Convolution Using Discrete Cosine Transforms for Improving Performance of Convolutional Neural Networks

Izumi Ito
Information and Communications Engineering
Tokyo Institute of Technology
Tokyo, Japan
ito@ict.e.titech.ac.jp

***Abstract*—Convolutional neural networks (CNNs) are widely used in many areas. They feature convolutional layers that focus on spatial local node connections rather than full node connections. This makes networks much more efficient for spatial information. The convolution is a mathematical operation on two functions and can be calculated using the discrete Fourier transform (DFT). Due to the close relation to the DFT, the discrete cosine transforms (DCTs) can be used for the calculation. In this paper, we focus on the convolution using DCTs for improvement of the performance of CNNs. The periodicity and symmetry inherent in the DCTs generate larger output feature maps. The proposed method in simple CNNs is demonstrated and the efficacy of the proposed method is testified using CIFAR-10 dataset.**

***Index Terms*—convolutional neural network, discrete cosine transform, convolution, dropout, fast training**

I. Introduction

Convolutional neural networks (CNN) are widely used in many applications, such as image recognition, image classification, and medical image understanting [1] [2]. CNNs have convolutional layers that calculate the convolution between inputs and filters, where the weights of filters are trained to extract effective features.

Various methods for improving network performance have been proposed. The batch normalization forcibly adjusts the distribution of activations for each layer to suppress overfitting [3]. The dropout avoids the use of a certain percentage of nodes per minibatch to reduce the dependence on specific nodes [4]. The data augmentation (DA) enlarges the number of samples by geometrical transformation, such as flipping, rotation, and scaling [5]. In frameworks, the DA is provided by calling a function in preprocessing for datasets, where manual transformation is not needed.

The convolution is a mathematical operation on two functions. As is well known, the discrete Fourier transform (DFT) is able to calculate the convolution. For the purpose of fast training of CNNs, the FFT, the fast algorithm of DFT, was used for calculating the convolution [6]. Due to the close relation to the DFT, the discrete cosine transforms (DCTs) can calculate convolution under certain conditions. For increasing the number of samples, DCTs were used for flipping DA [7].

In this paper, we proposed the convolution using DCTs rather than spatial convolution in order to obtain extended output feature maps that are almost four times larger than the normal one. By using the extended feature maps together with the dropout, filter weights are learned efficiently, which improves network performance. We demonstrate the proposed method on simple CNNs and show the efficacy using CIFAR-10 dataset [8].

II. Preliminaries

For simplicity, we use one-dimensional expression unless otherwise confusion.

A. Linear convolution and circular convolution

Let $x(n)$ and $h(n)$ be a sequence of length M and a filter of length L, respectively, where $M > L$.

The convolution of $x(n)$ with $h(n)$ is able to be calculated by P-point forward and inverse DFT as

$$x(n) \otimes h(n) = \frac{1}{P} \sum_{k=0}^{P-1} X(k)H(k)W_P^{-nk} \tag{1}$$

where $\otimes$ denotes the convolution operator, $X(k)$ and $H(k)$ represent the DFT coefficients of $x(n)$ and $h(n)$, respectively, and $W_P = \exp(-j2\pi/P)$. Note that $x(n)$ and $h(n)$ are zero-padded to the end of them so that the length becomes P.

Since P is the output lenghth, if $P = M + L - 1$, (1) results in the linear convolution. If $P = M$, it becomes the circular convolution, where the values at more than M points are added to the first part of the linear convolution. That is, the period P of the DFT is important. Below, to represent the length of sequences explicitly, the length is expressed as the subscript. e.g., $x_N(n)$ indicates that the length of sequence $x(n)$ is N.

B. Circular convolution between symmetrically extended sequences

Both DFT and DCT have periodicity. In addition, DCT features symmetry. For the relation between DFT and DCT type-2

ISBN: 978-1-6654-6798-8

(DCT-2: the most popular type of DCTs), the symmetrically extended sequence (SES) of $x_N(n)$ is defined as

$$\hat{x}_{2N}(n) = x_{2N}(n) + x_{2N}(-n-1) \tag{2}$$

where

$$x_{2N}(n) = \begin{cases} x_N(n), & 0 \le n \le N-1 \\ 0, & N \le n \le 2N-1 \end{cases}. \tag{3}$$

The $2N$-point DFT coefficients, $\hat{X}(k)$, of $\hat{x}_{2N}(n)$ are related to the N-point DCT-2 coefficients, $X_C(k)$, of $x_N(n)$ [9], for $k = 0, 1, \ldots, N-1$ as

$$\hat{X}(k) = (1/C_k) X_C(k) W_{2N}^{-k/2} \tag{4}$$

where

$$X_C(k) = 2C_k \sum_{n=0}^{N-1} x_N(n) \cos\left(\frac{\pi(n+1/2)k}{N}\right), \tag{5}$$

$$C_k = \begin{cases} 1/\sqrt{2}, & k = 0 \text{ or } N \\ 1, & \text{otherwise} \end{cases}. \tag{6}$$

From (1), the circular convolution of $\hat{x}_{2N}(n)$ and $\hat{h}_{2N}(n)$ is obtained by $2N$-point inverse DFT as

$$\hat{x}_{2N}(n) \otimes \hat{h}_{2N}(n) = \frac{1}{2N} \sum_{k=0}^{2N-1} \hat{X}(k) \hat{H}(k) W_{2N}^{-nk}. \tag{7}$$

From (4), it is developed as

$$\hat{x}_{2N}(n) \otimes \hat{h}_{2N}(n) = \frac{1}{N} \sum_{k=0}^{N} C_k^2 X_C(k) H_C(k) \cos\left(\frac{\pi nk}{N}\right). \tag{8}$$

Thus, the first N points of the circular convolution of $2N$-point SESs are calculated using N-point DCTs.

III. Convolution using DCTs for the extended feature map

The circular convolution of two SESs contains four linear convolutions. To isolate each linear convolution, zero-values are padded to the head of the sequences in addition to the end of the sequences.

Let z_1 be the number of zero-padding of the head of the sequence $x_M(n)$ and let z_2 be the number of zero-padding of the head of the filter $h_L(n)$:

$$x_N(n) = \begin{cases} x_M(n-z_1), & z_1 \le n \le z_1 + M - 1 \\ 0, & \text{otherwise} \end{cases}, \tag{9}$$

$$h_N(n) = \begin{cases} h_L(n-z_2), & z_2 \le n \le z_2 + L - 1 \\ 0, & \text{otherwise} \end{cases}. \tag{10}$$

The circular convolution between $\hat{x}_{2N}(n)$ and $\hat{h}_{2N}(n)$, which are the extension according to (2) of $x_N(n)$ and $h_N(n)$, respectively, is expressed by the superposition of four linear convolutions as

$$\hat{x}_{2N}(n) \otimes \hat{h}_{2N}(n) = y_{2N}^{(1)}(n) + y_{2N}^{(2)}(n) + y_{2N}^{(3)}(n) + y_{2N}^{(4)}(n) \tag{11}$$

where

$$y_{2N}^{(1)}(n) = \begin{cases} x_M(n) \otimes h_L(n), & R_1 \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

$$y_{2N}^{(2)}(n) = \begin{cases} x_M(-n-1) \otimes h_L(n), & R_2 \\ 0, & \text{otherwise} \end{cases} \tag{13}$$

$$y_{2N}^{(3)}(n) = \begin{cases} x_M(n) \otimes h_L(-n-1), & R_3 \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

$$y_{2N}^{(4)}(n) = \begin{cases} x_M(-n-1) \otimes h_L(-n-1), & R_4 \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

and $R_i = l_i \le n \le l_i + (L + M - 1) - 1$, $i = 1, 2, 3, 4$.

From the condition that the respective ranges ($R_i, i = 1, 2, 3, 4$) do not overlap, when M is an even number, the minimum numbers of z_1, z_2, and N are obtained as

$$z_1 = z_2 + L, \tag{16}$$

$$z_2 = (M-2)/2 + 1, \tag{17}$$

$$N = 2(M + L). \tag{18}$$

Under the conditions in (16), (17), and (18), the order of the outputs of the four linear convolutions is $y_{2N}^{(1)}(n)$, $y_{2N}^{(3)}(n)$, $y_{2N}^{(2)}(n)$, and $y_{2N}^{(4)}(n)$ in the linear convolution of SESs, $\hat{x}_{2N}(n)$ and $\hat{h}_{2N}(n)$. In the circular convolution of the SESs, the values at more than $2N$ points are wrapped around, the order becomes $y_{2N}^{(2)}(n)$, $y_{2N}^{(1)}(n)$, $y_{2N}^{(4)}(n)$, and $y_{2N}^{(3)}(n)$. Since the convolution using DCTs calculate the first N-point of the circular convolution of the SESs, we can obtain $y_{2N}^{(1)}(n)$ and $y_{2N}^{(2)}(n)$ by using DCTs. We use two-dimensional version of whole results of the convolutions using DCTs as the extended feature map.

IV. The Proposed model for CNNs

We propose the model that uses the convolution using DCTs in CNNs to generate the extended feature map for improvement of network performance.

A. The model

We replace the spatial convolution in the first layer of CNNs by the convolution using DCTs. Fig. 1 shows the steps of the first layer in the forward pass. The images of size $M \times M$ in a dataset are zero-padded according to (9), (16), and (18), and the $N \times N$-point DCT is applied to them according to (5) in order to obtain DCT coefficients beforehand. The weights of filters of size $L \times L$ are initially set with random numbers, which are trainable. The filters are zero-padded according to (10), (17), and (18), and the $N \times N$-point DCT is applied to them. The DCT coefficients of a zero-padded image are multiplied by the DCT coefficients of a zero-padded filter element by element. Finally, the inverse transform is applied to the DCT product according to (8) and the bias is added to them.

The extended output feature map consists of four linear convolutions, one between the image and the filter, one between the horizontal flipped image and the filter, one between the vertical flipped image and the filter, and one between the

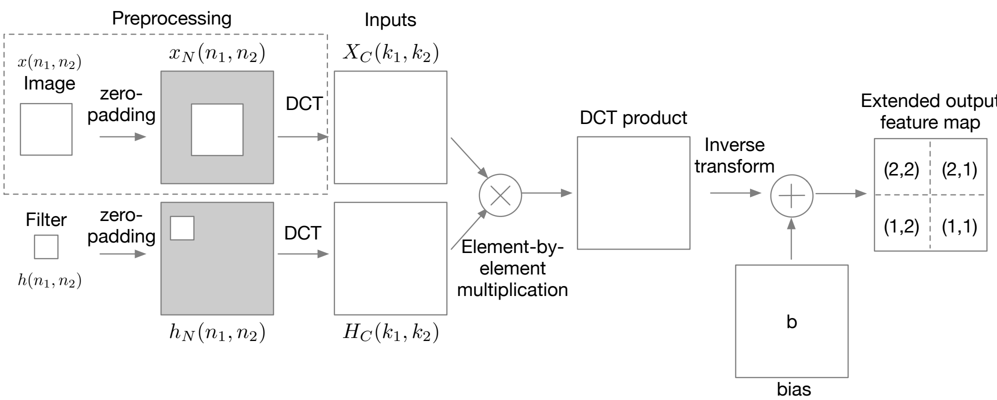

Fig. 1. Steps in the first layer. The numbers, $(1,1)$, $(1,2)$, $(2,1)$, and $(2,2)$ in the extended output feature map correspond to two-dimensional expression of the superscript of linear convolutions in (12) and (13).

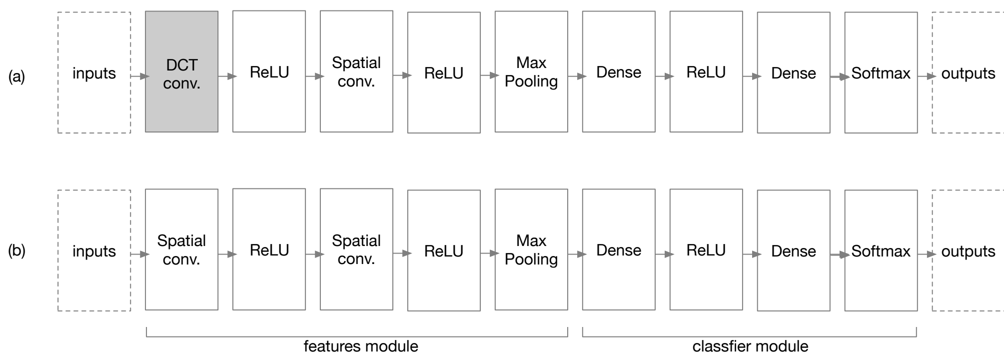

Fig. 2. Model 1 for experiments. (a) proposed. (b) normal. The model consists of two convolutional layers in the features module, two fully connected layers (Dense) followed by the softmax layer in the classifier module.

horizontal and vertical flipped image and the filter. That is, the size of extended output feature maps is four times larger than that of inputs. Since there is a correlation among four linear convolutions, the extended feature map is not efficient as it is. Therefore, using the extended feature map in conjunction with the dropout method reduces the correlation and makes efficient training, which are different from the DA.

B. Computational complexity

Table I summarizes the computational complexity of circular convolution using DCT between an image and a filter that are padded with zero-values so that the size of them is $N \times N$. The number of multiplication (mul.) and addition (add.) operation is based on the fast DCT algorithms [10], [11].

Let B and F be the size of minibatches and number of filters, respectively. The spatial convolution needs $SF(M-L+1)^2L^2$ multiplications, while the convolution using DCT requires $F(N^2 \log_2 N + 2N)$ multiplications for the DCT coefficients of filters, and $BF(2(N^2 \log_2 N + 2N) + N^2)$ multiplications for the DCT product and the inverse transform.

When the filter size is small, the use of DCTs is computationally expensive. However, when the filter size is large, the convolution using DCTs is less computational load than spatial convolution.

TABLE I
COMPUTATIONAL COMPLEXITY OF THE CONVOLUTION USING DCTS OF SIZE $N \times N$.

	ope.	number of operation
forward transform	mul.	$2 \times (N^2 \log_2 N + 2N)$
	add.	$2 \times (3N^2 \log_2 N - 2N^2 + 2N)$
DCT product	mul.	N^2
inverse transform	mul.	$N^2 \log_2 N + 2N$
	add.	$3N^2 \log_2 N - 2N^2 + 2N$

V. EXPERIMENTAL RESULTS

The proposed method is evaluated using CIFAR10 dataset.

A. Experimental setup

We use the CIFAR-10 dataset which consists of 60,000 color images of size 32×32 with three channels in ten classes. In this experiments, 50,000 images were used for training the network, and 10,000 images were utilized for validation only. We compared the Models 1 and 2 of the proposed method to the normal models that use spatial convolution.

TABLE II
TEST ACCURACY OF MODEL I FOR CLASSIFYING 10 CLASSES IN CIFAR-10.

method	acc.
spatial conv.	0.6236
spatial conv. with flipping DA	0.6937
proposed	0.7404

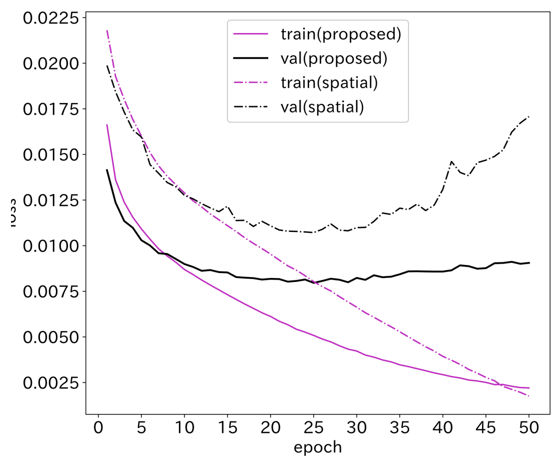

Fig. 3. Learning curve (loss) of Model 1

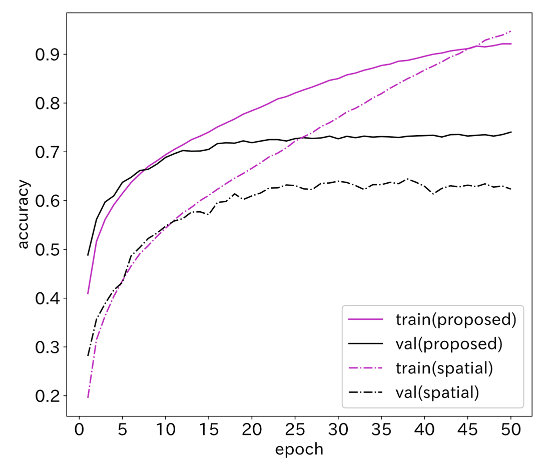

Fig. 4. Learning curve (accuracy) of Model 1

Model 1 consists of two convolutional layers and a 2×2 max-pooling layer in the features module and two fully connected layers followed by the softmax layer in the classifier module, where rectified linear unit (ReLU) activation function was used after the convolutional layers and the fully-connected layers. The convolutional layers have 32 filters of size 3×3. The dropout was applied with the ratio of 0.8 in the features module. Figs. 2(a) and 2(b) show Model 1 of the proposed method and the normal method, respectively. The difference of them is the first layer.

In Model 2, the features module of Model 1 was repeated three times with different number of filters, which is followed by the classifier module of Model 1. The number of filters in the first convolutional layer was 32, otherwise, the number of filters was doubled input in the feature module. The dropout was applied in each features module with the ratio 0.2, 0.3, and 0.4.

In both models, stochastic gradient descent (SGD) was utilized as the optimizer. The cross entropy error was employed as the loss function The initial filter coefficients were set with random numbers. The configurations had minibatches of size 100. Each configuration was trained for 50 epochs with the learning rate 0.01.

B. Experimental results

Figs. 3 and 4 show the learning curves of Model 1 in terms of loss and accuracy, respectively. The solid line shows the proposed method, and the dashed line is the normal method. The lines in black and magenta represent training and validation, respectively. Table II summarizes the accuracy of Model 1, where the proposed method achieved the best accuracy of 0.74040, while the normal method was 0.6236. Since we observed the overfitting in the model of normal method from Fig. 3, we used flipping DA for the normal method. When flipping DA was used in the preprocessing, the accuracy of the normal method increased to 0.6937. In Model 2, the accuracy of the proposed method was 0.8749, while the normal method and the normal method with flipping DA were 0.7672 and 0.7871, respectively. From the results, the efficacy of the proposed method was testified.

VI. CONCLUSIONS

We proposed the convolution using DCTs in the first layer in CNNs to generate the extended feature map for improvement of network performance. Since there is a correlation among the extended feature map, the feature map is used in conjunction with the dropout method to reduce the dependence on specific nodes. We have demonstrated effectiveness of the proposed method in two models of CNNs using the CIFAR10 dataset.

Since the size of the extended feature map is fixed due to the period of DCT, the proposed method can use different sizes of filters in a layer, which enables multi-resolution analysis that was not feasible in a simple layer before. In the future work, we will evaluate the method in some applications and measure the learning time with GPU implementations.

REFERENCES

[1] G. V. A.Dhillon, "Convolutional neural network: a review of models, methodologies and applications to object detection," *Progress in Artificial Intelligence*, pp. 85–112, 2020.

[2] D.R.Sarvamangala and R. V. Kulkarni, "Convolutional neural networks in medical image understanding: a survey," *Evolutionary Intelligence*, vol. 15, pp. 1–22, 2022.

[3] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *arXiv:1502.03167*, 2015.

[4] A. I. N.Srivastava, G.Hinton and R.Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. E96-A, no. 7, pp. 1518–1525, July 2013.

[5] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big data*, pp. 1–48, 2019.

[6] M. Mathieu, M. Henaff, and Y. LeCun, "Fast training of convolutional networks through FFTs," in *International Conference on Learning Representations*. arXiv:1312.5851, 2014.

[7] I. Ito, "Flipping data augmentation of convolutional neural networks using discrete cosine transforms," in *Proceedings of the 29th European Signal Processing Conference (EUSIPCO 2021)*, 2021.

[8] A. Krizhevsky, "Learning multiple layers of features from tiny images," pp. 32–33, 2009. [Online]. Available: https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf

[9] A. V. Oppenheim, R. W. Sshafer, and J. R. Buck, *Discrete-time signal processing*. New Jersey: Prentice Hall, Inc., 1999.

[10] Z. Wang, "Fast algorithms for the discrete w transform and for the discrete fourier transform," vol. 32, no. 4, pp. 803–816, 1984.

[11] ——, "On computing the discrete fourier and cosine transforms," vol. 33, no. 4, pp. 1341–1344, 1985.