Scene understanding based on Multi-Scale Pooling of deep learning features

DongYang Li [1,a], Yue Zhou [2,b]

[1,2] Institute of Image Processing and Pattern Recognition, Shanghai Jiaotong University, Shanghai 200240, China

[a] jingshui1216@gmail.com, [b] zhouyue@sjtu.edu.cn

Keywords: CNNs, MOP-CNN, SPP-net, Scenes understanding

Abstract. Deep convolutional neural networks (CNNs) have recently shown impressive performance as generic representation for recognition. However, the feature extracted from global CNNs lack geometric invariance, which limits their robustness for classification and detection of highly variable objects .To improve the invariance of the features without degrading their discriminative power and speed up the calculation, we follow the next two method. Firstly, we adopt the scheme called multi-scale orderless pooling (MOP-CNN) which extracts CNNs activation from local patches of the image at multiple scale levels, performs orderless VLAD pooling of these activations at each level separately, and concatenates the result. Second, to speed up the calculation, we adapt the SPP-net as the CNNs architecture. Using SPP-net, we compute the feature maps from the entire image only once, and then pool features in arbitrary regions (sub-images) to generate fixed-length representations for training the detectors. This method avoids repeatedly computing the convolutional features. On the challenging SUN397 Scenes classification datasets, our method achieves competitive classification results.

Introduction

Recently, more and more people havedrawn their attention on deep convolutional neural networks(CNNs) [1] since the breakthrough achieved by Krizhevsky [2] on the ImageNet dataset. Based on this CNNs architecture above, researchers in vision community have made substantial improvements in image classification [2, 3, 4, 5] and object detection[6, 4, 7]. These owe to not only the improvement of the CNNs architectures and training algorithms, but also using CNNs features as generic representation. Meanwhile, combining deep architectures and classical computer vision have gotten big gains in image classification as well as object detection. Here we are considering whether we can get more accurate result by combing features extracted at multiple local image windows and not increasing the calculation time sharply in the meantime. Inspired by previous work of Spatial pyramid pooling [8, 9, 10], SPP-net[7] was proposedthat can significantly outperforms global CNNs activations for the traditional task like classification. We adapta novel and simple pooling scheme producing a feature that significantly outperforms global CNNs activations for tasks like image classification object detection.

Image representation counts. In the past dozen years, most of the quantitative improvements to image understanding can be attributed to the introduction of improved image representations, from the Bag-of-Features(BoF)[8] to Spatial Pyramid Matching(SPM)[9, 10].These methods have a common characteristic that they are largely handcrafted. They simply comprise dense sampling of local image patches,describe them by means of visual descriptors such as SIFT, encode them into a high dimensional representation, and then pool over the image. DNNs, as exemplified by the system of Krizhevskyet al. [2], are a completely different architecture.They accept the raw image pixels as the input,and then sent them through five convolutional layers, each of which filters the feature maps and then max-pools the output within local neighborhoods.Now, the representation still preserves a great deal of global spatial information for the above five layers justcompute the adjacent information which doesn't change the relative position in space.Though max-pooling within each feature map helps to boost invariance to small-scale deformations [11].However,the preserved spa-

tial information may undermine the invariance to large-scale,more global deformations. The five layers are followedby two fully connected layers, finally producing an activation of 4096 dimensions.MOP-CNN[12] shows that the final representation is still fairly sensitive to global translation, rotation, and scaling and it directly translates into a loss of accuracy for classification tasks.

SPM is an updated version of BOF in that it adds enough spatial information. As we can see, DNNs activations preserve too much spatial information. MOP-CNN is an effective framework that builds a more orderless representation on top of CNN activation to improve recognition performance. From Fig. 1 we can see how it's realized. Simply put, we use the same network to extract features from local patches at multiple scales. Firstly, we extract feature from the whole image, so global spatial information is still kept. Second, we extract features from sub-images which allow us to capture more local details of the image. Then VLAD [13] encoding is used to aggregate local patch responses and this will build a more invariant representation for the orderless nature of VLAD. Finally, we concatenate the original global deep activations with the VLAD features for the sub-images at multi scales to form our new image representation.

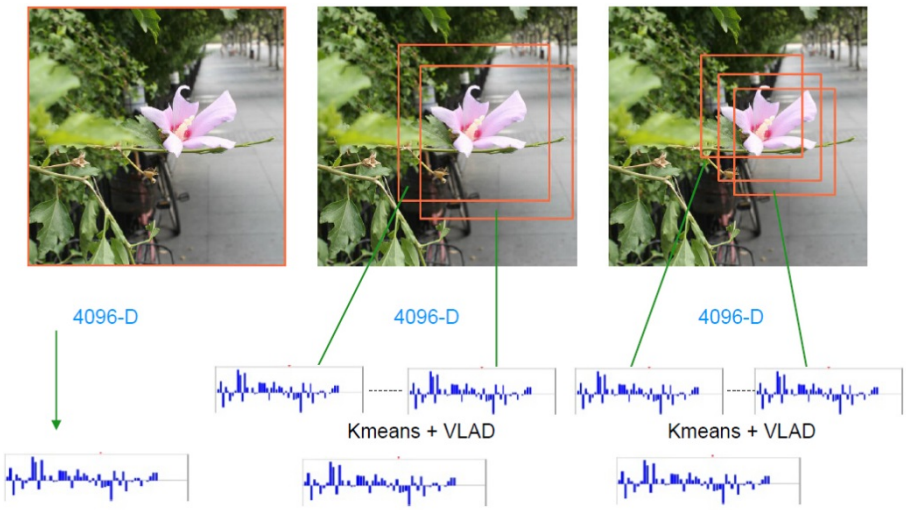
As we can see in Fig. 1, every time we extract deep activation feature from one local patch, it needs the convolutional computation of the five layers. This will cost a lot of time. If we can find a way that only needs to apply the deep network once to the entire image, which will save us from the computation time. Here we utilize the SPP-net as our CNNs architecture which can accept a full image as the input and can map the local image patch to the feature maps. From Fig. 2, we can see how SPP-nets realized. This network add a SPP layer on the top of the last convolutional layer, which will pools the features and produce a fixed length vector no matter what the size of the input image. Fig. 3 will tell us how to achieve the projection between the feature maps and the image.

Detail of the technology

Convolutional Layers and Feature Maps. The most popular architectures [2, 3] of DNNs include seven layer. The first five layers are convolutional layers, some of which followed by pooling layer. The last two layers are fully-connected layers, with an N-way softmax as the output, where N is the number of categories. These architectures need a fixed-size image as the input, which constrains the applicability of the deep network for different sizes of input. The constrain of fixed-size input only come from the fully-connected layers which demand the fixed-length vectors as the input. Now that the convolutional layer can accept inputs of arbitrary sizes and the constrain of input size come from the fully-connected layers, we will solve this problem in next chapter. The output of convolutional layers can preserve much more spatial information which will do damage to the accuracy of the image classification and detection. VLAD pooling will give an answer to this problem.

The Spatial Pyramid Pooling Layer. Fixthe size of the convolution kernel though the convolutio nal layers can accept arbitrary input sizes, the sizes of outputs they produce are also variable. With t he fully-connected layers require fixed-length vectors, we need some improve to get fixed-size vect ors. Inspired by BoF and SPM, which pools the features together, we adopt the spatial pyramid pool ing as our method that produce the needed vectors. In contrastto sliding window pooling in the prev ious deep networks[2, 3]whose kernel sizes are fixed thus the number of sliding windows depends o n the input size, these spatial bins have sizes proportional to the image size, so the number of bins is fixed regardless of the image size. Replacing the last pooling layer (e.g.,pool5 , after the last convol utional layer) with a spatial pyramid pooling layer, the deep network can accept input images of arbi trary size. Fig. 2 illustrates our method. Just like sliding window pooling, we pool the responses of e ach filter in each spatial bin (throughout this paper we use max pooling). Now we will give an exam ple of how to get a fixed length output. Consider an image with a given size, we can get the bin size s needed for the pyramid pooling through computing before carry out the project. Supposing that th e size of the feature maps after conv5 is a*a(e.g., 13*13) and the pyramid level has n*n bins. Next, we implement sliding window pooling for this pooling level, where the window size $\text{win} = \lceil a / n \rceil$ a nd stride $str = \lfloor a / n \rfloor$ with $\lceil \square \rceil$ and $\lfloor \square \rfloor$ denoting ceiling and floor operations. Now we will get an n*

n*K dimensions vector(kis the number of filters in the last convolutional layer) which is the input to the fully-connected layer.



(a) Level1: (Left) global activation. (b)Level2: (Mid) pooled features.(c) Level3: (Right) pooled features.

Fig.1.Overview of multi-scale orderless pooling for CNNs activations (MOP-CNN).Our proposed feature is a concatenation of the feature vectors from three levels: (a)Level 1, corresponding to the 4096-dimensional CNNs activation for the full 480*480image; (b) Level 2, formed by extracting activations from 256*256 patches and VLADpooling them with a codebook of 100 centers; (c) Level 3, formed in the same way aslevel 2 but with 128*128 patches.

Combining the spatial pyramid pooling layer with the fifth convolutional layer, the input image can be of any sizes. This not only allows arbitrary aspect ratios, but also allows arbitrary scales. We can pass the input image through the whole network whatever the size is. If we resize the input image to different scales, the network(with the same filter sizes) will extract features at different scales.

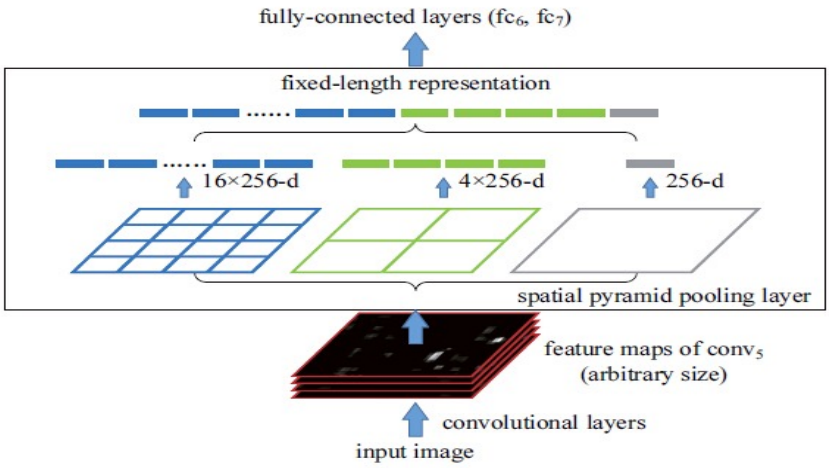


Fig. 2: A network structure with a spatial pyramid pooling layer. Here 256 is the filter number of the last convolutional layer.

Mapping a Window in image domain to Feature Maps. If we want to compute features in multi scales in one image, we don't need input all cropped or resized windows in image domain. We can just align the window on the feature maps. In this paper, weproject the corner point of the window onto a pixel in the feature maps, such that this corner point(in the image domain) is closest to the center of the receptive field of that pixel. This projection depends on the network architecture. This can be illustrated in Fig. 3.

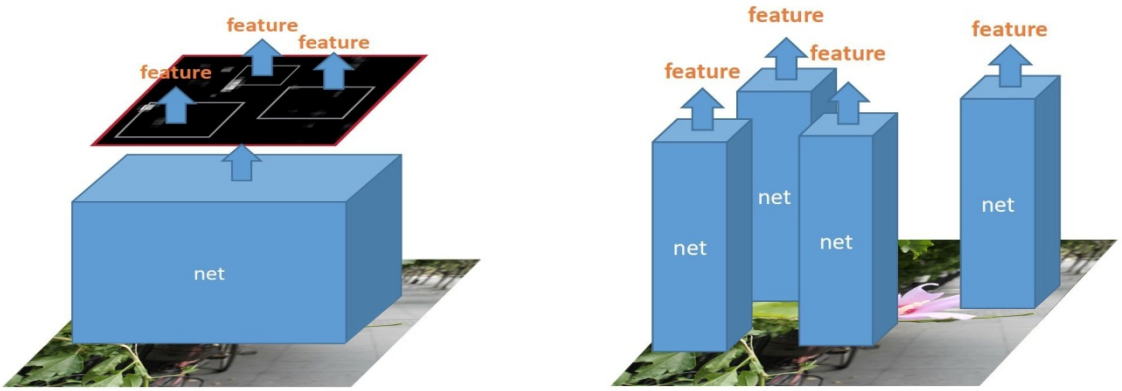


Fig. 3: A compare between our method and traditional method in extracting features from different patches in one image. A) Left: Our method passes the whole image through the network, and projects the features to the patches. B) Right: Traditional method needs pass every image patch through the network. Obviously, our method can reduce computing time sharply.

The proposed MOP-CNN

As basically illustrated in Fig. 1, we need features extracted from multiple scales. In this paper, our representation has three scale levels, corresponding to CNN activations of the global 480*480 image and 256*256 and 128*128 patches, respectively. To extract these activations, we use the Caffe GPU implementation [14] training our architecture. Given an input image, we resample it to 480*480 pixels, subtract the mean of the pixel values, and feed the patch through the network. Combining Fig. 1 with Fig. 3, to get features from patches of different scales, we can map the patches to the feature maps of the fifth convolutional layer. Then feed corresponding feature maps to the SPP layer and the last two fully-connected layers and we get the 4096-dimensional output. For the first level, we simply take the 4096-dimensional CNN activation for the wholeimage. For the remaining two levels, we extract activations for all 128*128 and 256*256 patches sampled with a stride of 64 pixels. Here we can see that one image pass through the convolutional layers only once but we can get all the features of patches from multiple scales. This decrease calculation time greatly.

Next, we need to pool the activations of these multiple patches to form the second and third levels by single feature vectors of reasonable dimensionality. Here we introduce Vectors of Locally Aggregated Descriptors (VLAD) as our method. At each level, the dimension of the features we extract from respective patches is 4096, which is so high. So in order to make computation more efficient, we use PCA to reduce them to 500 dimensions. For each level, we learn a separate k-means codebook for each level with k = 100 centers. The VLAD descriptor (soft assignment version from [15])is constructed by assigning each patch to its r nearest cluster centers and aggregating the residuals of the patches minus the center. For each of the k clusters, the residuals (vector differences between descriptors and cluster centers) are accumulated, and thesums of residuals are concatenated into a single k*500 dimensional descriptor. Following [13], we power- and L2-normalize the pooled vectors.

Obviously, the dimension of the descriptor is too high for many large-scale applications, so we further perform PCA on the pooled vectors to make 4096 dimensions vectors.Finally, given the original 4096-dimensional feature vector from level one and the two 4096-dimensional pooled PCA-reduced vectors from levels two and three, we rescale them to unit norm and concatenate them to form our final image representation.

Experiment

Pretraining. We pre-trained our DNNs network on the training dataset of ImageNet 2012. Our training details follow the practices of previous work [2, 3,7]. We use two sizes (180*180 in addition to 224*224) of the training images in training. With a given image, we resize it so that the

smaller dimension is 256, and a 224*224 crop is picked from the center or the four corners from the entire image. Then we augment the data by horizontal flipping and color altering [2]. Instead of cropping from the entire image, we resize the 224*224 image mentioned above to 180*180. Following the best practice of SPP, we train our model using multi-size training images and multi-level pooling in the SPP-layer. As a comparison, we also train network with single level pooling in the SPP-layer. As our baseline model, we implement the 7-layer network of Zeiler and Fergus's "fast" (smaller) model [3], which produces competitive results with a moderate training time. Our implementation is based on the publicly available code of Caffe. Our experiments are run on a GeForce GTX K40 GPU.

Results on image classification. Next we will train a one vs all SVM network. For a given image, we resize the image such that min(w; h) = 480, and extract the feature maps from the entire image as the level features. To get the level2 and level3 features, we fix the patch sizes, one for 256*256 and one for 128*128, with the stride of 64 pixels. And now we get a 4096 * 3 dimensions vector through our DNNs pre-trained in above section. In all of the following experiments, we train classifiers using the linear SVM implementation from the INRIA JSGD package [16]. We fix the regularization parameter to 10^{-5} and the learning rate to 0.2, and train for 100 epochs.

Table 1 shows our results on the SUN397 dataset. From the results we can see that our method with multi-level works better than the above work of MOP-VLAD, while the singe level one worse than MOP-VLAD. Of course all these three works better than published state-of-the-art results. Our method exceed traditional method using a combination of artificial features just like works done by Xiao et al. [17] and Fisher Vectors [18]. The reason why the single level one behaves worse than the MOP is that when project the image patches onto feature maps, a lot of details may be lost. However, the multi-level network not only makes up for the loss of the detail, but also adds additional information. So we are able to achieve 52.23% accuracy which is better than the MOP-CNN. Another reason it's get better, I believe is that we can get features from the entire image, thus keeping more information.

Complexity and Running Time. Comparing with the MOP-CNN, our method is much faster with comparable accuracy. Given an image, using the MOP we need repeatedly applies the deep convolutional network to n windows per image, it is time consuming, where n is the patch numbers. While in our method, we extract the feature maps from the entire image only once. Then we apply the spatial pyramid pooling on each candidate window of the feature maps to pool a fixed-length representation of this window. Because the time-consuming convolutional network is only applied once, our method can run orders of magnitude faster. In practice, MOP takes 4.96s per image, while our version takes only 0.24s per image. So ours is 20 faster than MOP-CNN.

Table 1. Results of scene recognition on SUN397.

Method	Feature dimension	accuracy
MOP-CNN	12288	51.98
Ours(single-level)	12288	49.78
Ours(multi-level)	12288	52.23
Xiao et al. [17]	-	38
FV [18]	256000	47.2

Conclusions and future work.

A network with a SPP-layer, can handle different scales, sizes and aspect rations. These important issues received little consideration in the development of deep networks. So here we demonstrate the generalization ability of SPP again. Another way, traditional method for features combination should be paid more attention. In the future work, we will apply method to the domain of object detection and image retrieval.

References

[1] Le Cun, B. Boser, et al. "Handwritten digit recognition with a back-propagation network." Advances in neural information processing systems. 1990.

[2] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks."Advances in neural information processing systems. 2012.

[3] Zeiler, Matthew D., and Rob Fergus. "Visualizing and Understanding Convolutional Neural Networks." arXiv preprint arXiv:1311.2901 (2013).

[4] Sermanet, Pierre, et al. "Overfeat: Integrated recognition, localization and detection using convolutional networks." arXiv preprint arXiv:1312.6229 (2013).

[5] Chatfield, Ken, et al. "Return of the devil in the details: Delving deep into convolutional nets." arXiv preprint arXiv:1405.3531 (2014).

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in CVPR, 2014.

[7] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” in ECCV, 2014, pp. 346–361.

[8] Sivic, Josef, and Andrew Zisserman. "Video Google: A text retrieval approach to object matching in videos." Computer Vision, 2003.Proceedings.Ninth IEEE International Conference on. IEEE, 2003.【bow】

[9] K. Grauman and T. Darrell, “The pyramid match kernel: Discriminative classification with sets of image features,” in ICCV, 2005.

[10] S. Lazebnik, C. Schmid, and J. Ponce, “Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories,” in CVPR, 2006.

[11] Lee, H., Grosse, R., Ranganath, R., Ng, A.Y.: Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: ICML. (2009) 609-616.

[12] Gong, Yunchao, et al. "Multi-scale orderless pooling of deep convolutional activation features." Computer Vision–ECCV 2014.Springer International Publishing, 2014.392-407.

[13] Jégou, Hervé, et al. "Aggregating local descriptors into a compact image representation." Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on.IEEE, 2010.

[14] Jia, Y.: Caffe: An open source convolutional architecture for fast feature embedding. http://caffe.berkeleyvision.org/ (2013).

[15] Bergamo, A., Sinha, S.N., Torresani, L.: Leveraging structure from motion to learn discriminative codebooks for scalable landmark classification. In: Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition. CVPR'13 (2013).

[16] Akata, Zeynep, et al. "Good practice in large-scale learning for image classification." Pattern Analysis and Machine Intelligence, IEEE Transactions on 36.3 (2014): 507-520.

[17] Xiao, Jianxiong, et al. "Sun database: Large-scale scene recognition from abbey to zoo." Computer vision and pattern recognition (CVPR), 2010 IEEE conference on.IEEE, 2010.

[18] Sánchez, Jorge, et al. "Image classification with the fisher vector: Theory and practice." International journal of computer vision 105.3 (2013): 222-245.