Noname manuscript No.
(will be inserted by the editor)

Graph-based Particular Object Discovery

Oriane Siméoni · Ahmet Iscen · Giorgos Tolias · Yannis Avrithis · Ondřej Chum

Received: date / Accepted: date

Abstract Severe background clutter is challenging in many computer vision tasks, including large-scale image retrieval. Global descriptors, that are popular due to their memory and search efficiency, are especially prone to corruption by such a clutter. Eliminating the impact of the clutter on the image descriptor increases the chance of retrieving relevant images and prevents topic drift due to actually retrieving the clutter in the case of query expansion. In this work, we propose a novel salient region detection method. It captures, in an unsupervised manner, patterns that are both discriminative and common in the dataset. Saliency is based on a centrality measure of a nearest neighbor graph constructed from regional CNN representations of dataset images. The proposed method exploits recent CNN architectures trained for object retrieval to construct the image representation from the salient regions. We improve particular object retrieval on challenging datasets containing small objects.

1 Introduction

Particular object retrieval becomes very challenging when the object of interest is covering a small part of the image. In this case, the amount of relevant information is significantly reduced. Large objects might be partially occluded, while small objects are on a background that covers most of the image. A combination of both, occlusion and cluttered background, is not rare either. These conditions naturally arise from image acquisition and make naive approaches fail, including global template matching or semi-robust template matching [31].

Fig. 1 The saliency map (right) computed for an input image (left) based on common-structure analysis on *Instre* dataset. Background clutter and objects not relevant for this dataset are automatically removed. The image is represented only by the region detected on the saliency map.

Ideally, image descriptors should be extracted only from the relevant part of the image, suppressing the irrelevant clutter and occlusions. In this paper, we attempt to determine the regions containing the relevant information, as shown in Figure 1, in a fully unsupervised manner.

Methods based on robust matching of *hand-crafted local features* are naturally insensitive to occlusion and background clutter. The locality of the features allows to match small parts of images in regions containing the object of interest, while the incorrect matches are typically removed by robust geometric consistency check [35]. Methods based on efficient matching of vector-quantized local-feature descriptors were introduced in context of image retrieval by Sivic and Zisserman [47].

Retrieval methods based on descriptors extracted by *convolutional neural networks* (CNNs) have become popular because they combine good precision and recall, efficiency of the search, and reasonable memory footprint [5,40]. Deep neural networks are capable of learning, to some extent, what information in the image is relevant, which results in a good

Oriane Siméoni and Yannis Avrithis
Inria, Univ Rennes, CNRS, IRISA
E-mail: {oriane.simeni,ioannis.avrithis}@inria.fr

Ahmet Iscen, Giorgos Tolias and Ondřej Chum
VRG, FEE, CTU in Prague
E-mail: {ahmet.iscen,giorgos.tolias,chum}@cmp.felk.cvut.cz

performance even with global descriptors [51,4,19]. However, if the signal to noise ratio is low, *e.g*. the object is relatively small, multiple objects are present, *etc*., the global CNN descriptors fail [15,14].

A class of methods inspired by *object detection* have recently emerged. Instead of attempting to match the whole image to the query, the problem is changed to finding a rectangular region in the image that best matches the query [51, 42]. An inefficient search by sliding window is intractable for large collections of images. The exhaustive enumeration is approximated by similarity evaluation on a number of pre-selected regions. The regions are either selected geometrically to cover the whole image at different scales, as in R-MAC [51], or by considering the content by object or region proposal methods [42,48,11].

Another direction of suppressing irrelevant content is saliency detection [19,30]. For each image, a saliency map, that captures more general region shapes compared to (a small set of) rectangles, is first estimated. The contribution of each pixel (or region) is then proportional to the saliency of that location.

In this work we introduce a very simple pooling scheme that inherits the properties of both saliency detection and region based pooling and that, like all previous approaches, is applied to each image in the database *independently*. In addition, we investigate the use of the resulting regional representation for automatic, offline object discovery and suppression of background clutter, which considers the image collection *as a whole*. Unlike previous approaches, we do this in an unsupervised way. As a consequence, our representation takes two saliency detection steps into account. One that acts per image and depends solely on its content and another that considers the image collection as a whole and captures frequently appearing objects.

In both cases, we derive a *global* representation that outperforms comparable state-of-the-art methods in retrieving small objects on standard benchmarks, while the memory footprint and online cost is only a fraction of more powerful *regional* representations [40,15]. Moreover, we show that our representation benefits significantly from *query expansion* methods.

We make the following contributions:

1. We show that it is possible to select a set of candidate image regions based on CNN activations of off-the-shelf networks trained on retrieval tasks without bounding box annotations.
2. We obtain a global dataset-dependent "significance" of such regions via a neighborhood graph.
3. We represent the dataset by sampling and pooling CNN activations according to a dense saliency map of automatically discovered objects.
4. We thereby improve the state of the art on particular object retrieval, especially in a large scale dataset containing small objects.

Compared to the prior version of this work [45], we make the following improvements. We apply our method to the recent GeM representation [39]. We use a multi-scale representation in saliency computation. We analyze multiple graph centrality measures and validate their impact on detection quality using bounding box annotations. Finally, we evaluate using the recently revisited and challenging $\mathcal{R}$Oxford and $\mathcal{R}$Paris benchmarks [37].

Section 2 discusses our contributions against related work. Section 3 describes our methodology including our pooling scheme in Section 3.3 and our object discovery approach in Section 3.8. We present experimental results in Section 4 and draw conclusions in Section 5.

2 Related work

Local features and geometric matching offer an attractive way for retrieval systems to handle occlusions, clutter, and small objects [47,35,16]. One of their drawbacks is high query complexity and large storage cost; an image is typically represented by several thousands features. Many methods attempt to decrease the amount of indexed features by removing background clutter while maintaining the relevant information. The selection procedure is either applied independently per image or considers an image collection as a whole. Common examples of the former case are bursty feature detection [44], symmetry detection [50] or use of semantic segmentation [1,32]. The methods of the second category, are scalable enough to jointly process the whole collection and perform feature selection by the following assumption. A feature that repeats over multiple instances of the same object in the dataset is likely to appear in novel views of the object too. Representative cases are common object discovery [52,49], co-occurrence detection [8], or methods using GPS information [10,23].

The work by Turcot and Lowe [52] performs pairwise spatial verification on hand-crafted local features across all images and only indexes verified features. With an additional off-line cost, the on-line stage is sped up and the memory footprint is reduced. However, unique views of objects are not verified and thus discarded. In this work, we address a similar selection problem based on more powerful CNN-based representation rather than local features.

Recent advances on deep learning [3,51,19,12,38] dispense with the large memory footprint by using global descriptors and cast the problem of instance search as Euclidean nearest neighbor search. Nevertheless, background clutter and occlusion are better handled by regional representation. Regional descriptors significantly increase the performance

when they are indexed independently [40, 15] but this comes at a prohibited memory and computational cost for large scale scenarios. Region Proposal Networks (RPN) are applied either off-the-shelf [42] or after fine-tuning [48] for instance search. The RPNs reduce the number of regions per image only to the order of tens. Our work focuses on aggregating regional representation that keeps the complexity low but we rather detect regions around salient objects and objects that frequently appear in the dataset.

Recent work uses CNN activation statistics to construct a saliency map in an unsupervised manner. These methods consider each dataset image independently. CRoW [19] computes spatial and channel weights based on activation maps, and weighs each feature accordingly. Similarly, Laskar and Kannala [25] weigh each R-MAC region. Traditional methods are applied on top of CNN activations for the same purpose. Jeong *et al*. [17] use the Hessian-affine detector on activation maps to obtain repeatable regions. Pang *et al*. [34] use heat diffusion within an image to eliminate bursty features and keep the discriminative ones.

Another line of research trains a network to estimate the saliency map or find regions of interest. Zhu *et al*. [58] train an attention layer applied over multiple scales and use it for visual place recognition. This way, the network learns to discard background clutter. Similarly, Kim and Yoon [22] train a regional attention network to focus on important parts of an image. Jimenez *et al*. [18] construct saliency maps and perform region detection to construct global image vectors, which is similar to our goal. However, they employ object detectors trained on ImageNet classes, which does not apply to networks fine-tuned for retrieval on new classes. Mohedano *et al*. [28] evaluate deep and non-deep saliency models in order to detect regions of interest from an image.

All aforementioned methods either act per image without supervision or learn from a collection with ground-truth. By contrast, our method operates on the entire dataset jointly and at the same time is fully unsupervised.

The problem that this work is dealing with has been addressed previously in the literature but on different tasks. Common objects or regions in an image collection has been addressed in several different ways. Kim and Torralba [21] start from local features and detect region candidates using PageRank [33]. Bagon *et al*. [6] use *Local Self-Similarity Descriptors*, while Rubinstein *et al*. [41] use SIFT flow. More recently, Cho *et al*. [7] start from region proposals to discover dominant objects, while Kwak *et al*. [24] extend such a discovery process to videos.

3 Method

Like [52], our objective is to remove transient and non-distinctive objects as in Figure 1 and rather focus on objects appearing frequently in a dataset. Beginning with the activation map of a convolutional layer in a CNN, one would need access to a local representation to automatically discover such objects. On the other hand, knowing what these objects are would help forming a local representation by selecting regions depicting them, which appears to be a chicken-and-egg problem. Without an initial region selection, we risk "discovering" uninformative but frequently appearing "stuff"-like patches, for instance sky.

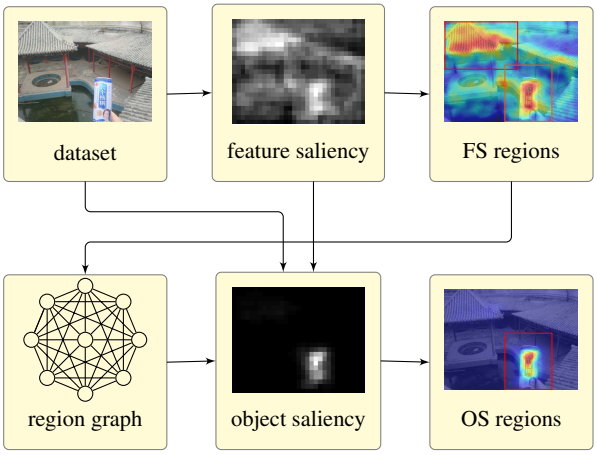

Fig. 2 Overview of our offline unsupervised process. On the top row, CNN activations of dataset images are used to extract a *feature saliency* map, on which a set of regions is detected. On the bottom row, a *centrality* measure is obtained per region from a region k-NN graph. Using this measure, a dense *object saliency* map is formed from the original CNN activations and the feature saliency. This map is focusing on objects automatically discovered in the dataset, with background clutter removed. Finally, another set of regions is detected on the object saliency map to extract descriptors and represent the dataset for retrieval.

3.1 Overview

Fortunately, it is possible to make an initial selection based on CNN activations alone, without any training and without bounding box annotations. As described in Section 3.3, the mechanism is inspired by CroW [19] and Grad-CAM [43] and generates a *feature saliency* map. This initiates our offline analysis illustrated in Figure 2. A small set of rectangular regions is detected per image from this map as discussed in Section 3.4. This first round of detection is applied independently per image and depends only on its content.

Each region in the dataset is associated to a feature saliency score and a visual descriptor, pooled from the activation map of the corresponding image, as discussed in Section 3.5. It is now possible to compute a *centrality* score per region, representing the "significance" of each region in the dataset. This is based on a region k-NN graph and is discussed in Sections 3.6 and 3.7.

Now, given a new image, we can infer the "significance" of every region from its nearest neighbors in the graph, yielding a dense *object saliency* map as discussed in Section 3.8. This is a regression problem and we suggest a non-parametric k-NN solution. Finally, we detect a small set of rectangular regions on this saliency map and extract a global descriptor to represent dataset images for retrieval, as discussed in Section 3.9. This second detection procedure takes into account all salient and repeating objects appearing in the dataset.

The entire process is fully unsupervised and only assumes on the-shelf networks trained on a classification or retrieval task without bounding box annotations.

3.2 Notation

We represent the activation map of a convolutional layer as a non-negative 3d tensor $A \in \mathbb{R}^{h\times w\times c}$ where h, w are the spatial resolution (height, width) and c is the number of feature channels. The set of valid spatial positions is $P := [h] \times [w]$[1] and the set of all rectangles with vertices in P is denoted by $\mathcal{R}$. By A_{pj} we represent an element of A at position $p \in P$ and channel $j \in [c]$. By $A_{\bullet j} \in \mathbb{R}^{h\times w}$ we denote the 2d feature map of A corresponding to channel $j \in [c]$. By $A_{p\bullet} \in \mathbb{R}^c$ we denote the vector containing all feature channels at position $p \in P$. By $\nu(\mathbf{x})$ we denote the ℓ^2-normalized vector $\mathbf{x}/\left\|\mathbf{x}\right\|_2$.

3.3 Feature saliency

Inspired by *cross-dimensional weighting and pooling* (CroW) [19] and *class activation mapping* (CAM) [56], we construct a 2d saliency map of an image based on a convolution activation of that image alone. Following CroW, we compute an idf-like weight per channel $\mathbf{b} \in \mathbb{R}^c$ with elements

$$b_j = \log\left(\frac{(\mathbf{a}+\epsilon)^\top \mathbf{1}}{a_j+\epsilon}\right) \tag{1}$$

for $j \in [c]$, where $\mathbf{a} := \frac{1}{wh}\sum_{p\in P} \mathbb{1}[A_{p\bullet}] \in \mathbb{R}^c$ is the average number of nonzero elements per channel. We then compute a weighted sum over channels

$$F := \sum_{j\in[c]} b_j A_{\bullet j} \tag{2}$$

Finally, we obtain the 2d *feature saliency* (FS) map $\hat{F} \in \mathbb{R}^{h\times w}$ by normalizing F according to [19]. Examples of feature saliency maps are presented in Section 4. Despite its simplicity, this kind of saliency can focus on objects of interest when the background is simple enough. It fails however in the presence of clutter. Contrary to CroW, we use the feature channel weights when computing the 2d spatial weights, amplifying channels with sparse activation. This order of summation is the same as in CAM. However, we are working with channel weights obtained by a sparsity property on any convolutional layer, without any assumption on the network topology. CAM on the other hand, assumes global average pooling followed by a fully connected layer mapping channels to classes and uses the parameters of this layer to obtain a saliency map per class.

[1] Here, $[i]$ is the set $\{1,\dots,i\}$ for $i \in \mathbb{N}$.

3.4 Region detection

We are given a 2d saliency map S, which can be either the feature saliency described in section 3.3 or the object saliency described in Section 3.8. We use an *expanding Gaussian mixture* (EGM) model [2] to detect a number of salient rectangular regions. This is a variant of expectation-maximization (EM) that iteratively performs local averaging (E- and M-steps) interleaved with a selection process (P-step) similar to non-maximum suppression (NMS). In doing so, it dynamically estimates the number of regions.

The original algorithm applies to point sets and isotropic Gaussian components. Here we extend it to functions, considering that a saliency map is a function $S : P \to \mathbb{R}$. We use it to fit a number of components, each modeling a rectangular region in 2d coordinate space. We also extend it to a diagonal covariance model, so that a rectangle is modeled by an axis-aligned ellipse.

In particular, given 2d saliency map $S \in \mathbb{R}^{h\times w}$, we represent it as a set of Gaussian functions $s_i : \mathbb{R}^2 \to \mathbb{R}$ with

$$s_i(\mathbf{x}) := S_{p_i}\mathcal{N}(\mathbf{x}|p_i, \sigma I_2) \tag{3}$$

for $i \in [\ell]$, $\mathbf{x} \in \mathbb{R}^2$ where $\mathcal{N}$ is the normal density, $\ell = |P|$ is the number of positions and we represent P as $\{p_1,\dots,p_\ell\}$. Here, σ is a *scale* parameter that determines how coarse or fine the region representation will be for the given saliency map. Similarly, we represent components as Gaussian functions $q_k : \mathbb{R}^2 \to \mathbb{R}$ with

$$q_k(\mathbf{x}) := \pi_k\mathcal{N}(\mathbf{x}|\mu_k, \Sigma_k) \tag{4}$$

for $k \in [m]$, $\mathbf{x} \in \mathbb{R}^2$, where m is the number of components and $\pi_k \in \mathbb{R}$, $\mu_k \in \mathbb{R}^2$ and $\Sigma_k \in \mathbb{R}^{2\times 2}$ are the mixing coefficient, mean and diagonal covariance matrix respectively of component k. Means represent region centers, while the (inverse) eigenvalues of covariance matrices represent heights and widths. We initialize components as $q_k \leftarrow s_k$ for $k \in [m]$, with $m \leftarrow \ell$. In the *expectation* (E)-step, we compute the *responsibility*

$$\gamma_{ik} \leftarrow \frac{\langle s_i, q_k\rangle}{\sum_{j\in[m]}\langle s_i, q_j\rangle} \tag{5}$$

of component $k \in [m]$ for sample $i \in [\ell]$, where $\langle f, g\rangle$ is the L^2 inner product of square-integrable functions $f, g : \mathbb{R}^d \to$

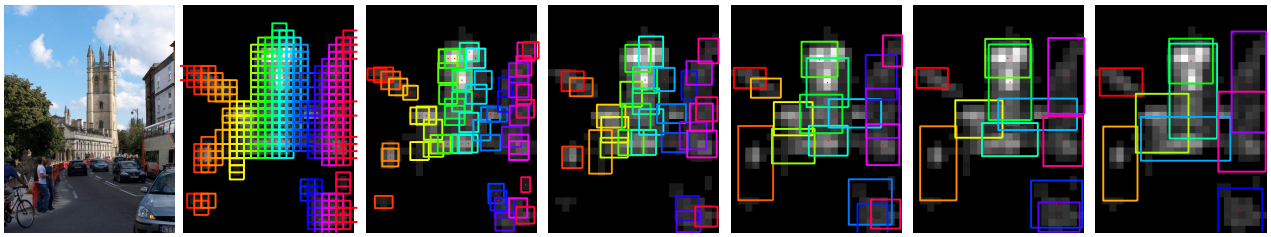

Fig. 3 Evolution of regions during EGM iterations on the feature saliency map of an image of *Magdalen tower* from Oxford buildings dataset, shown on the left. Below each image we display the iteration i and the number of regions m.

$\mathbb{R}$, computed in closed form for Gaussian functions [2]. In the *maximization* (M)-step, we update parameters as

$$\pi_k \leftarrow \frac{\ell_k}{\ell} \tag{6}$$

$$\mu_k \leftarrow \frac{1}{\ell_k} \sum_{i=1}^{n} \gamma_{ik} p_i \tag{7}$$

$$\Sigma_k \leftarrow \frac{1}{\ell_k} \sum_{i=1}^{n} \gamma_{ik} \operatorname{diag}(p_i - \mu_k)^{\circ 2} \tag{8}$$

where $\ell_k := \sum_{i=1}^{n} \gamma_{ik}$ is the effective number of points assigned to component k and $X^{\circ 2} := X \circ X$ is the Hadamard product power for a vector or matrix X.

Finally, in the *purge* (P)-step, similarly to NMS, we process components in descending order of mixing coefficient and we decide whether to keep a component or not depending on its overlap with the collection of previously kept components. Overlap is measured by a generalized responsibility function similar to (5), and again inner products are given in closed form [2]. This means that the number of components m is potentially reducing at each iteration.

Figure 3 shows how regions are formed during EGM iterations, starting from one small region centered on each spatial position. We get 4 clean regions on the ground truth building, as well as 6 regions on background objects, which, although less salient, cannot be removed based on the feature saliency alone.

3.5 Region pooling and whitening

Given a rectangular region $R \in \mathcal{R}$ of an image with feature saliency map $\hat{F} \in \mathbb{R}^{h \times w}$, we associate to it *feature saliency* $f := \mu_{\hat{F}}(R) \in \mathbb{R}$, where

$$\mu_{\hat{F}}(R) := \frac{1}{|R|} \sum_{p \in R} \hat{F}_p \tag{9}$$

is the average of 2d map $\hat{F}$ over R.

In addition, given the activation map $A \in \mathbb{R}^{h \times w \times c}$ of the same image, it is standard practice that a descriptor is obtained by pooling over R, for instance sum [4], weighted sum [19], max [3,51], or generalized-mean (GeM) [39] pooling. We adopt the latter choice to extract descriptor $\mathbf{z} := m_A(R) \in \mathbb{R}^c$, where

$$m_A(R) := \left(\frac{1}{|A_{q\bullet}|} \sum_{q \in R} (A_{q\bullet})^{\omega} \right)^{\frac{1}{\omega}} \tag{10}$$

is the generalized-mean of 3d tensor A over R along the spatial dimensions, and ω is a pooling parameter that is learned. This has been the basis of fine-tuning in [39] and produces a global description, referred to as GeM, in the special case where there is a single region $R = P$. In contrast, we detect a set of regions based on saliency maps in this work.

It is also standard practice to perform a sequence of post-processing steps including normalization, PCA and whitening [4,19,51,11,38]. We follow [39] in performing supervised whitening by simultaneous diagonalization [27]. In particular, given a descriptor $\mathbf{z} \in \mathbb{R}^c$, we ℓ^2-normalize, center, whiten, PCA-project and renormalize by function $w : \mathbb{R}^c \to \mathbb{R}^d$ to generate a d-dimensional descriptor:

$$w(\mathbf{z}) := \nu(U_{\mathcal{D}}^{\top} T_{\mathcal{D}}(\nu(\mathbf{z}) - \mu_{\mathcal{D}})). \tag{11}$$

Parameters $T_{\mathcal{D}}$, $U_{\mathcal{D}}$, $\mu_{\mathcal{D}}$ are trained on an independent labeled dataset $\mathcal{D} = \{Z, y\}$ where $Z = \{\mathbf{z}_1, \ldots, \mathbf{z}_n\} \subset \mathbb{R}^c$ and $y : [n]^2 \to \{0, 1\}$ maps a pair of descriptors to 1 if "positive" (similar) or 0 if "negative" (dissimilar). In particular, given set $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, define covariance matrix

$$C_l(X) := \frac{1}{|Y_l|} \sum_{(i,j) \in Y_l} (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^{\top} \tag{12}$$

where $Y_l := \{(i, j) \in [n]^2 : y(i, j) = l\}$. Then, the $c \times c$ *whitening matrix* $T_{\mathcal{D}} := C_1(\hat{Z})^{-\frac{1}{2}}$ is the inverse square root of the intra-class covariance matrix of $\hat{Z} = \nu(Z)$. The *PCA matrix* $U_{\mathcal{D}}$ is the $c \times d$ matrix having as columns the top d eigenvectors of $C_0(T_{\mathcal{D}}\hat{Z})$, that is, the inter-class covariance matrix of the whitened counterpart of $\hat{Z}$. Finally, the *mean vector* is $\mu_{\mathcal{D}} := \frac{1}{n} \sum_{i \in [n]} \nu(\mathbf{z}_i)$.

3.6 Graph construction

Given an image dataset, we assume here a set of regions $\{R_1, \dots, R_n\}$ are detected from the saliency maps (Section 3.4), a *feature saliency* vector $\mathbf{f} := (f_1, \dots, f_n) \in \mathbb{R}^n$ is computed with the corresponding average saliency per region in (9), and a set of descriptors $V := \{\mathbf{v}_1, \dots, \mathbf{v}_n\} \subset \mathbb{R}^d$ are extracted from the activation maps, whitened and normalized per region (Section 3.5).

Based on the above information, we construct a k-NN graph on those regions in order to compute a global *centrality* score per region as discussed in Section 3.7, which enables us to form an *object saliency* map on a new image, described in Section 3.8. Approximate techniques for k-NN graph construction [9] can be used to handle large-scale databases.

We construct a weighted undirected graph having the set of descriptors V as vertices. Following [15], the edge weights are defined according to *mutual k-nearest neighbors* (NN) in the descriptor space. In particular, given descriptors $\mathbf{v}, \mathbf{u} \in \mathbb{R}^d$, we measure their *similarity* by $s(\mathbf{v}, \mathbf{u}) = \max(0, \mathbf{v}^\top \mathbf{u})^\beta$, where exponent $\beta > 0$ is a parameter. We define the sparse symmetric nonnegative *adjacency matrix* $W \in \mathbb{R}^{n \times n}$ with elements w_{ij} being $s(\mathbf{v}_i, \mathbf{v}_j)$ if $\mathbf{v}_i, \mathbf{v}_j$ are mutual k-NN in V and zero otherwise.

We define the $n \times n$ *degree matrix* $D := \mathrm{diag}(W\mathbf{1})$ where $\mathbf{1} \in \mathbb{R}^n$ is the all-ones vector, and the *symmetrically normalized adjacency matrix*

$$\mathcal{W} := D^{-1/2} W D^{-1/2}, \tag{13}$$

with the convention $0/0 = 0$. Following [15, 14], we define the $n \times n$ matrices $L_\alpha := (D - \alpha W)/(1 - \alpha)$ and

$$\mathcal{L}_\alpha := D^{-1/2} L_\alpha D^{-1/2} = (I - \alpha \mathcal{W})/(1 - \alpha), \tag{14}$$

where $\alpha \in [0, 1)$. Both are positive-definite [15, 14].

3.7 Graph centrality

With the above definitions in place, the objective is to compute a vector $\mathbf{g} \in \mathbb{R}^n$ where each element g_i represents the significance of vertex $\mathbf{v}_i$ in the graph, for $i \in [n]$. We choose *graph centrality* [26] to estimate $\mathbf{g}$. Centrality is a global measure of significance of vertices in a graph. Different definitions exist, each one reflecting a different kind of vertex importance. Herein, we consider a number of centrality measures on adjacency matrix W, which we then evaluate in the experimental section.

Degree centrality is the simplest one to define and to compute. It is defined as the degree of each vertex, *i.e.* the diagonal of degree matrix D. Large value means that the vertex is connected to many other vertices with edges of large weight (similarity).

Eigenvector centrality, also known as eigencentrality, corresponds to the dominant eigenvector of W. Centrality value of vertex $\mathbf{v}_i$ is given by the i-th element of this eigenvector. Large value means that the vertex is connected to many vertices which themselves have high centrality.

PageRank centrality is maybe the most well-known [33] centrality measure, and is a variant of the eigencentrality. It is given by the dominant left eigenvector of the transition matrix $\alpha D^{-1} W + (1 - \alpha) J/n$, where J is an $n \times n$ matrix of ones. It is efficiently computed with power iteration. It models a random walk on the graph and its score represents the probability of visiting a vertex.

Betweenness centrality reflects the number of times a vertex is part of the shortest path between any two other vertices of the graph. In contrast to all previously mentioned measures that are computed based on edge importance (similarity w_{ij}), it is computed based on an edge cost. In this case, we are using Euclidean distance $||\mathbf{v}_i - \mathbf{v}_j||$.

Closeness centrality is inversely proportional to the average length of the shortest path to all other nodes. Closeness centrality for vertex $\mathbf{v}_i$ is equal to $\sum_{\mathbf{v}_j \neq \mathbf{v}_i} \frac{n-1}{|\mathbf{v}_i \to \mathbf{v}_j|}$, where $|\mathbf{v}_i \to \mathbf{v}_j|$ is the length of the shortest path between $\mathbf{v}_i$ and $\mathbf{v}_j$. Similarly to betweenness, closeness applies to edge cost, and we use the Euclidean distance.

Katz centrality [20] is given by the solution $\mathbf{g}^* \in \mathbb{R}^n$ of the linear system

$$\mathcal{L}_\alpha \mathbf{g} = \mathbf{1}. \tag{15}$$

As in [15], we solve this system by the *conjugate gradient method* (CG) [29]. Any method would be equally appropriate because this is computed just once offline.

It is interesting to observe that in [57], given a vertex $\mathbf{v}_i \in V$ as a *query*, the linear system $\mathcal{L}_\alpha \mathbf{x} = \mathbf{e}_i$ is considered instead, where $\mathbf{e}_i$ is the i-th canonical basis vector. A random walk iteration is applied, which (slowly) converges to $\mathbf{x}^*$, where each element x^*_j represents the "similarity" of vertex $\mathbf{v}_j$ to the query $\mathbf{v}_i$. In [15], the same linear system is rather solved directly and more efficiently with CG. One may then interpret (15) as follows. If we denote by $\mathbf{x}^*_i$ the solution to $\mathcal{L}_\alpha \mathbf{x}_i = \mathbf{e}_i$ for $i \in [n]$, then the solution of (15) is $\mathbf{g}^* = \sum_{i \in [n]} \mathbf{x}^*_i$. It follows that each element g^*_j measures the "expected similarity" of $\mathbf{v}_j$ to a query vertex of the graph for $j \in [n]$, averaged over all query vertices.

In fact, Katz centrality was introduced as such a global measure before being adapted by a *boundary condition* to measure relevance to individual vertices by Hubbell [13]. This work has a long history before being rediscovered *e.g.* by [33, 57], as summarized in the study of *spectral ranking* [53].

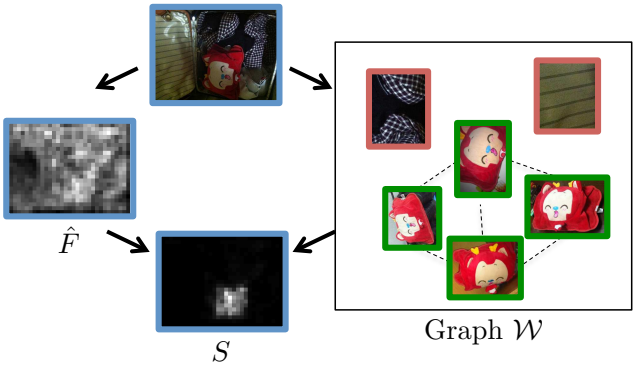

Fig. 4 Computing the *object saliency* map S of an image from Instre dataset (top), as defined in (16). For each patch, its neighbors in the graph (right) are found. Common patterns with high centrality in green outline, outliers with low centrality in red. S (bottom) then focuses on patches similar to common patterns and combines with feature saliency $\hat{F}$ (left).

3.8 Saliency map construction

Given the region descriptor set V, the region saliency vector $\mathbf{f}$ and the associated centrality vector $\mathbf{g}^*$ of an entire dataset, the problem is to construct a new saliency map $S \in \mathbb{R}^{h \times w}$ for an image in the dataset. The image is represented by its activation map $A \in \mathbb{R}^{h \times w \times c}$. Since this saliency is based on regions or patterns appearing frequently in the dataset, which are commonly associated to repeating objects, we call it *object saliency* (OS).

We compute S by a sliding window iteration over each position $p \in P$. The saliency value S_p at p is found as a linear combination of the centrality values of the nearest neighbors in V of a patch centered at p. In particular, we consider a square patch R_p of side a centered at p. We compute the vector $\mathbf{u}_p := w(m_A(R_p)) \in \mathbb{R}^d$ by max-pooling over R_p, whitening and normalizing as discussed in Section 3.5. If N_p is the set of indices of the k-NN of $\mathbf{u}_p$ in V, we compute S_p as

$$S_p := \hat{F}_p^\theta \sum_{i \in N_p} s(\mathbf{v}_i, \mathbf{u}_p) f_i^\theta g_i^*. \tag{16}$$

That is, each neighboring region descriptor $\mathbf{v}_i$ is weighted by its similarity to patch descriptor $\mathbf{u}_p$, its feature saliency f_i and its centrality g_i^*, while the entire sum is scaled by the feature saliency $\hat{F}_p$ at the current position p of the image being considered. Exponent θ controls the relative importance of feature saliency of the current image and neighbors compared to centrality. The object saliency computation is illustrated in Figure 4. Looking at the input image and is feature saliency map $\hat{F}$ alone, it is not evident which is the object of interest and which is clutter. This is only found by discovering other instances of the same object in the dataset, as represented by the graph.

3.9 Saliency-based epresentation

The object saliency map S highlights patterns that appear frequently in the dataset, with the background clutter removed. It is only natural then to apply the same method described in Section 3.4 to this map in order to detect a small number of regions per image. Unlike the regions detected from the feature saliency map $\hat{F}$, these new regions are more likely to appear in a new image. For the purpose of evaluation, we investigate both saliency maps.

For each region R detected from a saliency map ($\hat{F}$ or S) in a dataset image with activation map A, we apply GeM pooling and ℓ^2-normalization. All descriptors are then summed and the resulting descriptor is whitened with $w : \mathbb{R}^c \to \mathbb{R}^d$ as described in Section 3.5. We apply whitening on the aggregated vector and not separately per region. This follows the spirit of R-MAC [51], but performs GeM pooling instead of max, and the regions are detected in the saliency map rather than being uniformly distributed. This yields a global image representation in $\mathbb{R}^d$.

Pooling based on saliency is in fact the idea explored in CroW [19], but here we follow the nonlinear two-level pooling of R-MAC (first within a region, then over regions) rather than the one-level sum of CroW. This is more powerful and has also been the basis of fine-tuning in [12, 39].

3.10 Multi-scale representation

Multi-scale descriptor extraction with CNNs is becoming standard practice [12, 39]. It consists of the following steps. Re-sample the image to multiple scales, feed each scale separately to the network, obtain an ℓ_2-normalized vector per scale, sum-pool over scales, re-normalize, whiten, and re-normalize. We follow the same principle, but use the representation of Section 3.9 per scale. Saliency maps are simply constructed independently per scale. Finally, pooling over scales is done with the generalized-mean as in [39].

We additionally adopt the multi-scale concept for graph construction. Regions are detected independently per scale and the corresponding descriptors form new vertices.

4 Experiments

We apply the proposed representation on image retrieval. In particular, we have two variants of our method that both use the region detection described in Section 3.4. The saliency map which the detection is performed on is different in each case. FS.EGM uses the feature saliency map described in Section 3.3, and OS.EGM uses the object saliency map described in Section 3.4. The former is image specific, while the latter is both image and database specific.

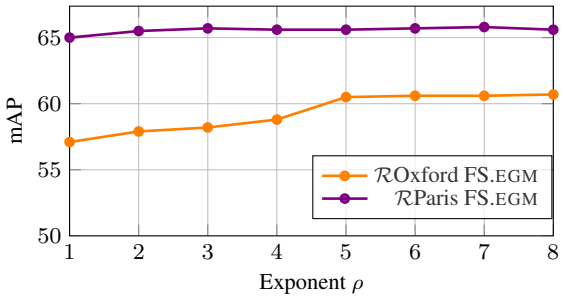

Fig. 5 mAP on $\mathcal{R}$Oxford and $\mathcal{R}$Paris versus saliency exponent ρ in FS.EGM.

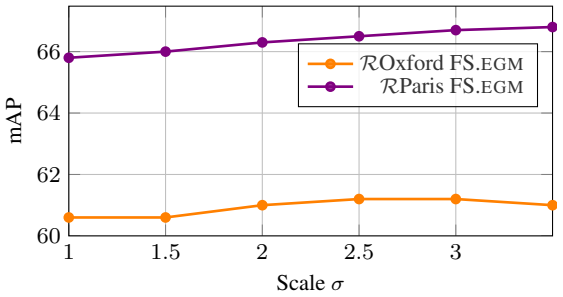

Fig. 6 mAP on $\mathcal{R}$Oxford and $\mathcal{R}$Paris versus EGM scale parameter σ in FS.EGM.

4.1 Experimental setup

Test sets. We use three image retrieval benchmarks for our experiments. We evaluate on the revisited benchmark [37] of the Oxford Buildings [35] and Paris [36] datasets. We refer to the revisited datasets as $\mathcal{R}$Oxford and $\mathcal{R}$Paris respectively. We also use the more recently introduced Instre [54] dataset. Instre contains around 27k images of small objects in cluttered scenes while objects appear with different variations, such as rotation, occlusion and scale changes, making it a challenging case. We use the evaluation protocol introduced in [15] for Instre. Search performance in all datasets is measured with mean average precision (mAP).

Image Representation. We use the global image representation as described in Section 3.9. This reduces image similarity to cosine, which is common practice [51]. Feature extraction is performed with the VGG network [46] with GeM pooling that is fine-tuned for image retrieval [39]. We use supervised whitening [38,39] as described in Section 3.5. Compared to global GeM pooling, our variants are different in that regions are detected from salient and repeating objects, while aggregation and whitening is identical. Detection is applied to dataset images only, while we use the provided bounding boxes on the query side.

Implementation Details. To simplify region detection, each saliency map is masked above threshold τ and element-wise raised to exponent ρ before detection, which removes the weakest regions and increases the contrast between foreground and background objects. We fix threshold $\tau = 0.01$ in order to remove noise from saliency maps. We set exponent $\rho = 1$ and scale parameter $\sigma = 1$ before any parameter tuning is performed. We determine OS parameter θ in (16) by visual inspection of OS and set $\theta = 3$ throughout our experiments. We perform our experiments on a 16-core Intel Xeon 2.00GHz CPU. It takes 36s to create the graph on Instre, while centrality computation takes negligible amount of time. Saliency computation and detection per image takes 0.02s for FS and 0.23s for OS.

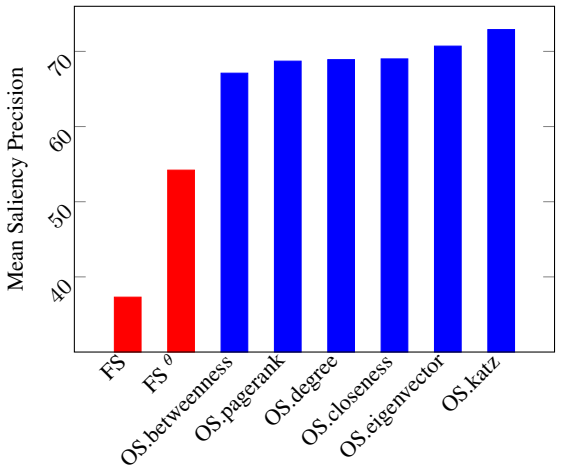

Fig. 7 Saliency precision evaluation of FS and OS created with several centralities maps measured on all images of Instre.

4.2 Parameter tuning

In this section, we show the impact of FS.EGM and OS.EGM detection parameters on the retrieval performance. We tune the parameters on $\mathcal{R}$Oxford and $\mathcal{R}$Paris, while showing that their impact is similar on both datasets.

Feature saliency detection is evaluated first by FS.EGM, while we do not compute object saliency and OS.EGM yet. Figure 5 shows the effect of ρ on FS, which controls the contrast of the the saliency map. We observe that large ρ is needed to remove as much clutter as possible from the noisy FS activations. We set $\rho = 7$ for the rest of our experiments. Scale σ is used during EGM sampling as explained in Section 3.4. Its impact in performance on FS is shown in Figure 6. Setting $\sigma = 2.5$ results in good performance and regions that are large enough for FS.EGM.

Centrality measure We use the different centrality measurements presented in Section 3.7, compute the OS map, and evaluate the quality of saliency maps. We use Instre's bounding box annotation as ground truth and measure *saliency precision* as the sum of saliency map values inside the bounding box normalized by the total sum over the entire image.

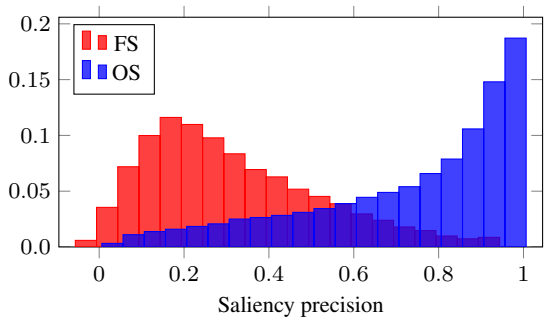

Fig. 8 Histogram of saliency precision for FS and OS maps measured on all images of Instre.

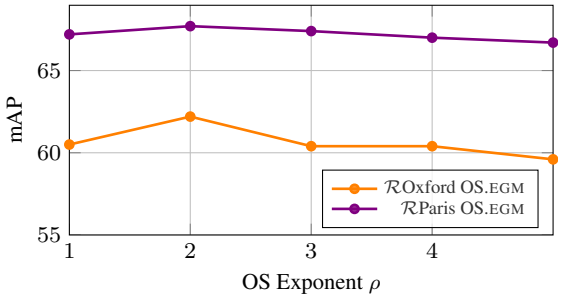

Fig. 9 mAP on $\mathcal{R}$Oxford and $\mathcal{R}$Paris versus saliency exponent ρ in OS.EGM.

High precision means that a saliency map captures the repeating object and discards the background.

We present the average saliency precision over the entire Instre dataset in Figure 7. The comparison includes saliency maps created with FS and FS $^{\theta}$, since the latter is used in (16). Improvements brought by OS compared to FS are significant regardless of the centrality measurement, while Katz centrality gives the best precision. Figure 8 shows the distribution of precision over all Instre images for FS and OS. We use Katz centrality measurement for the rest of our experiments and simply refer to it as OS.

Object saliency detection is then evaluated for retrieval. Now, the feature saliency parameters are fixed, Katz centrality is selected and EGM detection is applied on the new saliency map. We observe that OS behaves quite differently compared to FS, because foreground objects are much cleaner. The impact of parameters σ and ρ is shown in Figures 9 and 10 respectively. It is remarkable that a much lower exponent is needed in this case. We choose $\rho = 2$ and $\sigma = 2.5$.

Visual examples for saliency maps and detections for FS.EGM and OS.EGM are shown in Figure 11. In all cases, OS is cleaner and focuses on objects that FS cannot discriminate.

4.3 Comparison to other methods

We compare our method against GeM descriptor [39]. We additionally evaluate the multi-scale variants for GeM, FS.EGM and OS.EGM, as described in Section 3.10. All methods are tested with k-NN search and global diffusion [15], which is a method for query expansion (QE) and is known to significantly improve performance. Results are given in Table 1. We report results for both Medium and Hard settings on $\mathcal{R}$Oxford and $\mathcal{R}$Paris.

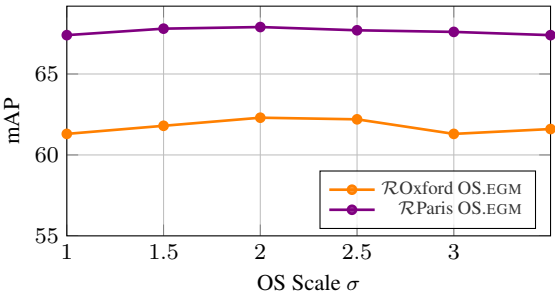

Fig. 10 mAP on $\mathcal{R}$Oxford and $\mathcal{R}$Paris versus EGM scale parameter σ in OS.EGM.

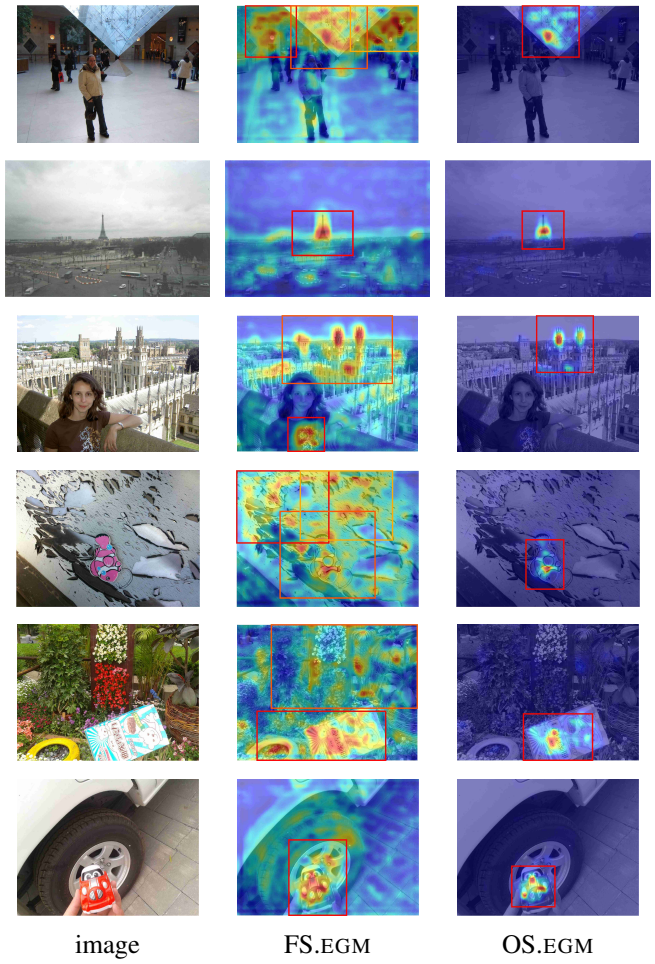

Fig. 11 Examples of images from $\mathcal{R}$Oxford (first 2 rows) and Instre (last 3 rows) datasets, along with smoothed FS and OS maps superimposed on the images and regions detected by EGM, in red.

		Medium		Hard	
Method	Instre	$\mathcal{R}$Oxford	$\mathcal{R}$Paris	$\mathcal{R}$Oxford	$\mathcal{R}$Paris
Single Scale					
GeM [39]	54.2	60.8	67.0	33.3	42.1
FS.EGM	54.6	61.2	66.5	33.4	41.7
OS.EGM	58.3	62.2	67.7	34.9	43.4
Single Scale + QE					
GeM [39]	73.3	69.3	83.9	42.3	71.8
FS.EGM	72.8	71.0	84.1	42.3	71.4
OS.EGM	76.5	**72.2**	83.8	**46.5**	69.9
Multi Scale					
GeM [39]	57.0	62.0	69.3	33.7	44.3
FS.EGM	57.7	63.0	68.7	34.5	43.9
OS.EGM	61.3	64.2	69.9	35.9	46.1
Multi Scale + QE					
GeM [39]	75.0	69.3	83.9	41.1	**73.9**
FS.EGM	74.6	71.0	84.1	40.6	72.5
OS.EGM	**77.4**	69.0	**85.4**	41.9	72.3

Table 1 mAP comparison of our method against baselines on all tested datasets. QE refers to query expansion by diffusion [15].

FS.EGM does not always improve the performance, since objects captured are not necessarily relevant for the particular dataset. This is what OS.EGM captures and boosts the search performance, especially on Instre. We outperform GeM pooling in all datasets and scenarios, except for $\mathcal{R}$Paris- Hard with query expansion. Note that we use the default diffusion parameters which are tuned on GeM-like descriptors. OS.EGM provides larger improvements on Instre, which is more challenging due to small objects and severe background clutter. This is exactly where our detection is essential.

There are several other approaches that deal with region detection or saliency masks. Their results are not directly comparable, because they are not evaluated on $\mathcal{R}$Oxford, but Oxford5k. Therefore, they are not included in Table 1. Nevertheless, we outperform their reported results. We achieve 86.6 on Oxford5k with OS.EGM and no query expansion. Salvador *et al*. [42] use the off-the-shelf VGG and fine-tune RPN on the test set. Without using query expansion, they obtain 71.0 on Oxford5k. Similarly, Jimenez *et al*. [18] learn class weights and apply them on the activation maps of off-the-shelf VGG and achieve 73.6 on Oxford5k. Song *et al*. [48] train on different datasets, and achieve 78.3 on Oxford5k. The results obtained by learning a saliency mask in [30] are not comparable since spatial verification with local features is always applied in the end. Zheng *et al*. [55] achieve 83.4 with regional representation on Oxford5k. They employ both CNN and local features, while we only rely on CNN and much more compact representation. Finally, according to our knowledge, [28] is the only work that evaluates on Instre, which is rather challenging due to small objects. However, their results are not directly comparable as they use a different CNN model (ResNet101).

Region cross-matching methods [40] represent an image with multiple vectors, sacrificing memory footprint and complexity for accuracy. In particular, the memory is linear in the number of regions, while the complexity is quadratic. We compare our global representation with region cross-matching (R-Match) and regional diffusion [15] in Figure 12. We sample regions uniformly at 3 scales as in R-MAC [51] and apply GeM pooling separately for each region. Different numbers of regions are obtained by GMM reduction, exactly as in [15].

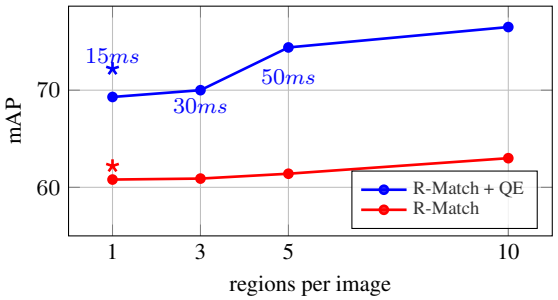

Fig. 12 mAP comparison of our global OS.EGM ($\star$) to R-Match with uniformly sampled regional descriptors, with and without diffusion on $\mathcal{R}$Oxford (Medium setup). Text labels refer to query time.

Compared to regional descriptors, we require about 4 times less memory to achieve the same performance. The runtime complexity gain is in the order of 4^2, which holds for the case of R-Match and also for the first part of diffusion where Euclidean nearest neighbors are found. The diffusion complexity is O(m), where m is the number of non-zero entries of the graph. We found that m is 3.7 times smaller in our case and our measurements of actual query timings agree with this ratio.

5 Conclusions

We propose a region detection approach that is dataset specific but requires no supervision. It captures not only salient objects by considering each image individually but also frequently appearing ones by considering the dataset as a whole. As a result, we avoid separate indexing of regional descriptors and construct a global descriptor by pooling over data-dependent regions, which performs well under background clutter and severe occlusions. We demonstrate that this approach is effective in particular object retrieval where background clutter is a common problem.

Acknowledgments This work was supported by the OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics". The Tesla K40 used for this research was donated by the NVIDIA Corporation.

References

1. R. Arandjelović and A. Zisserman. Visual vocabulary with a semantic twist. In *ACCV*, 2014. 2

2. Y. Avrithis and Y. Kalantidis. Approximate gaussian mixtures for large scale vocabularies. In *ECCV*, pages 15–28. Springer, 2012. 4, 5
3. H. Azizpour, A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson. From generic to specific deep representations for visual recognition. *arXiv preprint arXiv:1406.5774*, 2014. 2, 5
4. A. Babenko and V. Lempitsky. Aggregating deep convolutional features for image retrieval. In *ICCV*, 2015. 2, 5
5. A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. In *ECCV*, 2014. 1
6. S. Bagon, O. Brostovski, M. Galun, and M. Irani. Detecting and sketching the common. In *CVPR*, 2010. 3
7. M. Cho, S. Kwak, C. Schmid, and J. Ponce. Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals. In *CVPR*, 2015. 3
8. O. Chum and J. Matas. Unsupervised discovery of co-occurrence in sparse high dimensional data. In *CVPR*, June 2010. 2
9. W. Dong, M. Charikar, and K. Li. Efficient k-nearest neighbor graph construction for generic similarity measures. In *WWW*, March 2011. 6
10. S. Gammeter, L. Bossard, T. Quack, and L. V. Gool. I know what you did last summer: Object-level auto-annotation of holiday snaps. In *ICCV*, 2009. 2
11. A. Gordo, J. Almazan, J. Revaud, and D. Larlus. Deep image retrieval: Learning global representations for image search. *ECCV*, 2016. 2, 5
12. A. Gordo, J. Almazan, J. Revaud, and D. Larlus. End-to-end learning of deep visual representations for image retrieval. *arXiv preprint arXiv:1610.07940*, 2016. 2, 7
13. C. H. Hubbell. An input-output approach to clique identification. *Sociometry*, 1965. 6
14. A. Iscen, Y. Avrithis, G. Tolias, T. Furon, and O. Chum. Fast spectral ranking for similarity search. *CVPR*, 2018. 2, 6
15. A. Iscen, G. Tolias, Y. Avrithis, T. Furon, and O. Chum. Efficient diffusion on region manifolds: Recovering small objects with compact cnn representations. In *CVPR*, 2017. 2, 3, 6, 8, 9, 10
16. H. Jégou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. *IJCV*, 87(3):316–336, February 2010. 2
17. D.-J. Jeong, S. Choo, W. Seo, and N. I. Cho. Regional deep feature aggregation for image retrieval. In *ICASSP*, 2017. 3
18. A. Jimenez, J. M. Alvarez, and X. Giro-i Nieto. Class-weighted convolutional features for visual instance search. *BMVC*, 2017. 3, 10
19. Y. Kalantidis, C. Mellina, and S. Osindero. Cross-dimensional weighting for aggregated deep convolutional features. In *arXiv*, 2015. 2, 3, 4, 5, 7
20. L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953. 6
21. G. Kim and A. Torralba. Unsupervised detection of regions of interest using iterative link analysis. In *NIPS*, 2009. 3
22. J. Kim and S.-E. Yoon. Regional attention based deep feature for image retrieval. *BMVC*, 2018. 3
23. J. Knopp, J. Sivic, and T. Pajdla. Avoiding confusing features in place recognition. In *ECCV*, 2010. 2
24. S. Kwak, M. Cho, I. Laptev, J. Ponce, and C. Schmid. Unsupervised object discovery and tracking in video collections. In *CVPR*, 2015. 3
25. Z. Laskar and J. Kannala. Context aware query image representation for particular object retrieval. In *Scandinavian Conference on Image Analysis*, 2017. 3
26. N. MEJ. *Networks: an introduction*. Oxford University Press, Oxford, 2010. 6
27. K. Mikolajczyk and J. Matas. Improving descriptors for fast tree matching by optimal linear projection. In *CVPR*, 2007. 5
28. E. Mohedano, K. McGuinness, X. Giro-i Nieto, and N. E. O'Connor. Saliency weighted convolutional features for instance search. *arXiv preprint arXiv:1711.10795*, 2017. 3, 10
29. J. Nocedal and S. Wright. *Numerical optimization*. Springer, 2006. 6
30. H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han. Large-scale image retrieval with attentive deep local features. In *arXiv*, 2016. 2, 10
31. A. Oliva and A. Torralba. Building the gist of a scene: The role of global image features in recognition. *Progress in brain research*, 155:23–36, 2006. 1
32. D. Omercevic, R. Perko, A. T. Targhi, J.-O. Eklundh, and A. Leonardis. Vegetation segmentation for boosting performance of mser feature detector. In *Computer Vision Winter Workshop*, 2008. 2
33. L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: bringing order to the web. 1999. 3, 6
34. S. Pang, J. Ma, J. Xue, J. Zhu, and V. Ordonez. Image retrieval using heat diffusion for deep feature aggregation. *arXiv preprint arXiv:1805.08587*, 2018. 3
35. J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *CVPR*, June 2007. 1, 2, 8
36. J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *CVPR*, June 2008. 8
37. F. Radenović, A. Iscen, G. Tolias, Y. Avrithis, and O. Chum. Revisiting oxford and paris: Large-scale image retrieval benchmarking. In *CVPR*, 2018. 2, 8
38. F. Radenović, G. Tolias, and O. Chum. CNN image retrieval learns from bow: Unsupervised fine-tuning with hard examples. In *ECCV*, 2016. 2, 5, 8
39. F. Radenović, G. Tolias, and O. Chum. Fine-tuning cnn image retrieval with no human annotation. *IEEE Trans. PAMI*, 2018. 2, 5, 7, 8, 9, 10
40. A. S. Razavian, J. Sullivan, S. Carlsson, and A. Maki. Visual instance retrieval with deep convolutional networks. *ITE Transactions on Media Technology and Applications*, 4:251–258, 2016. 1, 2, 3, 10
41. M. Rubinstein, A. Joulin, J. Kopf, and C. Liu. Unsupervised joint object discovery and segmentation in internet images. In *CVPR*, 2013. 3
42. A. Salvador, X. Giró-i Nieto, F. Marqués, and S. Satoh. Faster r-cnn features for instance search. In *CVPRW*, 2016. 2, 3, 10
43. R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra. Grad-CAM: Why did you say that? visual explanations from deep networks via gradient-based localization. *arXiv preprint arXiv:1610.02391*, 2016. 3
44. M. Shi, Y. Avrithis, and H. Jegou. Early burst detection for memory-efficient image retrieval. In *CVPR*, 2015. 2
45. O. Simeoni, A. Iscen, G. Tolias, Y. Avrithis, and O. Chum. Unsupervised object discovery for instance recognition. In *WACV*, 2018. 2
46. K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2014. 8
47. J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003. 1, 2
48. J. Song, T. He, L. Gao, X. Xu, and H. T. Shen. Deep region hashing for efficient large-scale instance search from images. In *arXiv*, 2017. 2, 3, 10
49. G. Tolias, Y. Avrithis, and H. Jégou. Image search with selective match kernels: aggregation across single and multiple images. *IJCV*, 2016. 2
50. G. Tolias, Y. Kalantidis, and Y. Avrithis. Symcity: Feature selection by symmetry for large scale image retrieval. In *ACM Multimedia*, 2012. 2
51. G. Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of cnn activations. *ICLR*, 2016. 2, 5, 7, 8, 10
52. P. Turcot and D. G. Lowe. Better matching with fewer features: The selection of useful features in large database recognition problems. In *ICCVW*, 2009. 2, 3
53. S. Vigna. Spectral ranking. *arXiv preprint arXiv:0912.0238*, 2009. 6

54. S. Wang and S. Jiang. Instre: a new benchmark for instance-level object retrieval and recognition. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 11:37, 2015. 8
55. L. Zheng, S. Wang, J. Wang, and Q. Tian. Accurate image search with multi-scale contextual evidences. *IJCV*, 120(1):1–13, 2016. 10
56. B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *cvpr*, June 2016. 4
57. D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *NIPS*, 2003. 6
58. Y. Zhu, J. Wang, L. Xie, and L. Zheng. Attention-based pyramid aggregation network for visual place recognition. *arXiv preprint arXiv:1808.00288*, 2018. 3