EFFICIENT BAG-OF-FEATURE KERNEL REPRESENTATION FOR IMAGE SIMILARITY SEARCH

F. Precioso[1], M. Cord[1], D. Gorisse[2], N. Thome[1]

[1]LIP6, UMR CNRS 7606, UPMC - Sorbonne Universites
4 place Jussieu, 75005 Paris, France, lastname@lip6.fr
[2]Yakaz
34 rue de clery, 75002 Paris, Paris, France, david@yakaz.com

ABSTRACT

Although "Bag-of-Features" image models have shown very good potential for object matching and image retrieval, such a complex data representation requires computationally expensive similarity measure evaluation. In this paper, we propose a framework unifying dictionary-based and kernel-based similarity functions that highlights the tradeoff between powerful data representation and efficient similarity computation. On the basis of this formalism, we propose a new kernel-based similarity approach for Bag-of-Feature descriptions. We introduce a method for fast similarity search in large image databases. The conducted experiments prove that our approach is very competitive among State-of-the-art methods for similarity retrieval tasks.

Index Terms— Bag-of-Features, similarity, kernels, Bag-of-Words, visual dictionary, image retrieval

1. INTRODUCTION

Complex descriptions representing each image by one set of vectors, called Bag-of-Features (BoF), showed to be powerful for image similarity search in image and video databases [1, 2]. Designing powerful similarities between images and ranking schemes on unordered sets of local features is a very challenging task.

As an alternative to standard voting scheme [1], Bag-of-Words models (BoW) based on visual dictionary computation have been developed both to overcome the amount of data to be processed [3]. Recently, alternatives very successful in categorization tasks, based on a generative coding of BoW, have been considered [4, 5].

Another popular alternative to the voting scheme that provides a similarity function on BoF are kernel on bags. Grauman and Darrell [6] proposed the pyramid matching kernel to compute the similarity between sets of PoIs. Their scheme provides good results, but is restricted to kernels which can be explicitly formulated as a dot product in an induced space. Other approaches attempting to go beyond explicit feature mappings belong to the class of kernels on sets of vectors, derived from kernels on vectors [7, 8]. These kernels have been successfully adapted for object retrieval [9, 10, 11]. The major advantage of such a framework is that it becomes possible to use unspecified kernel functions. However, the similarity computation rapidly becomes intractable when the size of the database increases.

In this paper, we propose a new approach based on kernel on bags, combining accuracy of kernel-based similarities and efficiency of dictionary-based approaches. We validate the relevance of the proposed kernel on two image datasets for semantical similarity search and near duplicate detection. Our formalism encompasses in a unified framework dictionary-based and kernel on set-based similarity functions. We evaluate the performances of our system both in terms of retrieval accuracy and computational efficiency, and compare our new kernel with State-of-the-art techniques (BoW approaches, kernel on bags, and super-vector coding [5]).

2. BOF SIMILARITY FOR IMAGE SEARCH

Let any image I_j be represented by a feature bag B_j composed of s unordered feature vectors $\mathbf{b}_{sj} \in \mathbb{R}^p$: $B_j = \{\mathbf{b}_{sj}\}_s$. Let $\mathcal{B}$ be the database of images and $\mathcal{F}$ the database of feature vectors $\mathbf{b}$. Let I_q be the query image represented by the bag $B_q = \{\mathbf{b}_{rq}\}_r$.

In content-based image retrieval, if $score_j = score(B_j, B_q)$ denotes the similarity score between the query image I_q and the image I_j, the search-by-similarity process in $\mathcal{B}$ aims at ranking the database thanks to the scores: $\underset{B_j \in \mathcal{B}}{Rank}(score_j)$. The scores $score_j$ considered here may be written as:

$$score_j = \sum_{\mathbf{b}_{rq} \in B_q} \sum_{\mathbf{b}_{sj} \in B_j} f(\mathbf{b}_{rq}, \mathbf{b}_{sj})$$

where f is a similarity function that reflects the similarity between two local descriptors $\mathbf{b}_{rq}$ and $\mathbf{b}_{sj}$.

If the similarity function f is a kernel function (let us then denote it k), $score_j$ is then also a kernel corresponding to the well-know class of kernels on bags proposed in [7, 8]:

$$score_j = K(B_q, B_j) = \sum_{\mathbf{b}_{rj} \in B_j} \sum_{\mathbf{b}_{sq} \in B_j} k(\mathbf{b}_{rj}, \mathbf{b}_{sq}) \qquad (1)$$

k is called the *minor kernel* on local descriptors $\mathbf{b}_{rj}$.

This means that it exists an embedding function Φ (possibly implicit) which maps any bag B_j to a vector $\Phi(B_j)$ in a Hilbert space such that $score_j$ is then defined by a dot product in the induced space:

$$score_j(B_q) = < \Phi(B_q), \Phi(B_j) > \qquad (2)$$

This class of kernels has been later extended [9, 10] by raising the minor kernel k to the power of q in order to increase the gap between high and low matches:

$$K^q_{power}(B_j, B_q) = \sum_{\mathbf{b}_{sj} \in B_j} \sum_{\mathbf{b}_{rq} \in B_q} k^q(\mathbf{b}_{sj}, \mathbf{b}_{rq}) \qquad (3)$$

However, computing this score between the query image I_q and all the images from the database $\mathcal{F}$ is clearly computationally expensive if one thinks to the amount of combinations between all local descriptors of I_q and all local descriptors of all other images.

3. OUR ULTRA FAST KERNEL SCHEME

If some strategies have been developed to build kernels on bags using explicit mapping Φ [6, 12], we propose here a kernel on bags framework, called $UFast_K$, compliant with implicit mappings while remaining computationally tractable.

More precisely, our kernel, generalizing eq.(1), is defined by:

$$UFast_K(B_j, B_q) = \sum_{\mathbf{b}_{rq} \in B_q} \sum_{\mathbf{b}_{sj} \in B_j} k(\mathbf{b}_{rq}, \mathbf{b}_{sj}) f_Q(\mathbf{b}_{rq}, \mathbf{b}_{sj}) \tag{4}$$

where f_Q is a function associated to a quantifier Q of the local descriptor space. By this way, many irrelevant matches in eq.(4) may be discarded, so that the whole similarity get closer to the exact match kernel between sets of points. This scheme exploits the fact that a large number of f_Q values are zero over the dataset $\mathcal{F}$ to optimize the computation of the similarity $UFast_K$ using indexing structures.

We use here the strategy developed for BoW: the quantifier Q maps a local descriptor $\mathbf{x} \in \mathbb{R}^d$ to the integer index $m \in [\![1, |\mathcal{W}|]\!]$ of the closest codeword $\mathbf{w}_m$, among the codebook $\mathcal{W}$, to the descriptor $\mathbf{x}$. f_Q is defined as the indicator function:

$$f_Q(\mathbf{x}, \mathbf{y}) = \mathbb{1}_{Q(\mathbf{x})=Q(\mathbf{y})} \tag{5}$$

The algorithm 1 details the steps of our similarity search over $\mathcal{F}$.

Algorithm 1

1: Database image similarity $score_j$ are initialized to 0.
2: **for all** query image feature $\mathbf{b}_{rq}$ **do**
3: a fast f_Q search on all features of the dataset $\mathcal{F}$ quantized in the same cluster as $\mathbf{b}_{rq}$
4: **for all** I_j such that a feature $\mathbf{b}_{sj}$ has been found **do**
5: increase the similarity by $score_j := score_j + k(\mathbf{b}_{sj}, \mathbf{b}_{rq})$
6: **end for**
7: **end for**

As one can see in step 3 of the algorithm, fast strategies may be implemented to boost the search using indexing structure. Inverted files strategies may be used with classical clustering techniques like K-means if the data are sparse [13]. Among all techniques, we have focused in our work on the powerful Locality-Sensitive Hashing scheme (LSH) which achieves this search step with a computational complexity sublinear with the amount of local descriptors in $\mathcal{F}$. A cluster (or a bucket) of LSH is hence defined by the intersection of L hash tables, each one concatenating H hash functions.

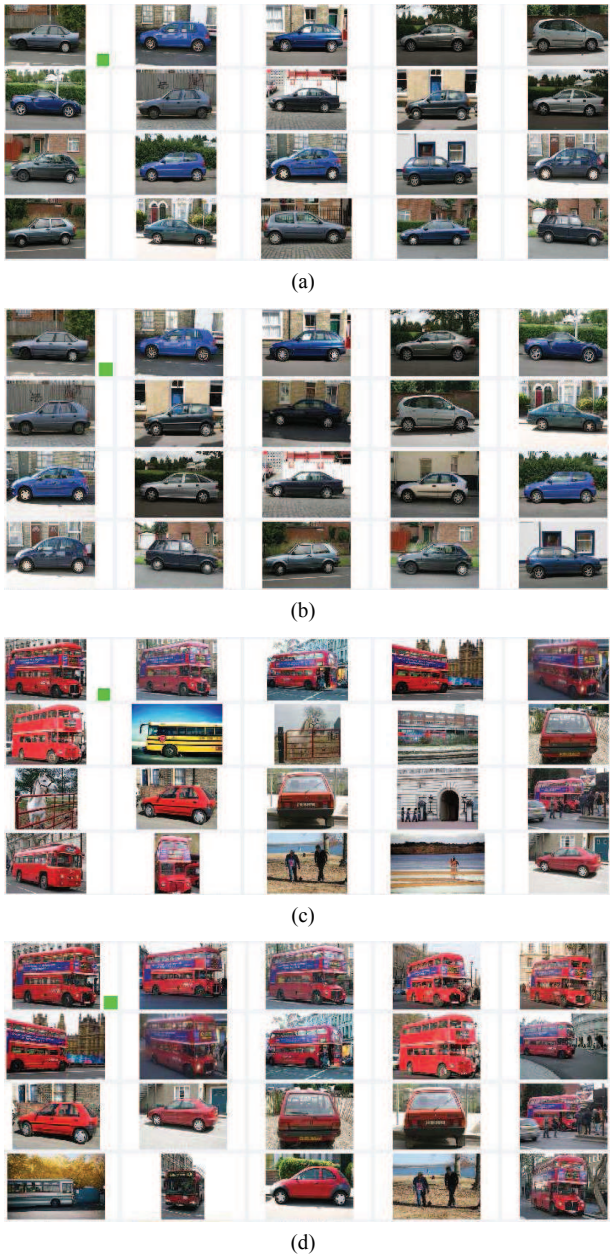
The breakdown in complexity of $UFast_K$ compared to exhaustive kernels on bags lies in the restriction of all combinations of minor kernel computations inside the M clusters (or buckets in LSH) which can be emphasized by rewriting $UFast_K$:

$$UFast_K(B_j, B_q) = \sum_{m=1}^{M} \kappa(mini_m(B_j), mini_m(B_q)) \tag{6}$$

where $mini_m(B_j) = \{b_{sj} \in B_j | Q(b_{sj}) = m\}$.

Another interest of our $UFast_K$ is to reduce the amount of noise by taking into account only neighbors of each query local descriptor while the exhaustive kernel on bags accumulates small similarity values corresponding to very dissimilar PoIs between compared BoFs. This is illustrated on Fig.(1). Indeed, for the class *cars*, almost all objects have the same pose and they cover most of the image, thus most local descriptors are semantically relevant. In such a case, similarity search results are equivalent for exhaustive kernel on bags, Fig.(1(a)), and for our Ultra Fast kernel scheme, Fig.(1(b)). Our scheme povides thus highly reduced computational complexity for no accuracy loss. If the class of objects contains more appearance variations, as for *bus*, our Ultra Fast kernel provides even better results than exhaustive kernel, comparing results of Fig.(1(d)) with Fig.(1(c)), because the amount of semantically useless local descriptors is reduced by the clustering defining $mini_m$ representation of BoF.

In our framework, any minor kernels k in eq.(4) may be used. We focus on corresponding κ functions (eq.(6)) in the next section.



(a) (b) (c) (d)

Fig. 1. VOC2006: 5 300 images, 260 PoI / image, 1.4 Million of color SIFT (384 dimensions)

4. CHOICES FOR κ

We introduce in our formalism different choices for κ leading to def ne classical BoF similarity measures. We thus encompass in one scheme all BoF similarities, including our Ultra Fast scheme, highlighting computational complexity and accuracy differences.

4.1. linear kernel

The standard similarity measure between image I_j and query image I_q, when they are represented by Bow vectors, is the inner product of these vectors. By def ning the kernel on mini Bags:

$$\kappa(mini_m(B_j), mini_m(B_q)) = |mini_m(B_j)|.|mini_m(B_q)| \quad (7)$$

in our formalism (eq.(6)), $UFast_K$ corresponds to the standard inner product between BoW vectors.

4.2. Shifted-linear kernel

In order to ref ne the similarity between BoWs and not anymore counting only local descriptors falling into the same cluster, one can account for the position in the feature space of each local descriptor with respect to the center of its assigned cluster:

$$\kappa(mini_m(B_q), mini_m(B_j)) = \sum_{\mathbf{b}_{rq} \in B_q} \sum_{\mathbf{b}_{sj} \in B_j} < \mathbf{b}_{rq} - \mathbf{w}_m, \mathbf{b}_{sj} - \mathbf{w}_m > \quad (8)$$

We show in appendix A that this representation corresponds to the Vector of Locally Aggregated Descriptors (VLAD) proposed by Jegou *et al.* [5].

4.3. χ^2-gaussian kernel

We propose in this paper to preserve all the accuracy of considering a χ^2-gaussian kernel as minor kernel k.

$$\kappa(mini_m(B_q), mini_m(B_j)) = \sum_{\mathbf{b}_{rq} \in B_q} \sum_{\mathbf{b}_{sj} \in B_j} k^q(\mathbf{b}_{rq}, \mathbf{b}_{sj}) f_{LSH}(\mathbf{b}_{rq}, \mathbf{b}_{sj}) \quad (9)$$

Based on the properties of LSH, we can easily prove that our $UFast_K$, with the above κ function, is indeed a Mercer kernel by demonstrating the positive def nite property of its Gram matrix $\mathcal{K}_{ij} := (k^q(\mathbf{b}_{ri}, \mathbf{b}_{sj}) f_{LSH}(\mathbf{b}_{ri}, \mathbf{b}_{sj}))_{ij}$.

The computational complexity of kernel on bags given in eq.(1) is $O(n^2)$ with n the number of local descriptors in each image while the complexity of our Ultra Fast Scheme is $O((\frac{n}{M})^2 M) = O(\frac{n^2}{M})$ with M the number of clusters. Classically M and n are of the same magnitude order, leading to a complexity of approximately $O(n)$.

5. EXPERIMENTS

We perform search-by-similarity on the VOC2006 image database [14] and on a 100K image dataset (INRIA Holidays+Flickr [5]). In all experiments, the minor kernel between local descriptors is the χ^2-gaussian kernel at power $q = 5$. Mean Average Precision (MAP) is used to compare retrieval performances. We consider performances both in terms of accuracy and eff ciency.

5.1. $UFast_K$ vs Bow & Kernel on Bags performances

VOC2006 contains 5304 images organized in 10 classes. Queries I_q are randomly sampled from the database and each search-by-similarity method performs a database ranking[1]. The MAP is computed by considering as relevant, any image in the same class as the current I_q. This experiment allows to focus on semantical similarity aspects: the similarity between any pair of images from the same class should be high, and low otherwise, and the VOC2006 classes are def ned by highly semantic concepts as bottle, bus, cars ... so that each class always contains very different objects.

Bags of SIFT are extracted for all images. For $UFast_K$ settings, we use the standard implementation E^2LSH from [15] ($R = 225$, $L = 100$, $H = 20$). We compare $UFast_K$ with K_{power} (eq.(3)) and the Bag-of-Words method with the standard parameter settings [16, 17]: database SIFT vector K-means clustering, tf-idf weighting scheme, and Euclidean distance.

MAP and time results (expressed in terms of ratio for kernel approaches) are reported on Tab.1 with 4 dictionary sizes: 100, 200, 300 and 1000 for BoW. First, let say that the MAP scores are not very high because the similarity search is evaluated here without any learning. Far away from near-duplicate or copy-detection, the evaluation here really focuses on highly semantical similarity purpose. It highlights the ability of the methods to get high similarity scores for categories of objects and not only for the same object taken under various points of view.

	BoW_1	BoW_2	BoW_3	BoW_{1k}	K_{power}	$UFast_K$
MAP %	6.5	6.5	6.1	5.3	9.4	9.5

Table 1. Similarity search performance of $UFast_K$, BoW and K_{power} methods on VOC2006 database.

We can observe on Tab.1 that BoWs do not allow to reach same performances as kernel approaches. Indeed, the kernel MAPs exhibit relative gain of at least 45% over the BoWs. The best score for BoWs is obtained with 200 words.

Our $UFast_K$ kernel gives the best results in term of MAP and time computation. The MAP is slightly better than the K_{power} MAP, while drastically reducing the computation time: the kernel K_{power} is 175 times slower than $UFast_K$.

5.2. Performances on 100K dataset (INRIA Holidays+Flickr)

We now compare $UFast_K$ with State-of-the-art approaches for image retrieval task on large databases. These databases contains 1491 images for INRIA Holidays, at which are added 100,000 "distractors" images from Flickr to reach a 100K dataset [5]. We have used the experimental setup of [5] and the local features publicly available from authors' webpage. The comparison here is more near duplicate detection oriented. Tab.2 shows State-of-the-art results provided by

Method		MAP
BoW (from [18])	$1K$ words	41.4
	$20K$ words	44.6
	$200K$ words	54.9
VLAD (from [5])	16 words, D=2048	49.6
	64 words, D=8192	52.6
UFast$_K$		76

Table 2. Comparison of the accuracy of $UFast_K$ with State-of-the-art approaches on INRIA Holidays dataset (1491 images)

authors in [18, 5] on the 1491 image of INRIA Holidays dataset with the same experimental protocol. These results show that our method is more accurate than BoW approaches which quantize local features but consider only hard-assignment and even more accurate than VLAD which integrates a normalized linear kernel on shifted vectors.

[1] Because of the various size of the categories in this database, ranking lists over the 100 best images will be considered here.

Tab.3 present the results for experiments on the 100K (INRIA Holidays+Flick) dataset conducted as def ned by the experimental protocol in [5]. The results given for VLAD are actually extracted from the curves in [5] and based on details provided by the authors. We can thus see that for about the same time of computation we have a relative gain of 10% in MAP. These results show that the non-linear mapping used in $UFast_K$ is more accurate than the linear one used in original VLAD for that database for comparable running times.

Method	MAP	Time (sec)
VLAD (from [5]) 64 words, D=8192	39	$9-10$
UFast$_K$	43	12

Table 3. Performances on 100K dataset of $UFast_K$ vs VLAD

6. CONCLUSION

In this paper, we focus on designing a kernel-based similarity approach to f nd a trade-off between powerful data representation and eff cient similarity scheme. We propose a new kernel on Bag-of-Features, the Ultra Fast kernel, based on the kernels on sets of vectors derived from kernels on vectors. Our framework encompasses dictionary-based and kernel on set-based similarity functions, including recent Fisher representation simplif cation. We have evaluated our method in a semantic similarity search context and for a near copy detection task showing the good tradeoff between accuracy of exhaustive kernel on Bags and eff ciency of Bow representation, our kernel provides. The designed kernel fulf lling Mercer's conditions, we have hence integrated it in a SVM framework to evaluate its performances in an active learning context [19] and we are currently working on evaluating it for large scale classif cation task.

A. VLAD IS A LINEAR-SHIFTED KERNEL ON BAGS

Starting from eq.(8), we can write:

$$\begin{aligned}\kappa(mini_m(B_q), mini_m(B_j)) &= \sum_{\mathbf{b}_{rq}\in B_q}\sum_{\mathbf{b}_{sj}\in B_j} < \mathbf{b}_{rq}-\mathbf{w}_m, \mathbf{b}_{sj}-\mathbf{w}_m > \\ &= < \sum_{\mathbf{b}_{rq}\in B_q} \mathbf{b}_{rq}-\mathbf{w}_m, \sum_{\mathbf{b}_{sj}\in B_j} \mathbf{b}_{sj}-\mathbf{w}_m > \\ &= < \sum_{\mathbf{b}_{rq}\in B_q} \phi_m(\mathbf{b}_{rq}), \sum_{\mathbf{b}_{sj}\in B_j} \phi_m(\mathbf{b}_{sj}) > \end{aligned} \quad (10)$$

with ϕ_m a mapping function from $\mathbb{R}^d$ to $\mathbb{R}^{d\times M}$ such that:

$$\phi_m(\mathbf{x}) = \mathbf{x}\otimes U_m - \mathbf{w}_m \otimes U_m \quad (11)$$

where U_m, a vector of $M-1$ zeros and a 1 at the m^{th} component, $m = Q(\mathbf{x})$ as def ned for eq.(5) and $\mathbf{w}_m$ is the codeword assigned the vector $\mathbf{x}$.

With the notations of [5], i^{th} component of the VLAD descriptor associated to m^{th} cluster is def ned by:

$$\mathbf{v}_{m,i} = \sum_{\mathbf{x}\in B|Q(\mathbf{x})=m} \mathbf{x}_i - \mathbf{w}_{m,i} = \sum_{\mathbf{x}\in B|Q(\mathbf{x})=m} \phi_m(\mathbf{x}) \quad (12)$$

where $\mathbf{x}_i$ and $\mathbf{w}_{m,i}$ respectively denote the i^{th} component of the local descriptor $\mathbf{x}$ and its assigned visual word $\mathbf{w}_m$. We can f nally rewrite our κ as:

$$\kappa(mini_m(B_q), mini_m(B_j)) = < \mathbf{v}_{m,q}, \mathbf{v}_{m,i} > \quad (13)$$

In [5], the VLAD vector $\mathbf{v}_i$ is further L^2-normalized which leads, instead of considering $UFast_{K_{VLAD}}$ with the above κ, to consider the normalized kernel:

$$UFast_{K_{VLAD}}(B_q, B_j) = \frac{UFast_{K_{VLAD}}(B_q, B_j)}{\sqrt{UFast_{K_{VLAD}}(B_q, B_q) UFast_{K_{VLAD}}(B_j, B_j)}}$$

B. REFERENCES

[1] D.G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol. 60, no. 2, pp. 91–110, 2004.

[2] D. Chen, NM. Cheung, S. Tsai, V. Chandrasekhar, G. Takacs, R. Vedantham, R. Grzeszczuk, and B. Girod, "Dynamic selection of a feature-rich query frame for mobile video retrieval," in *ICIP*, 2010.

[3] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *IEEE ICCV*, 2003, vol. 2, pp. 1470–1477.

[4] X. Zhou, N. Cui, Z. Li, F. Liang, and T. S. Huang, "Hierarchical Gaussianization for Image Classif cation," in *ICCV*, 2009.

[5] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *CVPR*, 2010.

[6] K. Grauman and T. Darrell, "Pyramid match hashing: Sublinear time indexing over partial correspondences," *CVPR*, 2007.

[7] R. Kondor and T. Jebara, "A kernel between sets of vectors," in *Machine Learning-International Workshop then conference*, 2003, vol. 20, p. 361.

[8] C. Wallraven, B. Caputo, and A. Graf, "Recognition with local features: the kernel recipe," in *ICCV*, 2003.

[9] S. Lyu, "Mercer kernels for object recognition with local features," *CVPR*, 2005.

[10] P.-H. Gosselin, M. Cord, and S. Philipp-Foliguet, "Kernel on bags for multi-object database retrieval.," *CIVR*, 2007.

[11] Z. Zhang, Z.-N. Li, and M. S. Drew, "Learning image similarities via probabilistic feature matching," in *ICIP*, 2010.

[12] Y. Chen and J.Z. Wang, "Image categorization by learning and reasoning with regions," *JMLR*, vol. 5, pp. 913–939, 2004.

[13] Ondřej Chum, James Philbin, Michael Isard, and Andrew Zisserman, "Scalable near identical image and shot detection," in *ACM CIVR*, New York, NY, USA, 2007, pp. 549–556, ACM.

[14] M. Everingham, A. Zisserman, C. K. I. Williams, and L. Van Gool, "Pascal voc2006," 2006.

[15] A. Andoni, "E2lsh," http://www.mit.edu/∼andoni/LSH/.

[16] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *CVPR*, 2007.

[17] J. Yang, Y.G. Jiang, A.G. Hauptmann, and C.W. Ngo, "Evaluating bag-of-visual-words representations in scene classif cation," in *MIR*. ACM, 2007.

[18] H. Jégou, M. Douze, and C. Schmid, "Packing bag-of-features," in *ICCV*, 2009.

[19] D. Gorisse, M. Cord, and F. Precioso, "Salsas: Sub-linear active learning strategy with approximate k-nn search," *Pattern Recognition*, vol. In Press, Corrected Proof, pp. –, 2010.