PART-BASED CONVOLUTIONAL NEURAL NETWORK FOR VISUAL RECOGNITION

Lingxiao Yang[1], *Xiaohua Xie*[*] [2] , *Peihua Li*[3], *David Zhang*[1], *Lei Zhang*[1]

[1] Department of Computing, The Hong Kong Polytechnic University, China
[2] School of Data and Computer Science, Sun Yat-Sen University, China
[3] School of Information and Communications Engineering, Dalian University of Technology, China

ABSTRACT

Mid-level element based representations have been proven to be very effective for visual recognition. We present a method to discover discriminative elements based on deep Convolutional Neural Networks (CNNs), namely Part-based CNN (P-CNN), which acts as the role of encoding module in part-based representation. The P-CNN can be attached at arbitrary layer of a pre-trained CNN and be trained using image-level labels. The training of P-CNN essentially corresponds to the optimization and selection of discriminative mid-level visual elements. For an input image, the output of P-CNN is naturally the part-based coding and can be directly used for image recognition. By applying P-CNN to multiple layers of a pre-trained CNN, more diverse visual elements can be obtained for visual recognitions. Experiments are conducted on two recognition tasks and their results demonstrate the effectiveness of the proposed method.

Index Terms— Mid-level element, convolutional neural networks, scene recognition, action categorization

1. INTRODUCTION

The essential challenge of machine visual recognition is the large gap between low-level features and high-level semantics. The mid-level visual element based models provide an effective way to narrow the gap and have achieved significant progress in a variety of visual recognition tasks [1, 2, 3, 4, 5, 6, 7, 8, 9]. In practice, the visual elements mostly refer to image parts or features that correlate to specific semantic content, *e.g.* an object part, an object entirety, a visual phrase, *etc*, but are not restricted to be any one of them.

There have been many methods proposed to discover mid-level parts. The pioneer work proposed by Singh *et al.* [3] utilizes a cross-validation strategy to learn discriminative elements. Juneje [5] employed Examplar-LDA [10] to train and refine the parts. Sun and Ponce [6] designed a latent SVM with group sparsity to automatically optimize and select parts. Doersch *et al.* [4] utilized the meanshift algorithm to optimize the parts and achieved state-of-the-art results on scene recognition. Recently, many researchers discovered useful parts within a pre-trained CNN model. Specifically, Parizi *et al.* [1] first selected a few discriminative parts by the similar method in [6], and then jointly trained part filters and classifiers by a global classification loss, which leads to significant improvement in scene categorization. Li *et al.* [2] proposed a pattern-mining approach to identify mid-level elements within images. We propose to jointly optimize the parts and the classifier. However, there are considerable difference between our method and [1, 2]. First, the manner in [1] uses block coordinate descent to alternatively update parts and classifiers. In this paper, we utilize the backpropogation algorithm which allows our method to simultaneously update parts and classifiers. Furthermore, our framework can automatically rank visual elements even without group sparsity regularization. This strategy is much simpler than previous methods [1, 2] since we avoid either training a classifier or using "maximizing coverage" [4] to select the discriminative elements. Moreover, our method can discover diverse discriminant elements on different CNN layers and achieve state-of-the-art results on two challenging visual recognition tasks.

Our method can be summarized as follows. Given a pre-trained CNN, we run the network on a large set of training images and employ an unsupervised max-margin analysis [11] to sample candidate visual elements from the feature maps at considered layers. At each considered layer, we employ additional specially designed CNN to facilitate the learning of visual elements. We then attach the newly designed layer using sampled candidate visual elements as the initialization of filters. Here, the new layer contains a successive convolution, pooling, and rectified linear unit (ReLU), followed by a SVM classifier related to special recognition tasks, formulating a joint optimization framework. We call the additional layer as part-based CNN (P-CNN) since it forces to optimize part-like visual elements. The reason for the designing is that we find the encoding phase of element-based representation [1, 2, 3, 4, 5, 6] is very similar to some operators within traditional CNNs (See Fig. 1).

[*] Corresponding author.

This project is supported by the Natural Science Foundation of China (No. 61672544), Guangdong Natural Science Foundation (No. 2015A030311047), Fundamental Research Funds for the Central Universities (No. 161gpy41), and Shenzhen Innovation Program (No. JCYJ20150401145529008).

For a given image, the responses of visual elements of multiple layers can be intergraded to serve different visual recognition tasks. We have applied the proposed method to scene recognition and action categorization. Experiments on benchmarks show that the proposed method outperforms many state-of-the-arts.

2. THE PROPOSED METHOD

Fig. 1 shows the overview of the proposed P-CNN. First, we describe our definition of the visual element and the method that generates initial candidate elements in Section 2.1, and then detail the P-CNN in Section 2.2. In Section 2.3, we present a manner that applies the P-CNN to visual recognitions. Section 2.4 illustrates a visualization technique for the elements. In Section 2.5, we develop a method that uses P-CNN on multiple layers of the pre-trained CNNs.

2.1. Visual elements definition and initialization

Each visual element $\boldsymbol{p}^l$ is defined as a $1 \times 1 \times D^l$ dimensional vector on the CNN feature maps, where D^l is the number of channels of the l-th layer feature map. Under this definition, the spatial size of visual element (also known as receptive field) and the response location of part convolution can be inferred by the technique proposed in [12]. For example, on the "$Conv_3$" layer of the AlexNet [13], the dimension of each element is $1 \times 1 \times 384$, while in the image domain, the spatial size of each element is 99×99. The technique from [12] helps us better understand and visualize the elements.

For the single scale, given a pre-trained CNN, we extract a $R^l \times C^l \times D^l$ dimensional feature map at the l-th layer for each image, resulting in $R^l \times C^l$ visual elements ($\boldsymbol{p}^l$) according to our definition. The R^l and C^l represent the rows and columns of the feature map at the l-th layer, respectively. For each visual element, we calculate the score $\phi(\boldsymbol{p}^l) = (squeeze^1(\boldsymbol{p}^l) - \boldsymbol{\mu}^l)^T {\boldsymbol{\Sigma}^l}^{-1}(squeeze(\boldsymbol{p}^l) - \boldsymbol{\mu}^l)$, where $\boldsymbol{\mu}^l \in R^{D^l}$ and $\boldsymbol{\Sigma}^l \in R^{D^l \times D^l}$ are the mean and covariance of all visual elements at the l-th layer. We estimate $\boldsymbol{\mu}^l$ and $\boldsymbol{\Sigma}^l$ from around 500,000 visual elements, sampled from the l-th layer feature maps of training images regardless of their category labels. The $\boldsymbol{p}^l$ with larger ϕ is expected to be more discriminative [11]. We discard 50% least discriminative elements of each image and remove near-duplicate ones by Non-Maximum Suppression (NMS). In our experiments, the threshold for NMS is set as 0.5. We then sample N^l elements $\boldsymbol{P} \in R^{1 \times 1 \times D^l \times N^l}$ without labels as the initialized parts in the P-CNN. For the multi-scale case, we re-scale images with s scaling factors, and conduct this operation over all scales (*i.e.* using NMS across all scales).

[1] A matlab function returns an array with the same elements as input, but with all singleton dimensions removed.

2.2. P-CNN for visual element discovery at one layer

Our Part-based CNN consists of a convolution, a L-level spatial pyramid pooling (SPP), and a non-linear ReLU activation (Fig. 1). For convenience, we denote r as the total number of spatial regions in SPP [14], and φ as the whole P-CNN procedure. The inputs to φ are the multi-scale feature maps at the l-th layer of a pre-trained CNN:

$$\boldsymbol{v}^l = \varphi(CNN^l(\boldsymbol{I}), \boldsymbol{P}^l), \tag{1}$$

where $CNN^l(\boldsymbol{I})$ is the l-th layer feature map of the input image I. The $\boldsymbol{P}^l = \{\boldsymbol{p}_n^l, n = 1, 2, ..., N^l\}$ stands for a collection of candidate parts in the l-th layer. For simplicity, we omit the superscript l in the following. The output of P-CNN φ is a rN-dimensional feature for every image. We propose to attach a classifier on the output of φ as follows:

$$\min_{\boldsymbol{\Theta}} \frac{1}{M} \sum_{i=1}^{M} \sum_{k=1}^{K} \mathcal{L}(y_{ik}, v_i, \theta_k) + \lambda \sum_{k=1}^{K} ||\theta_k||_2^2, \tag{2}$$

where $\Theta = \{\theta_k, k = 1, 2, ...K\}$ groups all classifiers. M and K are the numbers of the training samples and categories, respectively. v_i is the output from our P-CNN for each image, and y_{ik} indicates whether the i-th instance is associated with the k-th category ($y_{ik} = 1$) or not ($y_{ik} = -1$). L2-SVM is adopted as loss ($\mathcal{L}$) in Eq. (2) since it empirically performs better than softmax [15].

Now we discuss the optimization procedure of P-CNN. First, the L2-SVM is differentiable thus the backpropagation algorithm on the classifier layer is almost identical to the softmax in standard CNNs [15]. Second, for the backpropagation through SPP layer, we sum gradients across different levels (Fig. 1 (b)). With above analysis, the P-CNN can be trained in an end-to-end manner. To avoid overfitting, we add another regularization term on the parts:

$$\min_{\tilde{\boldsymbol{\Theta}}} \frac{1}{M} \sum_{i=1}^{M} \sum_{k=1}^{K} \mathcal{L}(y_{ik}, v_i, \theta_k) + \lambda \sum_{k=1}^{K} ||\theta_k||_2^2 + \gamma \sum_{n=1}^{N} ||p_n||_2^2, \tag{3}$$

where $\tilde{\boldsymbol{\Theta}}$ groups all parameters including the parts $\boldsymbol{P}$. Different from previous works [1, 6] that adopt group sparsity for parts. Here, we find the l_2-norm holds a similar functionality.

2.3. Visual element based image coding

After optimizing both visual parts and classifiers, we can use them to directly predict a new instance. However, the stage of initialization may produce redundant or useless parts. We propose to further select elements by using the learned classifier. Denote $\Theta = [\theta_1, \theta_2, ..., \theta_{rN}]$ as a $rN \times K$ matrix. Thus every part in each region is associated with one row of Θ, then we select the most Z discriminative parts with larger $||\Theta_j||_2, j = 1, ..., rN$, where Θ_j represents the j-th row of the Θ. Finally, we apply P-CNN with selected elements to encode images and re-train the classifier for recognitions. Note that the selected elements are associated with the specific spatial grids according to above discussion. In this case,

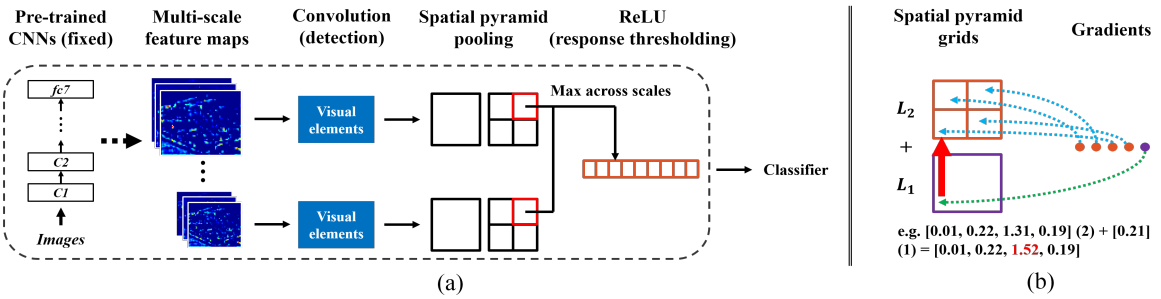

Fig. 1. Illustration of P-CNN attached at one layer of the pre-trained CNN and its optimization. (a) The P-CNN pipeline. The equivalent steps to the part-based image representation are claimed in parenthesis. (b) The backpropagation through a 2-level spatial pyramid pooling (SPP) layer. Here, the numbers in square brackets and parentheses denote the gradients from successive layer and the spatial pyramid level respectively. The number order is the same to gradient dots. We sum the third and fifth gradients (red arrow) as both they come from the same location in different levels (Best viewed in color).

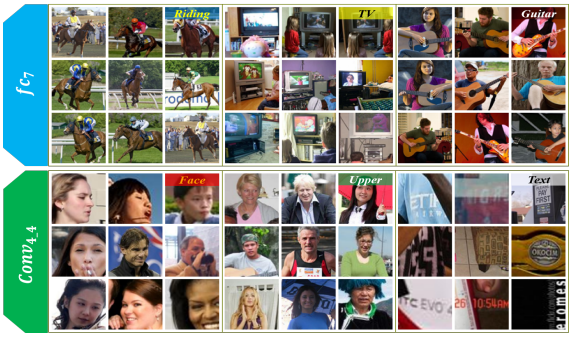

Fig. 2. Visualization of 6 visual parts discovered by P-CNN on two layers of VGG19 [16] from Action40 database. For each part, we show 9 test image patches with top responses (Best views in color)

we only need to convolute each element and record the max response in its correspondence region in SPP.

2.4. Visualization of visual element

To intuitively understand the learned visual elements, first, we run the optimized model on testing images and obtain a set of responses. For each response from the $\boldsymbol{v}_i$, we first back-project it on the feature maps at l-th layer, which is worked on the fact that each response is related to a visual element in a specific spatial region. Then, the center and the spatial size of corresponding patch in image coordinates can be acquired using the method proposed in [12]. Finally, for each learnt visual element, we can visualize the image patches with larger response value from testing images. These image patches illustrate the main pattern of the concerned visual element. Some examples are shown in Fig. 2.

2.5. P-CNN on multiple layers

Due to the hierarchical abstraction ability of CNNs, we propose to employ P-CNN on multiple layers of a pre-trained CNN to discover diverse parts. Specifically, the P-CNN together with a SVM classifier is attached and individually optimized at each considered layer. For the recognition application, the parts-based image features at different layers are concatenated into a long feature vector, further fed into a classifier. In practice, we only discover visual parts at higher layers, *i.e.*, after 3-rd layer, since the filters at lower layers are more likely to capture low-level patterns such as edges, colors, and curves. We demonstrate that the cooperation of visual elements from different layers can further improve the performances in Section 3.2.

3. EXPERIMENTS

We perform experiments on two popular tasks: scene (MITindoor [17]) recognition and action categorization (Action40 [18]). Our splits are followed by the authors' suggestions in each dataset. We report *Mean Accuracy* (mAcc) for MITindoor, and *Mean Average Precision* (mAP) for Action40.

3.1. Implementations

We employed two popular CNN models, AlexNet [13] and VGG19 [16], which are both pre-trained on the ILSVRC12 [19] database. Given an image, we resize its smaller dimension to 454 (448) for AlexNet (VGG19), while maintaining its aspect ratio. We then extract the l-th layer feature maps in 5 different scales $\{2^s, s = (-1, -0.5, 0, 0.5, 1)\}$. For single scale baselines, we extract the feature by resizing the image to 227×227 (224×224) for AlexNet (VGG19).

We sample $N = 2k$ candidate visual elements at all considered layers. For a fair comparison with other part-based models, the spatial level L is set to 2 (1×1 and 2×2) and hence the total number of grids r is 5. For the convolution response removing, the threshold is set to -0.5 suggested by [4], resulting in a special ReLU unit $max(-0.5, x)$. For the joint optimization of P-CNN and SVM classifier, we first subtract all elements of each image by the mean over training

Table 1. The hyper-parameters for all experiments. Different values for VGG19 features are in parentheses. C represents the $Conv$ layers, while fc means the fully-connect layers.

Descriptions	[$C3$, $C4$, $C5$]	[$fc6$, $fc7$]
Standard Deviation	$4e-2$	$4e-3$
Momentum rate	0.9	0.9
Classifier Learning Rate	$4e-5$ ($4e-6$)	$4e-6$
λ	$4e-2$ ($4e-3$)	$4e-2$
Element Learning Rate	0.4	4
γ	$4e-3$	$4e-3$

set, and normalize each one using its own mean and standard deviation, followed by a whitening operation. After preprocessing, the output of P-CNN ($\boldsymbol{v}$) is very high (typically above 10^4). For a convenience of tuning parameters, we scale all elements by a factor of 0.01. The optimization of our P-CNN model is achieved by using the Stochastic Gradient Descent (SGD) with momentum technique. The weights of classifiers are initialized under a Gaussian distribution. We list all parameters in Tab. 1. We executed 5 epoches in total and all learning rates decrease by half after each epoch. We also include the horizontal reflections of original images for training. Note that, all parameters are just fine-tuned on a small set of training images of MITindoor dataset.

We optimize and select $Z = 2k$ discriminative ones from initial $10k$ (rN) elements. Then we use P-CNN with selected elements to encode images. According to the discussion in Section 2.2, each image is encoded as a feature vector of $2k$ dimension. For each image, we also average the feature vectors from itself and its horizontal flip, followed by the l_2 normalization. For multiple layers, we process each layer individually, and concatenate features from different layers to form a $10k$-dimensional feature vector. In the re-training stage, we use the LibLinear [20] to train one-vs-all linear SVMs. The l_2 loss SVM with l_2 regularization is used, and the penalty factor is set to 5 for all experiments. We denote our method as **P-CNN-X**, where **X** stands for different experiment setting.

3.2. Results

We first compare the proposed P-CNN with three pooling baselines, including max pooling on single (SMP) and multiple scale features (MMP), and spatial pyramid pooling (SPP) on multiple scale features. We then show the comparisons between ours with other related methods.

Under the feature extraction framework described in Section 3.1, the SMP and MMP from the fc_7 layers are the generic features used in prior works [21, 22]. Note that, we only consider using SPP on the multi-scale feature maps because it always outperforms the one working on single scale. Tab. 2 (a) shows that the MMP is superior than SMP, but the best performance is obtained by SPP. Based on these observations, we discover parts using P-CNN with SPP on the multi-scale feature maps. Tab. 2 (d) also demonstrates P-CNN on multiple layers (P-CNN-Full) performs better than the one just using single layer (P-CNN-fc_7). Here, we only include the results obtained by using P-CNN on fc_7 layer because it achieves better performance than the one from other layers.

Table 2. Performances obtained by different methods on 2 databases. For fair comparison, all methods are using single network. (a) Three baseline methods. (b) The state-of-the-art methods. (c) Semantic parts based methods using deep features. (d) Our methods. Results marked with $*$ were achieved using Hybrid CNN [23]. Other results are obtained using VGG16/19 [16] or AlexNet [13] (in parentheses).

	Methods	MITIndoor (mAcc)	Action40 (mAP)
a	fc_7-SMP	69.35 (58.36)	72.40 (58.46)
	fc_7-MMP	78.89 (69.32)	80.56 (65.40)
	fc_7-SPP	80.92 (70.90)	81.60 (67.20)
b	FVCNN [24]	81.00 (69.70)	-
	MPP [25]	(**75.97**)	-
	SCBODF [26]		80.00
c	MDPM [2]	77.63 (69.69)	-
	Joint [1]	73.30*	-
	EPM [8]	-	72.30 (60.20)
	ASPD [27]	-	75.40
d	P-CNN-fc_7	81.55 (73.30)	85.42 (71.74)
	P-CNN-Full	**82.60** (74.64)	**86.90** (**72.80**)

Tab. 2 (b) and (c) show the performances of compared methods on two databases. Note that, we do not contain many previous works using handcraft features such as [3, 5, 4], since all these methods are not better than the ones using deep features. The proposed P-CNN achieves the best recognition results on both tasks when using VGG19 model. In the comparisons to part-based methods (Tab. 2 (c)), our P-CNN is significantly better than all methods. For example, with the same network setting, the performances of our P-CNN is substantially better than the recently proposed MDPM *et al.* [2] on MITindoor database. When comparing with state-of-the-art methods in Tab. 2 (b), our P-CNN reaches the highest performance on Action40 dataset and still achieves comparable result with MPP on MITIndoor database. However, the feature dimension used in ours is about 6 times lower than MPP ($10k$ *v.s.* $65k$). All results in Tab. 2 demonstrate the effectiveness of the proposed P-CNN.

4. CONCLUSIONS

We present a part-level convolutional neural network (P-CNN) model, and derive part-based image classifiers that have achieved state-of-the-art or competitive results on 2 recognition tasks. The key enabler of our model is to integrate the hierarchical abstraction capacity of CNNs and the semantic concentration capacity of part-based image representation. Another merits of our method is that the optimization and selection of parts is directly driven by classification losses.

5. REFERENCES

[1] S. N. Parizi, A. Vedaldi, A. Zisserman, and P. F. Felzenszwalb, "Automatic discovery and optimization of parts for image classification," in *ICLR*, 2015.

[2] Y. Li, L. Liu, C. Shen, and A. van den Hengel, "Mining mid-level visual patterns with deep CNN activations," *in IJCV*, 2016.

[3] Saurabh Singh, Abhinav Gupta, and Alexei Efros, "Unsupervised discovery of mid-level discriminative patches," *in ECCV*, pp. 73–86, 2012.

[4] Carl Doersch, Abhinav Gupta, and Alexei A Efros, "Mid-level visual element discovery as discriminative mode seeking," in *NIPS*, 2013, pp. 494–502.

[5] M. Juneja, A. Vedaldi, C. V. Jawahar, and A. Zisserman, "Blocks that shout: Distinctive parts for scene classification," in *CVPR*, 2013.

[6] Jian Sun and Jean Ponce, "Learning discriminative part detectors for image classification and cosegmentation," in *ICCV*. IEEE, 2013, pp. 3400–3407.

[7] Abhishek Jain, Arpan Gupta, Mikel Rodriguez, and Larry S Davis, "Representing videos using mid-level discriminative patches," in *CVPR*. IEEE, 2013, pp. 2571–2578.

[8] Gaurav Sharma, Frédéric Jurie, and Cordelia Schmid, "Expanded parts model for semantic description of humans in still images," *in PAMI*, 2016.

[9] Peng-Ju Hsieh, Yen-Liang Lin, Yu-Hsiu Chen, and Winston Hsu, "Egocentric activity recognition by leveraging multiple mid-level representations," in *ICME*. IEEE, 2016, pp. 1–6.

[10] Bharath Hariharan, Jitendra Malik, and Deva Ramanan, "Discriminative decorrelation for clustering and classification," in *ECCV*, pp. 459–472. Springer, 2012.

[11] Mathieu Aubry, Bryan C Russell, and Josef Sivic, "Painting-to-3d model alignment via discriminative visual elements," *in TOG*, vol. 33, no. 2, pp. 14, 2014.

[12] K. Lenc and A. Vedaldi, "R-cnn minus r," in *BMVC*, 2015.

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012, pp. 1097–1105.

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

[15] Yichuan Tang, "Deep learning using linear support vector machines," *arXiv preprint arXiv:1306.0239*, 2013.

[16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2014.

[17] Ariadna Quattoni and Antonio Torralba, "Recognizing indoor scenes," in *CVPR*. IEEE, 2009, pp. 413–420.

[18] Bangpeng Yao, Xiaoye Jiang, Aditya Khosla, Andy Lai Lin, Leonidas Guibas, and Li Fei-Fei, "Human action recognition by learning bases of action attributes and parts," in *ICCV*. IEEE, 2011, pp. 1331–1338.

[19] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*. IEEE, 2009, pp. 248–255.

[20] Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin, "Liblinear: A library for large linear classification," *in JMLR*, vol. 9, pp. 1871–1874, 2008.

[21] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng, and Trevor Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *ICML*, 2014, pp. 647–655.

[22] Ali S Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson, "Cnn features off-the-shelf: an astounding baseline for recognition," in *CVPRW*. IEEE, 2014, pp. 512–519.

[23] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva, "Learning deep features for scene recognition using places database," in *NIPS*, 2014, pp. 487–495.

[24] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, and Andrea Vedaldi, "Deep filter banks for texture recognition, description, and segmentation," *in IJCV*, vol. 118, no. 1, pp. 65–94, 2016.

[25] Donggeun Yoo, Sunggyun Park, Joon-Young Lee, and In Kweon, "Multi-scale pyramid pooling for deep convolutional representation," in *CVPRW*, 2015, pp. 71–80.

[26] Fahad Shahbaz Khan, Joost van de Weijer, Rao Muhammad Anwer, Andrew D Bagdanov, Michael Felsberg, and Jorma Laaksonen, "Scale coding bag of deep features for human attribute and action recognition," *arXiv preprint arXiv:1612.04884*, 2016.

[27] F.S. Khan, Jiaolong Xu, J. van de Weijer, A.D. Bagdanov, R.M. Anwer, and A.M. Lopez, "Recognizing actions through action-specific person detection," *in TIP*, vol. 24, no. 11, pp. 4422–4432, Nov 2015.