Received 17 July 2022, accepted 31 July 2022, date of publication 8 August 2022, date of current version 26 August 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3197219

RESEARCH ARTICLE

A New Pointwise Convolution in Deep Neural Networks Through Extremely Fast and Non Parametric Transforms

JOONHYUN JEONG, INCHEON CHO, EUNSEOP SHIN, AND SUNG-HO BAE, (Member, IEEE)
Department of Computer Science and Engineering, Kyung Hee University, Yongin-si 17104, Republic of Korea

Corresponding author: Sung-Ho Bae (shbae@khu.ac.kr)

This work was supported in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) Grant funded by the Korea Government Ministry of Science and ICT (MSIT) under Grant 2022-0-00759 and, in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) grant funded by the Korea Government Ministry of Science and ICT (MSIT) Artificial Intelligence Innovation Hub under Grant 2021-0-02068.

ABSTRACT Some conventional transforms such as Discrete Walsh-Hadamard Transform (DWHT) and Discrete Cosine Transform (DCT) have been widely used as feature extractors in image processing but rarely applied in neural networks. However, we found that these conventional transforms can serve as a powerful feature extractor in channel dimension without any learnable parameters in deep neural networks. This paper firstly proposes to apply conventional transforms on pointwise convolution, showing that such transforms can significantly reduce the computational complexity of neural networks without accuracy degradation on various classification tasks and even on face detection task. Our comprehensive experiments show that the proposed DWHT-based model gained 1.49% accuracy increase with 79.4% reduced parameters and 49.4% reduced FLOPs compared with its baseline model on the CIFAR 100 dataset while achieving comparable accuracy under the condition that 81.4% of parameters and 49.4% of FLOPs reduced on SVHN dataset. Additionally, our DWHT-based model showed comparable accuracy with 89.2% reduced parameters and 26.5% reduced FLOPs compared to the baseline models on WIDER FACE and FDDB datasets.

INDEX TERMS Efficient deep neural network architecture, pointwise convolution, discrete Walsh-Hadamard transform, discrete cosine transform.

I. INTRODUCTION

Large Convolutional Neural Networks (CNNs) [1]–[4], [5] and automatic Neural Architecture Search (NAS) based networks [6]–[8] have evolved to show remarkable accuracy on various tasks such as image classification [9], [10] and object detection [11] by taking advantage of huge amount of learnable parameters and computations. However, these large number of weights and high computational costs enabled only limited applications for mobile devices that are constrained by power consumption, memory space and computation costs [12].

With regard to solving these problems, [13]–[16] proposed parameter and computation efficient blocks while maintaining almost same accuracy compared to other heavy CNN models. All of these blocks utilized depthwise separable convolution which deconstructed the standard convolution sized of $(3 \times 3 \times C)$ into spatial information specific depthwise convolution $(3 \times 3 \times 1)$ and channel information specific pointwise $(1 \times 1 \times C)$ convolution. The depthwise separable convolution achieved comparable accuracy compared to standard spatial convolution with significantly reduced parameters and FLOPs. These reduced resource requirements made the depthwise separable convolution as well as pointwise convolution (PC) more widely used in modern CNN architectures.

Nevertheless, we point out that the existing PC layer is still computationally expensive and occupies a large proportion in the number of weight parameters [13]. Although the demand toward the PC layer has been and will be growing

The associate editor coordinating the review of this manuscript and approving it for publication was Shadi Alawneh.

exponentially in modern neural network architectures, there have been a few studies on improving its efficiency. [15] proposed a grouped version of PC layer which splits a feature map into groups in terms of channel dimension, for reducing number of learnable parameters. In a similar manner, [17] presented a structured version of group PC layer with divide and conquer algorithm by recursively halving feature maps in channel dimensions. However, these previous works require learnable parameters for the PC layer (parametric-PC).

In this paper, we propose a new PC layer formulated by non-parametric and extremely fast conventional transforms. While deep neural networks are capable of extracting distinctive feature representations [18]–[23], the conventional time-to-frequency transforms have importantly been used as feature extractor in image processing and video compression due to their ability to reduce dimensionality of signals by concentrating energy into the low-frequency regions [24], [25] and to extract local features by decomposing image signals into various texture types [26]. Also, there are fast transform algorithms (e.g., Cooley-Tukey algorithm [27]) which significantly reduce the computation complexity by reusing the output of the previous step in butterfly-like pipeline architectures.

We point out that, although the transform algorithms have shown promising performance in feature representation and dimensionality reduction, they have hardly been incorporated into CNNs [28]. In this paper, we aim to answer following question: Can conventional transforms (e.g. Discrete Walsh-Hadamard Transform (DWHT) and Discrete Cosine Transform (DCT)) which were frequently used as spatial feature extractors [26], [29]–[31] also serve as a feature extractor in channel dimension of deep neural networks?

Through comprehensive experiments, we found that, although both of these transforms do not require learnable parameters, they can effectively capture the feature representation in channel dimension. Specifically, without any learnable parameters, orthogonality of these conventional transforms helps to reduce feature representational bottleneck (See Section IV-C1). Therefore, proposed PC layer equipped with these conventional transforms can sufficiently extract feature information in channel dimension. Also, this non-parametric property enables our proposed CNN models to be significantly compressed in terms of the number of parameters, allowing CNNs to be applied in low-power and complexity applications (i.e., efficient distributed training, less communication between server and clients), [32]. We note that especially DWHT is considered to be a good replacement of the conventional PC layer, as it requires no floating point multiplications but only additions and subtractions (i.e., multiplication with binarized weights in $+1/-1$) by which the computation overheads of PC layers can be significantly reduced. Furthermore, DWHT can take a strong advantage of its fast version where the computational complexity of the floating point operations is reduced from $\mathcal{O}(n^2)$ to $\mathcal{O}(n \log n)$. These non-parametric and low computational properties construct extremely efficient neural network from the perspective of parameter and computation as well as enjoying accuracy gain.

Our contributions are summarized as follows:

- We propose a new PC layer formulated with conventional transforms which can significantly reduce computational resources (memory usage, FLOPs).
- We demonstrate effectiveness of our proposed PC layer compared to conventional PC layer in terms of accuracy and computational resources on various classification tasks and even on face detection task.
- We investigate the optimal block structure and network hierarchy position for our prposed PC layer, along with analysis on orthogonality, which helps to reduce feature representation bottleneck.

II. RELATED WORK

A. DECONSTRUCTION AND DECOMPOSITION OF CONVOLUTIONS

For reducing computational complexity of the existing convolution methods, several approaches of rethinking and deconstructing the naive convolution structures have been proposed. [2] factorized a large sized kernel (e.g., 5×5) in a convolution layer into several convolution layers with small sized (3×3) kernels. [33] pointed out the limitation of existing convolution in the fixed receptive field. Consequently, they introduced learnable spatial displacement parameters, showing flexibility of dilation in the convolution layers. Based on [33], [34] proved that the standard convolution can effectively be deconstructed as a single PC layer with the spatially shifted channels. Based on that, they proposed a very efficient convolution layer, namely active shift layer, by replacing spatial convolutions with shift operations.

It is worth noting that the existing PC layer takes the huge proportion of computation and the number of weight parameters in modern lightweight CNN models [13], [14], [16]. Specifically, MobileNet-V1 [13] requires 94%, 74% of the overall computational cost and the overall number of weight parameters for the PC layer, respectively. Therefore, there were attempts to reduce computational complexity of the PC layer. [15] proposed ShuffleNet-V1 where the features are decomposed into several groups over channels and the PC operation was conducted for each group, thus reducing the number of weight parameters and FLOPs by the number of groups G. However, it was proved in [16] that the memory access cost increases as G increases, leading to slower inference speed. Similarly to the aforementioned methods, our work is to reduce computational complexity and the number of weight parameters in a convolution layer. However, our objective is more oriented on finding out mathematically efficient algorithms based on fast divide and conquer algorithms by utilizing the property of fixed harmonic kernels.

B. QUANTIZATION

In neural networks, quantization has been used to reduce the number of bits in weights and/or activations. [35] applied

8-bit quantization on weight parameters, which enabled considerable speed-up with small drop of accuracy. [36] applied 16-bit fixed point representation with stochastic rounding. Based on [37] which pruned the unimportant weight connections through thresholding the values of weight, [38] successfully integrated the pruning, 8 (or less) bit quantization and huffman encoding. The extreme case of quantized networks was evolved from [39], which approximated weights with the binary $(+1, -1)$ values. From [39] as the milestone, [40], [41] constructed Binarized Neural Networks (BNN) which stochastically binarize the real valued weights and activations during training. These binarized weights and activations lead to significantly fast run-time by replacing floating point multiplications with 1-bit XNOR operations.

Based on BNN [40], [41], a Local Binary CNN (LBCNN) [42] was proposed that utilizes binarized non-learnable weights in spatial convolution based on the conventional local binary patterns [43], thus replacing multiplications with addition/subtraction operations in spatial convolution. Our work shares some similarity to LBCNN [42] in using binary fixed weight values. However, the local binary patterns cannot be applied on the PC layers consisting of much larger portion of the parameters and computation in neural networks [13] compared to the spatial convolution layers. Also, applying good mathematical properties to reduce computations (e.g., the harmonic property of DCT/DWHT kernels) was not considered in LBCNN.

C. CONVENTIONAL TRANSFORMS

Several transform techniques have been applied for image processing and compression [44]–[46]. Discrete Cosine Transform (DCT) has been used as a powerful feature extractor [26]. For an N-point input sequence, the basis kernel of DCT is defined as a list of cosine values as below:

$$C_m = [\cos(\frac{(2x+1)m\pi}{2N})], \quad 0 \leq x \leq N-1 \tag{1}$$

where m is the index of a basis, and DCT captures higher frequency information in the input signal as m increases. This property led DCT to be widely applied in image/video compression techniques that emphasize the powers of image signals in low frequency regions [47].

Discrete Walsh Hadamard Transform (DWHT) is a very fast and efficient transform by using only $+1$ and -1 elements in kernels. These binary elements in kernels allow DWHT to compute without any multiplication operations but addition/subtraction operations. Therefore, DWHT has been widely used for fast feature extraction in many practical applications, such as texture image segmentation [29], face recognition [30], and video shot boundary detection [31].

Furthermore, DWHT can take advantage of a structured-wiring-based fast algorithm (Algorithm 1, Figure 12) as well as allowing very high efficiency in encoding the spatial information [48]. The basis kernel matrix of DWHT is defined using the previous kernel matrix as below:

$$H^D = \begin{pmatrix} H^{D-1} & H^{D-1} \\ H^{D-1} & -H^{D-1} \end{pmatrix}, \tag{2}$$

where $H^0 = 1$ and $D \geq 1$. In this paper, we denote H^D_m as the m-th row vector of H^D in Eq. 2. Additionally, we adopt a fast DWHT algorithm to reduce computational complexity of the PC layer in neural networks, resulting in extremely fast and efficient ones.

III. METHOD

We propose a new PC layer which is computed with conventional transforms. The conventional PC layer can be formulated as follows:

$$Z_{ijm} = W_m^\top \cdot X_{ij}, \quad 1 \leq m \leq M \tag{3}$$

where (i, j) is the spatial index, and m is the output channel index. In Eq. 3, N and M are the number of input and output channels, respectively. $X_{ij} \in \mathcal{R}^N$ is a vector of input X at the spatial index (i, j), and $W_m \in \mathcal{R}^N$ is a vector of m-th weight W in Eq. 3. For simplicity, the stride is set as 1 and the bias is omitted in Eq. 3.

Our proposed method is to replace the learnable parameters W_m with the bases in the conventional transforms. For example, replacing W_m with H^D_m in Eq. 3, we now can formulate the new multiplication-free PC layer using DWHT. Similarly, the DCT basis kernels C_m in Eq. 1 can substitute for W_m in Eq. 3, formulating another new PC layer using DCT. Note that the normalization factors in the conventional transforms are not applied in the proposed PC layer, because Batch Normalization [49] performs a normalization and a linear transform which can be viewed as a normalization in the existing transforms.

The most important benefit of the proposed method comes from the fact that the fast algorithms of the existing transforms can be applied in the proposed PC layers for further reduction of computation. Directly applying our proposed PC layers yields computational complexity of $\mathcal{O}(N^2)$. Adopting the fast algorithms, we can significantly reduce the computational complexity of the PC layer from $\mathcal{O}(N^2)$ to $\mathcal{O}(N \log N)$ without any change of the computation results.

We demonstrate the pseudo-code of our proposed fast PC layer using DWHT in Algorithm 1 based on the fast DWHT structure shown in Figure 1. In Algorithm 1, for $\log N$ iterations, the even-indexed channels and odd-indexed channels are added and subtracted in an element-wise manner, respectively. The resulting elements which were added and subtracted are placed in the first $N/2$ elements and the last $N/2$ elements of the input of next iteration, respectively. In this computation process, each iteration requires only N operations of addition and subtraction. Consequently, Algorithm 1 yields complexity of $\mathcal{O}(N \log N)$ in only addition and subtraction. Compared to the existing PC layer that requires complexity of $\mathcal{O}(N^2)$ in multiplication, our method is extremely efficient compared to the conventional PC layer in terms of computation costs (as shown in Figure 2)

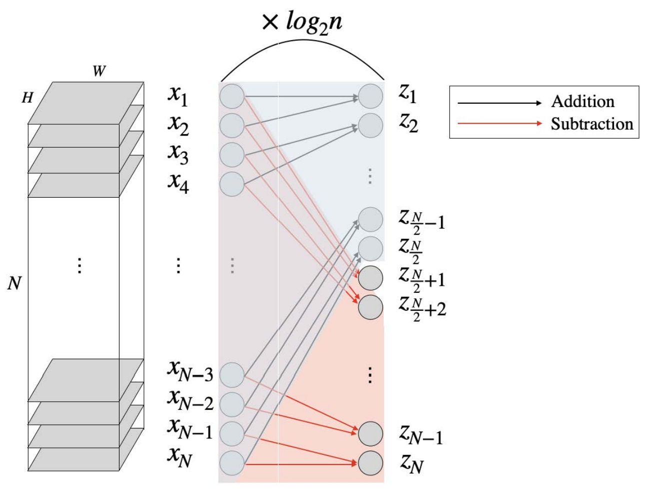


FIGURE 1. Entire architecture of our proposed fast DWHT-based PC layer. x_i denotes each of input feature map divided with channel dimension. Each of feature map numbered with even and odd channels are summed to be first half of output feature maps (i.e. sky blue shaded part) while subtracted (i.e. red shaded part) to be the other half of output feature maps. This structured addition and subtraction process is repeated $log_2 n$ times. Therefore, computational complexity reduces to $O(nlog_2 n)$ without any multiplication.

Algorithm 1 Pointwise Convolution Using Fast DWHT

```
Input: Input feature map $X \in \mathbb{R}^{B\times N\times H\times W}$
Output: Output feature map $X \in \mathbb{R}^{B\times M\times H\times W}$
 1: $n \leftarrow \log_2 N$
 2: if $N<M$ then
 3:     ZeroPad1D($X$, axis=1)    ▷ pad zeros along channel axis
 4: end if
 5: for $i \leftarrow 1$ to $n$ do
 6:     $o \leftarrow X[:, :: 2, :, :]$    ▷ Odd numbered feature maps
 7:     $e \leftarrow X[:, 1 :: 2, :, :]$    ▷ Even numbered feature maps
 8:     $X[:, : N/2, :, :] \leftarrow o + e$    ▷ Added to be first half of output
 9:     $X[:, N/2 :, :, :] \leftarrow o - e$    ▷ Subtracted to be the other half of output
10: end for
11: if $N>M$ then
12:     $X \leftarrow X[:, : M, :, :]$    ▷ Truncate along channel axis
13: end if
```

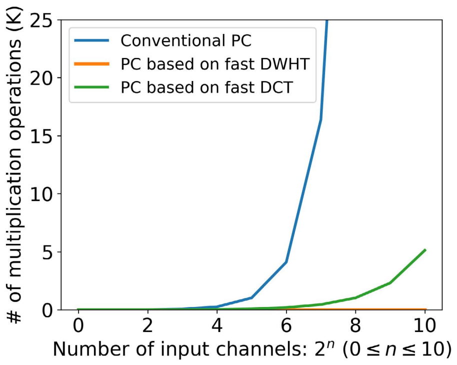


FIGURE 2. Comparison of the number of multiplications between our new PC layers and the conventional PC layer. x axis denotes logarithm of the number of input channels which range from 2^0 to 2^n. For simplicity, the number of output channels is set to be same as that of the input channel for all PC layers.

and power consumption of computing devices [50]. Note that, similarly to fast DWHT, DCT can also be computed fast with a butterfly architecture which recursively decomposes the N-point input sequence into two subproblems of $N/2$-point DCT [51].

Compared to DWHT, DCT takes advantage of using more natural shapes of cosine basis kernels, which tend to provide better feature extraction performance through capturing the frequency information. However, DCT inevitably needs multiplications for inner product between C and X vectors, and a look up table (LUT) for computing cosine kernel bases which can increase the processing time and memory access. On the other hand, as mentioned, the kernels of DWHT consist only of $+1, -1$ which allows for building a multiplication-free module. Furthermore, any memory access towards kernel bases is not needed if our structured-wiring-based fast DWHT algorithm (Algorithm 1; Figure 12) is applied. Our comprehensive experiments in Section III-A and III-B show that DWHT is more efficient than DCT when being applied in the PC layer in terms of trade-off between the computation cost and accuracy.

Note that, for securing more general formulation of our proposed PC layer, we padded zeros along the channel axis if the number of input channels is less than that of output channels while truncating the output channels when the number of output channels shrink compared to that of input channels as shown in Algorithm 1.

Figure 1 shows the architecture of the fast DWHT algorithm described in Algorithm 1. This structured-wiring-based architecture ensures that the receptive field of each output channels is N, meaning that each output channel is fully reflected against all input channels through $\log_2 N$ iterations. This property allows the proposed PC layer to fully capture the input channel correlations.

For successfully fusing the proposed PC layer into neural networks, we explore two themes: i) the optimal block search for the proposed PC layer; ii) the optimal insertion strategy of the proposed block found by i), in a hierarchical manner on the blocks of networks. We assumed that there is the optimal block unit structure and optimal hierarchy level (high-, middle-, low-level) position in the neural networks favored by these non-learnable transforms. Therefore, we conducted the experiments for the two aforementioned themes accordingly. We evaluated the effectiveness for each of our networks in accuracy according to the number of learnable weight parameters and FLOPs. For comparison, we counted total FLOPs with summation of the number of multiplications, additions and subtractions performed during the inference. Unless mentioned, we followed the default experimental setting as 128 batch size, 200 training epochs, 0.1 initial learning rate where 0.94 is multiplied per 2 epochs, and 0.9 momentum with the 5e-4 weight decay value using SGD optimizer. In all the experiments, the model accuracy was obtained by taking average of Top-1 accuracy values from three independent training results.

A. OPTIMAL BLOCK STRUCTURE FOR THE CONVENTIONAL TRANSFORMS

From a microscopic perspective, a block is the basic unit of neural networks, and it determines the efficiency of the weight parameter space and computation costs in terms of accuracy. Accordingly, to find the optimal block structure for our proposed PC layer, we perform comprehensive experiments to find out the optimal block including the proposed layer based on ShuffleNet-V2 [16]. The proposed block and its variant blocks are listed in Figure 3. As shown in (c) and (d) of Table 1, the ReLU [52] activation function significantly harms the accuracy of our neural networks equipped with the conventional transforms. This is because the harmonic kernels in conventional transforms tend to produce symmetric distributions with zero mean where much information can be eliminated by rectifying the negative valued coefficients. Further analysis on the reason for this phenomenon is described in Section IV-A. Additionally, we find out that the proposed PC layer yields approximately 1.16% higher accuracy compared to the PC layer with randomly initialized and fixed weights as shown in Table 1. These results imply that DWHT and DCT kernels can better extract better feature representations in channel dimension compared to the kernels which are randomly initialized and non-learnable. Compared to the baseline model in Table 1, DCT w/o ReLU and DWHT w/o ReLU blocks yield approximately 2.3% accuracy drop under the condition that 42% and 49.5% of learnable weight parameters and FLOPs are reduced, respectively. These results imply that the proposed blocks (i.e., (c) and (d) in Figure 3) are still inefficient in trade-off between accuracy and computation costs of neural networks, leading us to more exploring to search the optimal neural network architecture for the proposed PC layer. In the next subsection, we address this problem through applying conventional transforms on the optimal hierarchy level features (See Section III-B). Based on our comprehensive experiments, we set the block structure (d) as our default proposed block which will be exploited in all the following experiments.

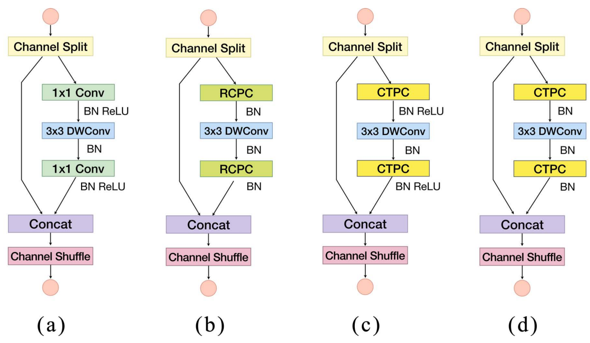


FIGURE 3. Structures of the block units under test: (a) the basic block of ShuffleNet-V2; (b) the block using random constant pointwise convolution (RCPC) layers; (c) the block using conventional transform pointwise convolution (CTPC) layers with ReLU applied after each of CTPC layer; (d) our proposed block using CTPC layers without ReLU. (b) is the block with randomly initialized weights by the uniform distribution $\mathcal{U}(-1/\sqrt{N/2}, 1/\sqrt{N/2})$ in the PC layer, where N is the number of input channels. These random weights are fixed during training.

TABLE 1. Performance result of the block units in Figure 3 on CIFAR100 dataset. All the experimented models are based on ShuffleNet-V2 with width hyper-parameter 1.1× which we customized to make the number of output channels in Stage-2, -3, -4 as 128, 256, 512, respectively, for comparison with DWHT having 2^n input channels. We replaced all of 13 basic blocks with stride 1 (i.e., (a) block) in the baseline model with (b), (c), (d) blocks, respectively. (c)-DWHT w/ ReLU denotes that CTPC layer in (c) block is based on DWHT, while (d)-DCT w/o ReLU denotes that CTPC layer in (d) block is based on DCT.

Model	Top-1 Acc (%)	# weights (ratio)	# FLOPs (ratio)
(a)-baseline	71.68 ± 0.26	1.57M (1x)	102.9M (1x)
(b)-RCPC w/o ReLU	68.16 ± 0.07	0.92M (0.58x)	102.9M (1x)
(c)-DWHT w/ ReLU	66.2 ± 0.22	0.92M (0.58x)	50.2M (0.48x)
(c)-DCT w/ ReLU	66.55 ± 0.5	0.92M (0.58x)	54.7M (0.53x)
(d)-DWHT w/o ReLU	69.42 ± 0.31	0.92M (0.58x)	50.2M (0.48x)
(d)-DCT w/o ReLU	69.23 ± 0.14	0.92M (0.58x)	54.7M (0.53x)

B. OPTIMAL POSITION FOR THE PROPOSED BLOCKS IN HIERARCHY LEVEL

In this section, we search on the optimal position of the proposed blocks in hierarchy level of neural networks. The optimal hierarchy level is defined such that the proposed networks have the minimal number of learnable weight parameters and FLOPs without accuracy drop. It is noted that applying our proposed block on the high-level position in the network provides much more reduced number of parameters and FLOPs rather than applying it on low-level position, because channel depth increases exponentially as the layer goes deeper in the network.

In Figure 4, we applied our optimal block (i.e., (d) block in Figure 3) on high-, middle- and low-level positions, respectively. In our experiments, we evaluate the performance of the networks depending on the number of blocks where the proposed optimal block is applied. The model under test is denoted as (transform type)-(# of the proposed blocks)-(hierarchy level in Low (L), Middle (M), and High (H) where the proposed optimal block is applied). For example, DWHT-3-L indicates the neural network model where the first three blocks in ShuffleNet-V2 consist of the proposed blocks, while the other blocks are the original blocks of ShuffleNet-V2. We fixed all the blocks with stride of 2 in the baseline model as in the original ShuffleNet-V2 blocks.

Figure 4 shows the performance of the proposed methods depending on the transform types {DCT, DWHT}, hierarchy level positions {L, M, H} and the number of the proposed blocks that replace the original ones in the baseline {3, 6, 10} in terms of Top-1 accuracy and the number of learnable weight parameters (or FLOPs). Since the baseline model has

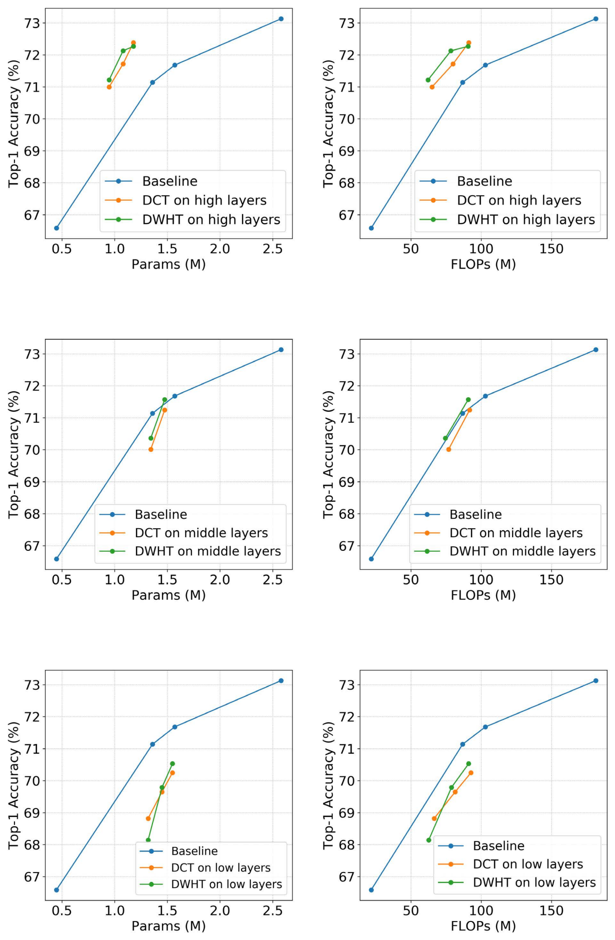


FIGURE 4. Performance curve of the proposed networks according to the hierarchy level of the block position on CIFAR100, Left: in the viewpoint of the number of learnable weight parameters (Params), Right: in the viewpoint of the number of FLOPs. The performance of the baseline models was evaluated by ShuffleNet-V2 with the width hyper-parameter of 0.5×, 1×, 1.1×, 1.5×. Our models are all experimented with 1.1× in width, and each dot in the figures represents the mean accuracy for the 3 network instances. Note that the upper left part from the blue line (baseline performance) is the superior region (i.e., better trade off between accuracy and computation resources) while lower right part from blue line is the inferior region.

only 7 blocks in the middle-level Stage (i.e., Stage-3), we performed the middle-level experiments only for DCT/DWHT-3-M and -7-M models where the proposed blocks are applied from the end of Stage-3 in the baseline model. In Figure 4, the performance of our 10-H (or 10-L), 6-H (or 6-L), 3-H (or 3-L) models (7-M and 3-M only for middle-level experiments) is listed in ascending order of the number of learnable weight parameters and FLOPs.

As shown in the first column of Figure 4, the proposed block achieved much better trade-off between the number of learnable weight parameters (or FLOPs) and accuracy on the high-level position compared to the baseline models. Meanwhile, applying the proposed block on middle- and low-level features yields slightly and severely worse performance in trade-off between accuracy and the number of parameters (or

TABLE 2. Performance result of hierarchically applying our optimal block on CIFAR100 dataset. All the models are based on MobileNet-V1 with the width hyper-parameter of 1×. We replaced both stride 1, 2 blocks in the baseline model with the optimal block that consists of [3 × 3 depthwise convolution - Batch Normalization - ReLU - CTPC - Batch Normalization] in series.

Model	Top-1 Acc (%)	# Weights (ratio)	# FLOPs (ratio)
Baseline	67.15 ± 0.3	3.31M (1x)	92.4M (1x)
DWHT-3-H	68.19 ± 0.35	1.47M (0.44x)	71.6M (0.77x)
DCT-3-H	68.21 ± 0.19	1.47M (0.44x)	72M (0.78x)
DWHT-6-H	68.65 ± 0.27	0.68M (0.2x)	46.7M (0.5x)
DCT-6-H	67.95 ± 0.53	0.68M (0.2x)	47.7M (0.51x)

FLOPs), respectively. This tendency is shown similarly for both DWHT-based models and DCT-based models, implying that there can be the optimal hierarchy level position of blocks favored by conventional transforms. We conjecture that enforcing learnable weight kernels of low level layers to be fixed during training impedes the low level features to play a principle role in maximally extracting information from input [53] and prevents rich information flowing from low-level to high-level layers thus leading to accuracy degradation due to the information bottleneck.

We also note that our DWHT-based models showed slightly higher or same accuracy with less FLOPs in all the hierarchy level positions compared to our DCT-based models. This is because the fast version of DWHT does not require any multiplication but needs a small amount of addition and subtraction operations compared to the fast version of DCT while it also has the sufficient ability as a feature extractor in channel dimension with the exquisite wiring-based structure (Figure 1).

For verifying the generality of the proposed method, we also applied our methods into MobileNet-V1. Inspired by the above results showing that the optimal hierarchy level position for conventional transforms can be found in the high-level, we replaced high-level blocks of baseline model (MobileNet-V1) to verify the effectiveness of the proposed method. The experimental results are described in Table 2. Remarkably, as shown in Table 2, our DWHT-6-H model yielded the 1.49% increase in Top-1 accuracy even under the condition that the 79.4% of parameters and 49.4% of FLOPs are reduced compared with the baseline 1× model. This outstanding performance improvement comes from the depthwise separable convolutions used in MobileNet-V1, where the PC layers play dominant roles in computation costs and memory space, i.e., they consume 94.86% in FLOPs and 74% in the total number of parameters in the whole network [13]. The full performance results for all the hierarchy level positions {L, M, H} and the number of blocks {3, 6, 10} (exceptionally, {3, 7} blocks for the middle level experiments) are described in Appendix A.

We further applied our MobileNet-V1 based models on SVHN dataset [54] in table 3. As on CIFAR100, we note the tendency that applying conventional transforms on high-level layers enables the baseline model to be extremely lightweight and computationally efficient also maintains on svhn dataset.

TABLE 3. Performance result of hierarchically applying our optimal block on SVHN dataset. The experimental settings are the same as in table 2.

Model	Top-1 Acc (%)	# Weights (ratio)	# FLOPs (ratio)
Baseline	95 ± 0.09	3.22M (1x)	92.4M (1x)
DWHT-3-H	95.23 ± 0.18	1.38M (0.43x)	71.6M (0.77x)
DCT-3-H	95.31 ± 0.07	1.38M (0.43x)	72M (0.78x)
DWHT-6-H	95.29 ± 0.08	0.59M (0.18x)	46.7M (0.5x)
DCT-6-H	95.24 ± 0.07	0.59M (0.18x)	47.7M (0.51x)

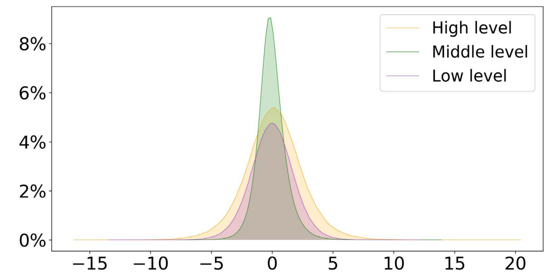
Especially, DWHT-6-H model showed comparable accuracy with the baseline model under the condition that 81.4% of parameters and 49.4% of FLOPs are reduced.

IV. EXPERIMENTS AND ANALYSIS

In this section, we analyze the significant accuracy degradation of applying ReLU after our proposed PC layer. We also analyze the active utilization of 3×3 depthwise convolution weight kernel values which takes an auxiliary role for conventional transform being non-learnable. Additionally, not only for classification task, we demonstrate task domain generality of the proposed method on face detection task with extensive experiments.

A. HINDRANCE OF ReLU IN FEATURE REPRESENTABILITY

As shown in Table 1, applying ReLU after conventional transforms significantly harmed the accuracy. This is due to the properties of conventional transform basis kernels that both H_m^D in Eq. 2 and C_m in Eq. 1 have the same number of positive and negative parameters in the kernels except for $m = 0$ and that the distributions of absolute values of positive and negative elements in kernels are almost identical. These properties imply that the output channel elements that have under zero value should also be considered during the forward pass; when forwarding X_{ij} in Eq. 3 through the conventional transforms if some important channel elements in X_{ij} that have larger values than others are combined with negative values of C_m or H_m^D, the important feature information in the output Z_{ijm} in Eq. 3 can reside in the value range under zero. Figure 5 shows that all the hierarchy level activations from both DCT and DWHT based PC layer have not only positive values but also negative values in almost same proportion. These negative values possibly include important feature information in channel dimension. Thus, applying ReLU on activations of PC layers which are based on conventional transforms discards crucial feature information contained in negative values that must be forwarded through, leading to significant accuracy drop as shown in the results of Table 1. Figure 6 demonstrates above theoretical analysis by showing that as the negative valued coefficients are fully rectified (i.e., $F = \text{ReLU}$), the accuracy is significantly degraded while fully reflecting the negative valued coefficients (i.e., $g = 1$) shows the best accuracy. From above kernel value based analysis and its experiments, we do not use non-linear activation function after the proposed PC layer.



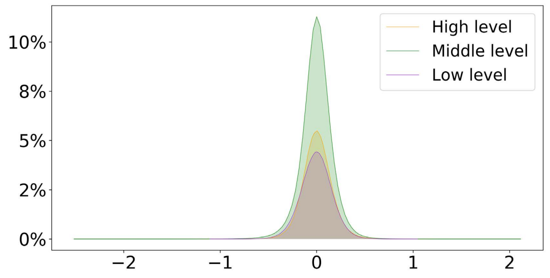


FIGURE 5. Histograms of hierarchy level (low-level, middle-level, high-level) activations after the proposed PC layer based on conventional transforms, Top: DWHT, Bottom: DCT. Both DWHT and DCT models are based on ShuffleNet V2 1.1× model where we replaced all of stride 1 blocks with (d)-DWHT w/o ReLU and (d)-DCT w/o ReLU blocks, respectively in Figure 3.

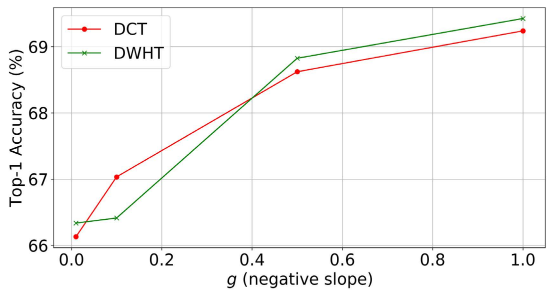


FIGURE 6. Ablation study of negative slope term g in activation function F, which is defined as $F(x) = max(0, x) + g * min(0, x)$. The performance of models were evaluated based on {DCT or DWHT}-13-H ShuffleNet-V2 1.1× where we applied F as an activation function after every DCT or DWHT based PC layer and Batch Normalization layer.

B. ACTIVE 3 × 3 DEPTHWISE CONVOLUTION WEIGHTS

In Figure 7 and Appendix B, it is observed that 3×3 depthwise convolution weights of last 3 blocks in DWHT-3-H and DCT-3-H have much less near zero values than that of baseline model. That is, the number of values which are apart from near-zero is much larger on DCT-3-H and DWHT-3-H models than on baseline model. We conjecture that these learnable weights whose values are apart from near-zero were actively fitted to the optimal domain that is favored by conventional transforms. Consequently, these weights are actively and sufficiently utilized to take the auxiliary role for conventional transforms which are non-learnable, deriving accuracy increase compared to the conventional PC layer as shown in Figure 4.

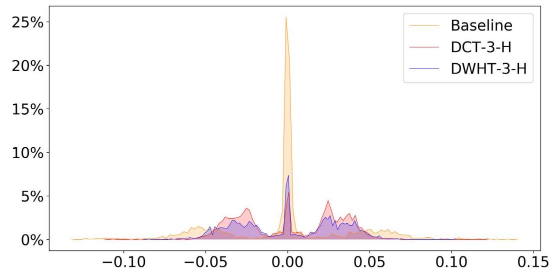


FIGURE 7. **Histogram of 3 × 3 depthwise convolution weights in the third block, out of last 3 blocks. DCT-3-H and DWHT-3-H models are based on ShuffleNet V2 1.1× model with (d) block. Baseline model is ShuffleNet V2 1.1× model.**

To verify the impact of activeness of these 3 × 3 depthwise convolution weights in the last 3 blocks, we experimented with regularizing these weights varying the weight decay values. Higher weight decay values strongly regularize the scale of 3 × 3 depthwise convolution weight values in the last 3 blocks. Thus, strong constraint on the scale of these weight values hinders active utilization of these weights, which results in accuracy drop as shown in Figure 8.

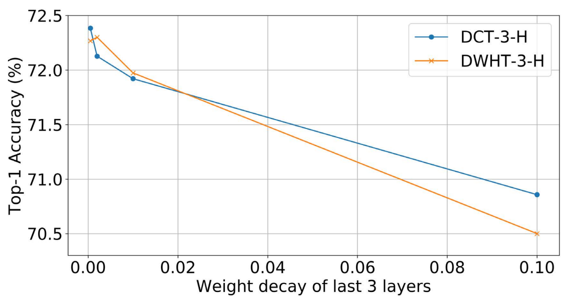


FIGURE 8. **Ablation study of weight decay values (5e-4, 2e-3, 1e-2, 1e-1). We applied these weight decay values only on 3 × 3 depthwise convolution weights of last 3 blocks in DCT-based model and DWHT-based model, while all the other learnable weights were regularized with weight decay of 5e-4.**

C. ORTHOGONALITY

In this section, we show that DCT and DWHT based PC layers can efficiently regularize the deep neural networks and reduce the representational bottleneck by its orthogonality. Formally, Orthogonality in DCT and DWHT based PC layer is given as below:

$$W_i \cdot W_j = 0 \tag{4}$$

where $i \neq j$ and $0 \leq i, j \leq M$ and W is C in Eq. 1 or H^D in Eq. 2.

1) RANK ANALYSIS

Previous works [55]–[59] showed the regularization effect of orthogonal kernel matrix, which enables faster convergence in training and consequently improved the accuracy. Moreover, Orthogonality ensures the kernel matrix to be full rank, which helps to reduce the representational bottleneck in the feature maps [60], [61]. Formally, we can rewrite the Eq. 3 as matrix multiplication as below:

$$Z = WX \tag{5}$$

where $W \in \mathbb{R}^{M \times N}$, $X \in \mathbb{R}^{N \times whB}$, B is the number of batch size and w, h is width and height of the input X. $Rank(Z)$ is upper bounded to $min(M, N)$ (assuming $whB \gg N$). The weight of DCT and DWHT based PC layers have full rank property originated from the orthogonality, enabling $Rank(Z)$ to have its upper-bound value (i.e. $min(M, N)$), while conventional PC layers are not ensured to have its upper-bound value. As the $Rank(Z)$ is maximized, it helps resolving the representational bottleneck which hinders the discriminative encoding of feature maps in channel dimension [60]. Without any cost, DCT and DWHT based PC layers naturally have orthogonal filter groups for each output channels (i.e. inter-channel orthogonality [55]). With this inter-channel orthogonality, DCT and DWHT based PC layers not only reduce redundancy and ensure diversity of kernels but also resolve the representational bottleneck in the channel dimension. Consequently, DCT and DWHT based PC layers showed better performance than conventional PC layers as shown in Table 2 and 3.

2) IMPACT OF ORTHOGONALITY

In order to verify the effect of inter-channel orthogonality in DCT and DWHT based PC layers, we enforced the inter-channel orthogonality to be destroyed in the DCT and DWHT based PC layers by maximizing the orthogonality regularization term as below:

$$\arg\max_{W} \lambda ||WW^T - I||_F^2 \tag{6}$$

where λ is the regularization coefficient and W is the kernel matrix of DCT (i.e. C in Eq. 1) and DWHT (i.e. H^D in Eq. 2) based PC layers. λ is decayed 0.1, 0.01 and 0.0001 at 20, 50 and 70 epoch, respectively following the observation of [55]. As shown in Figure 9, DCT and DWHT based models always suffered from significant accuracy degradation when the orthogonality is destroyed. Accuracy is severely degraded even when the orthogonality is weakly destroyed. Without the inter-channel orthogonality, DCT and DWHT cannot sufficiently extract feature representations in channel dimension due to lack of equipping the diverse filters and suffer from the representational bottleneck in feature maps, consequently showing a severe accuracy loss.

D. COMPARISON WITH OTHER ORTHOGONAL KERNELS

As the kernels of DCT and DWHT have inter-channel orthogonality, we compared our DCT and DWHT based PC layers with other orthogonal kernel matrices regularized with Soft Orthogonality [59], [62], [63] (namely, SO) and Spectral Restricted Isometry Property [55] (namely, SRIP).

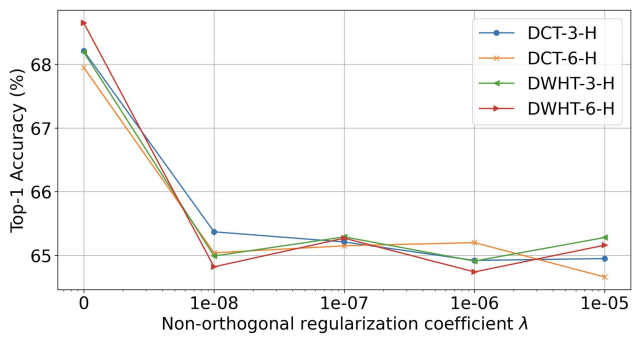


FIGURE 9. Ablation study of non-orthogonal coefficient λ (1e-8, 1e-7, 1e-6, 1e-5). We applied the non-orthogonal regularization term only on DCT or DWHT based PC layers. The performance of models were evaluated with MobileNet-V1 1$\times$ model on CIFAR100 dataset.

1) SETUP

We randomly initialized the weights of corresponding PC layers as RCPC layer in Figure 3. These weights are then regularized to be orthogonal for every training iteration with SO or SRIP method. For fair comparison with DCT and DWHT based PC layers, gradients for cross entropy loss are not updated on these layers.

2) RESULTS

In Table 4, DCT and DWHT based models obviously showed better accuracy than other orthogonal-regularized kernel matrices, which demonstrates the superior ability as a powerful feature extractor in such well-designed and orthogonal DCT and DWHT based PC layers over other orthogonal filters.

TABLE 4. Performance comparison between DCT, DWHT based PC layers and other orthogonal-regularized PC layers. The performance of models were evaluated with MobileNet-V1 1$\times$ model on CIFAR100 dataset.

Model	Orthogonal Regularization	Top-1 Acc (%)
DCT-3-H	-	68.21
DWHT-3-H	-	68.19
RCPC-3-H	SO	65.59
RCPC-3-H	SRIP	65.37
DCT-6-H	-	67.95
DWHT-6-H	-	68.65
RCPC-6-H	SO	66.54
RCPC-6-H	SRIP	65.30

E. FACE DETECTION

In order to demonstrate the domain-generality of the proposed method, we conducted comprehensive experiments on applying our proposed PC layers to object detection, specifically to the face detection task.

1) SETUP

For the face detection schemes such as anchor design, data augmentation and feature-map resolution design, we followed [64] which is one of the baseline methods in face detection field. It is noted that there is a huge demand on real-time face detection algorithms having high detection accuracy, which leads us to applying our PC layers to a lightweight face detection network. Therefore, instead of using VGG16 [2] as backbone network as in [64], we set MobileNet-V1 0.25$\times$ as our baseline backbone model where extra depthwise separable blocks are added for detecting more diverse scales of face in the images. In this baseline model, we replaced the conventional PC layers within last 3, 6 blocks with our DCT/DWHT based PC layers. We trained all the models on the WIDER FACE [65] train dataset and evaluated on WIDER FACE validation dataset and Face Detection Data Set and Benchmark (FDDB) dataset [66].

2) RESULTS

In Table 5, our DWHT-3-H and DWHT-6-H models showed comparable or even higher mAP values than the baseline model on all the WIDER FACE subsets with significantly reduced number of learnable parameters and FLOPs. Especially, DWHT-3-H model achieved 0.27% higher mAP than the baseline model under the condition that 79% of parameters and 16% of FLOPs are reduced on Hard subset. Regarding DCT-3-H and DCT-6-H models, they showed a solid improvement of mAP on Easy and Medium subsets with significantly reduced number of parameters and FLOPs compared to the baseline model. Furthermore, on FDDB dataset, our DWHT-6-H and DWHT-3-H models showed comparable or even 0.09% higher AP than the baseline model with significantly reduced number of learnable parameters and FLOPs. Our DCT-based models showed a small degree of degradation in AP compared to the baseline model, which is a mild degradation considering the reduced amount of parameters and FLOPs. Consequently, our comprehensive experiments on both datasets reveal the generality of our proposed method, enabling neural networks to be extremely lightweight and reduce the computational overhead.

TABLE 5. Quantitative comparison between the baseline model and our DCT/DWHT-based models on WIDER FACE validation dataset and FDDB dataset. For WIDER FACE dataset, we evaluate mAP of Easy, Medium and Hard subsets which correspond to large, medium and small scale faces, respectively. For FDDB dataset, AP means the true positive rate at 1,000 false positives and all the models were evaluated with discontinuous criterion.

Model	WIDER FACE			FDDB	# Weights (ratio)	# FLOPs (ratio)
	Easy (mAP%)	Mid (mAP%)	Hard (mAP%)	AP (%)		
Baseline	89.74	86.3	64.11	94.41	2.32M (1x)	129.4M (1x)
DWHT-3-H	89.82	86.27	64.38	94.5	0.48M (0.2x)	108.6M (0.84x)
DCT-3-H	90.25	86.83	61.49	93.79	0.48M (0.2x)	109M (0.84x)
DWHT-6-H	89.98	86.29	63.63	94.37	0.25M (0.11x)	95.1M (0.73x)
DCT-6-H	90.11	86.71	62.16	93.67	0.25M (0.11x)	96.3M (0.74x)

V. CONCLUSION

We propose the new PC layers through conventional transforms, which allow the neural networks to be efficient in complexity of computation and learnable weight parameters.

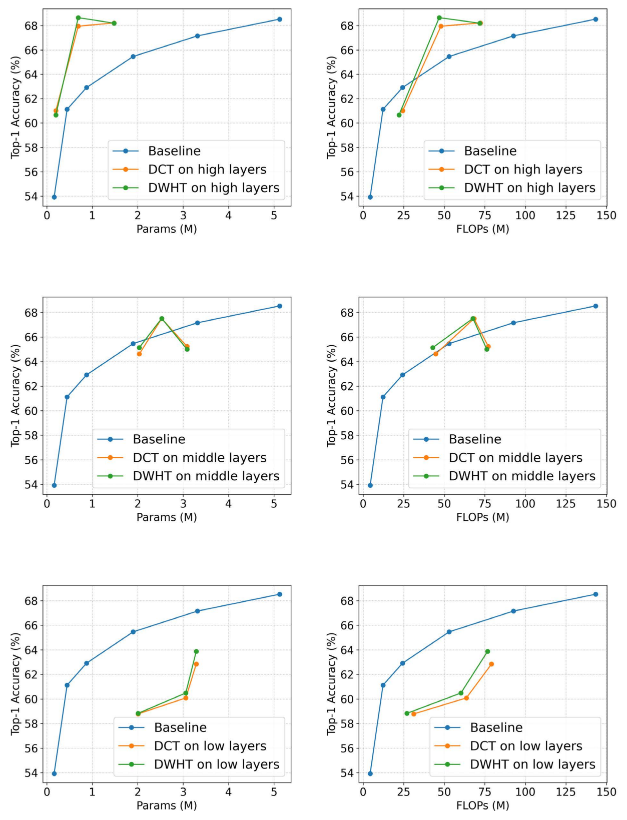


FIGURE 10. Performance curve of the proposed networks according to the hierarchy level of the block position on CIFAR100, Left: in the viewpoint of the number of learnable weight parameters, Right: in the viewpoint of the number of FLOPs. The performance of baseline models was evaluated by MobileNet-V1 architecture with width hyper-parameter 0.2×, 0.35×, 0.5×, 0.75×, 1×, 1.25×. Our proposed models were all experimented with 1× setting, and each dot in the figures represents mean accuracy of 3 network instances. Our models are experimented with 10-H, 6-H, 3-H models (first row), 7-M, 3-M-Rear, 3-M-Front models (second row) and 10-L, 6-L, 3-L models (final row), listed in ascending order of the number of learnable weight parameters and FLOPs.

With the purpose of successfully fusing our PC layers into deep neural networks, we found the optimal block unit structure and hierarchy level position in neural networks for conventional transforms, showing accuracy increase and great feature representability in channel dimension. We further revealed the hindrance of ReLU in terms of feature representation, the activeness of depthwise convolution weights on the last blocks and the effect of orthogonality in our proposed neural network. Finally, we showed the superiority of our method on the other task with the use of a low number of parameters and FLOPs.

LIMITATIONS AND FUTURE WORKS

While the scope of our method is currently restricted to small and medium scale datasets, we believe that scaling up to a much larger datasets such as ImageNet is a totally new research frontier, where a minimal amount of learnable parameters are necessarily required in PC layer due to the increased scale and difficulty of the dataset.

APPENDIX A GENERALITY OF APPLYING PROPOSED PC LAYERS IN OTHER NEURAL NETWORKS

In Figure 10, for the purpose of finding more definite hierarchy level of blocks favored by our proposed PC layers, we subdivided our middle level experiment scheme; DCT/DWHT-3-M-Front model denotes the model which applied the proposed blocks from the beginning of Stage-3 in the baseline while DCT/DWHT-3-M-Rear model denotes the model which applied from the end of Stage-3. The performance curves of all our proposed models in Figure 10 show that if we apply the proposed optimal block within the first 6 blocks in the network, the Top-1 accuracy is mildly or significantly deteriorated compared to the required computational cost and number of learnable parameters, informing us the important fact that there are the definite hierarchy level blocks which are favored or not favored by our proposed PC layers in the network.

APPENDIX B HISTOGRAM OF 3 × 3 DEPTHWISE CONVOLUTION WEIGHTS IN HIGH-LEVEL BLOCKS

See Fig. 11.

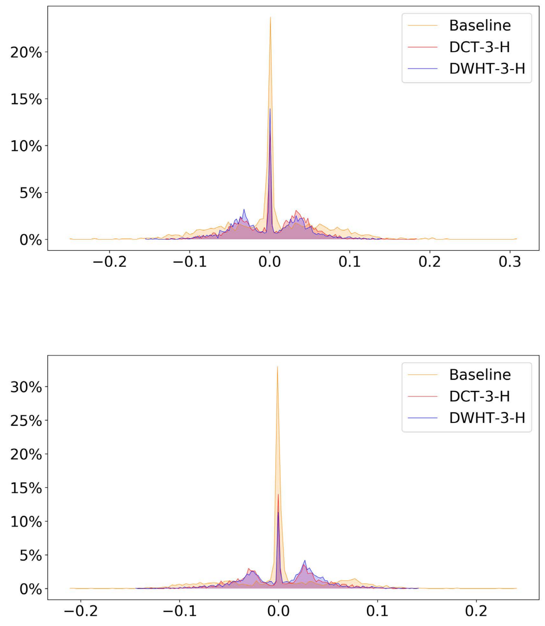


FIGURE 11. Histograms of 3 × 3 depthwise convolution weights, Top: histogram of first block out of last 3 blocks, Bottom: histogram of second block out of last 3 blocks. DWHT-3-H and DCT-3-H models are based on ShuffleNet-V2 1.1× model with (d)-DWHT w/o ReLU and (d)-DCT w/o ReLU block in Figure 3, respectively. Baseline model is ShuffleNet-V2 1.1× model.

APPENDIX C ALGORITHM FLOWCHART

See Fig. 12.

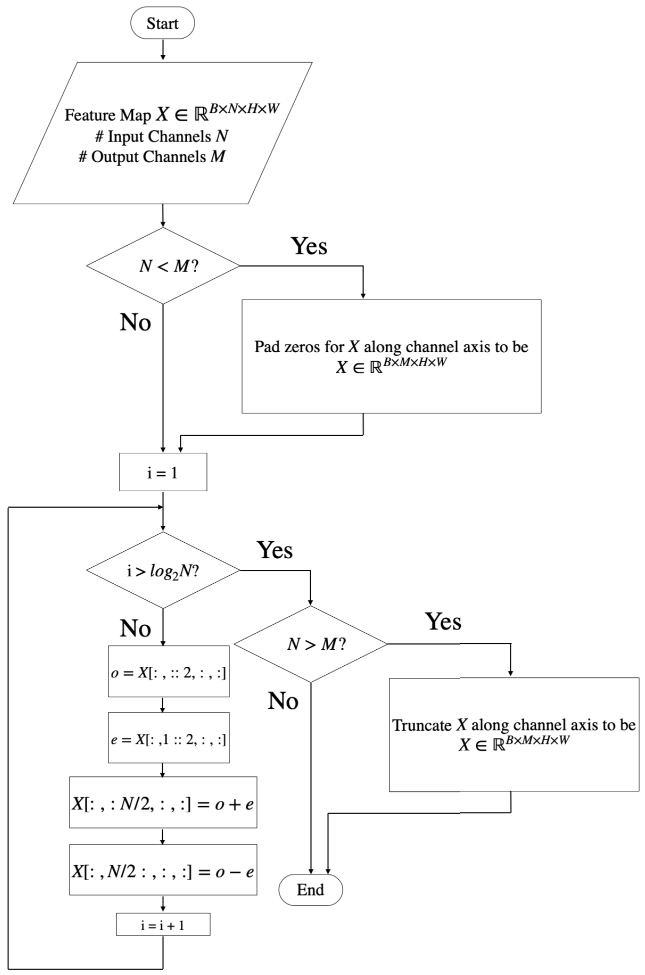

Start
Feature Map $X \in \mathbb{R}^{B\times N\times H\times W}$
# Input Channels N
# Output Channels M
$N < M$?
Yes
No
Pad zeros for X along channel axis to be $X \in \mathbb{R}^{B\times M\times H\times W}$
i = 1
i > log_2N?
Yes
No
$N > M$?
Yes
No
Truncate X along channel axis to be $X \in \mathbb{R}^{B\times M\times H\times W}$
$o = X[: , :: 2, : , :]$
$e = X[: ,1 :: 2, : , :]$
$X[: , : N/2, : , :] = o + e$
$X[: , N/2 : , : , :] = o - e$
i = i + 1
End

FIGURE 12. **Flowchart of fast DWHT-based PC layer (Figure 1, Algorithm 1).**

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.
[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*.
[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.
[4] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.
[5] C. Szegedy, S. Ioffe, and V. Vanhoucke, "Inception-v4, Inception-ResNet and the impact of residual connections on learning," 2016, *arXiv:1602.07261*.
[6] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8697–8710.
[7] C. Liu, B. Zoph, M. Neumann, J. Shlens, W. Hua, L.-J. Li, L. Fei-Fei, A. Yuille, J. Huang, and K. Murphy, "Progressive neural architecture search," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, Sep. 2018, pp. 19–34.
[8] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," 2018, *arXiv:1802.01548*.
[9] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.
[10] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.S. thesis Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, 2009.
[11] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 740–755.
[12] A. Canziani, A. Paszke, and E. Culurciello, "An analysis of deep neural network models for practical applications," 2016, *arXiv:1605.07678*.
[13] G. Andrew Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017, *arXiv:1704.04861*.
[14] M. Sandler, G. A. Howard, M. Zhu, A. Zhmoginov, and L. Chen, "Inverted residuals and linear bottlenecks: Mobile networks for classification, detection and segmentation," 2018, *arXiv:1801.04381*.
[15] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," 2017, *arXiv:1707.01083*.
[16] N. Ma, X. Zhang, H. Zheng, and J. Sun, "ShuffleNet V2: Practical guidelines for efficient CNN architecture design," 2018, *arXiv:1807.11164*.
[17] K. Alizadeh Vahid, A. Prabhu, A. Farhadi, and M. Rastegari, "Butterfly transform: An efficient FFT based neural architecture design," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 12021–12030.
[18] M. Amrani, A. Bey, and A. Amamra, "New SAR target recognition based on Yolo and very deep multi-canonical correlation analysis," *Int. J. Remote Sens.*, pp. 1–20, Aug. 2021.
[19] M. Amrani and F. Jiang, "Deep feature extraction and combination for synthetic aperture radar target classification," *J. Appl. Remote Sens.*, vol. 11, no. 4, 2017, Art. no. 042616.
[20] M. Amrani, S. Chaib, I. Omara, and F. Jiang, "Bag-of-visual-words based feature extraction for SAR target classification," *Proc. SPIE*, vol. 10420, pp. 327–332, Jul. 2017.
[21] M. Amrani, K. Yang, D. Zhao, X. Fan, and F. Jiang, "An efficient feature selection for SAR target classification," in *Proc. Pacific Rim Conf. Multimedia*. Cham, Switzerland: Springer, 2017, pp. 68–78.
[22] M. Amrani, F. Jiang, Y. Xu, S. Liu, and S. Zhang, "SAR-oriented visual saliency model and directed acyclic graph support vector metric based target classification," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 11, no. 10, pp. 3794–3810, Oct. 2018.
[23] M. Amrani, M. Hammad, F. Jiang, K. Wang, and A. Amrani, "Very deep feature extraction and fusion for arrhythmias detection," *Neural Comput. Appl.*, vol. 30, no. 7, pp. 2047–2057, Oct. 2018.
[24] A. Amine, S. Ghouzali, M. Rziza, and D. Aboutajdine, "Investigation of feature dimension reduction based DCT/SVM for face recognition," in *Proc. IEEE Symp. Comput. Commun.*, Jul. 2008, pp. 188–193.
[25] J. A. Tropp, "Improved analysis of the subsampled randomized Hadamard transform," *Adv. Adapt. Data Anal.*, vol. 3, nos. 1–2, pp. 115–126, Apr. 2011.
[26] S. Dabbaghchian, M. P. Ghaemmaghami, and A. Aghagolzadeh, "Feature extraction using discrete cosine transform and discrimination power analysis with a face recognition technology," *Pattern Recognit.*, vol. 43, no. 4, pp. 1431–1440, 2010.
[27] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Math. Comput.*, vol. 19, no. 90, pp. 297–301, Apr. 1965.
[28] A. Ghosh and R. Chellappa, "Deep feature extraction in the DCT domain," in *Proc. 23rd Int. Conf. Pattern Recognit. (ICPR)*, Dec. 2016, pp. 3536–3541.
[29] A. Vard, A. Monadjemi, K. Jamshidi, and N. Movahhedinia, "Fast texture energy based image segmentation using directional Walsh–Hadamard transform and parametric active contour models," *Expert Syst. Appl.*, vol. 38, no. 9, pp. 11722–11729, Sep. 2011.
[30] M. Hassan, I. Osman, and M. Yahia, "Walsh–Hadamard transform for facial feature extraction in face recognition," *World Acad. Sci., Eng. Technol.*, vol. 29, pp. 194–198, May 2007.
[31] L. P. Gg and S. Domnic, "Walsh–Hadamard transform kernel-based feature vector for shot boundary detection," *IEEE Trans. Image Process.*, vol. 23, no. 12, pp. 5187–5197, Dec. 2014.
[32] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," 2016, *arXiv:1602.07360*.
[33] Y. Jeon and J. Kim, "Active convolution: Learning the shape of convolution for image classification," 2017, *arXiv:1703.09076*.
[34] Y. Jeon and J. Kim, "Constructing fast network through deconstruction of convolution," 2018, *arXiv:1806.07370*.

[35] V. Vanhoucke, A. Senior, and M. Mao, "Improving the speed of neural networks on CPUs," in *Proc. Deep Learn. Unsupervised Feature Learn. Workshop*, 2011.
[36] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1737–1746.
[37] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 1135–1143.
[38] S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," 2015, *arXiv:1510.00149*.
[39] M. Courbariaux, Y. Bengio, and J. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," 2015, *arXiv:1511.00363*.
[40] M. Courbariaux and Y. Bengio, "BinaryNet: Training deep neural networks with weights and activations constrained to +1 or -1," 2016, *arXiv:1602.02830*.
[41] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, "Quantized neural networks: Training neural networks with low precision weights and activations," 2016, *arXiv:1609.07061*.
[42] F. Juefei-Xu, V. N. Boddeti, and M. Savvides, "Local binary convolutional neural networks," 2016, *arXiv:1608.06049*.
[43] F. Juefei-Xu and M. Savvides, "Subspace-based discrete transform encoded local binary patterns representations for robust periocular matching on NIST's face recognition grand challenge," *IEEE Trans. Image Process.*, vol. 23, no. 8, pp. 3490–3505, Aug. 2014.
[44] S. Wang and J. Hu, "A Hadamard transform-based method for the design of cancellable fingerprint templates," in *Proc. 6th Int. Congr. Image Signal Process. (CISP)*, Dec. 2013, pp. 1682–1687.
[45] A. B. Watson, "Image compression using the discrete cosine transform," *Math. J.*, vol. 4, no. 1, p. 81, 1994.
[46] A. D. Andrushia and R. Thangarjan, "Saliency-based image compression using Walsh–Hadamard transform (WHT)," in *Biologically Rationalized Computing Techniques for Image Processing Applications*. Cham, Switzerland: Springer, 2018, pp. 21–42.
[47] K. R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, Applications*. New York, NY, USA: Academic, 2014.
[48] W. K. Pratt, J. Kane, and H. C. Andrews, "Hadamard transform image coding," *Proc. IEEE*, vol. 57, no. 1, pp. 58–68, Jan. 1969.
[49] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*.
[50] M. Horowitz, "1.1 Computing's energy problem (and what we can do about it)," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Feb. 2014, pp. 10–14.
[51] C. W. Kok, "Fast algorithm for computing discrete cosine transform," *IEEE Trans. Signal Process.*, vol. 45, no. 3, pp. 757–760, Mar. 1997.
[52] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn. (ICML)*, 2010, pp. 807–814.
[53] R. Shwartz-Ziv and N. Tishby, "Opening the black box of deep neural networks via information," 2017, *arXiv:1703.00810*.
[54] Y. Netzer *et al.*, "Reading digits in natural images with unsupervised feature learning," in *Proc. NIPS Workshop Deep Learn. Unsupervised Feature Learn.*, Granada, Spain, 2011, pp. 1–9.
[55] N. Bansal, X. Chen, and Z. Wang, "Can we gain more from orthogonality regularizations in training deep networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 4261–4271.
[56] P. Rodríguez, J. Gonzàlez, G. Cucurull, J. M. Gonfaus, and X. Roca, "Regularizing CNNs with locally constrained decorrelations," 2016, *arXiv:1611.01967*.
[57] D. Mishkin and J. Matas, "All you need is a good init," 2015, *arXiv:1511.06422*.
[58] L. Huang, X. Liu, B. Lang, A. W. Yu, Y. Wang, and B. Li, "Orthogonal weight normalization: Solution to optimization over multiple dependent stiefel manifolds in deep neural networks," 2017, *arXiv:1709.06079*.
[59] D. Xie, J. Xiong, and S. Pu, "All you need is beyond a good init: Exploring better solution for training extremely deep convolutional neural networks with orthonormality and modulation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6176–6185.
[60] D. Han, S. Yun, B. Heo, and Y. Yoo, "Rethinking channel dimensions for efficient model design," 2020, *arXiv:2007.00992*.
[61] Z. Yang, Z. Dai, R. Salakhutdinov, and W. W. Cohen, "Breaking the softmax bottleneck: A high-rank RNN language model," 2017, *arXiv:1711.03953*.
[62] R. Balestriero, "A spline theory of deep learning," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 374–383.
[63] R. Balestriero and R. Baraniuk, "Mad max: Affine spline insights into deep learning," 2018, *arXiv:1805.06576*.
[64] S. Zhang, X. Zhu, Z. Lei, H. Shi, X. Wang, and S. Z. Li, "S3FD: Single shot scale-invariant face detector," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 192–201.
[65] S. Yang, P. Luo, C. C. Loy, and X. Tang, "WIDER FACE: A face detection benchmark," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 5525–5533.
[66] V. Jain and E. Learned-Miller, "FDDB: A benchmark for face detection in unconstrained settings," UMass Amherst, Amherst, MA, USA, Tech. Rep. 6, 2010, vol. 2.

JOONHYUN JEONG received the bachelor's degree from the Department of Computer Science and Engineering, Kyung Hee University, South Korea, in 2019. His research interests include model compression and data augmentation in computer vision task.

INCHEON CHO received the bachelor's degree from the Department of Computer Science and Engineering, Kyung Hee University, South Korea, in 2020, where he is currently pursuing the M.S. degree with the Department of Computer Science and Engineering. His research interests include model compression and pruning for deep neural networks.

EUNSEOP SHIN received the bachelor's degree from the Department of Computer Science and Engineering, Kyung Hee University, South Korea, in 2020, where he is currently pursuing the M.S. degree with the Department of Computer Science and Engineering. His research interests include model compression and weight sharing and clustering for deep neural networks.

SUNG-HO BAE (Member, IEEE) received the B.S. degree from Kyung Hee University, South Korea, in 2011, and the M.S. and Ph.D. degrees from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea, in 2012 and 2016, respectively. From 2016 to 2017, he was a Postdoctoral Associate with the Computer Science and Artificial Intelligence Laboratory (CSAIL), Massachusetts Institute of Technology (MIT), MA, USA. Since 2017, he has been an Assistant Professor with the Department of Computer Science and Engineering, Kyung Hee University. He has been involved in model compression/interpretation for deep neural networks and inverse problems in image processing and computer vision.