Unifying Deep Local and Global Features for Image Search

Bingyi Cao* André Araujo* Jack Sim

Google Research, USA
{bingyi,andrearaujo,jacksim}@google.com

Abstract. Image retrieval is the problem of searching an image database for items that are similar to a query image. To address this task, two main types of image representations have been studied: global and local image features. In this work, our key contribution is to unify global and local features into a single deep model, enabling accurate retrieval with efficient feature extraction. We refer to the new model as DELG, standing for DEep Local and Global features. We leverage lessons from recent feature learning work and propose a model that combines generalized mean pooling for global features and attentive selection for local features. The entire network can be learned end-to-end by carefully balancing the gradient flow between two heads – requiring only image-level labels. We also introduce an autoencoder-based dimensionality reduction technique for local features, which is integrated into the model, improving training efficiency and matching performance. Comprehensive experiments show that our model achieves state-of-the-art image retrieval on the Revisited Oxford and Paris datasets, and state-of-the-art single-model instance-level recognition on the Google Landmarks dataset v2. Code and models are available at https://github.com/tensorflow/models/tree/master/research/delf.

Keywords: deep features, image retrieval, unified model

1 Introduction

Large-scale image retrieval is a long-standing problem in computer vision, which saw promising results [38,44,45,26] even before deep learning revolutionized the field. Central to this problem are the representations used to describe images and their similarities.

Two types of image representations are necessary for high image retrieval performance: global and local features. A global feature [26,1,17,47,48], also commonly referred to as "global descriptor" or "embedding", summarizes the contents of an image, often leading to a compact representation; information about spatial arrangement of visual elements is lost. Local features [28,7,64,39,34], on the other hand, comprise descriptors and geometry information about specific image regions; they are especially useful to match images depicting rigid objects. Generally speaking, global features are better at recall, while local features are better at precision. Global features can learn similarity across very different poses where local features would not be able to find correspondences; in contrast, the score provided by local feature-based geometric verification

* Both authors contributed equally to this work.

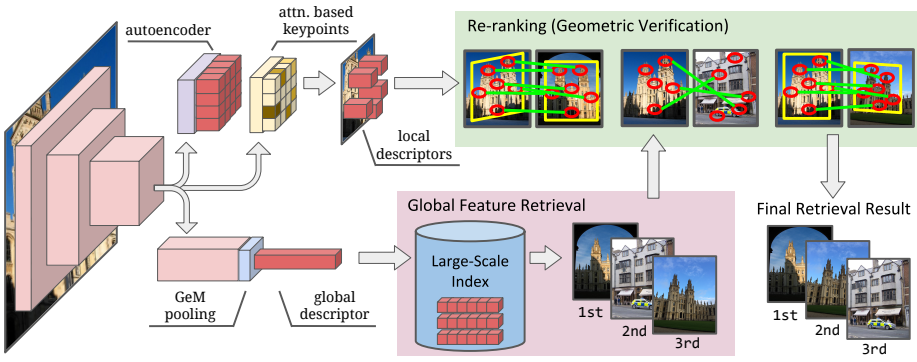


Fig. 1: Our proposed **DELG (DEep Local and Global features)** model (left) jointly extracts deep local and global features. Global features can be used in the first stage of a retrieval system, to efficiently select the most similar images (bottom). Local features can then be employed to re-rank top results (top-right), increasing precision of the system. The unified model leverages hierarchical representations induced by convolutional neural networks to learn local and global features, combined with recent advances in global pooling and attentive local feature detection.

usually reflects image similarity well, being more reliable than global feature distance. A common retrieval system setup is to first search by global features, then re-rank the top database images using local feature matching – to get the best of both worlds. Such a hybrid approach gained popularity in visual localization [55,50] and instance-level recognition problems [43,63].

Today, most systems that rely on both these types of features need to separately extract each of them, using different models. This is undesirable since it may lead to high memory usage and increased latency, e.g., if both models require specialized and limited hardware such as GPUs. Besides, in many cases similar types of computation are performed for both, resulting in redundant processing and unnecessary complexity.

Contributions. (1) Our first contribution is a unified model to represent both local and global features, using a convolutional neural network (CNN), referred to as DELG (DEep Local and Global features) – illustrated in Fig. 1. This allows for efficient inference by extracting an image's global feature, detected keypoints and local descriptors within a single model. Our model is enabled by leveraging hierarchical image representations that arise in CNNs [66], which we couple to generalized mean pooling [47] and attentive local feature detection [39]. (2) Second, we adopt a convolutional autoencoder module that can successfully learn low-dimensional local descriptors. This can be readily integrated into the unified model, and avoids the need of post-processing learning steps, such as PCA, that are commonly used. (3) Finally, we design a procedure that enables end-to-end training of the proposed model using only image-level supervision. This requires carefully controlling the gradient flow between the global and local network heads during backpropagation, to avoid disrupting the desired representations. Through systematic experiments, we show that our joint model achieves state-of-the-art performance on the Revisited Oxford, Revisited Paris and Google Landmarks v2 datasets.

2 Related Work

We review relevant work in local and global features, focusing mainly on approaches related to image retrieval.

Local features. Hand-crafted techniques such as SIFT [28] and SURF [7] have been widely used for retrieval problems. Early systems [32,28,40] worked by searching for query local descriptors against a large database of local descriptors, followed by geometrically verifying database images with sufficient number of correspondences. Bag-of-Words [54] and related methods [44,45,24] followed, by relying on visual words obtained via local descriptor clustering, coupled to TF-IDF scoring. The key advantage of local features over global ones for retrieval is the ability to perform spatial matching, often employing RANSAC [15]. This has been widely used [44,45,3], as it produces reliable and interpretable scores. Recently, several deep learning-based local features have been proposed [64,39,33,42,34,49,14,6,29]. The one most related to our work is DELF [39]; our proposed unified model incorporates DELF's attention module, but with a much simpler training pipeline, besides also enabling global feature extraction.

Global features excel at delivering high image retrieval performance with compact representations. Before deep learning was popular in computer vision, they were developed mainly by aggregating hand-crafted local descriptors [25,26,27,58]. Today, most high-performing global features are based on deep convolutional neural networks [5,4,60,1,17,47,48], which are trained with ranking-based [9,51,19] or classification losses [62,11]. Our work leverages recent learned lessons in global feature design, by adopting GeM pooling [47] and ArcFace loss [11]. This leads to improved global feature retrieval performance compared to previous techniques, which is further boosted by geometric re-ranking with local features obtained from the same model.

Joint local and global CNN features. Previous work considered neural networks for joint extraction of global and local features. For indoor localization, Taira et al. [55] used NetVLAD [1] to extract global features for candidate pose retrieval, followed by dense local feature matching using feature maps from the same network. Simeoni et al.'s DSM [53] detected keypoints in activation maps from global feature models using MSER [30]; activation channels are interpreted as visual words, in order to propose correspondences between a pair of images. Our work differs substantially from [55,53], since they only post-process pre-trained global feature models to produce local features, while we jointly train local and global. Sarlin et al. [50] distill pre-trained local [12] and global [1] features into a single model, targeting localization applications. In contrast, our model is trained end-to-end for image retrieval, and is not limited to mimicking separate pre-trained local and global models. To the best of our knowledge, ours is the first work to learn a non-distilled model producing both local and global features.

Dimensionality reduction for image retrieval. PCA and whitening are widely used for dimensionality reduction of local and global features in image retrieval [4,60,39,48]. As discussed in [23], whitening downweights co-occurences of local features, which is generally beneficial for retrieval applications. Mukundan et al. [35] further introduce a shrinkage parameter that controls the extent of applied whitening. If supervision in the form of matching pairs or category labels is available, more sophisticated methods [31,18] can be used. More recently, Gordo et al. [16] propose to replace PCA/whitening by a fully-connected layer, that is learned together with the global descriptor.

In this paper, our goal is to compose a system that can be learned end-to-end, using only image-level labels and without requiring post-processing stages that make training more complex. Also, since we extract local features from feature maps of common CNN backbones, they tend to be very high-dimensional and infeasible for large-scale problems. All above-mentioned approaches would either require a separate post-processing step to reduce the dimensionality of features, or supervision at the level of local patches – making them unsuitable to our needs. We thus introduce an autoencoder in our model, which can be jointly and efficiently learned with the rest of the network. It requires no extra supervision as it can be trained with a reconstruction loss.

3 DELG

3.1 Design considerations

For optimal performance, image retrieval requires semantic understanding of the types of objects that a user may be interested in, such that the system can distinguish between relevant objects versus clutter/background. Both local and global features should thus focus only on the most discriminative information within the image. However, there are substantial differences in terms of the desired behavior for these two feature modalities, posing a considerable challenge to jointly learn them.

Global features should be similar for images depicting the same object of interest, and dissimilar otherwise. This requires high-level, abstract representations that are invariant to viewpoint and photometric transformations. Local features, on the other hand, need to encode representations that are grounded to specific image regions; in particular, the keypoint detector should be equivariant with respect to viewpoint, and the keypoint descriptor needs to encode localized visual information. This is crucial to enable geometric consistency checks between query and database images, which are widely used in image retrieval systems.

Besides, our goal is to design a model that can be learned end-to-end, with local and global features, without requiring additional learning stages. This simplifies the training pipeline, allowing faster iterations and wider applicability. In comparison, it is common for previous feature learning work to require several learning stages: attentive deep local feature learning [39] requires 3 learning stages (fine-tuning, attention, PCA); deep global features usually require two stages, e.g., region proposal and Siamese training [17], or Siamese training and supervised whitening [47], or ranking loss training and PCA [48].

3.2 Model

We design our DELG model, illustrated in Fig. 1, to fulfill the requirements outlined above. We propose to leverage hierarchical representations from CNNs [66] in order to represent the different types of features to be learned. While global features can be associated with deep layers representing high-level cues, local features are more suitable to intermediate layers that encode localized information.

Given an image, we apply a convolutional neural network backbone to obtain two feature maps: $\mathcal{S} \in \mathcal{R}^{H_S \times W_S \times C_S}$ and $\mathcal{D} \in \mathcal{R}^{H_D \times W_D \times C_D}$, representing shallower and

deeper activations respectively, where H, W, C correspond to the height, width and number of channels in each case. For common convolutional networks, $H_D \leq H_S$, $W_D \leq W_S$ and $C_D \geq C_S$; deeper layers have spatially smaller maps, with a larger number of channels. Let $s_{h,w} \in \mathcal{R}^{C_S}$ and $d_{h,w} \in \mathcal{R}^{C_D}$ denote features at location h, w in these maps. For common network designs, these features are non-negative since they are obtained after the ReLU non-linearity, which is the case in our method.

In order to aggregate deep activations into a global feature, we adopt generalized mean pooling (GeM) [47], which effectively weights the contributions of each feature. Another key component of global feature learning is to whiten the aggregated representation; we integrate this into our model with a fully-connected layer $F \in \mathcal{R}^{C_F \times C_D}$, with learned bias $b_F \in \mathcal{R}^{C_F}$, similar to [17]. These two components produce a global feature $g \in \mathcal{R}^{C_F}$ that summarizes the discriminative contents of the whole image:

$$g = F \times \left(\frac{1}{H_D W_D} \sum_{h,w} d_{h,w}^p \right)^{1/p} + b_F \tag{1}$$

where p denotes the generalized mean power parameter, and the exponentiation $d_{h,w}^p$ is applied elementwise.

Regarding local features, it is important to select only the relevant regions for matching. This can be achieved by adopting an attention module M [39], whose goal is to predict which among the extracted local features are discriminative for the objects of interest. This is performed as $\mathcal{A} = M(\mathcal{S})$, where M is a small convolutional network and $\mathcal{A} \in \mathcal{R}^{H_S \times W_S}$ denotes the attention score map associated to the features from $\mathcal{S}$.

Furthermore, since hundreds to thousands of local features are commonly used, they must be represented compactly. To do so, we propose to integrate a small convolutional autoencoder (AE) module [21], which is responsible for learning a suitable low-dimensional representation. The local descriptors are obtained as $\mathcal{L} = T(\mathcal{S})$, where $\mathcal{L} \in \mathcal{R}^{H_S \times W_S \times C_T}$, and T is the encoding part of the autoencoder, corresponding to a 1×1 convolutional layer with C_T filters. Note that, contrary to $\mathcal{S}$, the local descriptors $\mathcal{L}$ are not restricted to be non-negative.

Each extracted local feature at position h, w is thus represented with a local descriptor $l_{h,w} \in \mathcal{L}$ and its corresponding keypoint detection score $a_{h,w} \in \mathcal{A}$. Their locations in the input image are set to corresponding receptive field centers, which can be computed using the parameters of the network [2].

The global and local descriptors are L_2-normalized into $\hat{g}$ and $\hat{l}_{h,w}$, respectively.

3.3 Training

We propose to train the model using only image-level labels, as illustrated in Fig. 2. In particular, note that we do not require patch-level supervision to train local features, unlike most recent works [14,49,36,29].

Besides the challenge to acquire the annotations, note that patch-level supervision could help selecting repeatable features, but not necessarily the discriminative ones; in contrast, our model discovers discriminative features by learning which can distinguish the different classes, given by image-level labels. In this weakly-supervised local feature

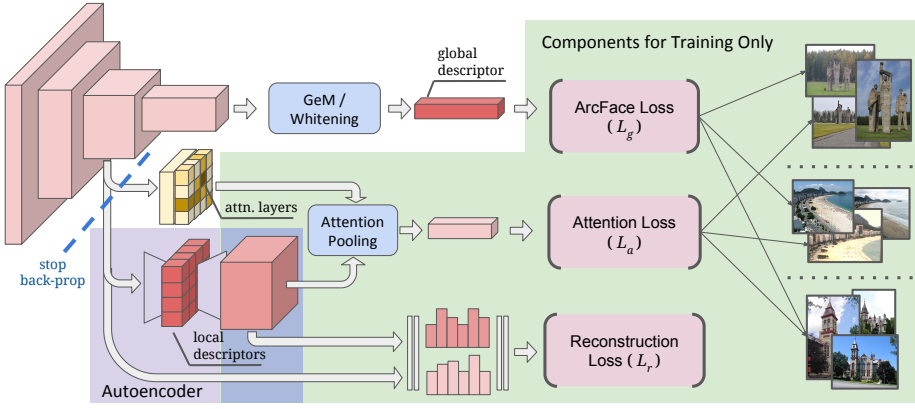


Fig. 2: Illustration of our **training pipeline**. The components highlighted in green are used solely during training. There are two classification losses: ArcFace for global feature learning (L_g), and softmax for attention learning (L_a). In both cases, the classification objective is to distinguish different landmarks (an instance-level recognition problem). The autoencoder (purple) is further trained with a reconstruction loss (L_r). The whole model is learned end-to-end, and benefits substantially from stopping gradient back-propagation from L_a and L_r into the CNN backbone.

setting, it is very important to control the gradient flow between the global and local feature learning, which is discussed in more detail below.

Global features. For global feature learning, we adopt a suitable loss function with L_2-normalized classifier weights $\hat{\mathcal{W}}$, followed by scaled softmax normalization and cross-entropy loss [61]; this is sometimes referred to as "cosine classifier". Additionally, we adopt the ArcFace margin [11], which has shown excellent results for global feature learning by inducing smaller intra-class variance. Concretely, given $\hat{g}$, we first compute the cosine similarity against $\hat{\mathcal{W}}$, adjusted by the ArcFace margin. The ArcFace-adjusted cosine similarity can be expressed as $\mathrm{AF}(u, c)$:

$$\mathrm{AF}(u, c) = \begin{cases} \cos(\mathrm{acos}(u) + m), & \text{if } c = 1 \\ u, & \text{if } c = 0 \end{cases} \tag{2}$$

where u is the cosine similarity, m is the ArcFace margin and c is a binary value indicating if this is the ground-truth class. The cross-entropy loss, computed using softmax normalization can be expressed in this case as:

$$L_g(\hat{g}, y) = -\log\left(\frac{\exp(\gamma \times \mathrm{AF}(\hat{w}_k^T \hat{g}, 1))}{\sum_n \exp(\gamma \times \mathrm{AF}(\hat{w}_n^T \hat{g}, y_n))}\right) \tag{3}$$

where γ is a learnable scalar, $\hat{w}_i$ refers to the L_2-normalized classifier weights for class i, y is the one-hot label vector and k is the index of the ground-truth class ($y_k = 1$).

Local features. To train the local features, we use two losses. First, a mean-squared error regression loss that measures how well the autoencoder can reconstruct $\mathcal{S}$. Denote

$\mathcal{S}' = T'(\mathcal{L})$ as the reconstructed version of $\mathcal{S}$, with same dimensions, where T' is a 1×1 convolutional layer with C_S filters, followed by ReLU. The loss can be expressed as:

$$L_r(\mathcal{S}', \mathcal{S}) = \frac{1}{H_S W_S C_S} \sum_{h,w} \|s'_{h,w} - s_{h,w}\|^2 \tag{4}$$

Second, a cross-entropy classification loss that incentivizes the attention module to select discriminative local features. This is done by first pooling the reconstructed features $\mathcal{S}'$ with attention weights $a_{h,w}$:

$$a' = \sum_{h,w} a_{h,w} s'_{h,w} \tag{5}$$

Then using a standard softmax-cross-entropy loss:

$$L_a(a', k) = -\log\left(\frac{\exp(v_k^T a' + b_k)}{\sum_n \exp(v_n^T a' + b_n)}\right) \tag{6}$$

where v_i, b_i refer to the classifier weights and biases for class i and k is the index of the ground-truth class; this tends to make the attention weights large for the discriminative features. The total loss is given by $L_g + \lambda L_r + \beta L_a$.

Controlling gradients. Naively optimizing the above-mentioned total loss experimentally leads to suboptimal results, because the reconstruction and attention loss terms significantly disturb the hierarchical feature representation which is usually obtained when training deep models. In particular, both tend to induce the shallower features $\mathcal{S}$ to be more semantic and less localizable, which end up being sparser. Sparser features can more easily optimize L_r, and more semantic features may help optimizing L_a; this, as a result, leads to underperforming local features.

We avoid this issue by stopping gradient back-propagation from L_r and L_a to the network backbone, i.e., to $\mathcal{S}$. This means that the network backbone is optimized solely based on L_g, and will tend to produce the desired hierarchical feature representation. This is further discussed in the experimental section that follows.

4 Experiments

4.1 Experimental setup

Model backbone and implementation. Our model is implemented using TensorFlow, leveraging the Slim model library [52].We use ResNet-50 (R50) and ResNet-101 (R101) [20]; R50 is used for ablation experiments. We obtain the shallower feature map $\mathcal{S}$ from the *conv4* output, and the deeper feature map $\mathcal{D}$ from the *conv5* output. Note that the Slim implementation moves the *conv5* stride into the last unit from *conv4*, which we also adopt – helping reduce the spatial resolution of $\mathcal{S}$. The number of channels in $\mathcal{D}$ is $C_D = 2048$; GeM pooling [47] is applied with parameter $p = 3$, which is not learned. The whitening fully-connected layer, applied after pooling, produces a global feature with dimensionality $C_F = 2048$. The number of channels in $\mathcal{S}$ is $C_S = 1024$; the autoencoder module learns a reduced dimensionality for this feature map with

$C_T = 128$. The attention network M follows the setup from [39], with 2 convolutional layers, without stride, using kernel sizes of 1; as activation functions, the first layer uses ReLU and the second uses Softplus [13].

Training details. We use the training set of the Google Landmarks dataset (GLD) [39], containing 1.2M images from 15k landmarks, and divide it into two subsets 'train'/'val' with 80%/20% split. The 'train' split is used for the actual learning, and the 'val' split is used for validating the learned classifier as training progresses. Models are initialized from pre-trained ImageNet weights. The images first undergo augmentation, by randomly cropping / distorting the aspect ratio; then, they are resized to 512×512 resolution. We use a batch size of 16, and train using 21 Tesla P100 GPUs asynchronously, for 1.5M steps (corresponding to approximately 25 epochs of the 'train' split). The model is optimized using SGD with momentum of 0.9, and a linearly decaying learning rate that reaches zero once the desired number of steps is reached. We experiment with initial learning rates within $[3 \times 10^{-4}, 10^{-2}]$ and report results for the best performing one. We set the ArcFace margin $m = 0.1$, the weight for L_a to $\beta = 1$, and the weight for L_r to $\lambda = 10$. The learnable scalar for the global loss L_g is initialized to $\gamma = \sqrt{C_F} = 45.25$. See also the appendices for results obtained when training models on GLDv2 [63].

Evaluation datasets. To evaluate our model, we use several datasets. First, Oxford [44] and Paris [45], with revisited annotations [46], referred to as $\mathcal{R}$Oxf and $\mathcal{R}$Par, respectively. There are 4993 (6322) database images in the $\mathcal{R}$Oxf ($\mathcal{R}$Par) dataset, and a different query set for each, both with 70 images. Performance is measured using mean average precision (mAP). Large-scale results are further reported with the $\mathcal{R}$1M distractor set [46], which contains 1M images. As in previous papers [48,56,37], parameters are tuned in $\mathcal{R}$Oxf/$\mathcal{R}$Par, then kept fixed for the large-scale experiments. Second, we report large-scale instance-level retrieval and recognition results on the Google Landmarks dataset v2 (GLDv2) [63], using the latest ground-truth version (2.1). GLDv2-retrieval has 1129 queries (379 validation and 750 testing) and 762k database images; performance is measured using mAP@100. GLDv2-recognition has 118k test (41k validation and 77k testing) and 4M training images from 203k landmarks; the training images are only used to retrieve images and their scores/labels are used to form the class prediction; performance is measured using μAP@1. We perform minimal parameter tuning based on the validation split, and report results on the testing split.

Feature extraction and matching. We follow the convention from previous work [47,17,39] and use an image pyramid at inference time to produce multi-scale representations. For global features, we use 3 scales, $\{\frac{1}{\sqrt{2}}, 1, \sqrt{2}\}$; L_2 normalization is applied for each scale independently, then the three global features are average-pooled, followed by another L_2 normalization step. For local features, we experiment with the same 3 scales, but also with the more expensive setting from [39] using 7 image scales in total, with range from 0.25 to 2.0 (this latter setting is used unless otherwise noted). Local features are selected based on their attention scores $\mathcal{A}$; a maximum of 1k local features are allowed, with a minimum attention score τ, where we set τ to the median attention score in the last iteration of training, unless otherwise noted. For local feature matching, we use RANSAC [15] with an affine model. When re-ranking global feature retrieval results with local feature-based matching, the top 100 ranked images from the first stage are considered. For retrieval datasets, the final ranking is based on the number of inliers,

DR Method	λ	Jointly trained	Stop gradients	GLD-pairs AP (%)
PCA [39]	-	✗	-	51.48
FC	-			52.67
AE [ours]	0	✗	-	49.95
	1			51.28
	5			52.26
	10			54.21
	20			53.51
	10	✓	✗	37.05
	10		✓	53.73

Table 1: **Local feature ablation.** Comparison of local features, trained separately or jointly, with different methods for dimensionality reduction (DR). We report average precision (AP) results of matching image pairs from the Google Landmarks dataset (GLD).

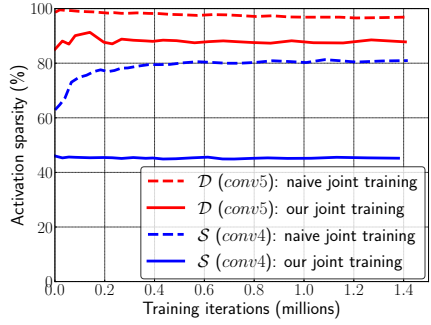


Fig. 3: Evolution of **activation sparsity** over training iterations for $\mathcal{D}$ (*conv5*) and $\mathcal{S}$ (*conv4*), comparing the naive joint training method and our improved version that controls gradient propagation. The naive method leads to much sparser feature maps.

then breaking ties using the global feature distance. For the recognition dataset, we follow a similar protocol as [63] to produce class predictions by aggregating scores of top retrieved images, where the scores of top images are given by $\frac{min(i,70)}{70} + \alpha c$ (here, i is the number of inliers, c the global descriptor cosine similarity and $\alpha = 0.25$). Our focus is on improving image features for retrieval/recognition, so we do not consider techniques that post-process results such as query expansion [10,47] or diffusion/graph traversal [22,8]. These are expensive due to requiring additional passes over the database, but if desired could be integrated to our system and produce stronger performance.

4.2 Results

First, we present ablation experiments, to compare features produced by our joint model against their counterparts which are separately trained, and also to discuss the effect of controlling the gradient propagation. For a fair comparison, our jointly trained features are evaluated against equivalent separately-trained models, with the same hyperparameters as much as possible. Then, we compare our models against state-of-the-art techniques. See also the appendices for more details, visualizations and discussions.

Local features. As an ablation, we evaluate our local features by matching image pairs. We select 200k pairs, each composed of a test and a train image from GLD, where in 1k pairs both images depict the same landmark, and in 199k pairs the two images depict different landmarks. We compute average precision (AP) after ranking the pairs based on the number of inliers. All variants for this experiment use τ equals to the 75^{th} percentile attention score in the last iteration of training. Results are presented in Tab. 1.

First, we train solely the attention and dimensionality reduction modules, for 500k iterations, all methods initialized with the same weights from a separately-trained global feature model. These results are marked as not being jointly trained. It can be seen that

Pooling	Loss	Jointly trained	Stop gradients	Medium $\mathcal{R}$Oxf	Medium $\mathcal{R}$Par	Hard $\mathcal{R}$Oxf	Hard $\mathcal{R}$Par
SPoC	Softmax	✗	–	51.2	72.0	26.3	47.7
SPoC	ArcFace	✗	–	59.8	80.8	35.6	61.7
GeM	ArcFace	✗	–	69.3	**82.2**	44.4	**64.0**
GeM	ArcFace	✓	✗	68.8	78.9	42.4	58.3
GeM	ArcFace	✓	✓	**69.7**	81.6	**45.1**	63.4

Table 2: **Global feature ablation.** Comparison of global features, trained separately or jointly, with different pooling methods (SPoC, GeM) and loss functions (Softmax, ArcFace). We report mean average precision (mAP %) on the $\mathcal{R}$Oxf and $\mathcal{R}$Par datasets.

our AE outperforms PCA and a simpler method using only a single fully-connected (FC) layer. Performance improves for the AE as λ increases from 0 to 10, decreasing with 20. Then, we jointly train the unified model; in this case, the variant that does not stop gradients to the backbone suffers a large drop in performance, while the variant that stops gradients obtains similar results as in the separately-trained case.

The poor performance of the naive jointly trained model is due to the degradation of the hierarchical feature representation. This can be assessed by observing the evolution of activation sparsity in $\mathcal{S}$ (*conv4*) and $\mathcal{D}$ (*conv5*), as shown in Fig. 3. Generally, layers representing more abstract and high-level semantic properties (usually deeper layers) have high levels of sparsity, while shallower layers representing low-level and more localizable patterns are dense. As a reference, the ImageNet pre-trained model presents on average 45% and 82% sparsity for these two feature maps, respectively, when run over GLD images. For the naive joint training case, the activations of both layers quickly become much sparser, reaching 80% and 97% at the end of training; in comparison, our proposed training scheme preserves similar sparsity as the ImageNet model: 45% and 88%. This suggests that the *conv4* features in the naive case degrade for the purposes of local feature matching; controlling the gradient effectively resolves this issue.

Global features. Tab. 2 compares global feature training methods. The first three rows present global features trained with different loss and pooling techniques. We experiment with standard Softmax Cross-Entropy and ArcFace [11] losses; for pooling, we consider standard average pooling (equivalent to SPoC [4]) and GeM [47]. ArcFace brings an improvement of up to 14%, and GeM of up to 9.5%. GeM pooling and ArcFace loss are adopted in our final model. Naively training a joint model, without controlling gradients, underperforms when compared to the baseline separately-trained global feature, with mAP decrease of up to 5.7%. Once gradient stopping is employed, the performance can be recovered to be on par with the separately-trained version (a little better on $\mathcal{R}$Oxf, a little worse on $\mathcal{R}$Par). This is expected, since the global feature in this case is optimized by itself, without influence from the local feature head.

Comparison to retrieval state-of-the-art. Tab. 3 compares our model against the retrieval state-of-the-art. Three settings are presented: (A) local feature aggregation and re-ranking (previous work); (B) global feature similarity search; (C) global feature search followed by re-ranking with local feature matching and spatial verification (SP).

In setting (B), the DELG global feature variants strongly outperform previous work for all cases (most noticeably in the large-scale setting): 7.1% absolute improvement in $\mathcal{R}$Oxf+1M-Hard and 7.5% in $\mathcal{R}$Par+1M-Hard. Note that we obtain strong improvements

Method	Medium				Hard			
	$\mathcal{R}$Oxf	+1M	$\mathcal{R}$Par	+1M	$\mathcal{R}$Oxf	+1M	$\mathcal{R}$Par	+1M
(A) Local feature aggregation + re-ranking								
HesAff-rSIFT-ASMK⋆+SP [58]	60.6	46.8	61.4	42.3	36.7	26.9	35.0	16.8
HesAff-HardNet-ASMK⋆+SP [34]	65.6	–	65.2	–	41.1	–	38.5	–
DELF-ASMK⋆+SP [39,46]	67.8	53.8	76.9	57.3	43.1	31.2	55.4	26.4
DELF-R-ASMK⋆+SP (GLD) [56]	**76.0**	**64.0**	**80.2**	**59.7**	**52.4**	**38.1**	**58.6**	**29.4**
(B) Global features								
AlexNet-GeM [47]	43.3	24.2	58.0	29.9	17.1	9.4	29.7	8.4
VGG16-GeM [47]	61.9	42.6	69.3	45.4	33.7	19.0	44.3	19.1
R101-R-MAC [17]	60.9	39.3	78.9	54.8	32.4	12.5	59.4	28.0
R101-GeM [47]	64.7	45.2	77.2	52.3	38.5	19.9	56.3	24.7
R101-GeM↑ [53]	65.3	46.1	77.3	52.6	39.6	22.2	56.6	24.8
R101-GeM-AP [48]	67.5	47.5	80.1	52.5	42.8	23.2	60.5	25.1
R101-GeM-AP (GLD) [48]	66.3	–	80.2	–	42.5	–	60.8	–
R152-GeM (GLD) [47]	68.7	–	79.7	–	44.2	–	60.3	–
R50-DELG **[ours]**	69.7	**55.0**	81.6	59.7	45.1	27.8	63.4	34.1
R101-DELG **[ours]**	**73.2**	54.8	**82.4**	**61.8**	**51.2**	**30.3**	**64.7**	**35.5**
(C) Global features + Local feature re-ranking								
R101-GeM↑+DSM [53]	65.3	47.6	77.4	52.8	39.2	23.2	56.2	25.0
R50-DELG **[ours]**	75.1	61.1	82.3	60.5	54.2	36.8	64.9	34.8
R50-DELG⋆ **[ours]**	–	60.4	–	60.3	–	35.3	–	34.1
R101-DELG (3 scales global & local) **[ours]**	77.5	61.7	**83.0**	62.6	56.7	38.0	65.2	36.2
R101-DELG⋆ (3 scales global & local) **[ours]**	–	61.6	–	62.3	–	36.9	–	35.5
R101-DELG **[ours]**	**78.5**	**62.7**	82.9	**62.6**	**59.3**	**39.3**	**65.5**	**37.0**
R101-DELG⋆ **[ours]**	–	62.5	–	62.5	–	38.5	–	36.3

Table 3: **Comparison to retrieval state-of-the-art.** Results (% mAP) on the $\mathcal{R}$Oxf/$\mathcal{R}$Par datasets (and their large-scale versions $\mathcal{R}$Oxf+1M/$\mathcal{R}$Par+1M), with both Medium and Hard evaluation protocols. The top set of rows (A) presents previous work's results using local feature aggregation and re-ranking. Other sets of rows present results using (B) global features only, or (C) global features for initial search then re-ranking using local features. DELG⋆ refers to a version of DELG where the local features are binarized. DELG and DELG⋆ outperform previous work in setups (B) and (C) substantially. DELG also outperforms methods from setting (A) in 7 out of 8 cases.

even when using the ResNet-50 backbone, while the previous state-of-the-art used ResNet-101/152, which are much more complex (2X/3X the number of floating point operations, respectively). To ensure a fair comparison, we present results from [47,48] which specifically use the same training set as ours, marked as "(GLD)" – the results are obtained from the authors' official codebases. In particular, note that "R152-GeM (GLD) [47]" uses not only the same training set, but also the same exact scales in the image pyramid; even if our method is much cheaper, it consistently outperforms others.

For setup (C), we use both global and local features. For large-scale databases, it may be impractical to store all raw local features in memory; to alleviate such requirement, we also present a variant, DELG⋆, where we store local features in binarized format, by simply applying an elementwise function: $b(x) = +1$ if $x > 0, -1$ otherwise.

Local feature re-ranking boosts performance substantially for DELG, compared to only searching with global features, especially in large-scale cases: gains of up to 9% (in $\mathcal{R}$Oxf+1M-Hard). We also present results where local feature extraction is performed with 3 scales only, the same ones used for global features. The large-scale results are similar, providing a boost of up to 7.7%. Results for DELG⋆ also provide large improvements, but with performance that is slightly lower than the corresponding

Method	Retrieval mAP (%)	Recognition μAP (%)
DELF-R-ASMK*+SP [56]	18.8	–
R101-GeM+ArcFace [63]	20.7	33.3
R101-GeM+CosFace [65]	21.4	–
DELF-KD-tree [39]	–	44.8
R50-DELG (global-only) [ours]	20.4	32.4
R101-DELG (global-only) [ours]	21.7	32.0
R50-DELG [ours]	22.3	59.2
R101-DELG [ours]	**24.5**	**61.2**

Table 4: **GLDv2 evaluation.** Results on the GLDv2 dataset, for the retrieval and recognition tasks, on the "testing" split of the query set. For a fair comparison, all methods are trained on GLD.

Method	Hard	Medium
R50-DELG (global-only)	45.1	69.7
Local feature re-ranking		
SIFT [28]	44.4	69.8
SOSNet [57]	45.5	69.9
D2-Net [14]	47.2	70.4
R50-DELG [ours]	**54.2**	**75.1**

Table 5: **Re-ranking experiment.** Comparison of DELG against other recent local features; results (% mAP) on the $\mathcal{R}$Oxf dataset.

unbinarized versions. Our retrieval results also outperform DSM [53] significantly, by more than 10% in several cases. Different from our proposed technique, the gain from spatial verification reported in their work is small, of at most 1.5% absolute. DELG also outperforms local feature aggregation results from setup (A) in 7 out of 8 cases, establishing a new state-of-the-art across the board.

GLDv2 evaluation. Tab. 4 compares DELG against previous GLDv2 results, where for a fair comparison we report methods trained on GLD. DELG achieves top performance in both retrieval and recognition tasks, with local feature re-ranking providing a significant boost in both cases – especially on the recognition task (29.2% absolute improvement). Note that recent work has reported even higher performance on the retrieval task, by learning on GLDv2's training set and using query expansion techniques [65] / ensembling [63]. On the other hand, DELG's performance on the recognition task is so far the best reported single-model result, outperforming many ensemble-based methods (by itself, it would have been ranked top-3 in the 2019 challenge) [63].

Re-ranking experiment. Tab. 5 further compares local features for re-ranking purposes. R50-DELG is compared against SIFT [28], SOSNet [57] (HPatches model, DoG keypoints) and D2-Net [14] (trained, multiscale). All methods are given the same retrieval short list of 100 images for re-ranking (based on R50-DELG-global retrieval); for a fair comparison, all methods use 1k features and 1k RANSAC iterations. We tuned matching parameters separately for each method: whether to use ratio test or distance threshold for selecting correspondences (and their associated thresholds); RANSAC residual threshold; minimum number of inliers (below which we declare no match). SIFT and SOSNet provide little improvement over the global feature, due to suboptimal feature detection based on our observation (i.e., any blob-like feature is detected, which may not correspond to landmarks). D2-Net improves over the global feature, benefiting from a better feature detector. DELG outperforms other methods by a large margin.

Latency and memory. Tab. 6 reports feature extraction latency and index memory footprint for state-of-the-art methods, corresponding to the three settings from Tab. 3; as a reference, we also present numbers for DELF (which uses an R50 backbone). Joint extraction with DELG allows for substantial speed-up, compared to running two separate local and global models: when using 3 local feature scales, separately running R50-GeM

Method	Extraction latency (ms)	Memory (GB) $\mathcal{R}$Oxf+1M	Memory (GB) $\mathcal{R}$Par+1M
(A) Local feature aggregation			
DELF-R-ASMK⋆ [56]	2260	27.6	–
(B) Global features			
R50-GeM [47]	100	7.7	7.7
R101-GeM [47]	175	7.7	7.7
(C) Unified global + local features			
R50-DELG (3 scales global & local) [ours]	118	439.4	440.0
R50-DELG [ours]	211	485.5	486.2
R101-DELG (3 scales global & local) [ours]	193	437.1	437.8
R101-DELG⋆ (3 scales global & local) [ours]	193	21.1	21.1
R101-DELG [ours]	383	485.9	486.6
R101-DELG⋆ [ours]	383	22.6	22.7
Local features			
DELF (3 scales) [39]	98	434.2	434.8
DELF (7 scales) [39]	201	477.9	478.5

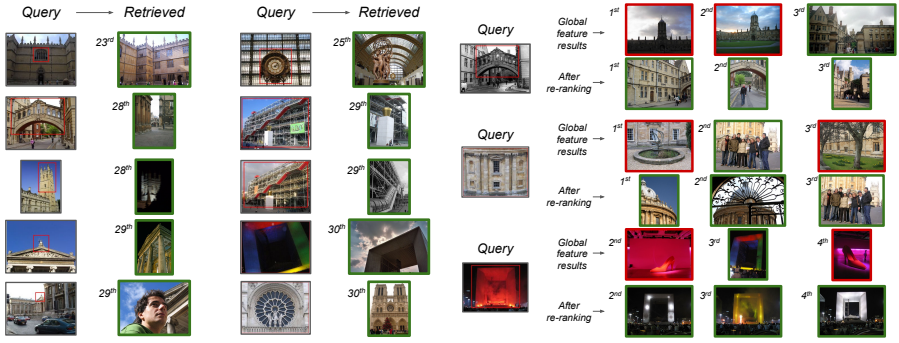
Table 6: Feature extraction **latency** and database **memory** requirements for different image retrieval models. Latency is measured on an NVIDIA Tesla P100 GPU, for square images of side 1024. (A) DELF-R-ASMK⋆ measurements use the code and default configuration from [56]; (B) ResNet-GeM variants use 3 image scales; (C) DELG and DELG⋆ are compared with different configurations. As a reference, we also provide numbers for DELF in the last rows.

and DELF would lead to 198ms, while the unified model runs with latency of 118ms (40% faster). For the R50 case with 7 local scales, the unified model is 30% faster. The binarization technique adds negligible overhead, having roughly the same latency.

Storing unquantized DELG local features requires excessive index memory requirements; using binarization, this can be reduced significantly: R101-DELG⋆ requires 23GB. This is lower than the memory footprint of DELF-R-ASMK⋆. Note also that feature extraction for DELG⋆ is much faster than for DELF-R-ASMK⋆, by more than $5\times$. R50-DELG with 3 scales is also faster than using a heavier global feature (R101-GeM [47]), besides being more accurate. As a matter of fact, several of the recently-proposed global features [47,17] use image pyramids with 3 scales; our results indicate that their performance can be improved substantially by adding a local feature head, with small increase in extraction latency, and without degrading the global feature.

Comparison against concurrent work. Our global feature outperforms the SOLAR [37] global feature method for all datasets and evaluation protocols, with gains of up to 3.3%. Compared to Tolias et al. [59], we obtain similar results on $\mathcal{R}$Oxf/$\mathcal{R}$Par; for the large-scale setting, their best average mAP across the two datasets is 50.1%, while R101-DELG obtains 50.4% and R101-DELG⋆ obtains 50.0% (in general, our method is a little better on $\mathcal{R}$Par and theirs on $\mathcal{R}$Oxf). On GLDv2-recognition, Tolias et al. [59] achieve 36.5% μAP, while R101-DELG obtains 61.2%.

Qualitative results. We give examples of retrieval results, to showcase the DELG model. Fig. 4a illustrates difficult cases, where the database image shows a very different viewpoint, or significant lighting changes; these images can still achieve relatively high ranks due to effective global features, which capture well the similarity even in such challenging scenarios. In these cases, local features do not produce sufficient matches.



(a) **Global feature retrieval: high recall.** (b) **Local feature re-ranking: high precision.**

Fig. 4: Sample DELG results on $\mathcal{R}$Oxf-Hard and $\mathcal{R}$Par-Hard. **(a)** Examples of difficult, high-ranked relevant retrieved images for 10 different queries. These retrieved database images have a low number of inliers after geometric verification (if any), which means that their similarity is mainly captured by the global feature. **(b)** Examples illustrating performance improvements from the re-ranking stage using local features. For each query (left), two rows are presented on the right, the top one showing results based on global feature similarity and the bottom one showing results after re-ranking the top 100 images with local features. Correct results are marked with green borders, and incorrect ones in red. While top retrieved global feature results are often ranked incorrectly, local feature matching can effectively re-rank them to improve precision.

Fig. 4b shows the effect of local feature re-ranking for selected queries, for which substantial gains are obtained. Global features tend to retrieve images that have generally similar appearance, but which sometimes do not depict the same object of interest; this can be improved substantially with local feature re-ranking, which enables stricter matching selectivity. As can be observed in these two figures, global features are crucial for high recall, while local features are key to high precision.

5 Conclusions

Our main contribution is a unified model that enables joint extraction of local and global image features, referred to as DELG. The model is based on a ResNet backbone, leveraging generalized mean pooling to produce global features and attention-based keypoint detection to produce local features. We also introduce an effective dimensionality reduction technique that can be integrated into the same model, based on an autoencoder. The entire network can be trained end-to-end using image-level labels and does not require any additional post-processing steps. For best performance, we show that it is crucial to stop gradients from the attention and autoencoder branches into the network backbone, otherwise a suboptimal representation is obtained. We demonstrate the effectiveness of our method with comprehensive experiments, achieving state-of-the-art performance on the Revisited Oxford, Revisited Paris and Google Landmarks v2 datasets.

Appendix A. Training cost

One of the advantages of DELG is that local and global features can be jointly trained in one shot, without the need of additional steps. In practice, we have observed that the training time for DELG is roughly the same as for the associated global feature, both taking approximately 1.5 day.

This is because the additional cost of learning the attention and autoencoder layers is small: there are only 5 extra trainable layers (2 in the attention module, 2 in the autoencoder, plus the attention loss classifier), and their gradients are not backpropagated to the network backbone. Other factors play a much more significant role in the training speed, e.g.: reading data from disk, transferring batch to GPU memory, applying image pre-processing operations such as resizing/cropping/augmentation, etc.

In short, training our local image features comes at a very small cost on top of global feature learning. Let us clarify, though, that while the training cost is roughly the same between the global and joint models, the inference cost for the joint model has an overhead for local feature detection (e.g., selecting the top local features across different scales in the image pyramid).

We can also compare DELG's training cost to DELF's: DELF would require one additional run for attention learning (6 hours), followed by a PCA computation step (3 hours). The advantage of DELG, compared to DELF, is that the attention and autoencoder layers are already adapting to the network backbone while it is training; for DELF, this process only happens after the backbone is fully trained.

Appendix B. Model selection and tuning

We provide more details on how our models were selected/tuned, and specify chosen parameters which were not mentioned in the main text. For more information, please refer to our released code/models.

Model selection. Our models are trained for 1.5M steps, corresponding to approximately 25 epochs of an 80% split of the GLD training set. We attempt three different initial learning rates for each configuration, and select the best one based on the performance on $\mathcal{R}$Oxf/$\mathcal{R}$Par (it is a convention in recent work [48,56,37] to perform ablations/tuning on these datasets). In all cases, we pick the model at the end of training and do not hand-pick earlier checkpoints which may have higher performance. These selected models are then used in all large-scale experiments, on $\mathcal{R}$Oxf+1M, $\mathcal{R}$Par+1M, GLDv2-retrieval and GLDv2-recognition.

Tuning image matching. For local feature-based matching with DELG, we experimented with two methods for proposing putative correspondences (before feeding them to RANSAC): ratio test [28] and distance criteria. RANSAC is used with 1k iterations, and any returned match is used for re-ranking (minimum number of inliers is zero – better results may be obtained by tuning the minimum number of inliers, which we did not do). We tuned the local descriptor matching threshold (either ratio or distance threshold) and the RANSAC residual threshold; for the recognition task, we further tuned α (a weight used to combine the local and global scores – see Sec. 4.1). Tuning is performed on $\mathcal{R}$Oxf/$\mathcal{R}$Par, and then the best configuration is fixed for experiments on $\mathcal{R}$Oxf+1M and $\mathcal{R}$Par+1M; similarly, we tune methods on the validation set

Method	GLD version	Medium				Hard			
		$\mathcal{R}$Oxf	+1M	$\mathcal{R}$Par	+1M	$\mathcal{R}$Oxf	+1M	$\mathcal{R}$Par	+1M
Global features									
R50-DELG	v1	69.7	55.0	81.6	59.7	45.1	27.8	63.4	34.1
R50-DELG	v2-clean	73.6	60.6	85.7	68.6	51.0	32.7	71.5	44..4
R101-DELG	v1	73.2	54.8	82.4	61.8	51.2	30.3	64.7	35.5
R101-DELG	v2-clean	**76.3**	**63.7**	**86.6**	**70.6**	**55.6**	**37.5**	**72.4**	**46.9**
Global features + Local feature re-ranking									
R50-DELG	v1	75.4	61.1	82.3	60.5	54.2	36.8	64.9	34.8
R50-DELG	v2-clean	78.3	67.2	85.7	69.6	57.9	43.6	71.0	45.7
R101-DELG (3 scales global & local)	v1	77.5	61.7	83.0	62.6	56.7	38.0	65.2	36.2
R101-DELG (3 scales global & local)	v2-clean	80.3	68.6	87.2	71.4	61.4	45.3	72.3	47.7
R101-DELG* (3 scales global & local)	v1	–	61.6	–	62.3	–	36.9	–	35.5
R101-DELG* (3 scales global & local)	v2-clean	–	67.4	–	71.0	–	42.9	–	46.5
R101-DELG	v1	78.5	62.7	82.9	62.6	59.3	39.3	65.5	37.0
R101-DELG	v2-clean	**81.2**	**69.1**	**87.2**	**71.5**	**64.0**	**47.5**	**72.8**	**48.7**
R101-DELG*	v1	–	62.5	–	62.5	–	38.5	–	36.3
R101-DELG*	v2-clean	–	68.6	–	71.1	–	45.1	–	47.3

Table 7: **Training dataset comparison for Revisited Oxford/Paris evaluations.** Results (% mAP) on the $\mathcal{R}$Oxf/$\mathcal{R}$Par datasets (and their large-scale versions $\mathcal{R}$Oxf+1M/$\mathcal{R}$Par+1M), with both Medium and Hard evaluation protocols, comparing models trained on GLD (v1) and GLDv2-clean. Training on GLDv2-clean provides a substantial performance boost in all cases.

of GLDv2-retrieval/GLDv2-recognition, then fix them for experiments on the testing set of GLDv2-retrieval/GLDv2-recognition. For DELG on $\mathcal{R}$Oxf/$\mathcal{R}$Par, we use a ratio test with threshold 0.95; for DELG*, we use distance-based matching with threshold 1.1. For DELG on GLDv2-retrieval/GLDv2-recognition, we also use distance-based matching, but in this case with a tighter threshold of 0.9. For all retrieval datasets, we set the RANSAC residual threshold to 20.0; on the recognition dataset, it is set to 10.0 (since the recognition task has a stronger focus on precision, we find that tighter matching parameters perform better). For the recognition dataset, α was tuned to 0.25.

Appendix C. Results with models trained on GLDv2

All results presented in the main part of the paper use models that were trained on the first version (v1) of the Google Landmarks Dataset, referred to as GLD [39]. In this appendix, we provide experimental results based on models that were trained on GLDv2 [63]. Besides being a larger and more comprehensive dataset, GLDv2 has the advantage of stability – all images have permissive licenses which allow indefinite retention; in contrast, GLD shrinks over time as images get deleted by users who uploaded them, due to copyright restrictions. We believe that, moving forward, it may be more suitable to rely on stable datasets such as GLDv2 for proper experimental comparisons, since in this case one can guarantee that identical datasets are used for training and evaluation.

For our experiments, we used a subset of the entire training set, referred to as "GLDv2-train-clean" (GLDv2-clean for short), containing $1.6M$ images of $81k$ landmarks. We divide it into two subsets 'train'/'val' with $80\%/20\%$ split. We train for 2M steps (corresponding to approximately 25 epochs of the 'train' split). All other training hyperparameters are kept the same as described in Sec. 4.1. When evaluating on

Method	GLD-train version	Retrieval mAP (%)	Recognition μAP (%)
R50-DELG (global-only)	v1	20.4	32.4
R50-DELG (global-only)	v2-clean	24.1	27.9
R101-DELG (global-only)	v1	21.7	32.0
R101-DELG (global-only)	v2-clean	26.0	28.9
R50-DELG	v1	22.3	59.2
R50-DELG	v2-clean	24.3	55.2
R101-DELG	v1	24.3	**61.2**
R101-DELG	v2-clean	**26.8**	56.4

Table 8: **Training dataset comparison for GLDv2 evaluations.** Results on the GLDv2 dataset, for the retrieval and recognition tasks, on the "testing" split of the query set, comparing models trained on GLD (v1) and GLDv2-clean. Training on GLDv2-clean provides a performance boost for the retrieval task, but a small degradation for the recognition task.

retrieval/recognition datasets, we did not tune any evaluation-related parameters and simply used the configurations that worked best for the GLD-trained models.

Image retrieval results on the $\mathcal{R}$Oxf/$\mathcal{R}$Par datasets are presented on Tab. 7, showing consistent improvement in all cases. First, consider results based solely on global features (top rows). There are striking mAP improvements, especially for the large-scale evaluations: for example, 11.4% absolute improvement on $\mathcal{R}$Par+1M-Hard and 8.9% on $\mathcal{R}$Oxf+1M-Medium, both results with the R101 backbone. Second, consider results based on global feature search followed by local feature re-ranking (bottom rows). Again, we see substantial improvement: for example, R101-DELG absolute improvements of 11.7% on $\mathcal{R}$Par+1M-Hard and 8.2% on $\mathcal{R}$Oxf+1M-Hard. In most large-scale cases, the improvement due to local feature re-ranking increases when using GLDv2-clean (with the exception of $\mathcal{R}$Oxf+1M-Medium), which may be due to better local features and/or a better short list of images selected for re-ranking.

Instance-level retrieval/recognition results on GLDv2 are presented on Tab. 8. Global-only retrieval results are improved significantly, with 4.3% absolute gain for R101-DELG; when re-ranking retrieval results with local feature-based geometric verification, the gains are smaller but still substantial, of 2.5%. For the recognition task, we surprisingly observe a decrease in performance when training the model on GLDv2-clean: decrease of 3.1% on the global-only setup, and 4.8% when re-ranking with local features, both numbers again for the R101-based model. Note that a similar observation was reported in [63].

Appendix D. Memory footprint

For reporting the memory footprint of DELG/DELG*, we follow a similar convention to previous work [46,56] and report total storage required for local and global descriptors. Note that local feature geometry information is not counted in the reported numbers, and would add some overhead.

Our main focus in this paper is to propose a new model for unified local and global feature extraction, so we did not thoroughly explore techniques for efficient quantization. As the DELG* results show, there is great promise in aggressively quantizing local

descriptors, leading to reasonable storage requirements – which can likely be improved with more effective quantizers. Similarly, global descriptors and local feature geometry could be quantized to improve the total memory cost substantially.

Appendix E. Local feature matching visualizations

We present more qualitative results, to illustrate local feature matching with DELG: Fig. 5 presents examples of correct matches, and Fig. 6 presents examples of incorrect matches. For these examples, we use the R50-DELG model and the $\mathcal{R}$Oxf/$\mathcal{R}$Par datasets. These visualizations depict the final obtained correspondences, post-RANSAC.

For each row, one query and two index images are shown, and on the right their local feature matches are shown, with lines connecting the corresponding keypoints. Fig. 5 showcases DELG's robustness against strong viewpoint and illumination changes: for example, matches can be obtained across different scales and day-vs-night cases. Fig. 6 presents overtriggering cases, where a match is found even though different objects/scenes are presented: these tend to occur for similar patterns between query and index images (ie, similar windows, arches or roofs) which appear in similar geometric configurations. Generally, these do not affect retrieval results much because the number of inliers is low.

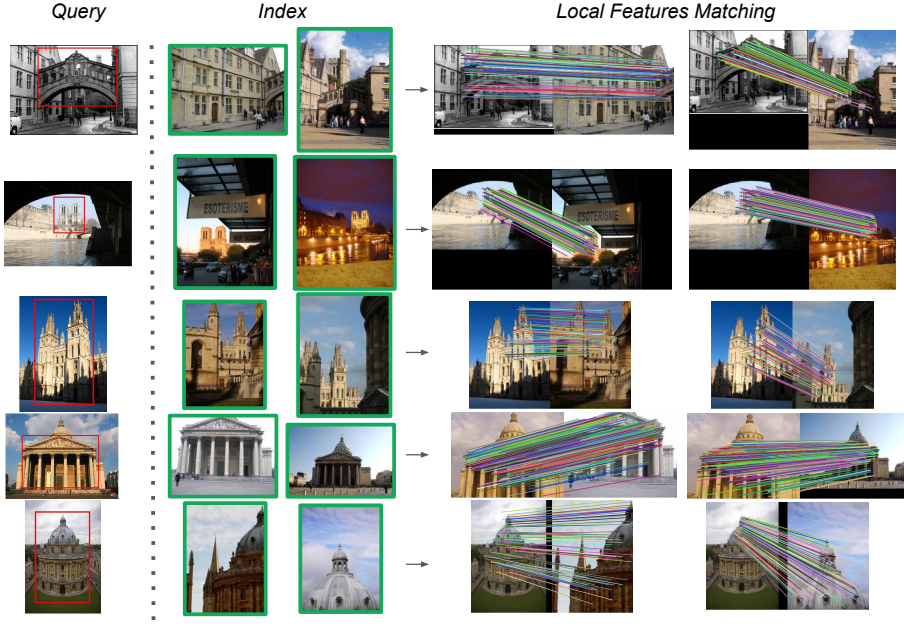


Fig. 5: Examples of correct local feature matches, for image pairs depicting the same object/scene.

Fig. 6: Examples of incorrect local feature matches, for image pairs depicting different objects/scenes.

Appendix F. Feature visualization

We provide visualizations of the features learned by the DELG model. This is useful to understand the hierarchical representation which we rely on for extraction of local and global features. We explore two types of visualizations, based on dataset examples with high activations and by optimizing input images with respect to a given layer/channel.

F.1 Feature visualization by dataset examples

For this experiment, we run our R50-DELG model over 200k images from the Google Landmarks dataset [39], and collect the images and feature positions with largest value in each channel of several activation maps. We specifically consider the activation maps at the outputs of the *conv2*, *conv3*, *conv4* and *conv5* blocks of layers in our model's ResNet architecture [20]. The feature positions with maximum activations can be mapped back to the relevant input image regions by computing the model's receptive field parameters [2]. Note that the region may partly fall outside the image, in which case we apply zero-padding for the visualization.

Fig. 7 presents image patches that produce the highest activations for selected channels of the above-mentioned layers. The activation values are noted in each subfigure's title and can be read by zooming in. For each selected channel of a specific layer, the 9 patches with largest activations are shown. The receptive field sizes (both horizontally and vertically) for each of these layers are: 35 (*conv2*), 99 (*conv3*), 291 (*conv4*) and 483 (*conv5*); these correspond to the sizes of image patches. One can notice that the types of patterns which maximally activate specific layers grow in complexity with network depth. This agrees with observations from previous work, where the hierarchical nature of CNN features is discussed [66,41]. Shallower layers such as *conv2* tend to focus on edges and simple textures; *conv3* responds highly to more complex shapes, such as edges resembling palm trees (channel 15) and arches (channel 48); *conv4* focuses on object parts, such as green dome-like shapes (channel 2), or arches (channel 30); *conv5* shows strong activations for entire objects, with substantial invariances to viewpoint and lighting: entire buildings (channel 3), islands (channel 23) or towers (channel 52) are captured.

These visualizations help with intuitive understanding of our proposed method, which composes local features from a shallower layer (*conv4*) and global features from a deeper layer (*conv5*). Features from *conv5* present high degree of viewpoint invariance, being suboptimal for localized matching and more suitable to global representations. In contrast, *conv4* features seem more grounded to localizable object parts and thus can be effectively used as local feature representations.

F.2 Feature visualization by optimization

In this experiment, we consider the same layers and channels as above, but now adopt a visualization technique by optimizing the input image to maximally activate the desired feature. First, the input image is initialized with random noise. Given the desired layer/channel, we backpropagate gradients in order to maximally activate it. Regularizers can be useful to restrict the optimization space, otherwise the network may find ways to activate neurons that don't occur in natural images.

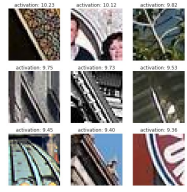
We reuse the technique from Olah et al.[41], with default parameters, and the results are presented in Fig. 8. Again, we notice that a hierarchical representation structure forms, with more complex patterns being produced as the network goes deeper from *conv2* to *conv5*. As expected, the produced images agree very well with the patches from Fig. 7, in terms of the types of visual contents. For example, for *conv3* channel 48, the optimized image shows arch-like edges while the dataset examples present image patches where those types of patterns occur.

Note also how deeper layers tend to specialize for the target task, by detecting image patches with parts and texture that are common to landmarks. For example, *conv4* shows detection of green dome-like shapes (channel 2) and arches (channel 30), which are common in these types of objects; *conv5*, on the other hand, shows building walls (channel 3) and rocky patterns that are common in ancient buildings or islands (channel 23).

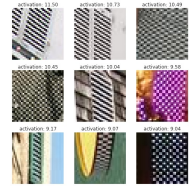
References

1. Arandjelović, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: NetVLAD: CNN Architecture for Weakly Supervised Place Recognition. In: Proc. CVPR (2016) 1, 3
2. Araujo, A., Norris, W., Sim, J.: Computing Receptive Fields of Convolutional Neural Networks. Distill (2019), https://distill.pub/2019/computing-receptive-fields 5, 19
3. Avrithis, Y., Tolias, G.: Hough Pyramid Matching: Speeded-up Geometry Re-ranking for Large Scale Image Retrieval. IJCV (2014) 3
4. Babenko, A., Lempitsky, V.: Aggregating Local Deep Features for Image Retrieval. In: Proc. ICCV (2015) 3, 10
5. Babenko, A., Slesarev, A., Chigorin, A., Lempitsky, V.: Neural Codes for Image Retrieval. In: Proc. ECCV (2014) 3
6. Barroso-Laguna, A., Riba, E., Ponsa, D., Mikolajczyk, K.: Key.Net: Keypoint Detection by Handcrafted and Learned CNN Filters. In: Proc. ICCV (2019) 3
7. Bay, H., Ess, A., Tuytelaars, T., Van Gool, L.: Speeded-Up Robust Features (SURF). CVIU (2008) 1, 3

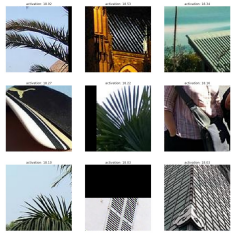
conv2, channel 0



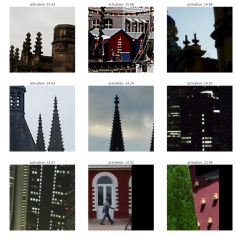
conv2, channel 12



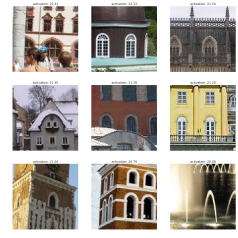
conv2, channel 49



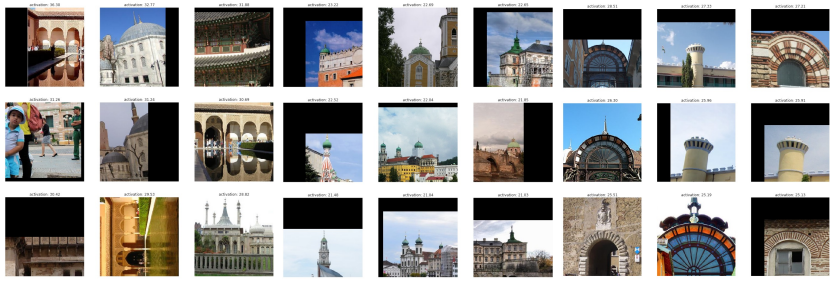
conv3, channel 15



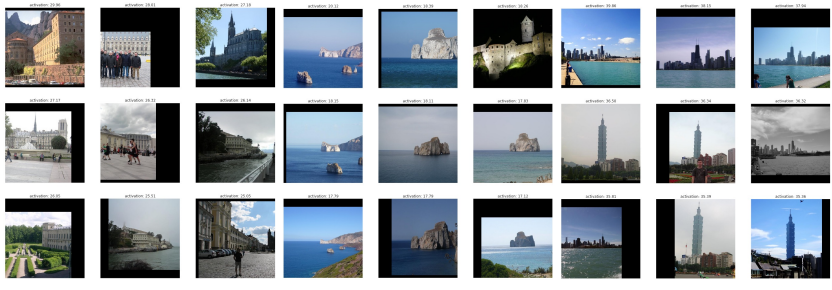
conv3, channel 29



conv3, channel 48



conv4, channel 1 *conv4*, channel 2 *conv4*, channel 30



conv5, channel 3 *conv5*, channel 23 *conv5*, channel 52

Fig. 7: Visualization of patterns detected by specific feature maps / channels, by presenting image patches that produce high activations.

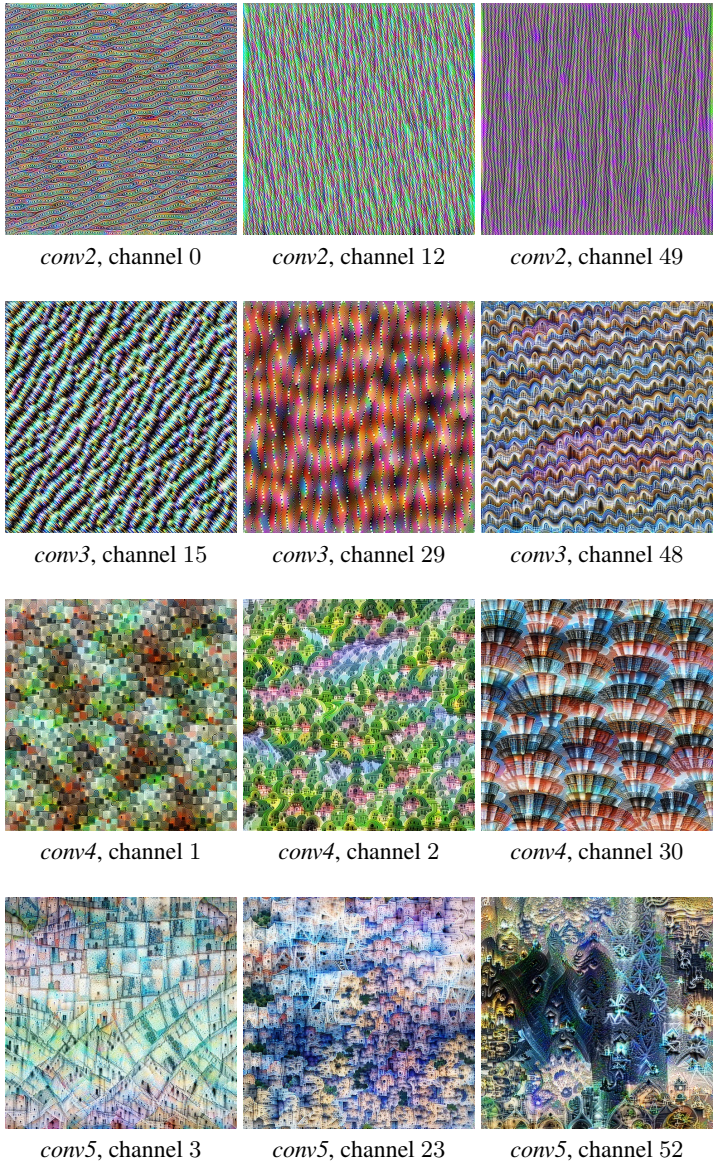


Fig. 8: Visualization of patterns detected by specific feature maps / channels, by optimizing the input image to produce high activations.

8. Chang, C., Yu, G., Liu, C., Volkovs, M.: Explore-Exploit Graph Traversal for Image Retrieval. In: Proc. CVPR (2019) 9
9. Chopra, S., Hadsell, R., LeCun, Y.: Learning a Dimilarity Metric Discriminatively, with Application to Face Verification. In: Proc. CVPR (2005) 3
10. Chum, O., Philbin, J., Sivic, J., Isard, M., Zisserman, A.: Total Recall: Automatic Query Expansion with a Generative Feature Model for Object Retrieval. In: Proc. ICCV (2007) 9
11. Deng, J., Guo, J., Xue, N., Zafeiriou, S.: ArcFace: Additive Angular Margin Loss for Deep Face Recognition. In: Proc. CVPR (2019) 3, 6, 10
12. DeTone, D., Malisiewicz, T., Rabinovich, A.: SuperPoint: Self-Supervised Interest Point Detection and Description. In: Proc. CVPR Workshops (2018) 3
13. Dugas, C., Bengio, Y., Nadeau, C., Garcia, R.: Incorporating Second-Order Functional Knowledge for Better Option Pricing. In: Proc. NIPS (2001) 8
14. Dusmanu, M., Rocco, I., Pajdla, T., Pollefeys, M., Sivic, J., Torii, A., Sattler, T.: D2-Net: A Trainable CNN for Joint Detection and Description of Local Features. In: Proc. CVPR (2019) 3, 5, 12
15. Fischler, M., Bolles, R.: Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Communications of the ACM (1981) 3, 8
16. Gordo, A., Almazan, J., Revaud, J., Larlus, D.: Deep Image Retrieval: Learning Global Representations for Image Search. In: Proc. ECCV (2016) 3
17. Gordo, A., Almazan, J., Revaud, J., Larlus, D.: End-to-end Learning of Deep Visual Representations for Image Retrieval. IJCV (2017) 1, 3, 4, 5, 8, 11, 13
18. Gordo, A., Rodriguez-Serrano, J.A., Perronin, F., Valveny, E.: Leveraging Category-Level Labels For Instance-Level Image Retrieval. In: Proc. CVPR (2012) 3
19. He, K., Lu, Y., Sclaroff, S.: Local Descriptors Optimized for Average Precision. In: Proc. CVPR (2018) 3
20. He, K., Zhang, X., Ren, S., Sun, J.: Deep Residual Learning for Image Recognition. In: Proc. CVPR (2016) 7, 19
21. Hinton, G.: Connectionist Learning Procedures. Artificial Intelligence (1989) 5
22. Iscen, A., Tolias, G., Avrithis, Y., Furon, T., Chum, O.: Efficient Diffusion on Region Manifolds: Recovering Small Objects with Compact CNN Representations. In: Proc. CVPR (2017) 9
23. Jégou, H., Chum, O.: Negative Evidences and Co-Occurences in Image Retrieval: The Benefit of PCA and Whitening. In: Proc. ECCV (2012) 3
24. Jégou, H., Douze, M., Schmid, C.: Hamming Embedding and Weak Geometric Consistency for Large Scale Image Search. In: Proc. ECCV (2008) 3
25. Jégou, H., Douze, M., Schmidt, C., Perez, P.: Aggregating Local Descriptors into a Compact Image Representation. In: Proc. CVPR (2010) 3
26. Jégou, H., Perronnin, F., Douze, M., Sanchez, J., Perez, P., Schmid, C.: Aggregating Local Image Descriptors into Compact Codes. IEEE Transactions on Pattern Analysis and Machine Intelligence (2012) 1, 3
27. Jegou, H., Zisserman, A.: Triangulation Embedding and Democratic Aggregation for Image Search. In: Proc. CVPR (2014) 3
28. Lowe, D.: Distinctive Image Features from Scale-Invariant Keypoints. IJCV (2004) 1, 3, 12, 15
29. Luo, Z., Shen, T., Zhou, L., Zhang, J., Yao, Y., Li, S., Fang, T., Quan, L.: ContextDesc: Local Descriptor Augmentation with Cross-Modality Context. In: Proc. CVPR (2019) 3, 5
30. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust Wide-Baseline Stereo from Maximally Stable Extremal Regions. Image and Vision Computing (2004) 3
31. Mikolajczyk, K., Matas, J.: Improving Descriptors for Fast Tree Matching by Optimal Linear Projection. In: Proc. ICCV (2007) 3

32. Mikolajczyk, K., Schmid, C.: An Affine Invariant Interest Point Detector. In: Proc. ECCV (2002) 3
33. Mishchuk, A., Mishkin, D., Radenovic, F., Matas, J.: Working Hard to Know your Neighbor's Margins: Local Descriptor Learning Loss. In: Proc. NIPS (2017) 3
34. Mishkin, D., Radenovic, F., Matas, J.: Repeatability Is Not Enough: Learning Affine Regions via Discriminability. In: Proc. ECCV (2018) 1, 3, 11
35. Mukundan, A., Tolias, G., Bursuc, A., Jegou, H., Chum, O.: Understanding and Improving Kernel Local Descriptors. IJCV (2019) 3
36. Mukundan, A., Tolias, G., Chum, O.: Explicit Spatial Encoding for Deep Local Descriptors. In: Proc. CVPR (2019) 5
37. Ng, T., Balntas, V., Tian, Y., Mikolajczyk, K.: SOLAR: Second-Order Loss and Attention for Image Retrieval. In: Proc. ECCV (2020) 8, 13, 15
38. Nistér, D., Stewenius, H.: Scalable Recognition with a Vocabulary Tree. In: Proc. CVPR (2006) 1
39. Noh, H., Araujo, A., Sim, J., Weyand, T., Han, B.: Large-Scale Image Retrieval with Attentive Deep Local Features. In: Proc. ICCV (2017) 1, 2, 3, 4, 5, 8, 9, 11, 12, 13, 16, 19
40. Obdrzalek, S., Matas, J.: Sub-linear indexing for large scale object recognition. In: Proc. BMVC (2005) 3
41. Olah, C., Mordvintsev, A., Schubert, L.: Feature Visualization. Distill (2017), https://distill.pub/2017/feature-visualization 19, 20
42. Ono, Y., Trulls, E., Fua, P., Yi, K.M.: LF-Net: Learning Local Features from Images. In: Proc. NIPS (2018) 3
43. Ozaki, K., Yokoo, S.: Large-scale Landmark Retrieval/Recognition under a Noisy and Diverse Dataset. arXiv:1906.04087 (2019) 2
44. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Object Retrieval with Large Vocabularies and Fast Spatial Matching. In: Proc. CVPR (2007) 1, 3, 8
45. Philbin, J., Chum, O., Isard, M., Sivic, J., Zisserman, A.: Lost in Quantization: Improving Particular Object Retrieval in Large Scale Image Databases. In: Proc. CVPR (2008) 1, 3, 8
46. Radenović, F., Iscen, A., Tolias, G., Avrithis, Y., Chum, O.: Revisiting Oxford and Paris: Large-Scale Image Retrieval Benchmarking. In: Proc. CVPR (2018) 8, 11, 17
47. Radenović, F., Tolias, G., Chum, O.: Fine-tuning CNN Image Retrieval with No Human Annotation. IEEE Transactions on Pattern Analysis and Machine Intelligence (2018) 1, 2, 3, 4, 5, 7, 8, 9, 10, 11, 13
48. Revaud, J., Almazan, J., de Rezende, R.S., de Souza, C.R.: Learning with Average Precision: Training Image Retrieval with a Listwise Loss. In: Proc. ICCV (2019) 1, 3, 4, 8, 11, 15
49. Revaud, J., Souze, C.D., Weinzaepfel, P., Humenberger, M.: R2D2: Repeatable and Reliable Detector and Descriptor. In: Proc. NeurIPS (2019) 3, 5
50. Sarlin, P.E., Cadena, C., Siegwart, R., Dymczyk, M.: From Coarse to Fine: Robust Hierarchical Localization at Large Scale. In: Proc. CVPR (2019) 2, 3
51. Schroff, F., Kalenichenko, D., Philbin, J.: FaceNet: A Unified Embedding for Face Recognition and Clustering. In: Proc. CVPR (2015) 3
52. Silberman, N., Guadarrama, S.: TensorFlow-Slim Image Classification Model Library. https://github.com/tensorflow/models/tree/master/research/slim, 2016 7
53. Simeoni, O., Avrithis, Y., Chum, O.: Local Features and Visual Words Emerge in Activations. In: Proc. CVPR (2019) 3, 11, 12
54. Sivic, J., Zisserman, A.: Video Google: A Text Retrieval Approach to Object Matching in Videos. In: Proc. ICCV (2003) 3
55. Taira, H., Okutomi, M., Sattler, T., Cimpoi, M., Pollefeys, M., Sivic, J., Pajdla, T., Torii, A.: InLoc: Indoor Visual Localization with Dense Matching and View Synthesis. In: Proc. CVPR (2018) 2, 3

56. Teichmann, M., Araujo, A., Zhu, M., Sim, J.: Detect-to-Retrieve: Efficient Regional Aggregation for Image Search. In: Proc. CVPR (2019) 8, 11, 12, 13, 15, 17
57. Tian, Y., Yu, X., Fan, B., Wu, F., Heijnen, H., Balntas, V.: SOSNet: Second Order Similarity Regularization for Local Descriptor Learning. In: Proc. CVPR (2019) 12
58. Tolias, G., Avrithis, Y., Jegou, H.: Image Search with Selective Match Kernels: Aggregation Across Single and Multiple Images. IJCV (2015) 3, 11
59. Tolias, G., Jenicek, T., Chum, O.: Learning and Aggregating Deep Local Descriptors for Instance-Level Recognition. In: Proc. ECCV (2020) 13
60. Tolias, G., Sicre, R., Jégou, H.: Particular Object Retrieval with Integral Max-Pooling of CNN Activations. In: Proc. ICLR (2015) 3
61. Wang, F., Xiang, X., Cheng, J., Yuille, A.: NormFace: L2 Hypersphere Embedding for Face Verification. In: Proc. ACM MM (2017) 6
62. Wang, H., Wang, Y., Zhou, Z., Ji, X., Li, Z., Gong, D., Zhou, J., Liu, W.: Cosface: Large Margin Cosine Loss for Deep Dace Recognition. In: Proc. CVPR (2018) 3
63. Weyand, T., Araujo, A., Cao, B., Sim, J.: Google Landmarks Dataset v2 - A Large-Scale Benchmark for Instance-Level Recognition and Retrieval. In: Proc. CVPR (2020) 2, 8, 9, 12, 16, 17
64. Yi, K.M., Trulls, E., Lepetit, V., Fua, P.: LIFT: Learned Invariant Feature Transform. In: Proc. ECCV (2016) 1, 3
65. Yokoo, S., Ozaki, K., Simo-Serra, E., Iizuka, S.: Two-stage Discriminative Re-ranking for Large-scale Landmark Retrieval. In: Proc. CVPR Workshops (2020) 12
66. Zeiler, M.D., Fergus, R.: Visualizing and Understanding Convolutional Networks. In: Proc. ECCV (2014) 2, 4, 19