Research on Machine Learning Optimization Algorithm of CNN for FPGA Architecture

Xiaodong Zhao1,a, Xunying Zhang2,b, Fan Yang1,c, Peiyuan Xu1,d, Wantong Li1,e, Fayang Chen3,f

1 Unmanned System Research Institute, Northwestern Polytechnical University, Xi’an, China
2 School of Electronics And Information, Northwestern Polytechnical University, Xi’an, China
3 Department of Flight Control, AVIC Xi’an Flight Automatic Control Research Institute, Xi’an, China
a xdz@nwpu.edu.cn
b *Corresponding author: zhangxy@nwpu.edu.cn, c yf4095@mail.nwpu.edu.cn, d xpy97916@mail.nwpu.edu.cn,
e m13001352576@163.com, f chen0805@126.com

Abstract—In order to meet the application requirements of deep learning detection and recognition algorithm for Field Programmable Gate Array (FPGA) computing structure, this paper proposes a machine learning optimization algorithm of convolutional neural network (CNN) based on pruning and Int8 quantization. By analyzing the machine learning intelligent recognition network, the Yolo v3 target detection network is selected to verify the optimization algorithm. Aiming at the Yolo v3 network with balanced speed and precision, the optimization algorithm is studied in detail. Based on Amazon Cloud FPGA EC2 instance platform and ZCU104 FPGA hardware platform, machine learning optimization algorithms are used to accelerate the application of Yolo v3 network detection and recognition. Satisfactory results are obtained on both FPGA computing platforms. The experimental results show that the machine learning optimization technology can improve the inference speed of neural network based on both FPGA platforms. The network structure optimized by pruning and Int8 quantization algorithms can achieve high acceleration speed, with very limited accuracy loss.

1. INTRODUCTION

Convolutional Neural Network (CNN) has a large number of layers and nodes, so it needs machine learning optimization technology and heterogeneous computing technology to reduce its storage and computing requirements. Machine learning optimization can effectively reduce the parameter redundancy, the storage occupation, the communication bandwidth and the computational complexity, which is helpful to the deployment of embedded hardware platform for CNN intelligent algorithms. Heterogeneous computing uses co-processing hardware engine to complete the deployment of CNN model in various embedded hardware computing platforms, including Field Programmable Gate Array (FPGA), ARM or Application Specific Integrated Circuit (ASIC), etc. Among all kinds of embedded
hardware computing platforms, FPGA has a wide range of applicability in the field of intelligent computing, because of its advantages of logic programmability and low power consumption. At present, a lot of researches focus on the intelligent computing structure design of FPGA. FPGA cloud platform architecture for CNN is proposed in [1]. Based on FPGA cloud platform, CNN operation is implemented and deployed in the cloud. The CNN resource based on FPGA cloud platform is used by local users through the network. Reference [2] points out that compared with the traditional technology, the existing RTL technology can be used as an accelerated low-power design scheme for CNN target recognition system. The most energy-saving design technology of CNN accelerator mainly focuses on high-level-synthesis (HLS), such as memory bandwidth optimization, network structure reconstruction, data reuse and batch normalization, but these attempts have reached the limit of the effect. The paper points out that RTL low power design technology can be applied to the original FIFO to reduce the power consumption in the process of data conversion. Very High Speed Integrated Circuit Hardware Description Language (VHDL) is used for implement of CNN classification algorithm based on Xilinx zynq-7000 FPGA in [3]. A scalable framework FPDeep is proposed in [4], which can help users effectively map CNN logic to multiple FPGAs and automatically generate RTL implementation. Two mechanisms are proposed. One is to balance the workload by using special partition and mapping strategy, so as to improve the utilization of DSP in FPGA; the other is to use only on-chip memory in convolution layer.

The above literatures only consider from the FPGA design level, and lack of algorithm design support. With the continuous development of machine learning optimization, some scholars consider improving the algorithm from the perspective of both FPGA design and machine learning optimization. The key to the research of universal machine learning optimization algorithm for FPGA computing architecture is, how to reduce the computing cost and storage space without affecting the network performance. At present, the research results based on neural network machine learning optimization algorithm mainly include neural network pruning, network model quantization, kernel sparseness, low rank decomposition, binary weight, lightweight network design and so on. The most suitable general methods include neural network pruning and network model quantization. This paper will focus on these two machine learning optimization methods.

A new pruning optimization algorithm based on energy perception is proposed in [5], which directly uses CNN energy consumption to guide the pruning process. Compared with the traditional pruning method, the energy estimation method uses the extrapolated parameters from the actual hardware. The algorithm realizes pruning by minimizing the error in the output feature mapping rather than the weight of the filter. For each layer, the weight is pruned first, and then the local fine-tuning is performed by the least square method to recover the accuracy quickly. After pruning all layers, reverse propagation is used to globally fine-tune the entire network. A load balancing pruning method is proposed in [6], which can compress the size of LSTM model by 20 times (pruning 10 times and quantizing twice), and the loss of the prediction accuracy can be ignored. The FPGA system based on Xilinx XCKU060 is 43 times faster and 3 times faster than Core i7 5930k CPU and Pascal Titan X GPU respectively in LSTM speech recognition benchmark test. Compared with CPU and GPU, its energy efficiency is 40 times and 11.5 times higher respectively. A network pruning algorithm for CNN is proposed in [7]. It takes a wide network as the input model, and automatically identifies and prunes the unimportant channels in the training process, so as to produce a simplified model with high accuracy. The algorithm is based on the idea of sparsity training, and is suitable for most CNN networks. However, due to less sparsity regularization terms, it still cannot achieve high precision when pruning deeper neural networks.

For the neural network quantization optimization algorithm, there have been a large number of scholars doing the corresponding research. A method to train quantized neural networks (QNNs) is proposed in [8], that is, neural networks only have weights and activation with very low precision during inference process. Low bit quantization makes QNNs greatly reduce the size of memory in the forward inference process, and uses bit operation to replace most arithmetic operations. The 4-bit recurrent neural network quantized by the quantization algorithm is tested on Penn Treebank dataset, and the accuracy is equivalent to that of 32-bit inference process. A quantized CNN architecture is
proposed in [9] to speed up computation and reduce model storage overhead. The filter core of the convolution layer and the weighted matrix of the fully connected layer are quantized to minimize the estimation error of each layer. At the same time, it is pointed out that even the embedded mobile devices can accurately classify the images in one second by using the quantized CNN model. A classical quantization scheme of Google is proposed in [10], and the corresponding training program is designed to maintain the end-to-end model accuracy after quantization. The proposed quantization scheme improves the contradiction between accuracy and device delay.

From the perspective of algorithm principle, CNN-based machine learning target detection and recognition algorithms can be roughly divided into two categories: the first category is region recommendation algorithms; the second category is regression algorithms. Among the series of regression algorithms [11] [12], Yolo v3 algorithm adopts the network structure of Darknet-53, which strengthens the recognition ability of small targets and improves the prediction accuracy on the premise of maintaining the speed advantage.

This paper firstly introduces the Yolo v3 intelligent target detection and recognition algorithm based on machine learning, and then analyzes the network structure, hardware resources, prediction process and deployment process of Yolo v3 algorithm in detail. Secondly, the machine learning optimization technology based on pruning and quantization is introduced. Thirdly, the Amazon Cloud FPGA platform and ZCU104 FPGA hardware platform are introduced. Finally, FPGA accelerator is implemented on two FPGA platforms, and the experimental results are given.

2. YOLO v3 TARGET DETECTION AND RECOGNITION

2.1. Yolo v3 Neural Network Structure

Yolo v3 algorithm is one of the most balanced target detection networks with high speed and high precision. Through the fusion of many advanced methods, Yolo v3 algorithm can make up all the shortcomings of the Yolo series algorithms, including fast speed, not good at detecting small objects, etc. Yolo v3 algorithm has clear structure and good real-time performance, and can get better detection effect for specific scenes. Yolo v3 algorithm uses the residual network structure for reference. Darknet-53 adopts the layer skipping connection mode of ResNet, and the performance is better than ResNet-152 and ResNet-101. Part of the detailed diagram for Yolo v3 network structure is shown in Fig. 1.

![Figure 1. Part of the detailed diagram for Yolo v3 network structure](image)

The convolution form of Yolo v3 neural network is shown in Caffe framework as in Fig. 2. Convolution Layer, Batch Normalization Layer and Leaky Relu Layer together constitute the smallest convolution component.
2.2. MAC Calculation Results of Yolo v3 Network
As shown in Fig. 3, the left part shows MAC calculation result for resolution of 416x416, and the right part shows calculation result for resolution of 608x608. The MAC calculation results reflect the hardware resources utilization of Yolo v3 network.

![Figure 3. The MAC calculation results for resolution of 416x416/608x608](image)

2.3. Bounding Box Prediction Process
The special layer in Yolo v3 algorithm is up-sample layer. The reason is that the deeper the network is, the better the feature expression effect is, and so the deep features are fully used for detection. What’s more, Yolo v3 algorithm outputs three feature maps of different scales, which is called as the predictions across scales. The improved point adopts feature pyramid networks as reference, and uses multi-scales to detect targets with different sizes. The finer the grid cell, the more precise the object can be detected.

The learning calculation formulas are shown as (1). The bounding box prediction map of Yolo v3 network is shown in Fig. 4. In the process of training, the real learning parameters are $t_c$, $t_r$, $t_w$ and $t_h$. $p_w$ and $p_h$ represent the width and height of anchor box mapping to feature graph. $t_c$ and $t_r$ represent the predicted coordinate offset values. $t_w$ and $t_h$ represent the scaling parameters.

$$
\begin{align*}
    b_c &= 1/(1+e^{-c_c}) + c_c \\
    b_r &= 1/(1+e^{-c_r}) + c_r \\
    b_w &= p_w e^{c_w} \\
    b_h &= p_h e^{c_h}
\end{align*}
$$

(1)

![Figure 4. Bounding box prediction map of Yolo v3](image)
2.4. Yolo v3 optimization and deployment process
The optimization and hardware deployment process can be divided into the following steps:

1. For a specific training data set, calculate the mAP of the original network which has not been optimized by pruning and quantization algorithm.
2. The network is optimized by pruning and quantization algorithm to train the optimized network structure and its parameters. Then, calculate the new mAP.
3. The mAP is calculated under the condition of comprehensive implementation of the pruning and quantization algorithms, so that the overall optimized mAP can meet the requirements.
4. Verify the optimized network for embedded hardware deployment to test the inference time. Then, the optimal neural network structure and parameters can be obtained.

3. MACHINE LEARNING OPTIMIZATION TECHNOLOGY

3.1. Pruning Optimization of Double Regular Terms
From the perspective of machine learning theory, L1 norm and L0 norm can achieve network sparsity. L1 norm is widely used because of its better optimization performance than L0 norm. L2 norm can prevent over fitting and improve the generalization ability of the model. Considering the characteristics of all kinds of L norms, this paper proposes a pruning optimization algorithm based on double regular terms. The specific strategy process of the algorithm is described as follows.

Firstly, the BatchNorm layer scaling factors based on L1 regular term and L2 regular term are taken as the optimization constraint regular terms at the same time. In the regularization framework, the regularization terms of two kinds of scaling factors are considered in the objective function. The objective function is expressed as

\[ L = \sum_{i,j} l(f(x_i, W_j), y_i) + \alpha \sum_{y \in \mathcal{Y}} \| \hat{y} \|_1 + \beta \sum_{y \in \mathcal{Y}} \| \hat{y} \|_2 \] (2)

The first term represents the neural network cost function, which represents the loss caused by the model prediction. The second term represents the regular term of L1 scaling factor. The third term represents the regular term of L2 scaling factor. \( \alpha \) and \( \beta \) represent the weight coefficients of the two regular terms in the optimization process. In the training process, the scaling factor parameters are used to measure the importance of all the channels, so as to delete the unimportant channel, compress the network model size, and improve the inference speed.

Secondly, the collaborative computation of training and pruning is carried out, and the optimal pruning network can be obtained by iteration. Through fine-tuning, it can quickly converge to the optimal network. Through repeated iterations, higher compression ratio can be obtained. The flow of pruning and training collaborative computing process is shown in Fig. 5.

![Collaborative computing process](image)

3.2. Int8 Quantization Optimization based on FPGA Computing Architecture
The numerical accuracy of neural network algorithm has a great influence on the occupation of hardware resources. CNN algorithm based on floating-point algorithm has the advantages of high precision and large dynamic range. However, floating-point format is very resource-consuming and it is difficult to realize hardware acceleration, it is feasible to use the quantization optimization algorithm to realize hardware acceleration of CNN algorithm.
The disadvantage of CNN algorithm based on fixed-point format is that its precision is limited, which may lead to quantization error. If the precision of weights decreases too much, the training process will not converge. This paper proposes a low bit training and quantization collaborative optimization algorithm for FPGA computing architecture. The purpose of the algorithm is to ensure that the weight quantization representation can meet the performance design requirements of CNN applications.

In the FPGA computing structure, the processing unit array is used to implement convolution operation of neural network. The processing unit array contains three levels of parallelism, which is described as follows: each unit contains multiple convolution engines, and each convolution engine computes the inner product of convolution in parallel; the convolution data of different convolution engines in each processing unit come from different input channels; the output channels of different convolution kernels are parallel. The structure of each processing unit is shown in Fig. 6.

![Figure 6. Block diagram of FPGA computing structure processing unit](image)

With the quantization optimization algorithm, the hardware structure of multiplier and adder can be simplified, and the fixed-point data with a certain bit width can be used. In order to avoid data overflow, the width of intermediate data can be extended. For 8-bit point design, 24 bit intermediate data can be used. The deviation is aligned with the accumulated data by means of a shifter, and the final result is obtained according to the data quantization results of each layer. The specific selection strategy of the weight quantization range of the proposed training and quantization collaborative algorithm is as follows.

Firstly, the quantization process parameters are calculated as (3). Where, \( s \) represents the weight with the maximum absolute value, \( m \) represents the upper limit of quantization bit width, \( b \) represents the quantization bit width, \( p \) represents the power-of-2 or zero after quantization. \( i \) and \( j \) represent the dimension of convolution kernel. The weight to be quantized is the nearest value in \( \mathbb{P} \), which is the final quantized value. The setting of optimal approximation zero conforms to the sparsity optimization design rule of CNN for FPGA computing structure.

\[
\begin{align*}
\text{s} &= \max \{ \text{abs}(W_i) \} \\
\text{m} &= \log_2(s) \\
\text{n} &= m + 1 - 2^{(n-j)} / 2 \\
\mathbb{P} &= \{ \pm 2^n, \pm 2^{n-1}, \ldots, \pm 2^0 \}
\end{align*}
\]

Secondly, in the training process, the training dataset is divided into groups step by step. Different quantization steps and training parameters are set in the group according to the experience threshold, and a part of the weights are quantified. After each round of quantization, a new round of process is carried out for a part of the weights that has not been quantized, so as to obtain better local quantization results.

Finally, when the internal parameters of each group are updated in back propagation, only the part that has not been quantified is updated. Repeat the first two steps until all parameters in each group are quantified. The quantized weight value can be directly mapped to the hardware shift calculation on FPGA. The quantization method for FPGA computing structure can accelerate the implementation efficiency of CNN on FPGA and obtain higher speedup ratio.
4. DIFFERENT VERIFICATION PLATFORMS BASED ON FPGA COMPUTING ARCHITECTURE

4.1. Amazon Cloud FPGA EC2 Instance Platform

Amazon AWS provides the example of Amazon Elastic Computing Cloud (EC2), including the FPGA computing instance, which can be programmed to create custom hardware acceleration for applications. EC2 is easy to program and is equipped with various resources needed to develop, simulate, debug and compile hardware acceleration code, which including FPGA Developer AMI and hardware development kit.

The characteristic of AWS FPGA is to realize programmable hardware acceleration. With EC2 technology, it is easy to deploy hardware acceleration with FPGA. FPGA Developer AMI includes scripts and tools for FPGA simulating design, compiling code, tools for debugging and compiling code. Users can deploy FPGA Developer AMI directly on EC2 instance and quickly pre-configure the required resources to write and test FPGA design. Amazon AWS EC2 operation example instance is shown in Fig. 7.

![FPGA Developer AMI](image1)

Figure 7. Amazon AWS EC2 operation example instance

4.2. ZCU104 FPGA Hardware Platform

The second FPGA simulation platform selected is ZCU104 platform of Xilinx's Ultrascale + MPSoC series. Based on ZCU104 platform, this chapter verifies the acceleration ability of the neural network compression optimization algorithm proposed and applied on the FPGA platform. The ZCU104 hardware accelerator for intelligent recognition algorithm is shown in Fig. 8.

![ZCU104 FPGA hardware accelerator](image2)

Figure 8. ZCU104 FPGA hardware accelerator

5. EXPERIMENT RESULTS

5.1. Amazon Cloud FPGA EC2 Instance Platform Experiment

The main process of Amazon Cloud FPGA EC2 instance simulation is as follows: firstly, the EC2 instance is established. Secondly, FPGA simulation module is called as shown in Fig. 9. Thirdly, the simulation module is configured to 75MHz as shown in Fig. 10.
The simulation test is carried out under the current configuration. When the weight of image (416x416) is reduced by 60%, after network pruning and INT8 quantization, the inference running time is 15 frames / s.

5.2. ZCU104 FPGA Hardware Platform Experiment

Based on the training data set, including real visible light image and infrared image, the machine learning optimization algorithm of CNN proposed in this paper is verified based on ZCU104 FPGA platform. The visible light data set includes 5700 training sets and 700 test sets; the infrared data set includes 2150 training sets and 350 test sets. The two data sets are respectively 5 categories, including tanks, vehicles, SUVs, armored cars and trucks. Some of the recognition results are shown in the following figures and table.
Figure 12. Optimized Visible light target recognition results on ZCU104 (After pruning with 30%/50% ratio and INT8 quantification)

Figure 13. Optimized Visible light target recognition results on ZCU104 (After pruning with 30%/50% ratio and INT8 quantification)

Figure 14. Infrared image target recognition results on ZCU104 (Before Optimization)

Figure 15. Optimized infrared image target recognition results on ZCU104 (After pruning with 30%/50% ratio and INT8 quantification)
Figure 16. Optimized infrared image target recognition results on ZCU104 (After pruning with 30%/50% ratio and INT8 quantification)

**TABLE I. COMPARISON OF ACCURACY AND SPEED BEFORE AND AFTER OPTIMIZATION ON ZCU104 FPGA PLATFORM**

| Data set            | Pruning and quantification                | mAP       | Frame rate/s |
|---------------------|------------------------------------------|-----------|--------------|
| Visible light Image (416x416) | Original                                 | 0.9105    | 1            |
|                     | Pruning with 10% ratio and INT8 quantification | 0.9123 +0.198% | 6            |
|                     | Pruning with 30% ratio and INT8 quantification | 0.9157 +0.571% | 10           |
|                     | Pruning with 50% ratio and INT8 quantification | 0.9226 +1.329% | 13           |
| Infrared Image (416x416) | Original                                 | 0.9833    | 1            |
|                     | Pruning with 10% ratio and INT8 quantification | 0.9846 +0.132% | 6            |
|                     | Pruning with 30% ratio and INT8 quantification | 0.9818 -0.153% | 10           |
|                     | Pruning with 50% ratio and INT8 quantification | 0.9757 -0.773% | 13           |

The decrease of mAP value indicates that the AP value of the most categories is on a downward trend. After pruning and INT8 quantification optimization, some categories recognize more target categories. The results show that CNN has redundancy, and it can be optimized by machine learning optimization method.

6. CONCLUSION
This paper proposes a machine learning optimization algorithm of CNN based on FPGA architecture, and completes the experiment verification based on Amazon Cloud FPGA EC2 instance platform and ZCU104 FPGA platform. The machine learning optimization algorithm proposed in this paper can easily realize algorithm expansion and reuse, and has great research value. The next research direction is to research more advanced machine learning optimization technology [13]-[19], and better apply the algorithms to FPGA platform.

ACKNOWLEDGMENT
The work was financially supported by Aeronautical Science Foundation of China (201907053005), Aeronautical Science Foundation of China (2019ZC053018). The authors thank the editor and reviewers for their helpful and valuable suggestions.

REFERENCES
[1] G. Wei, Y. Hou, Z. Zhao, Q. Cui, G. Deng and X. Tao, "Demo: FPGA-Cloud Architecture For CNN," 2018 24th Asia-Pacific Conference on Communications (APCC), Ningbo, China, 2018, pp. 7-8, doi: 10.1109/APCC.2018.8633447.
[2] H. Kim and K. Choi, "Low Power FPGA-SoC Design Techniques for CNN-based Object Detection Accelerator," 2019 IEEE 10th Annual Ubiquitous Computing, Electronics &
Mobile Communication Conference (UEMCON), New York City, NY, USA, 2019, pp. 1130-1134, doi: 10.1109/UEMCON47517.2019.8992929.

[3] S. Sağlam, F. Tat and S. Bayar, "FPGA Implementation of CNN Algorithm for Detecting Malaria Diseased Blood Cells," 2019 International Symposium on Advanced Electrical and Communication Technologies (ISAECT), Rome, Italy, 2019, pp. 1-5, doi: 10.1109/ISAECT47714.2019.9069724.

[4] T. Geng, T. Wang, A. Sanaullah, C. Yang, R. Patel and M. Herboird, "A Framework for Acceleration of CNN Training on Deeply-Pipelined FPGA Clusters with Work and Weight Load Balancing," 2018 28th International Conference on Field Programmable Logic and Applications (FPL), Dublin, 2018, pp. 394-3944, doi: 10.1109/FPL.2018.00074.

[5] T. Yang, Y. Chen and V. Sze, "Designing Energy-Efficient Convolutional Neural Networks Using Energy-Aware Pruning," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 6071-6079, doi: 10.1109/CVPR.2017.643.

[6] S. Han, J. L. Kang, H. Z. Mao, et al., "ESE: Efficient Speech Recognition Engine with Sparse LSTM on FPGA," Proceedings of the 2017 ACM/SIGDA International Symposium on Field-Programmable Gate Arrays, 2017: 75-84.

[7] Z. Liu, J. Li, Z. Shen, G. Huang, S. Yan and C. Zhang, "Learning Efficient Convolutional Networks through Network Slimming," 2017 IEEE International Conference on Computer Vision (ICCV), Venice, 2017, pp. 2755-2763, doi: 10.1109/ICCV.2017.298.

[8] S. Jo, H. Park, G. Lee and K. Choi, "Training Neural Networks with Low Precision Dynamic Fixed-Point," 2018 IEEE 36th International Conference on Computer Design (ICCD), Orlando, FL, USA, 2018, pp. 405-408, doi: 10.1109/ICCD.2018.00067.

[9] J. Wu, C. Leng, Y. Wang, Q. Hu and J. Cheng, "Quantized Convolutional Neural Networks for Mobile Devices," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 4820-4828, doi: 10.1109/CVPR.2016.521.

[10] B. Jacob et al., "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, 2018, pp. 2704-2713, doi: 10.1109/CVPR.2018.00286.

[11] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 779-788, doi: 10.1109/CVPR.2016.91.

[12] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 6517-6525, doi: 10.1109/CVPR.2017.690.

[13] T. Yang, Y. Chen and V. Sze, "Designing Energy-Efficient Convolutional Neural Networks Using Energy-Aware Pruning," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, 2017, pp. 6071-6079, doi: 10.1109/CVPR.2017.643.

[14] Z. Wang, S. Lin, J. Xie and Y. Lin, "Pruning Blocks for CNN Compression and Acceleration via Online Ensemble Distillation," in IEEE Access, vol. 7, pp. 175703-175716, 2019, doi: 10.1109/ACCESS.2019.2957203.

[15] S. Ge, Z. Luo, C. Zhang, Y. Hua and D. Tao, "Distilling Channels for Efficient Deep Tracking," in IEEE Transactions on Image Processing, vol. 29, pp. 2610-2621, 2020, doi: 10.1109/TIP.2019.2950508.

[16] Y. Xu, W. Dai, Y. Qi, J. Zou and H. Xiong, "Iterative Deep Neural Network Quantization With Lipschitz Constraint," in IEEE Transactions on Multimedia, vol. 22, no. 7, pp. 1874-1888, July 2020, doi: 10.1109/TMM.2019.2949857.

[17] F. Cardinaux et al., "Iteratively Training Look-Up Tables for Network Quantization," in IEEE Journal of Selected Topics in Signal Processing, vol. 14, no. 4, pp. 860-870, May 2020, doi: 10.1109/JSTSP.2020.3005030.
[18] Y. Choi, M. El-Khamy and J. Lee, "Universal Deep Neural Network Compression," in IEEE Journal of Selected Topics in Signal Processing, vol. 14, no. 4, pp. 715-726, May 2020, doi: 10.1109/JSTSP.2020.2975903.

[19] X. Yan, Y. Fan, K. Chen, X. Yu and X. Zeng, "QNet: An Adaptive Quantization Table Generator Based on Convolutional Neural Network," in IEEE Transactions on Image Processing, vol. 29, pp. 9654-9664, 2020, doi: 10.1109/TIP.2020.3030126.