Multipath Convolutional-Recursive Neural Networks for Object Recognition

Xiangyang Li[1,2], Shuqiang Jiang[2], Xinhang Song[2], Luis Herranz[2], and Zhiping Shi[1]

[1] Capital Normal University, College of Information Engineering, Beijing, China
{lixiangyang806,shizhiping}@gmail.com
[2] Key Lab of Intelligent Information Processing, Institute of Computing Tech., Beijing, China
sqjiang@ict.ac.cn,{xinhang.son,luis.herranz}@vipl.ict.ac.cn

Abstract. Extracting good representations from images is essential for many computer vision tasks. While progress in deep learning shows the importance of learning hierarchical features, it is also important to learn features through multiple paths. This paper presents Multipath Convolutional-Recursive Neural Networks(M-CRNNs), a novel scheme which aims to learn image features from multiple paths using models based on combination of convolutional and recursive neural networks (CNNs and RNNs). CNNs learn low-level features, and RNNs, whose inputs are the outputs of the CNNs, learn the efficient high-level features. The final features of an image are the combination of the features from all the paths. The result shows that the features learned from M-CRNNs are a highly discriminative image representation that increases the precision in object recognition.

Keywords: Multiple paths, convolutional neural networks, recursive neural networks, classification.

1 Introduction

Visual recognition is a major focus of research in computer vision and machine learning. In the past few years, many works have been done on visual descriptors [1-4]. These methods have shown robustness against visual complexities, such as changes in scale, illumination, affine distortions, and pose variations [5]. For example the SIFT operator [1], which can be understood and generalized as a way to go from pixels to patch descriptors, applies oriented edge filters to small paths and determines the dominant orientation through a winner-take-all operation. Finally, the resulting sparse vectors are added over large patches to form local orientation histograms. Designing algorithms to get meaningful features is important but requires deep domain knowledge, and it is often difficult to expand the scope and ease of its applicability.

To overcome this weakness of feature engineering, feature learning learns transformations of the data that make it easier to extract useful information when building classifiers or other predictors [6]. There are a variety of works about feature learning, such as hierarchical sparse coding [7], deep autoencoders [8], convolutional deep belief networks [9], and convolutional neural networks [10]. These approaches learn image features from raw pixels through multiple feature transforms layers.

Recent work on convolutional-recursive deep learning [11] is very efficient. Combining CNNs to extract low-level features from RGB and depth image, and RNNs to map the learned features into a lower dimensional feature space, the result of the model outperforms a lot of designed features and algorithms. Compared to standard feed forward neural networks with layers of similar size, CNNs have much fewer connections and parameters, thus being easier to train, while their theoretically-best performance is likely to be the same [12]. A single convolutional neural network layer provides useful translational invariance of low level features such as edges and allows part of an object to be deformable to some extent. Recursive neural networks, combine convolution and pooling into one efficient and hierarchical operation, project inputs into a lower dimensional space through multiple layers.

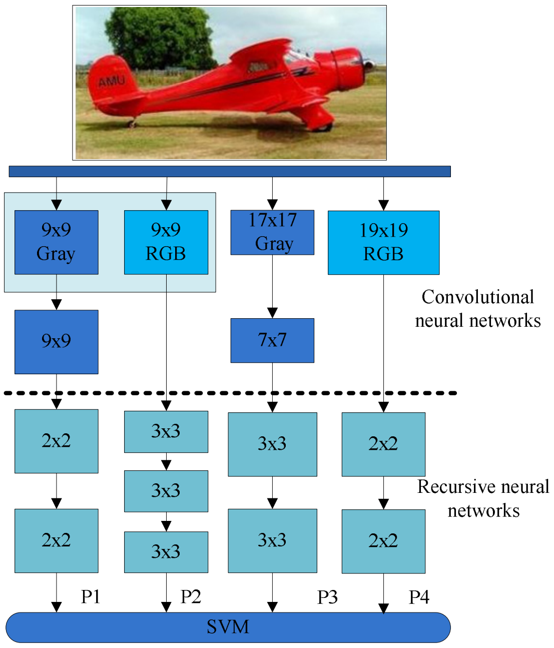

Fig. 1. Architecture of M-CRNNs

Concerned about the multi-facet nature of visual structures, Bo et al. [13] proposed multipath hierarchical matching pursuit. Discriminative structures, which we want to extract in feature learning procedures, may appear at varying scales with varying amounts of spatial and appearance invariance. Multipath hierarchical matching pursuits capture the heterogeneity, and build it into the learning architecture.

Motivated by this, and using the simplicity and high efficiency of convolution-recursive neural networks (CRNNs), we propose multipath convolutional-recursive neural networks (M-CRNNs). The overview of this framework is illustrated in Fig. 1.

A single path M-CRNNs is built upon a single-path CRNNs and learns features through many pathways on multiple bags of patches of varying size, by encoding each patch through multiple paths with a varying number of layers. The M-CRNNs architecture is important as it significantly and efficiently expands the richness of the image representation and contributes to improvement to image classification.

The remainder of this paper is organized as follows. In section 2, we propose M-CRNNs model and present learning procedures in detail. In section 3, experiment results are presented and discussed. Finally we conclude the paper in section 4.

2 Multipath Convolutional-Recursive Neural Networks

In this section, we describe our multipath convolutional-recursive neural networks. We first learn CNNs filters and then feed image patches into CNNs layers. The invariant features extracted from CNNs are given to recursive neural networks. RNNs learn higher order features that represent the image. All features from different paths are concatenated to form the final features that can then be used to classify images.

2.1 Convolutional Neural Networks

The main idea of CNNs is to convolve filters over the input image to extract distinctive features. We convolve an image of size (height and width) d_L with N square filters of size d_P, resulting in N filter responses, each of dimensionality $d_L - d_P + 1$. We then pool them with square regions of size d_l and a stride size of s, to obtain a pooled response with width and height equal to $r = (d_L - d_P + 1 - d_l) / s + 1$. So the output Y of the CNNs layer applied to one image is a $N \times r \times r$ dimensional 3D matrix. We apply this same procedure to both RGB images and grayscale images separately.

The structure of our layer of CNNs is similar to the one proposed by Jarrett et al. [14]. Different systems use different strategies to construct filters to extract features from images. For single layer models, filters sometimes are hard-wired, such as Gabor Wavelets [15]. For multi-stage vision systems, there is no prior knowledge that would conduct us to design sensitive filters for the second or higher layers. Hence filters are learned using supervised or unsupervised methods in many models. We use Predictive Sparse Decomposition (PSD) [16] and k-means to learn the filters.

A single stage of CNNs model includes four layers: filter bank layer, rectification layer, local contrast normalization layer and average pooling and subsampling layer. In the filter bank layer, the input is a 3D array with n_1 2D feature maps of size $n_2 \times n_3$. Each element is denoted as x_{ijt}, and each map is denoted as x_i. The filter bank has N filters. Each filter k_{ij} has a size of $l_1 \times l_2$, and transforms the input feature map x_i into the output feature map y_i. The module computes:

$$y_j = g_j \cdot f\left(\sum_i k_{ij} \otimes x_i\right) \tag{1}$$

where f is a nonlinear function such as tanh, $\otimes$ is the 2D discrete convolutional operator and g_j is a trainable scalar coefficient. The output of the filter bank layer is a 3D array y composed of N feature maps of size $m_1 \times m_2$. We have $m_1 = n_2 - l_1 + 1, m_2 = n_3 - l_2 + 1$. The rectification layer simply applies the function: $y_{ijk} = | x_{ijk} |$. The third layer performs local subtractive and divisive normalizations, enforcing a sort of local competition between adjacent features in a feature map, and between features at the same spatial location in different maps. The objective of average pooling and subsampling layer is to build robustness to small distortions, which plays the same role as the complex cells. Output value is $y_{ijk} = \sum_{pq} w_{pq} \cdot x_{i,j+p,k+q}$, where w_{pq} is uniform weight window ("boxcar filter"). Each output is then subsampled spatially by size S horizontally and vertically.

For multi-layer CNNs models, the higher layer feature extractor is fed with the results of its prior layer. The feature extraction procedure is the same as in the single-layer model.

2.2 Recursive Neural Networks

Recursive neural networks have been used in natural language parsing [18]. The main idea of RNNs is to learn hierarchical feature representations by repeatedly applying the same neural network recursively in a tree structure.

Unlike previous works, the structure of the RNNs we use allows each layer to merge blocks of adjacent vectors instead of only pairs of vectors. Each image is denoted as a 3D matrix $X \in \mathbb{R}^{K \times r \times r}$, and the columns are K-dimensional. We define a block to be a list of adjacent column vectors which are merged into a parent vector $p \in \mathbb{R}^{K}$. In our work, we only consider square blocks. Blocks are of size $K \times w \times w$. For example, if we merge vectors in a block with $w = 5$, the input of one merging procedure is a structure of total size $K \times 5 \times 5$. In general, we have w^2 vectors in each block. The neural network for computing the parent vector is:

$$p = f\left(W \begin{bmatrix} x_1 \\ \cdot \\ \cdot \\ \cdot \\ x_{w^2} \end{bmatrix} \right) \tag{2}$$

where the parameter matrix $W \in \mathbb{R}^{K \times b^2 K}$, f is a nonlinear function such as tanh. Eq. (2) will be applied to all blocks of vectors in X with the same weights W. The structure of recursive neural networks is illustrated in Fig. 2. In our work N RNNs are used, each of which output a k-dimensional vector. So the output of RNNs is a NK -dimensional vector.

2.3 Multipath Convolutional-Recursive Neural Networks

In many visual recognition models, images were regarded as unordered collections of small patches, such as in the bag-of-features model. These traditional models do not consider the spatial positions of patches and relationships between patches, which are useful for visual recognition. The spatial pyramid pooling strategy overcomes this drawback by organizing patches into spatial cells at multiple levels and then concatenating them to a long feature vector [19]. The compelling advantages of spatial pyramid model are:

- The pyramid model generates a larger number of features and decreases the chance of overfitting.
- Adds local invariance and stability of the learned features using spatial pooling.
- It can capture invariant structures at multiple levels, because of different pooling sizes in the pyramid.

A single path convolutional-recursive neural network has the first two advantages. To combine the advantage of multiple patch sizes and the strength of hierarchical architectures, our multipath convolutional-recursive neural networks learn image features in multiple pathways, varying patch sizes and number of layers. Different paths with different receptive fields can capture different features. This combination of layers contains more information.

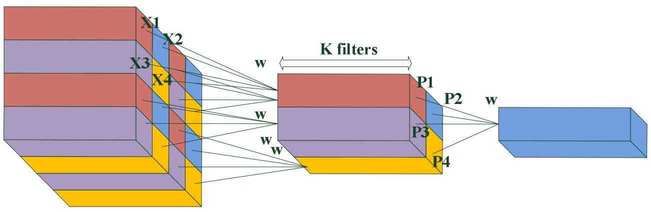

Fig. 2. Recursive neural network

The overview framework can be seen in Figure 1. Images of different size are put into CNNs to get low-level features. Each path corresponds to a specific patch size and number of layers. The final layer of each path is a feature vector for the whole image. All the features are then concatenated and used by a svm classifier for object recognition. The inputs of P1 and P3 are grayscale images, and the inputs of P2 and P4 are RGB images, so the features we finally get not only contain gray information, but also contain color information.

3 Experimental Results

In this section, we report the result of M-CRNNs for object category recognition. All our experiments are carried out on Caltech-101. The dataset contains 9,144 images of 101 object categories and one background category. Following the standard experiment setting, we trained models on 30 images and tested on no more than 50 images per category.

For the grayscale path, we first pre-process images with a procedure similar to [20]. First, we convert the image to grayscale and resize it with its largest dimension set to 151 pixels, while preserving its aspect ratio. Second, we subtract its mean deviation. Third, we apply divisive normalization. Finally, the image is zero-padded to 143×143 pixel. For the RGB path, the image was resized to 143×143.

3.1 Random Filters and Unsupervised Learned Filters

While convolutional pooling architectures can be inherently frequency selective and translation invariant, even with random weights [17], the filters in our model can be set to random values and kept fixed. In our framework, the filters convoluted with the RGB images are learned by k-means, and the ones convoluted with the grayscale images are learned by PSD. We compared the performance of learned filters and random-fixed filters in Table 1.

We estimate the accuracies of random filters and filters learned by PSD on the model. PP1 denotes: one-layer CNNs with 256 filters of size 9×9, 10×10 boxcar, 5 ×5 down-sampling (ds), and three-layer RNNs. PP2 denotes two-layer CNNs, the same configuration as P3 (as described in 3.2). Prefix R denotes the filters are randomly initialized, and PSD means the filters are learned in an unsupervised way using of PSD. As we can see in Table 1, for one-layer CNNs, the performance of the leaned filters and the randomly initialed one are nearly the same, but in two-layer CNNs, the learned ones are much better. So, the filters in our work are leaned through unsupervised training.

Table 1. Random filters and filters learned

R-PP1	52.6995	R-PP2	56.9440
PSD-PP1	52.5976	PSD-PP2	65.7046

3.2 Architecture of M-RCNNs

We outline the four paths used in our experiment with multipath convolutional-recursive neural networks in our experiments.

Path one (P1) is done on gray images. First images are pre-processed, two stages of CNNs and two stages of RNNs are performed to extract the features. The first CNN stage has 64 filters of size 9×9, 10×10 boxcar filter, and 5×5 ds. The second CNN stage is composed of 256 output feature maps, each of which combines a random subset of 16 feature maps from the first one. So the total number of convolutional filters is 256×16=4096. The average pooling model uses 6x6 boxcars filter with a 4×4 downsampling step. The output of a 3D matrix of size 256×4×4 is fed to a two- layer CNN with 128 RNNs. The final output of P1 is a 256×128 vector.

Path two (P2) is done on RGB images. First, 400000 random patches of size 9×9 ×3 are extracted. Then the patches are normalized and whitened. Pre-processed patches are clustered into 256 centers by running k-means. The boxcar filter and ds of the one layer CNNs are both 5×5. The output feature of size 256×27×27 is fed to a three layers RNNs. The size of the filters is 9×9×3 pixels (see Fig. 3).

Path three (P3) has two layers CNNs and three layers RNNs. The first stage of CNNs is with 64 filters of size 17×17, 7×7 boxcar and 5×5 ds. The second stage of CNNs is with 256×16=4096 filters of size 7×7, 3×3 boxcar and 2×2 ds.

Path four (P4) has one stage CNNs and two stages of RNNs. The size of filters is 19×19×3.

The filters in our architecture are shown in Figure 3. Left are the 64 filters of P1 in the first CNNs layers. In the middle, there are 64 filters in path P3 of the first CNNs layers. Right are the 256 filters in P2.

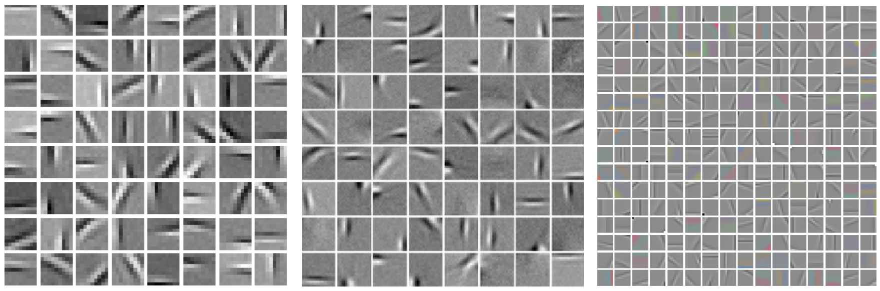

Fig. 3. Visualization of filters

The final features of an image are the concatenated features extracted from all the paths. The accuracy of different paths is shown in Table 2. Generally speaking, P1 works well for object classification due to its hierarchical architecture. Compared with the accuracy 65.5 of [14], the architecture of P1 illustrates that CRNNs works slightly better than single CNNs. Different paths can capture visual invariant features at different scales. By concatenating the entire paths, the combined features have higher accuracy than each single path.

Table 2. Accuracy of each path

P1	66.9610	P4	56.9779
P2	56.1290	P2+P3	71.4092
P3	65.7046	P1+P2+P3+P4	72.1902

3.3 Comparison to Other Methods

In this section we compare our model to related models in literature in Table 3. SVM-KNN [19] is a hybrid of SVM and NN which focus on shape and texture. Soft threshold coding (SIFT+T) [21] and spatial pyramid matching [22] approaches based on SIFT features. Invariant predictive sparse decomposition (ISPD) [23], convolutional deep belief networks (CDBN) [9], convolutional neural networks (CNNs) [14], and deconvolutional networks (DN) [24] are approaches of hierarchical feature learning. Our method outperforms many SIFT-based approaches as well as hierarchical feature learning methods.

Table 3. Classification accuracy on Caltech-101

SIFT+T	67.7	CDBN	65.4
SPM	64.4	CNNs	65.5
SVM-KNN	66.2	DN	66.9
IPSD	65.5	M-CRNNs	72.1

4 Conclusion

We have proposed multiple path convolutional-recursive neural networks to learn meaning full representations from raw images. The proposed method combines convolutional neural networks and recursive neural networks, and learns the filters in an unsupervised way, which allows for parallelization and high speeds. Learning features through several pathways with varying number of layers, makes the visual features capture invariances at different scales. The architecture feeds grayscale patches and RGB patches to the CNNs, so it contains gray information and color information. Our future work will aim at the combination of multiple paths and deep convolutional neural networks.

Acknowledgements. This work was supported in part by National Natural Science Foundation of China: 61322212 and 61350110237, in part by the Chinese Academy of Sciences Fellowships for Young International Scientists: 2011Y1GB05, in part by the Key Technologies R&D Program of China under Grant no. 2012BAH18B02.

References

1. Lowe, D.: Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2), 91–110 (2004)
2. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR, San Diego (2005)
3. Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., Gong, Y.: Locality-constrained linear coding for image classification. In: CVPR, San Francisco (2010)
4. Bo, L., Ren, X., Fox, D.: Kernel descriptors for visual recognition. In: NIPS, Vancouver (2010)
5. Lobel, H., Vidal, R., Soto, A.: Hierarchical joint Max-Margin learning of mid and top level representations for visual recognition. In: ICCV, Sydney (2013)
6. Bengio, Y., Courville, A., Vincent, P.: Representation learning: a review and new perspectives. IEEE Transaction on Pattern Analysis and Machine Intelligence 35(8), 1798–1828 (2013)

7. Yu, K., Lin, Y., Lafferty, J.: Learning image representations from the pixel level via hierarchical sparse coding. In: CVPR, Colorado Springs (2011)
8. Le, Q., Ranzato, M., Monga, R., Devin, M., Chan, K., Gorrado, G., Dean, J., Ng, A.: Building high-level features using large scale unsupervised learning. In: ICML, Scotland (2012)
9. Lee, H., Grosse, R., Ranganath, R., Ng, A.: Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In: ICML, Montreal (2009)
10. Lawrence, S., Giles, C., Tsoi, A., Back, D.: Face recognition: a convolutional neural-network approach. IEEE Transaction on Neural Networks 8(1), 98–113 (1997)
11. Socher, R., Huval, B., Bhat, B., Manning, D., Ng, A.: Convolutional-recursive deep learning for 3D object classification. In: NIPS, Nevada (2012)
12. Krizhevsky, A., Sutskever, I., Hinton, G.: ImageNet classification with deep convolutional neural networks. In: NIPS, Nevada (2012)
13. Bo, L., Ren, X., Fox, D.: Multipath sparse coding using hierarchical matching pursuit. In: CVPR, Portland (2013)
14. Jarrett, K., Kavukcuoglu, K., Ranzato, M., LeCun, Y.: What is the best multi-stage architecture for object recognition. In: ICCV, Xi'an (2009)
15. Serre, T., Wolf, L., Poggio, T.: Object recognition with features Inspired by visual cortex. In: CVPR, San Diego (2005)
16. Kavukcuoglu, K., Ranzato, M., LeCun, Y.: Fast inference in sparse coding algorithm with applications to object recognition. Technical report, Computational and Biological Learning Lab, Courant Institute, NYU (2008)
17. Saxe, A., Koh, P., Chen, Z., Bhand, M., Suresh, B., Ng, A.: On random weights and unsupervised feature learning. In: ICML, Washington (2011)
18. Socher, R., Maning, C., Ng, A.: Learning continuous phrase representation and syntactic parsing with recursive neural networks. In: NIPS, Vancouver (2010)
19. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: spatial pyramid matching for recognizing natural scene categories. In: CVPR, New York (2006)
20. Pinto, N., Cox, D., DiCarlo, J.: Why is real-world visual object recognition hard. PLOS Computational Biology 4(1) (2008)
21. Coates, A., Ng, A.: The importance of encoding versus training with sparse coding and vector quantization. In: ICML, Washington (2011)
22. Zhang, H., Berg, A., MaireM.,Malik, J.: SVM-KNN: discriminative nearest classification for visual category recognition. In: CVPR, New York (2006)
23. Kavukcuoglu, K., Ranzato, M., Fergus, R., LeCun, Y.: Learning invariant features through topographic filter maps. In: CVPR, Florida (2009)
24. Zeiler, M., Krishnan, D., Taylor, G., Fergus, R.: Deconvolutional networks. In: CVPR, San Francisco (2010)