BINDCTNET: A SIMPLE BINARY DCT NETWORK FOR IMAGE CLASSIFICATION

Xiangrui Xing[1,2], Xian Yu[1,2], Wenao Ma[1,2], Yue Huang[1,2], Delu Zeng[3], Xinghao Ding[1,2]*

1. Fujian Key Labortary of Sensing and Computing for Smart City, Xiamen, China
2. School of Information Science and Engineering, Xiamen University, Xiamen, China
3. School of Mathematics, South China University of Technology, Guangzhou, China
*dxh@xmu.edu.cn

ABSTRACT

Convolution neural networks play an important role in the image classification tasks. However, it is time consuming to train the network and the cost of memory resources is usually high. In this paper, a simple and effective network named BinDCTNet is presented by using the binary discrete cosine transform(BinDCT) to extract the feature-maps and a hyper-parameter to reduce dimension of the extracted feature. The proposed network has extremely low computing complexity and there is almost no parameters needed to be stored. Experiments are carried out on the hand written digit dataset MNIST and the vehicle logo VLOGO dataset. The results show that the proposed network achieves the state-of-the-art accuracy with fast speed and low memory cost, which makes it applicable on mobile and embedded devices.

Index Terms—*image classification, DCT, convolution neural networks, MNIST, vehicle logo recognition*

1. INTRODUCTION

Image classification is one of the most basic topics in machine learning and image processing. In the condition of limited computing and storage capacity such as on mobile and embedded devices, there is a great need for a lightweight, low latency and accurate network [1][2]. At the same time, using more effective methods to extract image features, compress the model and save computing and memory resources is also an important trend in machine learning [2][3][4][5][6]. MobileNet [2] and SqueezeNet [3] use new convolution methods that can reduce the parameters and compress the model. XNOR-Net [5] binarizes the weight to save the memory and computing cost. Deep Compression [6] uses pruning, weight sharing, weight quantization and Huffman coding to compress AlexNet and VGG-16 by 9 and 13 times, respectively, making it possible to be applied on the mobile and embedded devices. The above-mentioned networks have functions and accuracy of deep neural networks. However, deeper layers will increase memory and computing burden.

Shallow neural networks such as DNN [8] and LeNet-5 [9], are easier to be applied on limited computing and storage capacity platforms compared to the above-mentioned networks. These networks have less test time, but the training process is very time-consuming and the parameters cost a certain memory. PCANet [10] compared to the networks above, extracts filter banks from the input images directly using the PCA algorithm, which reduces a lot of training time. It also uses binary hashing and block-wise histograms to extract the features, so there are almost no parameters needed to be stored at the output layer. After that, DCTNet [11] has been proposed that uses 2D-DCT basis as filter banks in approximation of high ranked eigenvectors of PCA, reducing the pre-training process to obtain the filter banks. However, both PCANet and DCTNet use 2D-convlution to obtain the feature-maps, which consumes considerable amount of computing resources. Besides, the dimensions of extracted features are high, adding a burden to the SVM classifier.

In order to save computing and memory resources, BinDCTNet is proposed in this paper. The proposed network uses 1D discrete cosine transform instead of 2D-convlution to obtain the feature-maps, which significantly saves computing resources. Furthermore, binary discrete cosine transform (BinDCT)[12][13] is applied in this process, so the feature-maps can be computed by fixed-point addition operation basically. At the same time, the hyper-parameter Θ is introduced to reduce the dimensions of extracted features to $1/\Theta$ of the original. The improved network requires fewer computing and memory resources, and can be easily applied on the mobile and embedded devices. The proposed network is validated on the MNIST [9] handwritten dataset and the VLOGO [14][16] vehicle logo dataset, and compared with other shallow networks such as PCANet [10] and DCTNet [11]. The results indicate that the proposed network is at least three times faster than PCANet, with even less parameters and state-of-the-art-accuracy.

This work was supported in part by the National Natural Science Foundation of China under Grants 61571382, 81671766, 61571005, 81671674, U1605252, 61671309 in part by the Guangdong Natural Science Foundation under Grant 2015A030313007, in part by the Fundamental Research Funds for the Central Universities under Grant 20720160075, 20720150169.

2. PROPOSED NETWORK

The proposed network has three stages: stage one and stage two obtain the feature-maps by using BinDCT; while the output stage extracts the features by hashing, histograms and uniformly sampling, after that, the uniformly sampled features are sent to SVM classifier. The proposed network structure was shown in **Fig. 1.**

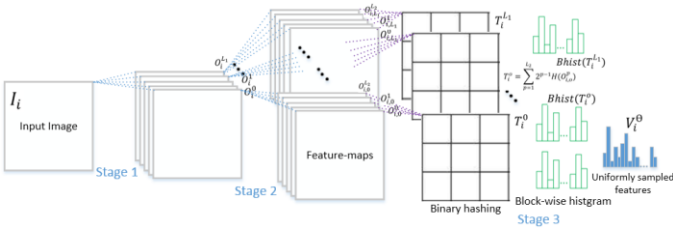

Fig. 1: Proposed network structure.

2.1. Stage One

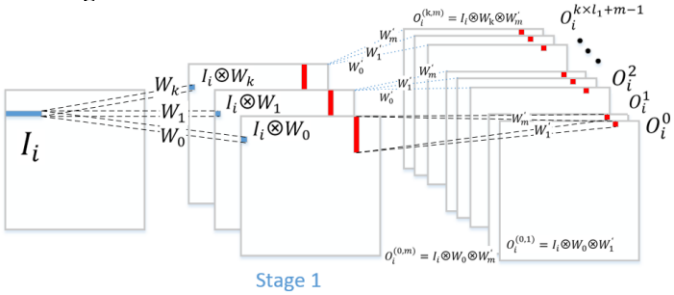

Fig. 2: Obtaining Feature-maps by 1D-DCT.

The boundary of input image is zero-padded so that the output feature-maps have the same size as the input image. Assuming the input images are I_i, the l_1 convolution kernels are set as:

$$W_k = D_N(\mathrm{k},:),\ k = 0,1 \dots l_1 - 1; l_1 < N;, \quad (1)$$

where D_N is N-point DCT transformation matrix [13] and $D_N(\mathrm{k},:)$ represents its k-th row, so $W_k \in R^{1\times \mathrm{N}}$. Similarly:

$$W_m^{'} = [D_N(\mathrm{m},:)]',\ m = 0,1 \dots l_1 - 1; l_1 < N; \quad (2)$$

where $[D_N(m,:)]'$ represents the transpose of the m-th row of D_N, so $W_m^{'} \in R^{N\times 1}$.

The feature-maps can be obtained as follows:

$$O_i^{(k,m)} = I_i \otimes W_k \otimes W_m^{'} \quad (3)$$

where $\otimes$ denotes 2D-convlution. Each time $W_k \in R^{1\times \mathrm{N}}$ convolving around each pixel can be seen as a row 1D-DCT around that pixel, providing one of the DCT coefficients. Similarly, each time $W_m^{'} \in R^{N\times 1}$ convolving around each pixel can be seen as a column 1D-DCT around that pixel, providing one of the DCT coefficients. Thus, the pixel in the feature-maps $O_i^{(k,m)}$ can be seen as k-th row and m-th column 2D-DCT coefficients of an $N \times N$ block in the input image [13][15]. In this way, the feature-maps are obtained through 1D-DCT without 2D-convlution or 2D-DCT. Furthermore, the DCT transformation matrix D_N can be replaced with binary DCT transformation matrix [12][13] D_N^{Bin}, which has only three values: 0, 1 and -1. Hence, this process is simplified and only have fixed-point addition operation. Which was shown in **Fig. 2**.

After this process, each input image can be converted to ${l_1}^2$ feature-maps. When $k = m = 0$, the extracted features are DC coefficients. $O_i^{(0,0)}$ is removed and the extracted feature-maps are rearranged:

$$O_i^q = O_i^{(k,m)}, q = k \times l_1 + m - 1 \quad (4)$$

where $k = 0,1 \dots l_1 - 1; m = 0,1 \dots l_1 - 1; l_1 < N$; and q is from 0 to $({l_1}^2 - 2)$. $L_1 = ({l_1}^2 - 1)$ feature-maps are obtained from each input image through Stage One.

2.2 Stage Two

Stage Two is similar with Stage One. The output of Stage One O_i^q is the input of Stage Two, and almost same procedure is repeated. Set l_2 convolution kernels and obtain the output image:

$$O_{i,\mathrm{q}}^{(k,m)} = O_i^q \otimes W_k \otimes W_m^{'} \quad (5)$$

where $k = 0,1 \dots l_2 - 1; m = 0,1 \dots l_2 - 1; l_2 < N;$ each input O_i^q will be converted to ${l_2}^2$ feature-maps. When $k = m = 0$, $O_{i,\mathrm{q}}^{(0,0)}$ is removed and the extracted feature-maps are rearranged:

$$O_{i,q}^p = O_{i,q}^{(k,m)}, p = k \times l_2 + m - 1 \quad (6)$$

where p is from 0 to $({l_2}^2 - 2)$. Each input O_i^q generates $L_2 = ({l_2}^2 - 1)$ feature-maps in Stage Two.

2.3 Stage Three

After the first two stages, each input image generates $L_1 L_2$ feature-maps $O_{i,q}^p$, $\mathrm{q} = 0,1,\dots,{l_1}^2 - 2$; $\mathrm{p} = 0,1,\dots,{l_2}^2 - 2$. Heaviside step transform is performed on each L_1 feature-maps, so that each pixel corresponds to L_2 binary value. These values are taken as a decimal number and the following maps are obtained:

$$T_i^q = \sum_{p=1}^{L_2} 2^{p-1} H(O_{i,q}^p) \quad (7)$$

where $\mathrm{q} = 0,1,\dots,{l_1}^2 - 2$; $\mathrm{p} = 0,1,\dots,{l_2}^2 - 2$, $H(\cdot)$ means Heaviside step function, and pixel values of T_i^q are in the range $[0, 2^{L_2} - 1]$.

After that, each input image I_i corresponds to L_1 maps $T_i^q, \mathrm{q} = 1,2,\dots,{l_1}^2 - 1$. These L_1 maps are divided into B blocks, and the histogram of each block is calculated to obtain a vector of 2^{L_2} dimensions. These vectors of B blocks are concatenated to obtain $Bhist(T_i^q)$. Thus, each input image I_i's feature vector is:

$$V_i = [Bhist(T_i^1), \dots, Bhist(T_i^{L_1})] \quad (8)$$

The dimensions of this vector are $L_1 \mathrm{B} \times (2^{L_2})$, which are too high. Therefore, the hyper-parameter Θ is introduced to uniformly sample the features V_i, reducing the dimensions to $L_1 \mathrm{B} \times (2^{L_2})/\Theta$, which is $1/\Theta$ of the original dimensions. The uniformly sampled features V_i^Θ will be sent into the linear SVM classifier to achieve image classification.

2.4 Memory and computing cost

Assuming the input image has the size of $m \times n$ pixels. In order to generate the same number and the same size of feature-maps, the filter bank numbers of PCANet [10] and

DCTNet [11] are assumed to be $L_1 = {l_1}^2 - 1$, $L_2 = {l_2}^2 - 1$, and the filter size is $N \times N$. 1D-DCTNet uses N-point DCT transformation matrix D_N in the first two stages. BinDCTNet replaces D_N with binary DCT transformation matrix D_N^{Bin}. Each element in D_N^{Bin} just costs 2 bit memory. BinDCTNet-θ introduce the hyper-parameter θ as described in 2.3

Filter bank memory cost

Changing 2D-filter banks to 1D can reduce the filter bank size. Each element in D_N^{Bin} cost just 2 bit memory, which costs even less memory as shown in **Table 1.**

Memory cost of extracted features and SVM weight vectors

Introducing hyper-parameter θ can reduce the dimensions of the features from $L_1\mathrm{B}(2^{L_2})$ to $L_1\mathrm{B}(2^{L_2})/\theta$, which are integer values. Assuming the images have N_S classifications. BinDCTNet-θ's SVM weight vector has $N_S L_1\mathrm{B}(2^{L_2})/\theta$ dimensions, which is less than other networks' $N_S L_1\mathrm{B}(2^{L_2})$ dimensions.

Computing cost

PCANet and DCTNet use 2D-convlution to obtain feature-maps and 1D-DCTNet just uses 1D-DCT transform, while the BinDCTNet just uses addition operation basically. Besides, all the values of feature-maps are integer values in BinDCTNet, thus the addition operation is fixed-point operation instead of floating-point arithmetic. The computing cost of the linear SVM classifier in the test process is almost reduced to $1/\theta$ of the original after introducing θ. The computing cost was shown in **Table 2** and **Table 3**.

Table 1:Filter bank memory cost

Network	Filter bank size	Value Type
PCANet	$N \times N \times (L_1 + L_2)$	Double
DCTNet	$N \times N \times max(L_1,\ L_2)$	Double
1D-DCTNet	$N \times max(l_1,\ l_2)$	Double
BinDCTNet	$N \times max(l_1,\ l_2)$	2 bit

Table 2: Computing cost of obtaining feature-maps from each image

Network	Multiplication Times	Addition Times
PCANet DCTNet	$mnL_1(L_2 + 1)N^2$	$mnL_1(L_2 + 1)(N^2 - 1)$
1D-DCTNet	$Nmn\left[L_1\left({l_2}^2 + l_2\right) + l_1\right]$	$\binom{N-}{1} mn\left[L_1\left(\begin{matrix}{l_2}^2\\ +l_2\end{matrix}\right) + l_1\right]$
BinDCTNet	0	$\approx 0.75(N - 1)mn\left[L_1\left(\begin{matrix}{l_2}^2\\ +l_2\end{matrix}\right) + l_1\right]$

Table 3: Computing cost of obtaining feature-maps from each image when N=8; $L_1 = L_2 = 8$; $l_1 = l_2 = 3$

Network	Multiplication Times	Addition Times
PCANet DCTNet	4608mn	4536mn
1D-DCTNet	792mn	693mn
BinDCTNet	0	495mn

3. EXPERIMENTS

The experiments are conducted on two datasets, MNIST [9] handwritten dataset and VLOGO [14][16] vehicle logo dataset. MNIST is a well-known dataset, and VLOGO is a dataset of vehicle logos from top ten popular manufacturers. Each sample of VLOGO is resized to 32×32. We set $L_1 = L_2 = 8$ and $l_1 = l_2 = 3$, N=8, so PCANet, DCTNet, 1D-DCTNet and BinDCTNet would have the same size and number of feature-maps.

3.1 Experiments on MNIST dataset

Total 5000 test samples are used to carry out experiments on different network structures. It can be seen from the **Table 4** and **Fig. 3** that the test error rate of different networks is not very different. From **Table 5**, it can be seen that the BinDCTNet can reduce the memory cost dramatically. As for computing complexity, the running time of 2000 training samples and 2000 test samples is tested on Core (TM) i3-3240 CPU with 4G memory and the results are presented in **Table 6** and **Fig. 4**. Although the reduction of the computational complexity is not very obvious on CPU, the proposed network is still at least three times faster than PCANet. PCANet (im2col) [10] constructs a new matrix in the convolution process to reduce some convolution time, but it is memory consuming and have no substantial optimization. Using 1D-DCT and Bin-DCT can also reduce the convolution time. By introducing the hyper-parameter θ, SVM training time and test time can be reduced.

Table 4: Test error rate on MNIST dataset

Training Size	PCANet	DCTNet	1D-DCTNet	Bin DCTNet	BinDCT Net-4	BinDCT Net-16
25	35.57%	36.70%	37.04%	40.00%	37.36%	32.08%
50	18.57%	27.18%	18.33%	20.40%	22.86%	23.38%
100	11.20%	13.00%	11.80%	11.36%	10.96%	16.18%
300	6.00%	5.26%	6.08%	5.46%	6.48%	6.82%
500	3.33%	4.10%	3.78%	4.10%	4.06%	5.40%
1000	2.40%	2.58%	2.68%	2.44%	2.98%	4.04%
2000	2.30%	2.02%	2.04%	2.14%	2.52%	2.94%
5000	1.00%	1.30%	1.32%	1.40%	1.60%	2.54%
10000	0.88%	1.08%	1.24%	1.19%	1.43%	2.00%

Table 5: Memory cost of different networks on MNIST dataset

Network	Memory cost of filter banks	Memory cost of SVM weight vectors	Memory cost of extracted features each image
PCANet	8*8*16 Double	10*73728 Double	73728 Sparse Integer
DCTNet	8*8*8 Double	10*73728 Double	73728 Sparse Integer
1D-DCTNet	8*3 Double	10*73728 Double	73728 Sparse Integer
BinDCTNet-1	8*3 2 bit	10*73728 Double	73728 Sparse Integer
BinDCTNet-4	8*3 2 bit	10*18432 Double	18432 Sparse Integer
BinDCTNet-16	8*3 2 bit	10*4608 Double	4608 Sparse Integer

Table 6: Time consumption of different networks on MNIST dataset (Training Size=2000)

Network Structure	PCANet(2D conv)	PCANet(im2col)	DCT Net	1D-DCTNet	Bin DCTNet	BinDCT Net-4	BinDCT Net-16
Network Training Time（sec）	30.18	25.13	25.71	16.75	15.82	10.89	10.52
Convolution time on training(sec)	12.57	4.15	12.64	3.68	3.24	3.31	3.48
SVM Training Time（sec）	4.46	4.78	5.01	5.08	5.12	1.25	0.38
Test time each Image（ms）	16.29	14.24	16.55	12.29	11.79	6.67	5.32
Convolution time of each sample(ms)	6.29	2.08	6.32	1.84	1.62	1.66	1.74
SVM Predict time of each sample(ms)	3.65	3.92	3.76	3.81	3.86	1.06	0.21

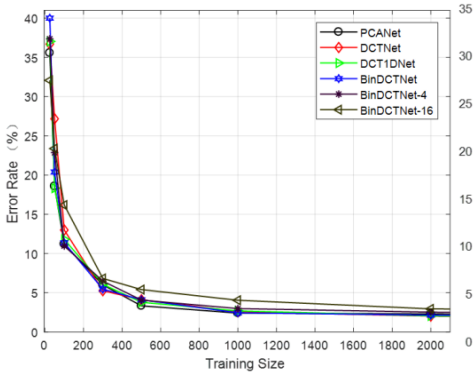

Fig. 3: Test error rate on MNIST dataset

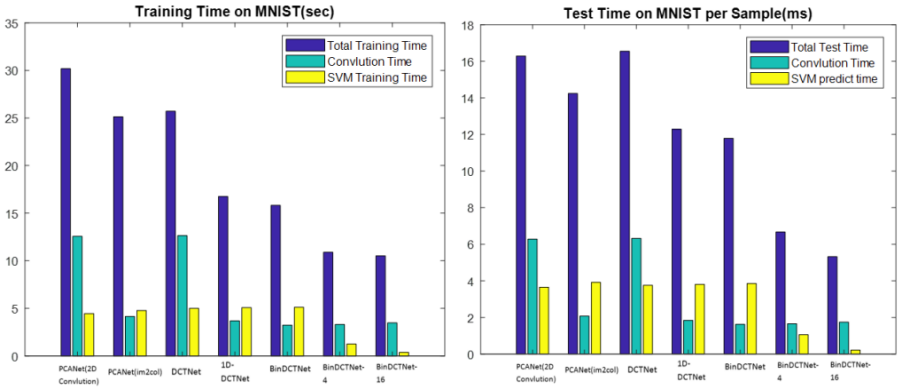

Fig. 4: Training time and test time on MNIST dataset (Training Size=2000).

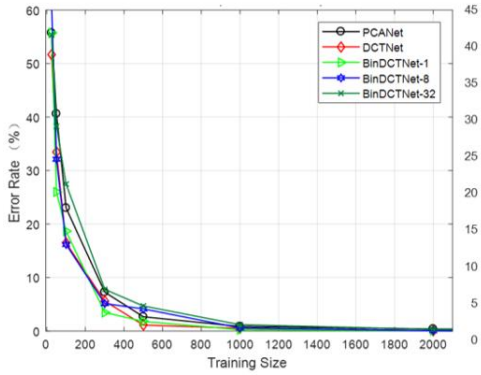

Fig. 5: Test error rate on VLOGO dataset

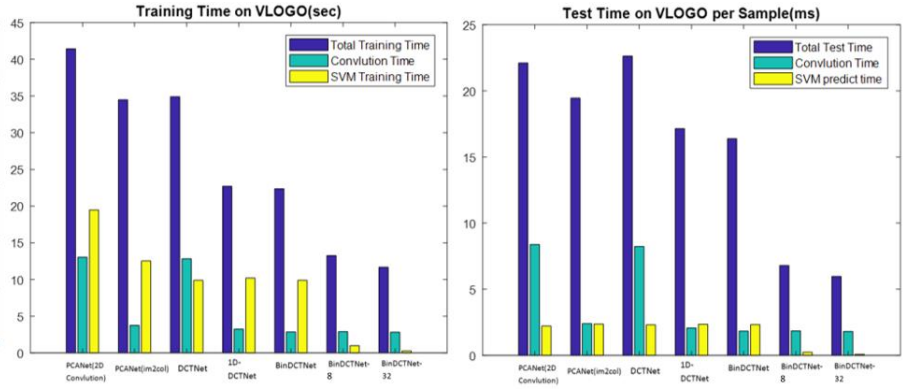

Fig. 6: Training time and test time on VLOGO dataset(Training Size=2000).

3.2 Experiments on VLOGO dataset

The accuracy of VLOGO dataset is better than MNIST dataset for almost all networks. When the training size is over 2000, the test error rate of all networks is almost zero by **Table 7** and **Fig. 5**. Memory cost and time consumption are shown in **Table 8, Table 9,** and **Fig 6.** The proposed network has very low computing and memory cost together with state-of-the-art accuracy.

Table 7: Test error rate on VLOGO dataset

Training Size	PCANet	DCTNet	1D-DCTNet	Bin DCTNet	BinDCT Net-8	BinDCT Net-32
25	55.73%	51.67%	61.60%	55.73%	61.00%	55.33%
50	40.60%	33.47%	33.40%	26.00%	32.13%	38.27%
100	23.00%	16.47%	20.40%	18.67%	16.20%	27.47%
300	7.27%	5.60%	5.47%	3.47%	5.13%	7.80%
500	2.67%	1.13%	1.13%	1.80%	4.13%	4.67%
1000	0.87%	0.53%	0.53%	0.27%	0.60%	1.20%
2000	0.40%	0.07%	0.33%	0.07%	0.07%	0.33%
5000	0	0	0.07%	0.07%	0.07%	0.07%

Table 8: Time consumption of different networks on VLOGO dataset (Training Size=2000)

Network Structure	PCANet(2D conv)	PCANet (im2col)	DCT Net	1D-DCTNet	Bin DCTNet	BinDCT Net-8	BinDCT Net-32
Network Training Time（sec）	41.43	34.48	34.90	22.69	22.36	13.25	11.66
Convolution time on training(sec)	13.05	3.73	12.81	3.22	2.84	2.88	2.80
SVM Training Time（sec）	19.47	12.51	9.87	10.17	9.87	0.98	0.25
Test time each image（ms）	22.11	19.45	22.63	17.15	16.39	6.79	5.97
Convolution time each image(ms)	8.39	2.40	8.24	2.07	1.83	1.85	1.80
SVM Predict time each image(ms)	2.21	2.37	2.30	2.35	2.33	0.23	0.08

Table 9: Memory cost of different networks on VLOGO dataset

Network	Memory cost of filter banks	Memory cost of SVM Weight Vectors	Memory cost of extracted features each image
PCANet	8*8*16 Double	10*100352 Double	100352 Sparse Integer
DCTNet	8*8*8 Double	10*100352 Double	100352 Sparse Integer
1D-DCTNet	8*3 Double	10*100352 Double	100352 Sparse Integer
BinDCTNet-1	8*3 2 bit	10*100352 Double	100352 Sparse Integer
BinDCTNet-4	8*3 2 bit	10*25088 Double	25088 Sparse Integer
BinDCTNet-16	8*3 2 bit	10*6272 Double	6272 Sparse Integer

4. CONCLUTION

In this paper, a new network named BinDCTNet based on PCANet is proposed to extract the feature-maps through Bin-DCT, where the network parameters is compressed by sampling the extracted features. The proposed network is extensively tested on MNIST and VLOGO datasets. The results have proved that the proposed network can save substantial amount of computing and memory resources. At the same time, the proposed network is very easy to be applied on the mobile and embedded devices.

5. REFERENCES

[1] Lane N D, Bhattacharya S, Georgiev P, et al. "An Early Resource Characterization of Deep Learning on Wearables, Smartphones and Internet-of-Things Devices" International Workshop on Internet of Things Towards Applications. ACM, 2015:7-12.

[2] Howard A G, Zhu M, Chen B, et al. "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. " 2017.

[3] Iandola F N, Han S, Moskewicz M W, et al. "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size." 2016.

[4] Bruna J, Mallat S. "Invariant Scattering Convolution Networks. " IEEE Transactions on Pattern Analysis & Machine Intelligence, 2013, 35(8):1872-1886.

[5] Rastegari M, Ordonez V, Redmon J, et al. "XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks. " 2016:525-542.

[6] Han S, Mao H, Dally W J. "Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding." Fiber, 2015, 56(4):3--7.

[7] Lane N D, Bhattacharya S, Mathur A, et al. "Squeezing Deep Learning into Mobile and Embedded Devices." IEEE Pervasive Computing, 2017, 16(3):82-88.

[8] Zhang J, Zheng Y, Qi D, et al. "DNN-based prediction model for spatio-temporal data." ACM Sigspatial International Conference on Advances in Geographic Information Systems. ACM, 2016:92.

[9] Lecun Y, Bottou L, Bengio Y, et al. "Gradient-based learning applied to document recognition." Proceedings of the IEEE, 1998, 86(11):2278-2324.

[10] Chan T H, Jia K, Gao S, et al. "PCANet: A Simple Deep Learning Baseline for Image Classification? " IEEE Transactions on Image Processing, 2015, 24(12):5017.

[11] Ng C J, Teoh A B J. "DCTNet: A simple learning-free approach for face recognition." Asia-Pacific Signal and Information Processing Association Summit and Conference. IEEE, 2015:761-768.

[12] Dang P P, Chau P M, Nguyen T Q, et al. "BinDCT and Its Efficient VLSI Architectures for Real-Time Embedded Applications." Journal of Imaging Science & Technology, 2005, 49(2):124-137(14).

[13] Cintra R J, Bayer F M, Tablada C J. "Low-complexity 8-point DCT approximations based on integer functions." Signal Processing, 2014, 99(99):201-214.

[14] Huang Y, Wu R, Sun Y, et al. "Vehicle Logo Recognition System Based on Convolutional Neural Networks With a Pretraining Strategy." IEEE Transactions on Intelligent Transportation Systems, 2015, 16(4):1951-1960.

[15] Bouguezel S, Ahmad M O, Swamy M N S. "A fast 8×8 transform for image compression." International Conference on Microelectronics. IEEE, 2010:74-77.

[16] http://smartdsp.xmu.edu.cn/VehicleLogoRecognition.html