Early burst detection for memory-efficient image retrieval

Miaojing Shi, Yannis Avrithis, Hervé Jégou

Early burst detection for memory-efficient image retrieval
– *Extended version* –

Miaojing Shi [*]
Peking University

Yannis Avrithis
University of Athens, NTUA

Hervé Jégou
Inria

Abstract

Recent works show that image comparison based on local descriptors is corrupted by visual bursts, which tend to dominate the image similarity. The existing strategies, like power-law normalization, improve the results by discounting the contribution of visual bursts to the image similarity.

In this paper, we propose to explicitly detect the visual bursts in an image at an early stage. We compare several detection strategies jointly taking into account feature similarity and geometrical quantities. The bursty groups are merged into meta-features, which are used as input to state-of-the-art image search systems such as VLAD or the selective match kernel. Then, we show the interest of using this strategy in an asymmetrical manner, with only the database features being aggregated but not those of the query.

Extensive experiments performed on public benchmarks for visual retrieval show the benefits of our method, which achieves performance on par with the state of the art but with a significantly reduced complexity, thanks to the lower number of features fed to the indexing system.

1. Introduction

SELF-SIMILARITY in images is a recurring concept in image analysis and computer vision. It is related to the statistics of natural images, which are highly redundant. This concept has been tackled in many applications. For instance, it is exploited in the non-local means de-noising algorithm [6], where each patch is updated by a weighted sum of similar patches. It is also useful for spatial verification and image rectification by exploiting the geometrical assumption of repeated patterns [28]. In the context of image matching, early works [10] like the one of Shechtman and Irani [33] compute the similarity between two images with the "self-similarity descriptor", a compact local descriptor that is densely computed on the image.

In recent works based on SIFT local features [19], the underlying statistical phenomenon is often referred to as *visual burstiness* [14], by analogy to the terminology used in a context of textual information retrieval. As in [18, 20] where models like the Dirichlet distributions [20] are proposed to better reflect the bursty nature of documents. Computer vision researchers question the issue of feature independence implicitly assumed in the bag-of-words model [14, 17, 8, 31, 41, 9], they show the importance of taking care of visual burstiness in image retrieval and image classification: the bursty features tend to dominate the similarity measure, which degrades the quality of the comparison, as other non-bursty yet possibly distinctive features have a comparatively lower contribution.

Various strategies have been proposed to discount the contribution of bursts on the similarity measure. Among the proposed approaches, some are inspired by text like the so-called power-law normalization [14, 17] for bag-of-words or the Polya or Dirichlet models [8]. These strategies have been pragmatically extended to and improved for more complex image vector representations such as VLAD [2, 9] or the Fisher vector [25, 8]. They are also standardly used in matching approaches like Hamming Embedding [15] or selective match kernels (SMK/ASMK) [37]. Recently, We stress a key difference between textual and visual bursts. In images, the space of features is continuous and not discrete as in text, which makes it difficult to determine which features are bursty or not. This may explain why bursts are handled by very simple techniques such as power-law, residual normalization [2, 37] or weighting votes [14].

This paper revisits the concept of visual bursts by considering several detection strategies aiming at identifying the bursts directly from the SIFT descriptors. Our approach is inspired by the work of Turcot and Lowe [42], who remove the features that are unlikely to match a query in an image collection. In contrast to their approach, which requires to off-line cross-match the whole database, we focus on detecting bursts within an image. Another difference is that we merge similar descriptors into a single representative vector instead of selecting features. Therefore we compress the representation without discarding any feature.

The effectiveness of our approach depends on two im-

[*] Miaojing Shi has worked on this paper while he was a visiting student at INRIA Rennes.

portant design choices. First, we construct the affinity matrix between patches. We combine (i) a probabilistic measure learned on a patch database [44] with (ii) a kernel defined on geometrical quantities such as scale and orientation, which were shown useful for burst detection by Torii *et al*. [41]. Second, we evaluate several kernelized clustering algorithms to produce groups from the affinity matrix. The clear winner amongst all clustering methods is a simple strategy performing the connected component analysis of the thresholded matrix.

Finally, we propose an asymmetric aggregation method: we apply our burst detection method on database side but do not aggregate the query descriptors. This further improves the performance while offering a memory footprint per image identical to the one of our symmetric burst aggregation.

The paper is organized as follows. Section 2 introduces our representation and retrieval model. Section 3 discusses the concept of visual burst and proposes our method to detect bursts. Section 4 evaluates our methods on public retrieval benchmarks. Our results demonstrate the benefits of our approach, which improves the state-of-the-art in memory/performance trade-off.

2. Representation and matching

We assume an image is represented by a set $\mathcal{F}$ of local features, where each feature $f \in \mathcal{F}$ is represented by a d-dimensional local descriptor u_f, local scale s_f and orientation θ_f. We ignore position because we do not use geometrical information to match two images, but rather to analyze similarities within images. We further assume that each descriptor $x = u_f$ is quantized on a codebook $\mathcal{C}$ of k visual words, or cells. We adopt the matching model [37], whereby the similarity of images $\mathcal{F}, \mathcal{G}$ is measured by

$$\mathrm{S}(\mathcal{X},\mathcal{Y}) = \nu(\mathcal{X})\nu(\mathcal{Y})\sum_{c\in\mathcal{C}} w_c \mathrm{M}(\mathcal{X}_c,\mathcal{Y}_c), \qquad (1)$$

where $\mathcal{X}, \mathcal{Y}$ are the descriptors of $\mathcal{F}, \mathcal{G}$ respectively, $\mathcal{X}_c$ are the descriptors of $\mathcal{X}$ assigned to cell c, M is a cell similarity function, w_c is a weighting factor for c and $\nu(\mathcal{X})$ is a normalization factor such that self-similarity of $\mathcal{X}$ is $\mathrm{S}(\mathcal{X},\mathcal{X}) = 1$. Although (1) is a general model that includes special cases of several popular methods like bag of words (BoW) [35], VLAD [17] and Hamming Embedding (HE) [13], it is clearly motivated by discarding geometric information for efficiency reasons.

We would rather like to detect burstiness at an early stage to fuse them, and to use any standard search infrastructure off-the-shelf. Therefore, given an image $\mathcal{F}$, we detect feature bursts in $\mathcal{F}$ and represent it by a set of aggregated descriptors. This is an off-line operation, *i.e*., the descriptors can be quantized, encoded and searched as usual.

For the cell similarity function M, we use VLAD [17, 17] and SMK/ASMK [37]. These are two methods targeted for different scenarios with different memory/speed/accuracy compromises. Our descriptor aggregation can be combined and provide benefits in both frameworks. Interestingly, the state-of-the-art ASMK, which is also an aggregated representation intended to address the burstiness problem, is complementary to our strategy.

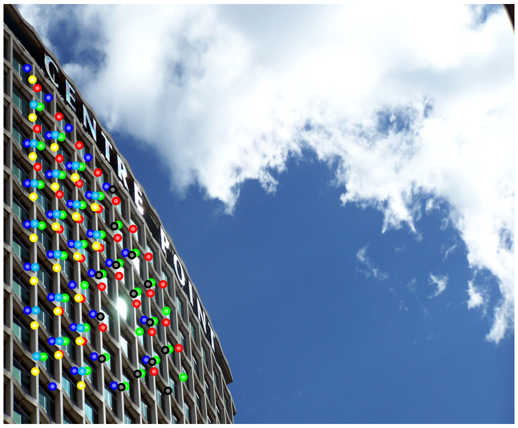

Figure 1. An image along with the features of the six most populated bursts detected. A dot is shown at the position of each feature, colored according to the burst it belongs to; the remaining features are not shown.

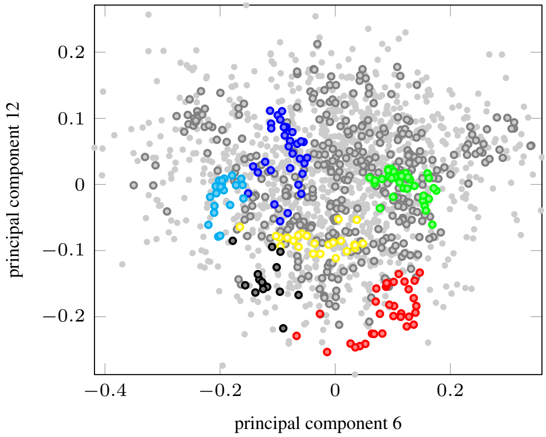

Figure 2. The 1610 features of the image of Fig. 1 in descriptor space. After PCA analysis, we plot two of the principal components found (component 6 *vs*. 12). Colored groups of points correspond to the six most populated bursts, exactly as in Fig. 1. Gray points correspond to smaller bursts. Smaller light gray dots are isolated descriptors that do not belong to any burst.

3. Early detection of visual bursts

What are bursts? As a representative example, Fig. 1 depicts a clean view of a building exhibiting a lattice structure of almost identical tiles formed by its windows. To visualize the resulting groups of similar patches, we extract normalized RootSIFT [1] descriptors on Hessian-affine [21] features, compute pairwise distances, and find the connected components formed by joining pairs whose distance is below a certain threshold. This is a simple but very effective approach. By coloring features according to their group, it is clearly seen that the same pattern of 5-6 groups appears around every window on the building surface.

Since patches are very similar in appearance, one might expect a high density of points around each burst in the descriptor space, so that bursts can be easily found by clustering or mode seeking. However, as illustrated in Fig. 2, this is far from being true. It is not easy to visualize what happens in a 128-dimensional space by a mere 2D projection. Still, whatever the projection, *isolated* descriptors (not belonging to any group) appear to have the same density as *bursty* ones (belonging to some group), while bursts have arbitrary shape and large extent. One cannot hope that bursts will fit within the cells of a codebook, especially if the the latter is trained on random samples without taking burstiness into account. Our approach to detect bursts before quantizing descriptors then makes sense.

Besides structure in man-made scenes, other sources of burstiness may be texture, *e.g.* in natural environments, or the fact that feature detectors may give multiple responses, *e.g.* along edges or around corners at different scales. Depending on the case, geometry also plays a role. In contrast to methods that focus on repeating structures and symmetries [41, 40], we consider individual features disregarding position and neighborhood: the search for repeating groups of features is too constrained and may fail to identify bursts, especially in natural scenes. However, we do investigate scale and orientation. For instance, similar patches in Fig. 1 have the same orientation, but this is not always the case on a textured surface. It is not known whether bursty features share the same appearance only or scale and orientation as well; this is something we determine experimentally.

Section 3.1 defines a *feature kernel* measuring the similarity of two individual features. Then in section 3.2 we consider a number of methods to detect bursts using the given kernel function and conduct a preliminary evaluation.

3.1. Feature kernel

Given two local features f, g in an image, we define a *feature kernel* function

$$k(f,g) = k_u(u_f, u_g) k_s(s_f, s_g) k_\theta(\theta_f, \theta_g), \qquad (2)$$

consisting of three factors, namely the *descriptor kernel* k_u, the *scale kernel* k_s and the *orientation kernel* k_θ. Intuitively, this function measures if the two corresponding patches in the image are similar in appearance and have similar scales and orientations. Factors k_s, k_θ can be optionally omitted.

Descriptor kernel. Given a pair of descriptors $x, y \in \mathbb{R}^d$, *descriptor kernel* k_u measures their similarity and is a function of the inner product $z = \langle x, y \rangle$, which is equivalent to their distance if x, y are ℓ_2-normalized. Seeing z as a random variable, we define this function as the probability, given z, that x, y belong to the same burst, *i.e.* they correspond to two matching image patches. In particular, we adopt a generative model for a binary classifier: if $\mathcal{B}$ is the class of descriptor pairs that belong to the same burst and $\overline{\mathcal{B}}$ is its complement, we define

$$k_u(x,y) = p(\mathcal{B} | \langle x, y \rangle), \qquad (3)$$

where

$$p(\mathcal{B}|z) = \frac{p(z|\mathcal{B})p(\mathcal{B})}{p(z|\mathcal{B})p(\mathcal{B}) + p(z|\overline{\mathcal{B}})p(\overline{\mathcal{B}})} \qquad (4)$$

is the posterior probability of $\mathcal{B}$ given z; $p(z|\mathcal{B}), p(z|\overline{\mathcal{B}})$ are the class-conditional densities of $\mathcal{B}, \overline{\mathcal{B}}$ respectively and $p(\mathcal{B}), p(\overline{\mathcal{B}})$ are their prior probabilities.

We train such a classifier from a dataset of matching/non-matching patch pairs [45]. This dataset consists of patches sampled from 3D reconstructions of the Statue of Liberty (New York), Notre Dame (Paris) and Half Dome (Yosemite). Two patches in two different views of the same 3D scene are matched if they are projections of the same 3D point. Such patches are very similar in appearance, so they provide a good model for bursts.

We extract a SIFT descriptor from each patch in the dataset and compute the inner product $z = \langle x, y \rangle$ for each pair (x, y) of descriptors. If $B, \overline{B}$ are the sets of all observations of z for matching and non-matching pairs respectively according to the ground truth, we model the class-conditional densities $p(z|\mathcal{B}), p(z|\overline{\mathcal{B}})$ by fitting normal densities to the samples of $B, \overline{B}$ respectively, according to maximum likelihood. The prior probabilities are $p(\mathcal{B}) = |B|/N, p(\overline{\mathcal{B}}) = |\overline{B}|/N$, where $N = |B| + |\overline{B}|$ is the total number of samples. Fig. 3 shows the posterior probability $p(\mathcal{B}|z)$ computed from (4). It appears that $k_u(x, y)$ is a sigmoid function of the inner product $z = \langle x, y \rangle$ that can discriminate well enough matching from non-matching descriptor pairs.

Scale and orientation kernels. Given two patch scales s, t, *scale kernel* $k_s(s, t)$ is given by a Gaussian kernel of their logarithms,

$$k_s(s,t) = \exp\left\{ -\lambda \log^2 \left(\frac{s}{t} \right) \right\}. \qquad (5)$$

This choice makes k_s invariant to absolute scale, as long as ratio s/t remains constant.

The situation is similar for orientation, but following the work [38], we use the equivalent of Gaussian for periodic distributions, which is the von Mises distribution. In particular, given two patch orientations θ, ϕ, *orientation kernel* $k_\theta(\theta, \phi)$ is given by the von Mises kernel

$$k_\theta(\theta, \phi) = \frac{\exp(\kappa \cos(\theta - \phi)) - \exp(-\kappa)}{2 \sinh(\kappa)}. \qquad (6)$$

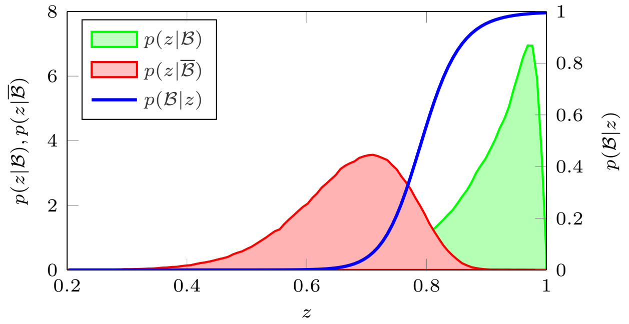

Figure 3. Distributions of $z = \langle x, y \rangle$ for matching and non-matching SIFT descriptor pairs (x, y) from the dataset of [45], where we fit class-conditional densities $p(z|\mathcal{B}), p(z|\overline{\mathcal{B}})$. Posterior probability $p(\mathcal{B}|z)$ is used to compute kernel k_u (3).

Parameters λ, κ are tuned experimentally according to [38].

3.2. Burst detection and aggregation

Given an image $\mathcal{F} = \{f_1, \dots, f_n\}$ with n features, define $n \times n$ affinity matrix K with elements

$$K_{ij} = k(f_i, f_j) \tag{7}$$

for $i, j = 1, \dots, n$, where kernel k is given by (2). The affinity matrix includes all pairwise feature similarities in $\mathcal{F}$ and is the only input for burst detection. Now, given the discussion so far, a candidate algorithm should

- be based on a kernel method, or more generally, operate on metric spaces. One reason is that bursts are of arbitrary shape in the feature space, as shown in Fig. 2. Another is that we have formulated the input as an affinity matrix, in order to combine with scale and orientation proximity. This excludes algorithms that represent points in a Euclidean space, like k-means.
- be able to automatically determine the number of groups such that non-matching features are never grouped, or at least have a parameter to control it so that the resulting number varies smoothly and can be close to the original size n. This is because we would like the number of aggregated descriptors to vary from n down to a certain percentage of n.

Since we look for groups of similar features as measured by kernel k, any clustering or mode seeking algorithm that respects the above constraints would do in theory. Thus we examine a number of existing methods.

Connected components. Features are treated as nodes of an undirected graph with an edge for each pair (f, g) of features with $k(f, g)$ above a threshold τ. Then we compute the connected components of the graph and consider each component with more than one feature to be a bursty group. The remaining components each contain one isolated feature. This is the fastest and most effective method.

Quick shift [43] is a very simple, fast mode seeking method that can operate in non-Euclidean spaces. Eventually, all points are connected into a single tree; after that, edges are disconnected according to a threshold τ. This aspect is similar to connected components: τ needs to be tuned to obtain the desired number of groups, but we evaluate a range of values in our experiments anyway.

Kernel k-means [32] is a simple kernel method, parametrized by the desired number k of groups. However, it has high complexity and is not designed for k being large.

Spectral methods. Representative spectral methods include spectral clustering [23] and normalized cuts [34, 11]. Certain group of bursty features is expected to contribute most of its energy to one of the leading eigenvectors of matrix K and give the largest projection on this eigenvector. It is possible to specify the number of groups — *e.g.*, apply k-means on those leading eigenvectors [23]. But the cost is prohibitive in general. We design a method which hierarchically applies spectral clustering to subdivide each group, while automatically determining the number of groups. We call this variant hierarchical spectral clustering.

Evaluation. We conduct a preliminary qualitative evaluation of the above methods on a limited sample of images to determine whether they indeed satisfy our constraints and make an initial selection before moving on to larger scale quantitative experiments. We present here an example on one representative image as shown in Fig. 4.

We have tuned each method such that the number of detected groups is 80-85% of the initial features; since all descriptors in a group are aggregated, this measure is referred as aggregation% in the experiments and is crucial in the memory-performance trade-off. It can be seen that most methods yield consistent groups; however, normalized cuts gives many inconsistent groups while the groups found by spectral clustering are too small. The result of connected components is the cleanest, and Fig. 6 verifies this fact: each group contains almost identical patches, precisely centered at the position of each feature.

Fig. 5 further shows the group size distributions. It appears that kernel k-means is problematic in the sense that the groups found are far less than the target, resulting in an aggregation% of 39%, rather than 80%. Although the required number of groups is given as a parameter, too many clusters become empty and the few remaining are most detected as bursts. Spectral clustering finds too many small groups. In general, we would like most features to remain isolated and the remaining ones form few bursts. In this sense, connected components, hierarchical spectral clustering and normalized cuts appear reasonable.

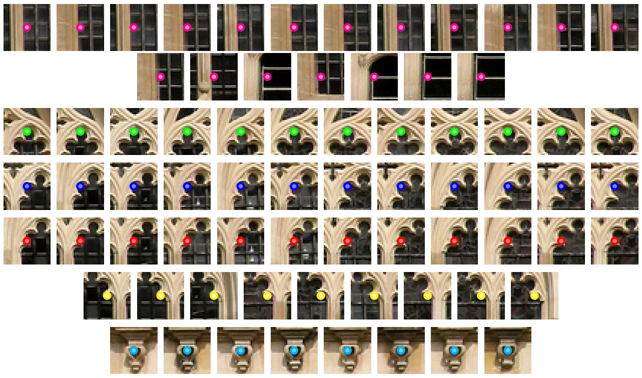

Figure 6. The six most populated bursts found by connected components on the example of Fig. 4. An image patche of size 30 × 30 pixels is shown for each feature; a dot is shown at each feature position, colored according to burst exactly as in Fig. 4.

Taking into account the preliminary evaluation and the fact that the spectral methods are quite slow to apply at large scale, we choose the first three methods of Fig. 4 for further quantitative evaluation in the context of retrieval.

Burst aggregation. Given an image, the result of burst detection is a partition of its features into groups. We simply take the average of the descriptors in each group and ℓ_2-normalize them. Discarding geometry, this yields a set of (aggregated) descriptors to represent the image, so any encoding or search model applies.

4. Experiments

We evaluate and compare the proposed burst detection and aggregation in the context of two different image retrieval models, namely VLAD and SMK/ASMK. We first discuss the evaluation protocol and give some implementation details; then we discuss the impact of parameters for each method and analyze the benefits we obtain in the memory-performance trade-off.

4.1. Experimental setup

Datasets. We conduct experiments on three retrieval benchmarks, namely *Holidays* [13], *Oxford* [26], and *Paris* [27]. We study the impact of parameters mainly on *Holidays*. To evaluate performance on larger scale, we add distractors from the *Flickr* 100k set [26] to *Holidays* and *Oxford*.

Descriptors. Local features are extracted with the Hessian-affine detector [21] on *Holidays* and its improved version [24] on *Oxford* and *Paris*. We adopt the default parameters of the detector, but we also use a lower threshold and larger patch size to yield different feature sets for *Holidays*. Since we are generating a reduced feature set by aggregating, this helps evaluate the performance at the same memory depending on the initial feature set. Table 1 shows the three different feature sets used. We use RootSIFT [1] descriptors in all our experiments.

feature set	#features	threshold	patch size
Holidays-S	4.4M	500	21
Holidays-M	4.3M	500	41
Holidays-L	6.6M	300	41

Table 1. Three feature sets obtained from *Holidays* with different detector parameters. *Holidays-S* is the default.

Evaluation. Retrieval performance is measured in terms of mean average precision (mAP). As we vary the number of detected bursts, we generate aggregated feature sets of varying size; the ratio to the original size, averaged over a dataset, is called aggregation%. We thus measure mAP as a function of aggregation% to evaluate the memory-performance trade-off. When an inverted file is used, we also measure the imbalance factor [36], which is directly related to the search cost and should be as close as possible to the optimal value of 1.

Burst detection and aggregation. On *Holidays*, we employ the proposed descriptor kernel and scale kernel only; referring to (2), we use kernel $k_u k_s$. On *Oxford* and *Paris* on the other hand, we use all three kernels, *i.e.* descriptor, scale and orientation; that is, $k_u k_s k_\theta$. This setting always gives the best performance. We initially evaluate three different burst detection methods, and then focus on connected components. We apply different thresholds to the affinity matrix to vary the number of bursts such that aggregation% varies in the range of 10-100%. In all graphs, the baseline is always the rightmost point. We follow two strategies: database descriptors are always aggregated, while query descriptors may be aggregated or not; these are called symmetric and asymmetric aggregation, respectively.

Retrieval models. We conduct experiments on two representative retrieval models, VLAD [17] and SMK/ASMK [37]. In particular, we use the efficient versions SMK*/ASMK* where descriptors are binarized. Both models target at reducing the effect of burstiness: VLAD by power-law normalization, and ASMK by its own aggregation after quantization. We present the benefit from our early burst detection but interestingly we also show it can be complementary to such methods.

Cost. The query cost is quadratic (linear) in aggregation% for symmetric (asymmetric) aggregation, *i.e.* always less than baseline. The off-line additional cost of burst detection and aggregation is $O(n^2)$ where n is the number of features per image. As a comparison, quantization with a flat vocabulary is $O(nk)$, where k is the vocabulary size. Therefore the fixed cost of early detection is negligible in the AMSK pipeline ($k \gg n$) and more expensive for VLAD ($k < n$), yet reasonable.

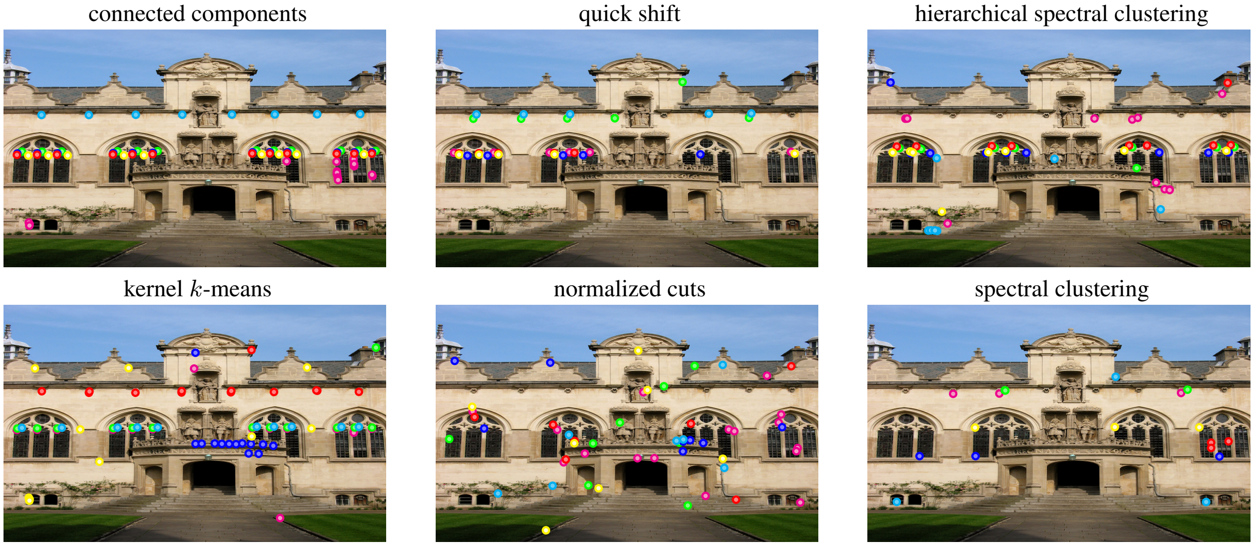

Figure 4. Feature grouping and burst detection with six methods. In each case, the six most populated bursts are shown with a dot at the position of each feature, colored according to the burst it belongs to.

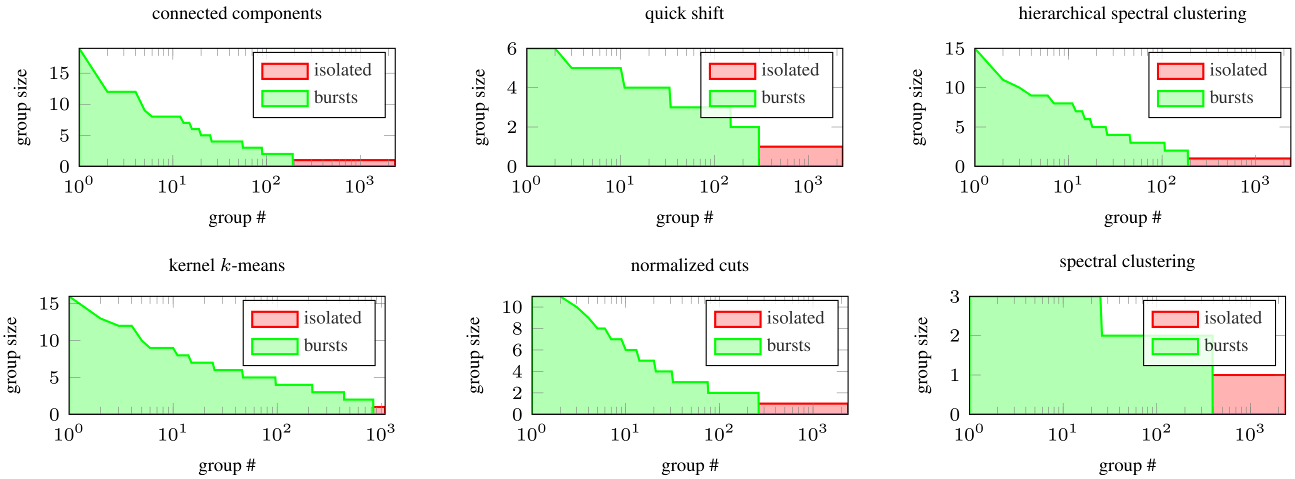

Figure 5. Size distribution of groups found by each method for the example of Fig. 4. Groups of two features or more are considered bursts and the remaining groups each contain one isolated feature. Observe the logarithmic axis: bursts are only a small fraction of groups.

4.2. Results on VLAD

We begin by providing a quantitative comparison of burst detection and aggregation methods, justifying our choice of algorithm. We then provide results on varying vocabulary size k, on the effect of power-law normalization and on large scale experiments. Unless otherwise stated, we use a vocabulary of $k = 16$, no power-law normalization and symmetric aggregation.

Burst detection and aggregation. Fig. 7 illustrates the performance of VLAD on *Holidays-L* under varying aggregation%. As discussed in Section 3.2, we compare the three most promising methods of the initial qualitative evaluation, *i.e*. connected components, quick shift and hierarchical spectral clustering. Connected components choice is always superior, while the other two methods do not achieve any benefit. Therefore, we focus all remaining experiments on connected components.

Vocabulary size. Fig. 8 shows the performance on *Holidays-L* as a function the vocabulary size k. There is an improvement of 2-3% over the baseline for a large range of aggregation%. The relative improvement is higher for smaller vocabularies. A possible explanation is that bursts are split into different cells when the vocabulary is larger.

Power-law. Fig. 8 also shows the effect of power-law normalization by repeating the measurements for $\alpha = 1$ and $\alpha = 0.7$. It can be seen that, by aggregating bursty features at an early stage, we obtain at least as much gain as

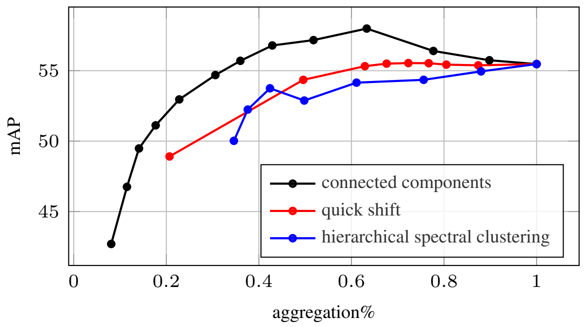

Figure 7. VLAD performance *vs*. aggregation% on *Holidays-L* for three burst detection methods. Vocabulary size k = 16; baseline power-law parameter $\alpha = 1$.

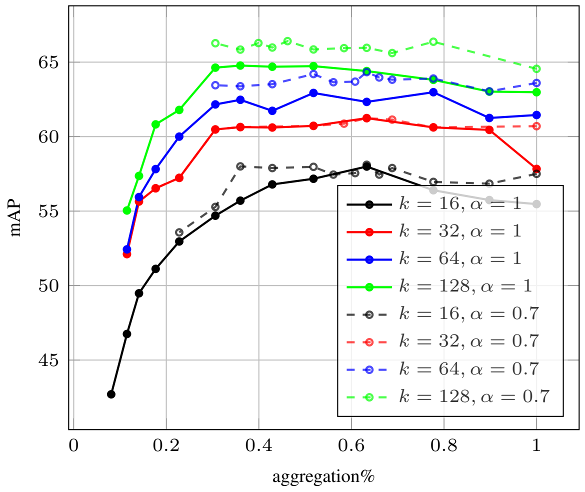

Figure 8. VLAD performance *vs*. aggregation% on *Holidays-L* for different vocabulary sizes k and different power-law parameter α.

with power-law on baseline VLAD. Furthermore, the two methods are complementary as power-law yields a further improvement of 1-1.5% on the highest performance of our early aggregation. With $k = 64$ for instance, mAP is 63.0 and 64.3 for $\alpha = 1$ and $\alpha = 0.7$, respectively. This result is already very close to the highest value of 65.8 obtained using residual normalization (RN) and local coordinate system (LCS) [9].

Large-scale. Table 2 shows large scale results on *Holidays-L* plus distractors. Recall that aggregation% of 1 refers to the baseline. It is impressive that we get absolute performance gain at reduced memory and query time.

4.3. Results on SMK*/ASMK*

We provide results on varying symmetric vs asymmetric aggregation, vocabulary size k, on the impact of SMK*/ASMK* selectivity, on the effect of the initial feature set, on the *Oxford* and *Paris* datasets, on the imbalance factor, on multiple assignment and on large scale experiments. We also include a comparison to the state-of-the-art. Unless otherwise stated, we use a vocabulary of k = 65k, selectivity parameter $\alpha = 3$ and asymmetric aggregation. This corresponds to aggregation% = 1 for SMK*, and less than 1 for ASMK* because ASMK* provides its own form of aggregation.

aggregation%	1.000	0.764	0.638	0.556
k = 16	41.3	42.7	44.1	45.0
k = 64	46.3	47.5	48.3	48.8

Table 2. VLAD mAP performance *vs*. aggregation% on *Holidays-L +Flickr* 100k distractors for two vocabulary sizes, 16 and 64.

Symmetric *vs*. asymmetric. Fig. 9 compares symmetric and asymmetric aggregation on *Holidays-L*; recall that in the latter case the query is not aggregated. It turns out that for low aggregation% asymmetric is superior, and this observation holds for all our experiments, so we limit to this strategy for the remaining results. This behavior can be explained by the fact that under severe aggregation on both images, most matches are lost. We also observe an impressive improvement on the memory-performance trade-off: we can keep only 30% of the original descriptors for a performance drop of merely 1%.

Vocabulary size. Fig. 10 compares different vocabulary sizes on *Holidays-L*. By adopting our early burst aggregation, we get absolute improvement over all sizes. We observe that the improved trade-off holds for all tested vocabulary sizes. The improvement is more significant for SMK*: performance remains constant or improves even for 60% aggregation before starting to drop. This is expected because ASMK* includes another form of aggregation.

Selectivity. The impact of SMK*/ASMK* selectivity is shown in Fig. 11. It appears that the previous observations hold under varying selectivity and our method maintains good performance over a wide range of aggregation%.

Initial features. Fig. 12 compares three different initial feature sets on *Holidays* and measures mAP *vs*. absolute number of descriptors/image, which directly reflects memory. Now comparing the three sets for any number of descriptors, the largest set maintains a gain of over 10% over the smallest one. This is another aspect of the trade-off and suggests a way to improve performance: *augment the initial features, aggregate, and gain in mAP at the same memory*.

More datasets. We also compare symmetric to asymmetric aggregation on *Oxford* and *Paris* as well, as shown in Fig. 13,14 respectively. These datasets are notoriously more difficult for methods operating under limited memory. Still, our method maintains a good memory efficiency. In *Paris* for instance, we can keep only 50% features for a 4% drop in mAP. Asymmetric aggregation is still superior for low aggregation%.

Imbalance factor. Fig. 15 and 16 investigate the imbalance factor [36] on *Holidays-L*, *Oxford* and *Paris*. By aggregating bursty features at an early stage, we make the inverted file more balanced. Interestingly, the imbalance factor exhibits a minimum at an aggregation% which gives at the same time a good memory-performance trade-off, *e.g.* 60%, 30% respectively for SMK*/ASMK* on *Holidays-L*. In terms of query cost, the benefit of improved imbalance factor should be multiplied by the benefit due to decreased memory: indeed, query time is linear in aggregation%.

Multiple assignment. We further evaluate our burst detection and aggregation by adopting multiple assignment (MA) in the ASMK* framework. We apply it on the query side only as in [15], which is consistent with not aggregating query descriptors in our asymmetric strategy. We use the five nearest visual words as in [37]. Fig. 17 shows ASMK* results for two initial feature sets. *Holidays-L* maintains a gain of 6-7% over *Holidays-S* at the same memory. Comparing to Fig. 12, one can observe that single and multiple assignment give similar performance when aggregating; this means we can have this boost at a fraction of query time without multiple assignment. Finally, Fig. 18 shows ASMK* results on *Oxford* and *Paris*; under multiple assignment, we even get absolute performance improvement in this case.

Large scale. Table 3 shows large scale results on *Holidays-L* and *Oxford* plus distractors. It is interesting that *e.g.* in *Oxford*, the result is more promising than at small scale: we can save 15% of memory at no performance cost and increase efficiency at the same time.

Comparison to the state of the art. Table 4 shows state-of-the-art results compared to our best results on ASMK*. We only compare to methods relating to vocabularies and descriptor representation and not *e.g.* spatial matching [24, 4], query expansion [7, 39], feature augmentation [42, 1] or nearest neighbor re-ranking [29].

The first group of methods relies on a large vocabulary (1M or more) and does not include a descriptor signature. Performance may be improved by learning a finer vocabulary on a larger training set [22], which is a costly off-line process, or using the extremely fine partition of a multi-index [3, 5], which cannot be fully inverted. The second group relies on a smaller vocabulary (100k or less) and embeds a descriptor signature, *e.g.* a Hamming code [12, 37] as in this work, or product quantization code [30, 16]. This approach is superior, but requires additional space.

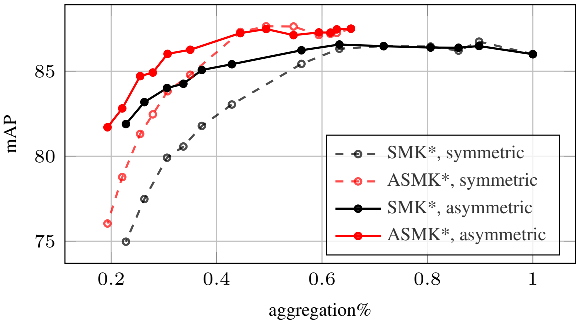

Figure 9. SMK*/ASMK* performance *vs.* aggregation% on *Holidays-L* for symmetric and asymmetric aggregation. Vocabulary size $k = 65$k; selectivity exponent $\alpha = 3$.

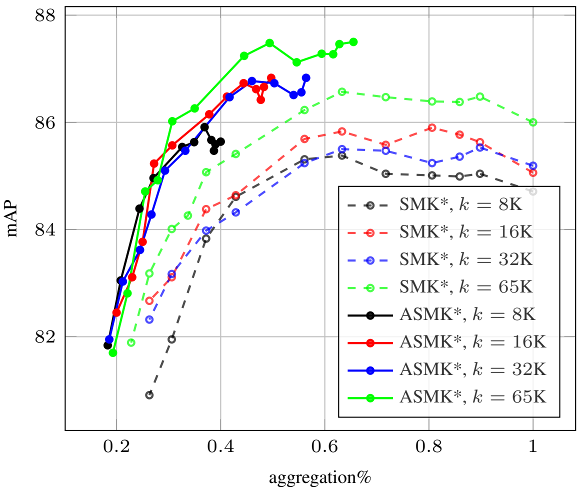

Figure 10. SMK*/ASMK* performance *vs.* aggregation% on *Holidays-L* for different vocabulary sizes k. Selectivity exponent $\alpha = 3$. Asymmetric aggregation.

dataset	*Holidays-L* 101k			*Oxford* 105k		
aggregation%	0.65	0.52	0.28	0.90	0.76	0.55
mAP	85.1	84.5	77.6	68.9	68.9	63.6

Table 3. ASMK* mAP performance on *Holidays-L* and *Oxford* plus *Flickr* 100k distractors. Vocabulary size: 65k. The first column of each dataset is the baseline.

The third group includes ASMK* [37] and this work. There is still a descriptor signature, but the number of descriptors is reduced, as indicated by aggregation%, which is different for each dataset. Despite the lower memory and faster query, these methods are superior to previous ones. Additionally, we get a performance gain over [37] using multiple assignment; in particular, the five nearest visual words as used in [37]. In *Holidays*, we start from the larger feature set *Holidays-L* and aggregate such that the total number of features is not higher than in [37], as in Fig. 12. In the remaining datasets, the gain is due to absolute improvement in the default feature set.

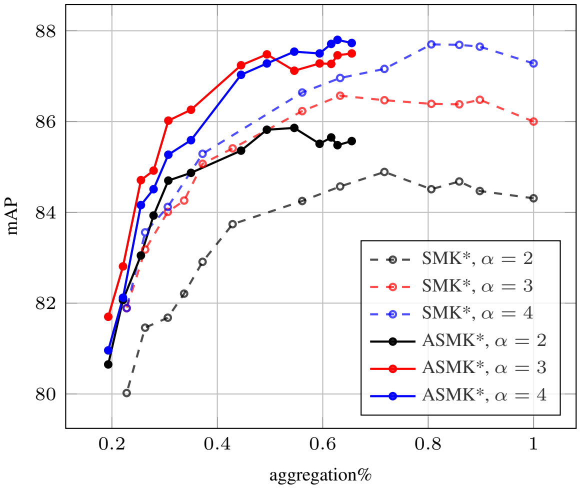

Figure 11. SMK*/ASMK* performance *vs*. aggregation% on *Holidays-L* for different values of selectivity exponent α. Vocabulary size $k = 65$k. Asymmetric aggregation.

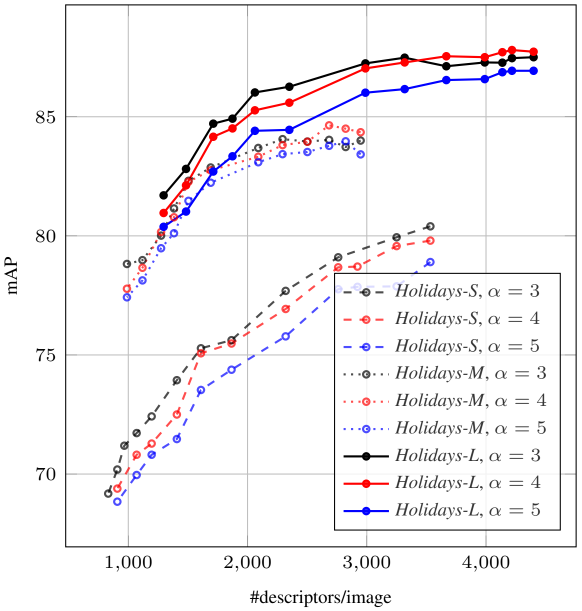

Figure 12. ASMK* performance *vs*. average number of aggregated descriptors per image on *Holidays* for three different initial feature sets and different values of selectivity exponent α. Vocabulary size $k = 65$k; asymmetric aggregation. Note that the rightmost measurement corresponds to aggregation% less than 1.

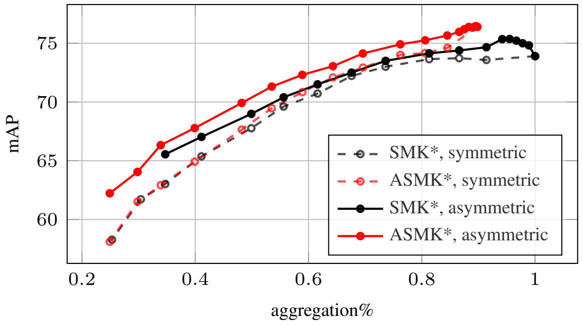

Figure 13. SMK*/ASMK* performance *vs*. aggregation% on *Oxford* for symmetric and asymmetric aggregation. Vocabulary size $k = 65$k; selectivity exponent $\alpha = 3$.

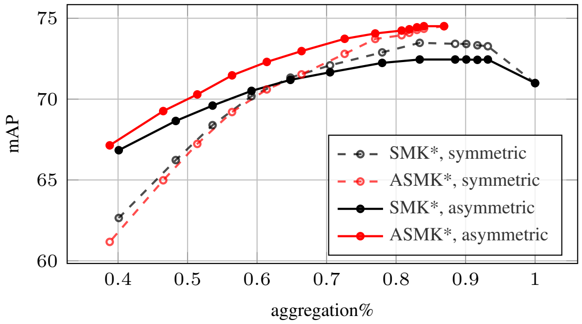

Figure 14. SMK*/ASMK* performance *vs*. aggregation% on *Paris* for symmetric and asymmetric aggregation. Vocabulary size $k = 65$k; selectivity exponent $\alpha = 3$.

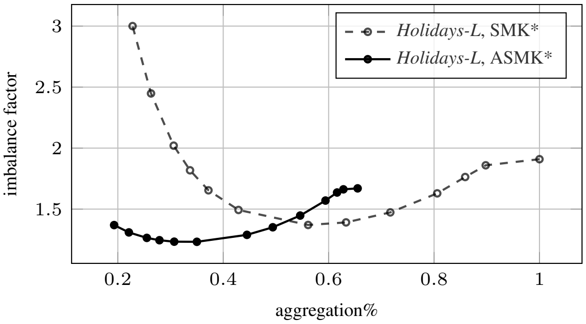

Figure 15. SMK*/ASMK* imbalance factor *vs*. aggregation% on *Holidays-L*. Vocabulary size $k = 65$k; selectivity exponent $\alpha = 3$.

5. Conclusion

Handling burstiness has a significant and positive impact on the accuracy of image search comparison. This paper has shown that our early detection of visual bursts as an effective strategy to produce image match kernel based on local descriptors. It is applied prior to the indexing stage, which might separate bursty features if quantization is employed, and exploits geometrical quantities jointly with feature similarity. Not only our strategy is as or more effective than other methods for handling visual bursts, but it is also complementary to concurrent like the ASMK search engine, leading to state-of-the-art results in a comparable setup.

Another key advantage is that, by fusing the descriptors before feeding them to the indexing or search system, we reduce the computational cost in both the quantization and

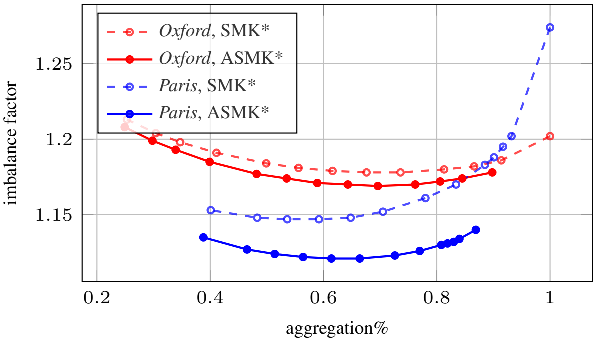

Figure 16. SMK*/ASMK* imbalance factor *vs*. aggregation% on *Oxford* and *Paris*. Vocabulary size $k = 65$k; selectivity exponent $\alpha = 3$.

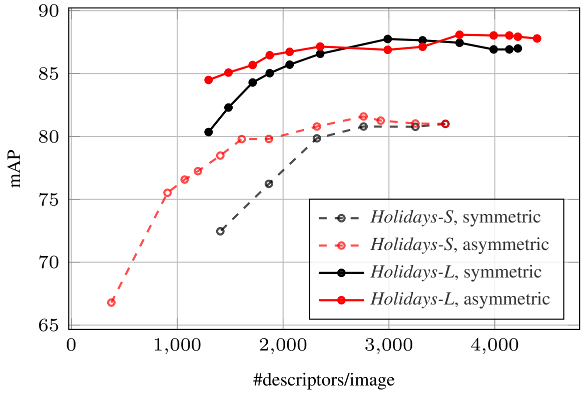

Figure 17. ASMK* performance *vs*. average number of aggregated descriptors per image on *Holidays* with multiple assignment for two initial feature sets and for symmetric and asymmetric aggregation. Vocabulary size $k = 65$k; selectivity exponent $\alpha = 3$.

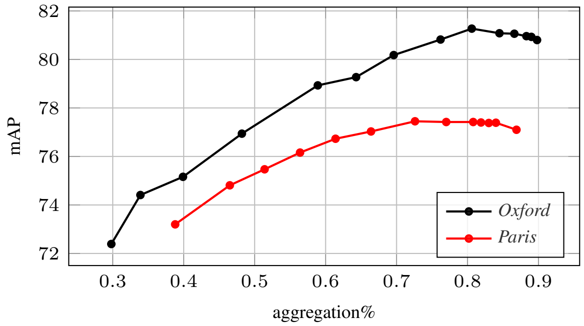

Figure 18. ASMK* performance *vs*. aggregation% on *Oxford* and *Paris* with multiple assignment. Vocabulary size $k = 65$k; selectivity exponent $\alpha = 3$. Asymmetric aggregation.

retrieval steps, typically by a factor of two. Our method also reduces the memory footprint in the same proportion for image search engines employing an inverted file.

Dataset	MA	Hol.	Paris	Oxf.
BoW [27]		-	-	40.3
BoW [27]	✓	-	-	49.3
BoW [24]		-	-	55.8
Fine vocab. [22]		74.9	74.9	74.2
Multi-index [3]	✓	-	69.6	70.3
HE [15]		74.5	-	51.7
HE [15]	✓	77.5	-	56.1
AHE+burst [12]		79.4	-	66.0
AHE+burst [12]	✓	81.9	-	69.8
Query ad. [30]		81.4	70.3	73.9
Query ad. [30]	✓	82.1	73.6	78.0
aggregation%		78%	86%	89%
ASMK* [37]		80.0	74.4	76.4
ASMK* [37]	✓	81.0	77.0	80.4
This work	✓	**88.1**	**77.5**	**81.3**

Table 4. Comparison of our best mAP result to state-of-the-art using inverted files as in BoW or also local descriptors as in HE. We only report results without spatial re-ranking.

Acknowledgements. This work was supported by ERC grant VIAMASS no. 336054. Miaojing Shi was partially supported by NBRPC 2011CB302400, NSFC 61121002, 61375026 and JCYJ 20120614152136201, and Yannis Avrithis was supported by the EU (European Social Fund) and Greek National Fund through the operational program "Education and Lifelong Learning" of the National Strategic Reference Framework, research funding program "ARISTEIA", project "ESPRESSO".

References

[1] R. Arandjelovic and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, 2012. 2, 5, 8

[2] R. Arandjelovic and A. Zisserman. All about VLAD. In *CVPR*, 2013. 1

[3] Y. Avrithis. Quantize and conquer: A dimensionality-recursive solution to clustering, vector quantization, and image retrieval. In *ICCV*. 2013. 8, 10

[4] Y. Avrithis and G. Tolias. Hough pyramid matching: Speeded-up geometry re-ranking for large scale image retrieval. *IJCV*, 107(1):1–19, 2014. 8

[5] A. Babenko and V. Lempitsky. The inverted multi-index. In *CVPR*, 2012. 8

[6] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. In *CVPR*, 2005. 1

[7] O. Chum, A. Mikulik, M. Perdoch, and J. Matas. Total recall II: Query expansion revisited. In *CVPR*, 2011. 8

[8] R. Cinbis, J. Verbeek, and C. Schmid. Image categorization using Fisher kernels of non-iid image models. In *CVPR*, 2012. 1

[9] J. Delhumeau, P. Gosselin, H. Jégou, and P. Perez. Revisiting the VLAD image representation. In *ACM Multimedia*, 2013. 1, 7

[10] T. Deselaers and V. Ferrari. Global and efficient self-similarity for object classification and detection. In *CVPR*, 2010. 1

[11] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral grouping using the Nyström method. *PAMI*, 26(2):214–225, 2004. 4

[12] M. Jain, H. Jégou, and P. Gros. Asymmetric hamming embedding. In *ACM Multimedia*, 2011. 8, 10

[13] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *ECCV*, 2008. 2, 5

[14] H. Jégou, M. Douze, and C. Schmid. On the burstiness of visual elements. In *CVPR*, 2009. 1

[15] H. Jégou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. *IJCV*, 87(3):316–336, 2010. 1, 8, 10

[16] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *PAMI*, 33(1):117–128, 2011. 8

[17] H. Jégou, F. Perronnin, M. Douze, J. Sanchez, P. Perez, and C. Schmid. Aggregating local image descriptors into compact codes. *PAMI*, 34(9):1704–1716, 2012. 1, 2, 5

[18] D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In *ECML*, 1998. 1

[19] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 1

[20] R. E. Madsen, D. Kauchak, and C. Elkan. Modeling word burstiness using the dirichlet distribution. In *ICML*, 2005. 1

[21] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *IJCV*, 60(1):63–86, 2004. 2, 5

[22] A. Mikulik, M. Perdoch, O. Chum, and J. Matas. Learning a fine vocabulary. In *ECCV*, 2010. 8, 10

[23] A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *NIPS*, 2002. 4

[24] M. Perdoch, O. Chum, and J. Matas. Efficient representation of local geometry for large scale object retrieval. In *CVPR*, 2009. 5, 8, 10

[25] F. Perronnin, J. Sánchez, and T. Mensink. Improving the fisher kernel for large-scale image classification. In *ECCV*, 2010. 1

[26] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, 2007. 5

[27] J. Philbin, O. Chum, J. Sivic, M. Isard, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *CVPR*, 2008. 5, 10

[28] J. Pritts, O. Chum, and J. Matas. Detection, rectification and segmentation of coplanar repeated patterns. In *CVPR*, 2014. 1

[29] D. Qin, S. Gammeter, L. Bossard, T. Quack, and L. Van Gool. Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors. In *CVPR*, 2011. 8

[30] D. Qin, C. Wengert, and L. Van Gool. Query adaptive similarity for large scale object retrieval. In *CVPR*, 2013. 8, 10

[31] J. Revaud, M. Douze, and C. Schmid. Correlation-based burstiness for logo retrieval. In *ACM Multimedia*, 2012. 1

[32] B. Schölkopf, A. Smola, and K. Muller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5):1299–1319, 1998. 4

[33] E. Shechtman and M. Irani. Matching local self-similarities across images and videos. In *CVPR*, 2007. 1

[34] J. Shi and J. Malik. Normalized cuts and image segmentation. *PAMI*, 22(8):888–905, 2000. 4

[35] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003. 2

[36] R. Tavenard, H. Jégou, and L. Amsaleg. Balancing clusters to reduce response time variability in large scale image search. In *CBMI*, 2011. 5, 8

[37] G. Tolias, Y. Avrithis, and H. Jégou. To aggregate or not to aggregate: Selective match kernels for image search. In *ICCV*, 2013. 1, 2, 5, 8, 10

[38] G. Tolias, T. Furon, and H. Jégou. Orientation covariant aggregation of local descriptors with embeddings. In *ECCV*, 2014. 3, 4

[39] G. Tolias and H. Jégou. Visual query expansion with or without geometry: refining local descriptors by feature aggregation. *Pattern Recognition*, 47(10):3466–3476, 2014. 8

[40] G. Tolias, Y. Kalantidis, and Y. Avrithis. Symcity: Feature selection by symmetry for large scale image retrieval. In *ACM Multimedia*, 2012. 3

[41] A. Torii, J. Sivic, T. Pajdla, and M. Okutomi. Visual place recognition with repetitive structures. In *CVPR*, 2013. 1, 2, 3

[42] P. Turcot and D. Lowe. Better matching with fewer features: the selection of useful features in large database recognition problems. In *ICCV*, 2009. 1, 8

[43] A. Vedaldi and S. Soatto. Quick shift and kernel methods for mode seeking. In *ECCV*, 2008. 4

[44] S. Winder and M. Brown. Learning local image descriptors. In *CVPR*, 2007. 2

[45] S. Winder and G. Hua. Picking the best daisy. In *CVPR*, 2009. 3, 4