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

Raphaël Marée[1], Pierre Geurts[2], Louis Wehenkel[2]

[1] GIGA Bioinformatics Platform, University of Liège, Belgium
[2] Systems and Modeling Unit, Montefiore Institute, University of Liège, Belgium

Abstract. 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 with state-of-the-art results 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. To be practically valuable, a CBIR method should combine computer vision techniques that derive rich image descriptions, and efficient indexing structures [2].

Following these requirements, our starting point is the method of [8], 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 [6]. 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 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 discussed in Section 4. Finally, we conclude with some perspectives.

To appear in Proc. ACCV 2007, LNCS, © *Springer-Verlag (www.springerlink.com)*

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 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" [15]. 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 [18] 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 [8, 11]. In this work, we use the same subwindow random sampling scheme than [8]: 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). 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.

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, 5, 9, 13, 14, 16] for efficient indexing and retrieval.

In this paper, we propose to use ensembles of totally randomized trees [6] for indexing (random) local patches. 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) 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. 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.

There exists a number of indexing techniques based on recursive partitioning. The two main differences between the present work and these algorithms is the 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 [9, 16] or taken at the median of the pixel component whose distribution exhibits the greatest spread in [5]). 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.

Note that totally randomized trees are a special case of the Extra-Trees method exploited in [8] 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).

2.3 Inducing image similarities from tree ensembles

A tree $\mathcal{T}$ defines the following similarity between two subwindows s and s' [6]:

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

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[3].

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

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

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 (1). On the other hand, the number of trees controls the smoothness

[3] 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.

of the similarity. With only one tree, the similarity (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:

$$k(I, I') = \frac{1}{|S(I)||S(I')|} \sum_{s \in S(I), s' \in S(I')} k_{ens}(s, s'), \tag{2}$$

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 (2) by extracting at random from each image an a priori fixed number of subwindows. Notice also that, although (2) suggests that the complexity of this evalation 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 (1) actually defines a positive kernel [6] among subwindows, equation (2) actually defines a positive (convolution) kernel among images [17]. 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 [17].

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. We propose the following procedure to achieve this goal.

Creation of the indexing structure. To build the indexing structure over the reference set, 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. At each leaf of each tree, we record for each image of the reference set that appears in the leaf the number of its subwindows that have reached this leaf.

Recall of reference images most similar to a query image. We compute the similarities between a query image I_Q and all N_R reference images, by propagating into each tree 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 of subwindows of I_R in the leaf

reached by s in $\mathcal{T}$, and by dividing the resulting score by $TN_{ls}N_{ts}$. This procedure estimates $k(I_Q, I_R)$ as given by (2), using N_{ls} and N_{ts} random subwindows from I_R and I_Q respectively. From these N_R similarities, one can 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. Note also that once the indexing structure has been built, the database of training subwindows and the original images are not required anymore to compute the similarity.

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 (2) is then obtained 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.

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 [9] and recently updated that now contains 640×480 color images of 2550 classes of 4 images each (10200 images in total), approximately 1.7GB of JPEG files. These images depict plants, people, cds, books, magazines, outdoor/indoor scenes, animals, household objects, etc., as illustrated by Figure 1. 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 10200 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 [9] range from 3.07 to 3.29 (ie. recognition rates[4] from 76.75% to 82.36%, see their updated website [5]), 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, [14] 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.

Fig. 1. Several images of the UK-Bench. One image for various objects (top), the four images of the same object (bottom).

Figure 2 shows the influence of the parameters of our method on the recognition performances. We obtain scores slightly above 3 (ie. around 75% recognition rate) with 1000 subwindows extracted per image, 10 trees, and a minimum number of subwindows per node $n_{\min}$ between 4 and 10. Note that the recognition rate still increases when using more subwindows. For example, not reported on these figures, a score of 3.10 is obtained when 5000 subwindows are extracted per image with only 5 trees ($n_{\min} = 10$).

[4] (Number of correct images in first 4 retrieved images /40800) $*$ 100%

[5] `http://www.vis.uky.edu/~stewe/ukbench/`

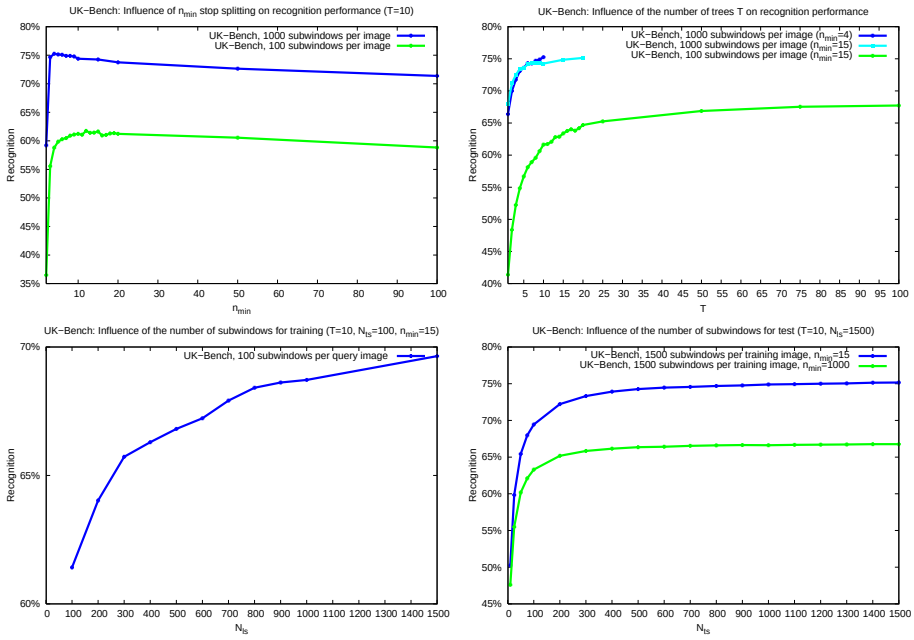

Fig. 2. Influence of the parameters on UK-Bench. Influence of stop splitting parameter, number of trees, number of training subwindows, number of test subwindows.

3.2 Image retrieval on ZuBuD

The Zürich Buildings Database [6] 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 1005 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 [13, 3, 16] is the classification recognition rate of the first retrieved image, with 93%, 89.6%, and 59.13% recognition rates respectively. In [12], 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 1000 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 this state-of-the-art recognition rate. In particular, only one tree is sufficient, or less than 500 subwindows.

[6] `http://www.vision.ee.ethz.ch/showroom/zubud/index.en.html`

3.3 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 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, ie. 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 1000 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 [9], using different 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.

4 Discussion

Some general comments about the influence of parameters can be drawn from our experiments. First, we observed that the more trees and subwindows, 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 neither be too small, nor too large. 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 1000) 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.

One 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 has 94.78% accuracy on ZuBuD is built in less than $1m30s$ on a single 2.4Ghz 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. In our experiment of Section 3.3, 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, random subwindow extraction and raw pixel description are straightforward.

In Section 3.3 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 2550 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.

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 for the indexing of subwindows. Note however that our result on ZuBuD equals the result obtained by [8] using extremely randomized trees that exploit the image labels during the training stage. This result suggests that for some problems, good image retrieval performances could be obtained with a fast and rather simple method and without prior information about the images. Beside a classification, information could also be provided in the form a set of similar or dissimilar image pairs. Nowak and Jurie [10] propose a method based on 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 [7], where a (kernel-based) similarity is generalized to never seen objects using ensembles of randomized trees.

5 Conclusions

In this paper, we used totally randomized trees to index randomly extracted subwindows for content-based image retrieval. Due to its conceptual simplicity (randomization is used both in image description and indexing), the method is fast. Good recognition results are obtained on two datasets with illumination, viewpoint, and scale changes. Moreover, incremental mode and model recycling were highlighted. In future works, other image descriptors and other stop splitting and scoring schemes might be evaluated. 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.

Acknowledgements

Raphaël Marée is supported by the GIGA (University of Liège) with the help of the Walloon Region and the European Regional Development Fund. Pierre Geurts is a research associate of the FNRS, Belgium.

References

1. C. Böhm, S. Berchtold, and D. A. Keim. Searching in high-dimensional spaces - index structures for improving the performance of multimedia databases. *ACM Computing Surveys*, 33(3):322 – 373, September 2001.
2. R. Datta, D. Joshi, J. Li, and J. Z. Wang. Image retrieval: Ideas, influences, and trends of the new age. *ACM Computing Surveys*, 39(65), 2007.
3. T. Deselaers, D. Keysers, and H. Ney. Classification error rate for quantitative evaluation of content-based image retrieval systems. In *Proc. 17th International Conference on Pattern Recognition (ICPR)*, pages 505–508, 2004.
4. T. Deselaers, D. Keysers, and H. Ney. Discriminative training for object recognition using image patches. In *Proc. International Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 157–162, June 2005.
5. J. H. Friedman, J. L. Bentley, and R.A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 3(3):209–226, September 1977.
6. P. Geurts, D. Ernst, and L. Wehenkel. Extremely randomized trees. *Machine Learning*, 36(1):3–42, 2006.
7. P. Geurts, L. Wehenkel, and F. d'Alché Buc. Kernelizing the output of tree-based methods. In *Proc. of the 23rd International Conference on Machine Learning (ICML)*, pages 345–352. ACM, 2006.
8. R. Marée, P. Geurts, J. Piater, and L. Wehenkel. Random subwindows for robust image classification. In *Proc. IEEE CVPR*, volume 1, pages 34–40. IEEE, 2005.
9. D. Nistér and H. Stewénius. Scalable recognition with a vocabulary tree. In *Proc. IEEE CVPR*, volume 2, pages 2161–2168, June 2006.
10. E. Nowak and F. Jurie. Learning visual similarity measures for comparing never seen objects. Proc. IEEE CVPR, 2007.
11. E. Nowak, F. Jurie, and B. Triggs. Sampling strategies for bag-of-features image classification. In *Proc. ECCV*, pages 490–503, 2006.
12. S. Obdržálek and J. Matas. Image retrieval using local compact DCT-based representation. In *Proc. 25th DAGM Symposium*, volume 2781, pages 490–497, 2003.
13. S. Obdržálek and J. Matas. Sub-linear indexing for large scale object recognition. In *Proc. British Machine Vision Conference (BMVC)*, pages 1–10, 2005.
14. J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. Proc. IEEE CVPR, 2007.
15. C. Schmid and R. Mohr. Local greyvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–534, 1997.
16. H. Shao, T. Svoboda, V. Ferrari, T. Tuytelaars, and L. Van Gool. Fast indexing for image retrieval based on local appearance with re-ranking. In *Proc. IEEE International Conference on Image Processing (ICIP)*, pages 737–749, 2003.
17. J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.
18. J. Zhang, M. Marszaek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: a comprehensive study. *International Journal of Computer Vision*, 73:213–238, 2007.