Unsupervised deep object discovery for instance recognition

Oriane Siméoni[1] Ahmet Iscen[1] Giorgos Tolias[2] Yannis Avrithis[1] Ondřej Chum[2]
[1]Inria Rennes [2]VRG, FEE, CTU in Prague
{oriane.simeoni,ahmet.iscen,ioannis.avrithis}@inria.fr
{giorgos.tolias,chum}@cmp.felk.cvut.cz

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 clutter. Eliminating the impact of the clutter on the image descriptor increases the chance of retrieving relevant images as well as preventing topic drift by 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. The descriptors derived on the salient regions improve particular object retrieval, most noticeably in a large collections 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 [25].

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 [28]. Methods based on efficient matching of vector-quantized local-feature descriptors were introduced in context of image retrieval by Sivic and Zisserman [35].

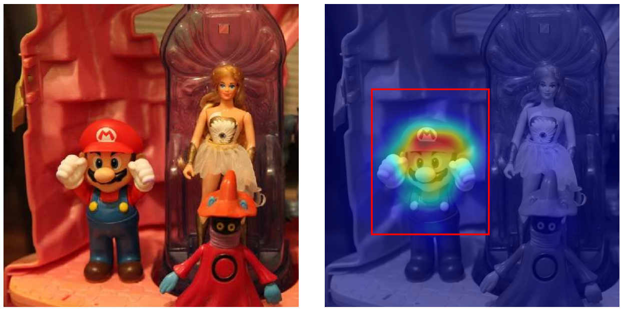

Figure 1. The saliency map (right) computed for an input image (left) based on common-structure analysis on *Instre* dataset. Background clutter and objects not relevant for this dataset are automatically removed. The image is represented only by the region detected on the saliency map.

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, 30]. Deep neural networks are capable of learning, to some extent, what information in the image is relevant, which results in a good performance even with global descriptors [39, 4, 17]. 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 [12, 11].

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 [39, 31]. 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 [39], or by considering the content by object or region proposal methods [31, 36, 8].

Another direction of suppressing irrelevant content is saliency detection [17, 24]. 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 compared to more powerful *regional* representations [30, 12]. Moreover, we show that our representation benefits significantly from *query expansion* methods.

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 [35, 28, 13]. 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 [33], symmetry detection [38] or use of semantic segmentation [1, 26]. 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 [40, 37], co-occurrence detection [6], or methods using GPS information [7, 19].

The work by Turcot and Lowe [40] 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, 39, 17, 9, 29] 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 [30, 12] but this comes at a prohibited memory and computational cost for large scale scenarios. Region Proposal Networks (RPN) are applied either off-the-shelf [31] or after fine-tuning [36] 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. Jimenez *et al.* [15] construct saliency maps and perform region detection to construct global image vectors, as we also do. However, they employ generic object detectors trained on ImageNet and this makes the method not applicable with fine-tuned networks which provide the best performance. The Hessian-affine detector is used on CNN activations to detect repeatable regions [14]. The major benefit in this work, though, comes from second order pooling and higher dimensional descriptors.

Saliency maps are another way to handle clutter and occlusions. Once more, there exist both examples of computation in an unsupervised manner [17, 20] or learned [24, 16] and applied per image afterwards. Our approach generates saliency maps in a fully unsupervised way that capture both salient objects on single images but also repeating objects appearing in a particular image collection.

3. Method

Like [40], our objective is to remove transient and nondistinctive 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.

3.1. Overview

Fortunately, it is possible to make an initial selection based on CNN activations alone, without any training and

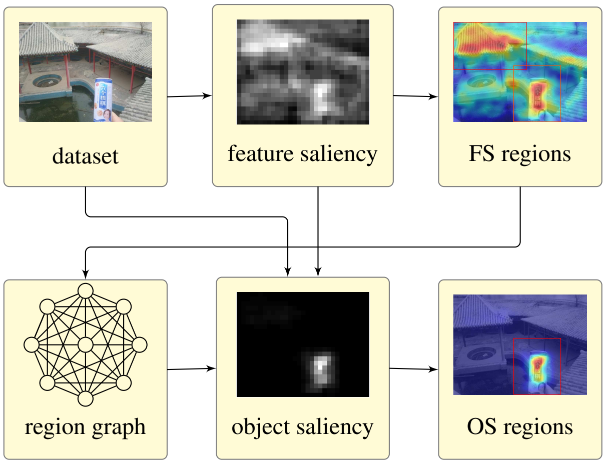

Figure 2. Overview of our offline unsupervised process. On the top row, CNN activations of dataset images are used to extract a *feature saliency* map, on which a set of regions is detected. On the bottom row, a *centrality* measure is obtained per region from a region k-NN graph. Using this measure, 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.

without bounding box annotations. As described in Section 3.3, the mechanism is inspired by CroW [17] and Grad-CAM [32] 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. Representation

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 $p \in P$.

3.3. Feature saliency

Inspired by *cross-dimensional weighting and pooling* (CroW) [17] and *class activation mapping* (CAM) [44], 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 [17]. 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.

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.

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

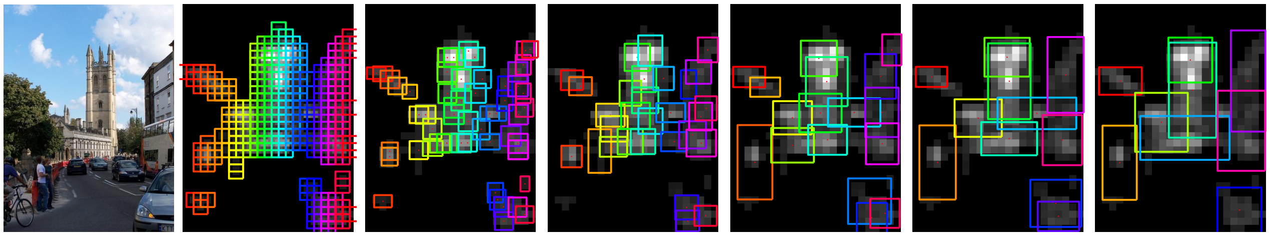

Figure 3. Evolution of regions during EGM iterations on the feature saliency map of an image of *Magdalen tower* from Oxford buildings dataset, shown on the left. Below each image we display the iteration i and the number of regions m.

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 $f, g : \mathbb{R}^d \to \mathbb{R}$, 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

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 [17] or max [3, 39] pooling. We adopt the latter choice to extract descriptor $\mathbf{z} := m_A(R) \in \mathbb{R}^c$, where

$$m_A(R) := \max_{q \in R} A_{q\bullet} \qquad (10)$$

is the maximum of 3d tensor A over R along the spatial dimensions. This has been the basis of fine-tuning in [29, 8].

A particular set of regions, uniformly sampled on a grid at different scales, is referred to as *regional maximum activation of convolutions* (R-MAC) [39]. Global description, referred to as MAC, is a 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.

Finally, we follow [29] in performing supervised whitening of the descriptors by simultaneous diagonalization [22]. In particular, given vector $\mathbf{z} \in \mathbb{R}^c$, we ℓ^2-normalize, center, whiten, PCA-project and renormalize to generate the *region descriptor* $\mathbf{v} := w(\mathbf{z}) \in \mathbb{R}^d$ for region R. Function $w : \mathbb{R}^c \to \mathbb{R}^d$ represents the entire whitening and normalization pipeline.

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 as discussed in 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 as discussed in 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.

We construct a weighted undirected graph having the set of descriptors V as vertices. Following [12], 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 $s(\mathbf{v}, \mathbf{u}) = (\mathbf{v}^\top \mathbf{u})^\beta$, 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* $D := \operatorname{diag}(W\mathbf{1})$ where $\mathbf{1} \in \mathbb{R}^n$ is the all-ones vector, and the *symmetrically normalized adjacency matrix*

$$\mathcal{W} := D^{-1/2} W D^{-1/2}, \qquad (11)$$

with the convention $0/0 = 0$. Following [12, 11], we define the $n \times n$ matrices $L_\alpha := (D - \alpha W)/(1 - \alpha)$ and

$$\mathcal{L}_\alpha := D^{-1/2} L_\alpha D^{-1/2} = (I - \alpha \mathcal{W})/(1 - \alpha), \qquad (12)$$

where $\alpha \in [0, 1)$. Both are positive-definite [12, 11].

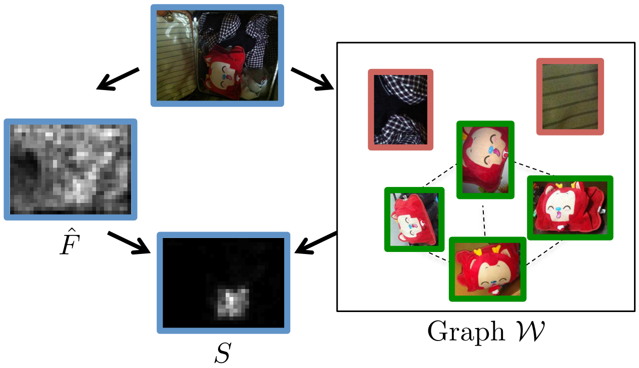

Figure 4. Computing the *object saliency* map S of an image from Instre dataset (top), as defined in (14). For each patch, its neighbors in the graph (right) are found. Common patterns with high centrality in green outline, outliers with low centrality in red. S (bottom) then focuses on patches similar to common patterns and combines with feature saliency $\hat{F}$ (left).

3.7. 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 define this *centrality vector* as the solution $\mathbf{g}^* \in \mathbb{R}^n$ of the linear system

$$\mathcal{L}_\alpha \mathbf{g} = \mathbf{1}. \qquad (13)$$

As in [12], we solve this system by the *conjugate gradients* (CG) [23] method. Any method would be equally appropriate because this is computed just once offline.

The solution $\mathbf{g}^*$ is a *graph centrality* measure [21], and in particular, *Katz centrality* [18]. Centrality is a global measure of significance of vertices in a graph, and PageRank [27] is maybe the most well-known. In fact, Katz centrality was introduced as such a global measure before being adapted by *boundary condition* $\mathbf{y}$ to measure relevance to individual vertices by Hubbell [10]. This work has a long history before being rediscovered *e.g.* by [27, 45], as summarized in the study of *spectral ranking* [41].

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

$$S_p := \hat{F}_p^\Theta \sum_{i \in N_p} s(\mathbf{v}_i, \mathbf{u}_p) f_i^\theta g_i^*. \tag{14}$$

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. Exponents Θ and θ control the relative importance of feature saliency of the current image and neighbors, respectively, 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. Representation

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 max 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. The difference here is that we apply whitening on the aggregated vector and not separately per region. This is the same representation as R-MAC evaluated in [29] and both yield a global image representation in $\mathbb{R}^d$, but here the regions are detected in the saliency map rather than being uniformly distributed.

Pooling based on saliency is in fact the idea explored in CroW [17], but here we follow the nonlinear two-level pooling of R-MAC (max followed by sum) rather than the one-level sum of CroW. This is more powerful and has also been the basis of fine-tuning in [8].

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 both image and database specific.

4.1. Experimental setup

Test sets. We evaluate on Oxford Buildings [28] and the more recently introduced Instre [42] 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 [12] for Instre. We add 100k distractors from Flickr [28] to Oxford5k to perform experiments at larger scale. We refer to it as Oxford105k. Search performance in all datasets is measured with mAP.

Image Representation. We represent each image by global image representation as described in Section 3.9. This reduces image similarity to cosine, which is common practice [39]. Feature extraction is performed with the VGG network [34] that is fine-tuned specifically for image retrieval [29]. Supervised whitening [29] is used for post-processing. The same network is additionally used to compare against two baselines. First, MAC global descriptor, which is obtained by global max pooling and the descriptor that the network is directly optimized for [29]. Second, the baseline approach (*Uniform*), which refers to regional max pooling for regions that are uniformly sampled at 3 scales, as in R-MAC [39]. 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 set $\rho = 1$, $\tau = 0.2$ and scale parameter $\sigma = 1$ before any parameter tuning is performed. We determine OS parameters Θ, θ in (14) by visual inspection of OS and set $\Theta = 2$, $\theta = 3$ throughout our experiments.

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 Oxford5k when using diffusion [12]. The remaining experiments evaluate the proposed representation with the chosen parameters on Instre and Oxford105k as well.

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, 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 $\rho = 5$ for the rest of our experiments.

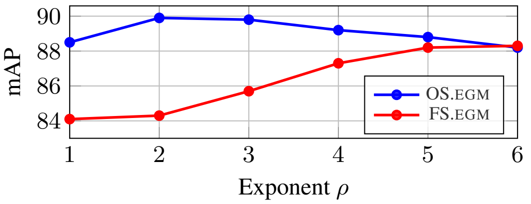

Figure 5. mAP on Oxford5k versus saliency exponent ρ for FS.EGM and OS.EGM.

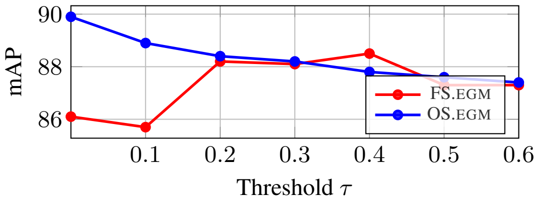

Figure 6. mAP on Oxford5k versus threshold τ for FS.EGM and OS.EGM.

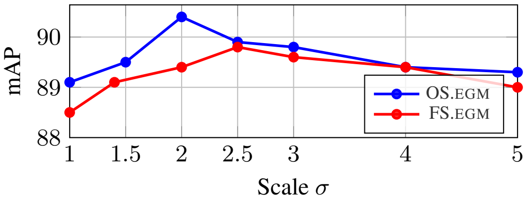

Figure 7. mAP on Oxford5k versus EGM scale parameter σ for FS.EGM and OS.EGM.

Figure 6 shows the effect of threshold τ, which is another selectivity parameter. We set $\tau = 0.4$. Scale σ is used during EGM sampling as explained in Section 3.4. Its impact in performance is shown in Figure 7. Setting $\sigma = 2.5$ results in good performance and regions that are large enough for FS.EGM.

Object saliency detection is then evaluated once the feature saliency parameters are fixed, and EGM detection is applied on the new saliency map. We observe that OS behaves quite differently to FS, because foreground objects are much cleaner. The impact of parameters σ and ρ is shown in Figures 5 and 7 respectively. It is remarkable that a much lower exponent is needed in this case. We choose $\rho = 2$ and $\sigma = 2$. Finally, we fix $\tau = 0$ for OS, as the saliency maps obtained with OS are exactly zero at background regions. The effect is shown in Figure 6.

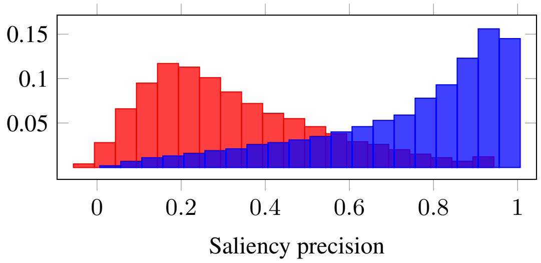

Figure 8. Histogram of saliency precision for FS and OS maps measured on all images of Instre.

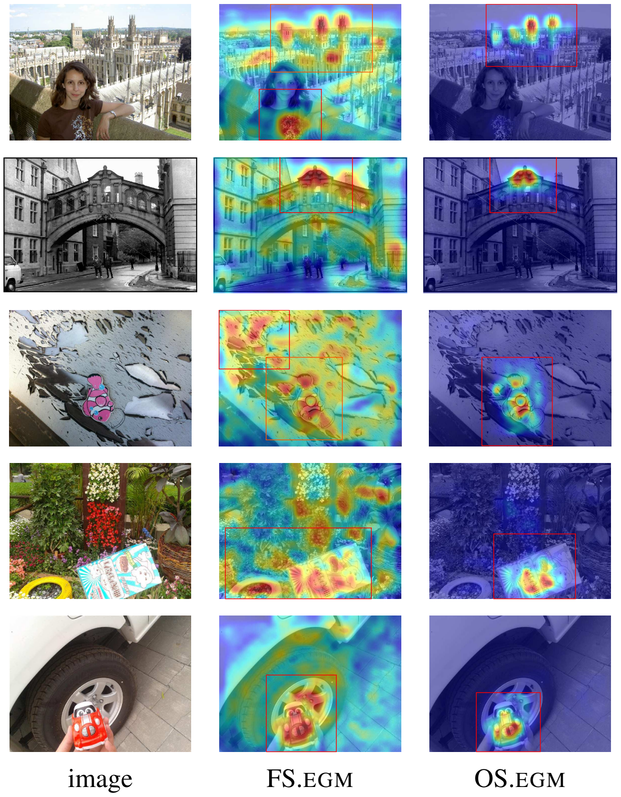

Figure 9. Examples of images from Oxford5k (first 2 rows) and Instre (last 3 rows) datasets, along with smoothed FS and OS maps superimposed on the images and regions detected by EGM, in red.

4.3. Evaluation of saliency maps

We exploit the fact that Instre dataset comes with bounding box annotation for all database images. We use the ground truth information to quantitatively evaluate the saliency maps. We define *precision* as the sum of saliency over ground truth regions, normalized by the sum over the entire image, and we measure it for FS and OS as shown in Figure 8. High precision means that a saliency map is well aligned to the ground truth bounding boxes. Given that these bounding boxes are not used anywhere, the improvement that OS offers is impressive. Visual examples for saliency maps and detections for FS.EGM and OS.EGM are shown in Figure 9. In all cases, OS is cleaner and focuses on objects that FS cannot discriminate.

4.4. Comparison to other methods

We compare our methods to the standard practice of uniform region sampling (Uniform) as in R-MAC and global max pooling (MAC). We additionally propose a variant of

Method	QE	Instre	Oxford	Oxford105k
MAC	-	48.5	79.7	73.9
Uniform [39]	-	47.7	77.7	70.1
FS.EGM *	-	48.4	77.5	70.2
OS.EGM *	-	50.1	79.6	71.8
OS.EGM-△*	-	53.7	79.8	71.4
MAC	✓	71.8	87.4	86.0
Uniform [39]	✓	70.3	85.7	82.7
FS.EGM *	✓	71.2	89.8	87.9
OS.EGM *	✓	72.7	**90.4**	**88.0**
OS.EGM-△*	✓	**75.4**	90.1	84.3

Table 1. mAP comparison of our methods marked with * against baselines on all tested datasets. QE refers to query expansion by diffusion [12].

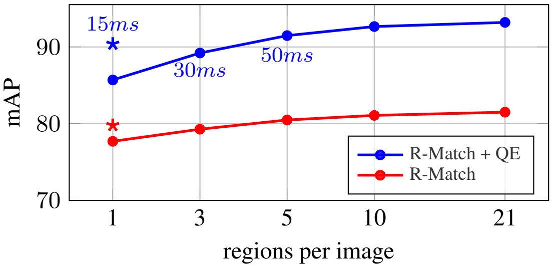

Figure 10. mAP comparison of our global OS.EGM ($\star$) to R-Match with uniformly sampled regional descriptors, with and without diffusion on Oxford5k. Text labels refer to query time.

OS.EGM, where further uniform region sampling at 2 scales is performed within each detected region. We refer to this as OS.EGM-△. All methods are tested with k-NN search and global diffusion [12], which is a method for query expansion or manifold search and is known to significantly improve performance. Results are given in Table 1.

FS.EGM improves performance compared to uniform sampling by focusing on salient objects. However, salient objects are not necessarily relevant for the particular dataset. This is what OS.EGM captures and boosts the search performance, especially on Instre. On all datasets, MAC is better than uniform sampling (R-MAC). This is known to be due to the fact that the network used [29] is directly fine-tuned to optimize MAC. However, when using diffusion, we outperform it on all datasets. This can be explained by the fact that diffusion boosts any items that are similar to the top-ranking ones according to the original similarity [12], so it is essential that these items are reliable. A global descriptor is affected by clutter in general. By contrast, our representation is global yet clutter-free. Our improvements are larger on Instre, which is more challenging due to small objects and severe background clutter. This is exactly where our detection is essential. Most Instre images are also quite different than the building images which the network is fine-tuned on. This is probably why our representations outperform MAC even without diffusion on this dataset.

There are several other previous approaches that deal with region detection or saliency masks, which are not directly comparable, so they are not included in Table 1. Nevertheless, we outperform their reported results. Salvador *et al.* [31] use the off-the-shelf VGG and fine-tune RPN in the test set. When not using query expansion, they obtain 71.0 in Oxford5k. Similarly, Jimenez *et al.* [15] learn class weights and apply them on the activation maps of off-the-shelf VGG and achieve 73.6 in Oxford5k. Song *et al.* [36] train on different datasets, and achieve 78.3 in Oxford5k. The results obtained by learning a saliency mask are not comparable since spatial verification with local features is always applied in the end [24]. Finally, Zheng *et al.* [43] 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, no work other than [12] evaluates on Instre which is rather challenging due to small objects.

Region cross-matching methods [30] 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 [12] in Figure 10. Different numbers of regions are obtained by GMM reduction, exactly as in [12].

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.

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. 3, 4

[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. 1, 4

[5] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. In *ECCV*, 2014. 1

[6] O. Chum and J. Matas. Unsupervised discovery of cooccurrence in sparse high dimensional data. In *CVPR*, June 2010. 2

[7] 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

[8] A. Gordo, J. Almazan, J. Revaud, and D. Larlus. Deep image retrieval: Learning global representations for image search. *ECCV*, 2016. 1, 5, 6

[9] 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

[10] C. H. Hubbell. An input-output approach to clique identification. *Sociometry*, 1965. 5

[11] A. Iscen, Y. Avrithis, G. Tolias, T. Furon, and O. Chum. Fast spectral ranking for similarity search. *arXiv*, 2017. 1, 5

[12] 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. 1, 2, 5, 6, 8

[13] H. Jégou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. *IJCV*, 87(3):316–336, February 2010. 2

[14] D.-j. Jeong, S. Choo, W. Seo, and N. I. Cho. Regional deep feature aggregation for image retrieval. In *ICASSP*, 2017. 2

[15] A. Jimenez, J. M. Alvarez, and X. Giro-i Nieto. Class-weighted convolutional features for visual instance search. *BMVC*, 2017. 2, 8

[16] H. Jin Kim, E. Dunn, and J.-M. Frahm. Learned contextual feature reweighting for image geo-localization. In *CVPR*, 2017. 2

[17] Y. Kalantidis, C. Mellina, and S. Osindero. Cross-dimensional weighting for aggregated deep convolutional features. In *arXiv*, 2015. 1, 2, 3, 5, 6

[18] L. Katz. A new status index derived from sociometric analysis. *Psychometrika*, 18(1):39–43, 1953. 5

[19] J. Knopp, J. Sivic, and T. Pajdla. Avoiding confusing features in place recognition. In *ECCV*, 2010. 2

[20] Z. Laskar and J. Kannala. Context aware query image representation for particular object retrieval. In *Scandinavian Conference on Image Analysis*, 2017. 2

[21] N. MEJ. *Networks: an introduction*. Oxford University Press, Oxford, 2010. 5

[22] K. Mikolajczyk and J. Matas. Improving descriptors for fast tree matching by optimal linear projection. In *CVPR*, 2007. 5

[23] J. Nocedal and S. Wright. *Numerical optimization*. Springer, 2006. 5

[24] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han. Large-scale image retrieval with attentive deep local features. In *arXiv*, 2016. 1, 2, 8

[25] 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

[26] 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

[27] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank citation ranking: bringing order to the web. 1999. 5

[28] 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, 6

[29] F. Radenović, G. Tolias, and O. Chum. CNN image retrieval learns from bow: Unsupervised fine-tuning with hard examples. *ECCV*, 2016. 2, 5, 6, 8

[30] 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, 8

[31] A. Salvador, X. Giró-i Nieto, F. Marqués, and S. Satoh. Faster r-cnn features for instance search. In *CVPRW*, 2016. 1, 2, 8

[32] 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

[33] M. Shi, Y. Avrithis, and H. Jegou. Early burst detection for memory-efficient image retrieval. In *CVPR*, 2015. 2

[34] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2014. 6

[35] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *ICCV*, 2003. 1, 2

[36] 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. 1, 2, 8

[37] G. Tolias, Y. Avrithis, and H. Jégou. Image search with selective match kernels: aggregation across single and multiple images. *IJCV*, 2016. 2

[38] G. Tolias, Y. Kalantidis, and Y. Avrithis. Symcity: Feature selection by symmetry for large scale image retrieval. In *ACM Multimedia*, 2012. 2

[39] G. Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of cnn activations. *ICLR*, 2016. 1, 2, 5, 6, 8

[40] 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

[41] S. Vigna. Spectral ranking. *arXiv preprint arXiv:0912.0238*, 2009. 5

[42] 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. 6

[43] L. Zheng, S. Wang, J. Wang, and Q. Tian. Accurate image search with multi-scale contextual evidences. *IJCV*, 120(1):1–13, 2016. 8

[44] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *cvpr*, June 2016. 3

[45] D. Zhou, J. Weston, A. Gretton, O. Bousquet, and B. Schölkopf. Ranking on data manifolds. In *NIPS*, 2003. 5