Efficient Object Pixel-Level Categorization using Bag of Features

David Aldavert[1], Arnau Ramisa[2], Ricardo Toledo[1], and Ramon Lopez de Mantaras[2]

[1] Computer Vision Center (CVC)
Dept. Ciències de la Computació
Universitat Autònoma de Barcelona (UAB), 08193, Bellaterra, Spain
{aldavert,ricard}@cvc.uab.cat,
[2] Artificial Intelligence Research Institute (IIIA-CSIC)
Campus de la UAB, 08193, Bellaterra, Spain
{aramisa,mantaras}@iiia.csic.es

Abstract. In this paper we present a pixel-level object categorization method suitable to be applied under real-time constraints. Since pixels are categorized using a bag of features scheme, the major bottleneck of such an approach would be the feature pooling in local histograms of visual words. Therefore, we propose to bypass this time-consuming step and directly obtain the score of a linear Support Vector Machine classifier. This is achieved by creating an integral image of the components of the SVM which can readily obtain the classification score for any image sub-window with only 10 additions and 2 products, regardless of its size. Besides, we evaluated the performance of two efficient feature quantization methods: the Hierarchical K-Means and the Extremely Randomized Forest. All experiments have been done in the Graz02 database, showing comparable, or even better results to related work with a lower computational cost.

1 Introduction

A method for robustly localizing objects is of major importance towards creating smart image retrieval tools able to search in digital image collections. In the last years, object recognition in images has seen impressive advances thanks to the development of robust image descriptors [1] and simple yet powerful representation method such as the *bag of features* [2–5]. Furthermore, the ever-growing collections of images available on the Internet make computational efficiency an imperative for methods that aim to be used in such a scenario.

In this work we propose a new method for fast pixel-wise categorization based on the bag of features object representation. Given that the method will have to be applied at every pixel of the image, it is essential to optimize it to perform in the least possible time. Although different bag of features approaches have been proposed, all of them consist on four basic steps. Namely, feature extraction from the image, feature quantization into visual words, accumulation of visual word into histograms and classification of the resulting histogram.

In order to accelerate the quantization step, Nister and Stewenius [3] proposed to use a Hierarchical K-Means vocabulary tree (HKM) created by recursively applying k-means to the clusters from the previous level. With this technique they were able to improve the recognition results by using larger dictionaries in a reasonable time. As an alternative to the HKM, Moosmann et al. [6] and Shotton et al. [7] proposed to use Extremely Randomized Forests (ERF) of K-D trees to improve the classification accuracy. In the approach proposed by Moosemann et al. random features are selected at high saliency areas and used in a bag of features scheme to classify the whole image. Besides, the probability of each object class for each individual feature is determined and used in an object probability map that iteratively decides the geometric parameters of new random features.

The accumulation step is an even more critical bottleneck for an object localization using bag of features since, as no geometrical information is used, many sub-windows of the image have to be evaluated. Some authors addressed this problem by reducing the number of evaluated windows, either by using a preprocessing step [8] or by searching the best sub-window as in an optimization problem [9]. This is done by defining an upper bound on the SVM classification, and using branch and bound to discard uninteresting areas. Although not used here, the strategy proposed by Lampert et al. is also applicable in our method.

Other authors have focused on accelerating the accumulation step. In the approach by Fulkerson et al. [4], the authors speed up, by means of integral images, the accumulation step in a sliding windows based analysis of the image. For this speed-up measure to be effective, it is important to use small dictionaries. However, various works [10, 3] show that large dictionaries typically obtain better classification results. Therefore, in order to compress the dictionary without losing classification accuracy, they propose to use Agglomerative Information Bottleneck (AIB) to create a coarse-to-fine-to-coarse architecture that is optimized for discrimination of object versus non-object. This approach shares some similitudes with the one presented here. However, In contrast to Fulkerson et al. , we propose to bypass the descriptor accumulation step, and make every computed feature vote directly with its classifier score in an integral image to reduce the computational cost of classifying an image sub-window to only 10 additions and 2 multiplications, regardless of its size.

The rest of the paper is organized as follows: In Section 2 the proposed methodology for object classification and localization in images is described. Then, in Section 3, our proposed method is evaluated with the Graz02 dataset and results are presented and discussed. Finally, in Section 4 the contributions and conclusions of this work, as well as future research directions, are summarized.

2 Efficient pixel-level categorization

Our method[1] uses an efficient categorization algorithm to assign a category label to each pixel of an image: First, region descriptors are densely sampled from the image and quantized into visual words using a codebook. Then, a sliding window scheme is used to assign a category label to each pixel of the image. Visual words within a window are accumulated in a histogram, which is later classified using a linear support vector machine. In Fig. 1 some pixel-level categorization results obtained using the proposed method are shown. Categorizing all the pixels from an image with this brute-force approach in a reasonable time requires each of the previous steps to be executed in a very efficient way.

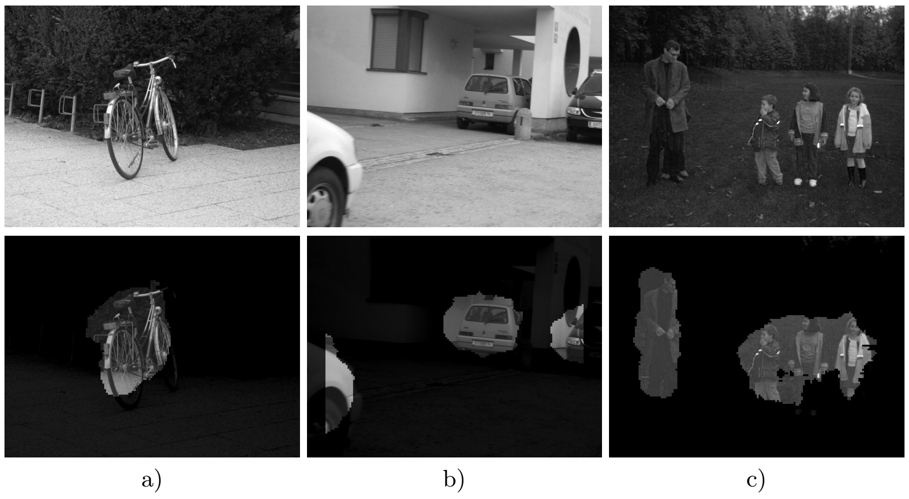

Fig. 1. Examples of pixel-level categorization results obtained with our method for a) bikes, b) cars and c) person in the Graz02 database.

2.1 Dense features

As previously mentioned, we use densely sampled image descriptors as input data. Dense sampling has several advantages when compared to keypoint-based approaches, such as extracting more information from the underlying image, and avoiding the time-consuming keypoint detection step [10]. Furthermore, if robust descriptors can be computed in an efficient way, it can even become faster than the keypoint-based alternative despite the larger number of descriptors computed.

[1] Source code and additional information available at `http://www.cvc.uab.cat/~aldavert/plor/`

With this in mind, we have decided to use the Integral Histograms of Oriented Gradients (IHOG) descriptor [11]. The IHOG is an approximation to the Histograms of Oriented Gradients (HOG) descriptor [12], which speeds up the descriptor extraction using integral images. First, each pixel votes according to its gradient orientation, weighted by its gradient magnitude, in a histogram of N orientation bins. Then, an integral image is generated for each one of the N orientation bins. Using these integral images, to compute an IHOG descriptor with N orientation bins and $P \times P$ position bins (i.e. a $N \times P \times P$ dimensions descriptor) we need just $N \times (P-1)^2$ memory accesses and $N \times P^2$ additions, regardless of the feature region size in pixels.

Unlike the HOG descriptor, the IHOG descriptor is incompatible with the Gaussian mask and the tri-linear interpolation to weight the contribution of the gradient module in the spatial bins of the descriptor used in the HOG. Another difference is that the IHOG descriptor uses L1 normalization instead the L2 normalization. Nevertheless, despite all these simplifications, the performance of the IHOG descriptor is only slightly worse than that of the HOG descriptor [11]. Moreover, neither HOG nor IHOG descriptors are rotation invariant. However, according to Zhang et. al. [13], the use of rotation invariant descriptors has a negative effect in the performance of bag of features approaches.

2.2 Codebook Generation

Once all descriptors have been computed from the image, it is necessary to quantize them into visual words using a codebook. The computational cost of quantizing a D-dimensional descriptor using linear codebook of V visual words is $O(DV)$. From the various alternatives that have been proposed to reduce this computational cost, in this work we have evaluated two: the Hierarchical K-Means (HKM) and the Extremely Randomized Forest (ERF).

The HKM defines a hierarchical quantization of the feature space. Instead of k being the final number of visual words of the codebook, it determines the branch factor (number of children of each node) of a tree. Given a set of training descriptors, an HKM is generated as follows: First, the k-means algorithm is used to split the training data into k groups. Then, this clustering process is recursively applied to the groups from the previous level until a maximum depth is reached. This recursive method creates a vocabulary tree (i.e. codebook) with a reduced computational cost both in the training and descriptor quantization phases. The computational complexity of quantizing a D-dimensional descriptor using a HKM with V visual words is $O(Dk \log_k V)$. In the original implementation of the HKM, all nodes of the tree are used as visual words to alleviate misclassification problems in the superior levels of the tree and the contribution of each node of the histogram is weighted using a TF-IDF scheme. However, the use of these two refinements have a modest impact in the performance of the HKM. Therefore, we have removed them from our implementation.

The ERF [6] uses a combination of several random K-D trees in order to quantize the feature space. Given a set of labeled training descriptors (i.e. descriptors with a category label associated), the K-D trees of the ERF are built recursively in a top-down manner as follows: Every node of the K-D trees splits the training descriptors from the previous level in two disjoint sets with a boolean test in a random descriptor vector position. The boolean test consists in dividing the descriptors in two groups according to a random threshold θ_t applied at descriptor vector dimension D_t, also chosen randomly. For each node, the random boolean test is scored using the Shannon entropy until a minimum value S_{min} is attained or a maximum number of trials T_{max} has been reached. Then, the selected random boolean test is the one that has a highest score. Parameter S_{min} can be used to select the randomness of the obtained K-D trees. For instance $S_{min} = 1$ creates a highly discriminant tree while $S_{min} = 0$ creates a completely random tree. The main advantage of the random K-D tree compared to other quantization methods is its low computational cost. Quantizing a D-dimensional descriptor vector using a random K-D tree with V visual words is $O(\log_2 V)$. Since a random K-D tree usually has less discriminative power than other clustering methods, like the k-means or the HKM, several K-D trees are combined together to obtain a more discriminative codebook. Finally, the resulting histogram of the ERF is created by concatenating the histograms generated by each K-D tree of the forest.

2.3 Integral Linear Classifiers

The "integral image" representation has been first introduced by Viola and Jones to quickly extract Haar-wavelet type features [14]. Since then, integral images have been applied to many different tasks like invariant feature extraction [15], local region descriptors [11], to compute histograms over arbitrary rectangular image regions [16] or to compute bag of feature histograms [4]. Inspired by these previous works, we propose the use of an integral image to quickly calculate the output score of the linear classifier which is applied to bag of features histograms.

To categorize a V dimensional histogram of visual words, we use a linear classifier with weight vector $\boldsymbol{W}$ and bias b. Then, the output score of the linear classifier is:

$$\frac{1}{\|\boldsymbol{X}\|}\sum_{i=0}^{V} x_i w_i + b > 0 \tag{1}$$

where x_i is the frequency of the i-th visual word of the codebook, $\|\boldsymbol{X}\|$ is the norm of histogram $\boldsymbol{X}$ and w_i is the i-th component of the linear classifier weight vector $\boldsymbol{W}$. If all components of $\boldsymbol{W}$ are positive, then sum of the previous equation can be calculated using an integral image. Therefore, we define the classifier weight vector $\tilde{\boldsymbol{W}}$ components as:

$$\tilde{w_i} = w_i - W_m \tag{2}$$

where W_m is the w_i component with the lowest value. Then, replacing $\boldsymbol{W}$ by $\tilde{\boldsymbol{W}}$ in Eq. 1 the output score of the linear classifier is:

$$\frac{1}{\|\boldsymbol{X}\|}\sum_{i=0}^{V} x_i \tilde{w}_i + \frac{W_m}{\|\boldsymbol{X}\|}\sum_{i=0}^{V} x_i + b > 0 \tag{3}$$

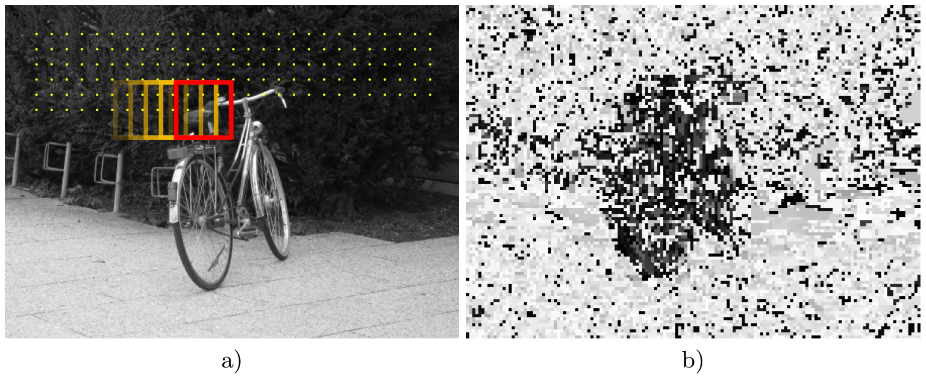

Fig. 2. Image containing the components of a linear classifier for bikes b) obtained from extracting dense features every four pixels in the original image a).

We normalize the histogram $\boldsymbol{X}$ using L1 norm (i.e. the amount of visual words that casted a vote in the histogram) since it is fast to compute using an integral image. Then, Eq. 3 becomes:

$$\frac{1}{N}\sum_{i=0}^{V} x_i \tilde{w}_i + W_m + b > 0 \tag{4}$$

where N is the L1 normalization of histogram $\|\boldsymbol{X}\|$. Once all $\tilde{\boldsymbol{W}}$ components are positive, the integral image can be used to calculate the sum in Eq. 4. For each linear classifier c, let $L_c(x, y)$ be the sum of components $\tilde{w_i^c}$ corresponding to the visual words at pixel (x, y). In Fig. 2 an example of L_c image for the *bikes* classifier is shown. Then, each image L_c is transformed into an integral image I_c, so that, the sum of Eq. 4 of a rectangular image region R can be calculated using the integral image I_c:

$$H_R = I_c(x_u, y_u) + I_c(x_b, y_b) - I_c(x_u, y_b) - I_c(x_b, y_u) \tag{5}$$

where (x_u, y_u) and (x_b, y_b) are respectively the upper left and bottom right corner coordinates of region R. Then, the output score of a linear classifier applied to any rectangular image region can be calculated as follows:

$$\frac{1}{N} H_R + W_m + b > 0 \tag{6}$$

Using integral images, the computational complexity of classifying any rectangular image region is reduced to 8 memory access, 10 additions and 2 products, independently of the size of rectangular region.

3 Experiments

We have evaluated the performance of our pixel-level categorization method on the Graz02 database [17]. The Graz02 database is a challenging database consisting on three categories (bikes, cars and people) where objects have an extreme variability in pose, orientation, lighting and different degrees of occlusion. The Graz02 annotation only provides a pixel segmentation mask for each image, so that, it is impossible to known how many object instances are present in the image. In consequence, to evaluate the performance of our pixel-level categorization method we use the pixel-based precision-recall curves as in [18]. Active pixels of the ground truth segmentation mask scorrectly categorized as object are counted as true positives, and as false negatives otherwise. Also, incorrectly classified background pixels of the ground truth segmentation mask are counted as false positives. Finally, we have taken the odd images as train and the even as test as in [18, 4]. However, due to the high variation we observed in the results depending on the train/test sets, we decided to also use random selection to split half of the images for train and half for test. The final result of a test when using the random sampling is the mean of a 1,000-repetitions experiment to ensure statistical invariance of the selected train/test sets.

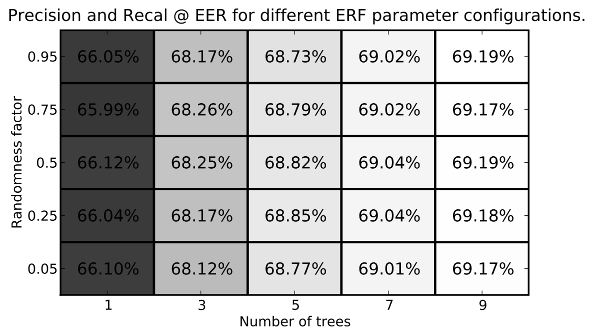

Fig. 3. Precision-recall at EER comparison for the ERF using different randomness factor and number of trees.

3.1 Parameter setting

The results were obtained using the same parameters in each experiment. The IHOG descriptors have been densely sampled each four pixels. Descriptors that have a low gradient magnitude before normalization are discarded as in [4]. Each IHOG descriptor is extracted from a 40×40 pixels patch and it has 8 orientation bins and 4 positional bins (i.e. a 32 dimensional descriptor). Therefore, as Graz02 images have a regular size of 640×480, a maximum of 16,500 descriptors are extracted per image. Then, bag of features histograms are computed accumulating the visual words that are inside a region of 80×80 pixels. Later, those histograms are categorized using a support vector machine. The SVM has been trained using logistic regression (LR-SVM)[19] with the LIBLINEAR software package [20]. Finally, the shown times results have been obtained using laptop with an Intel T7700 Core Duo CPU and 2Gb of RAM.

3.2 Parameters of the ERF

The performance of the ERF depends on the K-D tree randomness parameter and the amount of trees in the forest. Therefore, we wanted to evaluate which combination of those parameters gives better results for our categorization method. In Fig. 3 the mean precision-recall values at Equal Error Rate (EER) obtained for the different parameter combinations are shown. The results shows that the performance for the ERF largely depends on the amount of trees, while the randomness factor has little, if any, effect in the performance. For the remaining experiments, we have selected a randomness factor of 0.05 (i.e. a completely random forest) and 5 trees, which are a good compromise between performance and computational cost.

3.3 Comparison between HKM and ERF

To compare the performance of the HKM and the ERF, dense features have been computed for all the 450 training images, resulting in about 6,000,000 training features. For the HKM we have selected a branch factor of 10 as in [4] to generate a codebook with 200,000 visual words in average. For the ERF, using the parameters selected in the previous section, we have obtained a codebook of 150,000 visual words in average. We have done two different tests: the “Single” test only uses the images containing objects from the tested category (e.g. using the bike classifier only for the images where a bicycle can be actually found), and the “Multi” test uses all test images (e.g. using the bike classifier for all 450 test images). As can be seen in Table 1 the precision-recall values obtained at EER show that the ERF performs slightly better than the HKM, both in the “Single” and the “Multi” tests. In the “Multi” test we can see that, when 300 images not containing objects are added to the test set, precision decreases a reasonable 32%. Finally, Fig. 4 shows the precision-recall curves for the different categories of the Graz02 database. As can be seen, the ERF has a slightly better performance than the HKM.

3.4 Time cost evaluation

Finally, regarding the computational cost of the categorization approach, the average time needed to construct the HKM is of 5 minutes, while that of the ERF depends on the randomness factor and the number of trees used, ranging from 100 milliseconds for a completely random ERF with a single tree, to 12 minutes for a highly discriminative ERF with 9 trees. The cost of training a linear classifier using the LR-SVM is of about 2 minutes for the histograms generated with HKM codebook, and from 2 to 5 minutes for those generated with the ERF (it depends on the amount of trees of the ERF). In Fig. 4.d) we

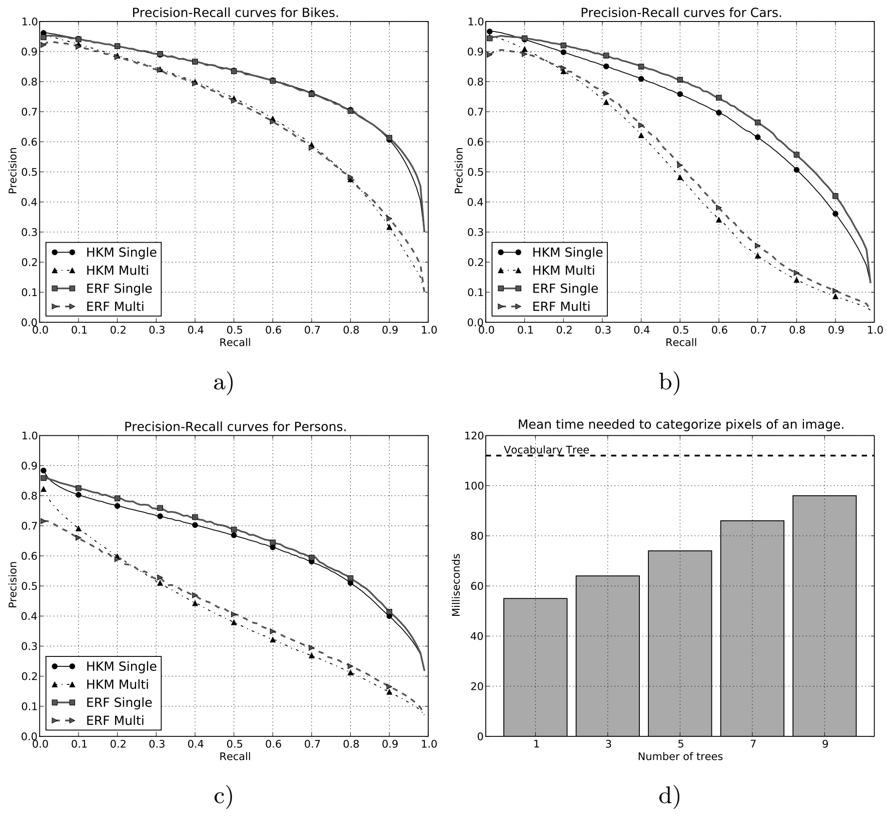

Fig. 4. Precision-recall curves obtained by the HKM and ERF codebooks using random sampling for the a) bikes, b) cars and c) persons categories of the Graz02 database. For each category, both methods have been evaluated using only the category images (Single) and using all testing images (Multi). In d) the mean time spent evaluating an image for the ERF and the HKM is shown.

can see the average time needed to categorize all image pixels using a HKM and ERF codebooks. Although being a bit slower in the training phase, the ERF is faster than the HKM in the categorization phase, where speed is truly essential. Using a ERF with 5 K-D trees, the whole scheme needs about 72.6 milliseconds to categorize an image, so that, we can process about 13 images per second.

4 Conclusions

In this paper we have presented an efficient method to assign a category label to each pixel of an image. Our main contribution is the introduction of integral linear classifier, which is used to bypass the accumulation step and directly obtain

Sampling	Test	Method	Bikes	Cars	Persons
Even/Pair	Single	HKM	**73.77% ± 0.20%**	63.90% ± 0.17%	61.80% ± 0.40%
		ERF	73.63% ± 0.32%	**65.68% ± 0.60%**	**62.59% ± 0.30%**
	Multi	HKM	**63.42% ± 0.24%**	47.09% ± 0.45%	44.36% ± 0.31%
		ERF	63.27% ± 0.32%	**47.62% ± 1.45%**	**46.34% ± 0.43%**
Random	Single	HKM	**74.33% ± 1.21%**	65.70% ± 1.45%	62.13% ± 1.30%
		ERF	74.17% ± 1.22%	**68.39% ± 1.44%**	**63.27% ± 1.38%**
	Multi	HKM	**64.55% ± 1.26%**	49.60% ± 1.61%	42.78% ± 1.86%
		ERF	63.98% ± 1.28%	**51.30% ± 2.03%**	**44.30% ± 2.12%**

Table 1. Comparison of the precision-recall values obtained at equal error rate on the Graz02 database both using odd/even and random train/test sampling. The categorization methods have been evaluated using only images that contain objects from the searched category in the "Single" test and using all the images in the "Multi" test.

the classification score for an arbitrary sub-window of the image. Besides, we have compared the performance of the Hierarchical K-Means (HKM) and Extremely Randomized Forest (ERF). The obtained results show that the ERF performs slightly better than the HKM and with a lower computational cost. We have shown that the proposed method with the ERF feature quantization approach is suitable for real-time applications. In future work, we plan to improve this method and use it in an online learning scheme with a mobile robot.

Acknowledgements

This work was supported by TIN 2006-15308-C02-02 project grant of the Ministry of Education of Spain, the CSD2007-00018 and the MIPRCV Consolider Ingenio 2010, the European Social Fund, the project grant 200450E550, FEDER funds, and the grant 2009-SGR-1434.

References

1. Lowe, D.: Distinctive image features from scale-invariant keypoints. Int. Journal of Computer Vision **60** (2004) 91–110
2. Csurka, G., Bray, C., Dance, C., Fan, L.: Visual categorization with bags of keypoints. In: Workshop on Stat. Learning in Computer Vision, ECCV. (2004) 1–22
3. Nister, D., Stewenius, H.: Scalable recognition with a vocabulary tree. In: Proc. of Computer Vision and Pattern Recognition. (2006) 2161–2168
4. Fulkerson, B., Vedaldi, A., Soatto, S.: Localizing objects with smart dictionaries. In: Proc. of European Conference on Computer Vision. (2008) 179–192
5. Sastre, R., Tuytelaars, T., Bascon, S.: Class Representative Visual Words for Category-Level Object Recognition. In: IbPRAI, 2009, Springer (2009)
6. Moosmann, F., Nowak, E., Jurie, F.: Randomized clustering forests for image classification. IEEE Trans. on Pat. Anal. and Machine Intel. **30** (2008) 1632–1646

7. Shotton, J., Johnson, M., Cipolla, R., Center, T., Kawasaki, J.: Semantic texton forests for image categorization and segmentation. In: Proc. of Computer Vision and Pattern Recognition. (2008) 1–8
8. Ramisa, A.: Localization and Object Recognition for Mobile Robots. PhD thesis, Universitat Autonoma de Barcelona (2009)
9. Lampert, C.H., Blaschko, M.B., Hofmann, T.: Beyond sliding windows: Object localization by efficient subwindow search. In: Proc. of Computer Vision and Pattern Recognition. (2008) 1–8
10. Nowak, E., Jurie, F., Triggs, B.: Sampling strategies for bag-of-features image classification. In: European Conference on Computer Vision. (2006) 490–503
11. Zhu, Q., Yeh, M.C., Cheng, K.T., Avidan, S.: Fast human detection using a cascade of histograms of oriented gradients. In: Proc. of Computer Vision and Pattern Recognition. (2006) 1491–1498
12. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Proc. of Computer Vision and Pattern Recognition. (2005) 886–893
13. Zhang, J., Marszalek, M., Lazebnik, S., Schmid, C.: Local features and kernels for classification of texture and object categories: A comprehensive study. Int. Journal of Computer Vision **73** (2007) 213–238
14. Viola, P., Jones, M.: Rapid object detection using a boosted cascade of simple features. In: Proc. of Computer Vision and Pattern Recognition. (2001) 511–518
15. Bay, H., Ess, A., Tuytelaars, T., Gool, L.V.: Surf: Speeded up robust features. Computer Vision and Image Understanding (CVIU) **110** (2008) 346–359
16. Porikli, F.: Integral histogram: A fast way to extract histograms in cartesian spaces. In: Proc. of Computer Vision and Pattern Recognition. (2005) 829–836
17. Opelt, A., Pinz, A., Fussenegger, M., Auer, P.: Generic object recognition with boosting. IEEE Trans. on Pat. Anal. and Machine Intel. **28** (2006) 416–431
18. Marszalek, M., Schmid, C.: Accurate object localization with shape masks. In: Proc. of Computer Vision and Pattern Recognition. (2007) 1–8
19. Lin, C.J., Weng, R.C., Keerthi, S.S.: Trust region newton methods for large-scale logistic regression. In: Int. Conf. on Machine Learning. (2007) 561–568
20. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: Liblinear: A library for large linear classification. J. Mach. Learn. Res. **9** (2008) 1871–1874