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

Abstract. We propose a novel discrete Fourier transform-based pooling layer for convolutional neural networks. The DFT magnitude pooling replaces the traditional max/average pooling layer between the convolution and fully-connected layers to retain translation invariance and shape preserving (aware of shape difference) properties based on the shift theorem of the Fourier transform. Thanks to the ability to handle image misalignment while keeping important structural information in the pooling stage, the DFT magnitude pooling improves the classification accuracy significantly. In addition, we propose the DFT+ method for ensemble networks using the middle convolution layer outputs. The proposed methods are extensively evaluated on various classification tasks using the ImageNet, CUB 2010-2011, MIT Indoors, Caltech 101, FMD and DTD datasets. The AlexNet, VGG-VD 16, Inception-v3, and ResNet are used as the base networks, upon which DFT and DFT+ methods are implemented. Experimental results show that the proposed methods improve the classification performance in all networks and datasets.

1 Introduction

Convolutional neural networks (CNNs) have been widely used in numerous vision tasks. In these networks, the input image is first filtered with multiple convolution layers sequentially, which give high responses at distinguished and salient patterns. Numerous CNNs, e.g., AlexNet [1] and VGG-VD [2], feed the convolution results directly to the fully-connected (FC) layers for classification with the soft-max layer. These fully-connected layers do not discard any information and encode shape/spatial information of the input activation feature map. However, the convolution responses are not only determined by the image content, but also affected by the location, size, and orientation of the target object in the image.

To address this misalignment problem, recently several CNN models, e.g., GoogleNet [3], ResNet [4], and Inception [5], use an average pooling layer. The structure of these models is shown in the top two rows of Fig. 1. It is placed between the convolution and fully-connected layers to convert the multi-channel 2D response maps into a 1D feature vector by averaging the convolution outputs in each channel. The channel-wise averaging disregard the location of activated
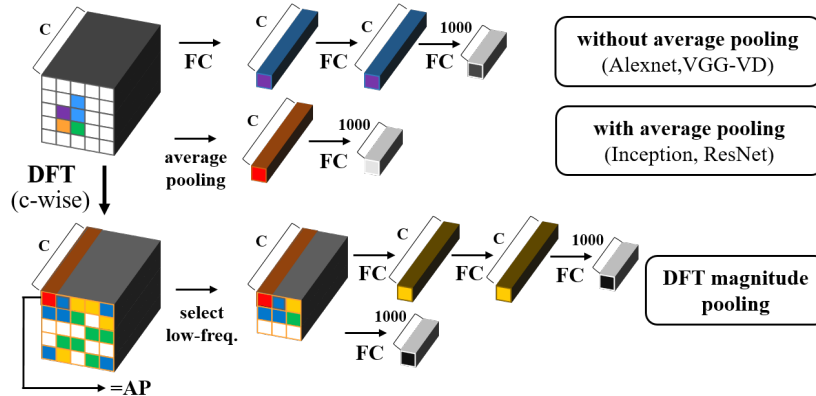

⋆ Corresponding author.

Fig. 1: Feature maps at the last layers of CNNs. Top two rows: conventional layouts, without and with average pooling. Bottom two rows: the proposed DFT magnitude pooling. The DFT applies the channel-wise transformation to the input feature map and uses the magnitudes for next fully-connected layer. Note that the top-left cell in the DFT magnitude is the same as the average value since the first element in DFT is the average magnitude of signals. Here C denotes the number of channels of the feature map.

neurons in the input feature map. While the model becomes less sensitive to misalignment, the shapes and spatial distributions of the convolution outputs are not passed to the fully-connected layers.

Fig. 2 shows an example of the translation invariance and shape preserving and properties in CNNs. For CNNs without average pooling, the FC layers give all different outputs for the different shaped and the translated input with same number of activations (topmost row). When an average pooling layer is used, the translation in the input is ignored, but it cannot distinguish different patterns with the same amount of activations (second row). Either without or with average pooling, the translation invariance and shape preserving properties are not simultaneously preserved.

Ideally, the pooling layer should be able to handle such image misalignments and retain the prominent signal distribution from the convolution layers. Although it may seem that these two properties are incompatible, we show that the proposed novel DFT magnitude pooling retains both properties and consequently improves classification performance significantly. The shift theorem of Fourier transform [6] shows that the magnitude of Fourier coefficients of two signals are identical if their amplitude and frequency (shape) are identical, regardless of the phase shift (translation). In DFT magnitude pooling, 2D-DFT (discrete Fourier transform) is applied to each channel of the input feature map, and the magnitudes are used as the input to the fully-connected layer (bottom rows of Fig. 1). Further by discarding the high-frequency coefficients, it is possible to maintain the crucial shape information, minimize the effect of noise, and reduce the number
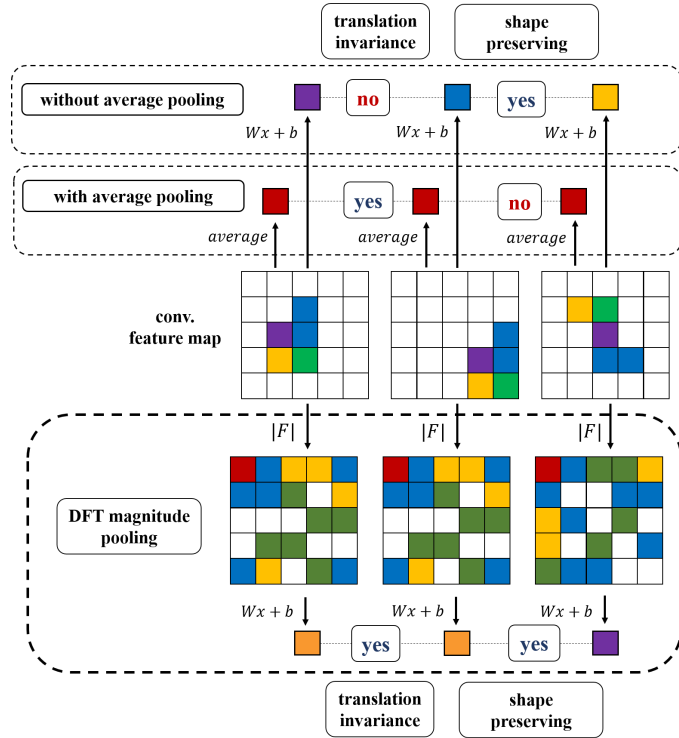
Fig. 2: Comparison of DFT magnitude with and without average pooling. The middle row shows the feature maps of the convolution layers, where all three have the same amount of activations, and the first two are same shape but in different positions. The output of the fully-connected layer directly connected to this input will output different values for all three inputs, failing to catch the first two have the same shape. Adding an average pooling in-between makes all three outputs same, and thus it achieves translation invariance but fails to distinguish the last from the first two. On the other hand, the proposed pooling outputs the magnitudes of DFT, and thus the translation in the input patterns is effectively ignored and the output varies according to the input shapes.

of parameters in the following fully-connected layer. It is worth noting that the average pooling response is same as the first coefficient of DFT (DC part). Thus the DFT magnitude is a superset of the average pooling response, and it can be as expressive as direct linking to FC layers if all coefficients are used.

For the further performance boost, we propose the DFT$^+$ method which ensembles the response from the middle convolution layers. The output size of a middle layer is much larger than that of the last convolution layer, but the DFT can select significant Fourier coefficients only to match to the similar resolution of the final output.

To evaluate the performance of the proposed algorithms, we conduct extensive experiments with various benchmark databases and base networks. We show that DFT and DFT$^+$ methods consistently and significantly improve the state-of-the-art baseline algorithms in different types of classification tasks.

We make the following contributions in this work:

(i) We propose a novel DFT magnitude pooling based on the 2D shift theorem of Fourier transform. It retains both translation invariant and shape preserving properties which are not simultaneously satisfied in the conventional approaches. Thus the DFT magnitude is more robust to image mis-alignment as well as noise, and it supersedes the average pooling as its output contains more information.

(ii) We suggest the DFT$^+$ method, which is an ensemble scheme of the middle convolution layers. As the output feature size can be adjusted by trimming high-frequency parts in the DFT, it is useful in handling higher resolution of middle-level outputs, and also helpful in reducing the parameters in the following layers.

(iii) Extensive experiments using various benchmark datasets (ImageNet, CUB, MIT Indoors, Caltech 101, FMD and DTD) and numerous base CNNs (AlexNet, VGG-VD, Inception-v3, and ResNet) show that the DFT and DFT$^+$ methods significantly improve classification accuracy in all settings.

2   Related Work

One of the most widely used applications of CNNs is the object recognition task [1,2,3,4,5] on the ImageNet dataset. Inspired by the success, CNNs have been applied to other recognition tasks such as scene [7,8] and fine-grained object recognition [9,10,11], as well as other tasks like object detection [12,13,14], and image segmentation [15,16,17]. We discuss the important operations of these CNNs and put this work in proper context.

2.1   Transformation Invariant Pooling

In addition to rich hierarchical feature representations, one of the reasons for the success of CNN is the robustness to certain object deformations. For further robustness over misalignment and deformations, one may choose to first find the target location in an image and focus on those regions only. For example, in the faster R-CNN [13] model, the region proposal network evaluates sliding windows in the activation map to compute the probability of the target location. While it is able to deal with uncertain object positions and outlier background regions, this approach entails high computational load. Furthermore, even with good object proposals, it is difficult to handle the misalignment in real images effectively by pre-processing steps such as image warping. Instead, numerous methods have been developed to account for spatial variations within the networks.

The max or average pooling layers are developed for such purpose [5,4,18]. Both pooling layers reduce a 2D input feature map in each channel into a scalar value by taking the average or max value.

Another approach to achieve translation invariance is orderless pooling, which generates a feature vector insensitive to activation positions in the input feature map. Gong *et al.* [19] propose the multi-scale orderless pooling method for image classification. Cimpoi *et al.* [20] develop an orderless pooling method by applying the Fisher vector [21] to the last convolution layer output. Bilinear pooling [9] is proposed to encode orderless features by outer-product operation on a feature map. The α-pooling method for fine-grained object recognition by Simon *et al.* [22] combines average and bi-linear pooling schemes to form orderless features. Matrix backpropagation [23] is proposed to train entire layers of a neural network based on higher order pooling. Gao *et al.* [24] suggest compact bilinear pooling that reduce dimensionality of conventional bilinear pooling. Kernel pooling [25] is proposed to encode higher order information by fast Fourier transform method. While the above methods have been demonstrated to be effective, the shape information preserving and translation invariant properties are not satisfied simultaneously in the pooling.

The spectral pooling method, which uses DFT algorithm, is proposed by [26]. It transforms the input feature map, crop coefficients of the low frequency of transformed feature map, and then the inverse transform is applied to get the output pooled feature map on the original signal domain. They use DFT to reduce the feature map size, so they can preserve shape information but do not consider the translation property. However, proposed approach in this work outputs the feature map satisfying both properties by the shift theorem of DFT.

2.2   Ensemble Using Multi-convolution Layers

Many methods have been developed to use the intermediate features from multi-convolution layers for performance gain [27]. The hypercolumn [28] features ensemble outputs of multi-convolution layers via the upsampling method upon which the decision is made. For image segmentation, the fully convolutional network (FCN) [15] combines outputs of multiple convolution layers via the upsampling method. In this work, we present DFT+ method by ensembling middle layer features using DFT and achieve further performance improvement.

3   Proposed Algorithm

In this section, we discuss the 2D shift theorem of the Fourier transform and present DFT magnitude pooling method.

3.1   2D Shift Theorem of DFT

The shift theorem [6] from the Fourier transform describes the shift invariance property in the one-dimensional space. For two signals with same amplitude and

frequency but different phases, the magnitudes of their Fourier coefficients are identical. Suppose that the input signal f_n is converted to F_k by the Fourier transform,

$$F_k = \sum_{n=0}^{N-1} f_n \cdot e^{-j2\pi kn/N},$$

a same-shaped input signal but phase-shifted by θ can be denoted as $f_{n-\theta}$, and its Fourier transformed output as $F_{k-\theta}$. Here, the key feature of the shift theorem is that the magnitude of $F_{k-\theta}$ is same as the magnitude of F_k, which means the magnitude is invariant to phase differences. For the phase-shifted signal, we have

$$F_{k-\theta} = \sum_{n=0}^{N-1} f_{n-\theta} \cdot e^{-j2\pi kn/N} = \sum_{m=-\theta}^{N-1-\theta} f_m \cdot e^{-j2\pi k(m+\theta)/N}$$

$$= e^{-j2\pi\theta k/N} \sum_{m=0}^{N-1} f_m \cdot e^{-j2\pi km/N} = e^{-j2\pi\theta k/N} \cdot F_k.$$

Since $e^{-j2\pi\theta k/N} \cdot e^{j2\pi\theta k/N} = 1$, we have

$$|F_{k-\theta}| = |F_k|. \tag{1}$$

The shift theorem can be easily extended to 2D signals. The shifted phase θ of Eq. 1 in 1D is replaced with (θ_1, θ_2) in 2D. These two phase parameters represent the 2D translation in the image space and we can show the following equality extending the 1D shift theorem, i.e.,

$$F_{k_1-\theta_1,k_2-\theta_2} = e^{-j2\pi(\theta_1 k_1/N_1 + \theta_2 k_2/N_2)} \cdot F_{k_1,k_2}.$$

Since $e^{-j2\pi(\theta_1 k_1/N_1 + \theta_2 k_2/N_2)} \cdot e^{j2\pi(\theta_1 k_1/N_1 + \theta_2 k_2/N_2)} = 1$, we have

$$|F_{k_1-\theta_1,k_2-\theta_2}| = |F_{k_1,k_2}|. \tag{2}$$

The property of Eq. 2 is of critical importance in that the DFT outputs the same magnitude values for the translated versions of a 2D signal.

3.2   DFT Magnitude Pooling Layer

The main stages in the DFT magnitude pooling are illustrated in the bottom row of Fig. 1. The convolution layers generate an $M \times M \times C$ feature map, where M is determined by the spatial resolution of the input image and convolution filter size. The $M \times M$ feature map represents the neuron activations in each channel, and it encodes the visual properties including shape and location, which can be used in distinguishing among different object classes. The average or max pooling removes location dependency, but at the same time, it discards valuable shape information.

In the DFT magnitude pooling, 2D-DFT is applied to each channel of the input feature map, and the resulting Fourier coefficients are cropped to $N \times N$

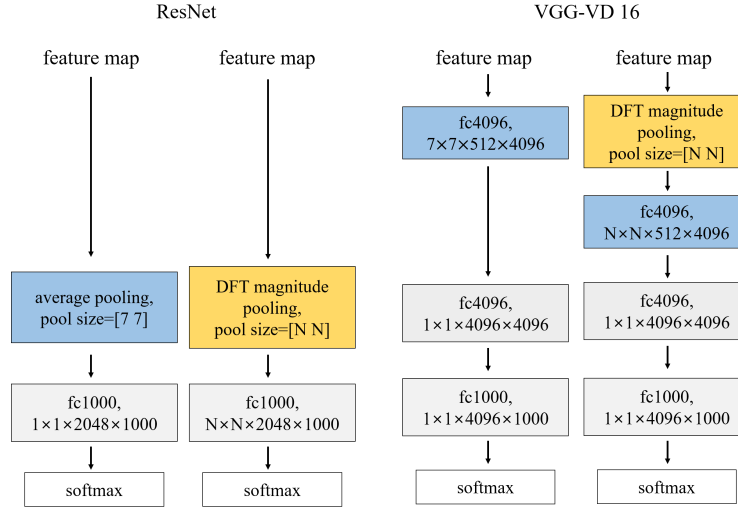ResNet   VGG-VD 16

Fig. 3: Examples of DFT magnitude pooling usage. It replaces the average pooling layer of ResNet [4] and it is inserted between the last convolution layer and first fc4096 layer of VGG-VD 16 [2].

by cutting off high frequency components, where N is a user-specified parameter used to control the size. The remaining low-frequency coefficients is then fed into the next fully-connected layer. As shown in Section 3.1, the magnitude of DFT polled coefficients is translation invariant, and by using more pooled coefficients of DFT, the proposed method can propagate more shape information in the input signal to the next fully-connected layer. Hence the DFT magnitude pooling can achieve both translation invariance and shape preserving properties, which are seemingly incompatible. In fact, the DFT supersedes the average pooling since the average of the signal is included in the DFT pooled magnitudes.

As mentioned earlier, we can reduce the pooled feature size of the DFT magnitude by only selecting the low frequency parts of the Fourier coefficients. This is one of the merits of our method as we can reduce the parameters in the fully-connected layer without losing much spatial information. In practice, the additional computational overhead of DFT magnitude pooling is negligible considering the performance gain (Table 1 and 2). The details of the computational overhead and number of parameters are explained in the supplementary material.

3.3   Late Fusion in DFT$^+$

In typical CNNs, only the output of the final convolution layer is used for classification. However, the middle convolution layers contain rich visual information that can be utilized together with the final layer's output. In [29], the SVM classifier output is combined with the responses of spatial and temporal networks where these two networks are trained separately. Similar to [29], we adopt the late
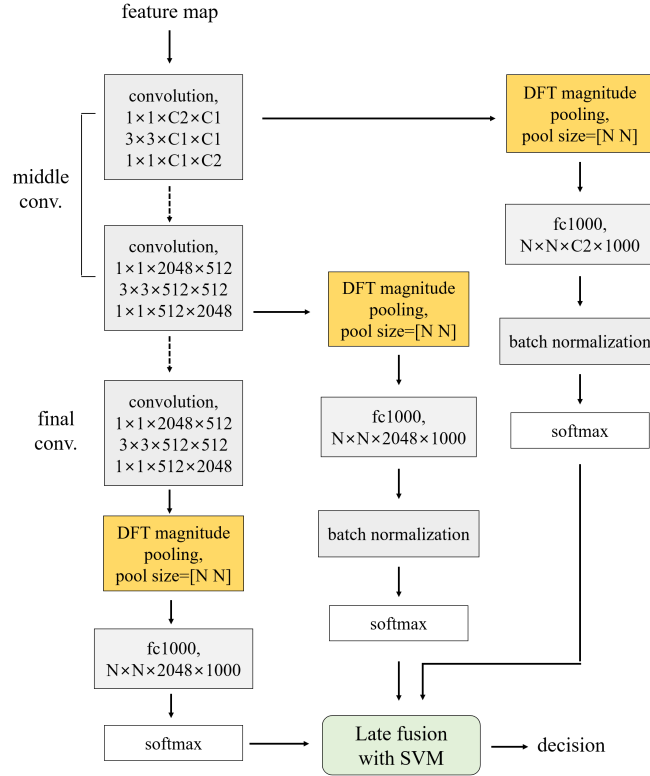
feature map

convolution,
1×1×C2×C1
3×3×C1×C1
1×1×C1×C2

middle conv.

convolution,
1×1×2048×512
3×3×512×512
1×1×512×2048

final conv.

convolution,
1×1×2048×512
3×3×512×512
1×1×512×2048

DFT magnitude
pooling,
pool size=[N N]

fc1000,
N×N×2048×1000

softmax

DFT magnitude
pooling,
pool size=[N N]

fc1000,
N×N×2048×1000

batch normalization

softmax

DFT magnitude
pooling,
pool size=[N N]

fc1000,
N×N×C2×1000

batch normalization

softmax

Late fusion
with SVM

decision

Fig. 4: Example of DFT$^+$ usage for ResNet. The DFT magnitude pooling, fully-connected and softmax layers together with batch-normalization are added to the middle convolution layers. The SVM is used for the late fusion.

fusion approach to combine the outputs of multiple middle layers. The mid-layer convolution feature map is separately processed through a DFT, a fully-connected, a batch normalization, and a softmax layers to generate the mid-layer probabilistic classification estimates. In the fusion layer, all probabilistic estimates from the middle layers and the final layer are vectorized and concatenated, and SVM on the vector determines the final decision.

Furthermore, we use a group of middle layers to incorporate more and richer visual information. The middle convolution layers in the network are grouped according to their spatial resolutions $(M \times M)$ of output feature maps. Each layer group consists of more than one convolution layers of the same size, and depending on the level of fusion, different numbers of groups are used in training and testing. The implementation of this work is available at `http://cvlab.hanyang.ac.kr/project/eccv_2018_DFT.html`. In the following section we present the detailed experiment setups and the extensive experimental results showing the effectiveness of DFT magnitude pooling.

4   Experimental Results

We evaluate the performance of the DFT and DFT$^+$ methods on the large scale ImageNet [35] dataset, and CUB [36], MIT67 [33], as well as Caltech 101 [37] datasets. The AlexNet [1], VGG-VD16 [2], Inception-v3 [5], ResNet-50, ResNet-101, and ResNet-152 [4] are used as the baseline algorithm. To show the effectiveness of the proposed approaches, we replace only the pooling layer in each baseline algorithm with the DFT magnitude pooling and compare the classification accuracy. When the network does not have an average pooling layer, e.g., AlexNet and VGG, the DFT magnitude pooling is inserted between the final convolution and first fully-connected layers.

The DFT$^+$ uses the mid layer outputs, which are fed into a separate DFT magnitude pooling and fully-connected layers to generate the probabilistic class label estimates. The estimates by the mid and final DFT magnitude pooling are then combined using a linear SVM for the final classification. In the DFT$^+$ method, batch normalization layers are added to the mid DFT method for stability in back-propagation. In this work, three settings with the different number of middle layers are used. The DFT$^+{}_1$ method uses only one group of middle layers located close to the final layer. The DFT$^+{}_2$ method uses two middle layer groups, and the DFT$^+{}_3$ method uses three. Fig. 3 and 4 show network structures and settings of DFT and DFT$^+$ methods.

For performance evaluation, DFT and DFT$^+$ methods are compared to the corresponding baseline network. For DFT$^+$, we also build and evaluate the average$^+$, which is an ensemble of the same structure but using average pooling. Unless noted otherwise, N is set to the size of the last convolution layer of the base network (6, 7, or 8).

Table 1: Classification error of the networks trained from scratch on the ImageNet (top1/top5 error). Both DFT and DFT$^+$ methods significantly improve the baseline networks, while average$^+$ does not improve the accuracy meaningfully.

Method	AlexNet (no-AP)	VGG-VD16 (no-AP)	ResNet-50 (with-AP)
Baseline	41.12 / 19.08	29.09 / 9.97	25.15 / 7.78
DFT	40.23 / 18.12	27.28 / 9.10	24.37 / 7.45
	-0.89 / -0.96	-1.81 / -0.87	-0.78 / -0.33
DFT$^+$	39.80 / 18.32	27.07 / 9.02	24.10 / 7.31
	-1.32 / -0.76	-2.02 / -0.95	-1.05 / -0.47
average$^+$	41.09 / 19.53	28.97 / 9.91	25.13 / 7.77
	-0.03 / +0.45	-0.12 / -0.06	-0.02 / -0.01

Table 2: Classification accuracy of transferring performance to different domains. DFT magnitude pooling results and the best results of DFT$^+$ method are marked as bold. The accuracy of DFT method is improved in all cases except Caltech101-AlexNet, and DFT$^+$ always outperforms average$^+$, as well as the baseline and DFT. See Section 4.2 for more details.

Data	Network	Base	DFT	DFT$^+_1$	average$^+_1$	DFT$^+_2$	average$^+_2$	DFT$^+_3$	average$^+_3$
CUB	AlexNet	64.9	**68.1**	**68.7**	64.9	68.5	64.7	68.6	64.9
	VGG-VD16	75.0	**79.6**	79.7	75.0	79.9	74.8	**80.1**	75.0
	Inception-v3	80.1	**80.9**	82.2	80.4	**82.4**	80.2	82.0	80.2
	ResNet-50	77.5	**81.0**	81.8	77.7	82.0	77.9	**82.7**	77.8
	ResNet-101	80.4	**82.1**	82.7	81.0	**83.1**	81.0	82.9	80.8
	ResNet-152	81.4	**83.7**	83.6	81.5	83.8	81.6	**83.8**	81.5
MIT Indoor	AlexNet	59.2	**59.4**	**59.9**	59.3	59.6	58.9	**59.9**	59.0
	VGG-VD16	72.2	**72.6**	74.2	73.1	74.6	72.8	**75.2**	73.1
	Inception-v3	73.2	**73.4**	76.9	74.5	**77.3**	74.5	74.3	73.9
	ResNet-50	73.0	**74.8**	**76.9**	75.0	76.3	75.2	75.9	75.0
	ResNet-101	73.3	**76.0**	76.1	75.1	**76.9**	75.2	76.6	74.9
	ResNet-152	73.5	**75.3**	76.4	75.5	**76.5**	75.3	76.3	74.9
Caltech 101	AlexNet	88.1	**87.4**	88.1	88.0	88.2	88.1	**88.3**	88.1
	VGG-VD16	93.2	**93.2**	93.4	93.3	93.4	93.2	**93.6**	93.2
	Inception-v3	94.0	**94.1**	95.2	94.2	95.1	94.2	94.5	94.0
	ResNet-50	93.2	**93.9**	94.6	93.5	**94.8**	93.3	94.7	93.5
	ResNet-101	93.1	**94.2**	94.0	93.4	94.2	93.3	**94.4**	93.2
	ResNet-152	93.2	**94.0**	94.3	93.7	**94.7**	93.7	94.4	93.3

4.1   Visual Classification on the ImageNet

We use the AlexNet, VGG-VD16, and ResNet-50 as the baseline algorithm and four variants (baseline with no change, DFT, DFT$^+$, and average$^+$) are trained from scratch using the ImageNet database with the same training settings and standard protocol for fair comparisons. In this experiment, DFT$^+$ only fuses the second last convolution layer with the final layer, and we use a weighted sum of the two softmax responses instead of using an SVM.

Table 1 shows that the DFT magnitude pooling reduces classification error by 0.78 to 1.81%. In addition, the DFT$^+$ method further reduces the error by 1.05 to 2.02% in all three networks. On the other hand, the A-pooling+ method hardly reduce the classification error rate.

The experimental results demonstrate that the DFT method performs favorably against the average pooling (with-AP) or direct connection to the fully-connected layer (no-AP). Furthermore, the DFT$^+$ is effective in improving classification performance by exploiting features from the mid layer.

4.2   Transferring to Other Domains

The transferred CNN models have been applied to numerous domain-specific classification tasks such as scene classification and fine-grained object recognition. In the following experiments, we evaluate the generalization capability, i.e., how well a network can be transferred to other domains, with respect to the pooling layer. The baseline, DFT and DFT$^+$ methods are fine-tuned using the CUB (fine-grained), MIT Indoor (scene), and Caltech 101 (object) datasets using the standard protocol to divide training and test samples. As the pre-trained models, we use the AlexNet, VGG-VD16, and ResNet-50 networks trained from scratch using the ImageNet in Section 4.1 . For the Inception-v3, ResNet-101, and ResNet-152, the pre-trained models in the original work are used. Also, the soft-max and the final convolution layers in the original networks are modified for the transferred domain. Table 2 shows that DFT magnitude pooling outperforms the baseline algorithms in all networks except one case of the AlexNet on the Caltech101 dataset. In contrast the A-pool+ model does not improve the results.

4.3   Comparison with state-of-the-art methods

We also compare proposed DFT based method with state-of-the-art methods such as the Fisher Vector(FV) [21] with CNN feature [20], the bilinear pooling [9,34], the compact bilinear pooling [24] and the texture feature descriptor e.g.Deep-TEN [30]. The results of the single image scale are reported for the fair comparison except that the results of Deep-TEN$_{multi}$ and FV$_{multi}$ of ResNet-50 are acquired

Table 3: Comparison of DFT and DFT$^+$ methods with state-of-the-art methods. DFT and DFT$^+$ methods gives favorable classification rate compared to previous state-of-the-art methods. DFT$^+$ method improves previous results based on ResNet-50 and also enhances the performance of state-of-the-art methods with VGG-VD 16 in most cases, while we use only single 224×224 input image. The results of the FV on all cases are reproduced by [30] and the B-CNN [9] on FMD [31], DTD [32] and MIT Indoor [33] with VGG-VD 16 are obtained by [34]. Numbers marked with $*$ are the results by 448×448 input image. More results under various experimental settings are shown in the supplementary material.

	VGG-VD 16						ResNet-50			
Method	Dataset					Method	Dataset			
	FMD	DTD	Caltech 101	CUB	MIT Indoor		FMD	Caltech 101	MIT Indoor
FV	75.0	-	83.0	-	67.8	FV$_{multi}$	78.2	-	76.1
B-CNN	77.8	69.6	-	84.0*	72.8	Deep-TEN	80.2	85.3	71.3
B-CNN$_{compact}$	-	64.5*	-	84.0*	72.7*	Deep-TEN$_{multi}$	78.8	-	76.2
DFT	78.8	72.4	93.2	79.6	72.6	DFT	79.2	93.9	74.8
DFT$^+$	**80.0**	**73.2**	**93.6**	80.1	**75.2**	DFT$^+$	**81.2**	**94.8**	**76.9**

on the multiscale setting. The input image resolution is 224×224 for all methods except some results of Bilinear(B-CNN) and compact bilinear(B-CNN$_{compact}$) pooling methods, which uses 448×448 images. The results of Table 3 shows that DFT and DFT$^+$ methods improves classification accuracy of state-of-the-art methods in most cases. DFT and DFT$^+$ methods does not enhance the classification accuracy with only one case: B-CNN and B-CNN$_{compact}$ of CUB dataset with VGG-VD 16, which use larger input image compared to our implementation. In the other cases, DFT$^+$ method performs favorably compared to previous transformation invariant pooling methods. Especially, DFT$^+$ method improves classification accuracy about 10% for Caltech 101. This is because the previous pooling methods are designed to consider the orderless property of images. While considering the orderless property gives fine results to fine-grained recognition dataset (CUB 2000-2201), it is not effective for object image dataset (Caltech 101). Since, shape information, that is the order of object parts, is very informative to recognize object images, so orderless pooling does not improve performance for Caltech 101 dataset. However, DFT and DFT$^+$ methods acquire favorable performance by also preserving the shape information for object images. Therefore, this result also validates the generalization ability of the proposed method for the deep neural network architecture.

Table 4: Experimental result of the DFT and DFT$^+$ methods with respect to the pooling size. Performance tends to get better as pooling size increases, but it can be seen that N=4 is enough to improve the baseline method significantly.

Dataset	Network	Base	DFT			DFT$^+{}_3$		
			N=2	N=4	full	N=2	N=4	full
	Alexnet	64.9	67.9	67.9	68.1	68.2	68.4	68.6
	VGG-VD 16	75.0	79.0	78.9	79.6	78.9	79.0	80.1
	Inception v3	80.1	78.3	79.1	80.9	80.3	80.7	82.0
CUB	ResNet-50	77.5	76.2	78.2	81.0	78.7	81.1	82.7
	ResNet-101	80.4	81.7	82.4	82.1	82.1	83.1	82.9
	ResNet-152	81.4	82.6	83.1	83.7	82.7	83.3	83.8
	Alexnet	59.2	59.4	59.3	59.4	61.2	61.6	59.9
	VGG-VD 16	72.2	75.2	74.1	72.6	75.5	75.4	75.2
	Inception v3	73.3	72.8	72.0	73.4	74.8	74.1	74.3
MIT Indoor	ResNet-50	73.0	73.5	73.8	74.8	76.0	75.6	75.9
	ResNet-101	73.3	74.0	75.4	76.0	74.5	76.2	76.6
	ResNet-152	73.5	73.4	75.6	75.3	74.0	76.3	76.3
	Alexnet	88.1	87.4	87.3	87.4	88.0	87.9	88.3
	VGG-VD 16	93.2	92.5	92.9	93.2	92.6	93.6	93.6
	Inception v3	94.0	93.1	93.0	94.1	94.0	93.8	94.5
Caltech 101	ResNet-50	93.2	92.8	92.8	93.9	93.2	93.3	94.7
	ResNet-101	93.1	93.4	94.0	94.2	93.5	93.7	94.3
	ResNet-152	93.2	93.8	94.2	94.0	93.9	94.0	94.4

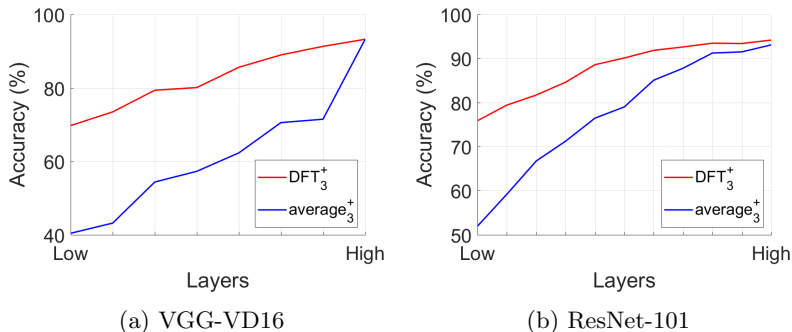(a) VGG-VD16   (b) ResNet-101

Fig. 5: Performance comparison of average with DFT magnitude pooling in average$^+_3$ and DFT$^+_3$ methods on Caltech 101. The reported classification accuracy values are obtained from the middle softmax layers independently.

5   Discussion

To further evaluate the DFT magnitude pooling, the experiment with regard to the pooling sizes are performed in Table 4. It shows that the small pooling size also improves the performance of the baseline method. Fig. 5 shows the classification accuracy of the individual middle layers by the DFT magnitude and average pooling layers before the late fusion. The DFT method outperforms the average pooling, and the performance gap is much larger in the lower layers than the higher ones. It is known that higher level outputs contain more abstract and robust information, but middle convolution layers also encode more detailed and discriminant features that higher levels cannot capture. The results are consistent with the findings in the supplementary material that the DFT method is robust to spatial deformation and misalignment, which are more apparent in the lower layers in the network (i.e., spatial deformation and misalignment are related to low level features than semantic ones). Since the class estimated by the DFT method from the lower layers is much more informative than those by the average pooling scheme, the DFT$^+$ achieves more performance gain compared to the baseline or the average$^+$ scheme. These results show that the performance of ensemble using the middle layer outputs can be enhanced by using the DFT as in the DFT$^+$ method.

The DFT$^+$ method can also be used to facilitate training CNNs by supplying additional gradient to the middle layers in back-propagation. One of such examples is the auxiliary softmax layers of the GoogleNet [3], which helps back-propagation stable in training. In GoogleNet, the auxiliary softmax with average pooling layers are added to the middle convolution layers during training. As such, the proposed DFT$^+$ method can be used to help training deep networks.

Another question of interest is whether a deep network can learn translation invariance property without adding the DFT function. The DFT magnitude

Table 5: Comparison of learnable DFT with the baseline DFT (top1/top5 error). The classification error is measured on the AlexNet with learning from scratch using the ImageNet.

Baseline	DFT	DFT-learnable	
		2D DFT-init	Random-init
41.12 / 19.08	40.23 / 18.12	40.64 / 18.76	40.71 / 18.87

pooling explicitly performs the 2D-DFT operation, but since DFT function itself can be expressed as a series of convolutions for real and imaginary parts (referred to as a DFT-learnable), it may be possible to learn such a network to achieve the same goal. To address this issue, we design two DFT-learnable instead of explicit DFT function, where one is initialized with the correct parameters of 2D-DFT, and the other with random values. AlexNet is used for this experiment to train DFT-learnable using the ImageNet. The results are presented in Table 5. While both DFT-learnable networks achieve lower classification error than the baseline method, their performance is worse than that by the proposed DFT magnitude pooling. These results show that while DFT-learnable may be learned from data, such approaches do not perform as well as the proposed model in which both translation invariance and shape preserving factors are explicitly considered.

6   Conclusions

In this paper, we propose a novel DFT magnitude pooling for retaining transformation invariant and shape preserving properties, as well as an ensemble approach utilizing it. The DFT magnitude pooling extends the conventional average pooling by including shape information of DFT pooled coefficients in addition to the average of the signals. The proposed model can be easily incorporated with existing state-of-the-art CNN models by replacing the pooling layer. To boost the performance further, the proposed DFT$^+$ method adopts an ensemble scheme to use both mid and final convolution layer outputs through DFT magnitude pooling layers. Extensive experimental results show that the DFT and DFT$^+$ based methods achieve significant improvements over the conventional algorithms in numerous classification tasks.

References

1. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Neural Information Processing Systems. (2012)
2. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. Arxiv (2014)
3. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. In: IEEE Conference on Computer Vision and Pattern Recognition. (2015)
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition. (2016)
5. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: IEEE Conference on Computer Vision and Pattern Recognition. (2016) 2818–2826
6. Bracewell, R.N., Bracewell, R.N.: The Fourier transform and its applications. Volume 31999. McGraw-Hill New York (1986)
7. Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., Oliva, A.: Learning deep features for scene recognition using places database. In: Neural Information Processing Systems. (2014) 487–495
8. Herranz, L., Jiang, S., Li, X.: Scene recognition with cnns: objects, scales and dataset bias. In: IEEE Conference on Computer Vision and Pattern Recognition. (2016) 571–579
9. Lin, T.Y., RoyChowdhury, A., Maji, S.: Bilinear cnn models for fine-grained visual recognition. In: IEEE International Conference on Computer Vision. (2015)
10. Krause, J., Jin, H., Yang, J., Fei-Fei, L.: Fine-grained recognition without part annotations. In: IEEE Conference on Computer Vision and Pattern Recognition. (2015) 5546–5555
11. Zhang, X., Xiong, H., Zhou, W., Lin, W., Tian, Q.: Picking deep filter responses for fine-grained image recognition. In: IEEE Conference on Computer Vision and Pattern Recognition. (2016)
12. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition. (2014) 580–587
13. Girshick, R.: Fast r-cnn. In: IEEE International Conference on Computer Vision. (2015) 1440–1448
14. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: IEEE Conference on Computer Vision and Pattern Recognition. (2016) 779–788
15. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition. (2015) 3431–3440
16. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. Arxiv (2016)
17. Liu, Z., Li, X., Luo, P., Loy, C.C., Tang, X.: Semantic image segmentation via deep parsing network. In: IEEE International Conference on Computer Vision. (2015)
18. Tolias, G., Sicre, R., Jégou, H.: Particular object retrieval with integral max-pooling of cnn activations. Arxiv (2015)
19. Gong, Y., Wang, L., Guo, R., Lazebnik, S.: Multi-scale orderless pooling of deep convolutional activation features. In: European Conference on Computer Vision. (2014) 392–407

20. Cimpoi, M., Maji, S., Vedaldi, A.: Deep filter banks for texture recognition and segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition. (2015) 3828–3836
21. Perronnin, F., Dance, C.: Fisher kernels on visual vocabularies for image categorization. In: IEEE Conference on Computer Vision and Pattern Recognition. (2007)
22. Simon, M., Rodner, E., Gao, Y., Darrell, T., Denzler, J.: Generalized orderless pooling performs implicit salient matching. Arxiv (2017)
23. Ionescu, C., Vantzos, O., Sminchisescu, C.: Matrix backpropagation for deep networks with structured layers. In: IEEE International Conference on Computer Vision. (2015) 2965–2973
24. Gao, Y., Beijbom, O., Zhang, N., Darrell, T.: Compact bilinear pooling. In: IEEE Conference on Computer Vision and Pattern Recognition. (2016) 317–326
25. Cui, Y., Zhou, F., Wang, J., Liu, X., Lin, Y., Belongie, S.J.: Kernel pooling for convolutional neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition. (2017)
26. Rippel, O., Snoek, J., Adams, R.P.: Spectral representations for convolutional neural networks. In: Neural Information Processing Systems. (2015) 2449–2457
27. Zheng, L., Zhao, Y., Wang, S., Wang, J., Tian, Q.: Good practice in cnn feature transfer. Arxiv (2016)
28. Hariharan, B., Arbeláez, P., Girshick, R., Malik, J.: Hypercolumns for object segmentation and fine-grained localization. In: IEEE Conference on Computer Vision and Pattern Recognition. (2015)
29. Simonyan, K., Zisserman, A.: Two-stream convolutional networks for action recognition in videos. In: Neural Information Processing Systems. (2014) 568–576
30. Zhang, H., Xue, J., Dana, K.: Deep ten: Texture encoding network. In: IEEE Conference on Computer Vision and Pattern Recognition. (2017)
31. Sharan, L., Rosenholtz, R., Adelson, E.: Material perception: What can you see in a brief glance? Journal of Vision 9(8) (2009) 784–784
32. Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., Vedaldi, A.: Describing textures in the wild. In: IEEE Conference on Computer Vision and Pattern Recognition. (2014) 3606–3613
33. Quattoni, A., Torralba, A.: Recognizing indoor scenes. In: IEEE Conference on Computer Vision and Pattern Recognition. (2009)
34. Lin, T.Y., Maji, S.: Visualizing and understanding deep texture representations. In: IEEE Conference on Computer Vision and Pattern Recognition. (2016) 2791–2799
35. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al.: Imagenet large scale visual recognition challenge. International Journal of Computer Vision 115(3) (2015) 211–252
36. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The caltech-ucsd birds-200-2011 dataset. Technical report (2011)
37. Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. Computer Vision and Image Understanding 106(1) (2007) 59–70