Color Quantization with Magnitude Sensitive Competitive Learning Algorithm

Enrique Pelayo(✉), David Buldain, and Carlos Orrite

Aragon Institute for Engineering Research, University of Zaragoza, Zaragoza, Spain
epelayoc@gmail.com, {buldain,corrite}@unizar.es
http://www.unizar.es

Abstract. In this paper we introduce a competitive neural model called Magnitude Sensitive Competitive Learning (MSCL) for Color-Quantization. The aim is to obtain a codification of the color palette taking into account some specific regions of interest in the image, such as salient area, center of the image, etc. MSCL algorithm allows distributing color vector prototypes in the desired data regions according to a magnitude function. This magnitude function can allocate the codewords (colors of the palette) not only in relation to their frequency but also in response to any other data-dependent magnitude tailored to the specific goal. As we show in five different examples in this paper, MSCL is able to surpass the performance of other standard Color Quantization algorithms.

Keywords: Color · Vector quantization · Competitive learning · Neural networks · Saliency · Binarization

1 Introduction

With the huge development of informatics in the modern society, large amount of scanned documents and images are transmitted and stored daily. Therefore, some kind of image processing to reduce storage and transmission is necessary. This is generally achieved by means of Vector Quantization (VQ) techniques. The idea behind VQ is the selection of a reduced number of prototypes that accurately represent the whole data set. When each data sample is a vector representing the color of a pixel, it is denoted as Color Quantization (CQ). This kind of algorithms are useful in certain applications related to segmentation, compression, and transmission of images.

A subset of VQ algorithms comprises Competitive Learning (CL) methods, where a neural network model is used to find an approach of VQ calculation in an unsupervised way. Their advantage over other VQ algorithms is that CL is simple and easily parallelizable. Well known CL approaches are K-means [13] (including

This work is partially supported by Spanish Grant TIN2010-20177 (MICINN) and FEDER and by the regional government DGA-FSE.

N.T. Nguyen (Ed.): TCCI XVII 2014, LNCS 8790, pp. 212–231, 2014.
DOI: 10.1007/978-3-662-44994-3_11

some of its variants as Weighted K-Means and improvements [5]), Frequency Sensitive Competitive Learning (FSCL) [1], Rival Penalized Controlled Competitive Learning [22], the Self-Organizing Map (SOM) [11], and Neural Gas (NG) [14].

Some of these methods, or their variants, have already been used in CQ and Color Segmentation tasks. Uchiyama and Arbib [20] developed Adaptive Distributing Units (ADU), a CL algorithm used in Color Segmentation that is based on a simple cluster splitting rule. More recently, Celebi [4] demonstrated that it outperforms other common algorithms in a CQ task. Fuzzy C-Means (FCM), is a well-known clustering method in which the allocation of data points to clusters is not hard, and each sample can belong to more than one cluster [3]. Celebi presented a relevant work using NG [6].

SOM has also been used in color related applications: in binarization [16], segmentation [12] and CQ [7–9,15] where author presents FS-SOM a frequency sensitive learning scheme including neighborhood adaptation that achieves similar results to SOM, but less sensitive to the training parameters. One variant of special interest is the neural network Self-Growing and Self-Organized Neural Gas (SGONG) [2], an hybrid algorithm using the GNG mechanism for growing the neural lattice and the SOM leaning adaptation mechanism. Author proved that it is one of the most efficient Color Reduction algorithms, closely followed by SOM and FCM.

Methods based in traditional competitive learning are focused on data density representation to be optimal from the point of view of reducing the Shannon's information entropy for the use of codewords in a transmission task. However it is not always desirable a codebook representation with direct proportion between its codeword density and the data density. For example, in the human vision system, the attention is attracted to visually salient stimuli, and therefore only scene locations sufficiently different from their surroundings are processed in detail. A simple framework to think about how Saliency may be computed in biological brains has been developed over the past three decades [10,19].

In a previous work [18], we introduced a new neural method for unsupervised learning denoted as Magnitude Sensitive Competitive Learning (MSCL), which has the property of distributing the unit centroids following any magnitude calculated from the unit parameters or the input data inside its Voronoi region. This controlled behavior allows to outperform standard Competitive Learning algorithms that only tend to concentrate neurons according to the input data density.

In this work we expand the previous paper [17], where we used a MSCL neural network in a CQ task. We add to the previous saliency examples, a methodology to achieve CQ avoiding dominant colors in the image. As a result, final color palette will enhance interesting areas of the image according to a user-specified magnitude (using different definitions of saliency).

1.1 Problem Formulation

Given an image I of size (x_{max}, y_{max}), we define a data sample $\mathbf{x}(t)$ from the pixel I(x,y) as the color vector of that pixel in the coordinates in the corresponding

color space:

$$\mathbf{x}(t) = Color(I(x,y)), \quad x = 1..x_{max} \quad \text{and} \quad y = 1..y_{max} \tag{1}$$

$$t = 1..N, \quad \text{where} \quad N = x_{max} * y_{max}. \tag{2}$$

Each pixel receives an additional value coming from a magnitude function, $mf(t)$. This function is proposed as to weight each pixel with a value of interest. As higher is the value of this magnitude, more relevant is the pixel. The goal is to train a neural network to get units representing the colors of the interesting pixels in higher detail. To do it, feature vectors $\mathbf{x}(t)$ are fed to the neural network with M units. The prototype of unit i ($i = 1 \ldots M$) is represented by a vector of weights $\mathbf{w}_i(t) = (w_{i1}, w_{i2}, w_{i3})$ in the 3-dimensional color space. The associated value of the magnitude in that unit, $mu_i(t)$, can be calculated from the values of $mf(t)$ at samples in its Voronoi region, or can be introduced to the network as the rest of input data.

Figure 1 shows this process. From one image (tiger example), we get a dataset with the 3-D color coordinates of each pixel. The dataset is presented to the

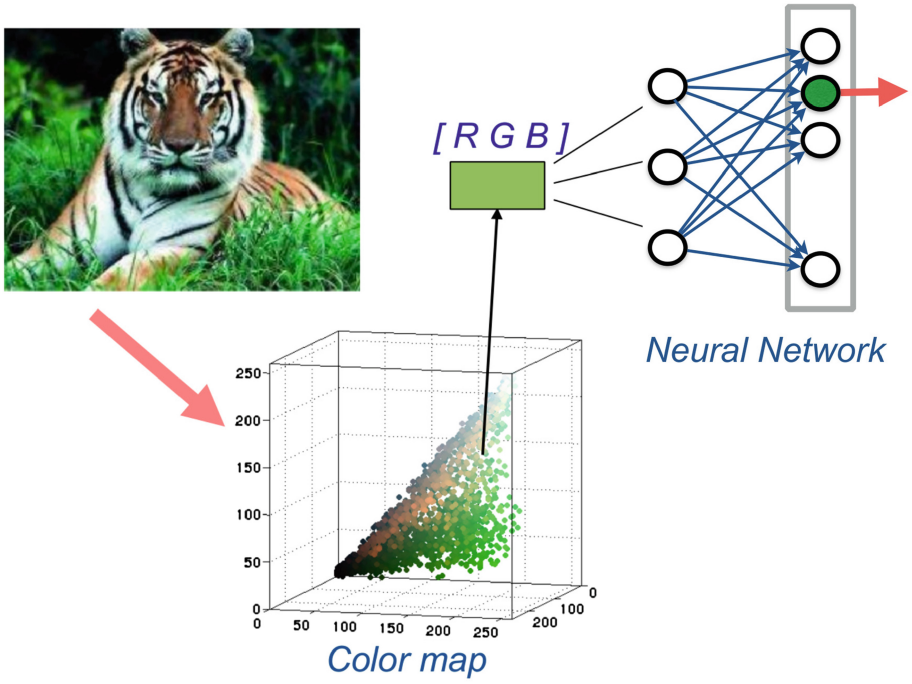

Fig. 1. Problem formulation of Color Quantization: pixels are considered 3-dimensional vectors that are processed as inputs for a competitive neural network with many units as colors in the palette. Magnitude value can be associated to the pixel, as another input to the network, or be associated to the units, as an internal parameter (Color figure onlline).

competitive neural network to generate the color palette with as many colors as units in the network. We paint the color distribution of the image using a [R G B] color space.

1.2 Proposed Approach

We propose the use of MSCL neural network, to train this 3-D dataset, taking into account the magnitude function, that can be defined to lead the training process of the palette to accomplish the desired task. This algorithm follows the general *Competitive Learning* steps:

Step 1. *Selecting the winner prototype.* Given an input data vector, the competitive units compete each other to select the winner neuron comparing their prototypes with the input. This winner unit, also called Best Matching Unit (BMU) is selected in MSCL as the one that minimizes the product of a user-defined Magnitude Function and the distance of the unit prototypes to the input data vector. This differs from other usual competitive algorithms where BMU is determined only by distance. MSCL is implemented by a two-step competition: global and local, as it is explained in next section.

Step 2. *Updating the winner and the unit magnitude.* Both winner's weights and magnitude (if the unit has an associated magnitude) are adjusted iteratively for each training sample, with a learning factor forced to decay with training time.

The idea behind the use of the magnitude term is that, in the case of a sample placed at equal distance from two competing units, the winner will be the unit with lower magnitude value. So, the result of the training process is that units will be forced to move from the data regions with low magnitude values to regions where the magnitude function is higher.

1.3 Paper Description

The remainder of this paper is organized as follows:

Section 2 describes the *Magnitude Sensitive Competitive Learning (MSCL)* method.

Section 3 shows the comparison of the new method with some of the well known algorithms mentioned in the introduction of this article, using five different examples of applications. The first proposed example gets a color quantization that we call homogeneous quantization (Subsect. 3.1). The second example gets a color quantization focused in the image center (Subsect. 3.2). The third example is focused on getting a color palette avoiding the dominant colors usually found in image background (Subsect. 3.3). Fourth example returns a color palette according with a certain image saliency (Subsect. 3.4). Last example shows the use of MSCL in a document image binarization (Subsect. 3.5).

Section 4 concludes with a brief discussion and ideas for future work.

2 The MSCL Algorithm

The next subsections describe the algorithm, which flowchart is shown in Fig. 2.

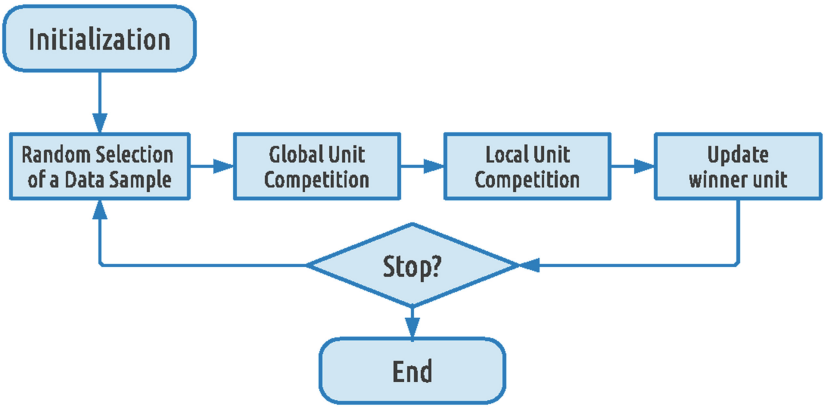

Fig. 2. MSCL flowchart

2.1 Initialization

M unit weights are initialized with data inputs randomly selected from the dataset, and their initial value of its magnitude is equal to the magnitude function at these samples. We also initialize to ones the value of a counter n_i of the number of times that each unit has been the best matching unit.

$$\mathbf{w}_i(1) = \mathbf{x}(1) \tag{3}$$
$$mu_i(1) = mf(1) \tag{4}$$
$$n_i(1) = 1. \tag{5}$$

2.2 Random Selection of Data Samples

A sample data $\mathbf{x}(t) = (x_{t1}, .., x_{td}) \in \Re^d$ (with N samples) is randomly selected at time t from the dataset, and its associated magnitude, $mf(t)$, is calculated. This process will be repeated until every data has been presented to the MSCL neural network. It is important to mention that it is recommended to retrain the neural network with the whole dataset several cycles, along T input data presentations (iterations), to make results independent of data-presentation ordering.

2.3 Global Unit Competition

K units with minimum distance from their weights to the input data vector are selected as winners in this first step. In all the simulations of this work it was used $K = 2$. These units form the S set ($size(S) = K$):

$$S = \{\mathbf{w}_k\} \vee \|\mathbf{x}(t) - \mathbf{w}_k(t)\| < \|\mathbf{x}(t) - \mathbf{w}_i(t)\| \ \ \forall i \notin S. \tag{6}$$

2.4 Local Unit Competition

In the second step, winner unit j is selected from units belonging to S as the one that minimizes the product of its magnitude value with the distance of its weights to input data vector, following:

$$j = argmin(mu_k(t) \cdot \|\mathbf{x}(t) - \mathbf{w}_k(t)\|) \ \ \forall k \in S. \tag{7}$$

2.5 Winner and Magnitude Updating

Only winner's weights, counter and magnitude are adjusted iteratively for each training sample, following:

$$n_i(t+1) = n_i(t) + 1 \tag{8}$$

$$\alpha = \left(\frac{1}{n_i(t+1)}\right)^{\beta} \tag{9}$$

$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \alpha\,(\mathbf{x}(t) - \mathbf{w}_j(t)) \tag{10}$$

$$mu_j(t+1) = mu_j(t) + \alpha\,(mf(t) - mu_j(t)) \tag{11}$$

where α is the learning factor calculated for the winner and forced to decay with iteration time and β is a scalar value between 0 and 1. Using this definition, when β is equal to one, the value of the magnitude at each unit becomes the moving average of the magnitude of the data samples belonging to its Voronoi region.

2.6 Stopping Condition

Training is stopped when a termination condition is reached. It may be the situation when all data samples has been presented to the MSCL neural network along certain number of cycles (if a limited number of samples is used), or the condition of low mean change in unit weights, or any other function that could measure the training stabilization.

2.7 The Magnitude Function

As it has been mentioned before, $mf(t)$ is a function that associates a positive scalar value to the input sample at time t. There are three ways to define this function:

1. As a magnitude map, where each sample has a fixed magnitude value obtained by an user defined function of the data. An example of magnitude map is the one used in the saliency section. The saliency function provides a value for each pixel of the image proportional to its interest. In this case, magnitude is like a data input for modulating the competitive behavior of the unit.

2. A function calculated depending on the unit weights and/or the values of data samples belonging to its Voronoi region. An example of this function is the used for Homogeneous Color Quantization. In that case $mf(t)$ is the value of the mean quantization error of the best matching unit of $\mathbf{x}(t)$.

3 Applications

In the following examples, data samples are 3D vectors corresponding to the RGB components of the image pixels. We have used the RGB space in order to have comparable results to other works, in spite that it is a non-uniform color space (instead of using this one, we could have used other color models as $L*a*b$ whose suitability has been demonstrated for interpreting the real world).

The goal is to get a reduced color palette to represent the colors in the image focused on different objectives. The next five examples show that, adequately selecting the magnitude function, it is possible to get an optimal palette according to the desired application.

3.1 Homogeneous Color Quantization

This example shows the case we call Homogeneous Color Quantization. The mean quantization error (q_{err}) for all samples within the Voronoi region of unit i is used as magnitude function. The q_{err} for sample $\mathbf{x}(t)$ is the distance between $\mathbf{x}(t)$ and the prototype (weights) of its corresponding best matching unit. This magnitude forces the palette colors to be uniformly distributed over the data distribution in the RGB space, independently of its data density, and giving as result Voronoi regions that present similar mean q_{err}.

We use the known Tiger, Lena and Baboon images for performance comparison in CQ tasks (marked in the table as $T*$, $L*$ and $B*$, where * is the number of colors in the palette). Homogeneous MSCL (M-h) and Centered MSCL (M-c, explained in next subsection) are compared against the most successful neural models used in different papers: SOM, FSCL, FCM, FS-SOM, ADU and SGONG. Training process applied learning rates between (0.7-0.01) along three cycles, except in ADU whose algorithm parameters selection follows [20]. The threshold for adding/removing a neuron used in SGONG was (0.1/0.05).

Figure 3 shows the color reduction effects for tiger image with ADU, Homogeneous MSCL and Centered MSCL. The upper part of Table 1 shows the mean of MSE (Mean Squared Error) in 10 trials with different number of palette colors (8, 16 and 32) calculated in all the images. Peak Signal-to-Noise Ratio (PSNR) measure can be easily calculated from MSE value. In general, ADU outperforms all other models, closely followed by SOM and FS-SOM. However, it is clear that ADU (top-right image in Fig. 3) paints the tiger skin with greenish color as an effect of the over-representation of green colors. Both MSCL results (bottom images in Fig. 3) tend to maintain orange colors in the tiger skin, as they are not focused in data density representation. Similar results are obtained in the Lena and Baboon images, as it can been seen in the table.

Fig. 3. Original Tiger image (*top-left*) and its reconstruction using 8 colors applying: ADU (*top-right*), Homogeneous MSCL (*bottom-left*) and Centered MSCL (*bottom-right*) (Color figure onlline).

3.2 CQ Focused on the Image Center

Previous example provides a CQ task giving equal importance to every pixel of the image, and not distinguishing between pixels of the foreground or the background. However the more interesting image regions are usually located in the foreground center. Using MSCL with the adequate magnitude function, it is possible to get a palette with colors mainly adapted to pixels located in the foreground, or any other desired point in the image. In this example we use the following magnitude function:

$$mf(t) = 1 - d(\mathbf{x}(t)) \tag{12}$$

where $d(\mathbf{x}(t))$ is the normalized distance, in the plane of the image (x, y), calculated from the corresponding pixel position to the center of the image. This magnitude function is normalized by the maximum.

We compare the performance of centered MSCL, with the same methods used in previous example. Number of colors and training parameters were also the same.

Prototypes of centered MSCL tend to focus on colors in the central part of the image. Therefore, MSE for the whole image is worse than those obtained using

Table 1. MSE calculated in the whole image and in the image center.

Pixels	*Image*	Som	FSCL	*M-h*	FCM	FSSom	ADU	*M-c*	Sgong
Whole img.	*T8*	987	1016	1037	1005	**985**	990	1095	987
	T16	566	596	577	606	564	**562**	667	570
	T32	334	343	341	357	328.1	**327.8**	409	574
	L8	401	416	424	451	**400.2**	406	406	400.9
	L16	216	234	215	234	216	**214**	217	218
	L32	121	126	122	141	120	**119**	125	222
	B8	1120	1126	1138	1151	**1117**	1126	1227	1121
	B16	633	641	633	693	**632.4**	632.8	751	635
	B32	380	389	380	440	**375.2**	375.9	479	442
Img. center	*T8*	1223	1311	1207	1263	1214	1244	**1151**	1226
	T16	626	710	596	735	631	608	**485**	655
	T32	361	381	356	408	353	355	**283**	407
	L8	445	472	436	552	440	447	**423**	447
	L16	265	294	273	301	262	266	**254**	267
	L32	161	167	160	187	159	159	**149**	163
	B8	1346	1354	1210	1421	1343	1338	**1062**	1321
	B16	708	740	683	833	705	689	**602**	714
	B32	381	412	387	515	372	374	**354**	539

other methods, as background is under-represented. However, when repeating the measures in the central area of the image (150×170 pixels), this algorithm (column *M-c* in the table) outperforms the others (as it can be seen in Table 1 in the three used images), because its color palette models in more detail the central region of the image.

3.3 CQ Avoiding Dominant Colors

Many natural images present few dominant background colors. That means that the majority of the image pixels are represented with these limited set of colors, while other small chunks of the image use a wider palette. By this reason, when it is applied traditional Competitive Learning algorithms on those kind of images for color quantization, color palette usually over-represent these dominant colors, and other secondary colors tend to disappear. In this example we will use MSCL to get a reduced color palette avoiding the color dominance.

To do it we use a two-step method. First we find the dominant colors of the image, and then, we apply MSCL to avoid these dominant colors by defining a magnitude function that gives higher values to the pixels that are more distant from them. We tested this methodology with 4 images (shown in Fig. 4: fish, flower, tower, goat) and compared it with the results of 5 neural models used in different papers: FSCL, FCM, Neural Gas (NG), K-Means and SOM.

Following we describe in detail the two-step method and the experimental results.

Fig. 4. Original images used in the example of MSCL avoiding dominant colors and one example of the corresponding dominant color palettes considering from 1 to 8 colors (*Left to right:* Fish, Flower, Tower, Goat.) (Color figure onlline).

Determination of Dominant Colors. First step is the determination of the dominant colors in the image. One simple way to do it, frequently used in the literature, is using some type of competitive learning algorithm to cluster pixel colors. Weights of units after training the neural network will be the dominant colors. A good candidate for this approach is FSCL method. FSCL can be considered a particular case of MSCL where $mf(t)$ is the number of hits of the best matching unit in sample $\mathbf{x}(t)$. We use this definition to implement FSCL as a data-density sensitive method that can cluster the data colors.

It is possible to use a unique simulation of FSCL to obtain the dominant colors. However, this method is dependent of the goodness of the unit initialization. So, in order to smooth this 'noisy' initialization in the results of the analysis, we use an ensemble of 50 FSCLs for each number of dominant colors (except for the case of one dominant color, calculated as the mean of the image colors). After generating the 50 networks of each ensemble, their prototypes are used to

train the final FSCL to get the 'averaged' dominant colors. We call $\mathbf{pal}_k$ with $k \in \{dominants\}$ to the final dominant-colors palette.

Bottom of Fig. 4 shows the resulting palettes from 1 to 8 dominant colors in the four test images. The evolution of these palettes shows that, in the fish dominant-color palette, orange does not appear until using 5 dominant colors. The flower needs 4 dominant colors to show a good red color, and the tower needs 8 dominant colors to include the red in the roof. The goat dominant-color palette shows that the palette is quite monochromatic.

MSCL Avoiding Dominant Colors. The magnitude function $mf(t)$ used in this example needs to present higher values as the pixel color is more distant from the dominant-color palette. So, for each palette of dominant colors, we define a function ($distcol(t)$) for each pixel as the distance in the color space of that pixel to the closest color in the palette $\mathbf{pal}_j(t)$:

$$j = argmin(\|\mathbf{pal}_k(t) - \mathbf{x}(t)\|) \quad k \in \{dominants\} \tag{13}$$

$$distcol(t) = \|\mathbf{pal}_j(t) - \mathbf{x}(t)\|. \tag{14}$$

Figure 5 shows how this magnitude function works in the case of the fish image using an 8-color palette that avoids two dominant colors. A fraction of the pixels in the color distribution is depicted jointly with the closest regions of the dominant colors (large red circles) and the prototypes generated with MSCL (8 blue circles). MSCL uses three palette-colors for the orange colors of the fish, two colors for the white tones, one stronger black and only two colors dedicated to the background colors with the anemone. One of these colors almost coincide with one of the dominant color (in the center of the graph). This result appears because a large amount of the pixels are in this zone, and MSCL also is forced to move a prototype to this zone to reduce quantization error in (7).

The reconstructed images in this example, with 8-color palettes, for comparing the different methods, appear in Fig. 6, showing from left to right and top to bottom: MSCL avoiding two dominant colors, NG, FSCL, FCM, K-MEANS and SOM. It can be appreciated that MSCL obtained a more vivid color representation for the fish, losing the detail in the anemonae, while other algorithms tend to concentrate the units in the most common colors, showing a lot of greyish tones of the anemone.

Results of Experiments Avoiding Dominant Colors. The main problem of the method is to determine what should be the optimum number of dominant colors. So, we propose to calculate the amount of pixels in High Magnitude Regions (HMR) as a measure of the level of detail that the MSCL method has to deal with. The HMR in the image can be estimated with a threshold of the magnitude function that we chose at the 50 % of the maximum magnitude value. Figure 10 shows this process for the fish example in an 8-color palette. Images from top to bottom in each column correspond to number of dominant colors from 1 to 8. The first column of images in this figure represents the value of $distcol(t)$ in form of image (the magnitude map). The second column shows the

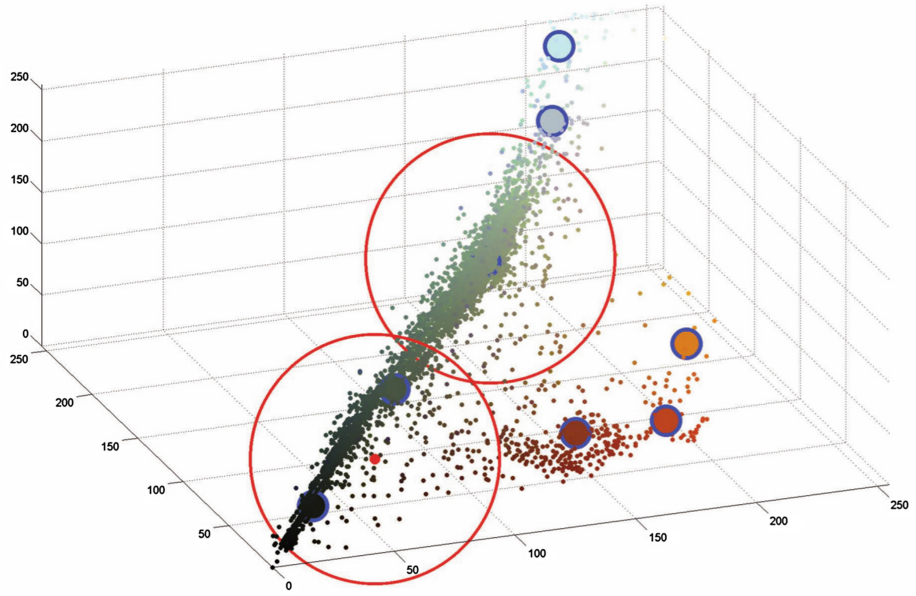

Fig. 5. Representation of a fraction of the pixels in the color distribution for the fish image. The large red circles represent the regions close to the two dominant colors of the image. The 8 blue circles represent the 8-color palette obtained for MSCL avoiding those dominant colors. MSCL uses three palette-colors for the orange colors of the fish, two colors for the white tones, and only three colors dedicated to the background colors (Color figure online).

HMR as the corresponding binarization of the magnitude maps. In this column it is possible to see that, considering only one dominant color, there are still quite a lot of pixels in HMR (corresponding to the fish and the darker areas in the background of the image). However, when two dominant colors are used, HMR extension is quite reduced and corresponds only to certain areas in the fish. Therefore the use of two dominant colors would be a good option for the fish image. The third column of images in this figure shows examples of the MSCL reconstruction for the corresponding number of dominant-colors avoidance.

As comparison, Fig. 11 shows in three columns the resulting HMRs for the other three images. In the first column, the flower image presents an interesting behavior for four dominant colors. In the tower image we have a similar situation, but for three colors (white, blue and dark grey). However, the goat image tends to keep similar HMR extensions. The reason of this behavior is possibly because it is a quite monochromatic image.

In order to visualize the effect of the number of dominant colors, we define the HMR-ratio as the number of pixels in HMR divided by the number of pixels in the image. We generated 50 palettes of dominant colors for each number of colors that varied from 1 to 20. The evolution of the averaged HMR-ratios are shown if the bottom graph of Fig. 9. The curves in the graphs have been smothered.

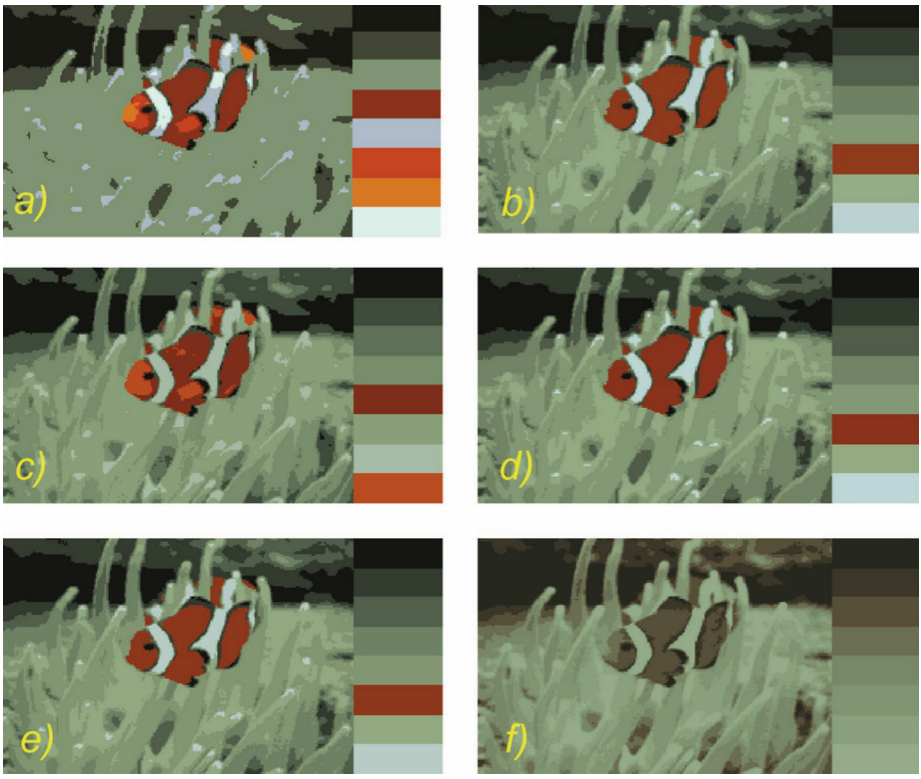

Fig. 6. Results of color quantization for the fish example using an 8-color palette with different methods: *(a)* MSCL avoiding two dominant colors, *(b)* NG, *(c)* FSCL, *(d)* FCM, *(e)* K-MEANS, *(f)* SOM. The corresponding color palettes are shown in the right of each image. As can be appreciated, MSCL gets a more vivid palette for the fish and presents a lower number of colors in the palette dedicated to the background with the anemone (Color figure onlline).

The abscissa shows the different number of dominant colors analysed in the four images. The ordinates show the mean value of the HMR-ratio. A lower value in this ratio means that there are fewer pixels in the image far from the dominant colors. Therefore that palette is a good representative of the dominant colors in the image.

The evolution of HMR-ratio for the fish image shows that there is an abrupt fall in this value from using one dominant color to the use of two. This value tends to keep in similar values until they are used 7 dominant colors. An explanation of this behavior can be visualized in Fig. 10 (image in row 7 and second column) where the dark band in the background is far from any dominant color, which makes the HMR-ratio to grow considerably in the bottom graph of Fig. 9.

It would be possible to detect the optimum number of dominant colors by analysing the HMR-ratio behavior, like detecting relative minimums or

thresholding its variation, however we will left open this possibility, as it is out of the scope of this work.

In order to evaluate the performance of the methods in the HMR, we propose to calculate the Sum Square Error of quantization (SSE) in the HMR, divided by the total SSE in the image, that we will call the SSE-ratio. Graphically this can be appreciated in Fig. 9 (top four graphs corresponding to the four example images). The abscissas in the graphs show several numbers of dominant colors, from 1 to 5, when dealing with generation of 8-color palettes. The different algorithms (FSCL, FCM, NG, K-MEANS, SOM and MSCL) were simulated 50 times to show the averaged SSE-ratio. As it can be seen, MSCL always presents the smallest SSE-ratio, for all the images and different number of dominant colors. That means that MSCL with dominant-color avoidance is able to maintain a reduced amount of error in the HMR, while the others methods tend to concentrate their SSE reduction in the rest of the image.

3.4 CQ Focused in Salient Colors

The aim of salient feature detection is to find distinctive local events in images. Some works [21] exploit the possibilities of color distinctiveness in salient detection. This example shows the MSCL algorithm generating a color palette focused on those salient regions. To achieve that, the chosen magnitude function is the mean computational global saliency (defined as in [21]). The magnitude is normalized by the maximum, and varies from one to values near zero in zones with low saliency (see image in Fig. 7 in the middle of the top row). We used 8 colors with decreasing learning rates between 0.7 and 0.01 for every algorithm.

Figure 7 shows the results. The first two algorithms (SOM, FS-SOM) only obtain a red color and present higher MSE values (SOM: 103.21 and FS-SOM: 103.07) in those pixels belonging to the white mask region of saliency (third top image of Fig. 7). However, using the global saliency (second top image of Fig. 7) as the magnitude for MSCL, the resulting image shows three red variants and the MSE error is lower (87.5). A drawback is that other colors are underrepresented, what means a minor problem if we want to detail the salient regions of the image.

3.5 Image Binarization

Binarization of a text grey-scale image is the process of assigning each pixel of a text image depending of its grey-scale value to one of two classes, one corresponding to the text and the other one to the background. First row of Fig. 8 shows the image of a badly illuminated document (image a), and the results of applying classical binarization algorithms: Otsu method (b), filtering of original image with Laplacian operator (c) and its binarization with Otsu (d). Otsu Method definitely fails to get an adequate binarization because of the dark grey values in the right margin of the paper. Filtering with Laplacian operator provides a better result, because it is an edge extraction mask. However, this method does not fill the letters.

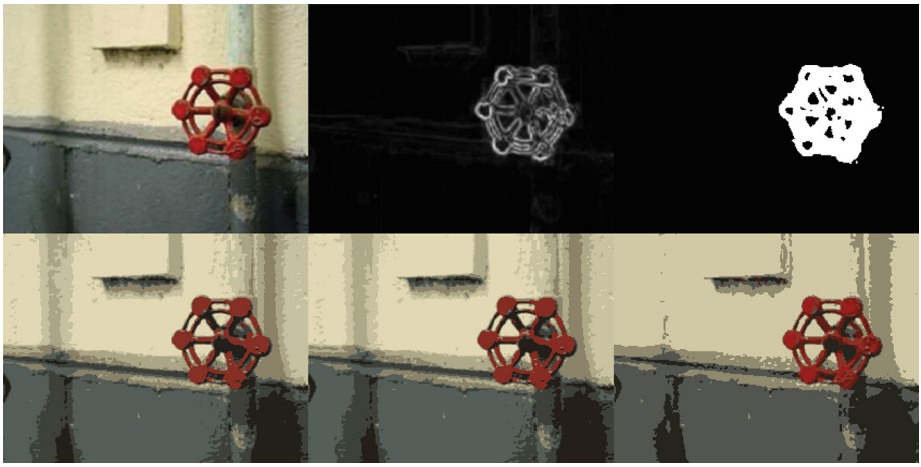

Fig. 7. Saliency example. *Top row, from left to right*: Original image, saliency map (clearer values for high saliency), the mask binary image used for MSE measurement and (*bottom row, from left to right*) the reconstructed image with an 8-colors palette from: SOM, FS-SOM and MSCL focused on the saliency (Color figure onlline).

Competitive learning can be used for this application by training 2 units to represent two levels of gray-scale, which should correspond to the background and foreground classes. Second row of Fig. 8 shows the results with: (e) SOM, (f) MSCL in homogeneous grey quantization, (g) MSCL with two features (explained below), and (h) Otsu binarization of last example. The MSCL

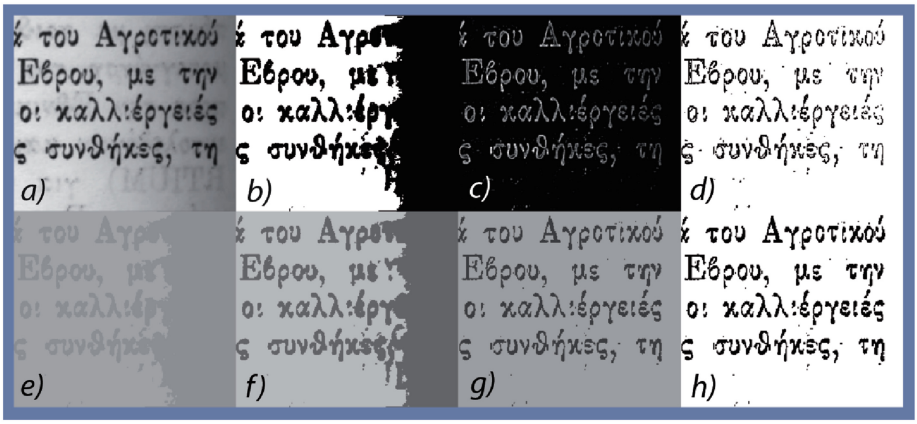

Fig. 8. Binarization example: in *top row* (a) original image, (b) Otsu method, (c) filtering with Laplacian operator, and (d) its binarization with Otsu; in *bottom row* (e) SOM, (f) MSCL in homogeneous grey quantization, (g) MSCL with two features, and (h) Otsu binarization of (g).

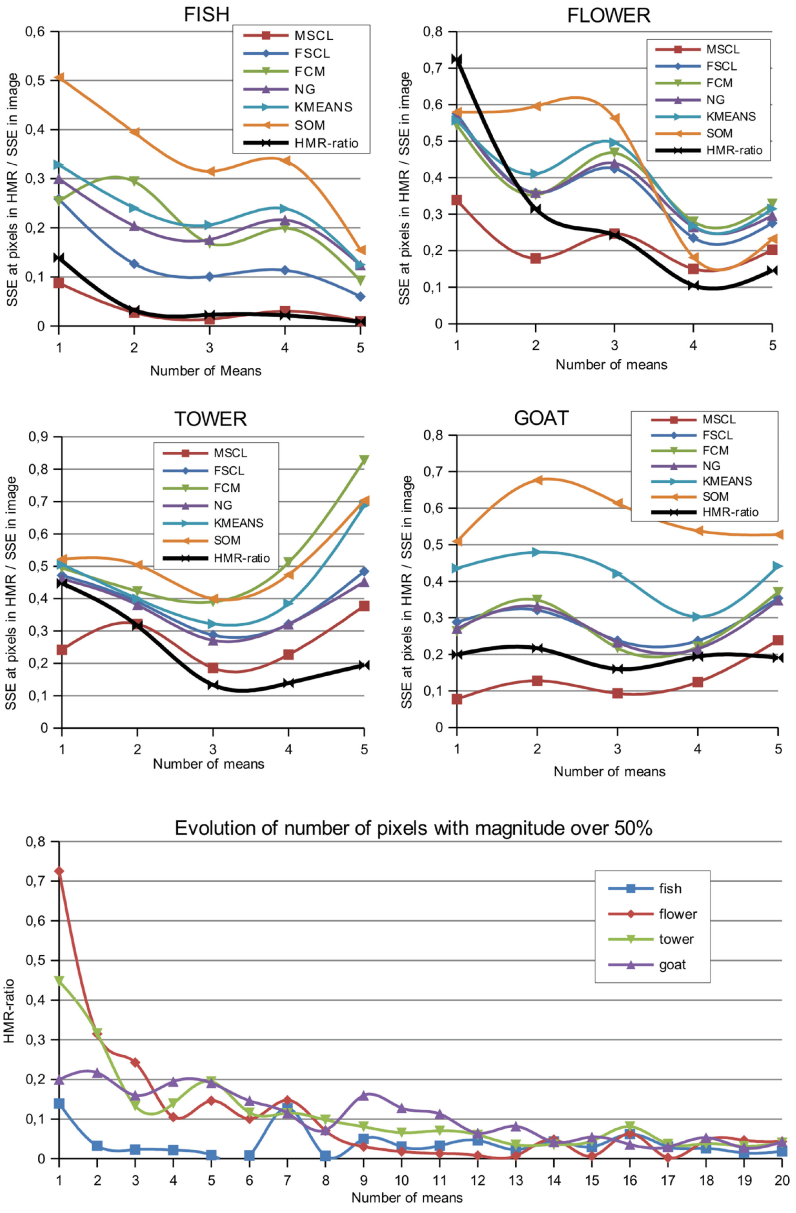

Fig. 9. The top four graphs correspond to each example image, when dealing with generation of 8-color palettes. The averaged Sum Square Error in the High Magnitude Region (HMR), divided by the total SSE in the image (SSE-ratio) is represented for the different algorithms (FSCL, FCM, NG, K-MEANS, SOM and MSCL). The abscissas in the graphs show several numbers of dominant colors, from 1 to 5. MSCL always presents a smaller SSE-ratio, for all the images and different number of dominant colors. The bottom graph represents the evolution of the averaged HMR-ratios (number of pixels in HMR divided by total number of pixels) when using from 1 to 20 dominant colors (Color figure onlline).

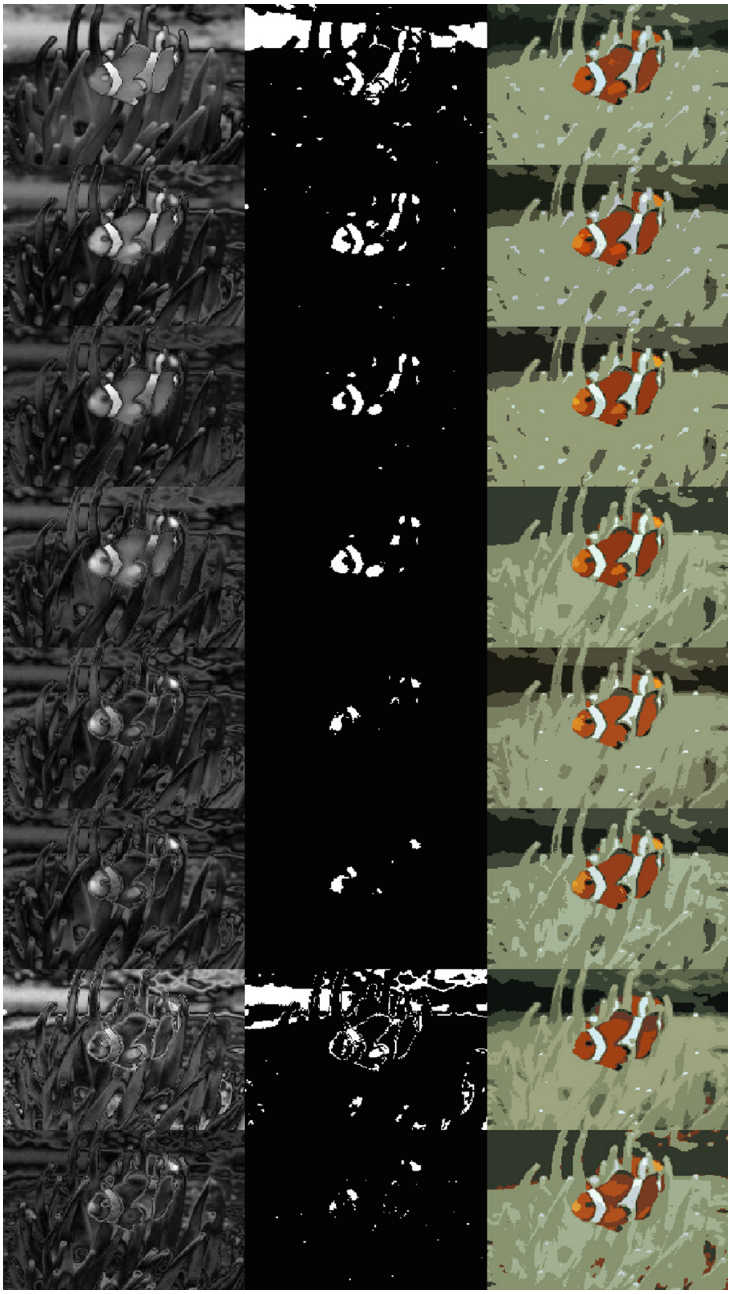

Fig. 10. Results of CQ of the Fish example in a 8 color palette, avoiding different number of dominant colors: (from top to bottom) with 1 to 8 dominant colors. (*In columns*): magnitude map, pixels with magnitude value over 50 % of the maximum (High Magnitude Region), and MSCL reconstruction for the corresponding number of dominant colors (Color figure onlline).

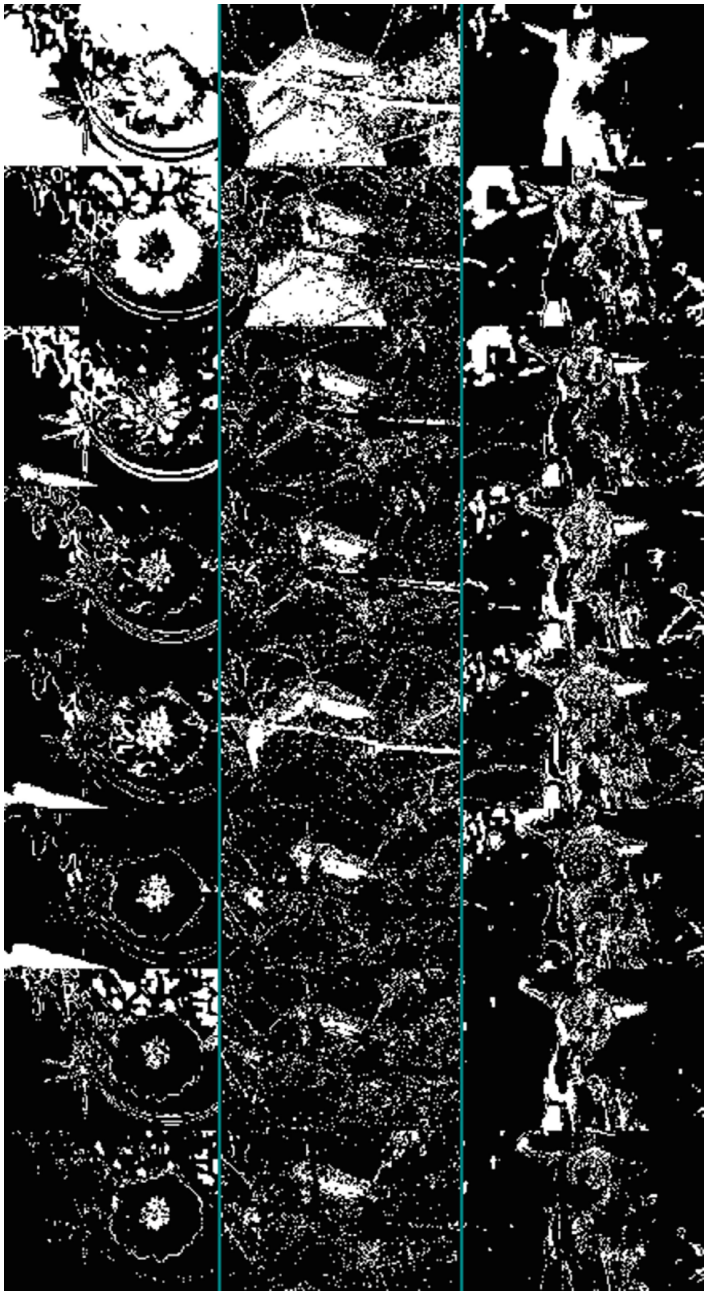

Fig. 11. High Magnitude Regions for different number of dominant colors. Images in rows correspond, from top to bottom, with 1 to 8 dominant colors. Images in the left column correspond to the flower example, the column in the middle for the tower and the right column for the goat image (Color figure onlline).

in (f) with only two neurons is equivalent to the Otsu Method. The reason is that the mean quantization error for each unit is proportional to the standard deviation of a data class when using as mean of the data the unit weights that represents the class.

The quantization result can be improved by using as input a combination of the gray-level values and the result of Laplace filtering. Therefore data samples will be two dimensional vectors combining the values of both features. Then if we apply MSCL using homogeneous quantization to this combined dataset we will get the two-level image (g) in Fig. 8 (the same image with binarized pixel intensity can be seen in next image (h)). This result is better than those achieved by other classical methods.

4 Conclusions

This paper has shown the capabilities of MSCL algorithm for Color Quantization. MSCL is a neural competitive learning algorithm, which includes a magnitude function as a modulation factor of the distance used for the unit competition. As other competitive methods, MSCL accomplishes a vector quantization of the data. However, unlike most of the competitive methods who are oriented to represent in more detail only those zones with higher data-density, the magnitude function in MSCL can address the competitive process to represent any region.

We compare MSCL with other vector quantization approaches in several image color quantization examples for different targets, such as focusing on homogeneous mean quantization error, focusing on image foreground, avoiding dominant colors, focusing on a saliency function and finally, application in text image binarization. As a result of these experiments we have showed that MSCL is more versatile than other competitive learning algorithms focusing only on density representations. MSCL forces the units to distribute their color prototypes following a tailored property, expressed by the appropriate magnitude of the data image. In this way, the palette exhibits more colors to accurately represent certain interesting zones of the image.

As future work we intend to use MSCL for vector quantization in multichannel satellite images. In this application, the number of colors is replaced by the number of channels, thus the dimension of the data increases considerably. By means of a magnitude map obtained from labelled image zones MSCL will allow to orient the vector quantification to the regions of interest such as specific crop areas.

References

1. Ahalt, S., Krishnamurthy, A., Chen, P., Melton, D.: Competitive learning algorithms for vector quantization. Neural Netw. **3**(3), 277–290 (1990)
2. Atsalakis, A., Papamarkos, N.: Color reduction and estimation of the number of dominant colors by using a self-growing and self-organized neural gas. Eng. Appl. Artif. Intell. **19**(7), 769–786 (2006)

3. Bezdek, J.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York (1981)
4. Celebi, M.: An effective color quantization method based on the competitive learning paradigm. In: Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition, IPCV, vol. 2, pp. 876–880 (2009)
5. Celebi, M.: Improving the performance of k-means for color quantization. Image Vision Comput. **29**(4), 260–271 (2011). (Rochester, N.Y.)
6. Celebi, M., Schaefer, G.: Neural gas clustering for color reduction. In: Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition, IPCV, vol. 1, pp. 429–432 (2010)
7. Chang, C., Xu, P., Xiao, R.: New adaptive color quantization method based on self-organizing maps. IEEE Neural Netw. **16**(1), 237–249 (2005)
8. Cheng, G., Yang, J., Wang, K., Wang, X.: Image color reduction based on self-organizing maps and growing self-organizing neural networks. In: 2006 Sixth International Conference on on Hybrid Intelligent Systems (HIS'06) (40572082), 24 Dec 2006 (2006)
9. Dekker, A.: Kohonen neural networks for optimal colour quantization. Netw. Comput. Neural Syst. **3**(5), 351–367 (1994)
10. Itti, L., Koch, C.: Computational modeling of visual attention. Nat. Rev. Neurosci. **2**(3), 194–203 (2001)
11. Kohonen, T.: Self-Organizing Maps. Springer, Heidelberg (2001)
12. Lazaro, J., Arias, J., Martin, J., Zuloaga, A., Cuadrado, C.: SOM segmentation of gray scale images for optical recognition. Pattern Recogn. Lett. **27**(16), 1991–1997 (2006)
13. Lloyd, S.: Least squares quantization in PCM. IEEE Trans. Inf. Theor. **28**(2), 129–137 (1982)
14. Martinetz, T., Berkovich, S., Schulten, K.: 'Neural-gas' network for vector quantization and its application to time-series prediction. IEEE Trans. Neural Netw. **4**(4), 558–569 (1993)
15. Nikolaou, N., Papamarkos, N.: Color reduction for complex document images. Int. J. Imaging Syst. Technol. **19**(1), 14–26 (2009)
16. Papamarkos, N.: A neuro-fuzzy technique for document binarisation. Neural Comput. Appl. **12**(3–4), 190–199 (2003)
17. Pelayo, E., Buldain, D., Orrite, C.: Focused image color quantization using magnitude sensitive competitive learning algorithm. In: IJCCI, pp. 516–521 (2012)
18. Pelayo, E., Buldain, D., Orrite, C.: Magnitude sensitive competitive learning. In: 20th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. vol. 1, pp. 305–310 (2012)
19. Treisman, A., Gelade, G.: A feature integration theory of attention. Cogn. Psychol. **12**, 97–136 (1980)
20. Uchiyama, T., Arbib, M.: Color image segmentation using competitive learning. IEEE Trans. Pattern Anal. Mach. Intell. **16**(12), 1197–1206 (1994)
21. Vazquez, E., Gevers, T., Lucassen, M., van de Weijer, J., Baldrich, R.: Saliency of color image derivatives: a comparison between computational models and human perception. J. Opt. Soc. Am. A: Opt. Image Sci. Vision **27**(3), 21–613 (2010)
22. Xu, L., Krzyzak, A., Oja, E.: Rival penalized competitive learning for clustering analysis, RBF net and curve detection. IEEE Trans Neural Netw. **4**, 636–649 (1993)