FCM algorithms developed in [12, 14, 21, 23, 30], the quantitative evaluation of the segmented images may be improved.

5.2 Size of the NN

The sizes of the NN employed in the experiments are proposed empirically, but an important question is, "Which is the adequate size of the NN in order to recognize the dominant colors of any image?" By setting the size or number of neurons of the NN, it defines the maximum number of colors the NN can recognize and thus also the maximum number of segments the image can be divided.

That is, using only FCM for image segmentation, the pixels are grouped strictly with the number of clusters defined a priori. With our approach the number of clusters varies depending on the number of the most activated neurons; therefore, the number of clusters computed is not always the number of neurons of the NN. For instance, an image with a few colors activates a few neurons of the NN several times, there are activated only the neurons that recognize the colors of the image; thus, the number of clusters is low.

Table 9 Average number of clusters computed by processing all the images of the BSD

NN	Final number of clusters	Total of clusters computed	Clusters updated
CNN9	5.9	6.6	0.7
CNN16	6.3	7.2	0.9
CNN25	6.5	6.5	0
SOM3×3	5.4	6.8	1.4
SOM4×4	5.1	8.7	3.6
SOM5×5	6.1	12	5.9

Table 10 Minimum and maximum of final number of clusters, total of clusters computed and clusters updated

NN	Final number of clusters		Total of clusters computed		Clusters updated	
	Min	Max	Min	Max	Min	Max
CNN9	3	9	3	11	0	3
CNN16	3	8	3	9	0	2
CNN25	3	10	3	10	0	0
SOM3×3	3	8	3	10	0	2
SOM4×4	3	8	3	16	0	9
SOM5×5	3	10	3	24	0	15

Using the CNN, there are activated a few neurons and it is almost not necessary to adjust the number of clusters, while with the SOM there are activated several neurons, but when the approach is proposed for adjusting the number of clusters, the number of clusters is reduced almost to the same values computed using the CNN.

Table 9 shows, by computing all the images of the BSD, the average final number of clusters, the average of total of clusters computed and the average of clusters updated, in the second, third and fourth columns, respectively, for each of the NN employed. The average of the final number of clusters goes from 12 clusters, with the SOM5×5, up to 6.5 clusters with the CNN25; when the number of clusters is updated the average final number of clusters is between 6.5 and 5.1 with the CNN25 and the SOM4×4, respectively. The SOM5×5 has, in average, the highest number of

Fig. 14 Images obtained by processing the images of Fig. 7 with number of clusters shown in Table 1, where the color vectors of the pixels are normalized (color figure online)

neurons activated, but also the highest number of clusters updated; hence, the final number of clusters, average, is similar to that obtained with the other NN.

By rounding the values obtained, the average of the total of clusters computed is between 6 and 12, but the average of final number of clusters is between 5 and 6. It is important to remark that not always 6 or 12 neurons of the NN are the most often excited or activated because, as mentioned in Sect. 3.3, the NN cannot recognize black or white. White and black pixels are counted independently; thus, the number of the most activated neurons may be less than 6 or 12.

Table 10 shows the minimum and maximum values of the final number of clusters, total of clusters computed and clusters updated for all the NN.

The minimum of the total of clusters computed is 3 for all the NN; it means it is required at least a NN with three neurons. The maximum number of clusters computed is different depending on the NN. The total of clusters computed goes from 24 using the SOM5×5 up to 9 clusters employing the CNN16, but the final number of clusters computed goes from 10 with the SOM5×5 and CNN25 up to 8 with the CNN16, SOM3×3 and SOM4×4; the maximum number of clusters updated goes from 15 with the SOM5×5 up to 0 using the CNN25. According to the data shown in Table 5, by employing the CNN the number of clusters updated is low, while using SOM the larger they are, the higher the number of clusters updated.

In the reviewed works there is not mentioned the number of clusters obtained with their proposal or if there is an optimal size for the SOM they use. We have found in references [20–23] the use of a 4×4-neuron SOM to compute the number of dominant colors within the images. In Ref. [16] is employed a two-layered NN: The first layer is a 16×16-neuron SOM and the second layer is a 20×1-neuron SOM, also to compute the dominant colors of the images.

Fig. 15 Images obtained by processing the images of Fig. 7 with number of clusters shown in Table 2, where the color vectors of the pixels are normalized (color figure online)

Mújica-Vargas et al. [12] show the segmented images obtained by processing the BSD images 35010, 35070, 118035, 124084 and 232038 using a FCM-based algorithm, where the numbers of clusters defined by the user for each image are 4, 3, 4, 5 and 5, respectively. Notice that these values are very close to the ones we obtain using our approach.

Considering the data of Tables 9 and 10, the number of clusters reported in the reviewed works, and the quantitative evaluation computed in Sect. 5.1, we propose to employ a 4×4-neuron SOM as adequate size for this NN architecture, but using our proposal for cluster number updating.

5.3 Normalizing the RGB color vectors

The images shown in Sects. 4.1 and 4.2, and the quantitative evaluations are obtained by processing the colors represented in the RGB space. This space is sensitive to non-uniform illumination, as claimed in Sect. 3.1, and the chromatic data of colors can be altered by the intensity of colors. As we state in Introduction section, humans are able to recognize the different parts within a scene by the chromatic features of colors; if only the chromaticity is processed, we obtain robustness to non-uniform illumination, to some extent, and the segmentation can be improved. Hence, we perform experiments using just the chromatic data of the color vectors; the chromaticity extraction is performed by normalizing the color vectors before they are grouped with FCM. In these experiments there are employed the number of clusters computed in Sects. 4.1 and 4.2.

In Sect. 3.1 we mention that in the RGB space the orientation of the vectors characterizes the chromaticity and the intensity by the magnitude. Therefore, the chromatic data are extracted by normalizing the color vectors; that is, let $\{\phi_1, \ldots, \phi_m\} \subset \mathbb{R}^3$ be the set of color vectors represented in the RGB space of a given image, and the vectors are

Fig. 16 Images obtained by processing the images of Fig. 7 with number of clusters shown in Table 3, where the color vectors of the pixels are normalized (color figure online)

normalized with $\tilde{\phi}_i = \phi_i / \|\phi_i\|$. Thus, all the colors have the same intensity and the chromaticity does not change because the orientation of the color vectors is not modified.

In other words, let $\tilde{\phi} = [r_u, g_u, b_u]$ be a normalized vector, the direction cosines of this vector are $\cos\alpha_\phi = r_u/\|\tilde{\phi}\|$, $\cos\beta_\phi = g_u/\|\tilde{\phi}\|$ and $\cos\theta_\phi = b_u/\|\tilde{\phi}\|$, but $\|\tilde{\phi}\| = 1$; therefore, the components of the vector $\tilde{\phi}$ are the cosines of the angles between the vector and the basis vectors. Thus, the orientation of ϕ is implicit in $\tilde{\phi}$. Figures 14, 15, 16, 17, 18 and 19 show the resulting images by segmenting the images of Fig. 7 with the number of clusters shown in Tables 1, 2, 3, 4, 5 and 6, respectively.

The images obtained are similar; the colors of the segmented areas are more homogeneous than the images shown in Sects. 4.1 and 4.2. The images obtained after processing the image 35010, with and without normalizing the color vectors, are almost the same. In the segmentation of the image 67079, the sky is segmented with two different kinds of blue hue; in image 108073, the reflection of the tiger over the water is more homogeneous, but the black lines of the tiger are not segmented. The segmentation of the horses of image 113044 is more homogeneous, also the segmentation of the background.

The hues of the green areas and the wall of the image 374067 are more homogeneous; the segmentation of image 198023 obtained with and without normalizing is almost the same, and more details of the woman's face can be appreciated without normalizing the color vectors. The segmentation of the image 210088 is more notable with vector normalization; the hue of the fish is different from the background, but also the hue of the background's vegetation is more homogeneous than the images obtained without normalizing.

The images obtained, by processing the image 86000, with and without normalizing the vectors, are very similar,

Fig. 17 Images obtained by processing the images of Fig. 7 with number of clusters shown in Table 4, where the color vectors of the pixels are normalized (color figure online)

Fig. 18 Images obtained by processing the images of Fig. 7 with number of clusters shown in Table 5, where the color vectors of the pixels are normalized (color figure online)

but the hue of the right side of the building is more homogeneous than the image obtained without normalizing the color vectors. In image 241004, the segmentation difference between the resulting images with and without vector normalization lies on the background's mountains; without normalization there are segmented two mountains, while with normalization the mountains are segmented as if they were only one.

The hue of the elephants of the image 296059 is more homogeneous; without normalizing the color vectors the elephants have several hues. In image 232038, the sky is segmented almost with the same hue, while without normalization, the same area is segmented with two blue hues. The sky area of image 36046 is segmented with two blue hues without normalization; with vector normalization, the sky is segmented with only one kind of blue hue, but the water area is segmented with the same blue hue of the sky; the green areas are segmented with different green hues.

The church and the sky of image 118035 are segmented homogenously with only one kind of hue; without vector normalization, the same parts of the image are segmented with two kinds of hue, respectively. In image 3063 the sky area is segmented with almost the same blue hue; without vector normalization, the same area is segmented with several kinds of blue hues. In the segmentation of image 35070, with vector normalization, it can be appreciated the insect, the leaf and the background; the hues of both the background and the leaf are more homogeneous than in the segmentation obtained without vector normalization.

With vector normalization, the sky area of image 97010 is more homogeneous, also the straw cumulus. The trees behind the barn are segmented in green hue; the hue of the barn is more homogeneous. In image 157032 the water area is segmented with two kinds of blue hue, while without vector normalization the same area is segmented with three

Fig. 19 Images obtained by processing the images of Fig. 7 with number of clusters shown in Table 6, where the color vectors of the pixels are normalized (color figure online)

kinds of blue hue; the other areas are segmented similarly with or without vector normalization.

The segmentations of image 160067 with or without vector normalization are almost the same. In image 201080 the hues of the sky and the building are more homogenous. The hues of both petals and leafs of the flowers shown in image 124084 are more homogenous than the segmentation obtained without vector normalization.

All the images of the BSD are processed, applying vector normalization, with the same number of clusters computed with our approach. The segmentation of the resulting images is evaluated quantitatively with the metrics PRI and VOI. Table 11 shows the average quantitative evaluation obtained.

The average quantitative evaluation is slightly higher than that obtained without normalizing the color vectors; the best segmentation values are, again, obtained with the SOM4×4. The quantitative evaluation improvement is because the areas of the images are segmented with more homogeneity, as shown in Figs. 14, 15, 16, 17, 18 and 19, than the images obtained without normalizing the color vectors.

6 Conclusions and future work

In this paper we have introduced a proposal to compute the number of clusters the data, pixels, are grouped using fuzzy *c*-means. The number of clusters is computed with an unsupervised neural network trained with chromaticity samples of different colors, where the number of the most activated neurons defines the number of clusters the color vectors of the image are grouped.

We have proposed a method to adjust the number of clusters by comparing the similitude between the chromaticity of the most activated neurons. The quantitative evaluation

Table 11 Average quantitative evaluation of the segmented images, by processing all the images of the BSD using our proposal and normalizing the RGB color vectors

NN	PRI	VOI
CNN9	0.7418	2.07
CNN16	0.7435	1.9641
CNN25	0.7399	2.0928
SOM3×3	0.7438	1.9494
SOM4×4	**0.7480**	**1.8510**
SOM5×5	0.7421	2.0482

Bold represents the proposed methods that obtained the highest scores

of the segmented images obtained with our proposal is, to some extent, close to the values reported in previous works, where the Berkeley segmentation database is employed as benchmark. The segmentation, quantitative evaluation, is improved slightly by normalizing the color vectors before they are grouped with the fuzzy *c*-means algorithm. It is possible to determine the number of colors within an image if chromatic data of colors are employed; also, using just the chromaticity of colors, our approach is robust to non-uniform illumination.

The neural network training we propose is performed just once, and then it can be used in any given image without training it again. According to quantitative evaluation and the average of the most activated neurons, the best performance is obtained using a self-organizing map with 4×4 neurons.

As future work, to implement the current proposal with the fuzzy *c*-means algorithms developed in previous works, we consider the quantitative evaluation of segmentation can be improved. Developing a fuzzy logic-based approach to determine whether the color of a pixel is black or white may be useful.

References

1. Ghoneim DM (2011) Optimizing automated characterization of liver fibrosis histological images by investigating color spaces at different resolutions. Theor Biol Med Model 8:25
2. Harrabi R, Braiek EB (2012) Color image segmentation using multi-level thresholding approach and data fusion tecniques: application in the breast cancer cells images. J Image Video Process 2012:11. https://doi.org/10.1186/1687-5281-2012-11
3. Gökmen V, Sügüt I (2007) A non-contact computer vision based analysis of color in foods. Int J Food Eng 3(5):article 5
4. Lopez JJ, Cobos M, Aguilera E (2011) Computer-based detection and classification of flaws in citrus fruits. Neural Comput Appl 20(7):975–981
5. Lepistö L, Kuntuu I, Visa A (2005) Rock image classification using color features in Gabor space. J Electron Imaging 14(4):1–3
6. Wang F, Man L, Wang B, Xiao Y, Pan W, Lu X (2008) Fuzzy-based algorithm for color recognition of license plates. Pattern Recognit Lett 29(7):1007–1020
7. Rotaru C, Graf T, Zhang J (2008) Color image segmentation in HSI space for automotive applications. J Real-Time Image Process 3(4):311–322
8. Bianconi F, Fernández A, González E, Saetta SA (2013) Performance analysis of colour descriptors for parquet sorting. Expert Syst Appl 40(5):1636–1644
9. Cano Marchal P, Martinez Gila D, Gamez Garcia J, Gomez Ortega J (2013) Expert system based on computer vision estimate the content of impurities in olive oil samples. J Food Eng 119(2):220–228
10. Zhang H, Fritts JE, Goldman SA (2008) Image segmentation evaluation: a survey of unsupervised methods. Comput Vis Image Underst 110(2):260–280
11. Wang L, Dong M (2012) Multi-level low-rank approximation-based spectral clustering for image segmentation. Pattern Recognit. Lett 33(16):2206–2215
12. Mújica-Vargas D, Gallegos-Funes FJ, Rosales-Silva AJ (2013) A fuzzy clustering algorithm with spatial robust estimation constraint for noisy color image segmentation. Pattern Recognit Lett 34(4):400–413
13. Nadernejad E, Sharifzadeh S (2013) A new method for image segmentation based on fuzzy *c*-means algorithm on pixonal images formed by bilateral filtering. Signal Image Video Process 7(5):855–863
14. Guo Y, Sengur A (2013) A novel color image segmentation approach based on neutrosophic set and modified fuzzy *c*-means. Circuits Syst Signal Process 32(4):1699–1723
15. Kim JY (2014) Segmentation of lip region in color images by fuzzy clustering. Int J Control Autom Syst 12(3):652–661
16. Ong S, Yeo N, Lee K, Venkatesh Y, Cao D (2002) Segmentation of color images using a two-stage self-organizing network. Image Vis Comput 20(4):279–289
17. Araujo A, Costa D (2009) Local adaptive receptive field self-organizing map for image color segmentation. Image Vis Comput 27(9):1229–1239
18. Stephanakis IM, Anastassopoulos GC, Iliadis LS (2010) Color segmentation using self-organizing feature maps (SOFMs) defined upon color and spatial image space. In: Artificial neural networks—ICANN 2010, LNCS 6352, Part I, pp 500–510
19. Halder A, Dalmiya S, Sadhu T (2014) Color image segmentation using semi-supervised self-organization feature map. Adv Signal Process Intell Recognit Syst 264:591–598
20. Ilea DE, Whelan PF (2008) CTex—an adaptive unsupervised segmentation algorithm based on color-texture coherence. IEEE Trans Image Process 17(10):1926–1939
21. Khan A, Jaffar MA (2015) Genetic algorithm and self organizing map based fuzzy hybrid intelligent method for color image segmentation. Appl Soft Comput 32:300–310
22. Khan A, Jaffar MA, Choi TA (2013) SOM and fuzzy based color image segmentation. Multimed Tools Appl 64(2):331–344
23. Khan A, Ullah J, Jaffar MA, Choi TA (2014) Color image segmentation: a novel spatial fuzzy genetic algorithm. Signal Image Video Process 8(7):1233–1243
24. Huang R, Sang N, Luo D, Tang Q (2011) Image segmentation via coherent clustering in L*a*b* color space. Pattern Recognit Lett 32(7):891–902
25. Gonzalez RC, Woods RE (2002) Digital image processing, 2nd edn. Prentice Hall, Upper Saddle River
26. Ito S, Yoshioka M, Omatu S, Kita K, Kugo K (2006) An image segmentation method using histograms and the human characteristics of HSI color space for a scene image. Artif Life Robot 10(1):6–10
27. Liu Z, Song YQ, Chen JM, Xie CH, Zhu F (2012) Color image segmentation using nonparametric mixture models with multivariate orthogonal polynomials. Neural Comput Appl 21(4):801–811

28. Sag T, Cunkas M (2015) Color image segmentation based on multiobjective artificial bee colony optimization. Appl Soft Comput 34:389–401
29. Salah MB, Mitiche A, Ayed IB (2011) Multiregion image segmentation by parametric kernel graph cuts. IEEE Trans Image Process 20(2):545–557
30. Tan KS, Isa NAM (2011) Color image segmentation using histogram thresholding—fuzzy *c*-means hybrid approach. Pattern Recognit 44(1):1–15
31. Huang R, Sang N, Lou D, Tang Q (2011) Image segmentation via coherent clustering in L*a*b* color space. Pattern Recognit Lett 32:391–902
32. Yang AY, Wright J, Ma Y, Sastry SS (2008) Unsupervised segmentation of natural images via lossy data compression. Comput Vis Image Underst 110(2):212–225
33. Nock R, Nielsen F (2004) Statistical region merging. IEEE Trans Pattern Anal Mach Intell 26(11):1452–1458
34. Mignotte M (2010) Penalized maximum rand estimator for image segmentation. IEEE Trans Image Process 19(6):1610–1624
35. Rashedi E, Nezamabadi-pour H (2013) A stochastic gravitational approach to feature based color. Eng Appl Artif Intell 26(4):1322–1332
36. Mignotte M (2014) A non-stationary MRF model for image segmentation from a soft boundary map. Pattern Anal Appl 17(1):129–139
37. Kohonen T (1990) The self-organizing map. Proc IEEE 78(9):1464–1480
38. Jang JR, Sun C, Mizutani E (1997) Neuro-fuzzy and soft computing: a computational approach to learning and machine intelligence. Prentice Hall, Upper Saddle River
39. Estrada FJ, Jepson AD (2009) Benchmarking image segmentation algorithms. Int J Comput Vis 85(2):167–181
40. Cover TM, Thomas JA (2006) Elements of information theory. Wiley, New York
41. Mignotte M, Helou C (2014) A precision-recall criterion based consensus model for fusing multiple segmentation. Int J Signal Process Image Process Pattern Recognit 7(3):61–82
42. Mignotte M (2014) A label field fusion model with a variation of information estimator for image segmentation. Inf Fusion 20:7–20