CNN Image Retrieval Learns from BoW: Unsupervised Fine-Tuning with Hard Examples

Filip Radenović Giorgos Tolias Ondřej Chum

CMP, Faculty of Electrical Engineering, Czech Technical University in Prague
{filip.radenovic,giorgos.tolias,chum}@cmp.felk.cvut.cz

Abstract. Convolutional Neural Networks (CNNs) achieve state-of-the-art performance in many computer vision tasks. However, this achievement is preceded by extreme manual annotation in order to perform either training from scratch or fine-tuning for the target task. In this work, we propose to fine-tune CNN for image retrieval from a large collection of unordered images in a fully automated manner. We employ state-of-the-art retrieval and Structure-from-Motion (SfM) methods to obtain 3D models, which are used to guide the selection of the training data for CNN fine-tuning. We show that both hard positive and hard negative examples enhance the final performance in particular object retrieval with compact codes.

Keywords: CNN fine-tuning, unsupervised learning, image retrieval

1 Introduction

Image retrieval has received a lot of attention since the advent of invariant local features, such as SIFT [1], and since the seminal work of Sivic and Zisserman [2] based on Bag-of-Words (BoW). Retrieval systems have reached a higher level of maturity by incorporating large visual codebooks [3,4], spatial verification [3,5] and query expansion [6,7,8]. These ingredients constitute the state of the art on particular object retrieval. Another line of research focuses on compact image representations in order to decrease memory requirements and increase the search efficiency. Representative approaches are Fisher vectors [9], VLAD [10] and alternatives [11,12,13]. Recent advances [14,15] show that Convolutional Neural Networks (CNN) offer an attractive alternative for image search representations with small memory footprint.

CNNs attracted a lot of attention after the work of Krizhevsky *et al.* [16]. Their success is mainly due to the computational power of GPUs and the use of very large annotated datasets [17]. Generation of the latter comes at the expense of costly manual annotation. Using CNN layer activations as off-the-shelf image descriptors [18,19] appears very effective and is adopted in many tasks [20,21,22]. In particular for image retrieval, Babenko *et al.* [14] and Gong *et al.* [22] concurrently propose the use of Fully Connected (FC) layer activations as descriptors, while convolutional layer activations are later shown to have superior performance [15,23,24,25].

Generalization to other tasks [26] is attained by CNN activations, at least up to some extent. However, initialization by a pre-trained network and re-training for another task, a process called *fine-tuning*, significantly improves the adaptation ability [27,28]. Fine-tuning by training with classes of particular objects, *e.g.* building classes in the work of Babenko *et al.* [14], is known to improve retrieval accuracy. This formulation is much closer to classification than to the desired properties of instance retrieval. Typical architectures for metric learning, such as siamese [29,30,31] or triplet networks [32,33,34] employ *matching* and *non-matching* pairs to perform the training and better suit to this task. In this fashion, Arandjelovic *et al.* [35] perform fine-tuning based on geo-tagged databases and, similar to our work, they directly optimize the the similarity measure to be used in the final task. In contrast to them, we dispense with the need of annotated data or any assumptions on the training dataset. A concurrent work [36] bears resemblance to ours but their focus is on boosting performance through end-to-end learning of a more sophisticated representation, while we target to reveal the importance of hard examples and of training data variation.

A number of image clustering methods based on local features have been introduced [37,38,39]. Due to the 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. These graphs are further used as an input to a Structure-from-Motion (SfM) pipeline to build a 3D model [40]. The SfM filters out virtually all mismatched images, and also provides camera positions for all matched images in the cluster. The whole process from unordered collection of images to 3D reconstructions is fully automatic.

In this paper, we address an unsupervised fine-tuning of CNN for image retrieval. We propose to exploit 3D reconstructions to select the training data for CNN. 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. During the training process the CNN is trained to learn what a state-of-the-art retrieval system based on local features and spatial verification would match. Such a system has large memory requirements and high query times, while our goal is to mimic this via CNN-based representation. We derive a short image representation and achieve similar performance to such state-of-the-art systems.

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 to be learned by the CNN. This is shown to enhance the derived image representation. (2) We show that the whitening traditionally performed on short representations [41] is, in some cases, unstable and we rather propose to learn the whitening through the same training data. Its effect is complementary to fine-tuning and it further boosts performance. (3) Finally, we set a new state-of-the-art based on compact representations for Oxford Buildings and Paris datasets by re-training well known CNNs, such as AlexNet [16] and VGG [42]. Remarkably, we are on par with existing 256D compact representations even by using 32D image vectors.

2 Related work

A variety of previous methods apply CNN activations on the task of image retrieval [22,15,23,24,25,43]. The achieved accuracy on retrieval is evidence for the generalization properties of CNNs. The employed networks were trained for image classification using ImageNet dataset, optimizing classification error. Babenko *et al.* [14] 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 utilized layers are different to the ones actually optimized during learning.

Constructing such training datasets requires manual effort. The same stands for attempts on different tasks [19,25] that perform fine-tuning and achieve increase of performance. In a recent work, geo-tagged datasets with timestamps offer the ground for weakly supervised fine-tuning of a triplet network [35]. 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 task of geo-localization. The employed training data are now much closer 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 reconstruction which allows for harder examples, compared to the ones that the current network identifies. Even though hard negative mining is a standard process [20,35], this is not the case with hard positive examples. Large intra-class variation in classification tasks requires the positive pairs to be sampled carefully; forcing the model to learn extremely hard positives may result in over-fitting. Another exception is the work Simo-Serra *et al.* [44] where they mine hard positive patches for descriptor learning. They are also guided by 3D reconstruction but only at patch level.

Despite the fact that one of the recent advances is the triplet loss [32,33,34], note that also Arandjelovic *et al.* [35] use it, there are no extenstive and direct comparisons to siamese networks and the contrastive loss. One exception is the work of Hoffer and Ailon [34], where triplet loss is shown to be marginally better only on MNIST dataset. We rather employ a siamese architecture with the contrastive loss and find it to generalize better and to converge at higher performance than the triplet loss.

3 Network architecture and image representation

In this section we describe the derived image representation that is based on CNN and we present the network architecture used to perform the end-to-end learning in a siamese fashion. Finally, we describe how, after fine-tuning, we use the same training data to learn projections that appear to be an effective post-processing step.

3.1 Image representation

We adopt a compact representation that is derived from activations of convolutional layers and is shown to be effective for particular object retrieval [26,25]. We assume that a network is fully convolutional [45] or that all fully connected layers are discarded. 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 all $W \times H$ activations for feature map $k \in \{1 \dots K\}$. The network output consists of K such sets of activations. The image representation, called Maximum Activations of Convolutions (MAC) [15,25], is simply constructed by max-pooling over all dimensions per feature map and is given by

$$\mathbf{f} = [\mathrm{f}_1 \dots \mathrm{f}_k \dots \mathrm{f}_K]^\top, \text{ with } \mathrm{f}_k = \max_{x \in \mathcal{X}_k} x \cdot \mathbb{1}(x > 0). \quad (1)$$

The indicator function $\mathbb{1}$ takes care that the feature vector $\mathbf{f}$ is non-negative, as if the last network layer was a Rectified Linear Unit (ReLU). The feature vector finally consists of the maximum activation per feature map and its dimensionality is equal to K. For many popular networks this is equal to 256 or 512, which makes it a compact image representation. MAC vectors are subsequently ℓ_2-normalized and similarity between two images is evaluated with inner product. The contribution of a feature map to the image similarity is measured by the product of the corresponding MAC vector components. In Figure 1 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 to ImageNet classes, *e.g.* cars and bicycles.

3.2 Network and siamese learning

The proposed approach is applicable to any CNN that consists of only convolutional layers. In this paper, we focus on re-training (*i.e.* fine-tuning) state-of-the-art CNNs for classification, in particular AlexNet and VGG. Fully connected layers are discarded and the pre-trained networks constitute the initialization for our convolutional layers. Now, the last convolutional layer is followed by a MAC layer that performs MAC vector computation (1). The input of a MAC layer is a 3D tensor of activation and the output is a non-negative vector. Then, an ℓ_2-normalization block takes care that output vectors are normalized. In the rest of the paper, MAC corresponds to the ℓ_2-normalized vector $\bar{\mathbf{f}}$.

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. 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 [29] that acts on the (non-)matching pairs and is defined as

$$\mathcal{L}(i,j) = \frac{1}{2}\left(Y(i,j)||\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)||^2 + (1 - Y(i,j))\left(\max\{0, \tau - ||\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)||\}\right)^2\right), \quad (2)$$

where $\bar{\mathbf{f}}(i)$ is the ℓ_2-normalized MAC vector of image i, and τ is a parameter defining when non-matching pairs have large enough distance in order not to be taken into account in the loss. We train the network using Stochastic Gradient Descent (SGD) and a large training set created automatically (see Section 4).

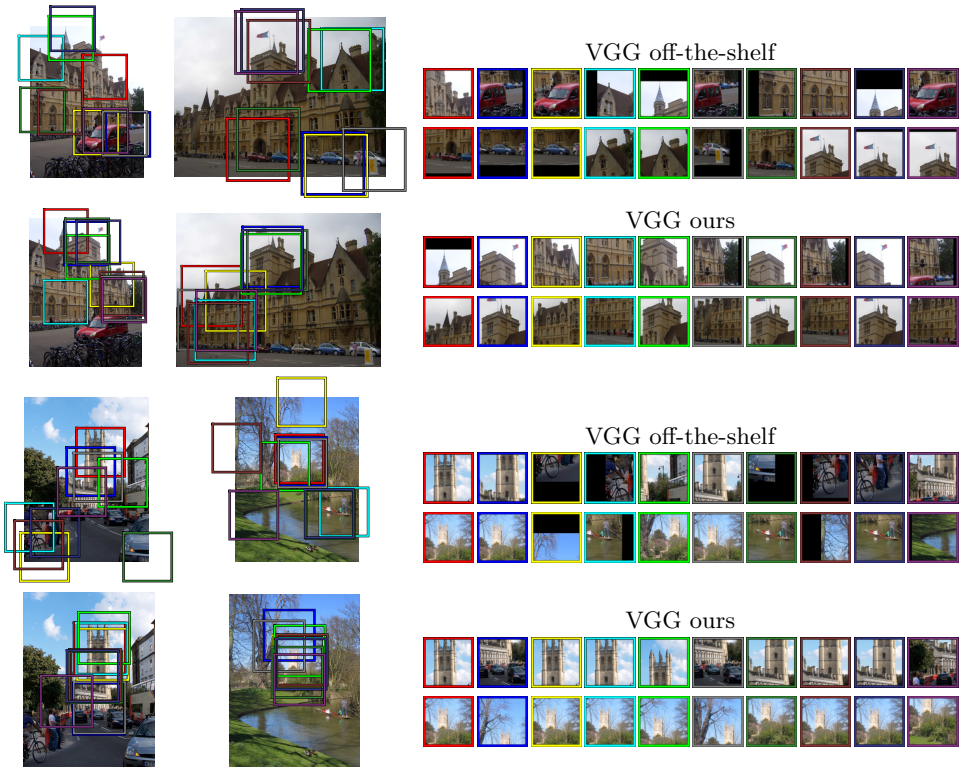

Fig. 1. Visualization of patches corresponding to the MAC vector components that have the highest contribution to the pairwise image similarity. Examples shown use CNN before (top) and after (bottom) fine-tuning of VGG. The same color corresponds to the same vector component (feature map) per image pair. The patch size is equal to the receptive field of the last pooling layer.

3.3 Whitening and dimensionality reduction

In this section, the post-processing of fine-tuned MAC vectors is considered. Previous methods [23,25] use PCA of an independent set for whitening and dimensionality reduction, that is the covariance matrix of all descriptors is analyzed. We propose to take advantage of the labeled data provided by the 3D models and use linear discriminant projections originally proposed by Mikolajczyk and Matas [46]. 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

$$C_S = \sum_{Y(i,j)=1} \left(\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)\right)\left(\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)\right)^\top. \tag{3}$$

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

$$C_D = \sum_{Y(i,j)=0} \left(\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)\right)\left(\bar{\mathbf{f}}(i) - \bar{\mathbf{f}}(j)\right)^\top. \tag{4}$$

The projection $P = C_S^{-\frac{1}{2}} \text{eig}(C_S^{-\frac{1}{2}} C_D C_S^{-\frac{1}{2}})$ is then applied as $P^\top(\bar{\mathbf{f}}(i) - \mu)$, where μ is the mean MAC 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.

4 Training dataset

In this section we briefly summarize the tightly-coupled BoW and SfM reconstruction system [40,47] that is employed to automatically select our training data. Then, we describe how we exploit 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.* [40] builds upon BoW with fast spatial verification [3]. It uses Hessian affine local features [48], RootSIFT descriptors [49], and a fine vocabulary of 16M visual words [50]. Then, query images are chosen via min-hash and spatial verification, as in [37]. 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 state-of-the-art SfM [51,52]. Different mining techniques, *e.g.* zoom in, zoom out [53,54], sideways crawl [40], help to build larger and 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.

4.2 Selection of training image pairs

A 3D model is described as a bipartite visibility graph $\mathbb{G} = (\mathcal{I} \cup \mathcal{P}, \mathcal{E})$ [55], where images $\mathcal{I}$ and points $\mathcal{P}$ are the vertices of the graph. 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

$$\mathcal{P}(i) = \{p \in \mathcal{P} : (i, p) \in \mathcal{E}\}. \tag{5}$$

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

Fig. 2. Examples of training query images (green border) and matching images selected as positive examples by methods (from left to right) $m_1(q)$, $m_2(q)$, and $m_3(q)$.

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 closest camera centers to the query. Due to the wide range of camera orientations, these do not necessarily depict the same object. We therefore propose three different ways to sample the positive image. The positives examples are fixed during the whole training process for all three strategies.

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

$$m_1(q) = \underset{i \in \mathcal{M}(q)}{\operatorname{argmin}} ||\bar{\mathbf{f}}(q) - \bar{\mathbf{f}}(i)||. \tag{6}$$

This strategy is similar to the one followed by Arandjelovic *et al.* [35]. They adopt this choice since only GPS coordinates are available and not camera orientations. Downside of this approach is that the chosen matching examples already have low distance, thus not forcing network to learn much out of the positive samples.

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,

$$m_2(q) = \underset{i \in \mathcal{M}(q)}{\operatorname{argmax}} |\mathcal{P}(q) \cap \mathcal{P}(i)|. \tag{7}$$

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.

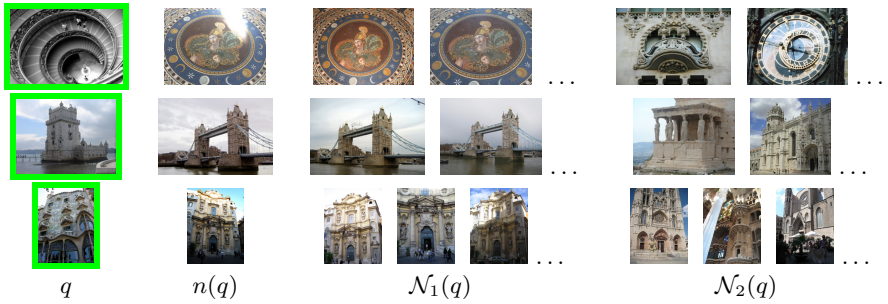

Fig. 3. Examples of training query images q (green border), hardest non-matching images $n(q)$ that are always selected as negative examples, and additional non-matching images selected as negative examples by $\mathcal{N}_1(q)$ and $\mathcal{N}_2(q)$ methods respectively.

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 scale change. The positive example in this case is

$$m_3(q) = \texttt{random}\left\{ i \in \mathcal{M}(q) : \frac{|\mathcal{P}(i) \cap \mathcal{P}(q)|}{|\mathcal{P}(q)|} \geq t_i,\ \texttt{scale}(i,q) \leq t_s \right\}, \quad (8)$$

where $\texttt{scale}(i,q)$ is the scale change between the two images. This method results in selecting harder matching examples which 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 2 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. Following a well-known procedure, we choose hard negatives [44,20], 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 other, $\mathcal{N}_2(q)$, 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 3. 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

Our training samples are derived from the dataset used in the work of Schonberger *et al.* [40], which consists of 7.4 million images downloaded from Flickr using keywords of popular landmarks, cities and countries across the world. The clustering procedure [37] gives 19,546 images to serve as query seeds. The extensive retrieval-SfM reconstruction [47] of the whole dataset results in 1,474 reconstructed 3D models. Removing overlapping models leaves us with 713 3D models containing 163,671 unique images from the initial dataset. The initial dataset contained 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.

The largest model has 11,042 images, while the smallest has 25. We randomly select 551 models (133,659 images) for training and 162 (30,012) for validation. The number of training queries per cluster is 10% of the cluster size for clusters of 300 or less images, or 30 images for larger clusters. A total number of 5,974 images is selected for training queries, and 1,691 for validation queries.

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 closest camera centers to the query. In particular, for method m_3, the inliers 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 cluster to serve as candidate negatives during re-mining. This constitutes a training set of 22,156 images and it corresponds to the case that all 3D models are included for training.

To perform the fine-tuning as described in Section 3, we initialize by the convolutional layers of AlexNet [16] or VGG [42]. We use learning rate equal to 0.001, which is divided by 5 every 10 epochs, momentum 0.9, weight decay 0.0005, parameter τ for contrastive loss 0.7, and batch size of 5 training tuples. All training images are resized to a maximum 362×362 dimensionality, 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) [3], on validation tuples.

5.2 Test datasets and evaluation protocol

We evaluate our approach on Oxford buildings [3], Paris [56] and Holidays[1] [57] datasets. First two are closer to our training data, while the last differentiates by containing similar scenes and not only man made objects or buildings. These

[1] We use the up-right version of Holidays dataset (rotated).

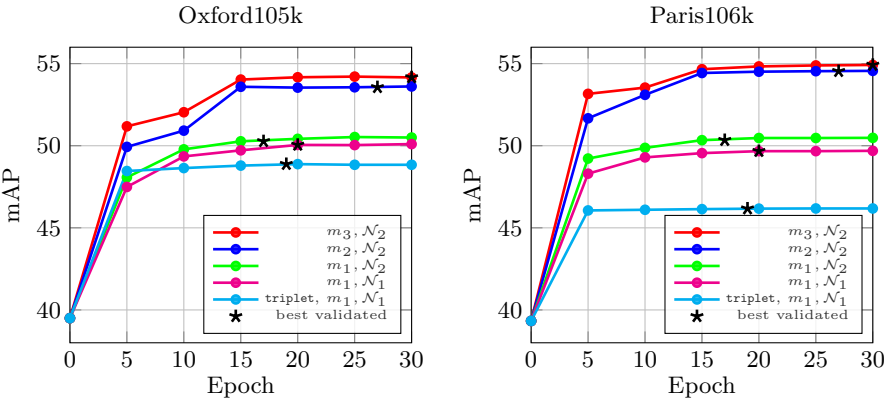

Fig. 4. Performance comparison of methods for positive and negative example selection. Evaluation is performed on AlexNet MAC on Oxford105k and Paris106k datasets. The plot shows the evolution of mAP with the number of training epochs. Epoch 0 corresponds to the off-the-shelf network. All approaches use contrastive loss, except if otherwise stated. The network with the best performance on the validation set is marked with $\star$.

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. However, to deliver a direct comparison with other methods, we also evaluate queries generated by keeping all activations that fall into this bounding box [23,35] when the full query image is used as input to the network. We refer to the cropped images approach as $\texttt{Crop}_{\mathcal{I}}$ and the cropped activations [23,35] as $\texttt{Crop}_{\mathcal{X}}$. 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.

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. Our different methods for positive and negative selection are evaluated independently in order to decompose the benefit of each ingredient. Finally, we also perform a comparison with the triplet loss [35], trained on exactly the same training data as the ones used for our architecture with the contrastive loss. Results are presented in Figure 4. The results show that positive examples with larger view point variability, and negative examples with higher content variability, both acquire a consistent increase in the performance. The triplet loss[2] 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.

[2] The margin parameter for the triplet loss is set equal to 0.1 [35].

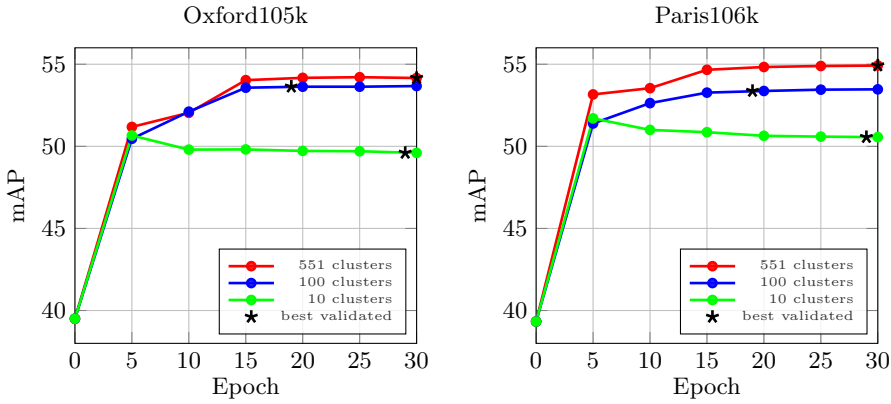

Fig. 5. Influence of the number of 3D models used for CNN fine-tuning. Performance is evaluated on AlexNet MAC on Oxford105k and Paris106k datasets using 10, 100 and 551 (all available) 3D models. The network with the best performance on the validation set is marked with $\star$.

Dataset variability. We perform fine-tuning by using a subset of the available 3D models. Results are presented in Figure 5 with 10, 100 and 551 (all available) clusters, while keeping the amount of training data, *i.e.* training queries, fixed. In the case of 10 and 100 models we use the largest ones, *i.e.* ones with the highest number of images. It is better to train with all 3D models due to the 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. All models are utilized in all other experiments.

Learned projections. The PCA-whitening [41] (PCA_w) is shown to be essential in some cases of CNN-based descriptors [14,23,25]. On the other hand, it is shown that on some of the datasets, the performance after PCA_w substantially drops compared with the raw descriptors (max pooling on Oxford5k [23]). We perform comparison of this traditional way of whitening and our learned whitening (L_w), described in Section 3.3. Table 1 shows results without post-processing and with the two different methods of whitening. 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 to 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 the VGG, the proposed L_w *always* outperforms the no post-processing baseline.

Our unsupervised learning directly optimizes MAC when extracted from full images, however, we further apply the fine-tuned networks to construct R-MAC representation [25] with regions of three different scales. It consists of extracting

Table 1. Performance comparison of CNN vector post-processing: no post-processing, PCA-whitening [41] (PCA_w) and our learned whitening (L_w). No dimensionality reduction is performed. Fine-tuned AlexNet 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 (18 out of 24 cases) or on par with the best method. On the contrary, PCA_w [41] often hurts the performance significantly. Best viewed in color.

Net	Post	Oxf5k		Oxf105k		Par6k		Par106k		Hol		Hol101k	
		MAC	R-MAC	MAC	R-MAC	MAC	R-MAC	MAC	R-MAC	MAC	R-MAC	MAC	R-MAC
Alex	–	60.2	**53.9**	**54.2**	**46.4**	67.5	**70.2**	**54.9**	**58.4**	73.1	**77.3**	61.6	**67.1**
	PCA_w	**56.9**	60.0	**44.1**	48.4	**64.3**	**75.1**	**46.8**	61.7	**73.0**	**81.7**	**59.4**	70.4
	L_w	**62.2**	**62.5**	52.8	**53.2**	**68.9**	74.4	54.7	**61.8**	**76.2**	81.5	**63.8**	**70.8**
VGG	–	78.7	**70.1**	72.7	**63.1**	**77.1**	**78.1**	69.6	**70.4**	**76.9**	**80.0**	65.3	**68.8**
	PCA_w	**76.1**	76.3	**68.9**	68.5	79.0	**84.5**	**69.1**	**77.1**	77.1	82.3	**63.6**	71.0
	L_w	**79.7**	**77.0**	**73.9**	**69.2**	**82.4**	83.8	**74.6**	76.4	**79.5**	**82.5**	**67.0**	**71.5**

MAC from multiple sub-windows and then aggregating them. Directly optimizing R-MAC during learning is possible and could offer extra improvements, but this is left for future work. Despite the fact that R-MAC offers improvements due to the regional representation, in our experiments it is not always better than MAC, since the latter is optimized during the end-to-end learning. We apply PCA_w on R-MAC as in [25], that is, we whiten each region vector first and then aggregate. Performance is significantly higher in this way. In the case of our L_w, we directly whiten the final vector after aggregation, which is also faster to compute.

Dimensionality reduction. We compare dimensionality reduction performed with PCA_w [41] and with our L_w. The performance for varying descriptor dimensionality is plotted in Figure 6. The plots suggest that L_w works better in higher dimensionalities, while PCA_w works slightly better for the lower ones. Remarkably, MAC reduced down to 16D outperforms state-of-the-art on BoW-based 128D compact codes [11] on Oxford105k (45.5 vs 41.4). Further results on very short codes can be found in Table 2.

Over-fitting and generalization. In all experiments, all clusters including any image (not only query landmarks) from Oxford5k or Paris6k datasets are removed. To evaluate whether the network tends to over-fit to the training data or to generalize, we repeat the training, this time using all 3D reconstructions, including those of Oxford and Paris landmarks. 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.

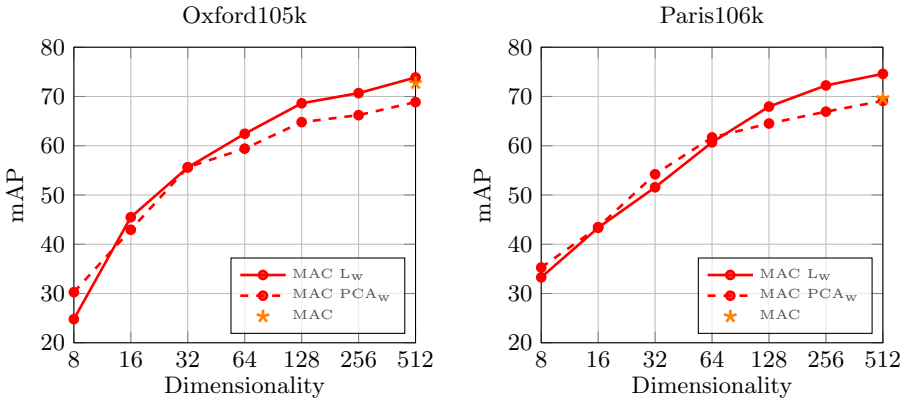

Fig. 6. Performance comparison of the dimensionality reduction performed by PCA_w and our L_w on fine-tuned VGG MAC on Oxford105k and Paris106k datasets.

Comparison with the state of the art. We extensively compare our results with the state-of-the-art performance on compact image representations and extremely short codes. The results for MAC and R-MAC with the fine-tuned networks are summarized together with previously published results in Table 2. The proposed methods outperform the state of the art on Paris and Oxford datasets, with and without distractors with all 16D, 32D, 128D, 256D, and 512D descriptors. On Holidays dataset, the Neural codes [14] win the extreme short code category, while off-the-shelf NetVlad performs the best on 256D and higher.

We additionally combine MAC and R-MAC with recent localization method for re-ranking [25] to further boost the performance. Our scores compete with state-of-the-art systems based on local features and query expansion. These have much higher memory needs and larger query times.

Observations on the recently published NetVLAD [35]: (1) After fine-tuning, NetVLAD performance drops on Holidays, while our training improves off-the-shelf results on all datasets. (2) Our 32D MAC descriptor has comparable performance to 256D NetVLAD on Oxford5k (ours 69.2 vs NetVLAD 63.5), and on Paris6k (ours 69.5 vs NetVLAD 73.5).

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 proposed method does not require any manual annotation and yet outperforms the state of the art on a number of standard benchmarks for wide range (16 to 512) of descriptor dimensionality. The achieved results are reaching the level of the best systems based on local features with spatial matching and query expansion, while being faster and requiring less memory. Training data, fine-tuned networks and evaluation code are publicly available[3].

[3] http://cmp.felk.cvut.cz/~radenfil/projects/siamac.html

Table 2. Performance comparison with the state of the art. Results reported with the use of AlexNet or VGG are marked by (**A**) or (**V**), respectively. Use of fine-tuned network is marked by (**f**), otherwise the off-the-shelf network is implied. D: Dimensionality. Our methods are marked with ⋆ and they are always accompanied by L_w. New state of the art highlighted in **red**, surpassed state of the art in **bold**, state of the art that retained the title in **outline**, and our methods that outperform previous state of the art on a gray background. Best viewed in color.

Method		D	Oxf5k		Oxf105k		Par6k		Par106k		Hol	Hol
			$\text{Crop}_{\mathcal{T}}$	$\text{Crop}_{\mathcal{H}}$	$\text{Crop}_{\mathcal{T}}$	$\text{Crop}_{\mathcal{H}}$	$\text{Crop}_{\mathcal{T}}$	$\text{Crop}_{\mathcal{H}}$	$\text{Crop}_{\mathcal{T}}$	$\text{Crop}_{\mathcal{H}}$		101k
Compact representations												
mVoc/BoW [11]		128	48.8	–	41.4	–	–	–	–	–	65.6	–
Neural codes† [14]	(**fA**)	128	–	**55.7**	–	**52.3**	–	–	–	–	**78.9**	–
MAC‡	(**V**)	128	53.5	**55.7**	43.8	45.6	69.5	**70.6**	53.4	**55.4**	72.6	**56.7**
CroW [24]	(**V**)	128	**59.2**	–	**51.6**	–	**74.6**	–	**63.2**	–	–	–
⋆ MAC	(**fV**)	128	**75.8**	**76.8**	**68.6**	**70.8**	77.6	78.8	68.0	69.0	73.2	58.8
⋆ R-MAC	(**fV**)	128	72.5	76.7	64.3	69.7	**78.5**	**80.3**	**69.3**	**71.2**	**79.3**	**65.2**
MAC‡	(**V**)	256	54.7	56.9	45.6	47.8	71.5	72.4	55.7	**57.3**	76.5	**61.3**
SPoC [23]	(**V**)	256	–	53.1	–	**50.1**	–	–	–	–	80.2	–
R-MAC [25]	(**A**)	256	56.1	–	47.0	–	72.9	–	60.1	–	–	–
CroW [24]	(**V**)	256	**65.4**	–	**59.3**	–	**77.9**	–	**67.8**	–	83.1	–
NetVlad [35]	(**V**)	256	–	55.5	–	–	–	67.7	–	–	**86.0**	–
NetVlad [35]	(**fV**)	256	–	**63.5**	–	–	–	**73.5**	–	–	84.3	–
⋆ MAC	(**fA**)	256	62.2	65.4	52.8	58.0	68.9	72.2	54.7	58.5	76.2	63.8
⋆ R-MAC	(**fA**)	256	62.5	68.9	53.2	61.2	74.4	76.6	61.8	64.8	81.5	**70.8**
⋆ MAC	(**fV**)	256	**77.4**	**78.2**	**70.7**	**72.6**	80.8	81.9	72.2	73.4	77.3	62.9
⋆ R-MAC	(**fV**)	256	74.9	**78.2**	67.5	72.1	**82.3**	**83.5**	**74.1**	**75.6**	81.4	69.4
MAC‡	(**V**)	512	56.4	**58.3**	47.8	**49.2**	72.3	**72.6**	58.0	**59.1**	76.7	**62.7**
R-MAC [25]	(**V**)	512	66.9	–	61.6	–	**83.0**	–	**75.7**	–	–	–
CroW [24]	(**V**)	512	**68.2**	–	**63.2**	–	79.6	–	71.0	–	84.9	–
⋆ MAC	(**fV**)	512	**79.7**	80.0	**73.9**	**75.1**	82.4	82.9	74.6	75.3	79.5	67.0
⋆ R-MAC	(**fV**)	512	77.0	**80.1**	69.2	74.1	**83.8**	**85.0**	**76.4**	**77.9**	82.5	**71.5**
Extreme short codes												
Neural codes† [14]	(**fA**)	16	–	**41.8**	–	**35.4**	–	–	–	–	**60.9**	–
⋆ MAC	(**fV**)	16	**56.2**	**57.4**	**45.5**	**47.6**	57.3	62.9	43.4	48.5	51.3	25.6
⋆ R-MAC	(**fV**)	16	46.9	52.1	37.9	41.6	**58.8**	**63.2**	**45.6**	**49.6**	54.4	**31.7**
Neural codes† [14]	(**fA**)	32	–	**51.5**	–	**46.7**	–	–	–	–	**72.9**	–
⋆ MAC	(**fV**)	32	**65.3**	**69.2**	**55.6**	**59.5**	**63.9**	**69.5**	51.6	**56.3**	62.4	41.8
⋆ R-MAC	(**fV**)	32	58.4	64.2	50.1	55.1	**63.9**	67.4	**52.7**	55.8	68.0	**49.6**
Re-ranking (R) and query expansion (QE)												
BoW(1M)+QE [6]		–	82.7	–	76.7	–	80.5	–	71.0	–	–	–
BoW(16M)+QE [50]		–	84.9	–	79.5	–	82.4	–	77.3	–	–	–
HQE(65k) [8]		–	**88.0**	–	**84.0**	–	82.8	–	–	–	–	–
R-MAC+R+QE [25]	(**V**)	512	77.3	–	73.2	–	**86.5**	–	**79.8**	–	–	–
CroW+QE [24]	(**V**)	512	72.2	–	67.8	–	85.5	–	79.7	–	–	–
⋆ MAC+R+QE	(**fV**)	512	85.0	**85.4**	81.8	**82.3**	**86.5**	**87.0**	78.8	79.6	–	–
⋆ R-MAC+R+QE	(**fV**)	512	82.9	84.5	77.9	80.4	85.6	86.4	78.3	**79.7**	–	–

†: Full images are used as queries making the results not directly comparable on Oxford and Paris.
‡: Our evaluation of MAC with PCA_w as in [25] with the off-the-shelf network.

Acknowledgment. Work was supported by the MSMT LL1303 ERC-CZ grant.

References

1. Lowe, D.: Distinctive image features from scale-invariant keypoints. IJCV (2004) 1
2. Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. In: ICCV. (2003) 1
3. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object retrieval with large vocabularies and fast spatial matching. In: CVPR. (2007) 1, 6, 9
4. Avrithis, Y., Kalantidis, Y.: Approximate Gaussian mixtures for large scale vocabularies. In: ECCV. (2012) 1
5. Shen, X., Lin, Z., Brandt, J., Wu, Y.: Spatially-constrained similarity measure for large-scale object retrieval. PAMI (2014) 1
6. Chum, O., Mikulik, A., Perdoch, M., Matas, J.: Total recall II: Query expansion revisited. In: CVPR. (2011) 1, 14
7. Danfeng, Q., Gammeter, S., Bossard, L., Quack, T., Gool, L.V.: Hello neighbor: Accurate object retrieval with k-reciprocal nearest neighbors. In: CVPR. (2011) 1
8. Tolias, G., Jégou, H.: Visual query expansion with or without geometry: refining local descriptors by feature aggregation. Pattern Recognition (2014) 1, 14
9. Perronnin, F., Liu, Y., Sanchez, J., Poirier, H.: Large-scale image retrieval with compressed Fisher vectors. In: CVPR. (2010) 1
10. Jégou, H., Perronnin, F., Douze, M., Sánchez, J., Pérez, P., Schmid, C.: Aggregating local descriptors into compact codes. PAMI (2012) 1
11. Radenović, F., Jegou, H., Chum, O.: Multiple measurements and joint dimensionality reduction for large scale image search with short vectors. In: ICMR. (2015) 1, 12, 14
12. Arandjelovic, R., Zisserman, A.: All about VLAD. In: CVPR. (2013) 1
13. Tolias, G., Furon, T., Jégou, H.: Orientation covariant aggregation of local descriptors with embeddings. In: ECCV. (2014) 1
14. Babenko, A., Slesarev, A., Chigorin, A., Lempitsky, V.: Neural codes for image retrieval. In: ECCV. (2014) 1, 2, 3, 11, 13, 14
15. Razavian, A.S., Sullivan, J., Maki, A., Carlsson, S.: A baseline for visual instance retrieval with deep convolutional networks. In: arXiv:1412.6574. (2014) 1, 3, 4
16. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: NIPS. (2012) 1, 2, 9
17. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. IJCV (2015) 1
18. Donahue, J., Jia, Y., Vinyals, O., Hoffman, J., Zhang, N., Tzeng, E., Darrell, T.: DeCAF: A deep convolutional activation feature for generic visual recognition. In: arXiv:1310.1531. (2013) 1
19. Razavian, A.S., Azizpour, H., Sullivan, J., Carlsson, S.: CNN features off-the-shelf: An astounding baseline for recognition. In: CVPRW. (2014) 1, 3
20. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: CVPR. (2014) 1, 3, 8
21. Iandola, F., Moskewicz, M., Karayev, S., Girshick, R., Darrell, T., Keutzer, K.: DenseNet: Implementing efficient ConvNet descriptor pyramids. In: arXiv:1404.1869. (2014) 1

22. Gong, Y., Wang, L., Guo, R., Lazebnik, S.: Multi-scale orderless pooling of deep convolutional activation features. In: ECCV. (2014) 1, 3
23. Babenko, A., Lempitsky, V.: Aggregating deep convolutional features for image retrieval. In: ICCV. (2015) 1, 3, 5, 10, 11, 14
24. Kalantidis, Y., Mellina, C., Osindero, S.: Cross-dimensional weighting for aggregated deep convolutional features. In: arXiv:1512.04065. (2015) 1, 3, 14
25. Tolias, G., Sicre, R., Jégou, H.: Particular object retrieval with integral max-pooling of CNN activations. In: ICLR. (2016) 1, 3, 4, 5, 11, 12, 13, 14
26. Azizpour, H., Razavian, A.S., Sullivan, J., Maki, A., Carlsson, S.: From generic to specific deep representations for visual recognition. In: CVPRW. (2015) 2, 4
27. Zhang, N., Donahue, J., Girshick, R., Darrell, T.: Part-based R-CNNs for fine-grained category detection. In: ECCV. (2014) 2
28. Oquab, M., Bottou, L., Laptev, I., Sivic, J.: Learning and transferring mid-level image representations using convolutional neural networks. In: CVPR. (2014) 2
29. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: CVPR. (2005) 2, 4
30. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality reduction by learning an invariant mapping. In: CVPR. (2006) 2
31. Hu, J., Lu, J., Tan, Y.P.: Discriminative deep metric learning for face verification in the wild. In: CVPR. (2014) 2
32. Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, J., Philbin, J., Chen, B., Wu, Y.: Learning fine-grained image similarity with deep ranking. In: CVPR. (2014) 2, 3
33. Schroff, F., Kalenichenko, D., Philbin, J.: FaceNet: A unified embedding for face recognition and clustering. In: CVPR. (2015) 2, 3
34. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. In: ICLR Workshop. (2015) 2, 3
35. Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: NetVLAD: CNN architecture for weakly supervised place recognition. In: CVPR. (2016) 2, 3, 7, 10, 13, 14
36. Gordo, A., Almazan, J., Revaud, J., Larlus, D.: Deep image retrieval: Learning global representations for image search. In: ECCV. (2016) 2
37. Chum, O., Matas, J.: Large-scale discovery of spatially related images. PAMI (2010) 2, 6, 9
38. Weyand, T., Leibe, B.: Discovering details and scene structure with hierarchical iconoid shift. In: ICCV. (2013) 2
39. Philbin, J., Sivic, J., Zisserman, A.: Geometric latent dirichlet allocation on a matching graph for large-scale image datasets. IJCV (2011) 2
40. Schönberger, J.L., Radenović, F., Chum, O., Frahm, J.M.: From single image query to detailed 3D reconstruction. In: CVPR. (2015) 2, 6, 9
41. Jégou, H., Chum, O.: Negative evidences and co-occurences in image retrieval: The benefit of PCA and whitening. In: ECCV. (2012) 2, 11, 12
42. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: arXiv:1409.1556. (2014) 2, 9
43. Zheng, L., Zhao, Y., Wang, S., Wang, J., Tian, Q.: Good practice in CNN feature transfer. In: arXiv:1604.00133. (2016) 3
44. Simo-Serra, E., Trulls, E., Ferraz, L., Kokkinos, I., Moreno-Noguer, F.: Fracking deep convolutional image descriptors. In: arXiv:1412.6537. (2014) 3, 8
45. Papandreou, G., Kokkinos, I., Savalle, P.A.: Modeling local and global deformations in deep learning: Epitomic convolution, multiple instance learning, and sliding window detection. In: CVPR. (2015) 4

46. Mikolajczyk, K., Matas, J.: Improving descriptors for fast tree matching by optimal linear projection. In: ICCV. (2007) 5
47. Radenović, F., Schönberger, J.L., Ji, D., Frahm, J.M., Chum, O., Matas, J.: From dusk till dawn: Modeling in the dark. In: CVPR. (2016) 6, 9
48. Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., Gool, L.V.: A comparison of affine region detectors. IJCV (2005) 6
49. Arandjelovic, R., Zisserman, A.: Three things everyone should know to improve object retrieval. In: CVPR. (2012) 6
50. Mikulik, A., Perdoch, M., Chum, O., Matas, J.: Learning vocabularies over a fine quantization. IJCV (2013) 6, 14
51. Frahm, J.M., Georgel, P., Gallup, D., Johnson, T., Raguram, R., Wu, C., Jen, Y.H., Dunn, E., Clipp, B., Lazebnik, S., Pollefeys, M.: Building Rome on a cloudless day. In: ECCV. (2010) 6
52. Agarwal, S., Furukawa, Y., Snavely, N., Simon, I., Curless, B., Seitz, S.M., Szeliski, R.: Building Rome in a day. Communications of the ACM (2011) 6
53. Mikulik, A., Chum, O., Matas, J.: Image retrieval for online browsing in large image collections. In: SISAP. (2013) 6
54. Mikulík, A., Radenović, F., Chum, O., Matas, J.: Efficient image detail mining. In: ACCV. (2014) 6
55. Li, Y., Snavely, N., Huttenlocher, D.P.: Location recognition using prioritized feature matching. In: ECCV. (2010) 6
56. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Lost in quantization: Improving particular object retrieval in large scale image databases. In: CVPR. (2008) 9
57. Jégou, H., Douze, M., Schmid, C.: Hamming embedding and weak geometric consistency for large scale image search. In: ECCV. (2008) 9