Multimed Tools Appl (2013) 64:197–218
DOI 10.1007/s11042-011-0900-4

Classification improvement of local feature vectors over the KNN algorithm

Mahmoud Mejdoub · Chokri Ben Amar

Published online: 29 October 2011

Abstract The KNN classification algorithm is particularly suited to be used when classifying images described by local features. In this paper, we propose a novel image classification approach, based on local descriptors and the KNN algorithm. The proposed scheme is based on a hierarchical categorization tree that uses both supervised and unsupervised classification techniques. The unsupervised one is based on a hierarchical lattice vector quantization algorithm, while the supervised one is based on both feature vectors labelling and supervised feature selection method. The proposed tree improves the effectiveness of local feature vector classification and outperforms the exact KNN algorithm in terms of categorization accuracy.

Keywords Feature vectors · Categorization · Indexing · Wavelets · Lattice vector quantization · Semantic

1 Introduction

Image categorization is still one of the most challenging tasks in computer vision. It consists in labelling an image according to its semantic category. Visual descriptors for image categorization generally consist of either global or local features. The former ones represent global information of images. On the contrary, local descriptors extract information at specific image locations that are relevant to characterize the visual content. Indeed, these techniques are able to emphasize local patterns shared by the images of the same category. Hereafter, we describe some image categorization methods based on local approaches.

M. Mejdoub (✉) · C. Ben Amar
Research Group on Intelligent Machines, University of Sfax, ENIS, BP W - 3038, Sfax, Tunisia
e-mail: mah.mejdoub@gmail.com

C. Ben Amar
e-mail: chokri.benamar@ieee.org

In [22], the images are modelled by calculating symbols from the binary representation of the 3×3 neighborhood pixels around the extracted interest points detected by SIFT [24]. Subsequently, these symbols are ordered taking into account saliency value to obtain string symbols. After that, the HMM (Hidden Markov Model) is learned on these channels of string symbols.

In [23], the authors proposed a categorization method that consists of two steps: the training step and the classification step. The training consists in selecting one prototype per category among a set of labelled images (training set). The classification step assigns to an unlabelled image (query image) the label of its closest prototype. Both training and query images are represented by their sets of Sparse Multiscale Patches (SMPs). The SMP descriptors have been designed in order to exploit local multiscale properties of images.

In [25], each image is represented by a tree that captures a multiscale image segmentation. The trees are matched to find the maximally matching subtrees across the set. The matched sub-trees are fused into a canonical tree which represents the learned model of the category. Recognition of objects in a new image and image segmentation delineating all object parts are achieved simultaneously by finding matches of the model with subtrees of the new image.

In [9], SIFT descriptors are computed at points on a regular grid with spacing M pixels. At each grid point, the descriptors are computed over four circular support patches with different radii; consequently, each point is represented by four SIFT descriptors. Multiple descriptors are computed to allow for scale variation between images. The dense features are vector quantized into V visual words using the K-means clustering. After that, the image spatial layout is used to obtain a pyramid histogram of visual words (PHOW) descriptor. The learning is carried through the multiple kernel SVM classifier.

In [26], the authors used local features based on either salient points or dense sampling. For salient point extraction, the Harris-Laplace point detector is chosen. Color features such as color moments and HSV-SIFT are computed over the area around the points. To construct a fixed-length feature vector, the bag-of-features model with a visual codebook is used. The visual codebook is constructed using the k-means algorithm and the spatial pyramid representation. The learning is carried through the SVM classifier.

The K-nearest neighbor (KNN) [2, 27, 28] classification is an instance-based learning algorithm particularly suited to be used [8, 13, 20, 23] when classifying images described by local features. Given a local query vector and a set of local feature vectors, the KNN algorithm searches the k nearest local feature vectors of the query vector. This includes the steps presented below:

1. Compute all the distances between the query vector and all database vectors.
2. Sort the computed distances.
3. Select the k reference vectors corresponding to the k smallest distances.

In this study, we propose a new categorization tree based on the KNN algorithm. The proposed categorization tree combines both unsupervised and supervised classification of local feature vectors. The unsupervised one is based on a hierarchical lattice [16, 18, 19] vector quantization algorithm, while the supervised one is based on both feature vectors labelling and supervised feature selection method. The labelling ensures the prediction of the semantic category of an unknown feature vector. This is done by (1) selecting the k nearest neighbors of the query vector using the tree and (2) applying a majority vote to determine the class label of the query vector. The feature selection method permits to eliminate the features that at the same time minimize the inter-category variability and maximize the intra-category variability.

The advantage of the proposed tree is that it realizes the trade-off between the accuracy and the speed-up of categorization. In our previous publications [16, 18, 19], we have presented our indexing method and we have showed its effectiveness in the improvement of the image retrieval speed-up. In this paper, we focus on the categorization rate improvement compared to the exact KNN algorithm.

The rest of the paper is organized as follows: Section 2 discusses the related work. Section 3 presents a general description of the proposed categorization method. Section 4 outlines the proposed KNN algorithm based on the regular lattice. Section 5 presents the proposed categorization. Section 6 presents the proposed feature selection methods. Section 7 presents the proposed method to search the proposed categorization tree in order to find the k nearest neighbors of a given local descriptor. Section 8 presents the proposed image categorization method. Some experimental results are presented in the final section in order to illustrate the effectiveness of the proposed categorization method.

2 General description of the proposed framework

In this section, we give a general description of the proposed approach for image categorization. This one includes the following steps:

1. Extract the salient objects from the image.
2. Associate to each salient object a fuzzy descriptor.
3. Organize the fuzzy descriptors extracted from all database images in the categorization tree.
4. Assign, in the categorization tree, a label to each salient object (the label represents the semantic category to which the image containing the salient object belongs).
5. Search the k nearest neighbors of each fuzzy descriptor using the proposed categorization tree.
6. From the semantic labels of the k nearest neighbors, translate the fuzzy descriptor to a high level descriptor called here probability vector. The coefficients of the latter represent the membership degrees of the salient objects as to each category. Thus, each salient object in the image is associated to a probability vector.
7. From the probability vectors describing the salient objects contained in the image, calculate a global probability vector describing the whole image.
8. Assign the image to the category receiving the greatest value in the global probability vector.

3 Feature extraction

In this paper, we focus on local feature vector extraction in order to describe the image prominent objects. Firstly, a robust salient feature detector based on biorthogonal Beta wavelet [1, 3, 15, 17, 19] is designed. The resulting salient points are not confined to corners, but indicate where "something" happens at different scales in the image. Secondly, a salient object is derived from the pixels located in a small neighborhood of 32×32 around each salient point.

3.1 Low level wavelet feature vector extraction

We convert the salient object to the Lab space. After that, we transform each color component to the spatio-frequency domain which is more representative than the spatial domain. To do this, the biorthogonal Beta wavelet is applied with two decomposition levels. Third, we compute in each scale 4 standard deviations associated to the coefficients of the four obtained bands (one approximation band and three detail bands). Thus, we obtain for each color component a low level feature vector with eight coefficients and for the salient object a feature vector x_j with 24 coefficients.

3.2 Fuzzy wavelet feature vector extraction

We divide the database images on a test and learning sets. After that, we group in the matrix $X = \{x_j, j = 1, ..., N\}$ the low level feature vectors x_j extracted from each salient object detected in each image of the learning set, with N denoting the number of feature vectors to be classified. Then, we apply the bag of words technique [5, 11] on these wavelet feature vectors. The competitive agglomeration algorithm (CA) [10] is used to cluster the feature vectors contained in X. The adopted bag of words model translates each low level feature vector x_j to a fuzzy feature vector containing N_{cluster} components. The ith component of the fuzzy feature vector is between 0 and 1 and reflects the membership degree of the low level feature vectors as to the ith visual salient object cluster.

4 KNN algorithm based on the regular lattice

A lattice Λ in $\mathbb{R}^n$ is composed of all integral combination of a set of linearly independent vectors $\mathbf{a}_i$ (the basis of the lattice) such that:

$$\Lambda = \{\mathbf{x} | \mathbf{x} = u_1\mathbf{a}_1 + u_2\mathbf{a}_2 + ...u_n\mathbf{a}_n\} \tag{1}$$

where the u_i are integers. The partition of the space is hence regular and depends only on the chosen basis vectors $\mathbf{a}_i \in \mathbb{R}^m$ $(m \geqslant n)$.

In this work, we use the $\mathbb{Z}^n$ basis (with n the dimension of the feature vectors) due to its simplicity. Indeed, the lattice $\mathbb{Z}^n$ uses the canonical basis and thus the hypercube partitions. The centroid of each partition is called a lattice vector (lattice point). As seen in Fig. 1, a hyper-pyramid of radius r and dimension n is made up of all the lattice vectors p such as $\|p\|_1 = \sum_{i=1}^{n} |p_i| = r$. r is called the norm of p.

The lattice vector quantization algorithm LVQ is an efficient algorithm for finding the closest lattice vector p of a given feature vector v. Indeed, the closest lattice

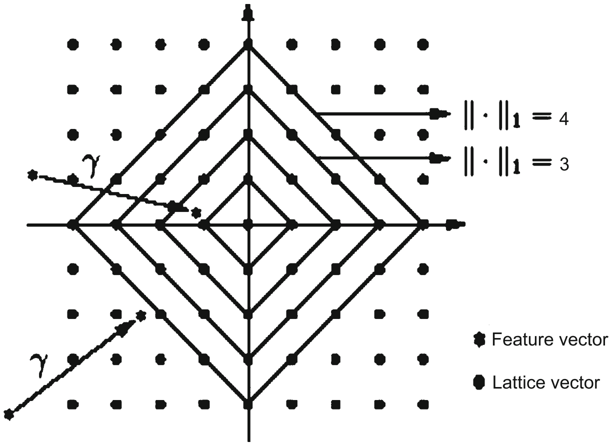

Fig. 1 Example of a $\mathbb{Z}^2$ lattice. The lattice vectors are distributed in concentric hyper-pyramids

vector (lattice point) p of v with a precision γ is given by quantizing v in the lattice point p using the (2).

$$p = \left[\frac{v}{\gamma} \right] \tag{2}$$

where [.] stands for the 'round' operator, and γ is a scaling factor.

Besides, the LVQ ensures the unsupervised classification of the feature vectors. Indeed, the feature vectors quantized in a given lattice point p are grouped into the same lattice partition centred around p.

4.1 Indexing the feature vectors

Firstly the feature vectors are quantized into lattice vectors p. Secondly we attribute a unique index for each p. The goal of the index is to determine rapidly the position of the lattice vector in the lattice. For that, we use our indexing method which was already presented in [16, 18, 19, 21]. It computes this index by classifying the lattice vectors according to their norm and the geometrical properties of the lattice. An index is composed of three indices: an index for the *norm* (I_N), an index for the *leader* ($I_{\mathcal{L}}$), and finally an index for the *permutation* (I_p):

- The index for the norm I_N is given by the l_1 norm $\|p\|_1 = \sum_{i=1}^{n} |p_i| = r$ of the lattice vector p of dimension n. It classifies the different lattice vectors p in different hyper-pyramids (shells). The l_1 norm is chosen because the sum of the used fuzzy feature vector coordinates is a constant and it is equal to 1.
- The lattice vectors lying on the same shell with index I_N are subdivided into a few vectors, called *leaders*. The leaders are vectors from which all the other lattice vectors of the corresponding shell can be generated by permutations and changes of its coordinates. Indeed, the leaders are vectors with positive coordinates sorted in an increasing (or decreasing) order. Therefore, leaders

for the norm r and the feature space dimension n are vectors which verify the conditions below:

1. $\sum_{i=1}^{n} p_i = r$;
2. $0 \leqslant p_i \leqslant p_j$, for all $i < j$.

The leader index $I_{\mathcal{L}}$ is calculated by computing the number of leaders coming before it in the lexicographical order. While the index for the permutation I_P is computed by sorting in the lexicographical order the possible permutations obtained from the leader. For better comprehension, we will give an example that illustrates the computing of the norm, leader and permutation index of the lattice point (1, 3, 1):

– Computing the norm index: $I_N = 1 + 3 + 1 = 5$.
– Computing the leader index: the leader of $(1, 3, 1)$ is $(1, 1, 3)$. We have 3 leaders in the norm $r = 5$ and the dimension $n = 3$. These leaders are given by $(0, 0, 5)$, $(0, 1, 4)$, $(0, 2, 3)$ that come in the lexicographical order before $(1, 1, 3)$; hence the leader index of $(1, 3, 1)$ is $I_{\mathcal{L}} = 3$.
– Computing the permutation index: The possible permutations of the leader $(1, 1, 3)$ are $(1, 1, 3)$ and $(1, 3, 1)$. As $(1, 1, 3)$ comes in the lexicographical order before $(1, 3, 1)$, the permutation index of $(1, 3, 1)$ is $I_p = 1$.

We denote that we presented the details of the computation of the leader index $I_{\mathcal{L}}$ and the permutation index I_p in our previous papers in [16, 18, 19].

This indexing method creates a hierarchical tree with three levels. The first corresponds to the norm index I_N. The second corresponds to the leader index $I_{\mathcal{L}}$. The third corresponds to the permutation index I_p. The index of each lattice point p given by $(I_N, I_{\mathcal{L}}, I_P)$ can be viewed as a branch in this tree. Each branch $(I_N, I_{\mathcal{L}}, I_P)$ is pointed on $C(p)$ and $Cat(p)$ that respectively represent the feature vectors quantized in p and the labels of the salient objects whose feature vectors are quantized in p (see Fig. 2).

Figure 3 presents an example of a regular lattice tree. The feature vectors $C(p_1)$ are indexed in the tree with $(I_N = 9, I_{\mathcal{L}} = 5, I_P = 6)$. Besides, we observe in this figure that p_1 and p_2 have the same norm index $I_N = 9$. As a consequence, the leader indices $I_{\mathcal{L}}$ of p_1 and p_2 are in the same table located in level 2 and pointed by $I_N = 9$. Besides, as the lattice points p_2 and p_3 have the same norm index $I_N = 9$ and the same leader index $I_{\mathcal{L}} = 1$, the permutation indices I_P of p_2 and p_3 are in the same table located in the level 3 and pointed by $(I_N = 9, I_{\mathcal{L}} = 1)$.

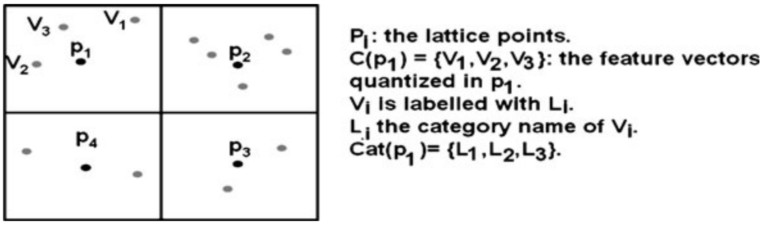
V3
V1
V2
P1
P2
P4
P3
Pi: the lattice points.
C(P1) = {V1,V2,V3}: the feature vectors quantized in P1.
Vi is labelled with Li.
Li the category name of Vi.
Cat(P1)= {L1,L2,L3}.

Fig. 2 Labelling of the feature vectors quantized in the same lattice point

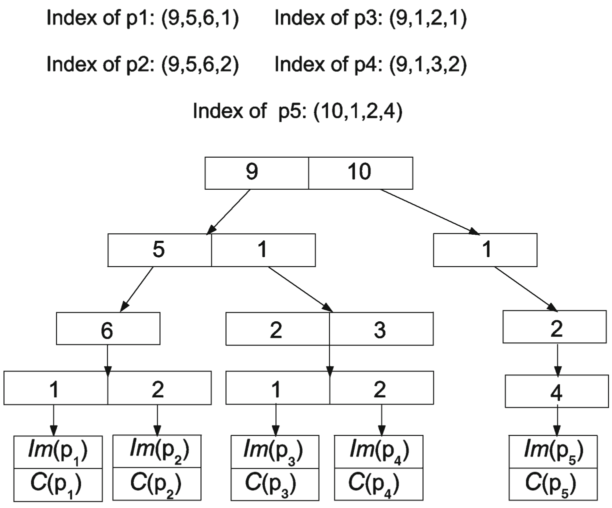

Fig. 3 Example of a regular lattice tree

4.2 Getting the k nearest feature vectors

Let us first define the nearest neighbor lattice vectors p of the origin lattice vector (the null vector $\mathbf{0}$) as:[1]

$$mask_d = \left\{ p \in \Lambda, \sum_i p_i^2 = d, d \in \mathcal{P} \subset \mathbb{N} \right\} \tag{3}$$

which defines the lattice vectors at the square distance d from the origin. For example, as shown in Fig. 4 for a lattice $\mathbb{Z}^2$, the neighbors at a square distance $d = 4$ from the origin of the lattice are given by $mask_4 = \left\{ \binom{0}{-2}, \binom{-2}{0}, \binom{0}{2}, \binom{2}{0} \right\}$. They correspond to the third neighborhood of the null vector $\mathbf{0}$.

The proposed basic procedure to retrieve the k nearest feature vectors of a query vector is given by the following steps:

1. The query feature vector is quantized by the lattice vector v_{query};
2. Retrieve in the database all the feature vectors quantized by the lattice vector v_{query} (it is done using the indexing method that we presented in [16, 18, 19]);
3. If the number k_1 of feature vectors quantized by v_{query} is at least equal to k, then go to step (6); or else go to step (4);
4. Determine the set E of lattice vectors corresponding to the neighbors of v_{query} up to a maximum square distance D. E is given by:

$$E = [E_1, E_2, ..., E_d, ..., E_D] \quad \text{for } d \in \mathcal{P} \tag{4}$$

with,

$$E_d = \left\{ \mathbf{v} \in \Lambda, v = v_{\text{query}} + \mathbf{x}, \mathbf{x} \in mask_d \right\}; \tag{5}$$

[1]This operation is done off-line.

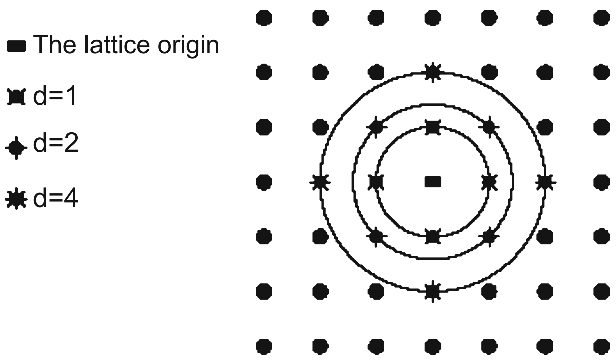

Fig. 4 The lattice points located at the neighborhood of the lattice origin in $\mathbb{Z}^2$

E_d with $1 \leq d \leq D$ represents the dth neighborhood of p, E_0 represents the 0th neighborhood of p .i.e. the neighborhood that contains only the lattice point p.

5. Retrieve the k_2 feature vectors quantized by vectors of E (satisfying $k_1 + k_2 \geq k$);
6. Performs the SSA (sequential search algorithm) on the k_1 or $k_1 + k_2$ feature vectors and return the k nearest feature vectors to the query vector;
7. Exit.

5 The categorization tree based on wavelet features

The categorization tree divides in an unsupervised way the feature vectors into successively smaller lattice partitions. This is done by progressively quantizing the feature vectors with finer scaling factors at each lower level of the tree.

The construction of the categorization tree is based on these three steps:

– In a given level i of the categorization tree, quantize the wavelet feature vectors in the lattice with a scaling factor γ_i. We call this lattice: *Lattice*$_i$.
– For each full lattice point p of *Lattice*$_i$:

  1. Reduce the dimensionality of the wavelet feature vectors quantized in p. This is done by applying the feature selection method presented in Section 6.
  2. In the level $i+1$, re-quantize the obtained feature vectors of reduced dimensionality with a finer scaling factor γ_{i+1} ($\gamma_{i+1} < \gamma_i$) in the lattice point children of p.

Figure 5 presents an example of the categorization tree. Each index $(I_N, I_{\mathcal{L}}, I_P)$ represents a branch in the categorization tree. The feature vectors $C(p_{1,1})$ quantized with a scaling factor γ_1 (at the level 1) in the lattice point $p_{1,1}$ indexed by $(I_N = 9, I_{\mathcal{L}} = 5, I_P = 6)$ are re-quantized with a scaling factor γ_2 (at the level 2) in the lattice points $p_{1,2}$ and $p_{2,2}$ children of $p_{1,1}$. ($p_{1,2}$ and $p_{2,2}$ are respectively indexed by $(I_N = 10, I_{\mathcal{L}} = 3, I_P = 6)$ and $(I_N = 3, I_{\mathcal{L}} = 6, I_P = 1)$).

The lattice points located at the bottom level of the tree are called final lattice points (for example $p_{2,1}$ and $p_{2,2}$). Those located at an intermediate level of the tree are called internal lattice point (for example $p_{1,1}$ and $p_{1,2}$). The branch of each internal lattice point p is pointed on $C(p)$, $Cat(p)$, $DIM(p)$ and the branches of the

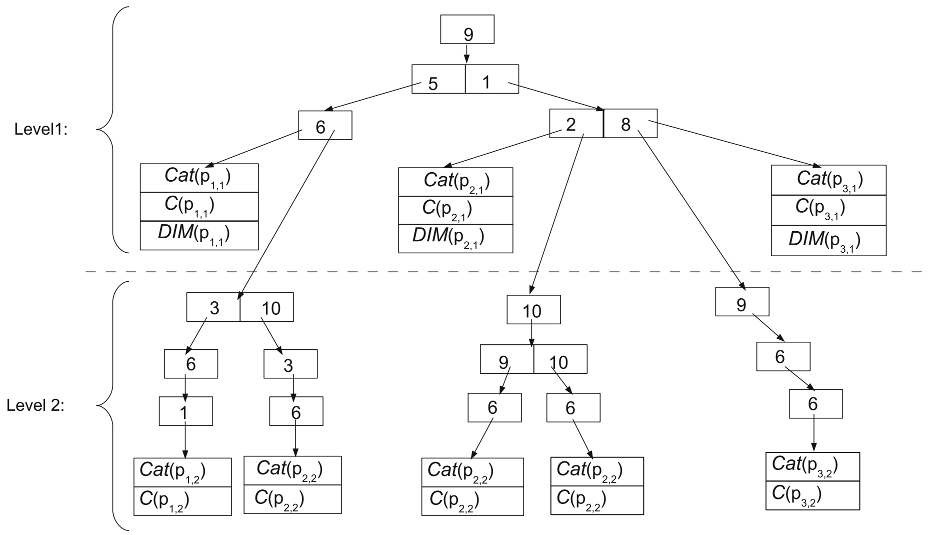

Fig. 5 Example of the categorization tree based on wavelet features

lattice point children of p (with $DIM(p)$ representing the dimensions which were kept for the lattice point p). The branch of each final lattice point p is pointed on the matrix $C(p)$ and the vector $Cat(p)$. The supervised classification is carried out through the labels stored in $Cat(p)$. The labels provide two functions:

1. The selection of the relevant dimensions from the feature vectors clustered in the same lattice point (see Section 6 that describes the proposed feature selection method).
2. The extraction of the high level descriptor (the probability vector) for each salient object (see Section 8 that describes in detail the high level descriptor extraction method).

6 Features selection

The feature selection method is based on the elimination of irrelevant dimensions. This elimination is applied separately on each group of feature vectors quantized in the same lattice point.

In a given lattice point p, a dimension i is considered as irrelevant if it simultaneously complies with the two following requirements:

The first requirement is verified if the dimension i minimizes the inter-category variability. Hereafter, we give in detail the proposed steps to decide if the dimension i verifies the first requirement:

1. We compute the centroid of each category in $Cat(p)$. (The centroid is computed as follows: the ith position in the centroid is the mean (average) of the ith positions of all the vectors labelled by the same category.)

2. We compute all pairwise distances between the ith components of the calculated centroids in step (1).
3. We compute dd_i: the mean of all obtained pairwise distances.
4. If $(max_i < \gamma/2$ or $min_i > 1 - \gamma/2)$ then
   - If $(dd_i \leq \alpha \times \gamma/2)$ then we consider that the dimension i verifies the first requirement.
     (with max_i and min_i denote respectively the greatest value and the lowest value in the dimension i, received by the feature vectors quantized in the same lattice point p).
5. Otherwise
   - If $(dd_i \leq \alpha \times \gamma)$ then we consider that the dimension i verifies the first requirement.

The second requirement is verified if the dimension maximizes the intra-category variability. Hereafter, we give in detail the proposed steps to decide if the dimension i verifies the second requirement:

1. We compute all pairwise distances between the ith components of the feature vectors associated in $Cat(p)$ to the same keyword.
2. We compute ds_i: the mean of all obtained pairwise distances.
3. If $(max_i < \gamma/2$ or $min_i > 1 - \gamma/2)$ then
   - If $(ds_i \geq \beta * \gamma/2)$ then we consider that the dimension i verifies the second requirement.
4. Otherwise
   - If $(ds_i \geq \beta * \gamma)$ then we consider that the dimension i verifies the second requirement.

As the maximum distance between two coefficients quantized with the scaling factor γ is equal to γ, α and β belong to [0, 1]. α and β are experimentally chosen so that they ensure the best trade-off between the number of reduced features and the categorization accuracy.

The selection of the most relevant feature dimensions allows us to ameliorate the categorization accuracy. Indeed, the feature vectors tend to be better clustered, due to the operated dimensionality reduction that eliminates the irrelevant dimensions. We illustrate in Fig. 6 the impact of the feature selection method in the amelioration of the categorization accuracy. We observe that:

- With the feature selection, the reduced feature vectors labelled by the same label "Flower" are quantized by $\gamma = 1/6$ in the same lattice point (1, 1, 1).
- Without the feature selection, the feature vectors labelled by the same label "Flower" are quantized by $\gamma = 1/6$ in distinct lattice points.

Thus, using the feature selection method the feature vectors associated to the same label tend to be clustered in the same lattice point.

Feature vectors labels	Flower	Flower	Bus	Relevance of the dimension
Feature vectors quantized in the same lattice point (1,0,0,0) by $\gamma = 1/2$	0.72	0.39	0.63	Irrelevant because $dd1 = \lvert(0.73 + 0.39)/2 - 0.6\rvert = 0.07 \leq \alpha * \gamma$ and $ds1 = \lvert 0.72 - 0.39\rvert = 0.33 \geqslant \beta * \gamma$
	0.09	0.24	0	Relevant because $dd2 = \lvert(0.09 + 0.24)/2 - 0\rvert = 0.165 > \alpha * \gamma/2$
	0.09	0.21	0.22	Relevant because $ds3 = \lvert 0.09 - 0.21\rvert = 0.12 < \beta * \gamma/2$
	0.09	0.16	0.15	Relevant because $ds4 = \lvert 0.09 - 0.16\rvert = 0.07 < \beta * \gamma/2$

(a) Elimination of the irrelevant feature

Feature vectors labels	Flower	Flower	Bus
Feature vectors	0.72	0.39	0.63
	0.09	0.24	0
	0.09	0.21	0.22
	0.09	0.16	0.15
Lattice points in which the feature vectors are quantized	2	4	4
	1	1	0
	1	1	1
	1	1	1

(b) Re-quantization of the feature vectors by $\gamma = 1/6$

Feature vectors labels	Flower	Flower	Bus
Reduced feature vectors	0.09	0.24	0
	0.09	0.21	0.22
	0.09	0.16	0.15
Lattice points in which the feature vectors are quantized	1	1	0
	1	1	1
	1	1	1

(c) Re-quantization of the reduced feature vectors by $\gamma = 1/6$

Fig. 6 Impact of the feature selection method in the amelioration of the accuracy categorization ($\alpha = 1/6$ and $\beta = 2/3$). **a** The feature vectors are quantized in the same lattice point $(1, 0, 0, 0)$ by $\gamma = 1/2$. **b** Without the elimination of the irrelevant dimension, the feature vectors with the same label "Flower" are re-quantized by $\gamma = 1/6$ in distinct lattice points. **c** After the elimination of the irrelevant dimension, the reduced feature vectors with the same label "Flower" are re-quantized by $\gamma = 1/6$ in the same lattice point

7 Getting the *k* similar salient objects

To search the k nearest neighbors of a given local fuzzy wavelet query feature vector v_{query}, the local query vector is quantized in a first lattice point using the coarser scaling factor γ_1. After that, the query vector traverses the tree downwards in order to extract the initial query result R containing the $nb(SSA)$ nearest wavelet feature vectors (with reduced dimensionality), with $k \leq nb(SSA)$. Finally, we apply SSA on the initial query result R.

When we access a lattice point p located at the level i of the tree, we have two possibilities:

- The lattice point p contains a number of vectors strictly less than k, then we browse the neighborhood of p to complete the missing vectors.
- The lattice point p contains a number of vectors strictly greater than k, then we perform the following steps:
    1. Reduce the dimensionality of the query vector (Relevant dimensions associated with each lattice point p have already been determined and stored in $DIM(p)$ at the stage of tree construction).
    2. Re-quantize the query vector, obtained after the dimensionality reduction, with a scaling factor γ_{i+1} associated with the level $i+1$;
    3. Descend to the lattice point child of p.

Suppose we have a query vector v_{query} quantized in a lattice point p located at a given level i of the tree. We detail hereafter the steps of the method *Search(p,k,*v_{query}*,i)* that searches the k nearest neighbors of v_{query} in the portion of the tree pointed by the lattice point p:

1. If (p is an internal point)
    (a) If (the number $k1$ of vectors quantized in p is strictly less than k), then we apply the following steps:
        i. Put in R the vectors quantized in p (these vectors are those that form the matrix $C(p)$).
        ii. Access lattice points q corresponding to the neighbors of p (E represents the set of lattice points located in the neighborhood of p (see Section 4.2)).
        iii. For each dth neighborhood of p ($1 \leq d \leq D$):
        (D corresponds to the smallest neighborhood that gives a number of feature vectors superior or equal to k (see Section 4.2))
            - For each lattice point q located at the dth neighborhood of p:
            - Compute vq that represents the nearest vector of v_{query}, quantized in the lattice point q:
                - If $p_i < q_i$ then v_i is given by the smallest number that verifies $round(vq_i) = q_i$.
                - If $p_i > q_i$ then v_i is given by the greatest number that verifies $round(vq_i) = q_i$.
                (p_i, q_i, vq_i denote respectively the ith component of the vectors p, q, v).
            - If ($k2 \leq k - k1_{d-1}$) then put in R the vectors quantized in q.
            ($k2$ corresponds to the number of vectors quantized in q; $k1_d$ corresponds to the number of vectors found in the dth neighborhood of p; $k1_0 = k1$ corresponds to the number of vectors quantized in the lattice point p)
            - or else

- Reduce the dimensionality of vq.
- Re-quantize the vector vq_{redu}, obtained after the dimensionality reduction, with a scaling factor γ_{i+1} associated with the level $i+1$.
- Descend to the lattice point child of q.
- Call the method *Search(child of q,* $k - \sum_{i=0}^{d-1} k1_i$, vq_{redu}, $i+1$*)*.

(b) Or else

i. If ($k1$ is strictly greater than k), then

- Reduce the dimension of v_{query} to obtain the vector v_{redu}.
- Re-quantize v_{redu} with a scaling factor γ_{i+1} associated with the level $i+1$.
- Descend to the lattice point child of p.
- Call the method *Search(child of p, k,* v_{redu}, $i+1$*)*.

ii. If($k1 == k$), put in R the feature vectors quantized in p.

2. Or else (i.e. p a final lattice point)

(a) If (the number of vectors quantized in p is strictly less than k), then apply the following steps:

i. Access to the lattice points q located in the neighborhood E of p.
ii. For each lattice point q located in E, put in R feature vectors quantized in q.

(b) Or else (i.e. the number of vectors quantized in p is greater than or equal to k), put in R the feature vectors quantized p.

3. Apply SSA on the feature vectors of the set R to get the k nearest vectors; for that we use the distance given by: $distance(v_{\text{query}}, v) = \frac{\sum_{1 \le i \le dim} (v_{\text{query},i} - v_i)^2}{dim}$ with v_{query} the query vector, $v_{\text{query},i}$ the ith component of v_{query}, v a vector in R, v_i the ith component of v and dim the dimension of the vector v.

8 Image categorization

In what follows we explain in detail the proposed image categorization approach (see Fig. 7):

For a given query image, we extract the salient objects. Then we calculate for each salient object its local fuzzy wavelet descriptor. Afterwards, we seek the k nearest neighbors of each local fuzzy descriptor. This search is based on the categorization tree. Thereafter, each local fuzzy descriptor is transformed into a local probability vector Vp. The ith coefficient of Vp represents the membership degree of the salient object as to the ith semantic category. It is given from the labels associated with the k nearest neighbors:

$$Vp(i) = \frac{N_i}{k}$$

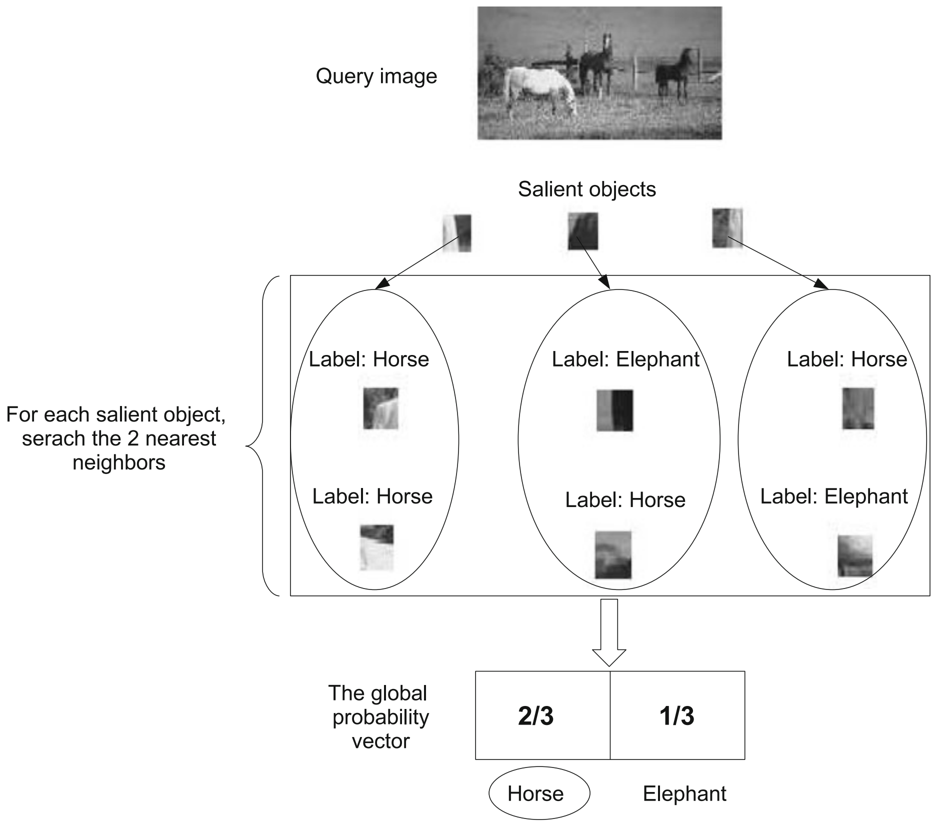

Fig. 7 Categorization of the query image: the query image is assigned to the Horse category

with:

- $Vp(i)$ represents the ith coefficient of the probability vector Vp.
- k the full number of retrieved salient objects.
- N_i represents the number of salient objects, in the k retrieved salient objects, which are labelled by the name of the ith category.

After that, to obtain the global probability vector associated with the whole image, we calculate the average of the local probability vectors associated to the salient objects contained in this image. Finally, an image test is associated with the category receiving a majority vote, i.e. that corresponding to the maximum component in the global probability vector.

9 Image retrieval based on the categorization tree

In what follows we explain in detail the proposed approach to retrieve the k nearest images of a given query image. It consists of two steps:

1. Off-line step: We compute the probability vector of each target image.

2. On-line step: We search for the query image the R similar images in the target images. This search is based on the computation of the distances between the global probability vector of the query image and the global probability vectors assigned to the target images. We only search in a small subset of the target images. This subset corresponds to the images that belong to the pertinent categories. These are derived from the probability vector of the query image. They represent the categories that receive the largest values in the probability vector. A coefficient in the probability vector is considered as a large value if it is superior to $th \times max$, with max corresponding to the maximal value in the probability vector and th a threshold fixed experimentally.
   Note that if we have images with the same distance from the query image, to rank them, we calculate the euclidian distance between the global fuzzy feature vector of the query image and the global fuzzy feature vectors of these images. For a given image, this global fuzzy vector corresponds to the average of the local fuzzy vectors contained in this image.

10 Experimentation and results

10.1 Evaluation criterions

Experiments were performed on a personal computer with configurations: Intel Core2 Duo (2 GHZ), 2.96 GB memory 778 MHz. We evaluate the categorization accuracy. We closely follow the experimental setup suggested by the dataset authors. Thus, for the Corel and Caltech-256 databases we report the average of the per-class classification accuracy [9]. It is given by the ratio of correct predictions to the total number of predictions. For the PASCAL VOC 2007, we measure the Mean Average Precision (MAP) [6]. For that, for each category we measure the Average Precision (AP) and we report the average over the 20 categories.

10.2 Database presentation

10.2.1 Corel database

The Corel [14] database contains images of ten categories. Each category contains 100 images. We closely follow the experimental setup suggested by the dataset authors, i.e. we randomly pick 50 images for training and 50 test images for test. We derive from each image database 1,000 salient points; thus, we form two vector sets (learning and test sets) each containing 500,000 fuzzy vectors of dimension $N_{\text{cluster}} = 250$.

10.2.2 Caltech-256 database

The Caltech-256 [9] dataset contains images of 256 object categories and an additional background class. Each category contains at least 80 images of varying size and quality. The images were downloaded from the web with the help of popular image search engines and then human-filtered. We closely follow the experimental setup suggested by the dataset authors, i.e. we use the first 250 categories of the dataset to measure the categorization rate. We randomly pick 30 images for training and 25 test images per category. We derive from each image database 1,000 salient points; thus we get a training set containing 7,500,000 wavelet fuzzy feature vectors and a test set containing 6,250,000 wavelet fuzzy feature vectors. The dimension of the wavelet fuzzy feature vectors obtained by the bag of words model is equal to $N_{\text{cluster}} = 600$.

10.2.3 PASCAL VOC 2007

The images of the PASCAL VOC 2007 [6] contain objects from 20 object categories. The datasets are extremely challenging because of significant variations of appearances and poses with frequent occlusions. PASCAL VOC 2007 consists of 9,963 images which are divided into two subsets: training data (5,011 images), and test data (4,952 images). We derive from each image database 1,000 salient points; thus we get a training set containing 5,011,000 wavelet fuzzy feature vectors and a test set containing 4,925,000 wavelet fuzzy feature vectors. The dimension of the wavelet fuzzy feature vectors obtained by the bag of words model is equal to $N_{\text{cluster}} = 1{,}024$.

10.3 Categorization trees based on local fuzzy descriptors

10.3.1 Corel database

For the learning set (500,000 vectors), we built a categorization tree with three levels. It is based on wavelet features and it has three levels; in each one, the wavelet feature vectors are quantized with a different scaling factors ($\gamma_1 = 1/2$ for the first level, $\gamma_2 = 1/6$ for the second one and $\gamma_3 = 1/18$ for the third one). We denote that we experimentally choose $\alpha = 0.2$ and $\beta = 0.2$ (α and β are used in Section 6 to select the relevant features). The mean number of the eliminated features for the categorization tree based on wavelet features is given by 41.7 features.

10.3.2 Caltech-256 database

The vectors of the training set are used to build the categorization tree. The categorization tree is made up of four levels. In each one the wavelet feature vectors are quantized with a different scaling factor ($\gamma_1 = 1/2$ for the first level, $\gamma_2 = 1/6$ for the second one, $\gamma_3 = 1/18$ for the third one and $\gamma_4 = 1/36$ for the last one). We denote that we experimentally choose $\alpha = 1/6$ and $\beta = 2/3$. The mean number of the eliminated features for the categorization tree based on wavelet features is given by 63.8 features.

10.3.3 PASCAL VOC 2007

The vectors of the training set are used to build the categorization tree. The categorization tree is made up of four levels. In each one the wavelet feature vectors are quantized with a different scaling factor ($\gamma_1 = 1/2$ for the first level, $\gamma_2 = 1/6$ for the second one, $\gamma_3 = 1/18$ for the third one and $\gamma_4 = 1/36$ for the last one). We denote that we experimentally choose $\alpha = 0.22$ and $\beta = 0.3$. The mean number of the eliminated features for the categorization tree based on wavelet features is given by 112.7 features.

10.4 Comparison with the exact KNN algorithm

We evaluated (Table 1) the categorization rates of the proposed tree comparing it to KNNcuda [7]. We denote that KNNcuda is a parallel implementation of the exact KNN obtained through the C-based API CUDA (Compute Unified Device Architecture). We test with KNNcuda because the exact KNN based on a sequential search is a greedy consuming search time algorithm. For the two databases, the proposed tree outperforms the exact KNNcuda algorithm. The better categorization accuracy obtained by the proposed method compared to the exact KNN is explained by the application of the dimensionality reduction method that ameliorates the classification quality of the feature vectors.

10.5 Comparison with the SR-tree and SOM

In order to show that our method is not only efficient in terms of accuracy of categorization, but also in terms of categorization speed-up, we compared it to the self organizing map algorithm (SOM) [12] and SR-tree [4].

– SR-tree [4] is an indexing tree structure designed for exact KNN search. It is based on a data-partitioning method that can yield possible overlapping regions. Given a vector that represents an object, a similarity query based on SR-tree takes the following three steps:

  1. Find in which partition the given vector resides.
  2. Locate the neighboring partitions where similar vectors may reside. This is often done by locating all the partitions that overlap with the search sphere.
  3. Finally, it computes the distances between the vectors in the neighboring regions (obtained from the second step) and the given vector to return the k nearest neighbors.

  SR-tree permits to reduce the search time in low dimensionality spaces. But, when the data dimension is high, this tree suffers from the curse of dimensionality. Indeed, it fails to divide points into neighborhoods and is forced to access almost all leaves. This often leads to a sequential search and thus a long time is needed to perform the k nearest neighbor search. In this paper, in order to reduce the search time of the SR-tree, the browsing of neighborhood partition in step 2 is prematurely stopped when we reach the first $nb(SSA)$ greater than or equal to the $nb(SSA)$ obtained by the proposed categorization tree. After that, step 3 is applied on the collected vectors in step 2.
– SOM [12] algorithm provides a partitioning of the data into groups or clusters so that data items into a certain group are more similar to each other than to data items in other groups. Each group is then represented by its centroid. Thus, the query point is compared to the centroids, instead of the original data items.

Table 1 Categorization rate comparison with KNNcuda

Corel		Caltech		PASCAL VOC 2007	
Proposed method	KNNcuda	Proposed method	KNNcuda	Proposed method	KNNcuda
0.763	0.722	0.346	0.297	0.438	0.373

We use a map with 100×250 neurons. To retrieve the k nearest neighbors, we seek the nearest cluster center to the query vector. If the found cluster contains a number of vectors equal or superior to k, we stop the search. Otherwise, we complete the search in the neighborhoods of the nearest cluster center and we stop when we reach the neighborhood that gives a number of vectors at least equal to k.

In order to evaluate the speed-up:

- For the Corel database, we considered 500,000 query vectors of the test set.
- For the Caltech and the PASCAL VOC 2007 databases, we used a test set with 1,250,000 query vectors.

The speed-up (Table 2) is given in second and it corresponds to the mean time needed to categorize the images (i.e. the mean time needed to compute the probability vector). The computation of the probability vector is made by searching in the indexing structure (SOM, SR-tree and the categorization tree) the $k = 5$ similar salient objects of the 1,000 ones that represent a query image.

We observe in Table 2 that the proposed categorization tree outperforms SOM and SR-tree not only in terms of categorization rate but also in terms of categorization speed-up. We denote that for the PASCAL VOC 2007 database, since the SOM and SR-tree methods are time consuming algorithms in high dimensionality feature spaces, the categorization rates in Table 2 are given for a codebook size given by $N_{\text{cluster}} = 250$. We highlight that in the case of a codebook size given by $N_{\text{cluster}} =$ 1,024, the speed-up of the categorization tree is equal to 321.6 s and the MAP is equal to 0.438.

The overall retrieval time RT is given by the equation $RT = CT + DT$, with CT corresponding to the categorization time i.e. the time needed to compute the probability vector of the query image and DT corresponding to the time required to compute the distances between the probability vector of the query image and the probability vectors of the target images. Since the image retrieval (see Section 9) is only applied on a small subset of the target images (i.e. the images that belong to the pertinent categories assigned to the query image), the overall retrieval time RT mainly consists of the categorization time CT needed to find these pertinent categories. For example, in the case of for the Caltech-256 database the required time DT to compute the distances between the probability vector of the query image and the probability vectors of the target images is on average equal to 94, 113 and 45 ms based respectively on the probability vectors computed by SOM, SR-tree and the proposed categorization tree. So the time DT is small compared to the time CT.

Table 2 Speed-up and categorization rate comparison with SOM and SR-tree

		SOM	SR-tree	Categorization tree
Corel	Categorization rates	0.693	0.672	0.763
	Time in s	1234.6	154.7	99.4
Caltech	Categorization rates	0.266	0.231	0.346
	Time in s	1781.2	703.9	243.3
PASCAL VOC 2007	Categorization rates	0.307	0.268	0.392
	Time in s	581.9	189.5	124.5

Table 3 Retrieval precision comparison between wavelet features and SIFT-HSV

		$R = 10$	$R = 20$	$R = 30$	$R = 40$
Corel	Wavelet features	0.744	0.732	0.717	0.706
	SIFT-HSV	0.748	0.727	0.714	0.705
Caltech	Wavelet features	0.334	0.318	0.301	0.292
	SIFT-HSV	0.338	0.316	0.302	0.294
PASCAL VOC 2007	Wavelet features	0.735	0.726	0.712	0.696
	SIFT-HSV	0.734	0.728	0.715	0.693

We denote that the coefficient *th* used to select the pertinent categories (see Section 9) is chosen experimentally equal to 2/3.

10.6 Retrieval precision comparison with SIFT-HSV

For each query image in the test set, all other test images were ranked according to their similarity to the query image. The number of same-class images among the top R images (the R nearest neighbors) was computed. When averaged across test images (either within or across classes), this yields a measure known as precision-at-top-R, $R = \{10, 20, 30, 40\}$. We built two categorization trees: the categorization tree that indexes the fuzzy wavelet based features and the categorization tree that indexes the fuzzy SIFT-HSV descriptors. Fuzzy SIFT-HSV descriptors are obtained applying on the SIFT-HSV descriptors [24] the same technique of bag of words presented in Section 3.2. We compare the retrieval precision when we use each categorization tree. We observe in Table 3 that we obtain comparable categorization rates with SIFT-HSV although we use a wavelet descriptor of size 24 against 128×3 for the color SIFT descriptor.

10.7 Categorization comparison with previous results

We compared in Table 4 the categorization performance of the proposed tree based on wavelet features with some previous publications that use local descriptors such as [9, 22, 23, 25, 26] (see introduction section that describes the methods adapted in these publications) and the proposed tree quantizing fuzzy color SIFT-HSV descriptors. For the Corel and Caltech-256 databases, it can be seen in Table 4 that the proposed method gives better results than the methods presented in [9, 23] that

Table 4 Categorization rate comparison with some state-of-the-art methods

	Proposed method	SIFT-HSV	Mouret et al. [22]	Piro et al. [23]
Corel	0.763	0.766	0.706	0.7532
			Todorovic and Ahuja [25]	Griffin et al. [9]
Caltech	0.346	0.348	0.315	0.341
			Van de Sande et al. [26]-salient	Van de Sande et al. [26]-dense
PASCAL VOC 2007	0.438	0.436	0.39	0.45

are based on dense local descriptors, even though we only select 1,000 salient points in the image. For the PASCAL VOC 2007 database, we compare with [26]. This publication uses SIFT-HSV descriptors based on either salient points extracted with the Harris detector or dense sampling. When the salient point based method is used in [26] ([26]-salient in Table 4), our method achieves a better categorization rate. When dense sampling is used ([26]-dense in Table 4), our method yields comparable results.

11 Conclusion

A new hierarchical categorization method based on a combined unsupervised and supervised classification is proposed. The proposed method outperforms the exact KNN algorithm in terms of categorization accuracy. This improvement is due to the proposed feature selection method that ameliorates the classification quality through the elimination of irrelevant dimensions.

References

1. Amar CB, Zaied M, Alimi MA (2005) Beta wavelets. synthesis and application to lossy image compression. Adv Eng Softw 36(7):459–474
2. Athitsos V, Sclaroff S (2005) Boosting nearest neighbor classifiers for multiclass recognition. In: CVPR '05, IEEE Computer Society, Washington, DC
3. Bellil W, Amar CB, Alimi MA (2003) Beta wavelet based image compression. In: International conference on signal system and design SSD03, pp 77–82
4. Bouteldja N, Gouet-Brunet V, Scholl M (2006) Evaluation of strategies for multiple sphere queries with local image descriptors. In: IST/SPIE conference on multimedia content analysis, management, and retrieval. San Jose, CA
5. Claveau V, Tirilly P, Gros P (2008) Language modeling for bag-of visual words image categorization. In: CIVR '08: proceedings of the 2008 international conference on content-based image and video retrieval, pp 249–258
6. Everingham M, Gool LV, Williams CKI, JohnWinn, Zisserman, A (2009) The PASCAL visual object classes (VOC) challenge. Int J Comput Vis (2009). doi:10.1007/s11263-009-0275-4
7. Garcia V, Debreuve E, Barlaud (2008) Fast k nearest neighbor search using GPU. CVPR workshop on computer vision on GPU
8. Giuseppe A, Falchi F (2010) kNN based image classification relying on local feature similarity. SISAP, pp 101–108
9. Griffin G, Holub A, Perona P (2007) Caltech 256 object category dataset. Technical report UCB/CSD-04-1366, California Institute of Technology
10. Grira N, Crucianu M, Boujemaa N (2005) Active semi supervised fuzzy clustering for image database categorization. In: 7th ACM SIGMM international workshop on multimedia information retrieval (MIR'05)
11. Hauptmann AG, Ngo CW, Yang J, Jiang YG (2007) Evaluating bag of visual words representations in scene classification. Multimedia information retrieval, pp 197–206
12. Kaski S, Kangas J, Kohonen T (1998) Bibliography of self-organizing map (SOM) papers: 1981–1997. Neural Comput Surv 1:102–350
13. Kimura A, Kawanishi T, Kashino K (2004) Similarity-based partial image retrieval guaranteeing same accuracy as exhaustive matching. In: Proc. international conference on multimedia and expo (ICME2004), vol 3. Taipei, Taiwan, pp 1895–1898
14. Li J, Wang JZ (2003) Automatic linguistic indexing of pictures by a statistical modeling approach. IEEE Trans Pattern Anal Mach Intell 25:1075–1088
15. Mejdoub M, Fonteles L, Benamar C, Antonini M (2007) Extraction d'une signature floue se basant sur la combinaison de différentes bases d'ondelettes. Traitement et analyse d'images méthodes et applications TAIMA

16. Mejdoub M, Fonteles L, Benamar C, Antonini M (2007) Fast algorithm for image database indexing based on lattice. In: 15th European signal processing conference, EUSIPCO. Pologne, pp 1799–1803
17. Mejdoub M, Fonteles L, Benamar C, Antonini M, (2007) Image retrieval system based on the beta wavelet transform. In: International conference on signal system and devices, SSD
18. Mejdoub M, Fonteles L, Benamar C, Antonini M (2008) Fast indexing method for image retrieval using tree-structured lattices. In: IEEE workshop on content based multimedia indexing, CBMI, London
19. Mejdoub M, Fonteles L, Benamar C, Antonini M (2009) Embedded lattices tree: an efficient indexing scheme for content based retrieval on image databases. J Vis Commun Image Represent 20:145–156
20. Mounira T, Lamrous S, Touati S (2007) Non-overlapping hierarchical index structure for similarity search. Int J Comput Sci 3(1):1544–1559
21. Moureaux J, Loyer P, Antonini M (1998) Low complexity indexing method for Z^n and D_n lattice quantizers. IEEE Trans Commun 46(12):1602–1609
22. Mouret M, Solnon C, Wolf C (2009) Classification of images based on hidden Markov models. In: 7th international workshop on content-based multimedia indexing, pp 169–174
23. Piro P, Anthoine S, Debreuve E, Barlaud M (2009) Sparse multiscale patches (SMP) for image categorization. In: Advances in multimedia modeling. Sophia-Antipolis, France
24. Tao Y, Skubic, M, Han TY, Xia, Chi X (2010) Performance Evaluation of SIFT-Based Descriptors for Object Recognition. In: Proceedings of the international multiconference of engineers and computer scientisits, IMECS
25. Todorovic S, Ahuja N (2006) Extracting subimages of an unknown category from a set of images. In: CVPR
26. Van de Sande K, Gevers T, Snoek C (2008) A comparison of color features for visual concept classification. In: CIVR, pp 141–150
27. Weinberger KQ, Blitzer J, Saul LK (2005) Distance metric learning for large margin nearest neighbor classification. In: NIPS
28. Zhang H, Berg AC, Maire M (2006) Discriminative nearest neighbor classification for visual category recognition. In: CVPR 06, IEEE computer society, Los Alamitos, CA, pp 2126–2136

Mahmoud Mejdoub was born in Sfax (Tunisia) in Mars 1980. He received the Engineer Diploma from the University Of National School of Engineers of Sfax (ENIS) in 2004, the Master in computer science from the University Of National School of Engineers of Sfax (ENIS) in 2005, the PhD degrees in Computer Engineering from the university of Nice Sophia antipolis France in 2010. Nowadays, he is an assistant in the department of computer science in faculty of science and a research member in the Creative team of I3S laboratory and the Research Group of Intelligent Machine (REGIM) Tunisia. His research interests include pattern recognition, semantic image retrieval and image processing. He was the chair of the Workshop on Intelligent Machines: Theories and Applications (WIMTA 2009).

Chokri Ben Ammar received the B.S. degree in Electrical Engineering from the National Engineering School of Sfax (ENIS) in 1989, and the M.S. and PhD degrees in Computer Engineering from the National Institute of Applied Sciences in Lyon, France, in 1990 and 1994, respectively. He spent a year at the University of "Haute Savoie" (France) as a teaching assistant and researcher before joining the higher School of Sciences and Techniques of Tunis as Assistant Professor in 1995. In 1999, he joined the Sfax University (USS), where he is currently an associate professor in the Department of Electrical Engineering of the National Engineering School of Sfax (ENIS), and the Vice director of the REsearch Group on Intelligent Machines (REGIM). His research interests include Computer Vision and Image and video analysis. These research activities are centered around Wavelets and Wavelet networks and their applications to data Classification and approximation, Pattern Recognition and image and video coding, content image retrieval and watermarking. He is a senior member of IEEE, and the chair of the IEEE SPS Tunisia Chapter, founded on January 12, 2009. He was the chair of the Workshop on Intelligent Machines: Theories and Applications (WIMTA 2008) and the chairman of the organizing committees of the "Traitement et Analyse de l'Information: Mthodes et Applications (TAIMA 2009)" conference, International Conference on Machine Intelligence ACIDCA-ICMI'2005 and International Conference on Signals, Circuits and Systems SCS'2004.