Fast shared boosting: application to large-scale visual concept detection

Hervé Le Borgne
CEA, LIST, Vision and Content Engineering Laboratory
18 route du Panorama, BP6, Fontenay-aux-Roses, F-92265 France;
herve.le-borgne@cea.fr

Nicolas Honnorat
MAS Laboratory, Ecole Centrale de Paris,
Grande Voie des Vignes, Chatenay-Malabry, France
nicolas.honnorat@ecp.fr

Abstract

This work addresses the problem of large-scale visual concept detection. Visual concepts are usually learned from an annotated image or video database with a machine learning algorithm, posing this problem as a multiclass supervised learning task. Some practical issues appear when the number of concept grows, in particular when one aims at developing applications for real users, restricting the constraints in terms of available memory and computing time (both for learning and testing). To cope with these issues, we propose in this article to use a multiclass boosting with feature sharing algorithm and reduce its computational complexity with a set of efficient improvements. This makes our algorithm able to handle a problem of classification with many classes in a reasonable time. The relevance of our algorithm is evaluated in the context of information retrieval, on the benchmark proposed into the ImageCLEF international evaluation campaign and shows competitive results.

1. Introduction

Detecting a concept into an image consists of assigning one or several labels to an image to indicate its presence. The term *concept* can recover at least two types of entities, namely the object and the scenes. In the case of objects, the concept detection consists of identifying the presence of at least one instance of the considered category in the image (e.g there is “a car” in this image). For the scene, the concept detections rather relates to a general atmosphere of the image (e.g detecting whether it represents an indoor or an outdoor scene, during day or night time, whether it is a urban or nature scene, representing a forest, a mountain or a beach...). In both cases, we are interested into the general category of the concept, not the particular instance that represent it in the considered image (e.g we are interested into detecting the presence of a car, not “the grey car of M. Smith”). This problem has interested the vision community for many years and numbers of approaches have been proposed (e.g [4, 9, 18, 13, 17, 10]). The problem is usually posed as a set of binary classification tasks, composed of two main steps, namely visual information (feature) extraction followed by supervised learning of each concept. The learned classifier is then applied to several locations and scales within a testing image and returns a confidence score on the presence of the considered concept (in case of scenes, it can usually be applied to a very restricted number of location, even eventually once to the global image [12]). The first step of visual information extraction is out of the scope of this paper and we refer to [10] for a comprehensive presentation of the process of local feature extraction and to [8] for examples of global features that are used in the following work.

In practice, one want to be able to detect several concepts at the same time. There are two well-known schemes to deal with such a multi-class classification problem. Considering a database of N images containing from 0 to C concepts, the “one versus all” approach consists of leaning C binary classifiers considering one concept $i = 1, \dots, N$ as positive and all others as negative. The “one against one” approach consists of learning $C(C-1)/2$ classifiers corresponding to all possible binary problems then combining their outputs with a majority vote. These two approaches are two extreme cases of the ECOC scheme [3]. In that case, each class is assigned to a unique binary string (named a codeword). During training for an example from class i, the desired outputs of these P binary functions are specified by the codeword for class i. New values of a test sample x are classified by evaluating each of the P binary functions to generate a p-bit string s. This string is then compared to each of the C codewords, and x is assigned to the class whose codeword is closest, according to some distance measure (such as a hamming distance), to the generated string s. "One versus all" is then the special case with codeword of unitary norm ($\|s\| = 1$) and "one against one" with all $C(C-1)/2$ codewords such that $\|s\| = 2$.

A problem with these approaches to multi-concept detection (i.e multiclass classification) is that they bear poorly scalability, in particular in terms of computational complexity and memory needs. In the following of this section we present an alternative approach based on boosting that partially respond to this problem. However, the learning complexity of this program is still high and the learning duration becomes prohibitive when the number of classes increases. Then in section 2 we present our method to cope with this computational complexity. The main idea is to reduce the complexity by exploring a limited part of the possible parameter space of the algorithm. An experimental validation of our approach is proposed in section 3 and we conclude in section 4.

1.1. Boosting and the multi-class problem

A classifier is qualified as *weak* when it performs only slightly better than random guessing. Boosting consists of building a strong classifier by iteratively adding weak classifiers that focus on the harder-to-learn parts of the learning samples distribution [14, 5]. An example of weak classifier is a regression "stump" of the form $h_m(v) = a\delta(v^f > \theta) + b\delta(v^f \leq \theta)$, where v^f is the f'th component of the feature vector v, θ is a threshold , δ the indicator function and (a, b) are the regression parameters. Many algorithms exists to build a strong classifier $H(v) = \sum h_m(v)$ and we refer to [6] for an exhaustive presentation of the boosting principles and algorithms.

To manage the multi-class problem, an original solution was proposed in [16], consisting of finding common features that can be shared across the classes, and training jointly the classifiers. More precisely, in order to learn a strong classifier $H(v,c) = \sum_{m=1}^{M} h_m(v,c)$, it solves at each iteration m the following least squares problem:

$$J_{wse} = \sum_{c=1}^{C} \sum_{i=1}^{N} w_i^c \left(z_i^c - h_m(v_i, c)\right)^2 \qquad (1)$$

where $w_i^c = e^{-z_i^c H(v_i,c)}$ are the weights for sample i and for the classifier for class c, and z_i^c are the labels (±1)for sample i for class c. Noting s^* the subset of classes that are

Algorithm 1 Multiclass boosting with feature sharing [16]. N is the number of training samples, M the number of boosting rounds and C the number of classes. In the simplest (but slowest) version of the algorithm, it explores all the possible subsets at each rounds: $s \in \mathcal{S} = 1, 2, \ldots, 2^C - 1$. v_i^f is the f'th feature of the i'th component, $z_i^c = \pm1$ are the labels for sample i related to class c, and w_i^c are the unnormalized sample weights that gives their relative importance into the training sample.

```
I. Initialize the weights $w_i^c = 1$ and set
     $H(v, c) = 0, i = 1..N, c = 1..C$
II. Repeat for $m = 1, 2, \ldots, M$
  (A) Repeat for the possible features $f = 1 \ldots D$
    (1) Repeat for the possible thresholds $t = 1 \ldots N_\theta$
      (a) Precompute parameters for efficient computation
          of (i) and (ii)
      (b) Repeat for the possible subsets of classes $s \in \mathcal{S}$
        (i) Fit shared stump $h_m^s(v_i, c)$
        (ii) Evaluate the error $J_{wse}(s)$
  (B) Find best subset $s^* = \arg\min_s J_{wse}(s)$
  (C) Update the class estimates :
        $H(v_i, c) := H(v_i, c) + h_m^{s^*}(v_i, c)$
  (D) Update the weights : $w_i^c := w_i^c e^{-z_i^c h_m^{s^*}(v,c)}$
```

shared, a stump is now defined as:

$$h_m^{s^*}\left(v_i^f, c\right) = \begin{cases} a & \text{if } v_i^f > \theta \text{ and } c \in s^* \\ b & \text{if } v_i^f \leq \theta \text{ and } c \in s^* \\ k^c & \text{if } c \notin s^* \end{cases}$$

where a, b and k^c are constants that analytically results from the minimization of the weighted square error (1). The strong classifier is then determined using algorithm 1. In its simplest version, it is quite slow since it requires fitting shared stumps and evaluate the error for all the $2^C - 1$ possible subsets of classes. The authors proposed to first select the class that has the best reduction of the error, then select the second class that has the best error reduction jointly with the previously selected class, then continuing this process by adding each time the best class among those not yet selected. This heuristic drastically reduces the complexity of the algorithm since the size of the exploration space of the possible subsets decreases from $2^C - 1$ to C^2. The computation of the shared regression stumps is accelerated thanks to a propagation of the computations from the leaves of the exploration graph to higher nodes (see [16] for details of the step (a) in Alg.1), but the parameters must still be estimated for each of the D dimensions of the features and N_θ possible thresholds θ. In the end, the total complexity of the algorithm for N training samples labeled on C classes is:

$$\mathcal{C}^0 = O\left(MDN_\theta\left[NC + C^3\right]\right) \qquad (2)$$

Another algorithm of shared boosting was presenting in [15], showing similar limitations in term of learning complexity.

Algorithm 2 Fast multiclass boosting with feature sharing. see algorithm 1 for the notations and text for details.

```
I. Initialize the weights $w_i^c = 1$ and set
   $H(v\ ,\ c) = 0, i = 1..N, c = 1..C$
II. Repeat for $m = 1, 2, \dots, M$
  $(A)$ Repeat for $i = 1 \dots N_f$
    (a) Choose one feature $f^\star$ in $1 \dots D$
    (b) Choose one threshold $t^\star$ in $1 \dots N_\theta$
    (c) Precompute parameters for efficient computation
        of (i) and (ii)
    (d) Repeat for $N_k$ possible subsets of classes $s \in \mathcal{S}$
      (i) Fit shared stump $h_m^s(v_i, c)$
      (ii) Evaluate the error $J_{wse}(f^\star, t^\star, s)$
  $(B)$ Find best subset $s^\star = \arg\min_s J_{wse}(f^\star, t^\star, s)$
  $(C)$ Update the class estimates :
      $H(v_i, c) := H(v_i, c) + h_m^{s^\star}(v_i, c)$
  $(D)$ Update the weights : $w_i^c := w_i^c e^{-z_i^c h_m^{s^\star}(v,c)}$
```

2. Fast learning for multi-class boosting

According to (2), the shared multi-class boosting algorithm exhibits a cubic growth rate with the number of classes, which makes it unsuited to this scalability. This section presents an extension of a previous work in which we proposed several strategies to overcome this limitation [7]. The main idea is to reduce the complexity by exploring a limited part of the possible space (see Alg. 2).

Instead of trying "all possible settings" for some parameters as in the original algorithm (Alg. 1), we either determine a reasonable value or, when no a priori choice seems better than an other, we pick a limited set of random values, compute a performance criterion and finally keep the best one. Choosing a random value can be considered as reasonable in some cases, as L. Breiman shown with random forest [1]. For instance, choosing randomly the threshold $t^\star$ of the stump (step (b) of Alg. 2) does not deteriorate the performances significantly. On the contrary, the choice of the feature number $f^\star$ (step (a) of Alg. 2) can not be done fully randomly. Hence, we chose to use the Fisher criterion on a random sharing of the class space, repeating this process N_f times. Fist, a subset of class s is chosen randomly. Let note $\overline{s}$ the complement of s, $m_s(f)$ the mean value of the image feature vectors v_i^f that belongs to one of classes of s and $v_s(f)$ their variance. The final choice is then :

$$f^\star = \min_{f \in 1 \dots D} \frac{\left[m_s(f) - m_{\overline{s}}(f)\right]^2}{v_s(f) + v_{\overline{s}}(f)} \quad (3)$$

It requires to pre-compute the sum (and its square) for each feature only once, then the mean and variance can be computed faster at each of the N_f random choices. To determine the best stump/weak classifier, we also pick randomly N_k possible subsets of classes $s \in \mathcal{S}$ (step (d) of Alg. 2) and retain the combination (feature number $f^\star$, threshold $t^\star$, class subset s) that has the lowest cost $J_{wse}(f^\star, t^\star, s)$ among the $N_f \times N_k$ possibilities. As in Alg. 1, we update the class estimate by adding the chosen weak classifier (stump) to the strong classifier and finally update the sample weights before the next step. Considering that the complexity of step (step (c) of Alg. 2) is $O(NC)$, the final complexity of our algorithm is thus:

$$\mathcal{C}^1 = O\left(M\left[NDC + N_f[\alpha D + \epsilon + NC + \beta N_k]\right]\right) \quad (4)$$

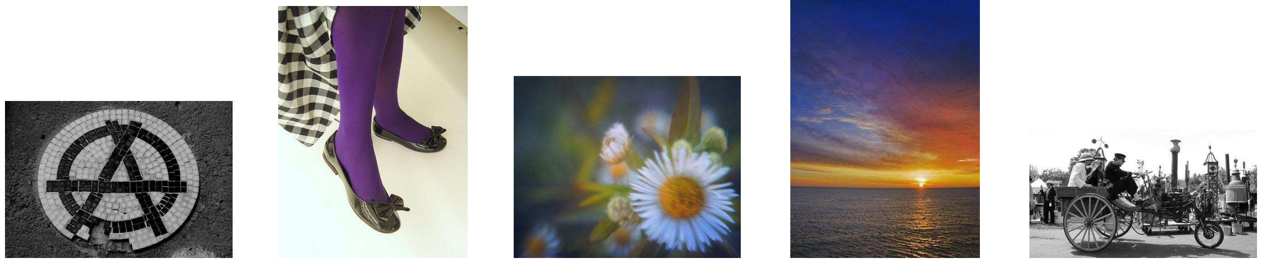
where α, ϵ and β are some constants corresponding to the unitary cost of respectively steps (a), (b) and (d). One could be surprised that the finally choosing share set s^* may be different from that one that has been used to compute the Fisher criterion. In fact the cost of fitting a shared step is much lower than the one to compute the Fisher criterion ($\beta << \alpha$) thus it makes sense to do it. Moreover, in practice, we noticed it improves the performances of the algorithm.

3. Experimental validation

We are interested in the problem of visual concept detection. For the experiments presented here we used the ImageCLEF VCDT experimental paradigm. The vcdt08 database [2] contains 1827 images for training and 1000 images for testing, with 17 concepts to detect. The vcdt09 database [11] has 53 concept categories, 5000 learning images and 3000 testing ones. The categories are not exclusive, i.e one image can contain several of the concepts (see figure 1).

The global feature we use are described in [8] and consist of a colour histogram taking into account the spatial arrangement of pixels (*cime* descriptor, size 64) and a combination of texture and colour descriptors (*tlep*, size 576). We also considered the use of local descriptors by computing SIFT features then a classical K-means to create a visual dictionary [18], resulting into bag-of-feature (*BoF* descriptor, size $K = 5000$). Points of interest are detected with a Harris-Laplace detector in addition to 1000 points randomly distributed over the image. This last refinement aims to better describe the concept that concerns the wholes images, their general aspect or their atmosphere.

The performance of the algorithm is measured with the equal error rate (EER) that indicates that the proportion of false acceptances is equal to the proportion of false rejections. The lower the equal error rate value, the higher the accuracy of the system. All experiments were conducted on one core of a Intel Q9400 CPU @ 2.66 GHz with 4 GB memory.



Building Sights			Landscape Nature	City life
No Visual Season	No Visual Season	No Visual Season	No Visual Season	No Visual Season
Outdoor	Indoor	Outdoor	Outdoor	Outdoor
		Plants	Sky	
		Flowers	Clouds	
			Water	Vehicle
Day	No Visual Time	Day	Day	Day
			Sunset Sunrise	
Neutral Illumination	Neutral Illumination	Neutral Illumination	Neutral Illumination	Neutral Illumination
No Blur	No Blur	Out of focus	No Blur	No Blur
No Persons	No Persons	No Persons	No Persons	Small Group
			Aesthetic Impression	
			Overall Quality	
			Fancy	

Figure 1. Example of pictures and their annotated concepts.

3.1. Exp 1: parameter choice

In Alg. 2, there are two parameters related to random choices: N_f is the number of random choices on the feature number and N_k is the number of random choices of class sharing subsets to fit the stump and compute the total cost. From equation (4), one can see they influence the complexity, but can also influence the performances of the algorithm, this experiment studying how.

In practice we used the vcdt08 benchmark ($N = 1827$, $C = 17$) with the global feature only ($D = 640$) and the number of iteration was fixed to $M = 100$. We conducted the experiment ten times (cross validation) and reported the average learning time in table 1 and the average EER in table 2 for different values of N_f and N_k.

The same experiment was conducted with the original algorithm of [16] that obtained an EER of 26.0 in 73.2 seconds. With the parameters $N_f = 50$ and $N_k = 10^4$ we obtain the same performances five times faster (15.4 sec). Moreover, from equation (2) and (4) we induce that this difference would grow with the number of concepts (C) to detect, showing the interest of our algorithm for large-scale visual concept detection (see also section 3.4). One can also notice that the couple $(N_f = 50, N_k = 10^4)$ seems sufficient on this experiment since multiplying them by ten leads to similar results. It may be different with larger feature descriptors (i.e when D increases).

	$N_f = 5$	50	500
$N_k = 10^2$	5.2	6.7	22
$N_k = 10^3$	5.3	8.0	28.6
$N_k = 10^4$	6.1	15.4	105.2
$N_k = 10^5$	13.7	92.6	877
[16] with greedy heuristic			73.2
[16] without heuristic			11966

Table 1. Average learning time (second / experiment) averaged across 10 experiments, on vcdt08 for different values of N_f and N_k.

	$N_f = 5$	$N_f = 50$	$N_f = 500$
$N_k = 10^2$	27.8 ± 1.4	26.8 ± 1.3	26.5 ± 1.1
$N_k = 10^3$	27.1 ± 1.2	26.7 ± 1.2	25.9 ± 1.2
$N_k = 10^4$	26.5 ± 1.3	26.0 ± 1.1	25.7 ± 0.9
$N_k = 10^5$	26.7 ± 1.1	25.7 ± 1.0	25.7 ± 1.0
[16] with greedy heuristic			26.0
[16] without heuristic			25.8

Table 2. Average EER (multiplied by 100) on vcdt08 for different values of N_f and N_k (the lower the better).

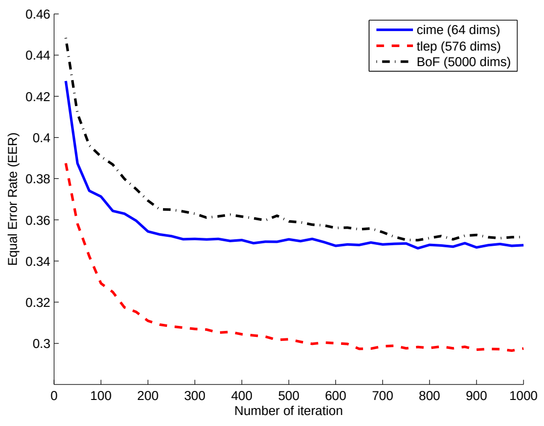


Figure 2. EER on vcdt09 according to different descriptor sizes and number of iteration.

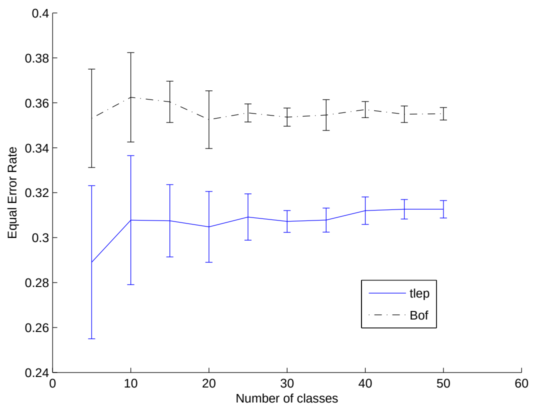


Figure 3. EER on vcdt09 according to the number of classes considered, for two descriptors.

3.2. Exp 2: convergence

This experiment aims at determining the number of iteration necessary to reach the convergence. It was conducted on the vcdt2009 benchmark ($N = 5000$, $C = 53$) with features of different sizes D. We fixed the parameters of the algorithm to ($N_f = 50$ and $N_k = 10^4$). The number of iteration varied from $M = 25$ to $M = 1000$ and we computed the EER every 25 iterations. Results are reported on Fig. 2.

With the CIME descriptor ($D_{cime} = 64$) the algorithm converges around 200 iterations, that is to say between two and three D_{cime}. We see similar behavior with the other global descriptor TLEP, for which the algorithm converges in about D_{tlep} iterations. For the local descriptor the convergence is relatively even faster since the algorithm is converging around 700 iterations with a descriptor of size $D_{BoF} = 5000$. However the raw performance with this descriptor is quite disappointing as it remains below the others. As mentioned below, the scope of this paper focuses on the learning method and does not concerns the quality of the descriptors. It seems nevertheless obvious that better local descriptors needs to be used for a powerful application. The experiment still shows that our algorithm converges in a few hundreds of iterations even when the size reaches several thousands descriptors.

3.3. Exp 3: influence of the number of classes

This experiment aims to studying the influence of the number of classes. For two different descriptors, tlep and bag-of-features, we selected ten times a restricted number of classes among the 53 available in vcdt09. The choice was made randomly. Then we learned and tested on this set of classes with the parameters $(N_f = 50, N_k = 10^4)$ and computed the average and standard deviation of the equal error rate resulting from these experiments (see figure 3).

The decay of the standard deviation simply reflects the lesser variety into the individual class results when more classes are took into account. The average EER value is almost the same whatever the value of the number of classes, showing the remarkable stability of the algorithm even when the number of classes grows.

	tlep	BoF
our method	31.1	36.94
one-versus-all	46.05	48.64

Table 3. EER ($\times 100$) on vcdt09 for our method and one-versus-all approach.

3.4. Exp 4: interested of sharing classes

In this experiment, we compared our algorithm to a one-versus-all approach on the same feature (since the goal is to compare the learning algorithm, independently of the used features). We thus run our algorithm in a one-versus-all approach, with he parameters $(N_f = 50, N_k = 10^4)$ and $M = 200$ iterations for each classifier. The results (table 3) are drastically in advantage of our method (same parameters but $M = 200$ for all the classes, making it much faster).

In fact, such a difference can be explained in the context of concept detection. Indeed, a given image can contain several concepts, and some of them can overlap. Hence, during the learning of concept A, images containing concept B can be both in the positive and the negative set. Excluding such images from the learning set would lead to reduce the number of learning images for concept A. The problem is that the more the number of classes to learn (with overlapped concepts) the fewer the number of available images to learn each concept. Sharing classes thus becomes increasingly relevant in the case of a large number of (overlapping) concepts to detect.

4. Conclusion

We presented a fast learning algorithm based on shared boosting. We reduced the complexity by limiting the exploration of the parameter space. The experimental results on the ImageCLEF benchmarks 2008 and 2009 showed the efficiency of our method in the context of visual concept detection. It is much faster than other learning algorithms while maintaining a similar or better accuracy. In particular, in the context of visual concept detection for which several concepts can be present within an image, our approach allows to keep a significant quantity of learning images. Its complexity grows linearly with the number of classes to detect, making it suitable to tackle with large-scale visual concept detection problems.

As future work, we are interested into studying the properties of the algorithm on sparse data.

5. Acknowledgment

We acknowledge support from the ANR (project Yoji) and the DGCIS for funding us through the regional business cluster Cap Digital (projects Géorama, Mediatic, POPS, and Roméo).

References

[1] L. Breiman. Random forests. *Machine Learning*, 1(45):5–32, 2001.

[2] T. Deselaers and A. Hanbury. The visual concept detection task in imageclef 2008. In *CLEF working notes*, pages 531–538, Aarhus, Denmark, 2008.

[3] Dietterich and Bakiri. Solving multiclass learning problems via ecocs. *Journal of AI Research*, (2):263–286, 1995.

[4] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *CVPR*, 2003.

[5] Y. Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, (121):256–285, 1995.

[6] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *Annals of Statistics*, (28):337–407, 2000.

[7] N. Honnorat and H. Le Borgne. Accélérer le boosting avec partage de caractéristiques. In *CORESA*, pages 203–208, Toulouse, France, 2009.

[8] M. Joint, P.-A. Moëllic, P. Hède, and P. Adam. Piria: a general tool for indexing, search and retrieval of multimedia content. In *SPIE ElectronicImaging*, San Jose, California, USA, 2004.

[9] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2006.

[10] M. Marszałek, C. Schmid, H. Harzallah, and J. van de Weijer. Learning object representations for visual object class recognition. In *Visual Recognition Challenge workshop, in conjunction with ICCV*, 2007.

[11] S. Nowak and P. Dunker. Overview of the clef 2009 large scale visual concept detection and annotation task. In *CLEF working notes*, Corfu, Greece, 2009.

[12] A. Oliva and A. Torralba. Modeling the shape of the scene: a holistic representation of the spatial envelope. *International Journal of Computer Vision*, 3(42):145–175, 2001.

[13] F. Perronnin. Universal and adapted vocabularies for generic visual categorization. *IEEE T. on Pattern Analysis and Machine Intelligence*, 7(30):1243–1256, 2008.

[14] R. Schapire. Strength of weak learnability. *Machine Learning*, (5):197–227, 1990.

[15] J. Shotton, J. Winn, and C. Rother. A. criminisi.textonboost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *International Journal of Computer Vision*, 1(81):2–23, 2009.

[16] A. Torralba, K. P. Murphy, and W. Freeman. Sharing visual features for multiclass and multiview object detection. *IEEE T. on Pattern Analysis and Machine Intelligence*, 5(29):854–869, 2007.

[17] J. C. van Gemert, J. M. Geusebroek, C. J. Veenman, and A. W. M. Smeulders. Kernel codebooks for scene categorization. In *European Conference on Computer Vision*, 2008.

[18] J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid. Local features and kernels for classication of texture and object categories: a comprehensive study. *International Journal of Computer Vision*, 2(73):213–238, 2007.