# Hybrid Approach for Efficient Quantization of Weights in Convolutional Neural Networks

混合方法的有效量化
卷积神经网络的权值

Sanghyun Seo
Department of Computer Engineering
Dongguk University
Seoul, Republic of Korea
shseo@dongguk.edu

Juntae Kim
Department of Computer Engineering
Dongguk University
Seoul, Republic of Korea
jkim@dongguk.edu

***Abstract*—Convolutional neural networks(CNN) have achieved outstanding results in the fields of image recognition which classifies objects in the input images. In the deep neural networks such as CNN, the number of layers and the number of neurons in each layer are large. In other words, the deep neural networks requires relatively large storage space and calculation process. However, in embedded devices for object recognition in autonomous vehicles, large storage space and high computational complexity are constraints. For this reasons, various methodologies have been proposed to apply CNN to small embedded hardware such as mobile devices, FPGA and ASIC efficiently. In this paper, we quantize the weights of AlexNet without a large drop in accuracy by using a hybrid quantizer using uniform quantizer and k-means clustering.**

换句话说，深层神经网络需要较大的存储空间和计算过程。

采用均匀量子化和k-means聚类的混合量子化方法，对AlexNet的权值进行量化，精度没有较大的下降。

***Keywords— Neural Networks Compression; Convolutional Neural Networks; Weights Quantization; Hybrid Quantizer***

## I. INTRODUCTION

A variety of CNN-based models have been proposed in the field of image recognition and have achieved remarkable results[1][2][3][4]. CNN generally consists of convolution layers, pooling layers, and fully-connected layers. In the convolution layer, a feature map is generated by performing a convolution of input data and filters. In the pooling layer, the feature map become smaller because of a result of the sub-sampling from original feature map, and the amount of computation is reduced to the next layer. In the fully-connected layer, the characteristics of the input data obtained through repetition of convolution and pooling are used as input data of fully-connected layer to perform final label classification. In this way, the CNN extracts various characteristics of the input data through the convolution layer and the pooling layer, and performs final object recognition in the fully-connected layer, thereby achieving classification accuracy exceeding human object recognition accuracy[4].

By effectively arranging the convolution layer and pooling layers in the object recognition field and extracting the characteristics of the input data, better object recognition performance can be expected. However, in order to better extract the characteristics of the input data, constructing deep neural networks. As the neural networks deepens, the computational amount of the neural networks also increases. Therefore, it is essential to have a high performance computing environment capable of performing deeper neural networks operations basically for learning and application of the deep neural networks.

In order to train CNN, a hardware operator such as a graphic processing unit(GPU) is necessary. In contrast to this, in case of performing only forward operation to apply already trained CNN, it is relatively free from hardware constraints. Nevertheless, forward computation of CNN still requires a large amount of computation, so it is still limited in applications such as mobile devices and hardware devices such as FPGA and ASIC. In the embedded environment for object recognition, the increase of the neural networks computation load lowers the power consumption efficiency of the hardware and increases the total time required to perform object recognition, which burdens the construction of a real-time object recognition model. In addition, since the production of high performance embedded hardware capable of performing a large scale operation requires a relatively high production cost, it poses a great difficulty in applying an object recognition model based on a composite neural networks in a real industry.

In order to utilize CNN in the embedded environment based on this problem consciousness, it is necessary to research the technique of reducing the size of the neural networks without changing the performance. For example, in FPGA-based hardware, the number of multipliers required for operation is reduced according to the size of bits representing input data[5]. Therefore, it is possible to reduce the number of FPGA-based hardware multipliers that actually perform computation by reducing the bit size of connection weights from the neuron to neuron. As a result, it is possible to increase the power efficiency of the hardware, and the real-time property of the object recognition model in the embedded environment can be secured.

Various studies have been conducted to apply CNN in embedded environment. First, a dedicated accelerator for CNN application is being researched[5], and CNN compression or optimization techniques are being studied[6]. In terms of CNN compression, this paper proposes a hybrid quantizer which uses both uniform quantizer and k-means clustering method for efficient quantization of the CNN object recognition model. We

This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT (NRF-2017M3C4A7083279).

2375-9356/18/$31.00 ©2018 IEEE
DOI 10.1109/BigComp.2018.00114

can show that a quantized CNN with little loss in classification accuracy compared with the original recognition model. This quantized AlexNet shows that it is suitable for applications in a small embedded environment by reducing the size of neural networks using codebook while maintaining the performance of existing AlexNet.

## II. Related Works

Compression of Neural networks means a set of methodologies that reduce the size of the weights and structure of neural networks while keeping its original performance as feasible as possible. Compression of neural networks is performed by the following two approaches. The first is an approach to try to compress neural networks in the training process, and the other is an approach to try to compress the trained neural networks. In particular, there is a case in which the weights of the input data and the neural network is constructed in units of bits[7][8]. On the other hand, there is a case of squeeze the size of the neural networks filter[9]. Most of these methods have a common use of the approach of compressing the weights of neural networks bit by bit or compressing the structure in the course of training neural networks.

While there is an approach to perform compression in the learning process of the neural networks, there is an approach to restore the performance of the compressed neural networks by performing re-training. For example, there is a pruning method[10]. It reduces neural networks by using pruning and re-training for restoring accuracy. This approach improves the compression efficiency of neural networks by applying quantization as well as pruning.

There is a methodology for quantizing the weight of the pre-trained neural networks[10][11]. This method is compressing the overall size of a neural networks by applying a codebook or look-up table to quantized weights[12]. Especially in recent years, it has been proposed to reduce the redundancy of neural networks while quantizing them[13]. The quantization technique of a neural network generally refers to a technique of converting a connection weight of a neural network represented by a floating point real value into a subset of a specific range divided by a quantization level. An example is the conversion of neural networks connection weights expressed in 32-bit floating point to representation in 8-bit bands. The bandwidth of the weights of neural networks can be greatly reduced through the quantization[5]. In the above example, reading an 8-bit value into memory requires only 25% of memory bandwidth, rather than reading a 32-bit floating point value. Therefore, a bottleneck for accessing the RAM can be prevented, and an 8-bit acceleration can be performed using a hardware chip[6][11]. It is possible to perform more efficient application in an embedded device or the like because the computation of the quantization bits is possible[12] [13].

## III. Weights Quantizaqtion using Hybrid Quantizer

Many existing studies attempt to reconstruct lost accuracy by re-training after neural networks compression through pruning, quantization, and so on. Among them, there is an approach minimizing the loss of accuracy using hessian weighted k-means and performs quantization[13]. In this paper, we reduce the burden of re-training after quantization and propose a quantization technique for the approach using codebook.

### A. Uniform and Non-Uniform Quantization

Quantization generally refers to dividing data with a continuous variation into finite levels of division and assigning a specific value to each level. The most basic form of quantization is uniform quantization. Uniform quantization is a method of having a quantizing interval of the same size within a certain range. For example, there is a method of setting the size of the quantizing interval by dividing the minimum and maximum values for specific input data by quantization bits desired to be quantized.

量子化通常是指将数据以连续变化的形式划分为有限的划分层次，并为每个层次分配一个特定的值。

均匀量子化是在一定范围内具有相同大小的量子化间隔的一种方法。

However, uniform quantization is most efficient when the distribution of input data is uniform. If the distribution of input data is a special distribution, such as a gaussian distribution, a laplace distribution, or a gamma distribution, then the step size of the optimal quantizer will be different fixed point quantization of deep convolutional networks. Non-uniform quantization is a method that can efficiently quantize input data that is not uniform in distribution. Typically, there is a Lloyd-Max quantizer that computes the MSE repeatedly based on the probability density function to quantize the high-density section of the probability density function and increase the quantizing interval if the density is low. This method is an efficient methodology to reduce quantization error rather than uniform quantization according to the distribution of input data.

### B. K-Means Clustering for Qu

通过k -means聚类将量化位调整为k，根据聚类结果将每个簇的数据量化到簇的中心值。

Quantization through K-means clustering is a good solution to solve the problem of quantization of CNN weights by the existing quantization method. Quantization through K-means adjusts the quantization bit to k and quantizes the data of each cluster to the center value of the cluster based on the clustering result. (1) means k-means clustering that performs scalar quantization to K bits.

$$\underset{C}{\operatorname{argmax}} \sum_{i=1}^{K\ bit} \sum_{w \in c_i} |c_i - w|^2 \qquad (1)$$

When C = $\{c_1, c_2, ..., c_i\}$, C iterates to minimize the squared error. As a result of (1), each w can be quantized to the center value of the corresponding cluster. As a result, weights are quantized by the size of the target quantization bit.

### C. Hybrid Quantizer using Uniform Quantizer and K-Means Clustering

The hybrid quantizer proposed in this paper consists of two stages. The first is to uniformly quantize the weights of CNN to 8 bits, then to perform a kind of non-uniform quantization so that the values can be mapped to all quantization levels by applying k-means. quantization suitable for the frequency of the weights can be performed by using k-means which are determined value of each weights.

首先将CNN的权值统一量化为8位，然后进行一种非均匀量化，通过k-means将值映射到所有量化级别。

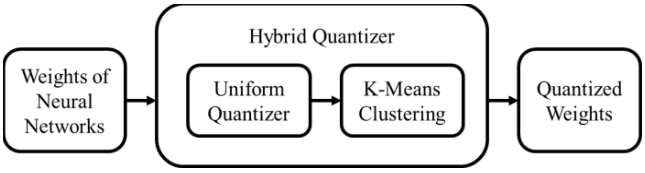


Fig. 1. Workflow of Proposed Hybrid Quantizer

Fig. 1. shows the workflow of proposed hybrid quantizer which combines quantization and k-means. The reason why we don't use only k-means clustering but hybrid quantizer is due to the approach that it does not guarantee the performance of the neural network even if the quantization error is small[13]. In other words, quantization with K-means clustering directly on weights may yield smaller quantization errors, but the results do not guarantee neural network performance. Rather, the performance of the neural network may be reduced even though the quantization error is small. Also, in the case of the proposed hybrid quantizer, since k-means is performed for the quantized weights in advance, the quantization speed can be increased by converting the addition operation to the multiplication through the improvement of the algorithm.

换句话说，直接对权值进行k均值聚类的量化可能会产生较小的量化误差，但结果并不能保证神经网络的性能。

此外，对于所提出的混合量化器，由于事先对量化的权值进行了k-means运算，通过改进算法将加法运算转换为乘法运算，可以提高量化速度

## IV. Experiments

In experiment, we use the AlexNet architecture that we learned on our own. AlexNet consists of five convolution layers and three fully-connected layers [1]. Trained weights of AlexNet are similar to the gaussian distribution. The AlexNet structure used in the experiment can perform object recognition for a total of 1000 labels using 1.3 million ImageNet training datasets. The trained object recognition model is tested by using 50,000 ImageNet validation datasets and measured top-1 accuracy and top-5 accuracy. Quantization error was also measured to evaluate the performance of each quantization methodology. The quantization error is the sum of the quantization errors of the eight layers of AlexNet.

### A. Experiments Quantizer Models

We experimentally quantize the weights of AlexNet using a total of three quantizers. The first model uses a uniform quantizer. This quantizer has a range of minimum and maximum values. The second model is a basic k-means quantizer which is Similar to Lloyd's Algorithms that quantizer have been used to perform scalar quantization. Finally, we tested the performance of a hybrid quantizer combining a uniform quantizer and a k-means quantizer. This is a model that performs non-uniform quantization once more on uniform quantized weights. Therefore, this model has the characteristics of a uniform quantizer and a k-means quantifier.

### B. Experiments Result and Discussion

The experimental results can be summarized as in table 1. Fig. 2. shows the top-1 accuracy of the proposed hybrid quantization methodology compared to a uniform quantizer. Fig. 3. is an example of the first and last layer of the weight distribution of quantized AlexNet.

TABLE I. TABLE 1 EXPERIMENT RESULTS

| Quantization Bit | Quantizer | Quantization Error | Top-1 Accuracy | Top-5 Accuracy |
|---|---|---|---|---|
| - | Original | | 54.38 | 78.15 |
| 5bit | quantization | 50157 | 50.6 | 76.05 |
| | k-means | 16348 | 53.91 | 77.72 |
| | hybrid quantizer | 17808 | 53.93 | 77.81 |
| 4bit | quantization | 100276 | 6.48 | 15.99 |
| | k-means | 31362 | 51.94 | 76.06 |
| | hybrid quantizer | 32625 | 52.6 | 76.98 |
| 3bit | quantization | 197422 | 0.17 | 0.55 |
| | k-means | 59103 | 31.65 | 55.38 |
| | hybrid quantizer | 59676 | 28.95 | 51.77 |
| 2bit | quantization | 286478 | 0.13 | 0.42 |
| | k-means | 106499 | 0.59 | 2.29 |
| | hybrid quantizer | 108093 | 0.54 | 2.13 |

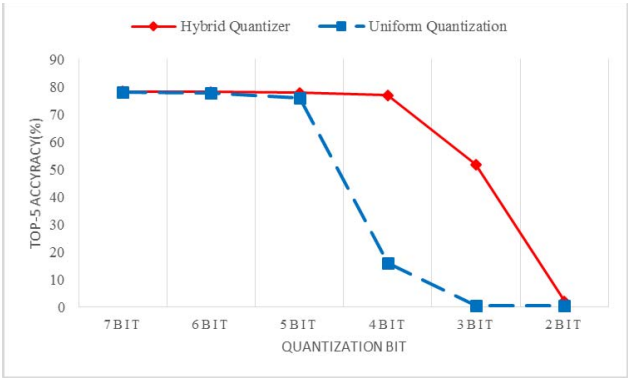


Fig. 2. Top-1 Accuracy Comparison

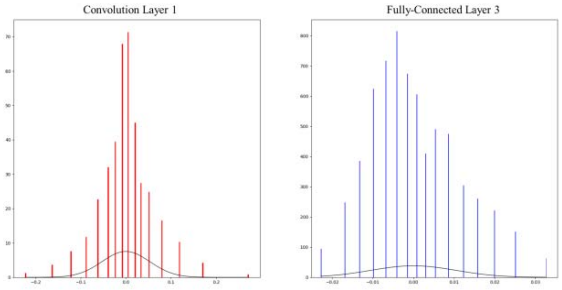


Fig. 3. Example of weights distribution in quantized AlexNet

Experiments were carried out by reducing the quantization bits from 8bits to 2bits and evaluated how many the CNN object recognition result was robust even at low quantization bits. In Fig. 2, Experimental results show that all models maintain their performance up to 5bit including uniform quantization model. However, since the quantization bit becomes 4bit, CNN using a uniform quantizer can't perform classification. In contrast, CNN using a proposed hybrid quantizer can perform classification without any significant accuracy decrease even at 4 bits, and

even at 3 bits, it maintains a half of the original accuracy. Fig. 3. shows the distribution of weights of quantized AlexNet with 4 bits. The weight of a neural network is a gaussian distribution due to the normalization in training process. Therefore, when quantizing a neural network by using non-uniform quantizer, A dense quantization level appears at the median of high frequency. In this way, the neural network can be operated using the weight of a small bandwidth.

There two summary of experiments. First, we can confirm that the proposed model is robust even at low quantization bits below 4 bits. In addition, the quantizer error of the uniform quantizer exponentially increases, and the performance of the neural network is completely collapsed from 4bit. Second, we can compare the simple k-means quantizer and the hybrid quantizer. The performance of the two quantizers is similar, but the quantization error of the simple k-means is small at all quantization bits. Nevertheless, CNN accuracy of 5bit and 4bit is high hybrid. In other words, it can be experimentally confirmed that the performance of the quantization does not necessarily guarantee the performance of the quantized neural network. Since the proposed hybrid quantizer performs clustering on weights already quantized, it is possible to improve the quantization speed by improving the algorithm.

## V. Conclusion

This paper proposes a hybrid quantizer for efficient compression of CNN. Experimental results show that the performance of the neural network can be maintained at a lower accuracy than the uniform quantization technique. In the experiment, comparison with the conventional quantization method using k-means was also performed. Although all bits did not outperform k-means, they did perform better in some intervals.

Particularly, in order to perform object recognition using a vision sensor in autonomous vehicles, various constraints are accompanied. For efficient application to autonomous vehicles, hardware cost, power efficiency, computation speed, and object recognition performance should all be considered. In future, it will be necessary to study the neural network compression technique that takes advantage of various quantization methodologies. More specifically, it is necessary to study a compressed neural network model that performs both object detection and object recognition.


## Acknowledgment

This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Science, ICT (NRF-2017M3C4A7083279).


## References


[1] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks.", Advances in neural information processing systems, 2012.

[2] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." arXiv preprint arXiv:1409.1556, 2014.

[3] Szegedy, Christian, et al. "Going deeper with convolutions." Proceedings of the IEEE conference on computer vision and pattern recognition. 2015.

[4] He, Kaiming, et al. "Deep residual learning for image recognition.", Proceedings of the IEEE conference on computer vision and pattern recognition, 2016.

[5] Qiu, Jiantao, et al. "Going deeper with embedded fpga platform for convolutional neural network.", Proceedings of the 2016 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays. ACM, 2016.

[6] Han, Song, Huizi Mao, and William J. Dally. "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding.", arXiv preprint arXiv:1510.00149, 2015.

[7] Rastegari, Mohammad, et al. "Xnor-net: Imagenet classification using binary convolutional neural networks.", European Conference on Computer Vision. Springer International Publishing, 2016.

[8] Courbariaux, Matthieu, et al. "Binarized neural networks: Training deep neural networks with weights and activations constrained to+ 1 or-1.", arXiv preprint arXiv:1602.02830, 2016.

[9] Iandola, Forrest N., et al. "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size.", arXiv preprint arXiv:1602.07360, 2016.

[10] Han, Song, Huizi Mao, and William J. Dally. "Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding.", arXiv preprint arXiv:1510.00149, 2015.

[11] Wu, Jiaxiang, et al. "Quantized convolutional neural networks for mobile devices.", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016.

[12] Bagherinezhad, Hessam, Mohammad Rastegari, and Ali Farhadi. "LCNN: Lookup-based Convolutional Neural Network.", arXiv preprint arXiv:1611.06473, 2016.

[13] Choi, Yoojin, Mostafa El-Khamy, and Jungwon Lee. "Towards the Limit of Network Quantization.", arXiv preprint arXiv:1612.01543, 2016.