Optimizing Information Theory Based Bitwise Bottlenecks for Efficient Mixed-Precision Activation Quantization

Xichuan Zhou[1 2], Kui Liu[1], Cong Shi[1], Haijun Liu[1], Ji Liu[3]
[1]School of Microelectronics and Communication Engineering, Chongqing University
[2]Key Laboratory of Dependable Service Computing in Cyber-Physical-Society, Ministry of Education
[3]AI platform, Kwai Inc.
zxc@cqu.edu.cn

Abstract

Recent researches on information theory shed new light on the continuous attempts to open the black box of neural signal encoding. Inspired by the problem of lossy signal compression for wireless communication, this paper presents a Bitwise Bottleneck approach for quantizing and encoding neural network activations. Based on the rate-distortion theory, the Bitwise Bottleneck attempts to determine the most significant bits in activation representation by assigning and approximating the sparse coefficients associated with different bits. Given the constraint of a limited average code rate, the bottleneck minimizes the distortion for optimal activation quantization in a flexible layer-by-layer manner. Experiments over ImageNet and other datasets show that, by minimizing the quantization distortion of each layer, the neural network with bottlenecks achieves the state-of-the-art accuracy with low-precision activation. Meanwhile, by reducing the code rate, the proposed method can improve the memory and computational efficiency by over six times compared with the deep neural network with standard single-precision representation. The source code is available on GitHub: https://github.com/CQUlearningsystemgroup/BitwiseBottleneck.

Introduction

Research on activation compression for efficient neural computing is an emerging popular topic, which reveals great potential for a variety of edge-computing and computer vision applications (Chen et al. 2015; Wu et al. 2017; McCool, Perez, and Upcroft 2017; Xu et al. 2017). Both stochastic and deterministic approaches were proposed for neural network activation quantization (Gupta et al. 2015; Courbariaux, Bengio, and David 2015; Zhou et al. 2016; Wu et al. 2016), which adopted various quantization functions to reduce the precision of activation representation while training the neural network. Though the loss of activation precision causes notable loss of classification accuracy, yet by reducing the number of bits required, the deep neural networks with quantized activation could improve both memory and time efficiency by orders of magnitude compared with the standard floating-point implementation (Courbariaux, Bengio, and David 2015; Kim and Smaragdis 2016; Courbariaux et al. 2016; Rastegari et al. 2016).

One challenge of the researches for activation quantization is the lack of theoretical ground (Sze et al. 2017). The performance of existing approaches relies on their choices of quantization strategy, and no minimal loss of representation precision can be determined given the average code rate. Recently, the concept of information theory based bottleneck is attracting attention for its potential to bring a better understanding of the deep learning optimization process (Tishby and Zaslavsky 2015; Shwartz-Ziv and Tishby 2017). The key idea of these approaches is to maintain the most valuable information of a signal $\mathbf{X}$ using a short code of $\hat{\mathbf{X}}$. In the context of deep neural network, $\hat{\mathbf{X}}$ is the signal of quantized activation, and we formalize this problem as that of finding a shortcode for $\hat{\mathbf{X}}$ that preserves the maximum information about $\mathbf{X}$. That is, we minimize the loss of the information caused by signal quantization through a 'bottleneck' in the deep neural network.

Technically, the proposed Bitwise Bottleneck approach is based on the *rate-distortion theory* (Berger 2003). As a lossy data compression operation, we attempt to determine the most significant bits in the activation by minimizing the quantization distortion. To achieve this, the Bitwise Bottleneck approach directly minimizes the distortion of quantization given the constraint of the maximum code rate of the compressed activation. Specifically, the bitwise bottleneck optimization is formulated as a sparse convex optimization problem, which attempts to minimize the distortion given constrained code rate. Since the nonsignificant bits generally have near-zero coefficients, the activation can be optimally compressed in a bitwise way.

The contributions of this paper are threefold. First, this paper presents an information theory based method for *bitwise activation quantization* and compression. Specifically, different from recent researches, the proposed method is the first attempt to optimize *bitwise* bottleneck for DNN activation quantization. Second, the code rates of different bottlenecks can be tuned adaptably by a single hyperparameter of the threshold of peak-signal-to-noise-ratio (PSNR) loss, allowing the proposed method to flexibly make a trade-off between efficiency and accuracy for different applications. Finally, the bitwise bottleneck minimizes the loss of information caused by activation quantization; therefore, the proposed method suffers almost no loss of classification accuracy while obtaining over six times of memory and com-

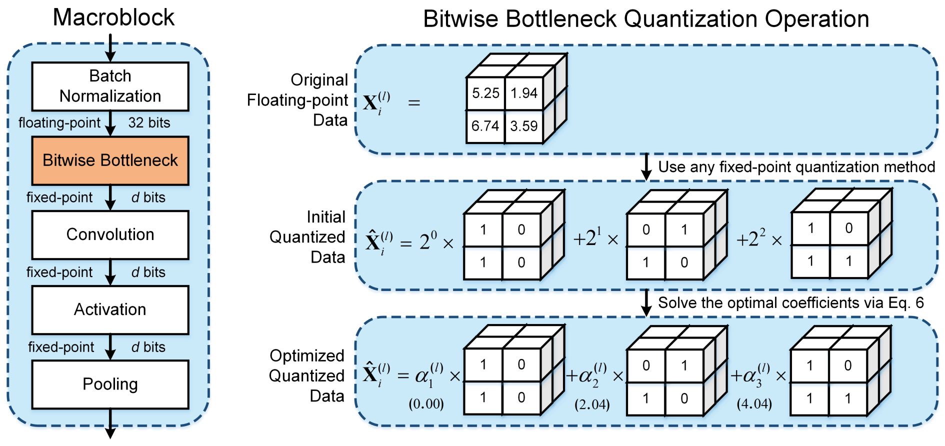


Figure 1: Illustration of the Bitwise Bottleneck approach. The bitwise bottleneck operation can be inserted in the convolutional neural network for activation quantization. The idea of the proposed method is to substitute the constant levels of the standard quantization, which are the power of two in this example, for variable coefficients. By exploiting the sparsity of the bitwise coefficients, the bottleneck could reduce the number of bits required for activation representation.

putational efficiency improvement.

Related Work

Emerging research on neural encoding based on the information theory is an interesting topic which attempts to answer the basic questions about the design principle of deep networks such as the optimal architecture and the optimal quantization scheme. Recently proposed deep neural networks with information bottlenecks usually had an encoder-decoder architecture for *feature-wise* compression, which treated the neural network as a trade-off between compression and prediction (Tishby and Zaslavsky 2015; Dai, Zhu, and Wipf 2018). As far as we know, this work is the first attempt for bitwise compression addressing the challenge of minimizing the code rate while optimally preserving activation integrity.

Despite information theory motivated researches, the research on deterministic or stochastic model-based activation quantization is also attracting more attention lately. Lee et al. quantized neural networks according to channel-level distribution (Lee et al. 2018). Zhao et al. used outlier channel splitting to eliminate the effect of outliers caused by quantization (Zhao et al. 2019a). Minimum mean squared error (Kravchik et al. 2019) and complementary approach (Banner, Nahshan, and Soudry 2019) were both used for reducing quantization error. Meanwhile, the research on data-driven activation quantization was also booming. Qiu et al. presented a Fisher vector method to encode activation with a fixed-point number using a deep generative model (Qiu, Yao, and Mei 2017). Jacob et al. quantized both the activations and weights as 8-bit integers using an iterative approach (Jacob et al. 2018). Li et al. proposed an entropy-based method for interpretable quantization (Li et al. 2019). Compared with information theory based methods, these approaches were generally based on different assumptions on the quantization function, many researchers used non-derivative quantization operation (Qiu, Yao, and Mei 2017; Jacob et al. 2018; Li et al. 2019), which might lead to uninterpretable and suboptimal results.

The Bitwise Bottleneck Method

Recently, there has been growing interest in applying information theory for deep neural network activation compression. By interpreting the deep neural network as a lossy data compression approach, the black box of the deep neural network may be opened and its performance can be optimized by the tool of rate-distortion theory, which is widely applied in the area of telecommunications (Shwartz-Ziv and Tishby 2017).

Rate-distortion Theory

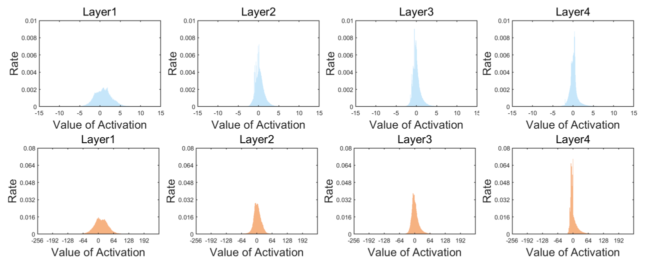
Assume the random variable $\mathbf{X}^{(l)} \in \mathbb{R}^{P \cdot Q \cdot K}$ is the floating-point neural network activation tensor associated with the lth layer, where P, Q, K are respectively the height, width and number of feature maps at the lth layer. The common quantization function $\mathcal{Q}(\cdot)$ can be written as

$$\hat{\mathbf{X}}^{(l)} = \mathcal{Q}(\mathbf{X}^{(l)}) \tag{1}$$

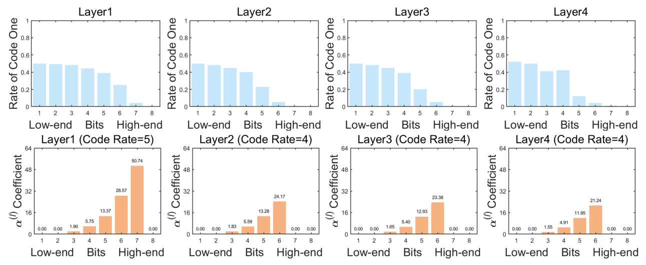
where $\hat{\mathbf{X}}^{(l)}$ is the fixed-point representation of the floating-point activation. The goal of lossy data compression is to achieve minimal distortion given the constraint of the maximum code rate. Assume $g(\cdot)$ is the function that indicates the number of bits of the given data. According to rate-distortion theory, the typical lossy data compression approach attempts to minimize the distortion function $d(\cdot)$ given the maximum number of bits η as

$$\min_{\mathcal{Q}(\cdot)} \mathbf{E}(d(\mathbf{X}^{(l)}, \mathcal{Q}(\mathbf{X}^{(l)}))) \quad \text{s.t.}\ \mathbf{E}(g(\hat{\mathbf{X}}^{(l)})) \leq \eta \tag{2}$$

In practice, the quantization function $\mathcal{Q}(\cdot)$ is nondifferentiable and nonconvex due to its integer output, which makes



(a) Activation distribution



(b) Bitwise rate of code one and coefficients of $\boldsymbol{\alpha}^{(l)}$

Figure 2: Feature-wise vs. bitwise activation sparsity. (a) The distributions of real-valued (upper) and quantized (lower) activations of the first four layers of ResNet50 over CIFAR10. (b) The average rate of code one of each bit of the quantized activation (upper), and the estimated coefficients $\boldsymbol{\alpha}^{(l)}$ (lower). It seems that the optimal code rates of different layers depend on the level of activation sparsity, and the Bitwise Bottleneck approach can adaptively remove the near-zero high-end bits and the less-informative low-end bits of the activation representation.

Eq. 2 difficult to solve. Different from typical rounding-based quantization approaches that settle for suboptimal solutions, this paper attempts to find the optimal solution by reformulating Eq. 2 as a sparse coding problem.

Activation Quantization via Bitwise Bottleneck Encoding

In practice, we could approximate the solution of Eq. 2 using a training sample set. Assume $\mathbf{X}_i^{(l)} \in \mathbb{R}^{P \cdot Q \cdot K}$ is the floating-point activation tensor associated with the *i*th sample output by *l*th layer. Fig. 1 illustrates the Bitwise Bottleneck approach. Formally speaking, assume $\hat{\mathbf{X}}_i^{(l)}$ is the D-bit fixed-point approximation of $\mathbf{X}_i^{(l)}$ as defined in Eq. 3, where $\hat{\mathbf{X}}_{ij}^{(l)} \in \{0,1\}^{P \cdot Q \cdot K}$ is a three-dimensional binary tensor representing the *j*th bit of $\hat{\mathbf{X}}_i^{(l)}$.

$$\hat{\mathbf{X}}_i^{(l)} = \mathcal{Q}(\mathbf{X}_i^{(l)}) = 2^0\hat{\mathbf{X}}_{i1}^{(l)} + 2^1\hat{\mathbf{X}}_{i2}^{(l)} + ... + 2^{D-1}\hat{\mathbf{X}}_{iD}^{(l)} \quad (3)$$

where each bitwise data matrix is assigned a constant coefficient of $2^0, ..., 2^{D-1}$. In practice, this bitwise activation representation allows the computation of fixed-point data to be implemented in a bitwise way. The computational and memory complexity are proportional to the number of bits of different representations.

Technically, the binary quantization of Eq. 3 inherently assumes that each of the D bits in the activation representation is needed, although different bits contain different but *fixed* amounts of information. By removing this assumption, the proposed method substitutes the fixed coefficient for a variable $\boldsymbol{\alpha}^{(l)} \in \mathbb{R}^D$ as

$$\hat{\mathbf{X}}_i^{(l)} = \mathcal{Q}(\mathbf{X}_i^{(l)}) = \alpha_1^{(l)}\hat{\mathbf{X}}_{i1}^{(l)} + \alpha_2^{(l)}\hat{\mathbf{X}}_{i2}^{(l)} + ... + \alpha_D^{(l)}\hat{\mathbf{X}}_{iD}^{(l)} \quad (4)$$

The Bitwise Bottleneck approach treats the design of neural networks as a trade-off between compression and prediction, which assumes that the bitwise bottlenecks can exploit the sparsity in the activation representation so that one can reduce the precision of activation representation without hurting classification accuracy. Fig. 2 illustrates the activation distributions of the first 4 layers of ResNet50 (He et al. 2016) over the CIFAR10 dataset. It seems that activations of different layers are sparse, but the sparsity varies from layer to layer. Fig. 2(b) illustrates the average rate of code one in each bit of the activation representation (upper graph). Despite the change of feature-wise sparsity as shown in Fig. 2(a), the bitwise sparsity over the high-end bits of activation representation is easier to detect. Therefore, one might be able to estimate the optimal and sparse coefficients $\boldsymbol{\alpha}^{(l)}$ associated with the most significant bits.

According to the rate-distortion theory (Eq. 2), the Bitwise Bottleneck attempts to find the optimal quantization scheme by minimizing the standard squared distortion rate over N training samples given that less than η bitwise coefficients are nonzero,

$$\arg\min_{\boldsymbol{\alpha}^{(l)}} \sum_{i=1}^{N} \|\mathbf{X}_i^{(l)} - \sum_{j=1}^{D} \alpha_j^{(l)}\hat{\mathbf{X}}_{ij}^{(l)}\|_2^2 \quad \text{s.t. } \|\boldsymbol{\alpha}^{(l)}\|_0 \le \eta \quad (5)$$

where $\hat{\mathbf{X}}_{ij}^{(l)}$ calculated by initial quantization operation are usually known as the quantization *codebook*. In practice, different initial quantization operations can be applied, and Eq. 3 is a simple example of the rounding quantization approach. It is worth noting that since the number of nonzero coefficients in $\boldsymbol{\alpha}^{(l)}$ equals the number of bits in the fixed-point representation, the constraint function of Eq. 5 actually limits the maximum number of bits in the quantized representation as required by the rate-distortion theory. Recent research shows that Eq. 5 is equivalent to the following L1-norm-based problem when fulfilling the sparsity requirement, which leads to a sparse solution (Baraniuk 2007).

$$\arg\min_{\boldsymbol{\alpha}^{(l)}} \sum_{i=1}^{N} \|\mathbf{X}_i^{(l)} - \sum_{j=1}^{D} \alpha_j^{(l)}\hat{\mathbf{X}}_{ij}^{(l)}\|_2^2 \quad \text{s.t. } \|\boldsymbol{\alpha}^{(l)}\|_1 \le \eta \quad (6)$$

The bitwise bottleneck operation solves Eq. 6 to determine the sparse significant bits and leads to the minimal distortion. In practice, one usually calculates Eq. 6 by solving its Lagrangian form as

$$\arg\min_{\boldsymbol{\alpha}^{(l)}} \sum_{i=1}^{N} \|\mathbf{X}_i^{(l)} - \sum_{j=1}^{D} \alpha_j^{(l)}\hat{\mathbf{X}}_{ij}^{(l)}\|_2^2 + \lambda\|\boldsymbol{\alpha}^{(l)}\|_1$$

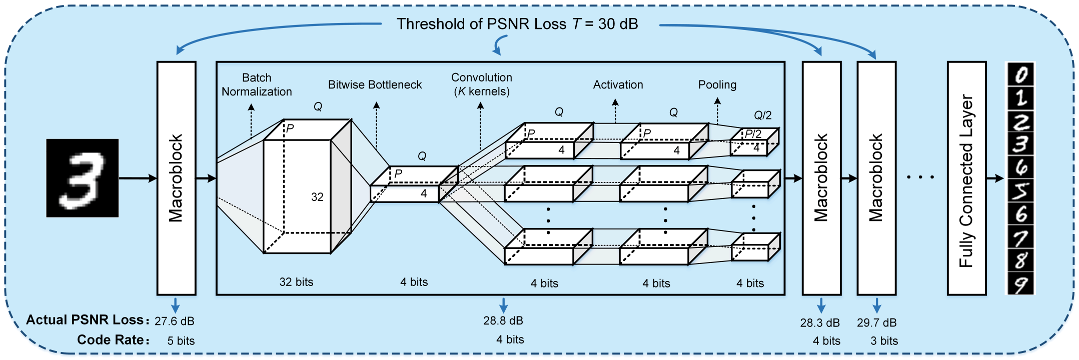


Figure 3: Deep neural network with bitwise bottleneck layers. The second macroblock is extended and shown in detail. By setting a single hyperparameter of the threshold of PSNR loss, the bitwise bottlenecks in different macroblocks can be trained to quantize the normalized activations flexibly with different optimal code rates.

where λ is the hyperparameter for controlling the trade-off between the optimized error rate and the code rate. Eq. 6 generally leads to a sparse solution of the coefficients $\boldsymbol{\alpha}^{(l)}$, so the activation bits associated with zero coefficients are removed during the inference stage. To avoid expensive computations caused by floating-point, we quantize the coefficient vector $\boldsymbol{\alpha}^{(l)}$ during the retraining process; therefore, the computational efficiency can be improved significantly.

Neural Network with Bitwise Bottlenecks

As shown in Eq. 6, the bitwise bottleneck operation calculates the optimal coefficients associated with different bits of the compressed activation representation so that a minimal distortion can be achieved given the maximum code rate. This section shows how the bitwise bottlenecks work in the deep neural network.

Fig. 3 shows an example of efficient neural computing with bitwise bottlenecks. The whole network is built based on the classic ResNet, although the bitwise bottleneck operation can be easily integrated into different networks. In practice, the bitwise bottleneck operation can be inserted in the macroblock of the deep neural network. A typical macroblock contains a bitwise bottleneck layer, a convolution layer, a pooling layer (optional), a batch normalization layer and an activation layer. Thanks to the bitwise bottleneck layer, which transforms the normalized floating-point activation to compressed fixed-point activation, the convolution layer can substitute the computationally expensive floating-point multiplications for efficient fixed-point bitwise multiplications.

The benefits of inserting the bitwise bottlenecks in deep neural networks are twofold. From the perspective of *memory efficiency*, compared with the standard deep neural networks using 32-bit single-precision activation representation, the bitwise bottlenecks can compress the activation into arbitrary low-precision (1 to 8 bits), obtaining an improvement of memory efficiency by 32 to 4 times. From the perspective of *computational efficiency*, single-precision floating-point multiplication generally requires considerably more hardware resources and calculation time compared with fixed-point multiplication. By employing the bitwise

Algorithm 1: Training Algorithm

```
Input: Pre-trained floating-point CNN model $\boldsymbol{\Theta}$,
       threshold of PSNR loss $T$, number of layers $L$,
       number of training samples $N$, number of bits
       for initial quantization $D$;
Output: An Optimized Quantized model $\hat{\boldsymbol{\Theta}}$;
1  Obtain the floating-point activation tensors
   $\mathbf{X}_i^{(1)}, ..., \mathbf{X}_i^{(L)}$ of $\boldsymbol{\Theta}$ for each sample;
2  for $l$ = 1, ..., $L$ do
3      while $t^{(l)} < T$ do
4          for $i$ = 1, ..., $N$ do
5              Obtain the codebook $\hat{\mathbf{X}}_{i1}^{(l)}, \hat{\mathbf{X}}_{i2}^{(l)}, ..., \hat{\mathbf{X}}_{iD}^{(l)}$ by
               initial quantization of $\mathbf{X}_i^{(l)}$;
6              Obtain the coefficient vector $\boldsymbol{\alpha}^{(l)}$ via Eq. 6
               and restore $\hat{\mathbf{X}}_i^{(l)}$ via Eq. 4;
7              Compute the PSNR loss between $\hat{\mathbf{X}}_i^{(l)}$ and
               $\mathbf{X}_i^{(l)}$ as $t_i^{(l)}$ ;
8          $t^{(l)} = \max(t_i^{(l)})$ for each $i = 1, ..., N$;
9          Increase $\lambda$;
10 Insert the bottleneck layers into $\boldsymbol{\Theta}$ as a new model $\hat{\boldsymbol{\Theta}}$;
11 Refine and quantize the weights and the vector $\boldsymbol{\alpha}^{(l)}$ of
   $\hat{\boldsymbol{\Theta}}$ by backpropagation until reaching convergence.
```

bottleneck layers before the convolution layers (as shown in Fig. 1), the inference latency of deep neural networks may be reduced by over 90%, as indicated in early research (Zhou et al. 2018).

Training Sparse Bitwise Bottlenecks

In practice, the sparsity level of the activation may vary from layer to layer in the neural network. It is desired that different macroblocks should use different quantization precisions. As one of the advantages of the proposed method, the optimal code rates of different bitwise bottlenecks can be approximated by tuning a single hyper-

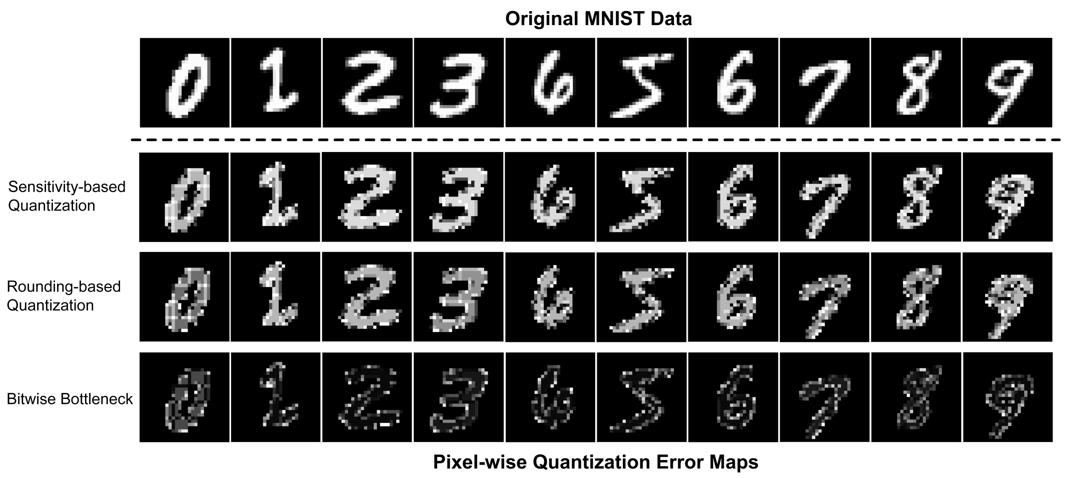

Original MNIST Data
Sensitivity-based Quantization
Rounding-based Quantization
Bitwise Bottleneck
Pixel-wise Quantization Error Maps

Figure 4: The bitwise bottleneck layer can compress the images using 5-bit representation, which is compared with the standard rounding-based quantization and sensitivity-based quantization methods. The error maps of pixel-wise absolute values of the difference between the compressed images and the original images are shown above.

parameter, which is the threshold of peak-signal-to-noise-ratio (PSNR) loss. The measurement of the PSNR is defined as $\mathrm{PSNR} = 10 \cdot \log_{10}\left(\frac{(2^D-1)^2}{\mathrm{MSE}}\right)$, where $\mathrm{MSE} = \frac{1}{P \cdot Q \cdot K}\|\mathbf{X}_i^{(l)} - \sum_{j=1}^{D} \alpha_j^{(l)} \hat{\mathbf{X}}_{ij}^{(l)}\|_2^2$. By increasing the hyperparameter of λ and comparing the PSNR loss with the threshold, each layer can approximate the respective acceptable minimal code rate independently. More detailed information about the training algorithm can be found in Algorithm 1.

An illustrative example of training the neural network with multiple bitwise bottlenecks is shown in Fig. 2. A closer look at the image reveals three interesting observations. First, the statistical phenomenon of bitwise sparsity exists. Specifically, the high-end bits of the activation representation are statistically near-zero for different layers of the neural network, which validates our assumption. Second, the optimal code rates of different layers depend on the level of activation sparsity. As an example, the activations of the first macroblock of ResNet50 are less sparse than that of the next three macroblocks, resulting in a higher acceptable minimal code rate. Third, it seems that the bitwise bottlenecks can *adaptively* reduce the coefficients $\alpha_j^{(l)}$ of both near-zero high-end bits and less-informative low-end bits of the activation representation.

Experiments

A list of experiments were carried out to evaluate the effectiveness of the proposed method.

Experimental Setting

Our experiments were performed on three standard benchmarks: the MNIST (LeCun et al. 1998), CIFAR10 (Krizhevsky, Hinton et al. 2009) and ImageNet (ILSVRC2012) (Deng et al. 2009) datasets. The MNIST dataset of 28×28 monochrome images contained 60 thousand training samples and 10 thousand test samples. The CIFAR10 dataset of 32×32 color images contained 50 thousand training samples and 10 thousand test samples, which had 10 classes. The ImageNet dataset contained 1.28 million training samples, 50 thousand verification samples, 100 thousand test samples and 1,000 classes.

The proposed Bitwise Bottleneck approach required an initial quantization operation to calculate the binary activation codebook at the first step. Two different initial quantization methods were adopted, including the Iterative Rounding method (Courbariaux, Bengio, and David 2014) and the DoReFa-Net method (Zhou et al. 2016). The Iterative Rounding method quantized the activation during the iterative backpropagation training process, which was widely used for deep neural network quantization. To validate our method, we used the standard average classification accuracy over the test dataset as our evaluation criterion. The results of the proposed method were achieved based on the ResNet50, and all the compared approaches were also based on ResNet50 for fair comparison.

All the experiments were performed on a computer equipped with Intel i7-7800X CPU and 2 NVIDIA TITAN Xp GPU. Similar to other studies, a backpropagation-based training process was adopted in the proposed method to refine and quantize the weight parameters of the deep neural network and the coefficients $\boldsymbol{\alpha}^{(l)}$. In our experiment, the ADAM (Kingma and Ba 2014) algorithm was applied to implement the backpropagation process, where the learning rate was 0.0001, and the batch size was 64. We retrained 4,000 iterations on the MNIST and CIFAR10 datasets and 5 epochs on the ImageNet dataset.

Visualization and Analysis

A group of experiments were performed over the MNIST dataset to analyze and visualize the performance of the proposed method compared with the initial and other quantization approaches. The bitwise bottlenecks reduced the code

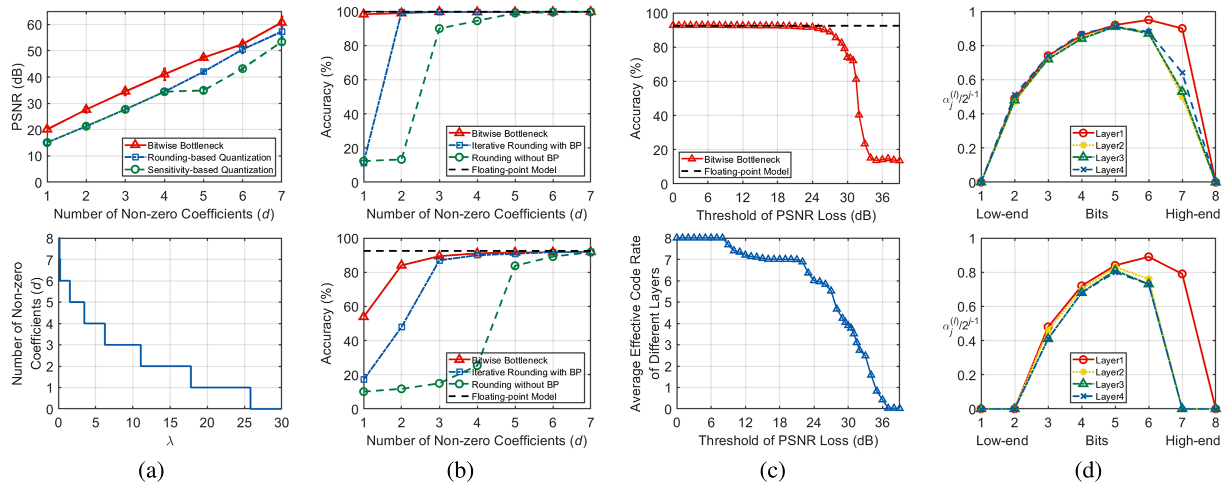


Figure 5: (a) The relation between the PSNR and the effective code rate, which equals the number of nonzero coefficients in $\boldsymbol{\alpha}^{(l)}$ and the relation between the effective code rate and the value of λ. (b) Accuracy of the deep neural network with bitwise bottlenecks over the MNIST (upper) and CIFAR10 (lower) datasets, which can compress the activation of the deep neural network with different effective code rates d. (c) The change in accuracy and average effective code rate of different layers when using the bitwise bottlenecks to compress the 8-bit fixed-point activation according to different PSNR loss thresholds. (d) The ratio of $\alpha_j^{(l)}/2^{j-1}$ with different thresholds of PSNR loss ($T = 26$ upper, $T = 30$ lower).

rate of the fixed-point representation from the initial quantization length of D bits to the effective code rate d, which was the number of nonzero coefficients in vector $\boldsymbol{\alpha}^{(l)}$ (Eq. 6). Fig. 4 visualized the loss of activation integrity caused by 5-bit activation quantization over the MNIST dataset. The absolute values of the difference between the compressed image data and the original image data were compared between the proposed method and the baseline quantization methods. In summary, the proposed Bitwise Bottleneck method showed the highest PSNR (47.30±0.51 dB) for activation quantization compared with the standard rounding-based method (42.02±0.54 dB) and sensitivity-based method (34.94±0.84 dB) (Li et al. 2020).

Fig. 5(a) shows the relation between the effective code rate d and the average PSNR achieved over the MNIST dataset. Overall, the proposed Bitwise Bottleneck approach achieved 1.98 dB to 12.44 dB higher PSNR compared with the classic rounding-based quantization and sensitivity-based quantization methods. The superiority of PSNR was consistent when different numbers of nonzero coefficients were chosen, which can be controlled by the hyperparameter of λ, as shown in the figure.

Similar to other activation quantization approaches, a backpropagation retraining process was applied after the activation quantization process to refine the weight parameters and the coefficients $\boldsymbol{\alpha}^{(l)}$ that were quantized to D bits with a round-based approach. Fig. 5(b) shows the final classification accuracy achieved with backpropagation. The average accuracy over both the MNIST and CIFAR10 datasets was evaluated with various numbers of nonzero coefficients d. Experimental results showed that the proposed method outperformed the baseline approach when fewer than four bits were used for activation quantization. The loss of accuracy caused by the rounding-based quantization became worse when the dataset was more complicated, while the Bitwise Bottleneck approach suffered almost no loss of accuracy compared with the floating-point model when more than three bits were used for activation representation.

The Bitwise Bottleneck approach featured its ability to automatically determine the effective code rates of different layers. By setting the threshold of PSNR loss accepted for activation quantization, the bitwise bottlenecks could estimate the minimal effective code rate by the sparse optimization. Fig. 5(c) shows the relation among the threshold of PSNR loss, the average effective code rate across different layers, and the classification accuracy. As shown in our experiments over the CIFAR10 dataset, the classification accuracy declined when the threshold of PSNR loss increased. However, when the PSNR loss was less than 24 dB, almost no decrease in classification accuracy was detected (less than 1%). The accuracy began to decrease when fewer than 6 bits on average were used for activation representation.

The Bitwise Bottlenecks approach handled the task of activation quantization as a trade-off between compression and prediction. Fig. 5(d) shows the ratio of $\alpha_j^{(l)}/2^{j-1}$ for each bit of the quantized activation, where 2^{j-1} is the natural coefficient of Eq. 3, which reflects how much information the jth bit contains. Two interesting observations could be found in the image. First, the low-end bits of the activation representation were removed by the bottlenecks, leading to minimal loss of information caused by bitwise compression. Second, despite containing more information, the high-end bits were also removed by the bottlenecks, which was caused by the bitwise sparsity in high-end bits (as shown in Fig. 2).

Method	Year	Weights	Activations	Top-1 Acc (%)	Top-5 Acc (%)
Floating-point Model	2016	32	32	75.6	92.8
Bitwise Bottleneck	–	8	5	**75.7**	**92.7**
DSQ	2019	8	5	75.2	92.5
Focused Compression	2019	5	32	74.9	92.6
Integer-only	2018	8	8	74.9	–
INQ	2017	5	32	74.8	92.5
DoReFa-Net (initial quantization)	2016	8	5	73.8	91.7
Iterative Rounding	2014	8	5	72.1	90.4
Bitwise Bottleneck	–	8	4	**74.8**	**92.2**
DSQ	2019	8	4	73.8	91.7
UNIQ	2018	4	8	73.4	–
ACIQ	2019	8	4	71.8	–
DoReFa-Net (initial quantization)	2016	8	4	70.1	89.3
Iterative Rounding	2014	8	4	70.0	89.4

Table 1: Comparing with state-of-the-art over the ImageNet dataset.

Compared with the State-of-the-Art

We evaluated the proposed method with the state-of-the-art activation quantization approaches over the ImageNet dataset. The original floating-point model and the proposed Bitwise Bottleneck approach were compared with the DSQ (Gong et al. 2019), ACIQ (Banner, Nahshan, and Soudry 2019), Focused Compression (Zhao et al. 2019b), Integer-only (Jacob et al. 2018), UNIQ (Baskin et al. 2018), INQ (Zhou et al. 2017), DoReFa-Net (Zhou et al. 2016) and Iterative Rounding (Courbariaux, Bengio, and David 2014) approaches. Thresholds of PSNR loss of 8 dB and 16 dB were adopted to quantize the activations, which resulted in an average effective code rate of $5.0\ (\pm 1)$ and $4.0\ (\pm 1)$ bits. The quantization method of DoReFa-Net (Zhou et al. 2016) was adopted for initial quantization. The standard TensorFlow quantization tool was used to quantize the weights to 8 bits and all the coefficients of $\boldsymbol{\alpha}^{(l)}$ were quantized to 6 bits.

We found in our experiments that most quantization methods could achieve almost the same accuracy as the floating-point model with 6 to 8 bits activation representation, but at 5-bit and below the accuracy began to drop obviously. Table. 1 summarizes the results of classification accuracies over the ImageNet dataset. As shown in the table, the top-1 accuracy loss of 5-bit activation representation was up to 3.5%, and top-1 accuracy loss of 4-bit activation representation was between 1.8% to 5.6%. However, the Bitwise Bottleneck method could achieve marginal loss of accuracy at 5-bit and 4-bit. The loss of accuracy at 4-bit was 0.8%, which was 4.7% higher than the initial quantization method (DoReFa-Net). What's more, the Bitwise Bottleneck method exceeded the Integer-only method with 8-bit activation, the INQ and Focused Compression methods with 32-bit floating-point activation.

Efficiency Improvement

From the perspective of reducing computational complexity, when the neural network was implemented in a bitwise way, a lower code rate led to linearly fewer bitwise operations. Taking the 8-bit fixed-point representation as an example, each fixed-point operation could be implemented by 8 bitwise operations, and if we cut off 1 bit, the number of operations could be reduced by 12.5%. In addition, by reducing a single bit, the hardware source requirement was also reduced by 12.5% when it was implemented using dedicated hardware. Generally, compared with the 32-bit activations representation, the bitwise bottlenecks could improve the computational efficiency by over 6.4 times without hurting the performance of the deep neural network. Besides, considering the weighs were quantized to 8-bit, the computational efficiency improvement could reach 25.6 times.

From the perspective of reducing memory occupation, reducing the activation code rate linearly decreased the memory requirement. The activation of a standard neural network was usually represented by 32-bit floating-point, which occupied 4 bytes of memory. However, the bitwise bottlenecks could compress them to fewer than 5 bits without hurting the classification accuracy, where the running memory could be reduced by 84%.

Conclusion and Discussion

This paper presents the Bitwise Bottleneck approach for neural network activation quantization. Based on the rate-distortion theory, the bitwise bottlenecks inserted in the deep neural network attempt to minimize the quantization error rate and code rate. Experiments over different datasets show that the proposed method could reduce the code rate of the activation of the deep neural network to one to five bits without hurting the performance, which could improve the memory efficiency as well as the computational efficiency of the neural network inference by over 6 times.

The proposed Bitwise Bottleneck method assumes that the activations of deep neural networks have a certain level of sparsity; what's more, in practice, the activations of different layers have different levels of sparsity. Although applying flexible quantization as adopted by different bitwise bottlenecks could reduce the quantization PSNR loss, the proposed method is subject to the constraint of activation sparsity. In the future, we plan to incorporate sparse regularization to intentionally introduce sparsity in the activation so that the bitwise bottlenecks could achieve higher activation integrity with a lower code rate.

Acknowledgments

This research is supported by the national natural science foundation of China (contract 61971072).

References

Banner, R.; Nahshan, Y.; and Soudry, D. 2019. Post training 4-bit quantization of convolutional networks for rapid-deployment. In *Advances in Neural Information Processing Systems*, 7948–7956.

Baraniuk, R. G. 2007. Compressive sensing. *IEEE signal processing magazine* 24(4).

Baskin, C.; Schwartz, E.; Zheltonozhskii, E.; Liss, N.; Giryes, R.; Bronstein, A. M.; and Mendelson, A. 2018. Uniq: uniform noise injection for the quantization of neural networks. *arXiv preprint arXiv:1804.10969* 3.

Berger, T. 2003. Rate-distortion theory. *Wiley Encyclopedia of Telecommunications* .

Chen, C.; Seff, A.; Kornhauser, A.; and Xiao, J. 2015. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE International Conference on Computer Vision*, 2722–2730.

Courbariaux, M.; Bengio, Y.; and David, J.-P. 2014. Training deep neural networks with low precision multiplications. *arXiv preprint arXiv:1412.7024* .

Courbariaux, M.; Bengio, Y.; and David, J.-P. 2015. Binaryconnect: Training deep neural networks with binary weights during propagations. In *Advances in neural information processing systems*, 3123–3131.

Courbariaux, M.; Hubara, I.; Soudry, D.; El-Yaniv, R.; and Bengio, Y. 2016. Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1. *arXiv preprint arXiv:1602.02830* .

Dai, B.; Zhu, C.; and Wipf, D. 2018. Compressing neural networks using the variational information bottleneck. *arXiv preprint arXiv:1802.10399* .

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, 248–255. Ieee.

Gong, R.; Liu, X.; Jiang, S.; Li, T.; Hu, P.; Lin, J.; Yu, F.; and Yan, J. 2019. Differentiable soft quantization: Bridging full-precision and low-bit neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, 4852–4861.

Gupta, S.; Agrawal, A.; Gopalakrishnan, K.; and Narayanan, P. 2015. Deep learning with limited numerical precision. In *International Conference on Machine Learning*, 1737–1746.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Jacob, B.; Kligys, S.; Chen, B.; Zhu, M.; Tang, M.; Howard, A.; Adam, H.; and Kalenichenko, D. 2018. Quantization and training of neural networks for efficient integer-arithmetic-only inference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2704–2713.

Kim, M.; and Smaragdis, P. 2016. Bitwise neural networks. *arXiv preprint arXiv:1601.06071* .

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Kravchik, E.; Yang, F.; Kisilev, P.; and Choukroun, Y. 2019. Low-bit Quantization of Neural Networks for Efficient Inference. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 0–0.

Krizhevsky, A.; Hinton, G.; et al. 2009. Learning multiple layers of features from tiny images. Technical report, Citeseer.

LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P.; et al. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE* 86(11): 2278–2324.

Lee, J. H.; Ha, S.; Choi, S.; Lee, W.-J.; and Lee, S. 2018. Quantization for rapid deployment of deep neural networks. *arXiv preprint arXiv:1810.05488* .

Li, Y.; Lin, S.; Zhang, B.; Liu, J.; Doermann, D.; Wu, Y.; Huang, F.; and Ji, R. 2019. Exploiting Kernel Sparsity and Entropy for Interpretable CNN Compression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2800–2809.

Li, Y.; Zhang, S.; Zhou, X.; and Ren, F. 2020. Build a compact binary neural network through bit-level sensitivity and data pruning. *Neurocomputing* .

McCool, C.; Perez, T.; and Upcroft, B. 2017. Mixtures of lightweight deep convolutional neural networks: Applied to agricultural robotics. *IEEE Robotics and Automation Letters* 2(3): 1344–1351.

Qiu, Z.; Yao, T.; and Mei, T. 2017. Deep quantization: Encoding convolutional activations with deep generative model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 6759–6768.

Rastegari, M.; Ordonez, V.; Redmon, J.; and Farhadi, A. 2016. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European Conference on Computer Vision*, 525–542. Springer.

Shwartz-Ziv, R.; and Tishby, N. 2017. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810* .

Sze, V.; Chen, Y.-H.; Yang, T.-J.; and Emer, J. S. 2017. Efficient processing of deep neural networks: A tutorial and survey. *Proceedings of the IEEE* 105(12): 2295–2329.

Tishby, N.; and Zaslavsky, N. 2015. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, 1–5. IEEE.

Wu, B.; Iandola, F.; Jin, P. H.; and Keutzer, K. 2017. Squeezedet: Unified, small, low power fully convolutional neural networks for real-time object detection for autonomous driving. In *Proceedings of the IEEE Conference* *on Computer Vision and Pattern Recognition Workshops*, 129–137.

Wu, J.; Leng, C.; Wang, Y.; Hu, Q.; and Cheng, J. 2016. Quantized convolutional neural networks for mobile devices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 4820–4828.

Xu, H.; Gao, Y.; Yu, F.; and Darrell, T. 2017. End-to-end learning of driving models from large-scale video datasets. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2174–2182.

Zhao, R.; Hu, Y.; Dotzel, J.; De Sa, C.; and Zhang, Z. 2019a. Improving neural network quantization using outlier channel splitting. *arXiv preprint arXiv:1901.09504* .

Zhao, Y.; Gao, X.; Bates, D.; Mullins, R.; and Xu, C.-Z. 2019b. Focused Quantization for Sparse CNNs. In *Advances in Neural Information Processing Systems*, 5585–5594.

Zhou, A.; Yao, A.; Guo, Y.; Xu, L.; and Chen, Y. 2017. Incremental network quantization: Towards lossless cnns with low-precision weights. *arXiv preprint arXiv:1702.03044* .

Zhou, S.; Wu, Y.; Ni, Z.; Zhou, X.; Wen, H.; and Zou, Y. 2016. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160* .

Zhou, X.; Li, S.; Tang, F.; Hu, S.; Lin, Z.; and Zhang, L. 2018. DANoC: An Efficient Algorithm and Hardware Codesign of Deep Neural Networks on Chip. *IEEE Trans Neural Netw Learn Syst* 29(7): 3176–3187.