DFT-based Transformation Invariant Pooling Layer for Visual Classification

Jongbin Ryu[1], Ming-Hsuan Yang[2], and Jongwoo Lim[1]

[1] Hanyang University
[2] University of California, Merced

Supplementary material

1 Run Time and Number of Parameters

The DFT magnitude pooling method captures shape information by pooling the AC components of the frequency domain in addition to the DC component that average pooling uses. The larger size of pooling output, the number of parameters in the next fully-connected layer increases and thus it requires more computational load. Table 1 shows the run time of the DFT and DFT$^+$ methods against the baseline model. Compared to the baseline models, the DFT magnitude pooling method only requires one additional 2D-DFT operation, and thus the overhead is negligible. It is worth noticing that the change of run time of the DFT magnitude pooling method with respect to the pooling size does not change significantly. In the DFT$^+$ method, due to the extra overhead to compute the responses from mid layers, the run time increases slightly. However, it only adds DFT magnitude pooling layers and thus the run time still remains manageable. The number of parameters of DFT magnitude pooling is N$\times$N-times larger than that of average pooling. As shown in Table 1, the computational overhead is not significant since the convolution operations amount to the most part. A DFT network with a small N achieves much improved performance, and the increase of parameter size is manageable as shown Table 2.

Table 1: Run time of the DFT and DFT$^+$ based models (in milliseconds) with respect to the pooling size. The overhead by the DFT magnitude pooling layer is negligible, and it is manageable for the DFT$^+$ based model. Note that DFT$^+{}_3$ is the largest and most complex network in our experiments.

Networks	Base	DFT			DFT$^+{}_3$		
		N=2	N=4	full	N=2	N=4	full
VGG-VD 16	7.03	7.20	7.25	7.28	8.55	8.64	8.71
ResNet-50	4.15	4.25	4.31	4.38	5.01	5.07	5.15

Table 2: Number of parameters with respect to the pooling size N.

Network	#classes	Baseline	DFT (N=2)	(N=4)
Res-152	1000 (ImageNet)	60.0M	66.1M	90.7M
	67 (MIT Indoors)	58.1M	58.5M	60.2M
VGG 16	Not dependent	138.0M	43.6M	68.8M

Table 3: Performance evaluation of the DFT magnitude pooling and baseline methods on images with large translation and scale variations in complex scenes.

Dataset	Network	S=0.25			S=0.50			S=0.75		
		Baseline	DFT	gain	Baseline	DFT	gain	Baseline	DFT	gain
CUB	AlexNet	21.69	31.64	**+9.94**	50.97	55.20	**+4.23**	60.22	64.00	**+3.78**
	Inception-v3	44.94	60.34	**+15.40**	68.42	75.70	**+7.28**	73.35	79.50	**+6.14**
Caltech 101	AlexNet	46.19	58.37	**+12.18**	75.44	77.63	**+2.19**	83.97	83.78	**-0.20**
	Inception-v3	74.42	78.65	**+4.22**	89.58	91.34	**+1.76**	91.57	93.21	**+1.64**

Table 4: Performance of CNNs on data augmentation (random 10-crops with horizontal flipping) and scale variation. The results show the DFT magnitude pooling method improves classification performance in all cases (top1/top5 error).

Network	Scale	Baseline	DFT
AlexNet	256	37.97 / 16.70	37.33 / 16.05
	384	40.80 / 18.67	39.38 / 17.75
	512	51.02 / 27.16	49.35 / 25.65
VGG-VD 16	256	26.61 / 8.62	25.26 / 7.78
	384	29.48 / 9.77	27.57 / 8.84
	512	38.17 / 15.17	35.74 / 13.73
ResNet-50	256	23.42 / 6.73	22.66 / 6.51
	384	24.27 / 6.99	23.46 / 6.68
	512	28.13 / 9.12	27.21 / 8.55

2 Evaluations for Data Augmentation and Variations

In the following experiments, we evaluate DFT and DFT$^+$ methods with respect to large translation, scale changes and data augmentation. We first evaluate the translation invariance property of the DFT magnitude pooling method using the images synthetically generated with large variations of translation and scale in complex scenes. Original images from the CUB and Caltech 101 datasets are scaled by 25%, 50%, and 75%, and then overlaid on randomly selected images from the MIT Indoor dataset. The networks are trained and tested as described in Section 4.2 with the synthesized images. Table 3 shows that the DFT magnitude pooling significantly outperforms the baselines that use an average pooling or direct connection to the fully-connected layer. The performance gain was 15.40%

Table 5: Experimental comparison of DFT with baseline networks on the CUB dataset. S=1 is single scale of size=256, S=2, S=3 are multi scale of [256, 384] and [256, 384, 512].

Network	Pooling type			
	Baseline (10 crops)			DFT
	#S=1	#S=2	#S=3	(1 crop)
VGG-VD 16	76.0	77.0	77.2	79.6
ResNet-50	78.1	78.1	78.2	81.0

Table 6: Performance evaluation on the DFT, DFT$^+$ methods and previous works with respect to the multi scale training. We additionally evaluate multi scale training introduced by Deep-TEN [1]. The result shows DFT and DFT$^+$ methods performs well in the multi scale training setting. The methods use the ResNet-50 [2] as the baseline network.

Method	Scale	Dataset	
		FMD	MIT Indoor
FV	Multi	78.2	76.1
Deep-TEN	Single	80.2	71.3
	Multi	78.8	76.2
DFT	Single	79.2	74.8
	Multi	81.0	78.6
DFT$^+$	Single	81.2	76.9
	Multi	**82.8**	**80.2**

for the Inception-v3 model on the CUB dataset, and 12.18% for the AlexNet on the Caltech 101 dataset when an object is rescaled by 25%. These experimental results show that the deep network with the DFT magnitude pooling layer maintains translation invariance properties effectively for visual classification.

Second, we evaluate CNNs of the DFT magnitude pooling and baseline under data augmentation and multi-scale evaluation for the ImageNet. We use 10 random crops with horizontal flipping for the data augmentation and three scales (256,384,512) for the multi-scale evaluation. Although we train three CNNs (AlexNet, VGG-VD 16 and ResNet-50) by the single scale of [256,256], we evaluate them on multiple test scales to confirm the consistent improvement of DFT magnitude pooling. The result of Table 4 shows that DFT magnitude pooling improves baseline methods in all cases. Further, the larger the scale, the higher the performance gap is achieved, due to the translation invariant property of DFT magnitude pooling. As a result, we can expect that DFT magnitude

Table 7: Performance evaluation on the DFT, DFT$^+$ methods and previous works to the horizontal flip and bounding box annotation. The left and right numbers of the CUB dataset are with and without bounding annotation. Overall, the DFT and DFT$^+$ methods perform well compared to previous works. All results are obtained by the VGG-VD 16 [3] model except B-CNN [D,M], which use the VGG-VD 16 and VGG-M of [4]. Numbers marked with $*$ is an result by without the fine-tuning network.

Mtehod	Input size	Dataset				
		FMD	DTD	CUB	MIT Indoor	Source
FV	224	75.0	-	-	67.8	[1]
	224	75.1	67.8	-	70.1	[5]
B-CNN [D,M]	448	-	-	84.1/**85.1**	-	[6]
B-CNN [D,D]	224	77.8	69.6	-	72.8	[5]
B-CNN [D,D]	448	-	-	84.0/84.8	-	[6]
B-CNN$_{\text{compact}}$	448	-	64.5 (67.7*)	84.0/ -	72.7	[7]
DFT (w/o flip)	224	78.8	72.4	75.3/79.6	72.6	-
DFT (w/ flip)	224	79.4	72.8	76.3/80.7	74.5	-
DFT$^+$ (w/o flip)	224	80.0	73.2	76.5/80.1	75.2	-
DFT$^+$ (w/ flip)	224	**80.2**	**74.0**	77.0/81.3	**77.8**	-

pooling further improves the performance of CNNs trained by large scale images and augmentation of scale jittering.

We evaluate the DFT magnitude pooling with respect to scale and cropping settings on the transferred domain (CUB) from ImageNet. we conduct experiments with scaling [3/4,4/3], cropping [224-256] and flip jittering (10 crops) but the DFT magnitude pooling uses just one crop for training and testing. Table 5 shows DFT magnitude pooling performs well in this setting while it uses only single crop.

Third, we evaluate DFT and DFT$^+$ methods with respect to various settings such as the multi scale training, the bounding box annotation and the horizontal flip. In general, the multi scale training, use of bounding box annotation and data augmentation by the horizontal flip are performed to validate the performance of CNNs. Previous works evaluate their algorithm under the horizontal flip for data augmentation, bounding box annotation [8] for the CUB dataset and multi scale training for CNNs. Thus, we first evaluate multi scale training suggested by [1] in Table 6. Second, DFT and DFT$^+$ methods are evaluated with respect to the data augmentation by the horizontal flip and bounding box annotation for CUB dataset in Table 7. The result of these additional evaluations also confirms that DFT and DFT$^+$ methods perform well under various environment.

3 Formula for forward and backward propagation

When x and y are the input and output of the DFT magnitude pooling method, the forward propagation function is $F(\xi) = \mathscr{F}(f(x))$, $y = \sqrt{F_r^2(\xi) + F_i^2(\xi)}$, and the backpropagattion function is $\frac{\partial y}{\partial F_r} = \frac{F_r(\xi)}{\sqrt{F_r^2(\xi)+F_i^2(\xi)}}$, $\frac{\partial y}{\partial F_i} = \frac{F_i(\xi)}{\sqrt{F_r^2(\xi)+F_i^2(\xi)}}$, where $\mathscr{F}$ denotes the Fourier transform, and F_r and F_i are the real and imaginary parts. The gradient is computed by the inverse Fourier transform as $\mathscr{F}^{-1}(\{F_r, F_i\})$.

References

1. Zhang, H., Xue, J., Dana, K.: Deep ten: Texture encoding network. In: IEEE Conference on Computer Vision and Pattern Recognition. (2017)
2. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition. (2016)
3. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. Arxiv (2014)
4. Chatfield, K., Simonyan, K., Vedaldi, A., Zisserman, A.: Return of the devil in the details: Delving deep into convolutional nets. Arxiv (2014)
5. Lin, T.Y., Maji, S.: Visualizing and understanding deep texture representations. In: IEEE Conference on Computer Vision and Pattern Recognition. (2016) 2791–2799
6. Lin, T.Y., RoyChowdhury, A., Maji, S.: Bilinear cnn models for fine-grained visual recognition. In: IEEE International Conference on Computer Vision. (2015) 1449–1457
7. Gao, Y., Beijbom, O., Zhang, N., Darrell, T.: Compact bilinear pooling. In: IEEE Conference on Computer Vision and Pattern Recognition. (2016) 317–326
8. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The caltech-ucsd birds-200-2011 dataset. Technical report, California Institute of Technology (2011)