Image Retrieval using Multi-scale CNN Features Pooling

Federico Vaccaro, Marco Bertini, Tiberio Uricchio, Alberto Del Bimbo
federico.vaccaro@stud.unifi.it, marco.bertini@unifi.it,
tiberio.uricchio@unifi.it, alberto.delbimbo@unifi.it

MICC - Universit degli Studi di Firenze

April 22, 2020

Abstract

In this paper, we address the problem of image retrieval by learning images representation based on the activations of a Convolutional Neural Network. We present an end-to-end trainable network architecture that exploits a novel multi-scale local pooling based on NetVLAD and a triplet mining procedure based on samples difficulty to obtain an effective image representation. Extensive experiments show that our approach is able to reach state-of-the-art results on three standard datasets.

1 Introduction

Content-based image retrieval (CBIR) has received large attention from computer vision and multimedia scientific communities since the early 1990s. Texture, color and shape visual cues have been used to index images. For about 10 years, approaches based on local invariant features like SIFT and Bag-of-Words representations have obtained state-of-the-art results. Since the inception of Convolutional Neural Networks (CNNs), approaches using either convolutional or fully connected layer activations obtained better results [25] than those that aggregate local manually engineered features. The most recent CNN-based approaches aggregate regional activations, learning image representations in an end-to-end approach [23].

In this paper, we present a novel multi-scale CNN regions pooling that aggregates local features before performing a second aggregation step using NetVLAD. This is used in an end-to-end learning approach in conjunction with a 3-stream Siamese network, to learn optimized image representations. A second contribution of this work is a triplet mining procedure that provides a diverse set of semi-hard and hard triplets, avoiding extremely hard ones that may hinder

learning. The proposed method is evaluated on three standard image retrieval datasets: INRIA Holidays, Oxford5K and Paris6K, obtaining state-of-the-art results.

The paper is organized as follows: discussion of previous works is provided in Sect. 2; description of the proposed method and its two contributions is given in Sect. 3; experiments on three standard CBIR datasets and a thorough comparison with competing state-of-the-art approaches are reported in Sect. 4; finally, conclusions are drawn in Sect. 5.

2 Previous work

Following the introduction of the Bag-of-Visual-Words model in [27], many works have improved aspects such as approximating local descriptors [11], learning improved codebooks [14], improving local features aggregation [19, 12, 4]. However, following the success obtained using CNNs for image classification tasks, CNN-based features have started to be used also for image retrieval tasks. A thorough survey that compares SIFT and CNN-based approaches is provided in [32].

2.1 CNN feature extraction

The most straightforward approach is to use the activations of fully connected or convolutional layers as descriptors, using the networks as feature extractors. AlexNet FC6 has been used in [25], outperforming local features approaches for instance retrieval in several standard datasets. In [3] the performance of different AlexNet layers and the effects of PCA have been evaluated. More recent approaches use max-pooled activations from convolutional layers [24, 2, 33].

CNN features can be aggregated using techniques like Bag-of-Words, applied to local convolutional features as in [15], VLAD, applied to global features as in [31] and to local patches as in [6, 31], or using Fisher Vectors, e.g. applied to localized local feature maps derived from objectness detectors as in [29]. Component-wise max-pooling of CNN features computed on object proposals has been used in [26]. The approach used to compute CNN features in these methods may have an impact on the computational performance: the approaches used in [6, 25] require to compute CNN features on a large number of sub-patches, a problem that is reduced in [29, 31] where object proposals and "dense sampling" from max-pooling of convolutional layers are used. As a result, faster pooling approaches were introduced. Regional maximum activation of convolutions (R-MAC) aggregation [28] consider a set of squared regions at different scales, and collects the maximum response in each channel. These descriptors are sum-pooled to create the final R-MAC descriptor. Hashing of CNN features, either global [5, 16] or local, based on objectness scores [30], have been used to speed-up image retrieval tasks.

2.2 End-to-end approaches

In this class of methods CNN models are fine-tuned on a training set, learning better representations or aggregations of features, and allowing to extract features in an end-to-end manner through a single pass of the model. Typically this results in an improved performance w.r.t. methods based on CNN feature extraction [7, 22].

In [1] has been proposed a layer called NetVLAD, pluggable in any CNN architecture and trainable through back-propagation, that is inspired by the commonly used VLAD representation. This allows to train end-to-end a network, obtaining state-of-the-art results in image retrieval tasks using an aggregation of VGG16 convolutional activations. Simultaneous learning of CNN and Fisher Vector parameters using a Siamese network and contrastive loss has been proposed in [17].

Both the two current state-of-the-art approaches [8, 23] follow an end-to-end approach, one using a three-stream Siamese network with triplet loss and the other using a two-stream Siamese network with contrastive loss.

In [8] an end-to-end learned version of R-MAC descriptor is presented, along with a triplet mining procedure to efficiently train a three streams Siamese Network using triplet loss. In this approach, a region proposal network selects the most relevant regions of the image, where local features are extracted, in three scales of the input images.

In [23] a trainable Generalized-Mean (GeM) pooling layer is proposed, along with learning whitening, for short representations. Two stream Siamese network is trained using contrastive loss. The authors use structure-from-motion information and hard-matching examples for CNN training, and use up to 5 image scales to extract features.

Our proposed method shares similarity with all of these approaches, but in addition to our proposed pooling and triplet mining, it has important subtle differences that increase performance of the resulting system. Differently from [6, 29, 31, 31] our method is fully trainable end-to-end; differently from [31] multiple scales and only one convolutional layer are used; differently from [6] the VLAD aggregation is performed contemporarily at all the scales, and differently from [29] there is no use of region proposals. Differently from [1], our input to the NetVLAD layer is not directly convolutional activations, but the concatenation of two max-pooled sets of activations.

3 The Proposed Method

The idea is to train a CNN network which provides optimized descriptors to perform image retrieval. The proposed method is inspired by the approaches used in [1, 8, 23]; the main differences are: *i)* how the CNN features are collected using two different aggregation steps: the first one through max-pooling operations, i.e. using 2-scales local features, followed by VLAD; *ii)* the triplet mining procedure used to train a three-stream Siamese network, that selects semi-hard

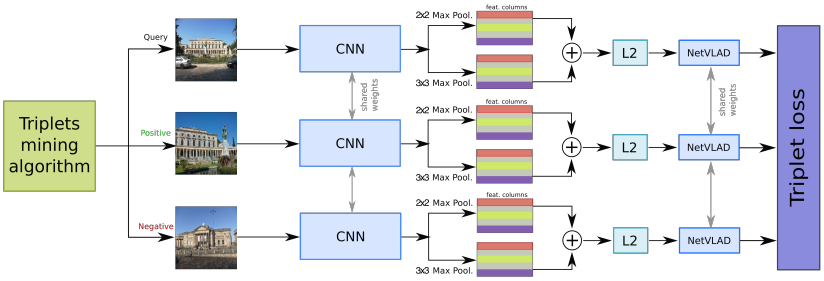

Figure 1: Schema of the proposed method: the three stream Siamese network is used at training time. At test time the query image is fed to the learned network to produce an effective image representation used to query the database.

and hard triplets, avoiding those that could be considered as extremely hard, i.e. whose visual similarity is very low due to minimal overlap, extreme zooming, etc. that may lead to overfitting and loss of generalization [23].

3.1 Pooling of local CNN features

Convolutional features are max-pooled using a 2×2 and 3×3 (both using stride=1) process, so to obtain representations at finer and larger detail. For each location of the two partitions the f activation maps are collected, creating a $1 \times 1 \times f$ "column feature" (as defined in [32]). This process, shown in Fig.2, is akin to dense grid-based sampling of SIFT descriptors [9]. Sets of column features are concatenated, to provide a multi-scale descriptor of the image.

All the local CNN features are then aggregated using a NetVLAD [1] layer. The activations of this layer are used as a descriptor of the content of the image. The NetVLAD layer is initialized with a K-Means clustering[1]. As in [1] for NetVLAD we use $K = 64$, resulting in a 32k-D representation.

The approach can be applied in principle to any CNN network. In the following experiments we have tested VGG16, as it is commonly used in many competing methods and comparisons. An overview of the method is shown in Fig. 4. The figure shows that we have used the penultimate convolutional layer in the 5^{th} block, since initial experiments have shown that using the last layer led to a reduced performance.

3.2 Training and Triplet Mining

In this work we use a ranking loss based on triplets of images; the idea is to learn a descriptor so that the representation of relevant images is closer to the descriptor of the query than that of irrelevant ones. The design of the network is shown in Fig. 1: the weights of the convolutional layers of the CNN network

[1]In the experiments we performed it on MirFLICKR25K dataset `http://press.liacs.nl/mirflickr/mirdownload.html`

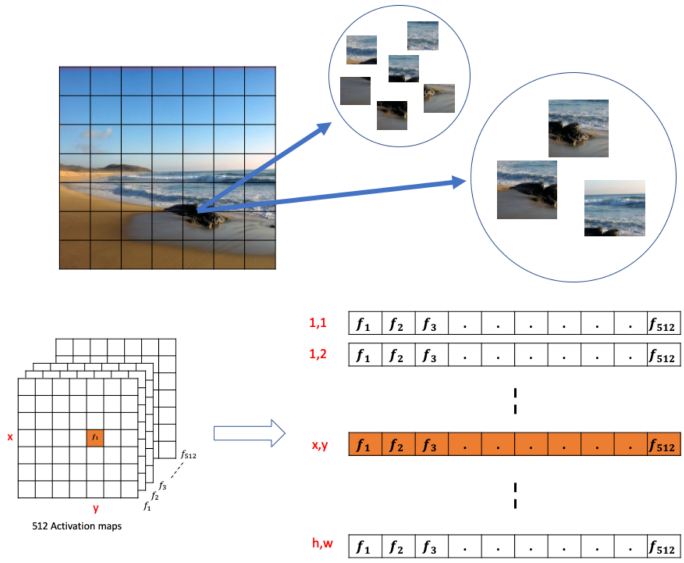

Figure 2: "Column feature" extraction: *top)* max-pooling with different scales, *bottom)* activation maps collection as column features: this is performed at each pooling scale.

and the NetVLAD layer are shared between the streams, since their size is independent of the size and aspect ratio of the images.

At training time the network is provided with image triplets. Given a query image Q with descriptor q, a positive image P with descriptor p, a negative image N with descriptor n, a distance $d()$ (squared Euclidean distance) and a scalar α that controls the margin, the triplet loss used is $L = max(\alpha + d(q,p) - d(q,n), 0)$. α is set to 0.1 as in [1].

An issue that may arise with this approach is due to the sampling of the triplets: e.g. a random approach may select triplets that do not incur in any loss and thus do not improve the model. We note that triplets may have different impact on the learning depending on the difficulty they pose. Some examples may be well separable if they are from different objects and may be easily learnt. In the contrary, similar but different objects may be challenging to be separated correctly. We may classify triplets as:
easy triplets: $d(q,p) < d(q,n) + \alpha < d(q,n)$ do not really improve the model;
semi-hard triplets: $d(q,p) < d(q,n)$ but $d(q,p) + \alpha > d(q,n)$ - these are more useful than easy triplets but may not add enough information;
hard triplets: $d(q,n) < d(q,p)$ - they produce a high loss.

The algorithm shown in Alg. 1 generates semi-hard and hard triplets (with a 0.5 probability, line 14) with the following logic:
case A: searches the index j for the first negative w.r.t. query. If the index is not the first then the index of the positive sample is $j - 1$ (line 15), resulting in

a semi-hard triplet.
case B: otherwise, searches the index of the first positive after the first negative, resulting in a hard triplet (lines 18-19).
case C: this deals with extremely hard triplets, e.g. due to strong changes in visual content like zoom or very different point of views of the same scene (see Fig. 3), so that positives are very far from the query, i.e. the index of the first positive is farther than a threshold t (line 21). In this case triplets are discarded, since they may lead to overfitting or poor generalization.

The number of classes k used in Alg. 1 is 512, the $mining\_batch\_size$ is 2048. The procedure select the triplets so that they belong to different classes (line 28), yielding on average 250 triplets and returned as mini batches composed by 24.

Algorithm 1 Triplet mining

```
1: procedure TRIPLET MINING(mining_batch_size, k, landmarks, t)
2:     Pick k random landmarks
3:     X, y ← pick mining_batch_size random images from
4:                     the selected landmarks and their labels
5:     features ← model.extract_features(X)
6:     triplets[] ← new list()
7:     for i ∈ [1, mining_batch_size] do
8:         feature = features[i]; query_label = y[i]
9:         indices[] ← Compute k-NN of feature
10:        q ← i; p ← null; n ← null
11:        for j ∈ [1, mining_batch_size] do
12:            if label[j] ≠ query_label and n = null then
13:                n ← j
14:                if j > 2 with Probability 0.5 then
15:                    p ← j − 1
16:                    break
17:                end if
18:            else if label[j] = query_label and n ≠ null then
19:                p ← j
20:            end if
21:            if p ≠ null and n ≠ null and p − n < t then
22:                triplet ← (X[q], X[p], X[n])
23:                triplets.append(triplet)
24:                break
25:            end if
26:        end for
27:    end for
28:    Keep just one triplet per class
29:    return triplets
30: end procedure
```

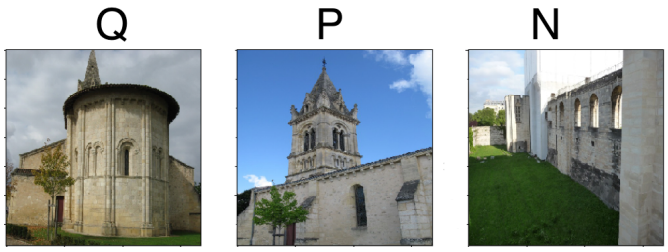

Figure 3: Example of discarded extremely hard triplet.

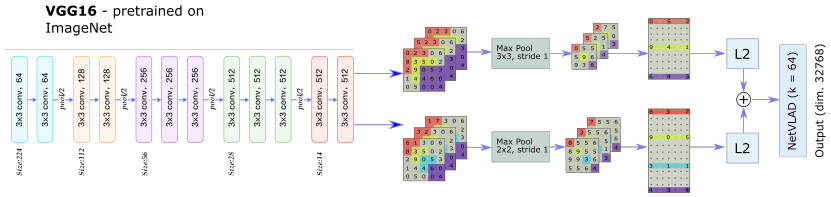

Figure 4: Overview of the proposed architecture, using VLAD aggregation of local multiscale max-pooling CNN features. VGG16 pre-trained on ImageNet is used as backbone.

Training of the network is performed using Google Landmark V2 dataset[2]. In particular we use the train split of the "cleaned" version[3] presented in [18], that contains 1,580,470 images and 81,313 labels. The mining process is performed every 8 iterations, to account for the fact that descriptors may change greatly, especially during the initial training. The network has been trained using the Adam [13] optimizer, with a starting learning rate of 10^{-5}, decreased to 10^{-6} after few epochs. The training images have been resized to resolution 336×336, regardless to the original aspect-ratio.

4 Experiments

For the convolutional part of the network we evaluate a popular architecture, commonly used in other competing approaches, i.e. VGG16, but other architectures can be plugged, as ResNet, etc.

4.1 Datasets and Metrics

We test our approach on three standard datasets: *i)* Oxford5k dataset [20], *ii)* Paris6k dataset [21], and *iii)* INRIA Holidays dataset [10]; the standard evaluation protocol for these datasets is mean average precision (mAP). To be comparable with most CNN-based methods evaluations we manually correct the

[2] `https://github.com/cvdfoundation/google-landmark`
[3] `https://www.kaggle.com/confirm/cleaned-subsets-of-google-landmarks-v2`

orientation of the images on the Holidays dataset, evaluating on the corrected images.

4.2 Multi-scale Pooling and Image

In the experiments reported in Tab. 1, we evaluate the effects of the first contribution of this work, i.e. using two max-pooling to obtain multi-scale features before the NetVLAD layer. Results show that using both 2×2 and 3×3 pooling improve the performance. A single resolution image is used as input. It must be noted that all the results improve upon the standard NetVLAD pooling [1] reported in Tab. 3, showing the benefit of the two-step local CNN features aggregation.

Table 1: Effects of multi-scale pooling (mAP).

Pooling	Holidays	Oxford5k	Paris6k
3×3	91.6	81.0	87.3
2×2	88.8	79.6	84.9
Both	**92.3**	**83.0**	**88.4**

Different resolutions may provide different clues regarding the appearance of objects in the scene. Hence, we extract and combine features at different resolutions, improving the performance of the multi-scale pooling. In the experiments reported in Tab. 2 we evaluate using different image resolutions at test time, evaluating the best combination on multiple datasets. Images are resized to 224×224, 336×336, 504×504 and 768×768 pixels, regardless of aspect ratio. The multi-resolution column reports the sizes used. In all these experiments multi-resolution pooling is used. Results show that image multi-resolution improves the performance. It is interesting to note that even the worst performing combination, i.e. without multi-resolution, the proposed method has better results than competing state-of-the-art approaches (see Tab. 3).

Table 2: Effects of using multi-scale images, tested on INRIA Holidays (mAP).

Holidays	Oxford5k	Paris6k	Image resolutions
92.3	83.0	88.4	336
93.2	83.4	88.9	336 + 504
93.2	**83.8**	**89.3**	224 + 336 + 504
93.2	83.6	89.3	224 + 336 + 504 + 768

4.3 Comparison with SOTA

In these experiments we evaluate the proposed method with current state-of-the art methods on all three datasets. Results are reported in Tab. 3; all the methods

reported in the table use VGG networks. Results of our method have been obtained using multi-resolution (224 + 336 + 504) and power normalization.

Table 3: Comparison with state-of-the-art methods (mAP).

Method	Holidays	Oxford5k	Paris6k
Our method	**93.2**	83.8	**89.3**
GeM [23]	89.5	**87.9**	87.7
R-MAC [8]	89.1	83.1	87.1
NetVLAD [1]	87.5	71.6	79.7
Fisher-Vector [17]	-	81.5	82.4
BoW-CNN [15]	-	73.9	82.0
R-MAC [28]	86.9	66.9	83.0

5 Conclusions

We presented a novel multi-scale local CNN features pooling that, by exploiting end-to-end learning on a Siamese network, is able to learn an effective images representation. This is also thanks to a novel triplet mining procedure that is able to diversify triplets based on their difficulty and focus the learning on the most significative ones. Results on three standard datasets shows that the proposed approach obtains state-of-the-art results for the task of image retrieval.
Acknowledgments. We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X Pascal GPU used for this research.

References

[1] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proc. of CVPR*, 2016.

[2] H. Azizpour, A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson. From generic to specific deep representations for visual recognition. In *Proc. of CVPR Workshops*, June 2015.

[3] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. In *Proc. of ECCV*, 2014.

[4] J. Delhumeau, P.-H. Gosselin, H. Jégou, and P. Pérez. Revisiting the vlad image representation. In *Proc. of ACM MM*, 2013.

[5] S. Ercoli, M. Bertini, and A. Del Bimbo. Compact hash codes for efficient visual descriptors retrieval in large scale databases. *IEEE Transactions on Multimedia (TMM)*, 19(11):2521–2532, Nov. 2017.

[6] Y. Gong, L. Wang, R. Guo, and S. Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. In *Proc. of ECCV*, 2014.

[7] A. Gordo, J. Almazán, J. Revaud, and D. Larlus. Deep image retrieval: Learning global representations for image search. In *Proc. of ECCV*, 2016.

[8] A. Gordo, J. Almazan, J. Revaud, and D. Larlus. End-to-end learning of deep visual representations for image retrieval. *International Journal of Computer Vision*, 124(2):237–254, 2017.

[9] A. Iscen, G. Tolias, P.-H. Gosselin, and H. Jégou. A comparison of dense region detectors for image search and fine-grained classification. *IEEE Transactions on Image Processing*, 24(8):2369–2381, 2015.

[10] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In *Proc. of ECCV*, 2008.

[11] H. Jégou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. *International Journal of Computer Vision*, 87(3):316–336, 2010.

[12] H. Jégou, F. Perronnin, M. Douze, J. Sánchez, P. Pérez, and C. Schmid. Aggregating local image descriptors into compact codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1704–1716, Sep. 2012.

[13] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proc. of ICLR*, 2014.

[14] A. Mikulik, M. Perdoch, O. Chum, and J. Matas. Learning vocabularies over a fine quantization. *International Journal of Computer Vision*, 103(1):163–175, 2013.

[15] E. Mohedano, K. McGuinness, N. E. O'Connor, A. Salvador, F. Marques, and X. Giro-i Nieto. Bags of local convolutional features for scalable instance search. In *Proc. of ACM ICMR*, 2016.

[16] O. Morère, J. Lin, A. Veillard, L.-Y. Duan, V. Chandrasekhar, and T. Poggio. Nested invariance pooling and rbm hashing for image instance retrieval. In *Proc. of ACM ICMR*, 2017.

[17] E.-J. Ong, S. Husain, and M. Bober. Siamese network of deep Fisher-vector descriptors for image retrieval. *arXiv preprint arXiv:1702.00338*, 2017.

[18] K. Ozaki and S. Yokoo. Large-scale landmark retrieval/recognition under a noisy and diverse dataset. *arXiv preprint arXiv:1906.04087*, 2019.

[19] F. Perronnin, J. Sánchez, and T. Mensink. Improving the Fisher kernel for large-scale image classification. In *Proc. of ECCV*, 2010.

[20] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proc. of CVPR*, 2007.

[21] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *Proc. of CVPR*, June 2008.

[22] F. Radenović, G. Tolias, and O. Chum. Cnn image retrieval learns from bow: Unsupervised fine-tuning with hard examples. In *Proc. of ECCV*, 2016.

[23] F. Radenović, G. Tolias, and O. Chum. Fine-tuning cnn image retrieval with no human annotation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(7):1655–1668, 2018.

[24] A. Razavian, J. Sullivan, A. Maki, and S. Carlsson. A baseline for visual instance retrieval with deep convolutional networks. *ITE Transactions on Media Technology and Applications*, 4, 12 2014.

[25] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: An astounding baseline for visual recognition. In *Proc. of CVPR Workshop of DeepVision*, 2014.

[26] K. Reddy Mopuri and R. Venkatesh Babu. Object level deep feature pooling for compact image representation. In *Proc. of CVPR Workshops*, June 2015.

[27] J. Sivic and A. Zisserman. Video google: a text retrieval approach to object matching in videos. In *Proc. of ICCV*, Oct 2003.

[28] G. Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of cnn activations. In *Proc. of ICLR*, 2016.

[29] T. Uricchio, M. Bertini, L. Seidenari, and A. Del Bimbo. Fisher encoded convolutional Bag-of-Windows for efficient image retrieval and social image tagging. In *Proc. of ICCV International Workshop on Web-Scale Vision and Social Media (VSM)*, 2015.

[30] L. Xie, R. Hong, B. Zhang, and Q. Tian. Image classification and retrieval are one. In *Proc. of ACM ICMR*, 2015.

[31] J. Yue-Hei Ng, F. Yang, and L. S. Davis. Exploiting local features from deep networks for image retrieval. In *Proc. of CVPR Workshops*, 2015.

[32] L. Zheng, Y. Yang, and Q. Tian. Sift meets cnn: A decade survey of instance retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(5):1224–1244, 2017.

[33] L. Zheng, Y. Zhao, S. Wang, J. Wang, and Q. Tian. Good practice in cnn feature transfer. *arXiv preprint arXiv:1604.00133*, 2016.