This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2846566, IEEE Transactions on Pattern Analysis and Machine Intelligence

Fine-tuning CNN Image Retrieval with No Human Annotation

Filip Radenović Giorgos Tolias Ondřej Chum

Abstract—Image descriptors based on activations of Convolutional Neural Networks (CNNs) have become dominant in image retrieval due to their discriminative power, compactness of representation, and search efficiency. Training of CNNs, either from scratch or fine-tuning, requires a large amount of annotated data, where a high quality of annotation is often crucial. In this work, we propose to fine-tune CNNs for image retrieval on a large collection of unordered images in a fully automated manner. Reconstructed 3D models obtained by the state-of-the-art retrieval and structure-from-motion methods guide the selection of the training data. We show that both hard-positive and hard-negative examples, selected by exploiting the geometry and the camera positions available from the 3D models, enhance the performance of particular-object retrieval. CNN descriptor whitening discriminatively learned from the same training data outperforms commonly used PCA whitening. We propose a novel trainable Generalized-Mean (GeM) pooling layer that generalizes max and average pooling and show that it boosts retrieval performance. Applying the proposed method to the VGG network achieves state-of-the-art performance on the standard benchmarks: Oxford Buildings, Paris, and Holidays datasets.

1 Introduction

In instance image retrieval an image of a particular object, depicted in a query, is sought in a large, unordered collection of images. Convolutional neural networks (CNNs) have recently provided an attractive solution to this problem. In addition to leaving a small memory footprint, the CNN-based approaches achieve high accuracy. Neural networks have attracted a lot of attention after the success of Krizhevsky *et al.* [1] in the image-classification task. Their success is mainly due to the use of very large annotated datasets, *e.g.* ImageNet [2]. The acquisition of the training data is a costly process of manual annotation, often prone to errors. Networks trained for image classification have shown strong adaptation abilities [3]. Specifically, using activations of CNNs, which were trained for the task of classification, as off-the-shelf image descriptors [4], [5] and adapting them for a number of tasks [6], [7], [8] have shown acceptable results. In particular, for image retrieval, a number of approaches directly use the network activations as image features and successfully perform image search [8], [9], [10], [11], [12].

Fine-tuning of the network, *i.e.* initialization by a pre-trained classification network and then training for a different task, is an alternative to a direct application of a pre-trained network. Fine-tuning significantly improves the adaptation ability [13], [14]; however, further annotation of training data is required. The first fine-tuning approach for image retrieval is proposed by Babenko *et al.* [15], in which a significant amount of manual effort was required to collect images and label them as specific building classes. They improved retrieval accuracy; however, their formulation is much closer to classification than to the desired properties of instance retrieval. In another approach, Arandjelovic *et al.* [16] perform fine-tuning guided by geo-tagged image databases and, similar to our work, they directly optimize the similarity measure to be used in the final task by selecting *matching* and *non-matching* pairs to perform the training.

In contrast to previous methods of training-data acquisition for image search, we dispense with the need for manually annotated data or any assumptions on the training dataset. We achieve this by exploiting the geometry and the camera positions from 3D models reconstructed automatically by a structure-from-motion (SfM) pipeline. The state-of-the-art retrieval-SfM pipeline [17] takes an unordered image collection as input and attempts to build all possible 3D models. To make the process efficient, fast image clustering is employed. A number of image clustering methods based on local features have been introduced [18], [19], [20]. Due to spatial verification, the *clusters* discovered by these methods are reliable. In fact, the methods provide not only clusters, but also a matching graph or sub-graph on the cluster images. The SfM filters out virtually all mismatched images and provides image-to-model matches and camera positions for all matched images in the cluster. The whole process, from unordered collection of images to detailed 3D reconstructions, is fully automatic. Finally, the 3D models guide the selection of matching and non-matching pairs. We propose to exploit the training data acquired by the same procedure in the descriptor post-processing stage to learn a discriminative whitening.

An additional contribution of this work lies in the introduction of a novel pooling layer after the convolutional layers. Previously, a number of approaches have been used. These range from fully-connected layers [8], [15], to different global-pooling layers, *e.g.* max pooling [9], average pooling [10], hybrid pooling [21], weighted average pooling [11], and regional pooling [12]. We propose a pooling layer based on a generalized-mean that has learnable parameters, either one global or one per output dimension. Both max and average pooling are its special cases. Our experiments show that it offers a significant performance boost over standard non-trainable pooling layers. Our architecture is shown in Figure 1.

Affiliation: *Visual Recognition Group, Faculty of Electrical Engineering, Czech Technical University in Prague*

E-mail: *{filip.radenovic,giorgos.tolias,chum}@cmp.felk.cvut.cz*

Data, networks, and code: *cmp.felk.cvut.cz/cnnimageretrieval*

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2846566, IEEE Transactions on Pattern Analysis and Machine Intelligence

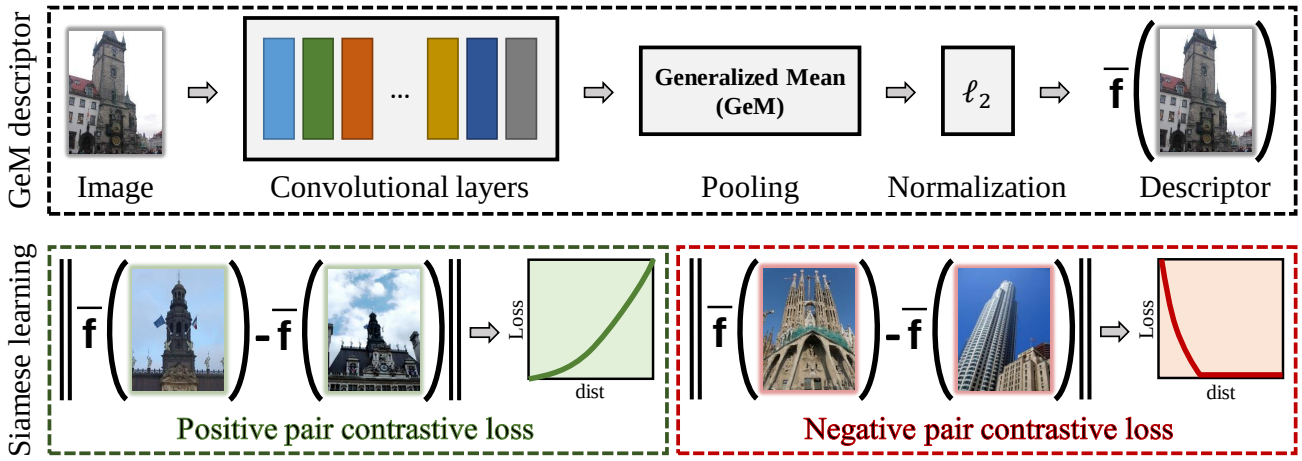


Fig. 1. The architecture of our network with the contrastive loss used at training time. A single vector $\bar{\mathbf{f}}$ is extracted to represent an image.

To summarize, we address the unsupervised fine-tuning of CNNs for image retrieval. In particular, we make the following contributions: (1) We exploit SfM information and enforce, not only hard non-matching (*negative*), but also hard-matching (*positive*) examples for CNN training. This is shown to enhance the derived image representation. We show that compared to previous supervised approaches, the variability in the training data from 3D reconstructions delivers superior performance in the image-retrieval task. (2) We show that the whitening traditionally performed on short representations [22] is, in some cases, unstable. We propose to learn the whitening through the same training data. Its effect is complementary to fine-tuning and it further boosts performance. Also, performing whitening as a post-processing step is better and much faster to train compared to learning it end-to-end. (3) We propose a trainable pooling layer that generalizes existing popular pooling schemes for CNNs. It significantly improves the retrieval performance while preserving the same descriptor dimensionality. (4) In addition, we propose a novel α-weighted query expansion that is more robust compared to the standard average query expansion technique widely used for compact image representations. (5) Finally, we set a new state-of-the-art result for Oxford Buildings, Paris, and Holidays datasets by re-training the commonly used CNN architectures, such as AlexNet [1], VGG [23], and ResNet [24].

This manuscript is an extension of our previous work [25]. We additionally propose a novel pooling layer (Section 3.2), a novel multi-scale image representation (Section 5.2), and a novel query expansion method (Section 5.3). Each one of the newly proposed methods boosts image-retrieval performance, and is accompanied by experiments that give useful insights. In addition, we provide an extended related work discussion including the different pooling procedures used in prior CNN work and descriptor whitening. Finally, we compare our approach to the concurrent related work of Gordo *et al.* [26], [27]. They significantly improve the retrieval performance through end-to-end learning which incorporates building-specific region proposals. In contrast to their work, we focus on the importance of hard-training data examples, and employ a much simpler but equally powerful pooling layer.

The rest of the paper is organized as follows. Related work is discussed in Section 2, our network architecture, learning procedure, and search process is presented in Section 3, and our proposed automatic acquisition of the training data is described in Section 4. Finally, in Section 5 we perform an extensive quantitative and qualitative evaluation of all proposed novelties with different CNN architectures and compare to the state of the art.

2 Related work

The CNN-based representation is an appealing solution for image retrieval and in particular for compact image representations. Previous compact descriptors are typically constructed by an aggregation of local features, where representatives are Fisher vectors [28], VLAD [29] and alternatives [30], [31], [32]. Impressively, in this work we show that CNNs dominate the image search task by outperforming state-of-the-art methods that have reached a higher level of maturity by incorporating large visual codebooks [33], [34], spatial verification [33], [35] and query expansion [36], [37], [38].

In this work, instance retrieval is cast as a metric learning problem, *i.e.*, an image embedding is learned so that the Euclidean distance captures the similarity well. Typical architectures for metric learning, such as the two-branch siamese [39], [40], [41] or triplet networks [42], [43], [44] employ *matching* and *non-matching* pairs to perform the training and better suit to this task. Here, the problem of annotations is even more pronounced, *i.e.*, for classification one needs only object category label, while for particular objects the labels have to be per image pair. Two images from the same object category could potentially be completely different, *e.g.*, different viewpoints of the building or different buildings. We solve this problem in a fully automated manner, without any human intervention, by starting from a large unordered image collection.

In the following text we discuss the related work for our main contributions, *i.e.*, the training data collection, the pooling approach to construct a global image descriptor, and the descriptor whitening.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2846566, IEEE Transactions on Pattern Analysis and Machine Intelligence

2.1 Training data

A variety of previous methods apply CNN activations on the task of image retrieval [8], [9], [10], [11], [12], [45]. The achieved accuracy on retrieval is evidence for the generalization properties of CNNs. The employed networks are trained for image classification using ImageNet dataset [2] by minimizing classification error. Babenko *et al.* [15] go one step further and re-train such networks with a dataset that is closer to the target task. They perform training with object classes that correspond to particular landmarks/buildings. Performance is improved on standard retrieval benchmarks. Despite the achievement, still, the final metric and the utilized layers are different to the ones actually optimized during learning.

Constructing such training datasets requires manual effort. In recent work, geo-tagged datasets with timestamps offer the ground for weakly-supervised fine-tuning of a triplet network [16]. Two images taken far from each other can be easily considered as non-matching, while matching examples are picked by the most similar nearby images. In the latter case, similarity is defined by the current representation of the CNN. This is the first approach that performs end-to-end fine-tuning for image retrieval and, in particular, for the geo-localization task. The used training images are now more relevant to the final task. We differentiate by discovering matching and non-matching image pairs in an unsupervised way. Moreover, we derive matching examples based on 3D reconstructions which allows for harder examples.

Even though hard-negative mining is a standard process [6], [16], this is not the case with hard-positive examples. Mining of hard positive examples have been exploited in the work Simo-Serra *et al.* [46], where patch-level examples were extracted though the guidance from a 3D reconstruction. Hard-positive pairs have to be sampled carefully. Extremely hard positive examples (such as minimal overlap between images or extreme scale change) do not allow to generalize and lead to over-fitting.

A concurrent work to ours also uses local features and geometric verification to select positive examples [26]. In contrast to our fully unsupervised method, they start from a landmarks dataset, which had to be manually cleaned, and the landmark labels of the dataset, rather than the geometry, were used to avoid exhaustive evaluation. The same training dataset is used by Noh *et al.* [47] to learn global image descriptors using a saliency mask. However, during test time the CNN activations are seen as local descriptors, indexed independently, and used for a subsequent spatial-verification stage. Such approach boosts accuracy compared to global descriptors, but at the cost of much higher complexity.

2.2 Pooling method

Early approaches to applying CNNs for image retrieval included methods that set the fully-connected layer activations to be the global image descriptors [8], [15]. The work by Razavian *et al.* [9] moves the focus to the activations of convolutional layers followed by a global-pooling operation. A compact image representation is constructed in this fashion with dimensionality equivalent to the number of feature maps of the corresponding convolutional layer. In particular, they propose to use max pooling, which is later approximated with integral max pooling [12].

Sum pooling was initially proposed by Babenko and Lempitsky [10], which was shown to perform well especially due to the subsequent descriptor whitening. One step further is the weighted sum pooling of Kalantidis *et al.* [11], which can also be seen as a way to perform transfer learning. Popular encodings such as BoW, VLAD, and Fisher vectors are adapted in the context of CNN activations in the work of Mohedano *et al.* [48], Arandjelovic *et al.* [16], and Ong *et al.* [49], respectively. Sum pooling is employed once an appropriate embedding is performed beforehand.

A hybrid scheme is the R-MAC method [12], which performs max pooling over regions and finally sum pooling of the regional descriptors. Mixed pooling is proposed globally for retrieval [21] and the standard local pooling is used for object recognition [50]. It is a linear combination of max and sum pooling. A generalization scheme similar to ours is proposed in the work of Cohen *et al.* [51] but in a different context. They replace the standard local max pooling with the generalized one. Finally, generalized mean is used by Morère *et al.* [52] to pool the similarity values under multiple transformations.

2.3 Descriptor whitening

Whitening the data representation is known to be very essential for image retrieval since the work of Jégou and Chum [22]. Their interpretation lies on jointly down-weighting co-occurrences and, thus, handling the problem of over-counting. The benefit of whitening is further emphasized in the case of CNN-based descriptors [5], [10], [12]. Whitening is commonly learned from a generative model in an unsupervised way by PCA on an independent dataset.

We propose to learn the whitening transform in a discriminative manner, using the same acquisition procedure of the training data from 3D models. A similar approach has been used to whiten local-feature descriptors by Mikolajczyk and Matas [53].

In constrast, Gordo *et al.* [26] learn the whitening in the CNN in an end-to-end manner. In our experiments we found this choice to be at most as good as the descriptor post-processing and less efficient due to slower convergence of the learning.

3 Architecture, Learning, Search

In this section we describe the network architecture and present the proposed generalized-pooling layer. Then we explain the process of fine-tuning using the contrastive loss and a two-branch network. We describe how, after fine-tuning, we use the same training data to learn projections that appear to be an effective post-processing step. Finally, we describe the image representation, search process, and a novel query expansion scheme. Our proposed architecture is depicted in Figure 1.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2846566, IEEE Transactions on Pattern Analysis and Machine Intelligence

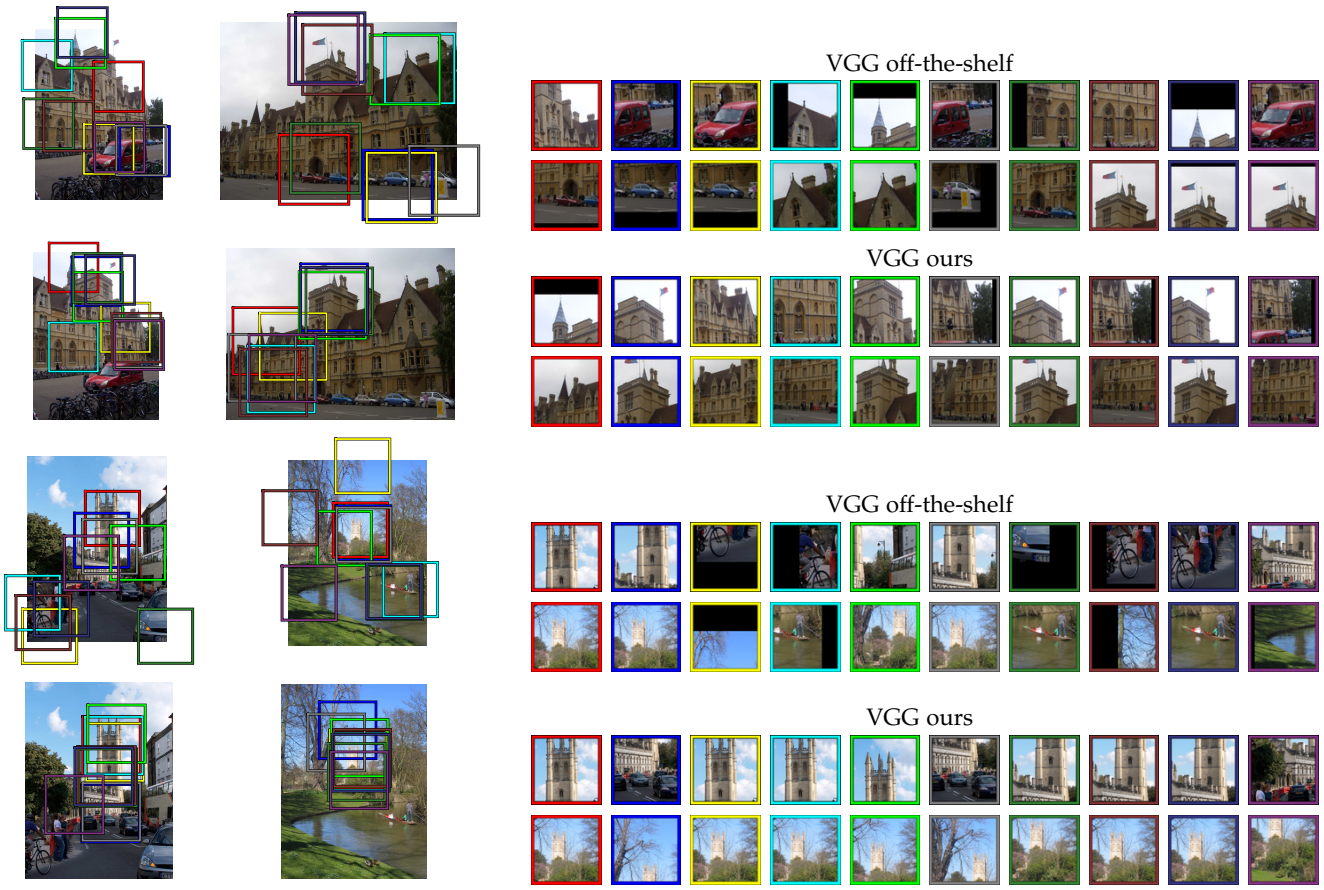


Fig. 2. Visualization of image regions that correspond to MAC descriptor dimensions that have the highest contribution, *i.e.* large product of descriptor elements, to the pairwise image similarity. The example uses VGG before (top) and after (bottom) fine-tuning. Same color corresponds to the same descriptor component (feature map) per image pair. The patch size is equal to the receptive field of the last local pooling layer.

3.1 Fully convolutional network

Our methodology applies to any fully convolutional CNN [54]. In practice, popular CNNs for generic object recognition are adopted, such as AlexNet [1], VGG [23], or ResNet [24], while their fully-connected layers are discarded. This provides a good initialization to perform the fine-tuning.

Now, given an input image, the output is a 3D tensor $\mathcal{X}$ of $W \times H \times K$ dimensions, where K is the number of feature maps in the last layer. Let $\mathcal{X}_k$ be the set of $W \times H$ activations for feature map $k \in \{1 \dots K\}$. The network output consists of K such activation sets or 2D feature maps. We additionally assume that the very last layer is a Rectified Linear Unit (ReLU) such that $\mathcal{X}$ is non-negative.

3.2 Generalized-mean pooling and image descriptor

We now add a pooling layer that takes $\mathcal{X}$ as an input and produces a vector $\mathbf{f}$ as an output of the pooling process. This vector in the case of the conventional global max pooling (MAC vector [9], [12]) is given by

$$\mathbf{f}^{(m)} = [\mathrm{f}_1^{(m)} \dots \mathrm{f}_k^{(m)} \dots \mathrm{f}_K^{(m)}]^\top, \qquad \mathrm{f}_k^{(m)} = \max_{x \in \mathcal{X}_k} x, \tag{1}$$

while for average pooling (SPoC vector [10]) by

$$\mathbf{f}^{(a)} = [\mathrm{f}_1^{(a)} \dots \mathrm{f}_k^{(a)} \dots \mathrm{f}_K^{(a)}]^\top, \qquad \mathrm{f}_k^{(a)} = \frac{1}{|\mathcal{X}_k|} \sum_{x \in \mathcal{X}_k} x. \tag{2}$$

Instead, we exploit the generalized mean [55] and propose to use generalized-mean (GeM) pooling whose result is given by

$$\mathbf{f}^{(g)} = [\mathrm{f}_1^{(g)} \dots \mathrm{f}_k^{(g)} \dots \mathrm{f}_K^{(g)}]^\top, \quad \mathrm{f}_k^{(g)} = \left(\frac{1}{|\mathcal{X}_k|} \sum_{x \in \mathcal{X}_k} x^{p_k} \right)^{\frac{1}{p_k}}. \tag{3}$$

Pooling methods (1) and (2) are special cases of GeM pooling given in (3), *i.e.*, max pooling when $p_k \to \infty$ and average pooling for $p_k = 1$. The feature vector finally consists of a single value per feature map, *i.e.* the generalized-mean activation, and its dimensionality is equal to K. For many popular networks this is equal to 256, 512 or 2048, making it a compact image representation.

The pooling parameter p_k can be manually set or learned since this operation is differentiable and can be part of the back-propagation. The corresponding derivatives (while skipping the superscript (g) for brevity) are given by

$$\frac{\partial \mathrm{f}_k}{\partial x_i} = \frac{1}{|\mathcal{X}_k|} \mathrm{f}_k^{1-p_k} {x_i}^{p_k - 1}, \tag{4}$$

$$\frac{\partial \mathrm{f}_k}{\partial p_k} = \frac{\mathrm{f}_k}{p_k^2} \left(\log \frac{|\mathcal{X}_k|}{\sum_{x \in \mathcal{X}_k} x^{p_k}} + p_k \frac{\sum_{x \in \mathcal{X}_k} x^{p_k} \log x}{\sum_{x \in \mathcal{X}_k} x^{p_k}} \right). \tag{5}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2846566, IEEE Transactions on Pattern Analysis and Machine Intelligence

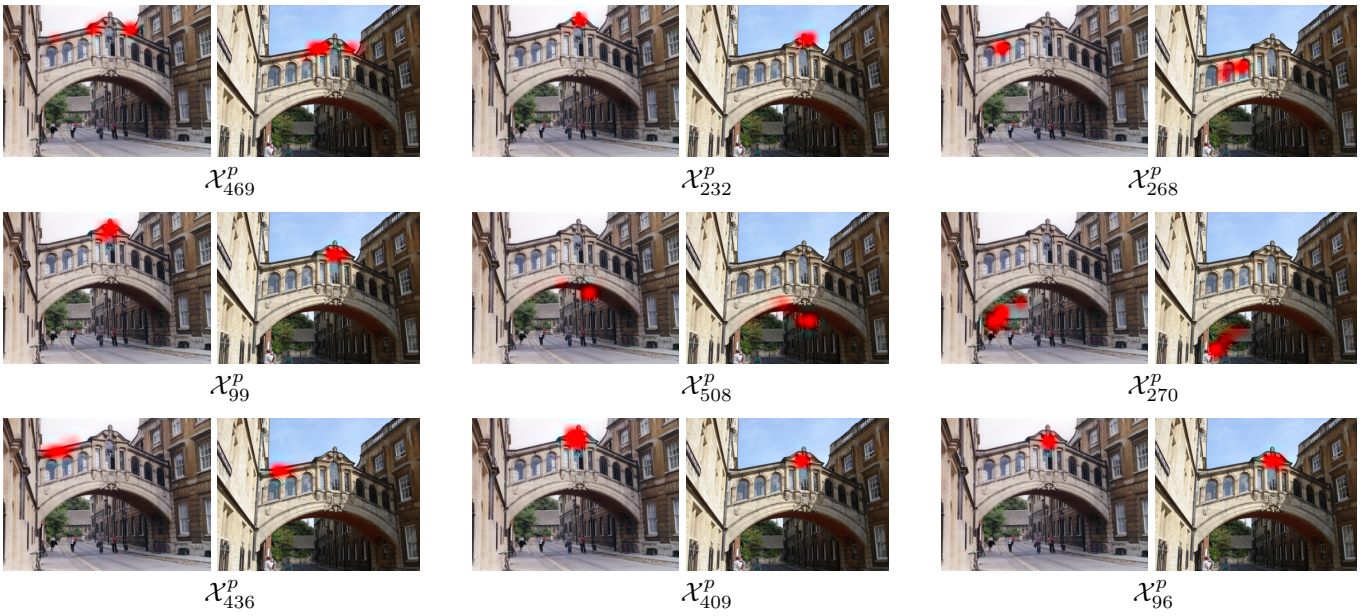


Fig. 3. Visualization of $\mathcal{X}^p_k$ projected on the original image for a pair of query-database image. The 9 feature maps shown are the ones that score highly, *i.e.* large product of GeM descriptor components, for the database image (right) but low for the top-ranked non-matching images. The example uses fine-tuned VGG with GeM and single p for all feature maps, which converged to 2.92.

There is a different pooling parameter per feature map in (3), but it is also possible to use a shared one. In this case $p_k = p, \forall k \in [1, K]$ and we simply denote it by p and not p_k. We examine such options in the experimental section and compare to hand-tuned and fixed parameter values.

Max pooling, in the case of MAC, retains one activation per 2D feature map. In this way, each descriptor component corresponds to an image patch equal to the receptive field. Then, pairwise image similarity is evaluated via descriptor inner product. Therefore, MAC similarity implicitly forms patch correspondences. The strength of each correspondence is given by the product of the associated descriptor components. In Figure 2 we show the image patches in correspondence that contribute most to the similarity. Such implicit correspondences are improved after fine-tuning. Moreover, the CNN fires less on ImageNet classes, *e.g.* cars and bicycles.

In Figure 4 we show how the spatial distribution of the activations is affected by the generalized mean. The larger the p the more localized the feature map responses are. Finally, in Figure 3 we present an example of a query and a database image matched with the fine-tuned VGG with GeM pooling layer (GeM layer in short). We show the feature maps that contribute the most into making this database image being distinguished from non-matching ones that have large similarity, too.

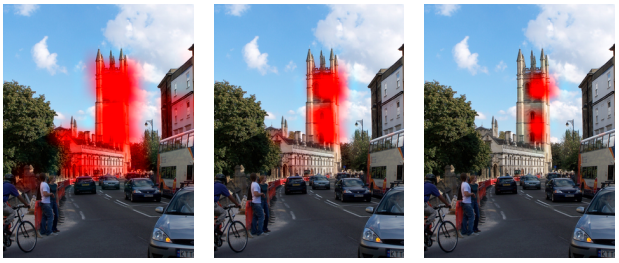


Fig. 4. Visualization of $\mathcal{X}^p_k$ projected on the original image for three different values of p. Case $p = 1$ corresponds to SPoC, and larger p corresponds to GeM before the summation of (3). Examples shown use the off-the-shelf VGG.

The last network layer comprises an ℓ_2-normalization layer. Vector $\mathbf{f}$ is ℓ_2-normalized so that similarity between two images is finally evaluated with inner product. In the rest of the paper, GeM vector corresponds to the ℓ_2-normalized vector $\bar{\mathbf{f}}$ and constitutes the image descriptor.

3.3 Siamese learning and loss function

We adopt a siamese architecture and train a two-branch network. Each branch is a clone of the other, meaning that they share the same parameters. The training input consists of image pairs (i, j) and labels $Y(i, j) \in \{0, 1\}$ declaring whether a pair is non-matching (label 0) or matching (label 1). We employ the contrastive loss [39] that acts on matching and non-matching pairs and is defined as

$$\mathcal{L}(i,j) = \begin{cases} \frac{1}{2}||\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)||^2, & \text{if } Y(i,j) = 1 \\ \frac{1}{2}\left(\max\{0, \tau - ||\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)||\}\right)^2, & \text{if } Y(i,j) = 0 \end{cases} \quad (6)$$

where $\bar{\mathbf{f}}(i)$ is the ℓ_2-normalized GeM vector of image i, and τ is a margin parameter defining when non-matching pairs have large enough distance in order to be ignored by the loss. We train the network using a large number of training pairs created automatically (see Section 4). In contrast to other methods [16], [42], [43], [44], we find that the contrastive loss generalizes better and converges at higher performance than the triplet loss.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2846566, IEEE Transactions on Pattern Analysis and Machine Intelligence

3.4 Whitening and dimensionality reduction

In this section, the post-processing of fine-tuned GeM vectors is considered. Previous methods [10], [12] use PCA of an independent set for whitening and dimensionality reduction, *i.e.* the covariance matrix of all descriptors is analyzed. We propose to leverage the labeled data provided by the 3D models and use linear discriminant projections originally proposed by Mikolajczyk and Matas [53]. The projection is decomposed into two parts: whitening and rotation. The whitening part is the inverse of the square-root of the intraclass (matching pairs) covariance matrix $C_S^{-\frac{1}{2}}$, where

$$C_S = \sum_{Y(i,j)=1} \left(\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)\right)\left(\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)\right)^\top. \quad (7)$$

The rotation part is the PCA of the interclass (non-matching pairs) covariance matrix in the whitened space $\mathrm{eig}(C_S^{-\frac{1}{2}} C_D C_S^{-\frac{1}{2}})$, where

$$C_D = \sum_{Y(i,j)=0} \left(\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)\right)\left(\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)\right)^\top. \quad (8)$$

The projection $P = C_S^{-\frac{1}{2}} \mathrm{eig}(C_S^{-\frac{1}{2}} C_D C_S^{-\frac{1}{2}})$ is then applied as $P^\top(\bar{\mathbf{f}}(i) - \mu)$, where μ is the mean GeM vector to perform centering. To reduce the descriptor dimensionality to D dimensions, only eigenvectors corresponding to D largest eigenvalues are used. Projected vectors are subsequently ℓ_2-normalized.

Our approach uses all available training pairs efficiently in the optimization of the whitening. It is not optimized in an end-to-end manner and it is performed without using batches of training data. We first optimize the GeM descriptor and then optimize the whitening.

The described approach acts as a post-processing step, equivalently, once the fine-tuning of the CNN is finished. We additionally compare with the end-to-end learning of whitening. Whitening consists of vector shifting and projection which is modeled in a straightforward manner by a fully connected layer[1]. The results favor our approach and are discussed in the experimental section.

3.5 Image representation and search

Once the training is finished, an image is fed to the network shown in Figure 1. The extracted GeM descriptor is whitened and re-normalized. This constitutes the global descriptor for an image at a single scale. Scale invariance is learned to some extent by the training samples; however, additional invariance is added by multi-scale processing during test time without any additional learning. We follow a standard approach [27] and feed the image to the network at multiple scales. The resulting descriptors are finally pooled and re-normalized. This vector constitutes a multi-scale global image representation. We adopt GeM pooling for this state too, which is shown, in our experiments, consistently superior to the standard average pooling.

Image retrieval is simply performed by exhaustive Euclidean search over database descriptors *w.r.t.* the query descriptor. This is equivalent to the inner product evaluation of ℓ_2 normalized vectors, *i.e.* vector-to-matrix multiplication, and sorting. CNN-based descriptors are shown to be highly compatible with approximate-nearest neighbor search methods, in fact, they are very compressible [27]. In order to directly evaluate the effectiveness of the learned representation, we do not consider this alternative in this work. In practice, each descriptor requires 4 bytes per dimension to be stored.

It has recently become a standard policy to combine CNN global image descriptors with simple average query expansion (AQE) [10], [11], [12], [27]. An initial query is issued by Euclidean search and AQE acts on the top-ranked nQE images by average pooling of their descriptors. Herein, we argue that tuning nQE to work well across different datasets is not easy. AQE corresponds to a weighted average where nQE descriptors have unit weight and all the rest zero. We generalize this scheme and we propose performing weighted averaging, where the weight of the i-th ranked image is given by $(\bar{\mathbf{f}}(q)^\top \bar{\mathbf{f}}(i))^\alpha$. The similarity of each retrieved image matters. We show in our experiments that AQE is difficult to tune for datasets of different statistics, while this is not the case with the proposed approach. We refer to this approach as α-weighted query expansion (αQE). The proposed αQE reduces to AQE for $\alpha = 0$.

4 Training Dataset

In this section we briefly summarize the tightly-coupled Bag-of-Words (BoW) image-retrieval and Structure-from-Motion (SfM) 3D reconstruction system [17], [56] that is employed to automatically select our training data. Then, we describe how we use the 3D information to select harder matching pairs and hard non-matching pairs with larger variability.

4.1 BoW and 3D reconstruction

The retrieval engine used in the work of Schonberger *et al.* [17] builds upon BoW with fast spatial verification [33]. It uses Hessian affine local features [57], RootSIFT descriptors [58], and a fine vocabulary of 16M visual words [59]. Then, query images are chosen via min-hash and spatial verification, as in [18]. Image retrieval based on BoW is used to collect images of the objects/landmarks. These images serve as the initial matching graph for the succeeding SfM reconstruction, which is performed using the state-of-the-art SfM pipeline [60], [61], [62]. Different mining techniques, *e.g.* zoom in, zoom out [63], [64], sideways crawl [17], help to build a larger and more complete model.

In this work, we exploit the outcome of such a system. Given a large unannotated image collection, images are clustered and a 3D model is constructed per cluster. We use the terms *3D model*, *model* and *cluster* interchangeably. For each image, the estimated camera position is known, as well as the local features registered on the 3D model. We drop redundant (overlapping) 3D models, that might have been constructed from different seeds. Models reconstructing the same landmark but from different and disjoint viewpoints are considered as non-overlapping.

1. The bias is equal to the projected shifting vector.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2846566, IEEE Transactions on Pattern Analysis and Machine Intelligence

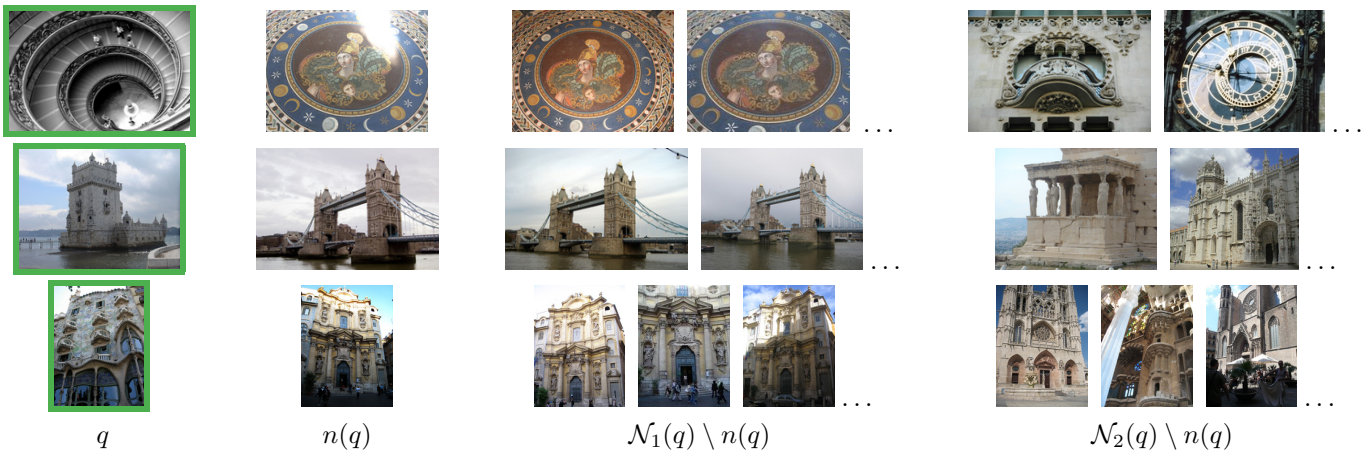


Fig. 5. Examples of training query image q (one per row shown in green border), and their corresponding negatives chosen by different strategies. We show the hardest non-matching image $n(q)$, and the additional non-matching images selected as negative examples by $\mathcal{N}_1(q)$ and our method $\mathcal{N}_2(q)$. The former chooses k-nearest neighbors among all non-matching images, while the latter chooses k-nearest neighbors but with at most one image per 3D model.

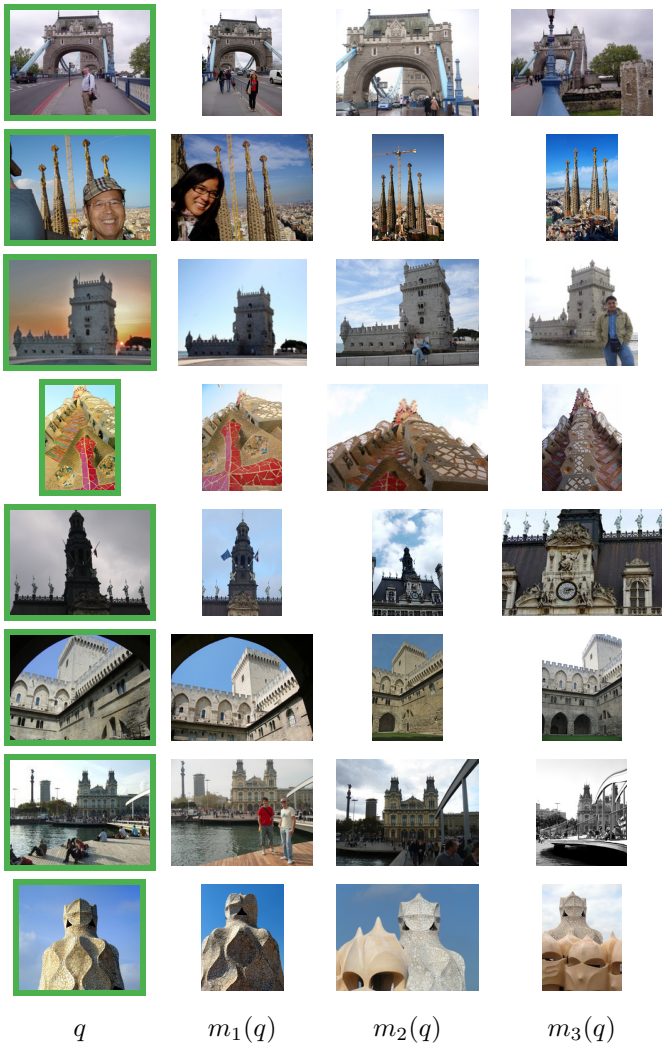


Fig. 6. Examples of training query images (green border) and matching images selected as positive examples by methods: $m_1(q)$ – the most similar image based on the current network; $m_2(q)$ – the most similar image based on the BoW representation; and our proposed $m_3(q)$ – a hard image depicting the same object.

4.2 Selection of training image pairs

A 3D model is described as a bipartite visibility graph $\mathbb{G} = (\mathcal{I} \cup \mathcal{P}, \mathcal{E})$ [65], where images $\mathcal{I}$ and points $\mathcal{P}$ are the vertices of the graph. The edges of this graph are defined by visibility relations between cameras and points, *i.e.* if a point $p \in \mathcal{P}$ is visible in an image $i \in \mathcal{I}$, then there exists an edge $(i, p) \in \mathcal{E}$. The set of points observed by an image i is given by

$$\mathcal{P}(i) = \{p \in \mathcal{P} : (i, p) \in \mathcal{E}\}. \tag{9}$$

We create a dataset of tuples $(q, m(q), \mathcal{N}(q))$, where q represents a query image, $m(q)$ is a positive image that matches the query, and $\mathcal{N}(q)$ is a set of negative images that do not match the query. These tuples are used to form training image pairs, where each tuple corresponds to $|\mathcal{N}(q)| + 1$ pairs. For a query image q, a pool $\mathcal{M}(q)$ of candidate positive images is constructed based on the camera positions in the cluster of q. It consists of the k images with camera centers closest to the query. Due to the wide range of camera orientations, these do not necessarily depict the same object. We therefore compare three different ways to select the positive image. The positive examples are fixed during the whole training process for all three strategies.

Positive images: CNN descriptor distance. The image that has the lowest descriptor distance to the query is chosen as positive, formally

$$m_1(q) = \underset{i \in \mathcal{M}(q)}{\operatorname{argmin}} ||\bar{\mathbf{f}}(q) - \bar{\mathbf{f}}(i)||. \tag{10}$$

This strategy is similar to the one followed by Arandjelovic *et al.* [16]. They adopt this choice since only GPS coordinates are available and not camera orientations. As a consequence, the chosen matching images already have small descriptor distance and, therefore, small loss too. The network is thus not forced to drastically change/learn by the matching examples, which is the drawback of this approach.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2846566, IEEE Transactions on Pattern Analysis and Machine Intelligence

Positive images: maximum inliers. In this approach, the 3D information is exploited to choose the positive image, independently of the CNN descriptor. In particular, the image that has the highest number of co-observed 3D points with the query is chosen. That is,

$$m_2(q) = \underset{i \in \mathcal{M}(q)}{\operatorname{argmax}} |\mathcal{P}(q) \cap \mathcal{P}(i)|. \tag{11}$$

This measure corresponds to the number of spatially verified features between two images, a measure commonly used for ranking in BoW-based retrieval. As this choice is independent of the CNN representation, it delivers more challenging positive examples.

Positive images: relaxed inliers. Even though both previous methods choose positive images depicting the same object as the query, the variance of viewpoints is limited. Instead of using a pool of images with similar camera position, the positive example is selected at random from a set of images that co-observe enough points with the query, but do not exhibit too extreme of a scale change. The positive example in this case is

$$m_3(q) = \texttt{rnd}\left\{ i \in \mathcal{M}(q) : \frac{|\mathcal{P}(i) \cap \mathcal{P}(q)|}{|\mathcal{P}(q)|} \geq t_i,\ \texttt{scale}(i,q) \leq t_s \right\}, \tag{12}$$

where $\texttt{scale}(i,q)$ is the scale change between the two images. This method results in selecting harder matching examples that are still guaranteed to depict the same object. Method m_3 chooses different image than m_1 on 86.5% of the queries. In Figure 6 we present examples of query images and the corresponding positives selected with the three different methods. The relaxed method increases the variability of viewpoints.

Negative images. Negative examples are selected from clusters different than the cluster of the query image, as the clusters are non-overlaping. We choose hard negatives [6], [46], that is, non-matching images with the most similar descriptor. Two different strategies are proposed: In the first, $\mathcal{N}_1(q)$, k-nearest neighbors from all non-matching images are selected. In the second, $\mathcal{N}_2(q)$, the same criterion is used, but at most one image per cluster is allowed. While $\mathcal{N}_1(q)$ often leads to multiple, and very similar, instances of the same object, $\mathcal{N}_2(q)$ provides higher variability of the negative examples, see Figure 5. While positives examples are fixed during the whole training process, hard negatives depend on the current CNN parameters and are re-mined multiple times per epoch.

5 Experiments

In this section we discuss implementation details of our training, evaluate different components of our method, and compare to the state of the art.

5.1 Training setup and implementation details

Structure-from-Motion (SfM). Our training samples are derived from the dataset used in the work of Schonberger *et al.* [17], which consists of 7.4 million images downloaded from Flickr using keywords of popular landmarks, cities and countries across the world. The clustering procedure [18] gives around 20k images to serve as query seeds. The extensive retrieval-SfM reconstruction [56] of the whole dataset results in $1,474$ reconstructed 3D models. Removing overlapping models leaves us with 713 3D models containing more than 163k unique images from the initial dataset. The initial dataset contains, on purpose, all images of Oxford5k and Paris6k datasets. In this way, we are able to exclude 98 clusters that contain any image (or their near duplicates) from these test datasets.

Training pairs. The size of the 3D models varies from 25 to 11k images. We randomly select 551 models (around 133k images) for training and 162 (around $30k$ images) for validation. The number of training queries per 3D model is 10% of its size and limited to be less or equal to 30. Around $6,000$ and $1,700$ images are selected for training and validation queries per epoch, respectively.

Each training and validation tuple contains 1 query, 1 positive and 5 negative images. The pool of candidate positives consists of $k = 100$ images with the closest camera centers to the query. In particular, for method m_3, the inlier-overlap threshold is $t_i = 0.2$, and the scale-change threshold $t_s = 1.5$. Hard negatives are re-mined 3 times per epoch, *i.e.* roughly every $2,000$ training queries. Given the chosen queries and the chosen positives, we further add 20 images per model to serve as candidate negatives during re-mining. This constitutes a training set of around 22k images per epoch when all the training 3D models are used. The query-tuple selection process is repeated every epoch. This slightly improves the results.

Learning configuration. We use MatConvNet [66] for the fine-tuning of networks. To perform the fine-tuning as described in Section 3, we initialize by the convolutional layers of AlexNet [1], VGG16 [23], or ResNet101 [24]. AlexNet is trained using stochastic gradient descent (SGD), while training of VGG and ResNet is more stable with Adam [67]. We use initial learning rate equal to $l_0 = 10^{-3}$ for SGD, initial stepsize equal to $l_0 = 10^{-6}$ for Adam, an exponential decay $l_0 \exp(-0.1i)$ over epoch i, momentum 0.9, weight decay 5×10^{-4}, margin τ for contrastive loss 0.7 for AlexNet, 0.75 for VGG, and 0.85 for ResNet, justified by the increase in the dimensionality of the embedding, and a batch size of 5 training tuples. All training images are resized to a maximum size of 362×362, while keeping the original aspect ratio. Training is done for at most 30 epochs and the best network is selected based on performance, measured via mean Average Precision (mAP) [33], on validation tuples. Fine-tuning of VGG for one epoch takes around 2 hours on a single TITAN X (Maxwell) GPU with 12 GB of memory.

We overcome GPU memory limitations by associating each query to a tuple, *i.e.*, query plus 6 images (5 positive and 1 negative). Moreover, the whole tuple is processed in the same batch. Therefore, we feed 7 images to the network, which represents 6 pairs. In a naive approach, when the query image is different for each pair, 6 pairs require 12 images.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2846566, IEEE Transactions on Pattern Analysis and Machine Intelligence

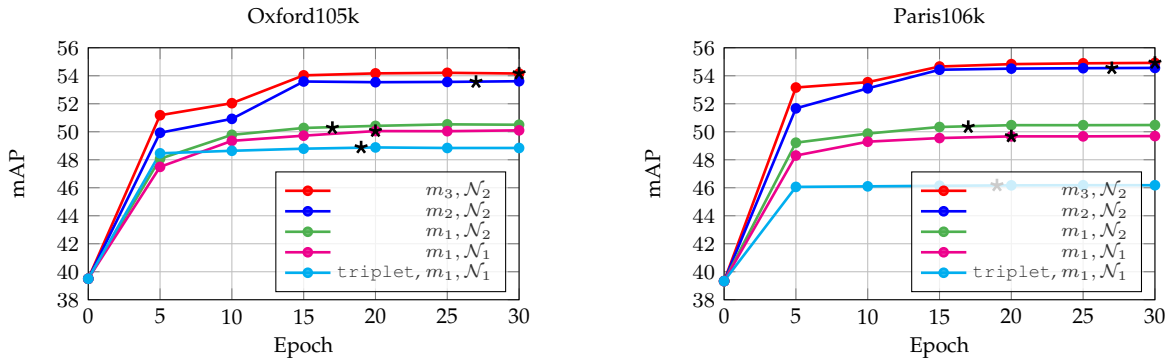


Fig. 7. Performance comparison of methods for positive and negative example selection. Evaluation is performed with AlexNet MAC on Oxford105k and Paris106k datasets. The plot shows the evolution of mAP with the number of training epochs. Epoch 0 corresponds to the off-the-shelf network. All approaches use the contrastive loss, except if otherwise stated. The network with the best performance on the validation set is marked with $\star$.

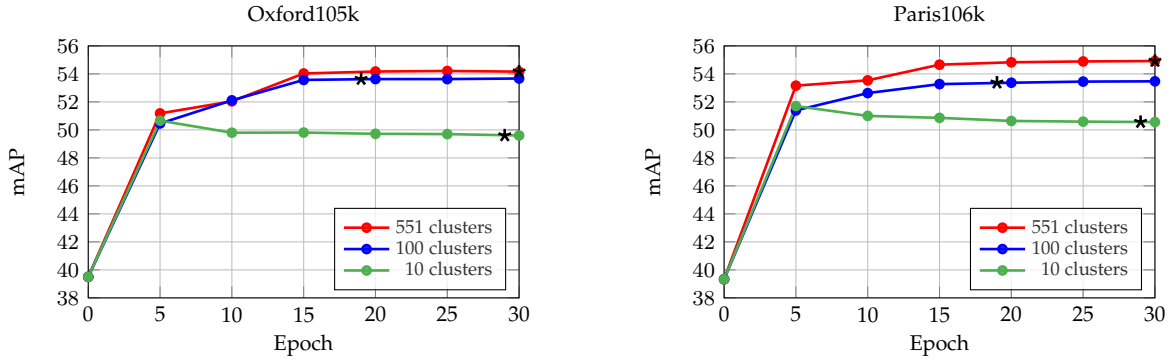


Fig. 8. Influence of the number of 3D models used for CNN fine-tuning. Performance is evaluated with AlexNet MAC on Oxford105k and Paris106k datasets using 10, 100 and 551 (all available) 3D models. The network with the best performance on the validation set is marked with $\star$.

5.2 Test datasets and evaluation protocol

Test datasets. We evaluate our approach on Oxford buildings [33], Paris [68] and Holidays[2] [69] datasets. The first two are closer to our training data, while the last is differentiated by containing similar scenes and not only man-made objects or buildings. These are also combined with 100k distractors from Oxford100k to allow for evaluation at larger scale. The performance is measured via mAP. We follow the standard evaluation protocol for Oxford and Paris and crop the query images with the provided bounding box. The cropped image is fed as input to the CNN.

Single-scale evaluation. The dimensionality of the images fed into the CNN is limited to 1024×1024 pixels. In our experiments, no vector post-processing is applied if not otherwise stated.

Multi-scale evaluation. Multi-scale representation is only used during test time. We resize the input image to different sizes, then feed multiple input images to the network, and finally combine the global descriptors from multiple scales into a single descriptor. We compare the baseline average pooling [27] with our generalized mean whose pooling parameter is equal to the value learned in the global pooling layer of the network. In this case, the whitening is learned on the final multi-scale image descriptors. In our experiments, a single-scale evaluation is used if not otherwise stated.

2. We use the up-right version of Holidays dataset where images are manually rotated so that depicted objects are up-right. This makes us directly comparable to [27]. A different version of up-right Holidays is used in our earlier work [25], where EXIF metadata is used to rotate the images.

5.3 Results on image retrieval

Learning. We evaluate the off-the-shelf CNN and our fine-tuned ones after different number of training epochs. The different methods for positive and negative selection are evaluated independently in order to isolate the benefit of each one. Finally, we also perform a comparison with the triplet loss [16], trained on the same training data as the contrastive loss. Note that a triplet forms two pairs. Results are presented in Figure 7. The results show that positive examples with larger viewpoint variability and negative examples with higher content variability acquire a consistent increase in the performance. The triplet loss[3] appears to be inferior in our context; we observe oscillation of the error in the validation set from early epochs, which implies over-fitting. In the rest of the paper, we adopt the $m_3, \mathcal{N}_2$ approach.

Dataset variability. We perform fine-tuning by using a subset of the available 3D models. Results are presented in Figure 8 with 10, 100 and 551 (all available) clusters, while keeping the amount of training data, *i.e.* number of training queries, fixed. In the case of 10 and 100 models, we use the largest ones. It is better to train with all 3D models due to the resulting higher variability in the training set. Remarkably, significant increase in performance is achieved even with 10 or 100 models. However, the network is able to over-fit in the case of few clusters. In the rest of our experiments we use all 551 3D models for training.

3. The margin parameter for the triplet loss is set equal to 0.1 [16].

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2846566, IEEE Transactions on Pattern Analysis and Machine Intelligence

TABLE 1
Performance (mAP) comparison after CNN fine-tuning for different pooling layers. GeM is evaluated with a single shared pooling parameter or multiple pooling parameters (one for each feature map), which are either fixed or learned. A single value or a range is reported in the case of a single or multiple parameters, respectively. Results reported with AlexNet and with the use of L_w. The best performance highlighted in **bold**.

Pooling	Initial p	Learned p	Oxford5k	Oxford105k	Paris6k	Paris106k	Holidays	Hol101k
MAC	inf	–	62.2	52.8	68.9	54.7	78.4	66.0
SPoC	1	–	61.2	54.9	70.8	58.0	79.9	70.6
GeM	3	–	**67.9**	60.2	74.8	61.7	83.2	73.3
	[2, 5]	–	66.8	59.7	74.1	60.8	**84.0**	73.6
	[2, 10]	–	65.6	57.8	72.2	58.9	81.9	71.9
	3	2.32	67.7	**60.6**	**75.5**	**62.6**	83.7	**73.7**
	3	[1.0, 6.5]	66.3	57.8	74.0	60.5	83.2	72.7
	[2, 10]	[1.6, 9.9]	65.3	56.4	71.4	58.6	81.4	70.8

TABLE 2
Performance (mAP) comparison of CNN vector post-processing: no post-processing, PCA-whitening [22] (PCA_w) and our learned whitening (L_w). No dimensionality reduction is performed. Fine-tuned AlexNet (Alex) produces a 256D vector and fine-tuned VGG a 512D vector. The best performance highlighted in **bold**, the worst in **blue**. The proposed method consistently performs either the best (22 out of 24 cases) or on par with the best method. On the contrary, PCA_w [22] often hurts the performance significantly. Best viewed in color.

Net	Post	Dim	Oxford5k		Oxford105k		Paris6k		Paris106k		Holidays		Hol101k	
			MAC	GeM	MAC	GeM	MAC	GeM	MAC	GeM	MAC	GeM	MAC	GeM
Alex	–	256	60.2	**60.1**	**54.2**	54.1	67.5	**68.6**	**54.9**	**56.9**	**74.5**	**78.7**	64.8	**70.9**
	PCA_w		**56.9**	63.7	**44.1**	**53.7**	**64.3**	73.2	**46.8**	57.4	75.4	82.5	**63.1**	71.8
	L_w		**62.2**	**67.7**	52.8	**60.6**	**68.9**	**75.5**	54.7	**62.6**	**78.4**	**83.7**	**66.0**	**73.7**
VGG	–	512	82.0	**82.0**	76.0	**76.9**	**78.3**	**79.7**	71.2	**72.6**	**79.9**	**83.1**	**69.4**	**74.5**
	PCA_w		**78.4**	83.1	**71.3**	77.7	80.6	84.5	**70.9**	76.9	82.2	86.6	70.0	75.9
	L_w		**82.3**	**85.9**	**77.0**	**81.7**	**83.8**	**86.0**	**76.2**	**79.6**	**84.1**	**87.3**	**71.9**	**77.1**

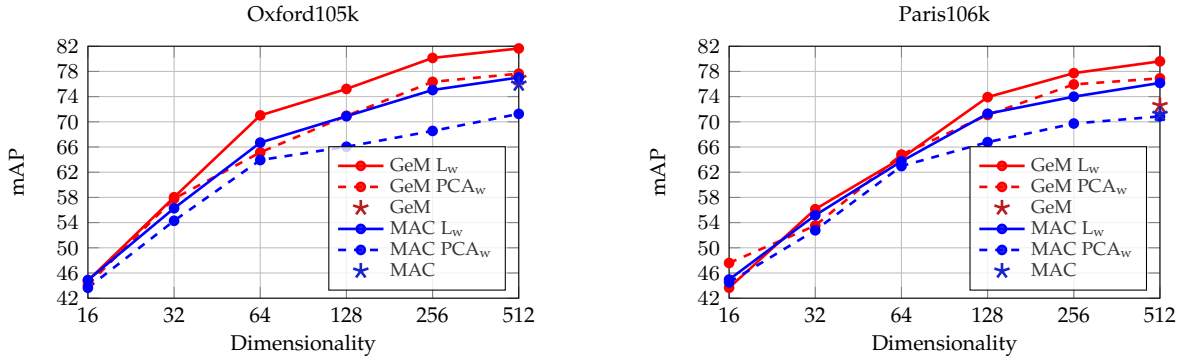


Fig. 9. Performance comparison of the dimensionality reduction performed by PCA_w and our L_w with the fine-tuned VGG with MAC layer and the fine-tuned VGG with GeM layer on Oxford105k and Paris106k datasets.

Pooling methods. We evaluate the effect of different pooling layers during CNN fine-tuning. We present the results in Table 1. GeM layer consistently outperforms the conventional max and average pooling. This holds for each of the following cases, (i) a single shared pooling parameter p is used, (ii) each feature map has different p_k and (iii) the pooling parameter(s) is (are) either fixed or learned. Learning a shared parameter turns out to be better than learning multiple ones, as the latter makes the cost function more complex. Additionally, the initial values seem to matter to some extent, with a preference for intermediate values. Finally, a shared fixed parameter and a shared learned parameter perform similarly, with the latter being slightly better. This is the case which we adopt for the rest of our experiments, *i.e.* a single shared parameter p that is learned.

Learned projections. The PCA-whitening [22] (PCA_w) is shown to be essential in some cases of CNN-based descriptors [10], [12], [15]. On the other hand, it is shown that on some datasets, the performance after PCA_w substantially drops compared to the raw descriptors (max pooling on Oxford5k [10]). We perform comparison of the traditional whitening methods and the proposed learned discriminative whitening (L_w), described in Section 3.4. Table 2 shows results without post-processing, with PCA_w and with L_w.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2846566, IEEE Transactions on Pattern Analysis and Machine Intelligence

TABLE 3
Performance (mAP) evaluation of the multi-scale representation using the fine-tuned VGG with GeM layer. The original scale and down-sampled versions of it are jointly represented. The pooling parameter used by the generalized mean is the same as the one learned in the GeM layer of the network and equal to 2.92. The results reported include the use of L_w.

Pooling over scales	Scale $1/1$	$1/\sqrt{2}$	$1/2$	$1/\sqrt{8}$	$1/4$	Oxford5k	Oxford105k	Paris6k	Paris106k	Holidays	Hol101k
–	■					85.9	81.7	86.0	79.6	87.3	77.1
Average	■	■				86.8	82.6	86.7	80.2	88.1	79.3
	■	■	■			87.2	82.4	87.3	80.6	89.1	79.6
	■	■	■	■		86.6	81.9	88.2	81.3	89.9	79.9
	■	■	■	■	■	85.1	80.1	88.8	81.6	90.6	80.5
Generalized mean	■	■				87.3	83.1	86.9	80.5	88.1	79.5
	■	■	■			87.9	83.3	87.7	81.3	89.5	79.9
	■	■	■	■		87.7	83.2	88.7	82.3	89.9	80.2
	■	■	■	■	■	86.8	82.4	89.4	82.7	91.1	81.4

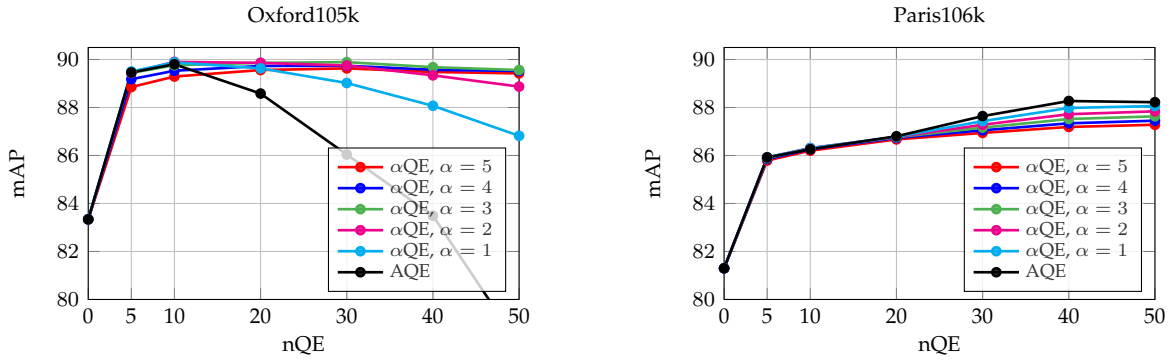


Fig. 10. Performance evaluation of our α-weighted query expansion (αQE) with the VGG with GeM layer, multi-scale representation, and L_w on Oxford105k and Paris106k datasets. We compare the standard average query expansion (AQE) to our αQE for different values of α and number of images used nQE.

TABLE 4
Performance (mAP) evaluation for varying descriptor dimensionality after reduction with L_w. Results reported with the fine-tuned VGG with GeM and the fine-tuned ResNet (Res) with GeM. Multi-scale representation is used at the test time for both networks.

Net	Dim	Oxf5k	Oxf105k	Par6k	Par106k	Hol	Hol101k
VGG	512	87.9	83.3	87.7	81.3	89.5	79.9
	256	85.4	79.7	85.7	78.2	87.8	77.2
	128	81.6	75.4	83.4	74.9	84.4	72.6
	64	77.0	69.9	77.4	66.7	81.1	66.2
	32	66.9	57.4	72.2	58.6	72.9	54.3
	16	56.2	44.4	63.5	45.5	60.9	36.9
	8	34.1	25.7	43.9	29.0	43.4	13.8
Res	2048	87.8	84.6	92.7	86.9	93.9	87.9
	1024	86.2	82.4	91.8	85.3	92.5	86.1
	512	84.6	80.4	90.0	82.6	90.6	83.2
	256	83.1	77.3	87.5	78.8	88.4	80.2
	128	79.5	72.2	84.5	74.3	85.9	76.5
	64	74.0	65.8	78.4	65.3	80.3	66.9
	32	57.9	48.5	70.8	56.1	71.2	51.9
	16	40.3	31.8	61.8	45.6	56.4	31.3
	8	25.3	16.3	44.3	27.8	37.8	11.4

Our experiments confirm that PCA_w often reduces the performance. In contrast to that, the proposed L_w achieves the best performance in most cases and is never the worst-performing method. Compared with the no post-processing baseline, L_w reduces the performance twice for AlexNet, but the drop is negligible compared to the drop observed for PCA_w. For VGG, the proposed L_w *always* outperforms the no post-processing baseline.

We conduct an additional experiment by appending a whitening layer at the end of the network during fine-tuning. In this way, whitening is learned in an end-to-end manner, along with the convolutional filters and with the same training data in batch-mode. Dropout [70] is additionally used for this layer which we find to be essential. We observe that convergence of the network comes much slower in this case, *i.e.* after 60 epochs. Moreover, the final achieved performance is not higher than our L_w. In particular, end-to-end whitening on AlexNet MAC achieves 49.6 and 52.1 mAP on Oxford105k and Paris106k, respectively, while our L_w on the same network achieves 52.8 and 54.7 mAP on Oxford105k and Paris106k, respectively. Therefore, we adopt L_w as it is much faster to train and more effective.

Dimensionality reduction. We compare dimensionality reduction performed with PCA_w [22] and with our L_w. The performance for varying descriptor dimensionality is plotted in Figure 9. The plots suggest that L_w works better in most dimensionalities.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2846566, IEEE Transactions on Pattern Analysis and Machine Intelligence

TABLE 5
Performance (mAP) comparison with the state-of-the-art image retrieval using VGG and ResNet (Res) deep networks, and using local features. F-tuned: Use of the fine-tuned network (yes), or the off-the-shelf network (no), not applicable for the methods using local features (n/a). Dim: Dimensionality of the final compact image representation, not applicable (n/a) for the BoW based methods due to their sparse representation. Our methods are marked with ⋆ and they are always accompanied by the multi-scale representation and our learned whitening L_w. Previous state of the art is highlighted in **bold**, new state of the art in **red outline**. Best viewed in color.

Net	Method	F-tuned	Dim	Oxford5k	Oxford105k	Paris6k	Paris106k	Holidays	Hol101k
Compact representation using deep networks									
VGG	MAC [9]†	no	512	56.4	47.8	72.3	58.0	79.0	66.1
	SPoC [10]†	no	512	68.1	61.1	78.2	68.4	83.9	**75.1**
	CroW [11]	no	512	70.8	65.3	79.7	72.2	85.1	–
	R-MAC [12]	no	512	66.9	61.6	83.0	75.7	86.9‡	–
	BoW-CNN [48]	no	n/a	73.9	59.3	82.0	64.8	–	–
	NetVLAD [16]	no	4096	66.6	–	77.4	–	88.3	–
	NetVLAD [16]	yes	512	67.6	–	74.9	–	86.1	–
	NetVLAD [16]	yes	4096	71.6	–	79.7	–	87.5	–
	Fisher Vector [49]	yes	512	81.5	76.6	82.4	–	–	–
	R-MAC [26]	yes	512	**83.1**	**78.6**	**87.1**	**79.7**	**89.1**	–
	⋆ GeM	yes	512	**87.9**	**83.3**	**87.7**	**81.3**	**89.5**	**79.9**
Res	R-MAC [12]‡	no	2048	69.4	63.7	85.2	77.8	91.3	–
	R-MAC [27]	yes	2048	**86.1**	**82.8**	**94.5**	**90.6**	**94.8**	–
	⋆ GeM	yes	2048	**87.8**	**84.6**	92.7	86.9	93.9	**87.9**
Re-ranking (R) and query expansion (QE)									
n/a	BoW+R+QE [36]	n/a	n/a	82.7	76.7	80.5	71.0	–	–
	BoW-fVocab+R+QE [59]	n/a	n/a	84.9	79.5	82.4	77.3	75.8	–
	HQE [38]	n/a	n/a	88.0	84.0	82.8	–	–	–
VGG	CroW+QE [11]	no	512	74.9	70.6	84.8	79.4	–	–
	R-MAC+R+QE [12]	no	512	77.3	73.2	86.5	79.8	–	–
	BoW-CNN+R+QE [48]	no	n/a	78.8	65.1	84.8	64.1	–	–
	R-MAC+QE [26]	yes	512	**89.1**	**87.3**	**91.2**	**86.8**	–	–
	⋆ GeM+αQE	yes	512	**91.9**	**89.6**	**91.9**	**87.6**	–	–
Res	R-MAC+QE [12]‡	no	2048	78.9	75.5	89.7	85.3	–	–
	R-MAC+QE [27]	yes	2048	**90.6**	**89.4**	**96.0**	**93.2**	–	–
	⋆ GeM+αQE	yes	2048	**91.0**	**89.5**	95.5	91.9	–	–

†: Our evaluation of MAC and SPoC with PCA_w and with the off-the-shelf network.
‡: Evaluation of R-MAC by [27] with the off-the-shelf network.

Multi-scale representation. We evaluate multi-scale representation constructed at test time without any additional learning. We compare the previously used averaging of descriptors at multiple image scales [27] with our generalized-mean of the same descriptors. Results are presented in Table 3, where there is a significant benefit when using the multi-scale GeM. It also offers some improvement over average pooling. In the rest of our experiments we adopt multi-scale representation, pooled by generalized mean, for scales 1, $1/\sqrt{2}$, and $1/2$. Results using the supervised dimensionality reduction by L_w on the multi-scale GeM representation are shown in Table 4.

Query expansion. We evaluate the proposed αQE, which reduces to AQE for $\alpha = 0$, and present results in Figure 10. Note that Oxford and Paris have different statistics in terms of the number of relevant images per query. The average, minimum, and maximum number of positive images per query on Oxford is 52, 6, and 221, respectively. The same measurements for Paris are 163, 51, and 289. As a consequence, AQE behaves in a very different way across these dataset, while our αQE is a more stable choice. We finally set $\alpha = 3$ and nQE $= 50$.

Over-fitting and generalization. In all experiments, all clusters including any image (not only query landmarks) from Oxford5k or Paris6k datasets are removed. We now repeat the training using all 3D models, including those of Oxford and Paris landmarks. In this way, we evaluate whether the network tends to over-fit to the training data or to generalize. The same amount of training queries is used for a fair comparison. We observe negligible difference in the performance of the network on Oxford and Paris evaluation results, *i.e.* the difference in mAP was on average $+0.3$ over all testing datasets. We conclude that the network generalizes well and is relatively insensitive to over-fitting.

Comparison with the state of the art. We extensively compare our results with the state-of-the-art performance on compact image representations and on approaches that do query expansion. The results for the fine-tuned GeM based networks are summarized together with previously published results in Table 5. The proposed methods outperform the state of the art on all datasets when the VGG network architecture and initialization are used. Our method is outperformed by the work of Gordo *et al.* on Paris with the ResNet architecture, while we have the state-of-the-art score on Oxford. We are on par with the state-of-the-

art on Holidays. Note, however, that we did not perform any manual labeling or cleaning of our training data, while in their work landmark labels were used. We additionally combine GeM with query expansion and further boost the performance.

6 Conclusions

We addressed fine-tuning of CNN for image retrieval. The training data are selected from an automated 3D reconstruction system applied on a large unordered photo collection. The reconstructions consist of buildings and popular landmarks; however, the same process is applicable to any rigid 3D objects. The proposed method does not require any manual annotation and yet achieves top performance on standard benchmarks. The achieved results reach the level of the best systems based on local features with spatial matching and query expansion while being faster and requiring less memory. The proposed pooling layer that generalizes previously adopted mechanisms is shown to improve the retrieval accuracy while it is also effective for constructing a joint multi-scale representation. Training data, trained models, and code are publicly available.

Acknowledgments

The authors were supported by the MSMT LL1303 ERC-CZ grant. We would also like to thank Karel Lenc for insightful discussions.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012. 1, 2, 4, 8

[2] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *IJCV*, 2015. 1, 3

[3] H. Azizpour, A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson, "From generic to specific deep representations for visual recognition," in *CVPRW*, 2015. 1

[4] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "DeCAF: A deep convolutional activation feature for generic visual recognition," in *ICML*, 2014. 1

[5] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *CVPRW*, 2014. 1, 3

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014. 1, 3, 8

[7] F. Iandola, M. Moskewicz, S. Karayev, R. Girshick, T. Darrell, and K. Keutzer, "DenseNet: Implementing efficient ConvNet descriptor pyramids," in *arXiv:1404.1869*, 2014. 1

[8] Y. Gong, L. Wang, R. Guo, and S. Lazebnik, "Multi-scale orderless pooling of deep convolutional activation features," in *ECCV*, 2014. 1, 3

[9] A. S. Razavian, J. Sullivan, S. Carlsson, and A. Maki, "Visual instance retrieval with deep convolutional networks," *ITE Trans. MTA*, 2016. 1, 3, 4, 12

[10] A. Babenko and V. Lempitsky, "Aggregating deep convolutional features for image retrieval," in *ICCV*, 2015. 1, 3, 4, 6, 10, 12

[11] Y. Kalantidis, C. Mellina, and S. Osindero, "Cross-dimensional weighting for aggregated deep convolutional features," in *ECCVW*, 2016. 1, 3, 6, 12

[12] G. Tolias, R. Sicre, and H. Jégou, "Particular object retrieval with integral max-pooling of CNN activations," in *ICLR*, 2016. 1, 3, 4, 6, 10, 12

[13] N. Zhang, J. Donahue, R. Girshick, and T. Darrell, "Part-based R-CNNs for fine-grained category detection," in *ECCV*, 2014. 1

[14] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *CVPR*, 2014. 1

[15] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky, "Neural codes for image retrieval," in *ECCV*, 2014. 1, 3, 10

[16] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *CVPR*, 2016. 1, 3, 5, 7, 9, 12

[17] J. L. Schönberger, F. Radenović, O. Chum, and J.-M. Frahm, "From single image query to detailed 3D reconstruction," in *CVPR*, 2015. 1, 6, 8

[18] O. Chum and J. Matas, "Large-scale discovery of spatially related images," *PAMI*, 2010. 1, 6, 8

[19] T. Weyand and B. Leibe, "Discovering details and scene structure with hierarchical iconoid shift," in *ICCV*, 2013. 1

[20] J. Philbin, J. Sivic, and A. Zisserman, "Geometric latent dirichlet allocation on a matching graph for large-scale image datasets," *IJCV*, 2011. 1

[21] A. Mousavian and J. Kosecka, "Deep convolutional features for image based retrieval and scene categorization," in *arXiv:1509.06033*, 2015. 1, 3

[22] H. Jégou and O. Chum, "Negative evidences and co-occurences in image retrieval: The benefit of PCA and whitening," in *ECCV*, 2012. 2, 3, 10, 11

[23] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *arXiv:1409.1556*, 2014. 2, 4, 8

[24] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016. 2, 4, 8

[25] F. Radenović, G. Tolias, and O. Chum, "CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples," in *ECCV*, 2016. 2, 9

[26] A. Gordo, J. Almazan, J. Revaud, and D. Larlus, "Deep image retrieval: Learning global representations for image search," in *ECCV*, 2016. 2, 3, 12

[27] A. Gordo, J. Almazan, J. Revaud, and D. Larlus, "End-to-end learning of deep visual representations for image retrieval," *IJCV*, 2017. 2, 6, 9, 12

[28] F. Perronnin, Y. Liu, J. Sanchez, and H. Poirier, "Large-scale image retrieval with compressed Fisher vectors," in *CVPR*, 2010. 2

[29] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid, "Aggregating local descriptors into compact codes," *PAMI*, 2012. 2

[30] F. Radenović, H. Jegou, and O. Chum, "Multiple measurements and joint dimensionality reduction for large scale image search with short vectors," in *ICMR*, 2015. 2

[31] R. Arandjelovic and A. Zisserman, "All about VLAD," in *CVPR*, 2013. 2

[32] G. Tolias, T. Furon, and H. Jégou, "Orientation covariant aggregation of local descriptors with embeddings," in *ECCV*, 2014. 2

[33] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *CVPR*, 2007. 2, 6, 8, 9

[34] Y. Avrithis and Y. Kalantidis, "Approximate Gaussian mixtures for large scale vocabularies," in *ECCV*, 2012. 2

[35] X. Shen, Z. Lin, J. Brandt, and Y. Wu, "Spatially-constrained similarity measure for large-scale object retrieval," *PAMI*, 2014. 2

[36] O. Chum, A. Mikulik, M. Perdoch, and J. Matas, "Total recall II: Query expansion revisited," in *CVPR*, 2011. 2, 12

[37] Q. Danfeng, S. Gammeter, L. Bossard, T. Quack, and L. V. Gool, "Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors," in *CVPR*, 2011. 2

[38] G. Tolias and H. Jégou, "Visual query expansion with or without geometry: refining local descriptors by feature aggregation," *Pattern Recognition*, 2014. 2, 12

[39] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *CVPR*, 2005. 2, 5

[40] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *CVPR*, 2006. 2

[41] J. Hu, J. Lu, and Y.-P. Tan, "Discriminative deep metric learning for face verification in the wild," in *CVPR*, 2014. 2

[42] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, "Learning fine-grained image similarity with deep ranking," in *CVPR*, 2014. 2, 5

[43] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *CVPR*, 2015. 2, 5

[44] E. Hoffer and N. Ailon, "Deep metric learning using triplet network," in *ICLRW*, 2015. 2, 5

[45] L. Zheng, Y. Zhao, S. Wang, J. Wang, and Q. Tian, "Good practice in CNN feature transfer," in *arXiv:1604.00133*, 2016. 3

[46] E. Simo-Serra, E. Trulls, L. Ferraz, I. Kokkinos, and F. Moreno-Noguer, "Fracking deep convolutional image descriptors," in *arXiv:1412.6537*, 2014. 3, 8

[47] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han, "Large-scale image retrieval with attentive deep local features," in *ICCV*, 2017. 3

[48] E. Mohedano, K. McGuinness, N. E. O'Connor, A. Salvador, F. Marques, and X. Giro-i Nieto, "Bags of local convolutional features for scalable instance search," in *ICMR*, 2016. 3, 12

[49] E.-J. Ong, S. Husain, and M. Bober, "Siamese network of deep fisher-vector descriptors for image retrieval," in *arXiv:1702.00338*, 2017. 3, 12

[50] C.-Y. Lee, P. W. Gallagher, and Z. Tu, "Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree," in *AISTATS*, 2016. 3

[51] N. Cohen, O. Sharir, and A. Shashua, "Deep simnets," in *CVPR*, 2016. 3

[52] O. Morère, J. Lin, A. Veillard, L.-Y. Duan, V. Chandrasekhar, and T. Poggio, "Nested invariance pooling and rbm hashing for image instance retrieval," in *ICMR*, 2017. 3

[53] K. Mikolajczyk and J. Matas, "Improving descriptors for fast tree matching by optimal linear projection," in *ICCV*, 2007. 3, 6

[54] G. Papandreou, I. Kokkinos, and P.-A. Savalle, "Modeling local and global deformations in deep learning: Epitomic convolution, multiple instance learning, and sliding window detection," in *CVPR*, 2015. 4

[55] P. Dollár, Z. Tu, P. Perona, and S. Belongie, "Integral channel features." in *BMVC*, 2009. 4

[56] F. Radenović, J. L. Schönberger, D. Ji, J.-M. Frahm, O. Chum, and J. Matas, "From dusk till dawn: Modeling in the dark," in *CVPR*, 2016. 6, 8

[57] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. V. Gool, "A comparison of affine region detectors," *IJCV*, 2005. 6

[58] R. Arandjelovic and A. Zisserman, "Three things everyone should know to improve object retrieval," in *CVPR*, 2012. 6

[59] A. Mikulik, M. Perdoch, O. Chum, and J. Matas, "Learning vocabularies over a fine quantization," *IJCV*, 2013. 6, 12

[60] J.-M. Frahm, P. Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, and M. Pollefeys, "Building Rome on a cloudless day," in *ECCV*, 2010. 6

[61] S. Agarwal, Y. Furukawa, N. Snavely, I. Simon, B. Curless, S. M. Seitz, and R. Szeliski, "Building Rome in a day," *Communications of the ACM*, 2011. 6

[62] J. L. Schönberger and J.-M. Frahm, "Structure-from-Motion revisited," in *CVPR*, 2016. 6

[63] A. Mikulik, O. Chum, and J. Matas, "Image retrieval for online browsing in large image collections," in *SISAP*, 2013. 6

[64] A. Mikulík, F. Radenović, O. Chum, and J. Matas, "Efficient image detail mining," in *ACCV*, 2014. 6

[65] Y. Li, N. Snavely, and D. P. Huttenlocher, "Location recognition using prioritized feature matching," in *ECCV*, 2010. 7

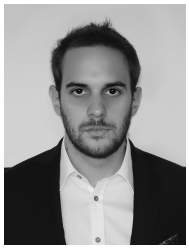
[66] A. Vedaldi and K. Lenc, "MatConvNet: Convolutional neural networks for matlab," in *ACM Multimedia*, 2015. 8

[67] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015. 8

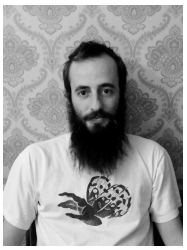
[68] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Lost in quantization: Improving particular object retrieval in large scale image databases," in *CVPR*, 2008. 9

[69] H. Jégou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *ECCV*, 2008. 9

[70] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *JMLR*, 2014. 11



Filip Radenović obtained his Master's degree from the Faculty of Electrical Engineering, University of Montenegro in 2013. Currently, he is a PhD candidate at the Visual Recognition Group, which is a part of the Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague. His research interests are mainly large-scale image retrieval problems.



Giorgos Tolias obtained his PhD from NTU of Athens and then moved to Inria Rennes as a post-doctoral researcher. Currently, he is a post-doctoral researcher at the Visual Recognition Group of CTU in Prague. He enjoys working on large-scale visual recognition problems.

Ondřej Chum is leading a team within the Visual Recognition Group at the Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University in Prague. He received the MSc degree in computer science from Charles University, Prague, in 2001 and the PhD degree from the Czech Technical University in Prague, in 2005. From 2006 to 2007, he was a postdoctoral researcher at the Visual Geometry Group, University of Oxford, United Kingdom. His research interests include large-scale image and particular object retrieval, object recognition, and robust estimation of geometric models. He is a member of Image and Vision Computing editorial board, and he has served in various roles at major international conferences. He co-organizes Computer Vision and Sports Summers School in Prague. He was the recipient of the Best Paper Prize at the British Machine Vision Conference in 2002. He was awarded the 2012 Outstanding Young Researcher in Image and Vision Computing runner up for researchers within seven years of their PhD. In 2017, he was the recipient of the Longuet-Higgins Prize.