HYBRID MULTI-LAYER DEEP CNN/AGGREGATOR FEATURE FOR IMAGE CLASSIFICATION

Praveen Kulkarni[1], *Joaquin Zepeda*[1], *Frederic Jurie*[2], *Patrick Perez*[1] *and Louis Chevallier*[1]

[1]Technicolor 975 avenue des Champs Blancs,CS 17616, 35576 Cesson Sévigné, France
[2]University of Caen Basse-Normandie, CNRS UMR 6072, ENSICAEN, France

ABSTRACT

Deep Convolutional Neural Networks (DCNN) have established a remarkable performance benchmark in the field of image classification, displacing classical approaches based on hand-tailored aggregations of local descriptors. Yet DCNNs impose high computational burdens both at training and at testing time, and training them requires collecting and annotating large amounts of training data. Supervised adaptation methods have been proposed in the literature that partially re-learn a transferred DCNN structure from a new target dataset. Yet these require expensive bounding-box annotations and are still computationally expensive to learn. In this paper, we address these shortcomings of DCNN adaptation schemes by proposing a hybrid approach that combines conventional, unsupervised aggregators such as Bag-of-Words (BoW), with the DCNN pipeline by treating the output of intermediate layers as densely extracted local descriptors.

We test a variant of our approach that uses only intermediate DCNN layers on the standard PASCAL VOC 2007 dataset and show performance significantly higher than the standard BoW model and comparable to Fisher vector aggregation but with a feature that is 150 times smaller. A second variant of our approach that includes the fully connected DCNN layers significantly outperforms Fisher vector schemes and performs comparably to DCNN approaches adapted to Pascal VOC 2007, yet at only a small fraction of the training and testing cost.

Index Terms— Deep Convolutional Neural Networks, Bag-of-Words, Fisher Vector aggregator

1. INTRODUCTION

In this paper we propose a new hybrid image feature for image classification obtained from a mix of the classical image feature extraction pipeline and the more recent and very successful Deep Convolutional Neural Network (DCNN) pipeline.

The classical image feature extraction pipeline consist of three major steps: 1) Extracting local descriptors such as SIFT [1] from the image; 2) mapping these descriptors to a higher dimensional space; 3) and sum or max-pooling the resulting vectors to form a fixed-dimensional image feature representation. Examples of methods corresponding to this classical approach include Bag-of-Words (BoW) [2], Fisher Vector (FV) [3], Locality-constrained Linear Encoding [4], Kernel codebooks [5], super-vector encoding [6] and VLAD [7]. We refer to these type of image feature extraction schemes as *aggregators* given that they aggregate local descriptors into a fixed dimensional representation. Generally these approaches require computationally inexpensive unsupervised models of the local descriptor distribution, and the resulting image features can be used to learn likewise inexpensive linear classifiers using SVMs.

The novel DCNN pipeline of [8] has drastically pushed the performance limits of image classification. DCNNs consist of multiple interconnected layers including spatial convolution layers, half-wave rectification layers, spatial pooling layers, normalization layers, and fully connected layers. While this method attains outstanding classification performance, it also suffers from large testing complexity, particularly due to the first fully connected layer, as well as large training complexity, since all the coefficients in the pipeline are learned in a supervised manner and require lots of training images. To address this latter issue, [9] proposed to use DCNN models pre-trained on the Imagenet dataset (consisting of many million images) and then transfer all but the last layer of this pre-trained DCNN to a new target dataset, where two new adaptation layers are learned. This reduces training time and the amount of required training data, but the training data needs to be annotated with bounding box information. The fact that the method works on a per-patch basis further increases the testing complexity relative to standard DCNNs.

Several approaches exist that, like ours, attempt to bridge the classical approach and the DCNN approach using hybrid mixes. Inspired by the popularity of DCNNs, Simonyan *et al.* [10] proposed to incorporate the deep aspect of DCNNs into traditional SIFT/FV schemes by stacking multiple layers of FV aggregators, with each layer operating on successively coarser overlapping spatial cells. Sydorov *et al.* [11] instead proposed viewing the standard FV aggregator as a deep architecture, substituting the unsupervised GMM parameters of the FV aggregator by supervised versions.

While these methods adopted only the deep aspect of DCNNs, our goal is to combine the advantages of both approaches (DCNNs and classical aggregators) using hybrid mixes of both pipelines. We do this by treating the output of the pre-trained intermediate layers of the DCNN architecture as local image descriptors, which we aggregate using standard aggregators such as BoW or FV. There is no need to carry out costly tuning of the DCNN adaptation layers [9] to the target dataset, as both BoW and FV rely on unsupervised learning. The closest related method in the literature is that of Gong *et al.* [12], who propose using the output of the previous-to-last fully connected layer as a local descriptor, computing this descriptor on multi-scale dense patches subsequently aggregated using VLAD on a per-scale basis. This approach is very complex because, contrary to our approach, one needs to compute the full DCNN pipeline not only on the original image but also on a large number of multi-scale patches and further apply two levels of PCA dimensionality reduction.

The remainder of this paper is organized as follows: In Section 2, we describe the two classical aggregators (BoW and FV) that we use in our experiments, as well as the DCNN architecture. In Sec-

This work was partially supported by the FP7 European integrated project AXES.

tion 3, we describe our hybrid image feature extraction pipeline. We evaluate our proposed method in Section 4 and provide concluding remarks in Section 5.

2. BACKGROUND

In this section we present an overview of two classical local descriptor aggregation methods: the BoW aggregator [13, 14, 15] and the FV aggregator [16]. Up until recently, such aggregation schemes together with SVM classifiers were the reference in image classification [17]. We then present an overview of the new state-of-the art DCNN image classification pipeline [8].

2.1. Image Classification using Local Descriptor Aggregators

The classical image classification procedure consists of first mapping images to a fixed-dimensional image feature space where linear classifiers are computed using SVMs. The image feature construction process operates by aggregating the local descriptors extracted from the image in question, $\mathbf{f} : \{\mathbf{x}_k \in \mathbb{R}^d\}_k \mapsto \mathbb{R}^D$, where the $\mathbf{x}^k$ are the local descriptors of the image.

The Bag-of-Words (BoW) aggregator offers one such way to map local descriptors to image features. A training set of local descriptors $\mathcal{T}$ from a representative set of images is first used to build a codebook $\mathbf{C} = [\mathbf{c}_j]_j$ using K-means. Letting $\mathcal{C}_j$ denote the Voronoi cell for codeword $\mathbf{c}_j$, the BoW aggregated image feature is the relative frequency of occurrence of local descriptors in the Voronoi cells:

$$\mathbf{f} = \left[\#\left(\{\mathbf{x}_k, \mathbf{x}_k \in \mathcal{C}_j\}_k\right)/\#\left(\{\mathbf{x}_k\}_k\right)\right]_j, \tag{1}$$

where we let $\#$ denote set cardinality. The BoW encoder offers an intuitive image feature and enjoys a low computational cost that can be important in user-in-the-loop applications such as [18].

A more recent image feature, the Fisher vector, offers an important gain in image classification performance [17]. The Fisher encoder requires that a training set of local descriptors $\mathcal{T}$ be used to learn a GMM model $\mathcal{G} = \{\beta_k, \mathbf{\Sigma}_k, \mathbf{c}_k\}_k$ with k-th mixture component having prior weight β_k, covariance matrix (assumed diagonal) $\mathbf{\Sigma}_k$ and mean vector $\mathbf{c}_k$. The first order Fisher vector for a given image can then be computed as follows:

$$\mathbf{f} = \left[\frac{1}{M}\sum_{k=1}^{M}\frac{p(j|\mathbf{x}_k)}{\sqrt{\beta_j}}\mathbf{\Sigma}_j^{-1}\left(\mathbf{x}_k - \mathbf{c}_j\right)\right]_j. \tag{2}$$

Both the BoW and Fisher aggregators are built from unsupervised models for the distribution of local descriptors, with supervision coming into play only at the classifier learning stage. Deep CNNs instead construct a fully supervised image-to-classification score pipeline.

2.2. Deep Convolutional Neural Networks (DCNNs)

Deep Convolutional Neural Networks have established an overwhelming presence in image classification starting with the 2012 ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [8]. The performance gap of DCNNs relative to the second entry in that year's competition (and relative to SIFT-based Fisher aggregation schemes [19]) is in excess of 10 percentage points in absolute improvement of top-5 error rate.

In Fig. 1 we illustrate the deep DCNN processing pipeline of [8]. It consists of convolutional layers, max-pooling layers, normalization layers and fully connected layers. At any given layer l, the

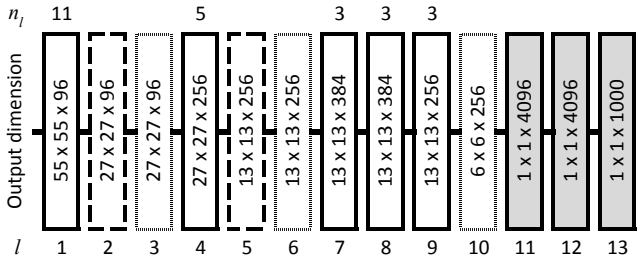

Fig. 1. Architecture of the Deep-CNN pipeline of [8] trained on ImageNet 2012 and used in this paper. Each layer, represented by a box, is labeled with the size $R_l \times C_l \times K_l$ of its output in (3). The K_l kernels at layer l have dimension $n_l \times n_l \times K_{l-1}$. The layer index l (respectively, kernel spatial dimension n_l) is indicated below (above) the box for each layer. The input image is assumed normalized to size $224 \times 224 \times 3$, and $4\times$ downsampling is applied during the first layer. *Dark-lined boxes:* convolutional layers; *dash-lined boxes:* normalization layers; *light-lined boxes:* max-pooling layers; *grayed-in boxes:* fully-connected layers.

layer's output data is an $R_l \times C_l \times K_l$ array

$$[\mathbf{x}_{ij}^l \in \mathbb{R}^{K_l}]_{i=1,\ldots,R_l;\, j=1,\ldots,C_l}, \tag{3}$$

that is the input to the next layer, with the input to layer $l = 1$ being an RGB image of size $R_0 \times C_0$ and $K_0 = 3$ color channels.

The *convolutional layers* ($l = 1, 4, 7-9$) first compute the spatial convolution of the input with K_l kernels of size $n_l \times n_l \times K_{l-1}$ and then apply entry-wise Rectified Linear Units (ReLUs) $\max(0, z)$. The *normalization layers* ($l = 2, 5$) normalize each $\mathbf{x} \in \{\mathbf{x}_{ij}^{l-1}\}_{ij}$ at the input using what can be seen as a generalization of the l_2 norm consisting of dividing each entry x_m of $\mathbf{x}$ by $(2 + 10^{-4}\sum_{n\in\mathcal{I}_m} x_n^2)^{0.75}$. The summation indices $\mathcal{I}_m$ are taken to be the m-th sliding window over the indices of all entries. The *max-pooling layers* ($l = 3, 6, 10$) carry out per-kernel spatial max-pooling by taking the maximum value from each spatial bin of size 3×3 spaced every 2 pixels.

The *fully connected layers* ($l = 11 - 13$) can be seen as convolutional layers with kernels having the same size as the layer's input data. The last layer ($l = 13$) uses a softmax non-linearity instead of the ReLU non-linearity used in other layers and acts as a multi-class classifier, having as many outputs as there are classes targeted by the system.

2.3. Transfer learning using DCNNs

The architecture in Fig. 1 contains more than 60 million parameters and training it can be a daunting task requiring expensive hardware, large annotated training sets (ImageNet 2012 contains 15 million images and 22,000 classes) and training strategies including memory management schemes, data augmentation and specialized regularization methods. Moreover, extending the architecture to new classes would potentially require re-training the entire structure, as the full architecture is learned for a specific set of target classes.

To address this last difficulty, Oquab *et al.* [9] use transfer learning to apply the architecture in Fig. 1 to new classes while incurring reduced training overhead. Their approach consists of substituting only the last fully-connected classification layer by two learned adaptation layers, a fully-connected ReLU layer with 4096 neurons followed by a fully-connected softmax classification layer with as

many neurons as target classes. The first 12 layers are transferred from the net in Fig. 1 (learned from ImageNet 2012 data), and only the new adaptation layers are learned using training data for the new set of target classes (*e.g.*, those of the Pascal VOC 2007 test bench).

While their approach reduces the training overhead and required training set size, training the adaptation layers still requires non trivial complexity as these contain a large number of parameters (more than 16 million). To obtain an adequately large training set from Pascal VOC 2007 data, they derive a patch-based training set, labeling every patch according to its intersection with the provided object bounding boxes. Their approach thus operates on a per-patch classification basis, and the overall class score is obtained by summing this per-patch scores over the entire image for each class. This brings the important benefit of also providing the object localization, but it requires laborious bounding-box annotations on the training set and costly training of millions of parameters.

3. A HYBRID DCNN/AGGREGATOR FEATURE

Inspired by the transfer learning approach of [9], in this section we propose a new hybrid feature that combines parts of the DCNN architecture in Fig. 1 trained on ImageNet 2012 with the unsupervised BoW or Fisher local descriptor aggregation schemes in (1) and (2). The resulting feature is used with one-vs-all linear SVM classifiers and hence new classes can be added with little training overhead and without the need for costly object bounding box annotations.

3.1. Per-layer aggregation of DCNN local descriptors

Our hybrid scheme is based on the observation that the vectors $\mathbf{x}_{ij}^l$ in (3) comprising the output of layers $l = 1, \ldots, 10$ in Fig. 1 (*i.e.*, all layers except fully-connected layers) can be treated as densely extracted local descriptors. We will hence build one aggregated feature $\mathbf{f}_l$ for each layer l (or a subset of layers $l \in \mathcal{L}$) and concatenate all the resulting aggregated layer features to form a single image feature

$$\mathbf{f} = [\mathbf{f}_l^T]_{l\in\mathcal{L}}^T. \tag{4}$$

Using only a subset of layers $\mathcal{L} \subseteq \{1, \ldots, 10\}$ allows us to control training, testing and storage complexity and further serves as a means of regularization.

3.2. Training per-layer aggregators

In order to train the per-layer aggregators adapted to the DCNN layers, we take each image from a representative set of training images and extract from it all vectors $\mathbf{x}_{ij}^l$ for $l = 1, \ldots, 10$. We then group all the resulting local descriptors $\mathbf{x}_{ij}^l$ for each layer l to form a training set $\mathcal{T}^l$ for the l-th layer. Each training set $\mathcal{T}^l$ of local descriptors is then used to train a codebook $\mathbf{C}^l$ for layer l using K-means when using BoW aggregators. Likewise, a GMM model $\mathcal{G}^l$ is learned for the l-th layer when using Fisher aggregators.

3.3. Extensions based on classic approaches

Our proposed approach shares similarities with several existing approaches and we now discuss these and related extensions.

One first observation is that the spatial support (relative to the original image) used to compute the $\mathbf{x}_{ij}^l$ is of size 11 (in each spatial dimension) for the first layer and grows by $4 \times 2 \cdot (n_a - 1)$ for each convolutional layer $1 < a \leq l$, yielding possible supports of size $11, 43, 59$ and 75. Dense approaches likewise compute local descriptors from supports of varying size $(16, 24, 32, 40)$ by means of multi-resolution spatial grids [17], but all descriptors for all supports are pooled together (for the benefit of scale invariance) and used to form a single aggregated image feature. A similar pooling approach could be used for DCNN local descriptors $\mathbf{x}_{ij}^l \in \mathbb{R}^{K_l}$ by first mapping all layers to a common dimensionality via, *e.g.*, PCA or discriminative dimensionality reduction.

The layer feature concatenation scheme (4) that we use instead is reminiscent of spatial pyramid matching [14, 15], where one feature $\mathbf{g}_c$ is computed for each spatial cell $c = 1, \ldots, 8$ and these are subsequently concatenated. Our concatenated image features $\mathbf{f}_l$ are instead computed from high-dimensional filtered versions of the image, and indeed this approach can be combined with SPM to produce per-spatial-cell layer features $\mathbf{f}_{lc}$.

Other standard successful approaches can also be combined with our proposed hybrid DCNN/aggregator features, including power normalization of the $\mathbf{x}_{ij}^l$ [20], application of an explicit Hellinger kernel-map to our hybrid feature [17] and late fusion with other feature channels. Alternate aggregation schemes such as VLAD or triangulation embedding [7, 21] can also be used, but we chose BoW for its low computational cost and Fisher given that is the best performing aggregator in classification.

4. RESULTS

In this section we validate our proposed hybrid DCNN/aggregator feature using the publicly available Pascal VOC 2007 dataset [22]. This dataset consists of 9163 images representing 20 visual categories and split into training, validation and test sets. We use the standard mean Average Precision (mAP) measure computed over the test set as a performance metric.

4.1. Impact of layer subset $\mathcal{L}$

In Fig. 2 we evaluate the impact on performance of the layer subset $\mathcal{L}$ in (4) used to build hybrid features. We consider three strategies for selecting $\mathcal{L}$: using a single layer, $\mathcal{L} = \{L\}$, using the first L layers, $\mathcal{L} = \{1, \ldots, L\}$, and using the last L layers, $\mathcal{L} = \{10, 9, \ldots, 10 - L + 1\}$. As seen in Fig. 2, the results for the single-layer strategy indicate that layers further down the pipeline are more informative (although the curve is not monotonic). Indeed the best strategy overall consists of using the last 5 layers (and using only 3 layers results in marginal performance decrease). The resulting hybrid feature performs substantially better than BoW+SPM with $4,000$ codewords and performs similar to FV+SPM with 256 mixture components [23], despite being 150 times smaller.

4.2. Impact of codebook size

In Fig. 3 we evaluate the impact on performance of varying the codebook size when using hybrid DCNN/BoW features built from the last 5 layers. A codebook of size 500 yields the best performance. And even with a codebook size of 30, which amounts to a feature vector size of 150, our method outperforms BoW + SPM.

4.3. Comparison to other approaches

In Table 2 we compare our results with some of the best results reported in the literature. We include results for hybrid features built using FV aggregators with 64 mixture components. Despite the established superiority of FV aggregation over BoW aggregation, the FV-based hybrid features perform poorly relative to BoW-based hybrid features. We believe that this is due to the small number of local

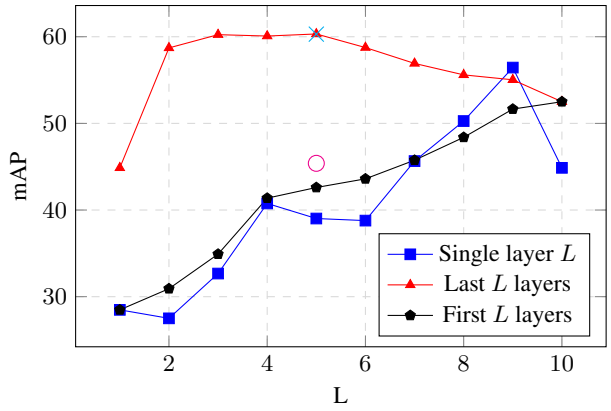

Fig. 2. The mAP is plotted for hybrid features built using a single layer, the last L layers and the first L layers (excluding fully connected layers 11-13), for codebook size 500. Baseline results for BoW and FV are displayed using ○ and × markers.

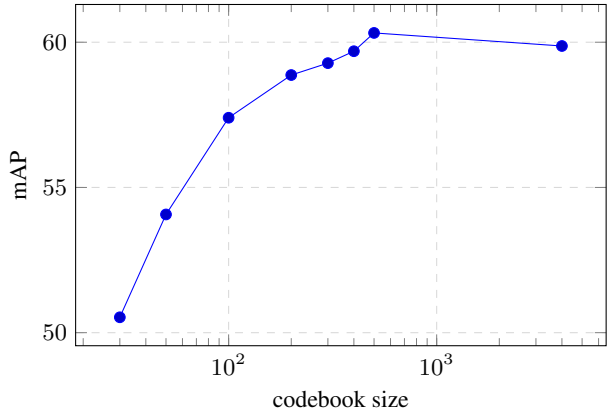

Fig. 3. The mAP vs the codebook size in log scale when using last L=5 layers.

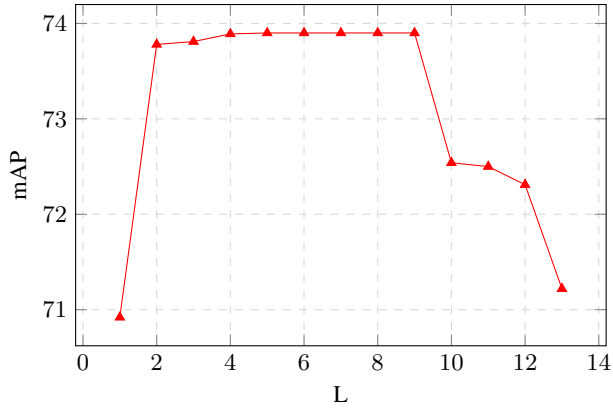

Fig. 4. Using last L layers from Fig. 1. Here we include the fully connected layers 11-13.

descriptors in DCNN layers, as this makes the vector-averaging process in (2) statistically noisy.

method	Training time+resource
PRE1000C [9]	≈ 1 day (GeForce GTX Titan GPU)
Hybrid DCNN/BoW, N=500	≈ 1hr + 5min (8 core CPU)

Table 1. Table illustrating training time for 500 codebook size and when using the last 5 layers. Training times are for the unsupervised learning part with and without supervised learning of linear SVM classifiers for all Pascal VOC 2007 classes. This is compared to the training time taken by the method [9].

The best performing system in Table 2 is PRE1000C [9]. Their approach consists of substituting layer 13 in Fig. 1 by two adaptation layers trained on Pascal VOC. As is the case for DCNN pipelines, this training procedure is time consuming and requires expensive GPU cards, as illustrated in Table 1. Furthermore, at testing time, their approach requires applying the full 13-layer DCNN pipeline to each of 500 patches from an image, increasing testing complexity considerably. Our approach requires a single DCNN pipeline pass over the non fully-connected layers, resulting in dramatically lower testing time, as the DCNN complexity is largely concentrated in the first fully-connected layer.

The same complexity problem is incurred by the feature construction scheme of [12], where the authors propose using the output of DCNN layer 13 as a local descriptor computed on multi-scale dense image patches. Inspired by this approach, we further consider stacking the output of the fully connected layers (11, 12, and 13) to our hybrid DCNN/aggregator feature. We illustrate the results of this approach in Fig. 4, where the non-fully connected layers are processed according to (4), and the fully-connected layers are concatenated without any processing. Note that using the 3 fully connected layers and the last non-fully connected layer results in performance close to 74 mAP points. This compares very well to the performance of 77.73 of PRE1000C in Table 2, particularly considering the drastic difference in training time and testing time.

method	feature dimension	mAP
BoW + SPM, N=4000 [17]	32000	45.39
FV (SIFT) [23]	262144	58.3
FV (SIFT + color) [23]	262144	60.3
PRE1000C [9]		77.73
Hybrid DCNN/FV, m=64	81920	54.56
Hybrid DCNN/BoW, N=30	150	50.53
Hybrid DCNN/BoW, N=500	2500	60.32

Table 2. Comparison of our results (using last 5 layers) with the state-of-the-art (N represents the codebook size in BoW).

5. CONCLUSION

In this work, we proposed a hybrid Deep Convolutional Neural Network (DCNN) / Bag-of-Words (BoW) image feature extraction approach. Treating the output of intermediate layers of a pre-trained DCNN as local descriptors allowed us to use an unsupervised Bag-of-Words aggregator to obtain an image feature that outperforms standard aggregators based on local descriptors substantially on the Pascal VOC 2007 benchmark. Appending the output of the fully-connected layers to our hybrid feature further improves the performance of our approach, making it competive with DCNNs variants adapted to Pascal VOC 2007, and at a fraction of the training and testing cost.

6. REFERENCES

[1] David G Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol. 60, no. 2, pp. 91–110, 2004.

[2] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray, "Visual categorization with bags of keypoints," in *Workshop on statistical learning in computer vision, ECCV*, 2004, vol. 1, pp. 1–2.

[3] Florent Perronnin and Christopher Dance, "Fisher kernels on visual vocabularies for image categorization," in *CVPR*, 2007, pp. 1–8.

[4] Jinjun Wang, Jianchao Yang, Kai Yu, Fengjun Lv, Thomas Huang, and Yihong Gong, "Locality-constrained linear coding for image classification," in *CVPR*, 2010, pp. 3360–3367.

[5] Jan C van Gemert, Jan-Mark Geusebroek, Cor J Veenman, and Arnold WM Smeulders, "Kernel codebooks for scene categorization," in *ECCV*, pp. 696–709. Springer, 2008.

[6] Xi Zhou, Kai Yu, Tong Zhang, and Thomas S Huang, "Image classification using super-vector coding of local image descriptors," in *ECCV*, pp. 141–154. Springer, 2010.

[7] Jonathan Delhumeau, Philippe-Henri Gosselin, Hervé Jégou, and Patrick Pérez, "Revisiting the VLAD image representation," in *Proceedings of ACM International Conference on Multimedia*, New York, New York, USA, 2013, vol. 21, pp. 653–656, ACM Press.

[8] Alex Krizhevsky, I. Sutskever, and Geoffrey E Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *NIPS*, 2012, pp. 1–9.

[9] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *CVPR*, 2014, pp. 1717–1724.

[10] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman, "Deep fisher networks for large-scale image classification," in *NIPS*, 2013, pp. 163–171.

[11] Vladyslav Sydorov, Mayu Sakurada, and Christoph H Lampert, "Deep fisher kernels–end to end learning of the fisher kernel gmm parameters," in *CVPR*, 2014, pp. 1402–1409.

[12] Yunchao Gong, Liwei Wang, Ruiqi Guo, and Svetlana Lazebnik, "Multi-scale orderless pooling of deep convolutional activation features," in *ECCV 2014*, pp. 392–407. Springer, 2014.

[13] Josef Sivic and Andrew Zisserman, "Video Google: A text retrieval approach to object matching in videos," *ICCV*, pp. 2–9, 2003.

[14] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *CVPR*. IEEE, 2006, vol. 2, pp. 2169–2178.

[15] Anna Bosch, Andrew Zisserman, and Xavier Munoz, "Representing shape with a spatial pyramid kernel," in *Proceedings of the 6th ACM international conference on Image and video retrieval*. ACM, 2007, pp. 401–408.

[16] Florent Perronnin, J Sánchez, and Thomas Mensink, "Improving the fisher kernel for large-scale image classification," *ECCV*, pp. 143–156, 2010.

[17] Ken Chatfield, Victor Lempitsky, Andrea Vedaldi, and Andrew Zisserman, "The devil is in the details: an evaluation of recent feature encoding methods," in *BMVC*, 2011, number 1, pp. 76.1–76.12.

[18] Omkar M Parkhi, Andrea Vedaldi, and Andrew Zisserman, "On-the-fly specific person retrieval," in *Image Analysis for Multimedia Interactive Services (WIAMIS), 2012 13th International Workshop on*. IEEE, 2012, pp. 1–4.

[19] Jorge Sanchez and Florent Perronnin, "High-dimensional signature compression for large-scale image classification," *CVPR*, pp. 1665–1672, June 2011.

[20] R Arandjelovic and Andrew Zisserman, "Three things everyone should know to improve object retrieval," *CVPR*, 2012.

[21] Hervé Jégou and Andrew Zisserman, "Triangulation embedding and democratic aggregation for image search," in *CVPR*, 2014.

[22] M Everingham, L Van Gool, CKI Williams, J Winn, and A Zisserman, "The pascal visual object classes challenge 2007 (voc 2007) results (2007)," 2008.

[23] Florent Perronnin, Jorge Sánchez, and Thomas Mensink, "Improving the fisher kernel for large-scale image classification," in *ECCV*, pp. 143–156. Springer, 2010.