IPSJ Transactions on Computer Vision and Applications Vol. 1 46–57 (Jan. 2009)

Regular Paper

Content-based Image Retrieval by Indexing Random Subwindows with Randomized Trees

Raphaël Marée,[†1] Pierre Geurts[†2,†3] and Louis Wehenkel[†2]

We propose a new method for content-based image retrieval which exploits the similarity measure and indexing structure of totally randomized tree ensembles induced from a set of subwindows randomly extracted from a sample of images. We also present the possibility of updating the model as new images come in, and the capability of comparing new images using a model previously constructed from a different set of images. The approach is quantitatively evaluated on various types of images and achieves high recognition rates despite its conceptual simplicity and computational efficiency.

1. Introduction

With the improvements in image acquisition technologies, large image collections are available in many domains. In numerous applications, users want to search efficiently images in such large databases but semantic labeling of all these images is rarely available, because it is not obvious to describe images exhaustively with words, and because there is no widely used taxonomy standard for images. Thus, one well-known paradigm in computer vision is "content-based image retrieval" (CBIR) ie. when users want to retrieve images that share some similar visual elements with a query image, without any further text description neither for images in the reference database, nor for the query image (see **Fig. 1**). To be practically valuable, a CBIR method should combine computer vision techniques that derive rich image descriptions, and efficient indexing structures[2]. Moreover, general methods are needed that can be applied on new problems without requiring time-consuming development and tuning. For example, in the biomedical field, a generic method could help to quickly generate hypotheses on biological processes based on the image similarities they induce, and the types of images and acquisition technologies that could be investigated are very numerous [16].

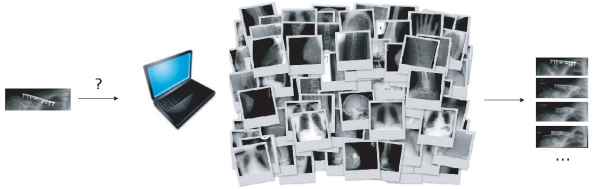

Fig. 1 Illustration of the goal of a CBIR system for one query image (left) from the IRMA-2005 dataset.

Following these requirements, our starting point is the image classification method of Refs. 13) and 15), where the goal was to build models able to predict accurately the class of new images, given a set of training images where each image is labeled with one single class among a finite number of classes. Their method was based on random subwindow extraction and ensembles of extremely randomized trees [7]. In addition to good accuracy results obtained on various types of images, this method has attractive computing times. These properties motivated us to extend their method for CBIR where one has to deal with very large databases of unlabeled images.

The paper is organized as follows. The proposed CBIR method is presented in Section 2. To assess its performances and usefulness as a foundation for image retrieval, we evaluate it on several datasets representing various types of images in Section 3, where the influence of its major parameters will also be evaluated. Method parameters and performances are further discussed in Section 3.5. Finally, we conclude with some perspectives.

2. Method Rationale and Description

We now describe the different steps of our algorithm: extraction of random subwindows from images (2.1), construction of a tree-based indexing structure for these subwindows (2.2), derivation of a similarity measure between images

†1 GIGA Bioinformatics Platform, University of Liège, Belgium
†2 Systems and Modeling Unit, Department of Electrical Engineering and Computer Science, University of Liège, Belgium
†3 Research Associate, FNRS, Belgium

from an ensemble of trees (2.3), and its practical use for image retrieval (2.4).

2.1 Extraction of Random Subwindows

Occlusions, cluttered backgrounds, and viewpoint or orientation changes that occur in real-world images motivated the development of object recognition or image retrieval methods that model image appearances locally by using the so-called "local features" [24]. Indeed, global aspects of images are considered not sufficient to model variabilities of objects or scenes and many local feature detection techniques were developped for years. These consider that the neighboorhood of corners, lines/edges, contours or homogenous regions capture interesting aspects of images to classify or compare them. However, a single detector might not capture enough information to distinguish all images and recent studies [28] suggest that most detectors are complementary (some being more adapted to structured scenes while others to textures) and that all of them should ideally be used in parallel. One step further, several recent works evaluated dense sampling schemes of local features, e.g., on a uniform grid [4] or even randomly [13),19]. In this work, we use the same subwindow random sampling scheme than Ref. 13): square patches of random sizes are extracted at random locations in images, resized by bilinear interpolation to a fixed-size (16×16), and described by HSV values (resulting into 768 feature vectors) for color images, or gray intensities (resulting into 256 feature vectors) for graylevel images. This provides a rich representation of images corresponding to various overlapping regions, both local and global, whatever the task and content of images. Using raw pixel values as descriptors avoids discarding potentially useful information while being generic, and fast. The procedure is illustrated in **Fig. 2**.

2.2 Indexing Subwindows with Totally Randomized Trees

In parallel to these computer vision developments, and due to the slowness of nearest neighbor searches that prevent real-time response times with hundreds of thousands of local feature points described by high-dimensional descriptors, several tree-based data structures and/or approximate nearest neighbors techniques have been proposed [1),6),9),18),21),23),25)] for efficient indexing and retrieval.

In this paper, we propose to look at ensembles of totally randomized trees [7)] for indexing (random) local patches. These kind of trees have been shown more accurate than single trees in classification settings, and competitive with other

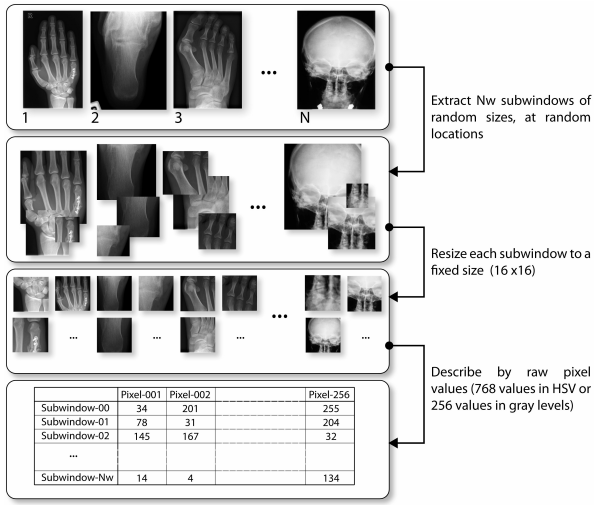

Fig. 2 Extraction of random subwindows described by raw pixel values.

tree-based ensemble techniques while being faster to build [7)]. Moreover, totally randomized trees can be used in an unsupervised way so they are suitable for content-based image retrieval with unlabeled images.

The method recursively partitions the training sample of subwindows by randomly generated tests. Each test is chosen by selecting a random pixel component (among the 768 subwindows descriptors for color images) and a random cut-point in the range of variation of the pixel component in the subset of subwindows associated to the node to split. Each test associated to an internal node of a tree thus simply compares a pixel component to a numerical threshold, see **Fig. 3**. The development of a node is stopped as soon as either all descriptors are constant in the leaf or the number of subwindows in the leaf is smaller than a predefined threshold $n_{\min}$. A number T of such trees are grown from the training sample. The method thus depends on two parameters: $n_{\min}$ and T. We will discuss below their impact on the similarity measure defined by the tree ensemble. Figure 3 illustrates one totally randomized tree.

There exists a number of indexing techniques based on recursive partitioning. The two main differences between the present work and these algorithms is the

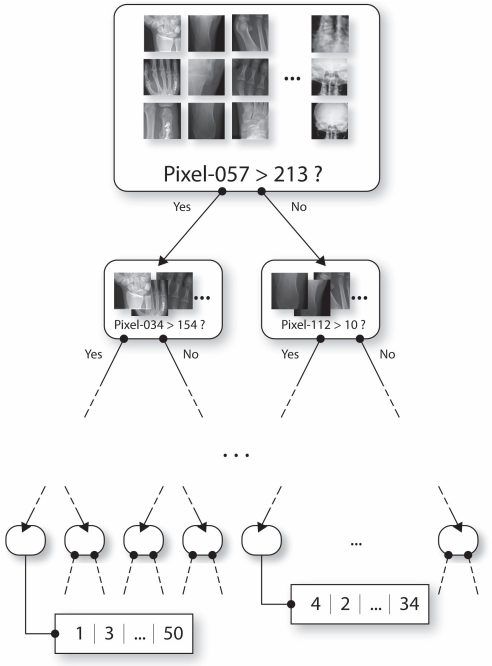

Fig. 3 Indexing subwindows with one single tree.

use of an ensemble of trees instead of a single one and the random selection of tests in place of more elaborated splitting strategies (e.g., based on a distance metric computed over the whole descriptors in Refs. 18) and 25) or taken at the median of the pixel component whose distribution exhibits the greatest spread in Ref. 6)). Because of the randomization, the computational complexity of our algorithm is essentially independent of the dimensionality of the feature space and, like other tree methods, is $O(N \log(N))$ in the number of subwindows. This makes the creation of the indexing structures extremely fast in practice. Indexing with totally randomized trees is also somehow related to the random projection method of locality-sensitive hashing (LSH) [9] used to approximate nearest neighbor searches. That method constructs multiple hash tables, each using a concatenation of different random hyperplanes to hash objects. Their basic assumption is that nearby objects are more likely to be hashed to the same bucket than distant ones. Once a new object is hashed to a bucket, a similarity measure is computed between this object and all reference objects that were hashed in the same bucket during indexing phase.

Note that totally randomized trees are a special case of the Extra-Trees method exploited in Ref. 13) for image classification. In this latter method, K random tests are generated at each tree node and the test that maximizes some information criterion related to the output classification is selected. Totally randomized trees are thus obtained by setting the parameter K of this method to 1, which desactivates test filtering based on the output classification and allows to grow trees in an unsupervised way. Note however that the image retrieval procedure described below is independent of the way the trees are built. When a semantic classification of the images is available, it could thus be a good idea to exploit it when growing the trees (as it would try to put subwindows from the same class in the same leaves). In this particular case, our approach would then be related to Ref. 17) that also proposed an extension of Ref. 13) that uses extremely randomized trees to build visual vocabularies before applying a SVM classifier. However, they use binary encoding in tree leaves, instead of the real-valued similarity measure we define in the next section, and perform image classification from labeled images, not image retrieval.

2.3 Inducing Image Similarities from Tree Ensembles

Denoting by N_L the number of subwindows of the learning sample that are located at a certain leaf L of a tree $\mathcal{T}$, this tree defines the following similarity between any two subwindows s and s'[7]:

$$k_{\mathcal{T}}(s, s') = \begin{cases} \frac{1}{N_L} & \text{if } s \text{ and } s' \text{ reach the same leaf } L \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

This expression amounts to considering that two subwindows are *very similar* if they fall in a same leaf that has a *very small* subset of training subwindows [⋆1].

The similarity induced by an *ensemble* of T trees is furthermore defined by:

⋆1 Intuitively, as it is less likely a priori that two subwindows will fall together in a small leaf, it is natural to consider them very similar when they actually do.

$$k_{ens}(s,s') = \frac{1}{T}\sum_{t=1}^{T} k_{\mathcal{T}_t}(s,s'). \quad (2)$$

This expression amounts to considering that two subwindows are similar if they are considered similar by a large proportion of the trees. The spread of the similarity measure is controlled by the parameter $n_{\min}$: when $n_{\min}$ increases, subwindows tend to fall more often in the same leaf which yields a higher similarity according to Eq. (2). On the other hand, the number of trees controls the smoothness of the similarity. With only one tree, the similarity Eq. (1) is very discrete as it can take only two values when one of the subwindows is fixed. The combination of several trees provides a finer-grained similarity measure and we expect that this will improve the results as much as in the context of image classification. We will study the influence of these two parameters in our experiments.

Given this similarity measure between subwindows, we derive a similarity between two images I and I' by:

$$k(I,I') = \frac{1}{|S(I)||S(I')|}\sum_{s\in S(I), s'\in S(I')} k_{ens}(s,s'), \quad (3)$$

where $S(I)$ and $S(I')$ are the sets of all subwindows that can be extracted from I and I' respectively. The similarity between two images is thus the average similarity between all pairs of their subwindows. Although finite, the number of different subwindows of variable size and location that can be extracted from a given image is in practice very large. Thus we propose to estimate Eq. (3) by extracting at random from each image an a priori fixed number of subwindows.

Notice also that, although Eq. (3) suggests that the complexity of this evaluation is quadratic in this number of subwindows, we show below that it can actually be computed in linear time by exploiting the tree structures.

Since Eq. (2) defines a positive kernel [7] among subwindows, Eq. (3) actually defines a positive (convolution) kernel among images [26]. This means that this similarity measure has several nice mathematical properties. For example, it can be used to define a distance metric and it can be directly exploited in the context of kernel methods [26].

2.4 Image Retrieval Algorithms

In image retrieval, we are given a set of, say N_R, reference images and we want to find images from this set that are most similar to a query image I_Q. We propose the following procedure to achieve this goal:

- We randomly extract N_{ls} subwindows of variable size and location from each reference image, resize them to 16×16, and grow an ensemble of totally randomized trees from them.
- We compute $k(I_Q, I_R)$ according to Eq. (3) for each reference image I_R and returns the N most similar images to the query according to these similarities.

Denoting by N_{ts} the number of subwindows extracted from the query image, the similarity Eq. (3) between two images I_R and I_Q can be rewritten:

$$k(I_Q,I_R) = \sum_{t=1}^{T}\frac{1}{T}\sum_{L\in\mathcal{T}_t}\frac{1}{N_L}\frac{N_{I_Q,L}}{N_{ts}}\frac{N_{I_R,L}}{N_{ls}}, \quad (4)$$

where the inner sum is over leaves of the t-th tree of the ensemble, and $N_{I_Q,L}$ (resp. $N_{I_R,L}$) is the number of subwindows from I_Q (resp. I_R) that reach leaf L.

Formulation (4) suggests the following steps for computing efficiently these similarities.

Creation of the Indexing Structure. During tree growing, at each leaf L of each tree, we record for each image I_R of the reference set that appears in the leaf the number of its subwindows, $N_{I_R,L}$, that have reached this leaf. Notice that once this indexing structure has been built, the database of training subwindows and the original images are not required anymore to compute the similarity.

Recall of Reference Images Most Similar to a Query Image. The similarities between the query I_Q and all N_R reference images are then computed by propagating into each tree the N_{ts} subwindows from the query image, and by incrementing, for each subwindow s of I_Q, each tree $\mathcal{T}$, and each reference image I_R, the similarity $k(I_Q, I_R)$ by the proportion $N_{I_Q,L}/N_L$ of subwindows in the leaf L reached by s that comes from I_R, and by dividing the resulting score by $TN_{ls}N_{ts}$.

One can thus identify the N most similar reference images in $O(NN_R)$ operations, and the complexity of the whole computation is on the average of $O(TN_{ts}(\log(N_{ls}) + N_R))$. Notice that the fact that information about the most similar reference images is gathered progressively as the number of subwindows of the query image increases could be exploited to yield an *anytime* recall procedure.

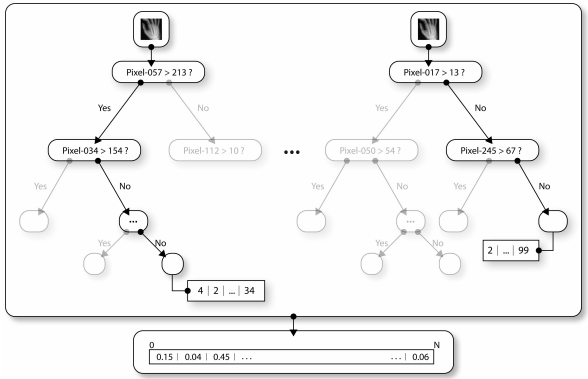

Fig. 4 Propagation of one subwindow into each tree, and update of the similarity measure to the N reference images.

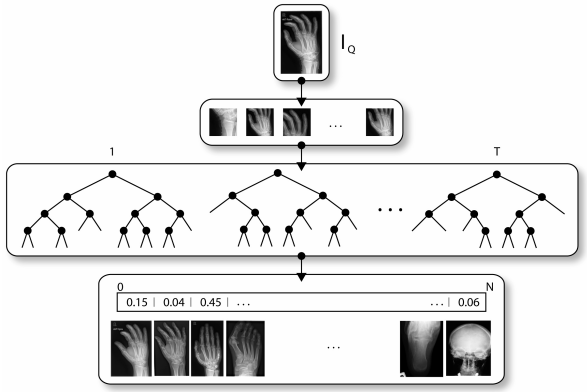

Fig. 5 Once all the subwindows of the query image I_Q are propagated, the N reference images are ranked according to their similarity to the query image.

The process is illustrated in **Figs. 4** and **5**.

2.4.1 Computation of the Similarity between Query Images.

The above procedure can be extended to compute the similarity of a query-image to another image not belonging to the reference set, an extension we name *model recycling*. To this end, one propagates the subwindows from each image through each tree and maintains counts of the number of these subwindows reaching each leaf. The similarity Eq. (3) is then obtained through Eq. (4) by summing over tree leaves the product of the subwindow counts for the two images divided by the number of training subwindows in the leaf and by normalizing the resulting sum.

2.4.2 Incremental Mode.

One can incorporate the subwindows of a new image into an existing indexing structure by propagating and recording their leaf counts. When, subsequently to this operation a leaf happens to contain more than $n_{\min}$ subwindows, the random splitting procedure would merely be used to develop it. Because of the random nature of the tree growing procedure, this incremental procedure is likely to produce similar trees as those that would be obtained by rebuilding them from scratch. In real-world applications such as World Wide Web image search engines, medical imaging in research or clinical routine, or software to organize user photos, this incremental characteristic will be of great interest as new images are crawled by search engines or generated very frequently.

3. Experiments

In this section, we perform a quantitative evaluation of our method in terms of its retrieval accuracy on datasets with ground-truth labels. We study the influence of the number of subwindows extracted in training images for building the tree structure (N_{ls}), the number of trees built (T), the stop-splitting criterion ($n_{\min}$), and the number of images extracted in query images (N_{ts}). Like other authors, we will consider that an image is relevant to a query if it is of the same class as the query image, and irrelevant otherwise. Then, different quantitative measures [3] can be computed. In order to compare our results with the state of the art, for each of the following datasets, we will use the same protocol and performance measures than other authors. Note that, while using class labels to assess accuracy, this information is not used during the indexing phase.

3.1 Image Retrieval on UK-Bench

The University of Kentucky recognition benchmark is a dataset introduced in Ref. 18) and recently updated that now contains 640×480 color images of 2,550 classes of 4 images each (10,200 images in total), approximately 1.7 GB of JPEG

Fig. 6 Several images of the UK-Bench. One image for 6 objects (top), and the four images of the same object (bottom) illustrating viewpoint and illumination changes.

files. These images depict plants, people, cds, books, magazines, outdoor/indoor scenes, animals, household objects, etc., as illustrated in **Fig. 6**. The full set is used as the reference database to build the model. Then, the measure of performance is an average score that counts for each of the 10,200 images how many of the 4 images of this object (including the identical image) are ranked in the top-4 similar images. The score thus varies from 0 (when getting nothing right) up to 4 (when getting everything right). Average scores of variants of the method presented in Ref. 18) range from 3.07 to 3.29 (ie. recognition rates ⋆1 from 76.75% to 82.36%, see their updated website ⋆2), using among the best detector and descriptor combination (Maximally Stable Extremal Region (MSER) detector and the Scalable Invariant Feature Transform (SIFT) descriptor), a tree structure built by hierarchical k-means clustering, and different scoring schemes. Very recently,[23] improved results up to a score of 3.45 using the same set of features but with an approximate k-means clustering exploiting randomized k-d trees.

We obtain scores slightly above 3 (ie. around 75% recognition rate) with 1,000 subwindows extracted per image, 10 trees, and a minimum number of subwindows per node $n_{\min}$ between 4 and 10. In particular, we obtain 75.25% recognition rate with $n_{\min} = 4$. Note that the recognition rate still increases when using more subwindows. For example, a score of 3.10 (77.5% recognition rate) is obtained when 5,000 subwindows are extracted per image with only 5 trees ($n_{\min} = 10$).

3.2 Image Retrieval on ZuBuD

The Zürich Buildings Database ⋆3 is a database of color images of 201 buildings. Each building in the training set is represented by 5 images acquired at 5 arbitrary viewpoints. The training set thus includes 1,005 images and it is used to build the model, while the test set (acquired by another camera under different conditions) that contains 115 images of a subset of the 201 buildings is used to evaluate the generalization performances of the model. The performance measured in[3),21),25)] is the classification recognition rate of the first retrieved image, with 93%, 89.6%, and 59.13% recognition rates respectively. In[20], a 100% recognition rate was obtained, but with recall times of over 27 seconds per image (with an exhaustive scan of the database of local affine frames).

We obtain 95.65% with 1,000 subwindows per image, $T = 10$, and several values of $n_{\min}$ inferior to 10. On this problem, we observed that it is not necessary to use so many trees and subwindows to obtain recognition rates superior to 90%. In particular, only one tree is sufficient, and less than 500 subwindows per image. Some retrieval results are given in **Fig. 7**.

3.3 Image Retrieval on IRMA-2005

The IRMA dataset contains 10000 X-Ray images, which have been acquired at Aachen University of Technology Hospital (RWTH), Germany. These images were acquired using different imaging techniques and modalities, with different

⋆1 (Number of correct images in first 4 retrieved images/40800) ∗ 100%
⋆2 http://www.vis.uky.edu/˜stewe/ukbench/

⋆3 http://www.vision.ee.ethz.ch/showroom/zubud/index.en.html

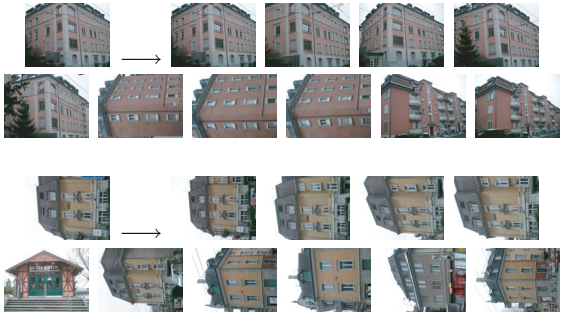

Fig. 7 ZuBuD: First 10 images retrieved for 2 query images.

relative directions of the device and the patient, and they represent different anatomic body parts and biological systems of patients of various ages, genders, and pathologies. All images are in grey levels and were downscaled to fit into a 512×512 bounding box maintaining the original aspect ratio. All images were classified according to the IRMA code [12]. Based on this code, 57 classes were defined. The training set includes 9,000 images and is used to build the model, while the test set contains 1,000 images used to evaluate the generalization performance of the model. The performance measure is the classification recognition rate of the first retrieved image. Classifying all images into the class with the highest number of training images yields a recognition rate of 29.7% (because of the class imbalance of the training and test sets), a nearest-neighbor classifier using euclidian distance computed on downscaled 32×32 images obtains 63.2%, and the best unsupervised method obtains 87.4% [5].

Using our method, we obtain 85.4% recognition rate with 1000 subwindows per image, $T = 10$, and $n_{\min} = 2$. Some retrieval results are given in **Fig. 8**.

3.4 Model Recycling on META and UK-Bench

In our last experiment, we evaluate the *model recycling* idea, ie. we want to assess if given a large set of unlabeled images we can build a model on these images, and then use this model to compare new images from another set.

To do so, we build up a new dataset called META that is basically the collection of images from the following publicly available datasets: LabelMe Set1-16, Caltech-256, Aardvark to Zorro, CEA CLIC, Pascal Visual Object Challenge

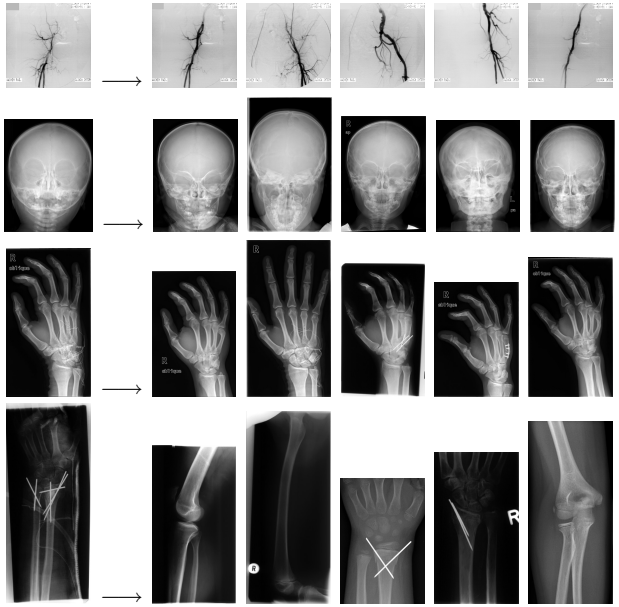

Fig. 8 IRMA-2005: First images retrieved for 4 query images. All images retrieved for the first 3 queries are from the correct classes, but none are correct for the last query although they exhibit some visual resemblances.

2007, Natural Scenes A. Oliva, Flowers, WANG, Xerox6, Butterflies, Birds. This sums up to 205763 color images (about 20 GB of JPEG image files) that we use as training data from which we extract random subwindows and build the ensemble of trees. Then, we exploit that model to compare the UK-Bench images between themselves. Using the same performance measure as in Section 3.1, we obtain an average score of 2.64, i.e., a recognition rate of 66.1%, with 50 subwindows per training image of META (roughly a total of 10 million subwindows), $T = 10$, $n_{\min} = 4$, and 1,000 subwindows per test image of UK-Bench. For comparison, we obtained a score of 3.01 ie. 75.25% recognition rate using the full UK-Bench set as training data and same parameter values. Unsurprisingly, the recognition rate is better when the model is built using the UK-Bench set as training data but we still obtain an interesting recognition rate with the META model. Nistér and Stewénius carried out a similar experiment in Ref. 18), using different

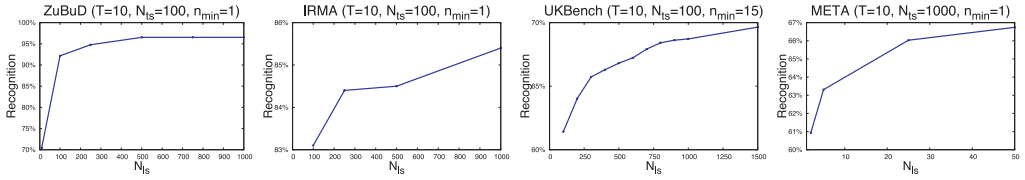

Fig. 9 Influence of the number of subwindows per training image on ZuBuD, IRMA-2005, UK-Bench, and META.

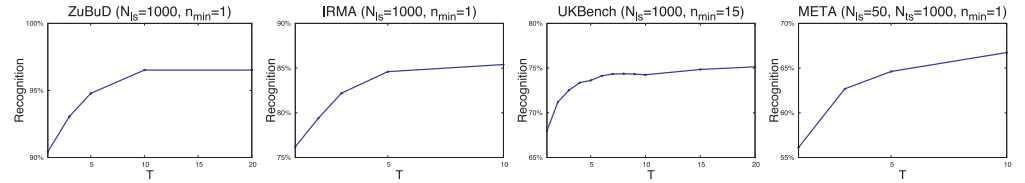

Fig. 10 Influence of the number of trees T on ZuBuD, IRMA-2005, UK-Bench, and META.

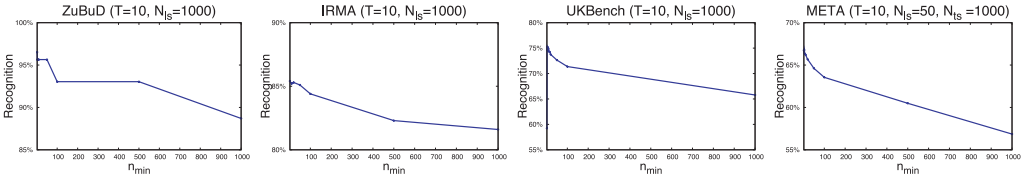

Fig. 11 Influence of the tree depth (minimum node size $n_{\min}$) on ZuBuD, IRMA-2005, UK-Bench, and META.

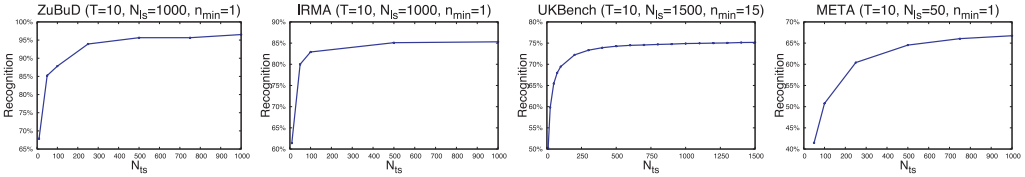

Fig. 12 Influence of the number of subwindows per query image on ZuBuD, IRMA-2005, UK-Bench, and META.

training sets (images from moving vehicles and/or cd covers) to build a model to compare UK-Bench images. They obtained scores ranging from 2.16 to 3.16 (using between 21 and 53 millions local features), which are also inferior to what they obtained exploiting the UK-Bench set for building the model.

3.5 Influence of Parameters

Figures 9, **10**, **11** and **12** show the influence of the parameters on recognition results for the 4 experimental setups. Some general comments can be drawn from these experiments. First, we observed that the more trees and subwindows (for training and testing), the better the results. We note that on ZuBuD, a small

number of trees and not so large number of subwindows already gives state-of-the-art results. We also found out that the value of $n_{\min}$ should be rather small. Note that it influences the recognition rate and increasing its value also reduces the memory needed to store the trees (as they are smaller when $n_{\min}$ is larger) and the required time for the indexing phase. It also reduces the prediction time, but with large values of $n_{\min}$ (such as 1,000) image indexes at terminal nodes of the trees tend to become dense, which then slows down the retrieval phase of our algorithm which exploits the sparsity of these vectors to speed up the updating procedure.

4. Discussion

4.1 Computational Requirements

Following the complexity analysis (see Section 2.2 for indexing and 2.4 for retrieval), one can claim that a clear advantage of the method is that the user can more or less control the performance of the method and its parameters could be choosen so as to trade-off recognition performances, computational requirements, problem difficulty, and available resources. For example, with our current proof of concept implementation in Java, one single tree that achieves more than 90% accuracy on ZuBuD is built in less than 1m30s on a single 2.4 Ghz processor, using a total of 1005000 training subwindows described by 768 values, and $n_{\min} = 4$. When testing query images, the mean number of subwindow tests in the tree is 42.10, and it takes about 2 msec to propagate 1,000 subwindows of one image through the tree. The tree requires approximately 9.5 MB including lists of image indexes at terminal nodes. Using T trees roughly multiplies indexing and retrieval times, as well as memory requirements, by T. In our experiment of Section 3.4, to find similar images in UK-Bench based on the model built on META, there are on average 43.63 tests per subwindow in one single tree. On average, all 1000 subwindows of one UK-Bench image are propagated in all the 10 trees in about 0.16 seconds. Moreover, the process of random subwindow extraction and raw pixel description is fast.

In comparison, other methods that use feature detectors and descriptors, and then perform an exhaustive nearest neighbor search in the database of local features, are much more time-consuming. For example, in Ref. 21) it takes over 27 seconds per query image to retrieve its similar images on ZuBuD dataset. Using binary tree structures, for example the LAF-TREE [21], retrieval time can be reduced downto 14 msec, but the whole system (using detection of maximally stable extremal regions, local affine frame constructions, and distance computations at terminal nodes) requires more operations with comparable performances (93% accuracy) than our approach. Noticeably, training of their tree takes approximately 30 hours and its representation requires about 1 GB of memory because local feature descriptors are still needed to perform distance computations at terminal nodes. The vocabulary tree technique [18] also significantly reduces retrieval computational cost compared to non-hierarchical schemes but still the propagation of each local feature descriptor through the tree implies at each level computing the distance between the descriptor vector and several cluster centers. In our trees, each internal test node simply compares a pixel component to a numerical threshold. Regarding the construction of a vocabulary tree, we also expect it to be much more time-consuming than building an ensemble of totally randomized trees as it involves multiple applications of the K-means algorithm on (subsets of) descriptor vectors. Their approach also requires to store cluster center descriptors in every internal tree nodes.

Finally, it is also worth noting that our method is highly parallelizable. Indeed, tree induction, subwindow extraction and their propagation in trees are processes that could be run independantly and their results subsequently aggregated.

4.2 Towards an Universal Model

In Section 3.4 we introduced the META database and model. While this database obviously does not represent the infinite "image space", it is however possible to extract a very large set of subwindows from it, hence we expect that the META model could produce scores distinctive enough to compare a wide variety of images. The results we obtained in our last experiment on the 2,550 object UK-Bench dataset are promising in that sense. Increasing the number of subwindows extracted from the META database and enriching it using other image sources such as the Wikipedia image database dump or frames from Open Video project might increase the generality and power of the META model.

4.3 Exploiting Additional Information

Our image retrieval approach does not require any prior information about

the similarity of training images. Note however that in some applications, such information is available and it could be a good idea to exploit it to design better similarity measures for image retrieval. When this information is available in the form of a semantic labeling of the images, it is easy to incorporate it into our approach, simply replacing totally randomized trees by extremely randomized trees [7] for the indexing of subwindows. It is interesting to note however that our accuracy results on ZuBuD and IRMA-2005 equal the results obtained in Refs. 13), 14) using extremely randomized trees that exploit the image labels during the training stage. This suggests that for some image collections, good image retrieval performances could be obtained with a fast to train and rather simple method and without prior information about the images. Note however that the "image retrieval" approach for classification comes with some additional burden during the prediction stage with respect to the full classification approach of Ref. 13) (since it requires to compute the similarities between the test image and all training images).

Besides a classification, information could also be provided in the form of a set of similar or dissimilar image pairs. Moosmann, et al.[17] propose a method based on extremely randomized trees for exploiting such pairwise constraints to design a similarity measure between images. When a more quantitative information is available about the similarity between training images, one could combine our approach with ideas from[8], where a (kernel-based) similarity is generalized to never seen objects using ensembles of randomized trees.

4.4 Other Applications of the Image Similarity Measure

Finally, note that while we focus here on image retrieval, the similarity (3) is a general purpose image similarity measure that could be exploited in many other applications, such as, e.g., clustering, dimensionality reduction, or supervised learning with kernel methods. As an illustration, **Fig. 13** shows a subset of images from 4 classes of the IRMA-2005 dataset as projected in the first two components computed by kernel PCA [26]. The similarity measure was obtained from $T = 10$ trees grown from the 9,000 training images using 1,000 subwindows per image and $n_{min} = 20$, and the PCA directions were computed from the 296 images (among 9000) of these 4 classes. In this graph, classes 41 and 48 and classes 42 and 49 are almost indistinguishable while these two groups of classes

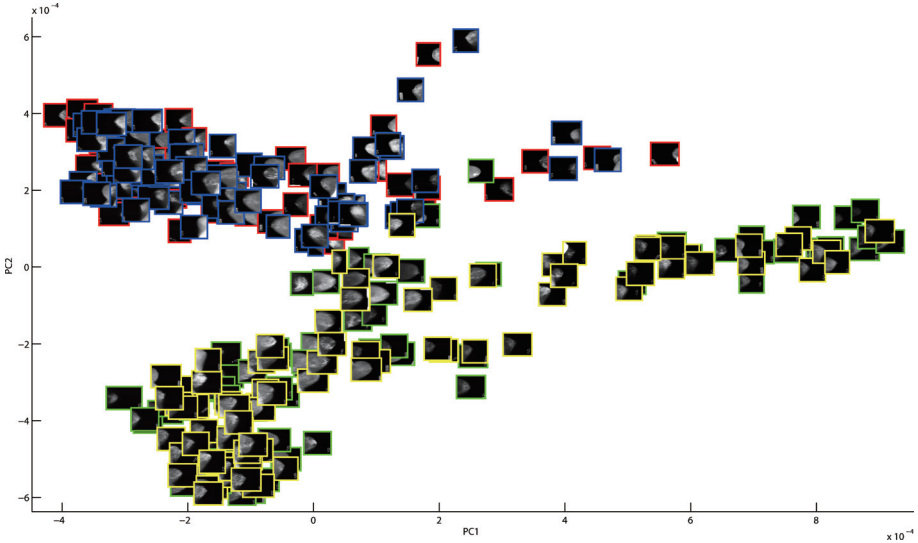

Fig. 13 Visualization of images from 4 classes (red: class 41, green: class 42, blue: class 48, yellow: class 49) from the IRMA-2005 dataset using kernel PCA on the similarity measure.

are well separated from each other. This was expected from a visual inspection of the images of these classes and due to their actual physical meaning. Indeed, classes 41 and 48 represent plain radiographies of right breasts, while classes 42 and 49 represent plain radiographies of left breasts. According to the IRMA code [12], the only difference between classes 41 and 48, and between 42 and 49, is the directional code which models the body orientation (in this case axial orientation for classes 41-42, and other orientation for classes 48-49).

5. Conclusions

In this paper, we used totally randomized trees to index randomly extracted subwindows from unlabeled images for content-based image retrieval. Due to its conceptual simplicity (raw pixel description, and randomization in image description and indexing), the method is fast, and it also requires less tuning than others. While it does not reach state-of-the-art results on each and every problem,

it achieves high recognition performances for diverse image databases including real-word applications (e.g. X-ray medical images or outdoor building pictures).

In future works, other image descriptors and other stop splitting and scoring schemes might be evaluated. For example, the extension of the proposed similarity measure to exploit spatial relationships between subwindows, like in the pyramid match kernel [10] or spatial pyramid matching [11], might be interesting to explore. However, such extensions would probably require additional computational costs.

In order to apply this method to very large and/or distributed databases of images, it will also be of interest to further adapt it to various parallelization schemes, for example schemes where similarities are aggregated from ensembles of tree-based indexing structures towards different subsets of images handled by different storage and computing nodes. Moreover, we highlighted incremental mode and model recycling that could also be useful in such real-world settings. In terms of other applications, the usefulness of the method for the problem of near duplicate image detection might be investigated. Finally, totally randomized trees might also be helpful to index high-dimensional databases of other types of content, and their empirical and theoretical comparisons with other randomized approaches (e.g., random projections methods in locality-sensitive hashing [9]), and randomized lists [27] or ferns [22]), would be worthwhile.

Acknowledgments Raphaël Marée is supported by the GIGA (University of Liège) with the help of the Walloon Region and the European Regional Development Fund. This work presents research results of the Belgian Network BIOMAGNET, funded by the Interuniversity Attraction Poles Programme, initiated by the Belgian State, Science Policy Office. The authors thank Vincent Botta for Figures 2–5.

References

1) Böhm, C., Berchtold, S. and Keim, D.A.: Searching in High-dimensional Spaces — Index Structures for Improving the Performance of Multimedia Databases, *ACM Computing Surveys*, Vol.33, No.3, pp.322–373 (2001).
2) Datta, R., Joshi, D., Li, J. and Wang, J.Z.: Image Retrieval: Ideas, Influences, and Trends of the New Age, *ACM Computing Surveys*, Vol.40, No.2, pp.1–60 (2007).
3) Deselaers, T., Keysers, D. and Ney, H.: Classification Error Rate for Quantitative Evaluation of Content-based Image Retrieval Systems, *Proc. ICPR*, pp.505–508 (2004).
4) Deselaers, T., Keysers, D. and Ney, H.: Discriminative Training for Object Recognition using Image Patches, *Proc. CVPR*, Vol.2, pp.157–162 (2005).
5) Deselaers, T., Müller, H., Clogh, P., Ney, H. and Lehmann, T.M.: The CLEF 2005 Automatic Medical Image Annotation Task, *IJCV*, Vol.74, No.1, pp.51–58 (2007).
6) Friedman, J.H., Bentley, J.L. and Finkel, R.: An algorithm for finding best matches in logarithmic expected time, *ACM Transactions on Mathematical Software*, Vol.3, No.3, pp.209–226 (1977).
7) Geurts, P., Ernst, D. and Wehenkel, L.: Extremely Randomized Trees, *Machine Learning*, Vol.36, No.1, pp.3–42 (2006).
8) Geurts, P., Wehenkel, L. and d'Alché Buc, F.: Kernelizing the output of tree-based methods, *Proc. ICML*, pp.345–352, ACM (2006).
9) Gionis, A., Indyk, P. and Motwani, R.: Similarity search in high dimensions via hashing, *Proc. VLDB*, pp.518–529 (1999).
10) Grauman, K. and Darrell, T.: The Pyramid Match Kernel: Efficient Learning with Sets of Features, *JMLR*, Vol.8, pp.725–760 (2007).
11) Lazebnik, S., Schmid, C. and Ponce, J.: Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories, *Proc. CVPR*, Vol.2, pp.2169–2178 (2006).
12) Lehmann, T., Schubert, H., Keysers, D., Kohnen, M. and Wein, B.: The IRMA code for unique classification of medical images, *SPIE*, Vol.5033, pp.109–117 (2003).
13) Marée, R., Geurts, P., Piater, J. and Wehenkel, L.: Random Subwindows for Robust Image Classification, *Proc. IEEE CVPR*, Vol.1, pp.34–40, IEEE (2005).
14) Marée, R., Geurts, P., Piater, J. and Wehenkel, L.: Biomedical Image Classification with Random Subwindows and Decision Trees, *Proc. ICCV workshop on Computer Vision for Biomedical Image Applications*, Liu, Y., Jiang, T. and C.Z. (ed.), LNCS, Vol.3765, pp.220–229, Springer-Verlag (2005).
15) Marée, R., Geurts, P., Visimberga, G., Piater, J. and Wehenkel, L.: An empirical comparison of machine learning algorithms for generic image classification, *Proc. Conference on innovative techniques and applications of artificial intelligence,* pp.169–182 (2003).
16) Megason, S.G. and Fraser, S.E.: Imaging in Systems Biology, *Cell*, Vol.130, No.5, pp.784–795 (2007).
17) Moosmann, F., Nowak, E. and Jurie, F.: Randomized Clustering Forests for Image Classification, *IEEE PAMI*, Vol.30, No.9, pp.1632–1646 (2008).
18) Nistér, D. and Stewénius, H.: Scalable Recognition with a Vocabulary Tree, *Proc. IEEE CVPR*, Vol.2, pp.2161–2168 (2006).
19) Nowak, E., Jurie, F. and Triggs, B.: Sampling Strategies for Bag-of-Features Image Classification, *Proc. ECCV*, pp.490–503 (2006).
20) Obdržálek, S. and Matas, J.: Image Retrieval Using Local Compact DCT-Based

Representation, *Proc. 25th DAGM Symposium*, Vol.2781, pp.490–497 (2003).
21) Obdržálek, S. and Matas, J.: Sub-linear Indexing for Large Scale Object Recognition, *Proc. BMVC*, pp.1–10 (2005).
22) Ozuysal, M., Fua, P. and Lepetit, V.: Fast Keypoint Recognition in Ten Lines of Code, *Proc. CVPR* (2007).
23) Philbin, J., Chum, O., Isard, M., Sivic, J. and Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching, *Proc. IEEE CVPR* (2007).
24) Schmid, C. and Mohr, R.: Local Greyvalue Invariants for Image Retrieval, *IEEE PAMI*, Vol.19, No.5, pp.530–534 (1997).
25) Shao, H., Svoboda, T., Ferrari, V., Tuytelaars, T. and Van Gool, L.: Fast Indexing for Image Retrieval Based on Local Appearance with Re-ranking, *Proc. IEEE ICIP*, pp.737–749 (2003).
26) Shawe-Taylor, J. and Cristianini, N.: *Kernel Methods for Pattern Analysis*, Cambridge University Press (2004).
27) Williams, B., Klein, G. and Reid, I.: Real-time SLAM relocalisation, *Proc. ICCV* (2007).
28) Zhang, J., Marszalek, M., Lazebnik, S. and Schmid, C.: Local features and kernels for classification of texture and object categories: A comprehensive study, *IJCV*, Vol.73, pp.213–238 (2007).

(Received February 29, 2008)
(Accepted October 15, 2008)
(Released January 30, 2009)

(Communicated by *Shin'ichi Satoh*)

Raphaël Marée received the Ph.D. degree in computer science in 2005 from the University of Liège, Belgium, where he is working for the Bioinformatics Core Facility at the GIGA Interdisciplinary Cluster for Applied Genoproteomics. His current research interests are in the broad area of data mining and computer vision techniques with specific focus on their applications to various problems in biology and medecine.

Pierre Geurts graduated as an Electrical Engineer (in computer science) in 1998 and received the Ph.D. degree in applied sciences, in 2002, both from the University of Liège, Belgium. He is currently a research associate at the Belgian Fund for Scientific Research (F.N.R.S.). His research interests include machine learning and its applications in various fields, such as bioinformatics, computer vision, and computer networks.

Louis Wehenkel graduated as an Electrical Engineer (electronics) in 1986 and received the Ph.D. degree in 1990, both at the University of Liège, where he is full Professor of Electrical Engineering and Computer Science. His research interests lie in the fields of stochastic methods for systems and modeling, machine learning and data mining, with applications in electric power systems planning, operation and control, image analysis and bioinformatics.