FEW BUT INFORMATIVE LOCAL HASH CODE MATCHING FOR IMAGE RETRIEVAL

*Zechao Hu and Adrian G. Bors**

Department of Computer Science, University of York, York YO10 5GH, UK

ABSTRACT

Content-based image retrieval (CBIR) aims to search for the most similar images from an extensive database to a given query content. Existing CBIR works either represent each image with a compact global feature vector or extract a large number of highly compressed low-dimensional local features, where each contains limited information. In this research study, we propose an expressive local feature extraction pipeline and a many-to-many local feature matching method for large-scale CBIR. Unlike existing local feature methods, which tend to extract large amounts of low-dimensional local features from each image, the proposed method models characteristic feature representations for each image, aiming to employ fewer but more expressive local features. For further improving the results, an end-to-end trainable hash encoding layer is used for extracting compact but informative codes from images. The proposed many-to-many local feature matching is then directly performed on the hash feature vectors from input images, leading to new state-of-the-art performance on several benchmark datasets.

***Index Terms*—** Content based image retrieval, image feature representation, local feature match, hash codes.

1. INTRODUCTION

Comparing images and finding those with similar content with a given query represents an important image processing application named Content Based Image Retrieval (CBIR). The crucial processing aspects that have to be addressed in CBIR are the image feature extraction and the similarity measure. There are two levels of image representation, corresponding to global and local feature modelling, when employing Convolution Neural Networks (CNN) for CBIR.

Global feature methods [1, 2, 3, 4] extract a compact feature vector for each image following a single forward passing through the network. In recent works several types of attention mechanisms have been proposed to re-weight the convolution feature tensor before compressing it into a compact global feature vector [5, 6, 7, 8, 9] instead of uniformly extracting features from the whole image. As for local feature methods, they could be further categorised into three categories. The first category employs an aggregation module to encode the local features into a compact feature vector [10, 11, 12, 13]. The second type keeps several local features from each image while employing a similarity measure in a many-to-many matching manner [14, 15]. The final category of methods utilises the spatial information of each extracted local feature vector to perform verification during a re-ranking stage [16, 17]. In a way, the first type of local feature methods is similar to the global feature representation approach, as they both eventually lead to single compact global feature vectors as image representations. However, most existing local feature methods tend to extract a large number of local features from the input image. Especially, due to the fixed receptive field of CNNs and the variation in the object size, each local feature may only correspond to a local part of a target object or region. To address this problem, local feature vectors are normally extracted from multi-scale resolutions of the input image. However, the increase in the number of features leads to unbearable processing costs at the online retrieval stage. Consequently, most existing works apply dramatic dimension reduction or binarization [14, 15, 16, 17]) for feature compression. This results in a large number of low-dimensional local features, where each contains relatively limited relevant information.

This research study proposes a new method for extracting a comprehensive and compact image information representation from pre-trained CNNs. Instead of storing huge amounts of low-dimensional local features, we employ clustering on selected local features from the CNN output to build compressed but expressive local feature representations for each image. Moreover, we propose a trainable hash encoding layer, which is optimized based on the idea of the Bi-half Net [18]. After training, the proposed feature extraction pipeline results in compact hash codes with limited information loss. Finally, a corresponding many-to-many similarity criterion is applied to the resulting expressive local features, leading to state of the art results while significantly lower memory resources are used when comparing to other CBIR approaches.

2. METHODOLOGY

In this section, we first describe a simple model structure that generates global feature vectors optimized with image-level

* Dr. A. G. Bors acknowledges the partial support from the EPSRC, UK, project COUSIN (EP/V009591/1)

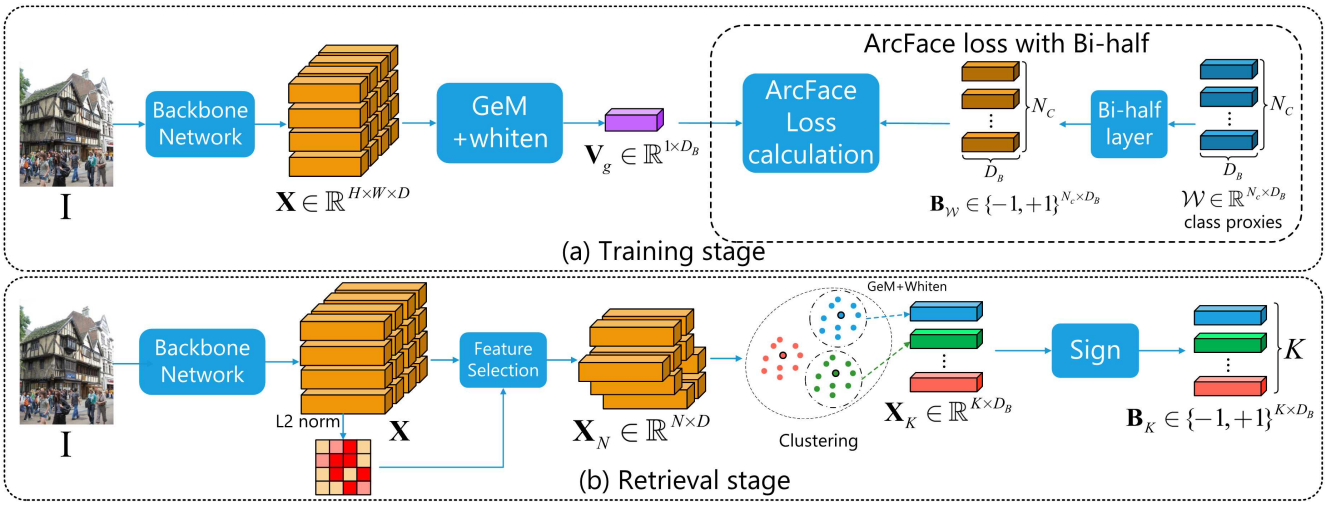

Fig. 1. Illustration of the proposed binary local feature model processing system.

labels at the training stage. Then, we discuss how the local features and hash code generation are optimized during training. After that, we propose a clustering based local feature extraction method along with a many-to-many local feature matching strategy for image matching at the retrieval stage.

2.1. Architecture

We consider ResNet [19] with an output channel count of $D = 2048$ as the backbone network. The feature tensor $\mathbf{X}$ output by the final convolution layer is GeM pooled [4] with a fixed power coefficient of 3. After that, the spatial pooling feature is whitened by a fully connected layer, resulting in the global feature vector $\mathbf{V}_g$ with the dimension of $\mathbf{D}_B$. As shown in Fig. 1 (a), at the training stage, the model is trained with the ArcFace margin loss [17] :

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

where $\widehat{\mathbf{V}}_g$ is the whitened L2 normalized global GeM feature vector for each input training image. $\mathrm{AF}(u, y)$ is the ArcFace-adjusted cosine similarity [17]:

$$\mathrm{AF}(u, y) = \begin{cases} \cos(\arccos(u) + m), & \text{if } y = 1 \\ u, & \text{if } y = 0 \end{cases} \quad (2)$$

while $\widehat{\mathbf{w}}_i$ refers to the trainable L2 normalized classifier weights for class i from the ArcFace weight matrix $\mathcal{W} \in \mathbb{R}^{N_c \times D}$ and N_c is the number of classes in the training dataset. In other words, the ArcFace loss potentially optimizes the cosine similarity not between single image pairs but between each training image and proxies of classes. According to the insight of spatial pooling from [14], optimizing cosine similarity between global spatial pooling features would implicitly optimize the L2 norm of the local descriptor from each entry of the feature tensor $\mathbf{X}$ output by the backbone network. As the Sign function has been widely applied for feature binarization [14, 17, 16], its direct application over real-value features, which are optimized with the real-value loss, could lead to information loss, degrading the whole model's performance. What exact attributes make a good binary code has been discussed in several studies [20, 21, 22]. Recently, the Bi-half Net [18] considers that the information per channel transmitted from the original continuous features to the corresponding binary code is maximized when the distribution of the binary values $\{-1, 1\}$ across all channels is equally balanced. The forward and backward processes for the Bi-half layer given the input feature tensor $\mathbf{F}$, are, [18] :

$$\text{Forward: } \mathbf{B} = \pi_0(\mathbf{F}) = \begin{cases} 1, & \text{top half of sorted } \mathbf{F} \\ -1, & \text{otherwise} \end{cases},$$

$$\text{Backward: } \frac{\partial \mathcal{L}}{\partial \mathbf{F}} = \frac{\partial \mathcal{L}}{\partial \mathbf{B}} + \varphi(\mathbf{F} - \mathbf{B}), \quad (3)$$

where φ is a hyper-parameter equal to the multiplicative inverse of the element count of the feature batch $\mathbf{F}$.

The Bi-half layer in [18] is applied on each batch of features at the training stage which could make the training unstable. Accordingly, in our implementation, we apply the Bi-half layer on the class proxy features $\mathcal{W}$, resulting in binary proxy features $\mathbf{B}_\mathcal{W}$. Then the ArcFace loss from Eq. (1) is re-written as :

$$L(\widehat{\mathbf{V}}_P, \mathbf{y}) = -\log\left(\frac{\exp(\gamma \times \mathrm{AF}(\widehat{\mathbf{V}}_g \widehat{\mathbf{b}}_i^\top, y_i))}{\sum_{j=1}^{N_c} \exp(\gamma \times \mathrm{AF}(\widehat{\mathbf{V}}_g \widehat{\mathbf{b}}_j^\top, y_j))}\right). \quad (4)$$

Intuitively speaking, enforcing these proxy features to have equal binary symbol probabilities could potentially make the binary code of the same class images be optimized towards a consistent goal across all batch steps. This eliminates the distraction caused by the batch size setting or the random image sample shuffle at the training stage.

2.2. Local feature extraction and matching

At the retrieval stage, as shown in Fig. 1 (b), for input Image $\mathbf{I}$, after feeding through the backbone network, L2 norm based feature selection is applied, keeping the top N local features $\mathbf{X}_N \in \mathbb{R}^{N\times D}$ with the highest L2 norm. Then, k-means clustering is employed for extracting a set of representative feature vectors by performing GeM pooling within each cluster. This is followed by whitening and applying the Sign function based binarization to obtain a set of clustered binary codes $\mathbf{B}_K \in \{-1, 1\}^{K\times D_B}$.

Let us consider a pair of images : the query image $\mathbf{I}_q$ and the candidate image $\mathbf{I}_c$ along with corresponding binary features $\mathbf{B}_{q,K} = [\mathbf{b}_{q,i}]$ and $\mathbf{B}_{c,K} = [\mathbf{b}_{c,i}]$ $(i, j = 1, ..., K)$. Their similarity score is defined by:

$$S(\mathbf{I}_q, \mathbf{I}_c) = \frac{\sum_{i=1}^{K}(1 - \min_j \mathrm{Hamm}(\mathbf{b}_{q,i}, \mathbf{b}_{c,j}))}{K}, \quad (5)$$

where $\mathrm{Hamm}(\cdot,\cdot)$ represents the Hamming distance. To further speed up the online retrieval procedure, inverted file indexing [23] is used to eliminate the obvious non-matching images. We use the local features from $\mathbf{X}_N$ to build the visual word codebook. At the feature extraction stage, both query and candidate image local features, $\mathbf{X}_{c,N}$ and $\mathbf{X}_{q,N}$, respectively, are clustered over visual words from the pre-trained codebook and we record the visual word index that each image is assigned to. Then during the retrieval stage, for each query image, we only pick out those candidate database images that would share at least one visual word with the query image to perform the local feature match and assess their similarity. The other candidate images which are not selected are simply set to have zero similarity score with the query image.

3. EXPERIMENTAL RESULTS

Experiment setup. The model uses ResNet101(50) as backbone. The margin $m = 0.15$ and $\gamma = 30$ for the ArcFace loss in Eq. (2) and Eq. (4). To speed up the model convergence, the GeM backbone network is pre-trained on the GLDv2 dataset [24] with ArcFace loss for 50 epochs[1]. The batchsize is set to 128, the initial learning rate is of 0.05 and we consider a cosine learning rate decay strategy [9]. Then, after PCA dimension reduction, class proxies and the fully connected layer are fine-tuned for extra 10 epochs with the Bi-half layer applied and the backbone network frozen. The final output feature dimension $D_B = 512$. We set $N = 500$ for the local feature selection while considering $K = 10$ clusters for k-means clustering. For the inverted file index, we use single scale 60,000 images from the GLDv2 dataset to train the codebook. From each image, 300 local features are picked up and compressed to a dimension of 128 by using PCA. The visual word count of the codebook is set to 65536. In addition, we consider 5 scales $\left\{\frac{1}{2\sqrt{2}}, \frac{1}{2}, \frac{1}{\sqrt{2}}, 1, \sqrt{2}\right\}$ for the multi-scale feature extraction scheme [4]. Local features extracted from different scales are merged together and jointly selected using the L2 norm. ROxf/RPar datasets [25] along with a 1 million images distractor set R1M [25] are used for evaluation.

Local match visualization. In Fig. 2 we visualize the contribution of each location to the image pair similarity score. For comparison, the L2 norm attention given by simple GeM pooling is visualized in the images from the fourth column. As we can observe, the L2 norm tends to uniformly highlight training data's relevant objects, while our local match method emphases the correct regions of interest. In a way, the visualization of local match 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.

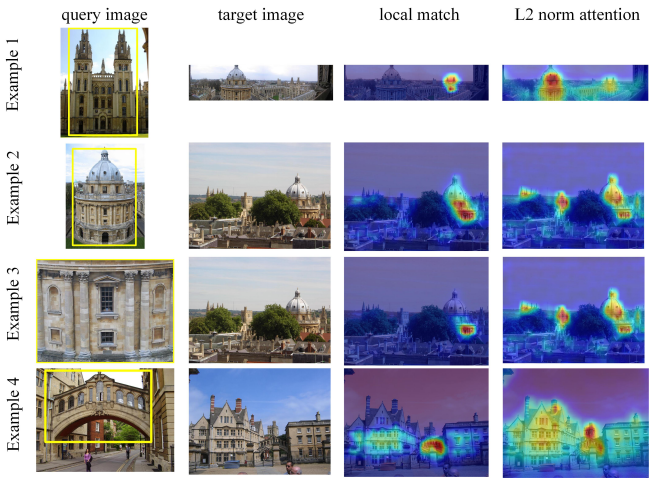

Fig. 2. Local match visualization.

Image retrieval results. Comparative image retrieval results for the proposed method ("LM-BiHalf") and other approaches are provided in Table 1. For fair comparison we re-implemented the GeM [4] and HOW [14] with ArcFace loss on GLDv2 dataset, denoting by †. We can observe that the proposed local match method "LM-BiHalf" leads to great accuracy improvement when compared to the baseline "GeM†". Especially, when considering the ResNet101 as the backbone network, on the *Hard* set of ROxf (RPar), our method reaches the best result 72.0% (83.6%). When considering the 1 million distractor set, our method has the best retrieval results on ROxf+1M and it also gives comparable results to the current SOTA work DOLG on RPar+1M.

[1]In Table 1, the pre-trained GeM network corresponds to "GeM†" . It has the same backbone network as our method "LM-BiHalf" and serves as a baseline for performance comparison.

[2]R101$^-$ represents the ResNet101 [19] without the final convolution block. According to the study from [14], HOW gives better result when discarding the final block and we follow this setting for our re-implementation.

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

Method	*Medium (%)*				*Hard (%)*			
	ROxf	ROxf+1M	RPar	RPar+1M	ROxf	ROxf+1M	RPar	RPar+1M
(A) Local feature								
DELF-D2R-R-ASMK*+SP [26]	76.0	64.0	80.2	59.7	52.4	38.1	58.6	29.4
R50$^-$-HOW-MDA [15]	82.0	68.7	83.3	64.7	62.2	45.3	66.2	38.9
R101$^-$-HOW†[14][2]	83.9	77.9	87.9	76.4	71.3	52.8	76.0	56.4
(B) Global feature								
R101-GeM (GLD) [8]	67.3	49.5	80.6	57.3	44.3	25.7	61.5	29.8
R50-GeM†[4]	79.8	69.0	87.3	73.1	60.4	44.2	74.0	52.0
R101-GeM†[4]	83.0	72.8	90.2	77.6	65.5	49.8	80.7	59.1
R101-DSM [27]	65.3	47.6	77.4	52.8	39.2	23.2	56.2	25.0
R101-SOLAR [8]	69.9	53.5	81.6	59.2	47.9	29.9	64.5	33.4
R101-DELG [17]	76.3	63.7	86.6	70.6	55.6	37.5	72.4	46.9
R101-DELG + SP [17]	81.2	69.1	87.2	71.5	64.0	47.5	72.8	48.7
R50-DOLG [9][3]	81.2	71.4	90.1	79.0	62.6	47.3	79.2	59.8
R101-DOLG [9][3]	82.3	73.6	90.9	**80.4**	64.9	51.6	81.7	**62.9**
(C) Our method								
R50-LM-BiHalf	84.4	72.4	91.0	74.8	67.9	50.7	81.6	53.9
R101-LM-BiHalf	**86.7**	**76.6**	**92.0**	79.3	**72.0**	**54.8**	**83.6**	61.4

Table 1. Image retrieval results on ROxf/RPar datasets (and their extended version +1M distractor set R1M).

Bi-half	*Medium (%)*		*Hard (%)*	
	ROxf	RPar	ROxf	RPar
✗	85.4	90.9	70.4	82.3
✓	86.7	92.0	72.0	83.6

Table 2. Ablation experimental results when considering the Bi-half layer applied on the proxy features $\mathcal{W}$.

4. ABLATION EXPERIMENTS AND DISCUSSION

Bi-half layer impact. We first verify the impact of the Bi-half layer on the model's performance. According to the results from Table 2, after employing the Bi-half layer, the retrieval results globally outperforms those without the Bi-half layer.

Method	Device	Memory (GB) ROxf/RPar+1M	Retrieval time (average ms)
HOW [14]	CPU	14	750
GeM [4]	Tesla GPU	8	250
DOLG [9]	Tesla GPU	2	220
DELG+SP [17]	Tesla GPU	22	383
LM-BiHalf	CPU	0.64	590

Table 3. Computation cost comparison.

Computation and memory costs. With the hyper-parameter setting described in Section 3, the memory cost for one image feature cache is about 0.64KB. It takes around 0.64GB to cache the entire ROxf/RPar dataset with the +1M distractor set. For the online retrieval search on ROxf/RPar with +1M distractor dataset and with the help of inverted file indexing, for one query image, it takes on average 590ms with a CPU. According to the results from Table 3, our method "LM-BiHalf" requires much less memory with a comparable time cost to other CBIR approaches.

5. CONCLUSION

In this paper, we propose extracting few but expressive binary codes as representation for input images. Extracted compact binary features are employed into a many-to-many local matching method for CBIR. Unlike other local matching methods which extract large sets of low-dimensional local features which may require complex matching kernel implementations, the proposed local matching method is based on the L2 norm local feature selection and simple clustering to extract the appropriate number of expressive local features. In addition, the adapted Bi-half binarization layer enriches the information capacity of each feature channel, relieving the information loss problem caused by feature compression. The proposed CBIR methodology enabled by deep feature space clustering and Bi-half binarization achieves new state of art performance on benchmark datasets while having much lower memory requirements than other methods.

6. REFERENCES

[1] Artem Babenko, Anton Slesarev, Alexandr Chigorin, and Victor Lempitsky, "Neural codes for image retrieval," in *Proc. of the European Conference on Computer Vision (ECCV), vol. LNCS 8689*, 2014, pp. 584–599.

[2] Artem Babenko and Victor Lempitsky, "Aggregating local deep features for image retrieval," in *Proc. of the IEEE Int. Conf. on Computer Vision (ICCV)*, 2015, pp. 1269–1277.

[3] Giorgos Tolias, Ronan Sicre, and Hervé Jégou, "Particular object retrieval with integral max-pooling of CNN activations," in *Int. Conf. on Learning Representations (ICLR), arXiv preprint arXiv:1511.05879*, 2016, pp. 1–12.

[4] Filip Radenović, Giorgos Tolias, and Ondřej Chum, "Fine-tuning CNN image retrieval with no human annotation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 7, pp. 1655–1668, 2018.

[5] Albert Gordo, Jon Almazán, Jerome Revaud, and Diane Larlus, "End-to-End learning of deep visual representations for image retrieval," *International Journal of Computer Vision (IJCV)*, vol. 124, no. 2, pp. 237–254, 2017.

[6] Yannis Kalantidis, Clayton Mellina, and Simon Osindero, "Cross-dimensional weighting for aggregated deep convolutional features," in *Proceedings of the European Conference on Computer Vision workshops*, 2016, pp. 685–701.

[7] Xiaomeng Wu, Go Irie, Kaoru Hiramatsu, and Kunio Kashino, "Weighted generalized mean pooling for deep image retrieval," in *Proc. of the IEEE Int. Conf. on Image Processing (ICIP)*, 2018, pp. 495–499.

[8] Tony Ng, Vassileios Balntas, Yurun Tian, and Krystian Mikolajczyk, "SOLAR: second-order loss and attention for image retrieval," in *Proc. of the European Conference on Computer Vision (ECCV), vol. LNCS 12370*, 2020, pp. 253–270.

[9] Min Yang, Dongliang He, Miao Fan, Baorong Shi, Xuetong Xue, Fu Li, Errui Ding, and Jizhou Huang, "DOLG: Single-stage image retrieval with deep orthogonal fusion of local and global features," in *Proc. of the IEEE Int. Conf. on Computer Vision (ICCV)*, 2021, pp. 11772–11781.

[10] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic, "NetVLAD: CNN architecture for weakly supervised place recognition," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 5297–5307.

[11] Eva Mohedano, Kevin McGuinness, Noel E O'Connor, Amaia Salvador, Ferran Marques, and Xavier Giró-i Nieto, "Bags of local convolutional features for scalable instance search," in *Proc. of the ACM Int. Conf. on Multimedia Retrieval (ICMR)*, 2016, pp. 327–331.

[12] Joe Yue-Hei Ng, Fan Yang, and Larry S Davis, "Exploiting local features from deep networks for image retrieval," in *Proc. of IEEE Conf. on Computer Vision and Pattern Recognition workshops (CVPR-w)*, 2015, pp. 685–701.

[13] Zechao Hu and Adrian G. Bors, "Expressive local feature match for image search," in *Proc. of the International Conference on Pattern Recognition (ICPR)*, 2022, pp. 1386–1392.

[14] Giorgos Tolias, Tomas Jenicek, and Ondřej Chum, "Learning and aggregating deep local descriptors for instance-level recognition," in *Proc. of the European Conference on Computer Vision (ECCV), vol. LNCS 12370*, 2020, pp. 460–477.

[15] Hui Wu, Min Wang, Wengang Zhou, and Houqiang Li, "Learning deep local features with multiple dynamic attentions for large-scale image retrieval," in *Proc. of the IEEE International Conference on Computer Vision (ICCV)*, 2021, pp. 11416–11425.

[16] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han, "Large-scale image retrieval with attentive deep local features," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 3456–3465.

[17] Bingyi Cao, André Araujo, and Jack Sim, "Unifying deep local and global features for image search," in *Proc. of the European Conference on Computer Vision (ECCV), vol. LNCS 12370*, 2020, pp. 726–743.

[18] Yunqiang Li and Jan van Gemert, "Deep unsupervised image hashing by maximizing bit entropy," in *Proc. of the AAAI Conf. on Artificial Intelligence*, 2021, vol. 35, pp. 2002–2010.

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

[20] Bin Xu, Jiajun Bu, Yue Lin, Chun Chen, Xiaofei He, and Deng Cai, "Harmonious hashing," in *Proc. Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 2013, pp. 1820–1826.

[21] Yair Weiss, Antonio Torralba, and Rob Fergus, "Spectral hashing," *Advances in Neural Information Processing Systems (NIPS)*, vol. 21, pp. 1753–1760, 2008.

[22] Yudong Chen, Zhihui Lai, Yujuan Ding, Kaiyi Lin, and Wai Keung Wong, "Deep supervised hashing with anchor graph," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019, pp. 9796–9804.

[23] Giorgos Tolias, Yannis Avrithis, and Hervé Jégou, "Image search with selective match kernels: aggregation across single and multiple images," *International Journal of Computer Vision (IJCV)*, vol. 116, no. 3, pp. 247–261, 2016.

[24] Tobias Weyand, André Araujo, Bingyi Cao, and Jack Sim, "Google Landmarks Dataset v2 - A Large-Scale Benchmark for Instance-Level Recognition and Retrieval," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 2575–2584.

[25] Filip Radenovic, Ahmet Iscen, Giorgos Tolias, Yannis Avrithis, and Ondrej Chum, "Revisiting Oxford and Paris: Large-Scale image retrieval benchmarking," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 5706–5715.

[26] Marvin Teichmann, Andre Araujo, Menglong Zhu, and Jack Sim, "Detect-to-retrieve: Efficient regional aggregation for image search," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5109–5118.

[27] Oriane Siméoni, Yannis Avrithis, and Ondrej Chum, "Local features and visual words emerge in activations," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 11651–11660.