Supplementary Material for
Quantized Convolutional Neural Networks for Mobile Devices

Jiaxiang Wu, Cong Leng, Yuhang Wang, Qinghao Hu, Jian Cheng

In this supplementary material, we include additional experimental results of our quantized CNN models. Also, the detailed optimization process with error correction for the convolutional layer is presented.

1 Additional Experimental Results

In the submission, we report the performance after quantizing all the convolutional layers in AlexNet, and quantizing all the full-connected layers in CaffeNet. Here, we present experimental results for some other settings.

1.1 Quantizing Convolutional Layers in CaffeNet

We quantize all the convolutional layers in CaffeNet, and the results are as demonstrated in Table 1. Furthermore, we fine-tune the quantized CNN model learned with error correction ($C'_s = 8, K = 128$), and the increase of top-1/5 error rates are 1.15% and 0.75%, compared to the original CaffeNet.

Table 1: Comparison on the speed-up rates and the increase of top-1/5 error rates for accelerating all the convolutional layers in CaffeNet, without fine-tuning.

Method	Para.	Speed-up	Top-1 Err. ↑	Top-5 Err. ↑
Q-CNN	4/64	3.32×	18.69%	16.73%
	6/64	4.32×	32.84%	33.55%
	6/128	3.71×	20.08%	18.31%
	8/128	4.27×	35.48%	37.82%
Q-CNN (EC)	4/64	3.32×	1.22%	0.97%
	6/64	4.32×	2.44%	1.83%
	6/128	3.71×	1.57%	1.12%
	8/128	4.27×	2.30%	1.71%

1.2 Quantizing Convolutional Layers in CNN-S

We quantize all the convolutional layers in CNN-S, and the results are as demonstrated in Table 2. Furthermore, we fine-tune the quantized CNN model learned with error correction ($C'_s = 8, K = 128$), and the increase of top-1/5 error rates are 1.24% and 0.63%, compared to the original CNN-S.

1.3 Quantizing Fully-connected Layers in AlexNet

We quantize all the fully-connected layers in AlexNet, and the results are as demonstrated in Table 3.

1.4 Quantizing Fully-connected Layers in CNN-S

We quantize all the fully-connected layers in CNN-S, and the results are as demonstrated in Table 4.

Table 2: Comparison on the speed-up rates and the increase of top-1/5 error rates for accelerating all the convolutional layers in CNN-S, without fine-tuning.

Method	Para.	Speed-up	Top-1 Err. ↑	Top-5 Err. ↑
Q-CNN	4/64	3.69×	19.87%	16.77%
	6/64	5.17×	45.74%	48.67%
	6/128	4.78×	27.86%	25.09%
	8/128	5.92×	46.18%	50.26%
Q-CNN (EC)	4/64	3.69×	1.60%	0.92%
	6/64	5.17×	3.49%	2.32%
	6/128	4.78×	2.07%	1.32%
	8/128	5.92×	3.42%	2.17%

Table 3: Comparison on the compression rates and the increase of top-1/5 error rates for compressing all the fully-connected layers in AlexNet, without fine-tuning.

Method	Para.	Compression	Top-1 Err. ↑	Top-5 Err. ↑
Q-CNN	2/16	13.96×	0.25%	0.27%
	3/16	19.14×	0.77%	0.64%
	3/32	15.25×	0.54%	0.33%
	4/32	18.71×	0.71%	0.69%
Q-CNN (EC)	2/16	13.96×	0.14%	0.20%
	3/16	19.14×	0.40%	0.22%
	3/32	15.25×	0.40%	0.21%
	4/32	18.71×	0.46%	0.38%

Table 4: Comparison on the compression rates and the increase of top-1/5 error rates for compressing all the fully-connected layers in CNN-S, without fine-tuning.

Method	Para.	Compression	Top-1 Err. ↑	Top-5 Err. ↑
Q-CNN	2/16	14.37×	0.22%	0.07%
	3/16	20.15×	0.45%	0.22%
	3/32	15.79×	0.21%	0.11%
	4/32	19.66×	0.35%	0.27%
Q-CNN (EC)	2/16	14.37×	0.36%	0.14%
	3/16	20.15×	0.43%	0.24%
	3/32	15.79×	0.29%	0.11%
	4/32	19.66×	0.56%	0.27%

2 Optimization with Error Correction for the Convolutional Layer

Assume we have N images to learn the quantization of a convolutional layer. For image I_n, we denote its input feature maps as $S_n \in \mathbb{R}^{d_s \times d_s \times C_s}$ and response feature maps as $T_n \in \mathbb{R}^{d_t \times d_t \times C_t}$, where d_s, d_t are the spatial sizes and C_s, C_t are the number of feature map channels. We use p_s and p_t to denote the spatial location in the input and response feature maps. The spatial location in the convolutional kernels is denoted as p_k.

To learn quantization with error correction for the convolutional layer, we attempt to optimize:

$$\min_{\{D^{(m)}\},\{B_{p_k}^{(m)}\}} \sum_{n,p_t} \left\| \sum_{(p_k,p_s)} \sum_{m} (D^{(m)} B_{p_k}^{(m)})^T S_{n,p_s}^{(m)} - T_{n,p_t} \right\|_F^2 \tag{1}$$

where D^m is the m-th sub-codebook, and $B_{p_k}^{(m)}$ is the corresponding sub-codeword assignment indicator for the convolutional kernels at spatial location p_k.

Similar to the fully-connected layer, we adopt a block coordinate descent approach to solve this optimization problem. For the m-th subspace, we firstly define its residual feature map as:

$$R_{n,p_t}^{(m)} = T_{n,p_t} - \sum_{(p_k,p_s)} \sum_{m' \neq m} (D^{(m')} B_{p_k}^{(m')})^T S_{n,p_s}^{(m')} \tag{2}$$

and then the optimization in the m-th subspace can be re-formulated as:

$$\min_{D^{(m)},\{B_{p_k}^{(m)}\}} \sum_{n,p_t} \left\| \sum_{(p_k,p_s)} (D^{(m)} B_{p_k}^{(m)})^T S_{n,p_s}^{(m)} - R_{n,p_t} \right\|_F^2 \tag{3}$$

Update $D^{(m)}$. With the assignment indicator $\{B_{p_k}^{(m)}\}$ fixed, we let:

$$L_{k,p_k} = \{c_t | B_{p_k}^{(m)}(k, c_t) = 1\} \tag{4}$$

We greedily update each sub-codeword in the m-th sub-codebook $D^{(m)}$ in a sequential style. For the k-th sub-codeword, we compute the corresponding residual feature map as:

$$Q_{n,p_t,k}^{(m)}(c_t) = R_{n,p_t}^{(m)}(c_t) - \sum_{(p_k,p_s)} \sum_{k' \neq k} \sum_{c_t \in L_{k',p_k}} D_{k'}^{(m)^T} S_{n,p_s}^{(m)} \tag{5}$$

and then we can alternatively optimize:

$$\min_{D_k^{(m)}} \sum_{n,p_t} \left\| \sum_{(p_k,p_s)} \sum_{c_t \in L_{k,p_k}} D_k^{(m)^T} S_{n,p_s}^{(m)} - Q_{n,p_t,k}^{(m)}(c_t) \right\|_F^2 \tag{6}$$

which can be transformed into a least square problem. By solving it, we can update the k-th sub-codeword.

Update $\{B_{p_k}^{(m)}\}$. We greedily update the sub-codeword assignment at each spatial location in the convolutional kernels in a sequential style. For the spatial location p_k, we compute the corresponding residual feature map as:

$$P_{n,p_t,p_k}^{(m)} = R_{n,p_t}^{(m)} - \sum_{\substack{(p'_k,p'_s) \\ p_k \neq p_k}} (D^{(m)} B_{p'_k}^{(m)})^T S_{n,p'_s}^{(m)} \tag{7}$$

and then the optimization can be re-written as:

$$\min_{B_{p_k}^{(m)}} \sum_{n,p_t} \left\| (D^{(m)} B_{p_k}^{(m)})^T S_{n,p_s}^{(m)} - P_{n,p_t,p_k} \right\|_F^2 \tag{8}$$

Since $B_{p_k}^{(m)} \in \{0,1\}^K$ is an indicator vector (only one non-zero entry), we can exhaustively try all sub-codewords and select the optimal one that minimize the objective function.