LEARNING ROTATION INVARIANCE IN DEEP HIERARCHIES USING CIRCULAR SYMMETRIC FILTERS

Dhruv Kohli, Biplab Ch Das, Viswanath Gopalakrishnan, Kiran Nanjunda Iyer

Samsung R&D Institute, Bangalore, India
{dhruv.kohli, biplab.das, viswanath.gk, kiran.iyer}@samsung.com

ABSTRACT

Deep hierarchical models for feature learning have emerged as an effective technique for object representation and classification in recent years. Though the features learnt using deep models have shown lot of promise towards achieving invariance to data transformations, this primarily comes at the expense of using much larger training data and model size. In the proposed work we devise a novel technique to incorporate rotation invariance, while training the deep model parameters. The convolution weight parameters in the network architecture are constrained to exhibit circular symmetry resulting in "rotation equivariance" of output feature maps. Rotation invariance is further achieved by max-pooling of the feature maps later in the hierarchy. We also show that by incorporating circular symmetry constraint into the training loss function, rotation invariance can be achieved with-in deep neural network framework with much lesser training data and model parameters. Our experiment results evaluated on rotated MNIST dataset further objectively validate the contribution.

***Index Terms*—** rotation equivariance and invariance, circular symmetric kernels, convolution neural networks.

1. INTRODUCTION

Learning robust features for image representation is the fundamental building block to scene and object understanding and hence has been an important topic of interest in computer vision research. Features exhibiting rotation invariance has always induced interest in the research community as object rotation is a common phenomenon occurring in images and videos. Hand designed features for rotation invariance has attracted some early interest as can be seen in the works of [1] and [2]. However, as it is commonly agreed now, specially designed hand engineered features do not adapt well to all different kinds of contexts from which the data might originate. Deep neural networks such as [3] and [4], learn the best features suited for object representation and classification from very large set of labeled databases and thus overcome the limitations of hand-engineered features. The transformations in data with respect to rotation and scale is handled reasonably well using deep networks when millions of parameters learn to capture different probable variations in data. The capacity to learn to capture different data transformations comes at the increased cost of training resulting from data augmentation. It also means increase in the size of model parameters which will resultantly increase the memory and performance specifications of recognition systems.

Multi-Column Deep Networks discussed in [5] use biologically plausible, wide and deep neural network architecture in which a separate column is trained for each data transformation. The predictions of each column are averaged to produce the final prediction. Laptev et al. [6] formulated features in convolutional neural networks to be transformation-invariant by using parallel siamese architectures and applying a transformation invariant pooling operator on their outputs. The proposed algorithm in [6] accumulates responses from original image and its multiple rotational transformations, and takes the maximum among them resulting in rotation invariant features. In this manner [6] uses smaller number of parameters than data augmentation techniques [7] as it learns to omit the not so useful information from the augmented data. However the parallel architectures still need to be employed for multiple rotational transformations of data in the initial layers of training and testing, which makes the computations unnecessarily high.

Our proposed method attempts to avoid learning using data augmentation as in [7] and data transformations as in [6], while trying to achieve the objective of rotation invariance. In this manner the proposed method will prove to be computationally efficient than [6] and more accurate than [7] while using lesser training data and parameter budget. The primary contributions of the paper can be described as the following two-fold,

1. We show that a convolution using circular symmetric kernel holds the property of "rotation equivariance" and

further discuss the method to incorporate circular symmetry in the training error loss function of CNN architecture.
2. We formulate the CNN architecture which can convert the "rotation equivariant" feature maps to rotation invariant classification.

The rest of the paper is organized as follows. Section 2 describes in detail about the proposed architecture and Section 3 details the experimental results followed by Section 4 which concludes the proposed work.

2. PROPOSED ARCHITECTURE

In this section, we show that circular symmetric kernel leads to "rotation equivariance" of output feature maps (section 2.1). Then, we modify the loss used while training CNN so as to incorporate circular symmetry in the kernels of its convolution layers (section 2.2) and finally describe the proposed rotation invariant CNN architecture (section 2.3). For readability, we follow the notation, $p_{(a,\xi)} = p(a\cos\xi, a\sin\xi)$, whenever possible.

2.1. Rotation Equivariant Kernel

In 2-dimensional image processing terms, the continuous convolution integral may be expressed as,

$$g(x_o, y_o) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)h(x_o - x, y_o - y)\,dxdy \tag{1}$$

After converting the above equation from cartesian to polar coordinates, we obtain,

$$g_{(r_o,\theta_o)} = \int_{0}^{2\pi}\int_{0}^{\infty} f_{(r,\theta)}\, h(r_o\cos\theta_o - r\cos\theta, r_o\sin\theta_o - r\sin\theta)\, rdrd\theta \tag{2}$$

where h represents the image, g represents the output feature map and f represents a circular symmetric kernel with the property,

$$\forall \alpha, \beta : f_{(r,\alpha)} = f_{(r,\beta)} \tag{3}$$

On rotating the image h with respect to origin by an angle ϕ in anticlockwise direction, we get the rotated image h^{rot} and the following relation can be verified between h^{rot} and h,

$$\forall \alpha, r' : h^{rot}_{(r',\alpha+\phi)} = h_{(r',\alpha)} \tag{4}$$

Also, the rotation matrix representing rotation of angle ϕ transforms the coordinates $(r_o\cos\theta_o - r\cos\theta, r_o\sin\theta_o - r\sin\theta)$ to $(r_o\cos(\theta_o+\phi) - r\cos(\theta+\phi), r_o\sin(\theta_o+\phi) - r\sin(\theta+\phi))$. Hence, we obtain,

$$h^{rot}(r_o\cos(\theta_o+\phi) - r\cos(\theta+\phi), r_o\sin(\theta_o+\phi) - r\sin(\theta+\phi)) = h(r_o\cos\theta_o - r\cos\theta, r_o\sin\theta_o - r\sin\theta) \tag{5}$$

So, by replacing equation 5 in equation 2 and using the circular symmetric property of kernel f (equation 3), we obtain,

$$g_{(r_o,\theta_o)} = \int_{0}^{2\pi}\int_{0}^{\infty} f_{(r,\theta+\phi)}\, h^{rot}(r_o\cos(\theta_o+\phi) - r\cos(\theta+\phi), r_o\sin(\theta_o+\phi) - r\sin(\theta+\phi))\, rdrd\theta \tag{6}$$

Now, applying the transformation $\theta \leftarrow \theta + \phi$, we obtain,

$$g_{(r_o,\theta_o)} = \int_{-\phi}^{2\pi-\phi}\int_{0}^{\infty} f_{(r,\theta)}\, h^{rot}(r_o\cos(\theta_o+\phi) - r\cos\theta, r_o\sin(\theta_o+\phi) - r\sin\theta)\, rdrd\theta \tag{7}$$

Since, for a given radius, we are integrating over the complete circular ring and by the circular symmetry of kernel (equation 3), replacing the limits of the integral with respect to θ, from 0 to 2π, doesn't change the value of the integral. Finally, by the definition of convolution in polar coordinates (equation 2), the integral evaluates to $g^{(rot)}_{(r_o,\theta_o+\phi)}$ where $g^{(rot)}$ is the feature map obtained after applying circular symmetric kernel over the rotated image h^{rot}. So, we have shown,

$$g_{(r_o,\theta_o)} = g^{(rot)}_{(r_o,\theta_o+\phi)} \tag{8}$$

From equation 8, it can be concluded that convolving circular symmetric kernel over the rotated image results in the rotation of the feature map which is generated by convolution of the kernel with the non-rotated image. Also, the angle of rotation of output feature map is equal to the angle with which the input image was rotated. Such circular symmetric kernels are thus called rotation equivariant kernels. An illustration of the output feature maps produced by convolving such kernels over an image and its rotated version is shown in Figure 1.

2.2. Achieving Circular Symmetry In CNN Architecture

For further discussion, we focus on the CNN architecture. We use Adam [8] variant of SGD to optimize for the parameters

(a) (b)

Fig. 1. Output feature maps after convolving circular symmetric kernels over an input image (a) and its rotation (b).

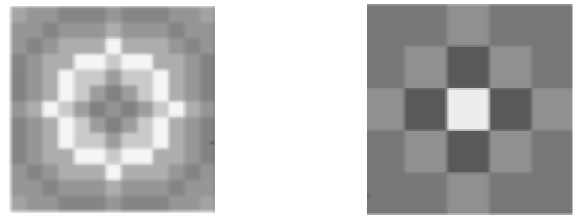
of the network so as to minimize

$$\Theta = \underset{\Theta}{\operatorname{argmin}} \left[\frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(x_i, \Theta) \right. \\ \left. + \lambda \sum_{l=1}^{L} \sum_{n=1}^{N_l \times d_l} \sum_{r=1}^{R_{l,n}} \sum_{\substack{\theta_i \\ \theta_i, \theta_j \in S_{l,n,r}}} \sum_{\theta_j} (\theta_i - \theta_j)^2 \right] \tag{9}$$

where x_1, ..., x_N is the training dataset, λ is a hyperparameter, L is the number of convolution layers, N_l is the number of kernels and d_l is the depth of each kernel in l_{th} convolution layer all of which are constrained with circular symmetry, $R_{l,n}$ is the maximum possible radius of the circular ring and $S_{l,n,r} \subseteq \Theta$ is the set of parameters on circular ring of radius r in the n_{th} constrained kernel of l_{th} convolution layer. Note that the number of kernels N_l is equal to the number of output channels and the depth d_l is equal to the number of input channels of l_{th} convolution layer. The first term in the optimization objective represents the mean prediction error on the training dataset where $\mathcal{L}$ is generally the cross entropy loss for multi-class classification task. The second term represents the circular symmetry constraint which is computed as the sum of squared euclidean distance between each pair of parameters lying on a circular ring of radius r in the n_{th} constrained kernel of the l_{th} convolution layer, summed over circular rings of all possible radii in each constrained kernel in each convolution layer. Minimizing the second term results in kernels with same values in circular rings, thus forming circular symmetric or rotation equivariant kernels. Examples of such kernels are shown in Figure 2. Also, higher value of λ leads to higher penalty due to second term.

2.3. Rotational Invariant CNN Architecture

With the CNN architecture shown in Figure 3 in which each kernel in each convolution layer is constrained with circular symmetry, the feature maps produced after convolution layer (f) will rotate by the same angle with which the input image gets rotated by the argument presented in section 2.1 and therefore, by applying a global maxpooling over these feature maps will result in same output for both the rotated and non-rotated input. Thus, the vector of scalars obtained after global maxpooling layer (g) is invariant to the rotation of the input image.



size: 13×13 size: 5×5

Fig. 2. Circular symmetric kernels

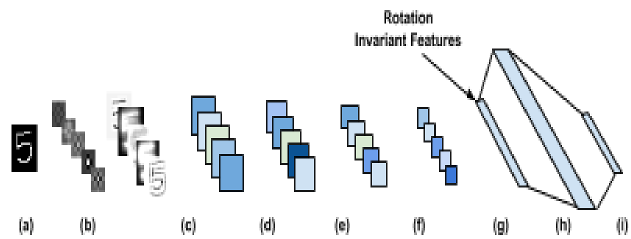


Fig. 3. Network topology description. Input image x (a) is passed through a sequence of convolution layers containing circular symmetric kernels (b,f) and a global maxpooling layer (g) until the vector of scalars is not achieved. This vector serves as an input to a fully connected layer (h) possibly with dropout [9] and further propagates to the network output (i).

3. EXPERIMENTS

To verify the ability of our proposed rotation invariant CNN architecture, we considered the problem of predicting digit class on MNIST dataset (MNIST-ORIG [10]) and on one of its variant in which the digits are rotated at an angle randomly chosen between 0 and 2π (MNIST-ROT [11]). In order to demonstrate the effectiveness of our proposed idea, we used two versions of the architecture shown in Figure 3, one with the circular symmetry constraint over each kernel in each convolution layer, making the architecture invariant to rotation of input and other with no constraint on kernels. The circular symmetry constraint on each kernel of a convolution layer is implemented in the form of a regularizer in Keras [12] with Theano [13] backend. The topology of the network used in our experiments is shown in table 1. The network takes 32×32 padded grayscale image as input and predicts the digit class corresponding to the input image.

We separately trained the two models on both datasets using Adam algorithm [8] for 1000 epochs with a batch size of 200 and a dropout [9] for fully connected layer. Both models have almost equal accuracy on test set of MNIST-ORIG when trained on train set of MNIST-ORIG and the same holds true for training and testing on MNIST-ROT dataset. When trained on train set of MNIST-ROT and tested on test set of MNIST-ORIG, and similarly, when trained on train set of MNIST-ORIG and tested on test set of MNIST-ROT, the accuracy achieved by the model constrained with circular symmetry is almost twice the accuracy achieved by the model without any constraint. The final accuracies of the model with and without circular symmetry constraint are presented in table 2. These observations suggest that the kernels learnt by the model with proposed circular symmetry constraint are robust to the rotational variations in the dataset leading to much better generalization over the unseen data as compared to the model without circular symmetry constraint.

The method described in [6], computes 24 rotations of the input image sampled uniformly from 0 to 2π and then accumulates responses after applying a siamese network over each of them, making it computationally expensive. The accumulation of responses is done using TI-Pooling operator [6] and the accumulated responses are, then, propagated to the digit class prediction using fully connected layers with dropout [9]. The accuracy achieved by our model with proposed circular symmetry constraint (rotation invariant CNN) on test set of MNIST-ROT is 94.31% (error of 5.69%) which is lesser than the accuracy of 97.8% (error of 2.2%) achieved by the method proposed in [6]. But, at the same time, our constrained model requires no augmentation of training data, hence, lesser parameters and no computation of rotational transformations of input, hence, lesser computation. In comparison to the method described in [6], our proposed rotation invariant CNN requires $4.5\times$ lesser parameters and $3\times$ lesser flops. The exact values are showcased in table 3.

Layer	**Parameters and output channel size**
input	size: 32×32, channel: 1
convolution	kernel: 5×5, channel: 40
relu	
convolution	kernel: 5×5, channel: 40
relu	
convolution	kernel: 5×5, channel: 40
relu	
convolution	kernel: 5×5, channel: 80
relu	
convolution	kernel: 5×5, channel: 80
relu	
global max pooling	
linear	channel: 5120
relu	
dropout	rate: 0.5
linear	channel: 10
softmax	

Table 1. The topology of the network used in our experiments.

Trained on	**Architecture**	**Test Accuracy, %**	
		MNIST-ROT	**MNIST-ORIG**
MNIST-ROT	Without Circular Symmetry	95.00	50.87
	With Circular Symmetry ($\lambda = 1$)	94.31	94.38
	With Circular Symmetry ($\lambda = 3$)	94.08	94.22
MNIST-ORIG	Without Circular Symmetry	33.46	99.42
	With Circular Symmetry ($\lambda = 1$)	50.66	99.43
	With Circular Symmetry ($\lambda = 3$)	62.41	99.08

Table 2. Accuracies obtained by the proposed CNN architecture with and without circular symmetric kernels over the test set of MNIST-ORIG [10] and MNIST-ROT [11] datasets.

Method	**Error, % MNIST-ROT**	**Parameters, M**	**Flops, M**
TI-Pooling [6]	2.2	3.47	248
Circular Symmetric Kernel (ours)	5.69	0.78	84

Table 3. Comparison of our model with circular symmetry constraint with TI-Pooling method described in [6]. Note that the flop computation for TI-Pooling excludes the flops used for computing 24 rotations of the input image.

4. CONCLUSIONS

We have proposed a novel method for learning rotation invariant features within framework of deep hierarchical networks by introducing circular symmetry constraints to the training error loss function of CNN architecture. It has been experimentally shown that constraining convolutional filter kernels in early stages of the CNN architecture improves the generalization performance of the systems when data transformations like rotation takes place. Effectively, the proposed system also shows that fusing specific design conditions intelligently to feature learning techniques can yield better results than allowing the system to learn in an unconstrained manner. We plan to further investigate how the proposed circular symmetric kernels can improve the performance of text and object recognition in terms of accuracy as well as computation.

5. REFERENCES

[1] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce, “Semi-local affine parts for object recognition,” in *British Machine Vision Conference (BMVC’04)*. The British Machine Vision Association (BMVA), 2004, pp. 779–788.

[2] Kristian Sandberg and Moorea Brega, “Segmentation of thin structures in electron micrographs using orientation fields,” *Journal of structural biology*, vol. 157, no. 2, pp. 403–415, 2007.

[3] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Ser-

manet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1–9.

[4] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[5] Dan C. Ciresan, Ueli Meier, and Jürgen Schmidhuber, "Multi-column deep neural networks for image classification," *CoRR*, vol. abs/1202.2745, 2012.

[6] Dmitry Laptev, Nikolay Savinov, Joachim M. Buhmann, and Marc Pollefeys, "Ti-pooling: Transformation-invariant pooling for feature learning in convolutional neural networks," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[7] David A Van Dyk and Xiao-Li Meng, "The art of data augmentation," *Journal of Computational and Graphical Statistics*, 2012.

[8] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[9] Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580*, 2012.

[10] Yann LeCun and Corinna Cortes, "MNIST handwritten digit database," 2010.

[11] Hugo Larochelle, Dumitru Erhan, Aaron Courville, James Bergstra, and Yoshua Bengio, "An empirical evaluation of deep architectures on problems with many factors of variation," in *Proceedings of the 24th international conference on Machine learning*. ACM, 2007, pp. 473–480.

[12] François Chollet, "Keras: Deep learning library for theano and tensorflow," *GitHub repository https://github.com/fchollet/keras*, 2015.

[13] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints*, vol. abs/1605.02688, May 2016.