Expressive Local Feature Match for Image Search

Zechao Hu and Adrian G. Bors
Department of Computer Science, University of York, York YO10 5GH, UK
E-mail: {zh955, adrian.bors}@york.ac.uk

***Abstract*—Content-based image retrieval (CBIR) aims to search the most similar images to a given query content, from a large pool of images. Existing state of the art works would extract a compact global feature vector for each image and then evaluate their similarity. Although some CBIR works utilize local features to get better retrieval results, they either would require extra codebook training or use re-ranking for improving the retrieved results. In this work, we propose a many-to-many local feature matching for large scale CBIR tasks. Unlike existing local feature based algorithms which tend to extract large amounts of short-dimensional local features from each image, the characteristic feature representation in the proposed approach is modeled for each image aiming to employ fewer but more expressive local features. Characteristic latent features are selected using k-means clustering and then fed into a similarity measure, without using complex matching kernels or codebook references. Despite the straightforwardness of the proposed CBIR method, experimental results indicate state of art results on several benchmark datasets.**

I. Introduction

Content-based image retrieval (CBIR) is the task of searching the most similar images from a given database with respect to a given query. The crucial mechanisms within a CBIR pipeline, which are explored for improvement in most research studies, are the feature extraction and the similarity measure used for the retrieval. Initial CBIR approaches would employ hand-crafted descriptors using either low-level features such as colour [1] and texture [2], or more complex information such as by modelling human observation based image saliency [3]. Due to the difficulty of bridging the low-level feature information and high-level semantic gap, such hand-crafted approaches eventually did not provide good-enough results and this field was revolutionized by the Convolution Neural Networks (CNN). The CNN based feature extraction could be divided into two categories as global and local features. Global feature methods [4]–[7] extract a compact feature vector for each image following a single forward processing passing through the network and then entering them into a similarity measure such as the L2 distance.

Local feature methods using CNNs preserve the direct correspondence to a location in the image. The way how local features are used for CBIR can be categorized into three different directions. The first category of local feature methods implements a separate aggregation method to encode local features into a compact feature vector [8]–[10]. Meanwhile, the second category would select several local feature vectors from each image and then employ a similarity measure in a many-to-many manner [11], [12], or use co-attention [13]. Normally such methods use specific matching kernels and would require extra codebook training. In a third category, certain methods [14], [15] utilize the spatial information of each local feature vector to perform a verification at the re-ranking stage. In addition, local feature methods normally involve a feature selection module to filter out unwanted features as some locations from the convolution feature tensor correspond to irrelevant background content.

Due to the fixed receptive field of CNNs and the potential variation in the object size, due to the image acquisition process, each local feature may only correspond to a part of a target object. To address this problem, local feature vectors are extracted from multi-scale representations of the input image and, despite employing a feature selection mechanism, result in too many local features extracted for each image. When considering the memory cost for feature storage at the offline stage and the time required during the retrieval stage, it will be impractical to cache a large number of high-dimensional local features for each image. Consequently, most local feature methods would employ a feature dimension reduction.

This research study proposes to process the latent-space of a pre-trained CNN when applied on pairs of query and candidate search images from a database. We propose extracting a more expressive local feature representations for each image through clustering on the features extracted by a baseline CNN. Such an approach is better and more efficient than storing large amounts of features for each image. Meanwhile, a corresponding many-to-many similarity measure is implemented on the extracted local features. The proposed approach does not require the support of large codebook training or complex matching kernels.

The contributions of this study are: 1) We propose a clustering-based representative feature extraction method from the feature tensor output resulting in fewer, but expressive local features; 2) We propose an effective many-to-many local matching method; 3) Extensive experimental results indicate state-of-the-art capabilities for the proposed method.

II. Related works

Global features. The first attempt at estimating compact feature vectors from probabilistic representations of convolution feature tensors was the Neural Code model [4], which implements a fully connected layer to transform the feature tensor output by the final convolution layer into a compact feature vector for image retrieval. In [16], spatial pooling is shown to be a better mechanism for instance retrieval. A series of spatial pooling methods were proposed to build compact feature vectors from the statistics of convolution

feature tensors, including sum-pooling [5], max-pooling [6], [17], [18] and generalized mean pooling [7], [19]. Afterwards, several modifications or complementary modules have been considered on the spatial pooling pipeline in various models, in order to further refine or enhance the compact global feature vector of each image. For instance, in [17], [18], an end-to-end trainable region proposal network (RPN) [20] is implemented to enhance regional max-pooling for image retrieval. The weighted generalized mean pooling (WGeM) [19] adds an extra convolutional layer at the end of a CNN backbone structure as a trainable spatial weighting module to localize and highlight potential regions of interest before the final global spatial pooling. The Second-Order Loss and Attention for image Retrieval (SOLAR) [21] explored using co-relations on the CNN feature tensor with a second-order attention layer. Traditional spatial attention modules generate heatmap-like attention maps for a given image. The second-order attention layer sensitivity maps for each location on the convolution feature tensor with respect to all others. These attention maps would highlight regions relevant to the training data, leading to better global feature representations for image retrieval. The recently proposed Deep Orthogonal Local and Global (DOLG) [22] employs an orthogonal fusion module to complement the global feature vector with local feature information.

Local features. The Bag Of Visual words (BOV) [23] is a representative local aggregation method for local features representative of an image. During the training, BOV extracts local feature vectors from a set of sample images and then applies k-means clustering. Each cluster center is treated as a "visual word" and a codebook is composed from all these visual words. At the retrieval stage, for each image, their local descriptors are extracted and clustered into visual words. Then the frequency of the visual words is built as the representation code for each image. The Bag of Local Convolutional Features (BLCF) [8] represents a method combining CNN-based local features with BOV. Another popular local feature aggregation method is the Vector of Locally Aggregated Descriptors (VLAD) [9], [24]. Similar to BOV, VLAD also needs to train the codebook and performs local feature clustering for the images from each database. The difference is that instead of using the frequency of each visual word, the distance between each local descriptor and the cluster center are accumulated and concatenated to build a compact fixed size feature vector for the whole image. The NetVLAD [10] modifies VLAD as an end-to-end trainable layer at the tail of a CNN structure. The experimental results show that the trainable VLAD outperforms the local feature fusion methods which are not based on deep learning.

The DEep Local Feature (DELF) [14] is a representative two-stage local feature model utilizing local features to re-rank initial retrieval results. It implements a score function with two processing layers on top of the final convolution layer for relevant local feature selection. During the initial retrieval stage, the compact global feature vector is built by a weighted sum of selected local features. During the re-ranking stage, after dimension reduction, geometry verification is performed with these local features to get the final retrieval result. DEep Local and Global features (DELG) model [15], was derived from DELF, by unifying the training procedures of global and local features into a single pipeline while further improving the performance of this two-stage image retrieval framework.

HOW [11] relies upon the Aggregated Selective Match Kernel (ASMK) [25] to directly perform many-to-many local feature matching with selected local features, achieving better results with lower memory costs than DELF [14]. However, HOW has to train the codebook as BOV. The codebook of HOW contains more than 60,000 visual words learned from 20 million local feature samples obtained from 20,000 training images with 1,000 local features extracted from each image.

III. Preliminary

In this section, we start with introducing spatial pooling and the baseline model structure. Afterwards, we present the proposed expressive local feature extraction and the many-to-many matching method for content-based image retrieval.

A. Spatial Pooling

Considering an input image $\mathbf{I}$, which after being processed by a convolutional network is mapped into a feature tensor $\mathbf{X} \in I\!R^{H \times W \times D}$, where H, W, D represent the height, width and channel counts. Eventually, the global spatial pooling compresses the feature tensor into :

$$\mathbf{V}_g = \left(\frac{1}{L} \sum_{l}^{L} {\mathbf{x}_l}^p \right)^{\frac{1}{p}}, \tag{1}$$

where $\mathbf{V}_g \in I\!R^{1 \times D}$, $L = H \times W$, $l = 1, \ldots, L$ and $\mathbf{x}_l$ indicates the local feature vector from $\mathbf{X}$ at location l, while p is a power coefficient. When $p = 1$, Eq. (1) becomes the sum-pooling (average-pooling) [5], while for $p \to \infty$ it is the max-pooling [6]. When $p \in (1, \infty)$, Eq. (1) becomes the Generalized Mean (GeM) pooling, [7]. The similarity measure between spatial pooling feature vector is usually performed by calculating cosine similarity (or L2 distance after normalization). Due to the usage of the power coefficient p, the generalized mean pooling is more selective than the simple average-pooing but involves more local features than max-pooling.

Given a query image I_q and a candidate image I_c, we have their corresponding feature tensors $\mathbf{X}_q$, $\mathbf{X}_c$ and global spatial pooling feature vectors $\mathbf{V}_q$ and $\mathbf{V}_c$. The cosine similarity between the query and candidate image representations is :

$$\begin{aligned} cos(\mathbf{V}_q, \mathbf{V}_c) &= (\eta(\mathbf{V}_q)\mathbf{V}_q)(\eta(\mathbf{V}_c)\mathbf{V}_c)^\mathsf{T} \\ &= \frac{\eta(V_q)\eta(V_c)}{(L_q L_c)^{\frac{1}{p}}} \sum_{d=1}^{D} \left(\sum_{l_q=1}^{L_q} \sum_{l_c=1}^{L_c} \left(x_{q,l_q,d} x_{c,l_c,d} \right)^p \right)^{\frac{1}{p}} \end{aligned} \tag{2}$$

where the L2 normalization is defined by $\eta(\mathbf{V}) = \mathbf{1} / \|\mathbf{V}\|$. According to Eq. (2), the cosine similarity between two global spatial pooling feature vectors can be treated as the sum of dimension-wise multiplications between the entries of

the corresponding feature tensors, representing the query and candidate images. According to [11], at the training stage, any loss function, including the contrastive loss [26] or the triplet loss [10], that optimizes the cosine similarity between feature vectors corresponding to an image pair, with the following consequences. Firstly, the L2 norm of irrelevant background locations is minimized ($\left\|\mathbf{X}_{l_{bg}}\right\| \rightarrow 0$) leading to little or no contribution to the final similarity score. On the contrary, the L2 norm of distinct foreground objects or region locations will be maximized. Meanwhile, the local features from matching location pairs will be moved towards becoming closer to each other, while those unmatched will be marginalized in the feature space.

B. Baseline model structure and training

The general framework of using a deep CNN for feature tensor extraction followed by a global spatial pooling layer for building a compact global feature vector was used in the recent state of the art works, such as DELG [15] and DOLG [22]. In this research study we also use ResNet101 [27] as the backbone network for feature tensor extraction. The feature tensor output from the final convolution layer is pooled by a generalized mean pooling layer from equation (1), with a fixed power co-efficient $p = 3$, followed by a trainable fully connected layer for feature whitening.

Following the approach in DELG [15], we also consider image-level class labels and the ArcFace margin loss [28] for the model training, defined by:

$$L(\widehat{\mathbf{V}}_g, \mathbf{y}) = -\log \frac{\exp(\gamma \times \mathrm{AF}(\widehat{\mathbf{V}}_g \widehat{\mathbf{w}}_i^{\mathsf{T}}, y_i))}{\sum_{j=1}^{N_c} \exp(\gamma \times \mathrm{AF}(\widehat{\mathbf{V}}_g \widehat{\mathbf{w}}_j^{\mathsf{T}}, y_j))}, \quad (3)$$

where $\widehat{\mathbf{V}}_g$ is whitened then L2 normalized global GeM feature vector as in (1), during training. $\widehat{\mathbf{w}}_i$ refers to the trainable L2 normalized classifier weights for class i from the ArcFace weight matrix $\mathcal{W} \in I\!R^{N_c \times D}$, N_c is the number of classes in the training dataset. $\mathbf{y}$ is an one-hot class label vector and i is the index of the ground-truth class of $\widehat{\mathbf{V}}_g$ ($y_i = 1$). γ is a trainable temperature parameter. $\mathrm{AF}(u, y)$ is the ArcFace-adjusted cosine similarity [15]:

$$\mathrm{AF}(u, y) = \begin{cases} \cos(\arccos(u) + m), & \text{if } y = 1 \\ u, & \text{if } y = 0 \end{cases} \quad (4)$$

where u is the cosine similarity, y indicates whether it is the ground-truth class and m is the ArcFace margin.

The ArcFace margin loss from Eq. (4) could also be referred as a "cosine classifier" [15]. Within the ArcFace weight matrix $\mathcal{W}$, each row $\mathbf{w}_i$, $i \in \{1, 2, 3, \ldots, N_c\}$ could be treated as a proxy feature vector for class i. In other words, these proxy features approximate corresponding original class image features. Accordingly, the ArcFace loss potentially optimizes the cosine similarity not between single image pairs but between the query and proxies of image classes. Compared to the traditional image pair similarity loss (contrastive loss or triplet loss) this kind of proxy based similarity loss does not need hard sample mining and would converge faster than the simple similarity loss between specific image pairs [29].

IV. Expressive local feature extraction and matching

In the following, we describe how we compress the feature tensor extracted by the backbone network in order to extract expressive local feature representations for performing many-to-many local matching for the CBIR task.

A. Local feature selection and clustering

The first problem when performing many-to-many local matches is the computation cost required by the large number of local features extracted from a given image. Considering an input image $\mathbf{I}$ of size $h \times w$, after feeding through a ResNet101 [27], considered as backbone, which contains 5 downsampling blocks, the output feature tensor $\mathbf{X}$ size becomes $\frac{h}{2^5} \times \frac{w}{2^5}$. For example for a high resolution image of 1024×1024 pixels, it would result in hundreds of local features being extracted. However, not all local features from the feature tensor are relevant for CBIR, and irrelevant features, such as for example those corresponding to the background should be discarded. Accordingly, we first perform local feature selection over the feature tensor output by the backbone network. As discussed in Section III-A, the L2 norm of each entry from the CNN feature tensor reflects its importance. As shown in the left part of Fig. 1, the feature selection is performed based on the L2 norm mask of the feature tensor $\mathbf{X}$. We keep only the top N features with the highest L2 norm, resulting in a set of local features $\mathbf{X}_{1:N} = \{\mathbf{x}_i | i = 1, \ldots, N\}$, where $\mathbf{x}_i \in I\!R^{1 \times D}$ indicates the i-th local feature vector from $\mathbf{X}_{1:N}$.

At this stage, each local feature can be treated as corresponding to a small region from the input image. Extracted features may not be comprehensive enough, as each local feature contains very limited information. For example it may only represent a small part from an object of interest and lack the higher-level semantic meaning. Meanwhile, we want to further reduce the number of candidate local features for the sake of controlling the computational complexity. Thus, for the initially selected local features $\mathbf{X}_{1:N}$, we employ the k-means clustering in order to build representative feature centers for groups of features. However, the clustering result for k-means varies with the number of clusters k and their initialization. This is an unwanted attribute for a stable image retrieval system and for this reason we adapt the k-means++ [30] for the cluster' center initialization. Considering the candidate image local features $\mathbf{X}_{1:N}$ as input, we consider the following steps for k-means++ :

1) Let $\mathbf{A}$ represent the local features from $\mathbf{X}_{1:N}$ that have not been selected as representative centers, while $\mathbf{B}$ represents the set of those chosen. In the beginning we have, $\mathbf{A} = \{\mathbf{x}_i | i = 1, \ldots, N\}$ and $\mathbf{B} = \emptyset$.
2) Choose $\mathbf{x}_m \in \mathbf{X}_{1:N}$, such that $m = \underset{\mathbf{x}_i \in \mathbf{A}}{\arg\max} \|\mathbf{x}_i\|$, as the first cluster center. Meanwhile, add $B = B \bigcup \mathbf{x}_m$, while it is deleted from $\mathbf{A} = \mathbf{A} \setminus \mathbf{x}_m$.
3) For each local feature vector $\mathbf{x}_i \in \mathbf{A}$ that has not been chosen as a center yet, compute the smallest distance with respect to all chosen initial centers $d(\mathbf{x}_i) =$

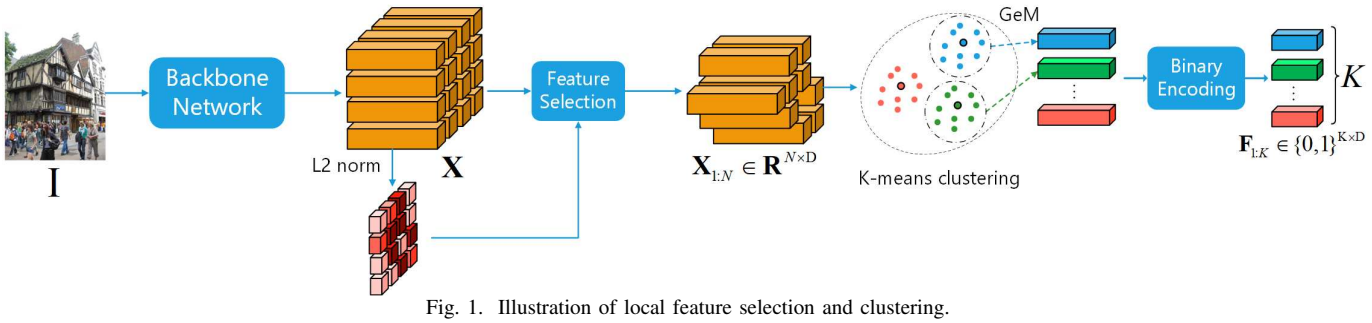

Fig. 1. Illustration of local feature selection and clustering.

$\min \|\mathbf{x}_i - \mathbf{x}_j\|$, $\mathbf{x}_j \in \mathbf{B}$, $j = 1, \ldots, |\mathbf{B}|$, where $|\cdot|$ denotes the cardinality of a set.

4) Choose $\mathbf{x}_l \in \mathbf{A}$, such that $l = \arg\max_{\mathbf{x}_i \in \mathbf{A}} d(\mathbf{x}_i)$ as another cluster center, adding it to $B = B \bigcup \mathbf{x}_l$ and deleting it from $\mathbf{A} = \mathbf{A} \setminus \mathbf{x}_l$.
5) Repeat Steps 3) and 4) until $|\mathbf{B}| \equiv K$.

The selected cluster centers are then used for initializing the standard k-means clustering. As shown in the right part of Fig. 1, after clustering, we perform Generalized Mean (GeM) pooling as in Eq. (1) within each cluster, to obtain its center.

To further reduce the memory requirement, a binary encoding function $b : \mathbb{R}^D \rightarrow \{0,1\}^D$ is applied to each cluster center:

$$b(x) = \begin{cases} 0, & \text{if } x \leq 0 \\ 1, & \text{if } x > 0 \end{cases} \tag{5}$$

Finally, a binary coded local feature set $\mathbf{F}_{1:K} = \{\mathbf{f}_i | i = 1, \ldots, K\}$ is extracted from the input image $\mathbf{I}$. $\mathbf{f}_i \in \{0,1\}^{1\times D}$ indicates the i-th binary coded local feature vector from $\mathbf{F}_{1:K}$, which consists of D bits.

B. Local feature match

Let us consider a pair of images, representing the query $\mathbf{I}_q$ and the candidate $\mathbf{I}_c$, from a given database. After the feature extraction as mentioned above, two corresponding binary coded local feature sets $\mathbf{F}_{q,1:K} = \{\mathbf{f}_{q,i} | i = 1, \ldots, K\}$ and $\mathbf{F}_{c,1:K} = \{\mathbf{f}_{c,j} | j = 1, \ldots, K\}$ are extracted. Then, a similarity matrix $\mathbf{M} = \{[m_{i,j}] \in K \times K\}$ is obtained by calculating the similarity score between each pair of query local feature $\mathbf{f}_{q,i}$ and the local feature candidates $\mathbf{f}_{c,j}$:

$$m_{i,j} = d(\mathbf{f}_{q,i}, \mathbf{f}_{c,j}), \tag{6}$$

where $d(\cdot,\cdot)$ is a similarity function. For feature vectors described as real numbers, it could be the cosine similarity while the Hamming distance can be considered between two binary coded sequences. The Hamming distance represents the count of those bits which are different between two vectors and ranges between $[0, D]$ for a vector of dimension D. It can be normalized to the range $[0, 1]$ by dividing it with the feature dimension D. In this case we evaluate the similarity using $1 - d(\mathbf{f}_{q,i}, \mathbf{f}_{c,j})/D$. Accordingly, the i-th row of the similarity matrix $\mathbf{M}$ stores the similarity score between $\mathbf{f}_{q,i}$ and each local feature from the candidate image feature set $\mathbf{F}_{c,1:K}$.

In principle, matrix $\mathbf{M}$ needs to be transformed into a single similarity score between the image pair $\{\mathbf{I}_q, \mathbf{I}_c\}$ and its calculation should be quick. Thus, we first define the similarity score between a single query local feature $\mathbf{x}_{q,i}$ and those from whole candidate image by:

$$s(\mathbf{f}_{q,i}, \mathbf{I}_c) = \max_j m_{i,j}. \tag{7}$$

Eventually, the similarity between images $\mathbf{I}_q$ and $\mathbf{I}_c$ is :

$$S(\mathbf{I}_q, \mathbf{I}_c) = \frac{\sum_{i=1}^{K} s(\mathbf{f}_{q,i}, \mathbf{I}_c)}{K}, \tag{8}$$

where K is the number of clusters.

V. Experiments

We first discuss our experimental setup, including the hyper-parameter setting and implementation details. Then we provide retrieval results for the proposed methodology followed by comparisons with the state of the art.

A. Experimental setup

Implementation details. We consider ResNet101 [27] as the backbone network. For the baseline GeM model training, we set the margin $m = 0.15$ and temperature $\gamma = \sqrt{D} = 30$ for the ArcFace loss in (4) and (3), respectively. We train the model on a clean subset of Google landmark dataset version 2 (GLDv2) [31], which contains more than 1.5M images grouped into 81,313 classes. GLDv2 was also used for training the state of the art DELG [15] and DOLG [22] models. We consider data augmentation by randomly cropping, ratio distorting and then resizing images to 512×512 pixels. The model is trained using a Stochastic Gradient Descent (SGD) optimizer with an initial learning rate of 0.05, weight decay of 0.0001, batch size of 128 images, and employing cosine learning rate decay strategy. The spatial pooling power coefficient is $p = 3$ in eq. (1). The baseline model is trained with 4 NVIDIA Tesla GPUs.

At the retrieval stage, if not otherwise specified, we initially select $N = 500$ local features and the number of clusters is set as $K = 10$ for the k-means clustering. After whitening and L2 normalization, all clustered local features are binary encoded by eq. (5).

Evaluation dataset. Revisited Oxford and Paris datasets [32] have commonly been used for large-scale CBIR performance evaluation in recent years. These databases are expanded versions of Oxford [33] and Paris [34] datasets after removing the images with incorrect annotation and adding several new

query images. Revisited Oxford (ROxf) contains 4993 images while Revisited Paris (RPar) has 6322 images. Both datasets contain 70 query images. The ground-truth matching images to each query image are divided into 3 groups, *Easy, Medium, Hard*, according to the level of difficulty in assessing the similarity in their image representation with respect to the corresponding query. In addition, R1M [32] is a new distractor set containing 1 million unbiased high resolution (1024×768 pixels) images for ROxf and RPar. The similarity measure between feature vectors is performed with cosine similarity. All retrieval results are assessed considering the mean average precision (mAP) [33].

Feature extraction with the multi-scale scheme. Multi-scale feature extraction has been widely used in CBIR for both global and local features. When not considering local feature matching, but directly feeding the feature tensor into the generalized mean pooling layer and using the resulting compact global feature tensor for similarity measure, we use 3 scales $\{1, \sqrt{2}, \frac{1}{\sqrt{2}}\}$, as in other global feature approaches [7], [15]. The global feature vectors corresponding to different scales are fused by average pooling then L2 normalized again. When considering the local feature matching, as described in Section IV, we select features from 5 scales $\{\frac{1}{2\sqrt{2}}, \frac{1}{2}, \frac{1}{\sqrt{2}}, 1, \sqrt{2}\}$, as in [11]. Local features resulting from processing different image scales are merged and L2 normalized.

B. Local match visualization

In Fig. 2, we show five examples of the query and target images, the local match maps and the L2 norm attention maps, on each row. The local matching map is displayed as a heatmap showing which locations contribute most to the final global similarity score for the given image pair. Matching scores for the locations that are not selected are set to zero. Local features grouped into the same cluster share the corresponding cluster center as their representative vector. Matching scores of all local features from different input image scales are projected back to the corresponding locations (or regions) from the original image and accumulate, resulting in the final score map. The L2 norm reflects the importance of each location, as discussed in Section III-A, or how much it contributes to the final feature vector obtained by global pooling. Accordingly, the L2 norm attention map also reflects each location's contribution to the image pair similarity. As we can observe, the L2 norm attention maps tend to be evenly distributed over the relevant content of the training data, in this case represented by landmarks or buildings. For some easy cases, when the target region is salient and/or of large scale, it can work very well, but not for some hard cases where there are multiple potential relevant objects. All examples shown in Fig. 2 are rather hard and the L2 norm cannot find the exact location to focus on and would highlight many unwanted regions or even indicate wrong places. On the contrary, when considering our local matching implementation, most corresponding local feature pairs between the query and target images would have the highest similarity score. As a result, these matching local feature locations would also represent most important contributions to the final similarity score of the image pair. When comparing examples 4 and 5, they have different query images but the same target. The local matching in this case highlights the exact part of the query content. In a way, the visualization of local matching maps looks like co-attention, as the importance of each local feature from the candidate image is no longer fixed as in the traditional global spatial pooling. Nevertheless, the effectiveness and contribution of the local features for each candidate image varies with the query feature. In each candidate image from Fig. 2, the local match score comes mostly from the region that matches the query content, even when the target object is not salient or is surrounded by some other similar class objects.

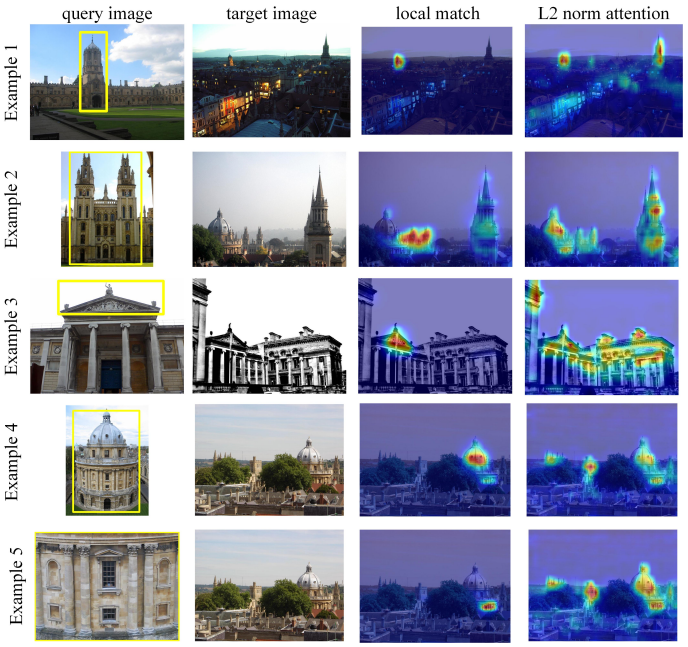

Fig. 2. Visualization of the local matching and L2 norm.

C. Retrieval results

Image retrieval comparison results with the current state of the art are provided in Table I. The retrieval performance is greatly improved by introducing the local match into the CBIR pipeline. Especially, on the *Hard* image sets of ROxf and RPar our method achieves the best results of 71.2% and 83.7%, respectively, outperforming both DELG (with spatial verification re-ranking) and the current state-of-the-art DOLG[1] method. When considering the +1M distractor set, our local match method still gives a great improvement over the baseline model and shows comparable results to DOLG. It seems that when considering the +1M distractor, the mAP improvement is not as significant as before, which means that our method is not robust enough when considering a large scale distractor set. As our baseline model is only trained with a simple global feature vector loss function, more robust local feature training strategy could be explored in future work.

[1]https://github.com/feymanpriv/DOLG

Method	*Medium (%)*				*Hard (%)*			
	ROxf	ROxf+1M	RPar	RPar+1M	ROxf	ROxf+1M	RPar	RPar+1M
(A) Local features + re-ranking								
HesAff-rSIFT-ASMK*+SP [25]	60.6	46.8	61.4	42.3	36.7	26.9	35.0	16.8
HardNet-ASMK*+SP [35]	65.6	-	65.2	-	41.1	-	38.5	-
DELF-ASMK*+SP [36]	67.8	53.8	76.9	57.3	43.1	31.2	55.4	26.4
DELF-D2R-R-ASMK*+SP [36]	76.0	64.0	80.2	59.7	52.4	38.1	58.6	29.4
HOW [11]	79.4	65.8	81.6	61.8	56.9	38.9	62.4	33.7
HOW-MDA [12]	82.0	68.7	83.3	64.7	62.2	45.3	66.2	38.9
(B) Global features								
Res101-R-MAC [17]	60.9	39.3	78.9	54.8	32.4	12.5	59.4	28.0
Res101-GeM (GLD) [7]	67.3	49.5	80.6	57.3	44.3	25.7	61.5	29.8
Res101-GeM-AP [37]	67.5	47.5	80.1	52.5	42.8	23.2	60.5	25.1
Res101-DSM [38]	65.3	47.6	77.4	52.8	39.2	23.2	56.2	25.0
Res101-SOLAR [21]	69.9	53.5	81.6	59.2	47.9	29.9	64.5	33.4
Res101-DELG (GLDv2) [15]	76.3	63.7	86.6	70.6	55.6	37.5	72.4	46.9
Res101-DELG (GLDv2) + SP [15]	81.2	69.1	87.2	71.5	64.0	47.5	72.8	48.7
Res101-DOLG[1] [22]	82.3	73.6	90.9	80.4	64.9	51.6	81.7	62.9
(C) Our method								
Res101-GeM (Baseline)	83.0	72.8	90.2	77.6	65.5	49.8	80.7	59.1
Res101-GeM-Local Match	85.9	77.2	92.0	79.9	71.2	55.1	83.7	61.7

TABLE I

IMAGE RETRIEVAL RESULTS ON ROXF/RPAR DATASETS (AND THEIR EXTENDED VERSION +1M DISTRACTOR SET R1M), CONSIDERING *Medium* AND *Hard* EVALUATION PROTOCOLS. GROUPS (A) AND (B) SEPARATELY SHOW THE RESULTS OF LOCAL FEATURE AND GLOBAL FEATURE METHODS. THE BOTTOM GROUP (C) SHOWS THE RESULTS OF OUR MODEL. "SP" REFERS TO THE SPATIAL VERIFICATION RE-RANKING [36].

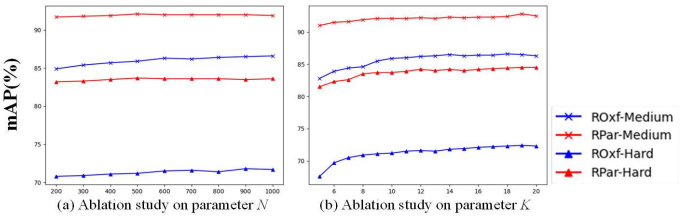

Fig. 3. Ablation study on clustering parameters.

D. Ablation experiments and discussion

Two diagrams in Fig. 3 (a), (b) show the impact of the number of the initially selected features N as well as the number of clusters K for k-means on the retrieval performance. A small N=200 could not model enough local features, while N=1000 is too large and may pick out too many background or irrelevant local features, making the feature selection meaningless. A smaller K could further reduce the computation cost but it would arbitrarily fuse many local features into larger clusters reducing the local matching benefits as well. Relatively larger K could further improve the retrieval performance, but it would also require additional computation costs for a limited improvement.

In the following, we discuss the computation and memory costs of the proposed local match method considering the hyper-parameter setting from Section V-A. After the binary encoding, each element of the clustered local feature vector is a 1 bit in a binary number. In this case, for each candidate image, the memory cost to cache its local features is $K \times D \times 1$ bits. With ResNet101 as the backbone network (D = 2048), $K = 10$, the memory cost for one candidate image cache is $10 \times 2048 \times 1$ bit ≈ 0.00256 MB. It takes around 2.5GB to cache the ROxf/RPar dataset along with the +1M distractor set. The feature extraction, including feeding through the backbone network, feature selection and the clustering procedure, takes in average 200ms to cache a single candidate image's local features when considering 5 input image scales. Such processing becomes rather time consuming when considering large-scale databases, but it is done offline and only once. For the online retrieval speed, searching on ROxf/RPar with +1M distractor dataset, for one query image it takes on average 2s with the multi-process python implementation on a CPU.

VI. CONCLUSION

In this paper we explore extracting few but expressive local features from an input image which along with a corresponding many-to-many local matching method leads to an optimal similarity evaluation under very challenging situations. Unlike other local matching methods that extract large numbers of low-dimensional local features which may require significant codebook training, the proposed local matching method is based on the L2 norm local feature selection and clustering to extract the appropriate number of expressive local features for the local matching. This approach was shown to work with binary encoding for further memory requirement reduction. The proposed model achieves new state of art performance on benchmark datasets.

REFERENCES

[1] M. J. Swain and D. H. Ballard, "Color indexing," *Int. Journal of Computer Vision*, vol. 7, no. 1, pp. 11–32, 1991.

[2] B. S. Manjunath and W.-Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 837–842, 1996.

[3] A. Papushoy and A. G. Bors, "Image retrieval based on query by saliency content," *Digital Signal Processing*, vol. 36, no. 8, pp. 156–173, 2015.

[4] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky, "Neural codes for image retrieval," in *Proc. European Conf. on Computer Vision (ECCV), vol. LNCS 8689*, 2014, pp. 584–599.

[5] A. Babenko and V. Lempitsky, "Aggregating local deep features for image retrieval," in *Proc. IEEE Int. Conf. on computer vision (ICCV)*, 2015, pp. 1269–1277.

[6] G. Tolias, R. Sicre, and H. Jégou, "Particular object retrieval with integral max-pooling of cnn activations," in *Proc. Int. Conf. on Learning Representations (ICLR), arXiv preprint arXiv:1511.05879*, 2016.

[7] F. Radenović, G. Tolias, and O. Chum, "Fine-tuning cnn image retrieval with no human annotation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 41, no. 7, pp. 1655–1668, 2018.

[8] E. Mohedano, K. McGuinness, N. E. O'Connor, A. Salvador, F. Marques, and X. Giró-i Nieto, "Bags of local convolutional features for scalable instance search," in *Proce. of ACM on Int. Conf. on Multimedia Retrieval (ICMR)*, 2016, pp. 327–331.

[9] J. Yue-Hei Ng, F. Yang, and L. S. Davis, "Exploiting local features from deep networks for image retrieval," in *Proc. of CVPR-workshops, vol. LNCS 9913*, 2015, pp. 685–701.

[10] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," in *Proc. IEEE Conf. on Computer Vision and pattern Recognition (CVPR)*, 2016, pp. 5297–5307.

[11] G. Tolias, T. Jenicek, and O. Chum, "Learning and aggregating deep local descriptors for instance-level recognition," in *Proc. European Conf. on Computer Vision (ECCV), vol. LNCS 12346*, 2020, pp. 460–477.

[12] H. Wu, M. Wang, W. Zhou, and H. Li, "Learning deep local features with multiple dynamic attentions for large-scale image retrieval," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 11 416–11 425.

[13] Z. Hu and A. G. Bors, "Conditional attention for content-based image retrieval," in *Proc. British Machine Vision Conference (BMVC)*, 2020.

[14] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han, "Large-scale image retrieval with attentive deep local features," in *Proc. IEEE Int. Conf. on Computer Vision (ICCV)*, 2017, pp. 3456–3465.

[15] B. Cao, A. Araujo, and J. Sim, "Unifying deep local and global features for image search," in *Proc. European Conf. on Computer Vision (ECCV), vol LNCS 12365*, 2020, pp. 726–743.

[16] A. S. Razavian, J. Sullivan, S. Carlsson, and A. Maki, "Visual instance retrieval with deep convolutional networks," *ITE Trans. on Media Technology and Applications*, vol. 4, no. 3, pp. 251–258, 2016.

[17] A. Gordo, J. Almazán, J. Revaud, and D. Larlus, "Deep image retrieval: Learning global representations for image search," in *Proc. European Conf. on computer vision (ECCV), vol. LNCS 9910*, 2016, pp. 241–257.

[18] ——, "End-to-end learning of deep visual representations for image retrieval," *International Journal of Computer Vision*, vol. 124, no. 2, pp. 237–254, 2017.

[19] X. Wu, G. Irie, K. Hiramatsu, and K. Kashino, "Weighted generalized mean pooling for deep image retrieval," in *Proc. IEEE Int. Conf. on Image Processing (ICIP)*, 2018, pp. 495–499.

[20] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: towards real-time object detection with region proposal networks," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.

[21] T. Ng, V. Balntas, Y. Tian, and K. Mikolajczyk, "Solar: second-order loss and attention for image retrieval," in *Proc. European Conf. on Computer Vision (ECCV), vol. LNCS 12370)*, 2020, pp. 253–270.

[22] M. Yang, D. He, M. Fan, B. Shi, X. Xue, F. Li, E. Ding, and J. Huang, "DOLG: Single-stage image retrieval with deep orthogonal fusion of local and global features," in *Proc. of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 11 772–11 781.

[23] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," in *null*, 2003, p. 1470.

[24] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid, "Aggregating local image descriptors into compact codes," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 34, no. 9, pp. 1704–1716, 2011.

[25] G. Tolias, Y. Avrithis, and H. Jégou, "Image search with selective match kernels: aggregation across single and multiple images," *Int. Journal of Computer Vision*, vol. 116, no. 3, pp. 247–261, 2016.

[26] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2005, pp. 539–546.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[28] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 4690–4699.

[29] Y. Movshovitz-Attias, A. Toshev, T. K. Leung, S. Ioffe, and S. Singh, "No fuss distance metric learning using proxies," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 360–368.

[30] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in *Proc. of the Annu. ACM-SIAM Symp. on Discrete Algorithms*, vol. 8, 01 2007, pp. 1027–1035.

[31] T. Weyand, A. Araujo, B. Cao, and J. Sim, "Google Landmarks Dataset v2 - A Large-Scale Benchmark for Instance-Level Recognition and Retrieval," in *Proc. IEEE/CVF Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 2572–2581.

[32] F. Radenovic, A. Iscen, G. Tolias, Y. Avrithis, and O. Chum, "Revisiting Oxford and Paris: Large-Scale image retrieval benchmarking," in *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 5706–5715.

[33] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2007, pp. 1–8.

[34] ——, "Lost in quantization: Improving particular object retrieval in large scale image databases," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2008, pp. 1–8.

[35] D. Mishkin, F. Radenovic, and J. Matas, "Repeatability is not enough: Learning affine regions via discriminability," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[36] M. Teichmann, A. Araujo, M. Zhu, and J. Sim, "Detect-to-retrieve: Efficient regional aggregation for image search," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5109–5118.

[37] J. Revaud, J. Almazán, R. S. Rezende, and C. R. d. Souza, "Learning with average precision: Training image retrieval with a listwise loss," in *Proc. IEEE Int. Conf. on Computer Vision (ICCV)*, 2019, pp. 5107–5116.

[38] O. Siméoni, Y. Avrithis, and O. Chum, "Local features and visual words emerge in activations," in *Proc. of IEEE/CVF Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 11 651–11 660.