IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 27 August 2023, accepted 9 October 2023, date of publication 18 October 2023, date of current version 27 October 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3325402

RESEARCH ARTICLE

Complexity-Aware Layer-Wise Mixed-Precision Schemes With SQNR-Based Fast Analysis

HANA KIM[1,2], (Graduate Student Member, IEEE), HYUN EUN[3], JUNG HWAN CHOI[3], AND JI-HOON KIM[1,2], (Senior Member, IEEE)

[1]Department of Electronic and Electrical Engineering, Ewha Womans University, Seoul 03760, Republic of Korea
[2]Graduate Program in Smart Factory, Ewha Womans University, Seoul 03760, Republic of Korea
[3]OPENEDGES Technology, Inc., Seoul 03063, Republic of Korea

Corresponding author: Ji-Hoon Kim (jihoonkim@ewha.ac.kr)

This work was supported by Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (Ministry of Science, ICT) (No. 2020-0-01080, Variable-precision deep learning processor technology for high-speed multiple object tracking) and also supported by the National Research Foundation (NRF), Korea, under project BK21 FOUR.

ABSTRACT Recently, deep neural network (DNN) acceleration has been critical for hardware systems from mobile/edge devices to high-performance data centers. Especially, for on-device AI, there have been many studies on hardware numerical precision reduction considering the limited hardware resources of mobile/edge devices. Although layer-wise mixed-precision leads to computational complexity reduction, it is not straightforward to find a well-balanced layer-wise precision scheme since it takes a long time to determine the optimal precision for each layer due to the repetitive experiments and the model accuracy, the fundamental measure of deep learning quality, should be considered as well. In this paper, we propose the layer-wise mixed precision scheme which can significantly reduce the time required to determine the optimal hardware numerical precision with Signal-to-Quantization Noise Ratio (SQNR)-based analysis. In addition, the proposed scheme can take the hardware complexity into consideration in terms of the number of operations (OPs) or weight memory requirement of each layer. The proposed method can be directly applied to inference, meaning that users can utilize well-trained neural network models without the need for additional training or hardware units. With the proposed SQNR-based analysis, for SSDlite and YOLOv2 networks, the analysis time required for layer-wise precision determination is reduced by more than 95% compared to conventional mean Average Precision(mAP)-based analysis. Also, with the proposed complexity-aware schemes, the number of OPs and weight memory requirement can be reduced by up to 86.14% and 78.03%, respectively, for SSDlite, and by up to 51.93% and 50.62%, respectively, for YOLOv2, with negligible model accuracy degradation.

INDEX TERMS Deep neural network (DNN), mixed-precision, signal to quantization noise ratio (SQNR), complexity-awareness.

I. INTRODUCTION

Recently, Artificial Intelligence (AI) is considered a powerful technique in the 4th industrial revolution and its various applications, such as object recognition [1], [2], [3], image classification [4], [5], voice recognition [6], [7], are widely used in emerging fields including manufacturing and autonomous driving. The complexity of AI Algorithms is significantly increasing to improve the accuracy of AI services, which leads to a growing demand for the hardware accelerator to process DNNs efficiently. Although DNN acceleration becomes necessary in mobile/edge devices as well as in high-performance data centers, the performance of the DNN model at mobile/edge devices is inferior due to its limited hardware resources [8], [9], [10], [11]. Accordingly, research on lightweight DNN that has a small model size

The associate editor coordinating the review of this manuscript and approving it for publication was Yue Zhang.

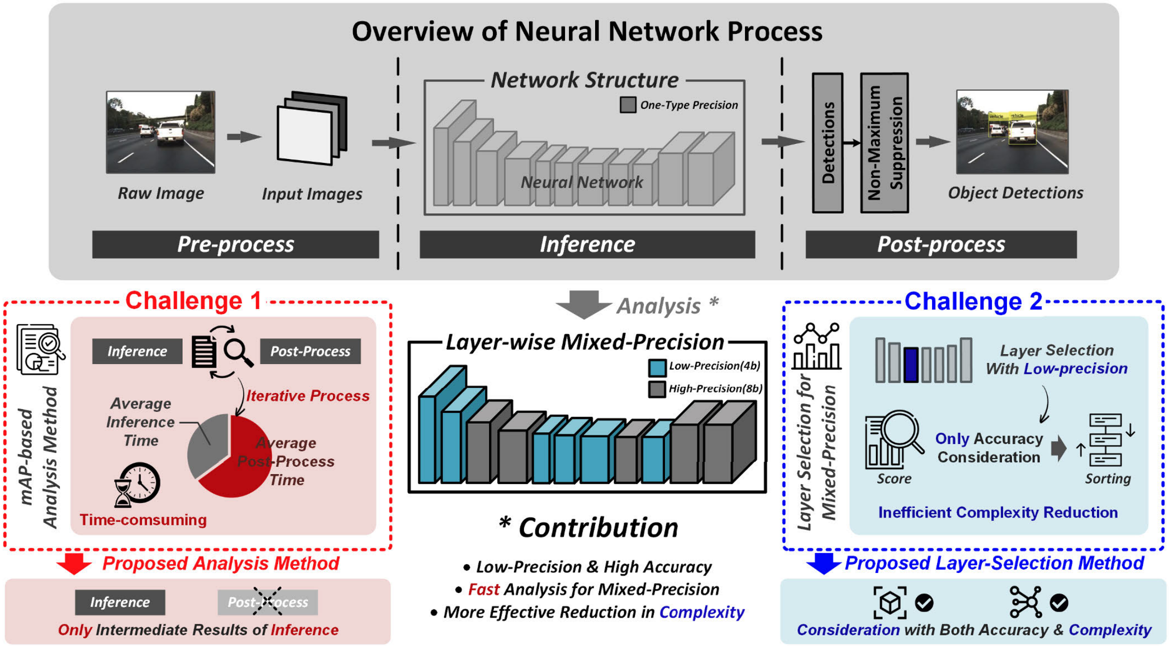

Overview of Neural Network Process
Raw Image
Input Images
Pre-process
Network Structure
One-Type Precision
Neural Network
Inference
Detections
Non-Maximum Suppression
Object Detections
Post-process
Challenge 1
mAP-based Analysis Method
Inference
Post-Process
Iterative Process
Average Inference Time
Average Post-Process Time
Time-comsuming
Proposed Analysis Method
Inference
Post-Process
Only Intermediate Results of Inference
Analysis *
Layer-wise Mixed-Precision
Low-Precision(4b)
High-Precision(8b)
* Contribution
• Low-Precision & High Accuracy
• Fast Analysis for Mixed-Precision
• More Effective Reduction in Complexity
Challenge 2
Layer Selection for Mixed-Precision
Layer Selection With Low-precision
Only Accuracy Consideration
Score
Sorting
Inefficient Complexity Reduction
Proposed Layer-Selection Method
Consideration with Both Accuracy & Complexity

FIGURE 1. Overview of neural network process and contribution of the proposed method.

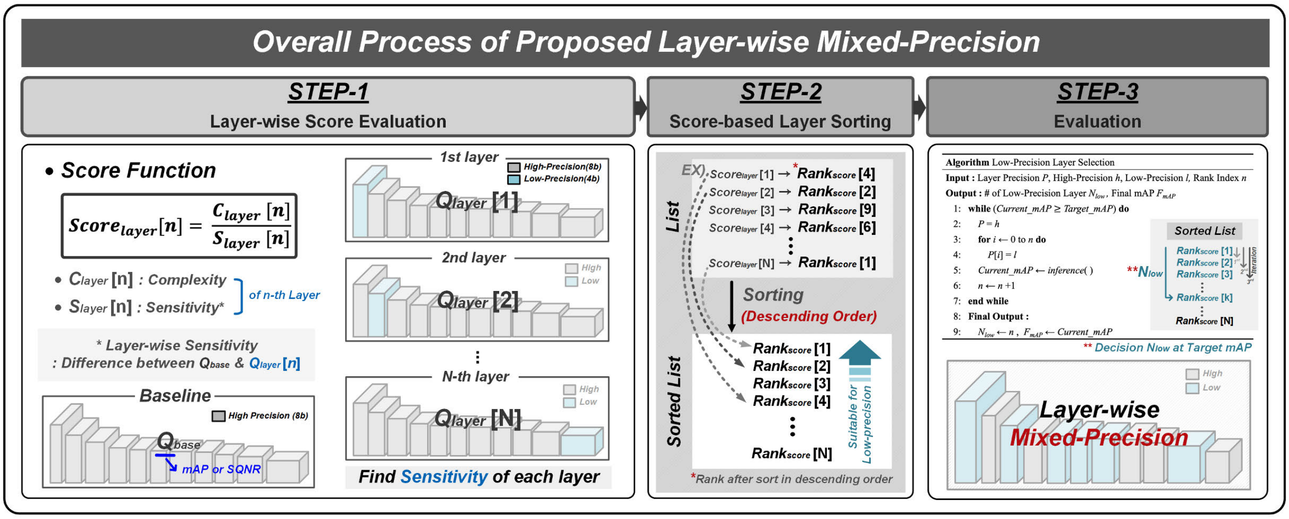
and reduced computational requirement is being actively conducted while maintaining the accuracy of the trained DNN model [12], [13], [14].

For efficient DNN acceleration without model accuracy degradation, there have been many studies regarding lightweight deep learning techniques such as network pruning [15], [16], clustering [17], [18], knowledge distillation [19], [20] and hardware optimization [21], [22]. These techniques can be considered for lightweight DNN processing, but it is not easy to apply in practical applications due to their irregularities and dependencies on the neural network [23]. On the other hand, a quantization technique has been proposed as the simplest and most powerful method of lightweight DNN. AdaRound is developed where the rounding is applied to minimize the change in output activation caused by quantization [24]. For further reduction in bandwidth or computation complexity, IBM has proposed a 4-bit quantization scheme, ACIQ [25]. Data-Free Quantization is also possible to create an 8-bit model without fine tuning the data as accurately as 32-bit model [26]. Aside from this, research to create low-bit precision using quantization is being conducted [27].

In DNN acceleration, with these low-precision number representation, it is possible to reduce computational resources and memory requirements. However, if aggressive uniform quantization with low precision is applied to all layers, it results in degradation of model accuracy, the fundamental measure of deep learning quality. To overcome these limitations, mixed precision-based quantization has been proposed which enables aggressive model compression and provides tradeoff between the model accuracy and the hardware complexity for DNN acceleration [28], [29].

As the schemes for mixed-precision, Mixed-Precision Quantization(MPQ) search problem is proposed. To avoid the iterative search and reduces search time, the proposed method is a joint training scheme that use one-time integer linear programming(ILP) problem [30]. Progressively decreasing bitwidth, which reduces weight bitwidth by going back to layer, is also proposed as a mixed-precision scheme [31]. Hessian AWare Quantization (HAWQ), which has an automatic selection of relative quantization precision of each layer based on the layer's Hessian spectrum, is proposed [32]. Bit-Mixer, Hardware-aware Automated Quantization (HAQ), etc. are also proposed [33], [34], [35]. In addition, research on mixed-precision with other data type, not integer type, are also proposed. These studies usually use Floating-Point(FP) for reducing the accuracy drops [36], [37], [38].

However, in the previous studies, the proposed schemes such as [30], [31], [32], [33], [34], and [35] of related works are applied in the training process. These schemes need separate schemes or retraining. Otherwise, they have additional hardware overhead to apply special schemes. These problems make it difficult to apply in the actual industry. On the other hand, schemes using Floating-Point for mixed-precision such as [36], [37], and [38] have more hardware overhead compared with an Integer type. In addition, the determination of optimal layer-wise precision is considerably time-consuming. Fig. 1 illustrates the use of mAP in traditional analysis, which necessitates time-consuming post-processing based on Non-Maximum Suppression (NMS). The top of the figure shows an overview


Overall Process of Proposed Layer-wise Mixed-Precision
STEP-1
Layer-wise Score Evaluation
STEP-2
Score-based Layer Sorting
STEP-3
Evaluation
• Score Function
Score_layer[n] = C_layer [n] / S_layer [n]
• C_layer [n] : Complexity
• S_layer [n] : Sensitivity*
of n-th Layer
* Layer-wise Sensitivity
: Difference between Q_base & Q_layer [n]
Baseline
High Precision (8b)
Q_base
mAP or SQNR
1st layer
High-Precision(8b)
Low-Precision(4b)
Q_layer [1]
2nd layer
High
Low
Q_layer [2]
N-th layer
Q_layer [N]
Find Sensitivity of each layer
List
EX)
Score_layer [1] → *Rank_score [4]
Score_layer [2] → Rank_score [2]
Score_layer [3] → Rank_score [9]
Score_layer [4] → Rank_score [6]
Score_layer [N] → Rank_score [1]
Sorting
(Descending Order)
Sorted List
Rank_score [1]
Rank_score [2]
Rank_score [3]
Rank_score [4]
Rank_score [N]
Suitable for Low-precision
*Rank after sort in descending order
Sorted List
**N_low
Rank_score [1]
Rank_score [2]
Rank_score [3]
Rank_score [k]
Rank_score [N]
1st
2nd
3rd
Iteration
** Decision N_low at Target mAP
High
Low
Layer-wise
Mixed-Precision

```
Algorithm Low-Precision Layer Selection
Input : Layer Precision $P$, High-Precision $h$, Low-Precision $l$, Rank Index $n$
Output : # of Low-Precision Layer $N_{low}$, Final mAP $F_{mAP}$
while ($Current\_mAP \geq Target\_mAP$) do
    $P = h$
    for $i \leftarrow 0$ to $n$ do
        $P[i] = l$
    $Current\_mAP \leftarrow inference()$
    $n \leftarrow n + 1$
end while
Final Output :
    $N_{low} \leftarrow n$, $F_{mAP} \leftarrow Current\_mAP$
```

FIGURE 2. Overall process of proposed layer-wise mixed-precision scheme.

of the conventional neural network process, which consists of pre-process, inference, and post-process. Although there have been significant improvements in the inference process due to algorithmic and hardware optimizations, relatively little research has been conducted on accelerating post-processing [39], [40], [41], [42], [43], [44]. Furthermore, it is necessary to perform the evaluation process iteratively for each layer in the network, repeating it as many times as the number of layers. Additionally, to prevent any degradation in model accuracy, it is imperative to confirm the inference result (mAP) at each iteration, thereby imposing a significant computational burden that results in extended analysis runtime. If only the accuracy is considered in determining layer-wise mixed-precision, it does not adequately address the hardware complexity or memory requirements, which are the most important concerns in mobile and edge devices [29].

In this paper, we propose layer-wise mixed-precision scheme for effective application of uniform quantization. As shown in Fig. 1, the bottom of the figure illustrates two challenges of the analysis method for mixed-precision, which are the time-consuming mAP-based analysis method and only accuracy consideration for mixed-precision consideration. And the center of the figure shows the final layer-wise mixed-precision results and this paper's contributions. We can apply different precision (4-bit / 8-bit) for each layer while maintaining model accuracy. Although we use 4-bit precision for low-precision and 8-bit precision for high-precision in our study, this can be fixed for all applications or can be configured differently to achieve the desired performance in certain applications. With the modified Signal-to-Quantization Noise Ratio (SQNR)-based analysis, the proposed analysis scheme uses only intermediate results of inference, which is the output activation, without time-consuming post-processing. Consequently, the proposed scheme provides much faster analysis compared to the conventional mAP-based approach, while also reducing hardware complexity through a complexity-aware scheme. Furthermore, the proposed schemes can be readily applied to pre-trained open-source neural network models without the need for additional training or hardware units. This brief is organized as follows. Section II presents the proposed layer-wise mixed-precision scheme with fast analysis and hardware complexity consideration. Experimental results for SSDlite and YOLOv2 networks are provided in Section III, and Section IV concludes this brief.

II. PROPOSED LAYER-WISE MIXED-PRECISION SCHEME

A. OVERALL LAYER-SELECT ALGORITHM FOR MIXED PRECISION

The overall process of the proposed layer-wise mixed-precision scheme is illustrated in Fig. 2 where several layers are chosen to be with low-precision (4-bits) and others are with high-precision (8-bits). The layer-wise mixed-precision is composed by using our proposed analysis scheme. For an optimized result from the proposed layer-wise mixed-precision, it is important to determine which layers can be processed with low-precision or high-precision. The following three steps summarize how to determine the layers with low precision:

STEP-1: For each layer, perform the evaluation of the layer sensitivity to low-precision with the complexity consideration. Through the iterative process, the sensitivity of each layer is calculated. In this step, we use our proposed score function, which considers both layer sensitivity and complexity. Also, the difference in the proposed *SQNR* values is used as a layer-wise sensitivity factor.

STEP-2: Sorting the layers according to the results from STEP-1. The list is composed of the layer-wise score. The scores are sorted in descending order. In the sorted list, the higher-ranking layers are more suitable for low-precision.

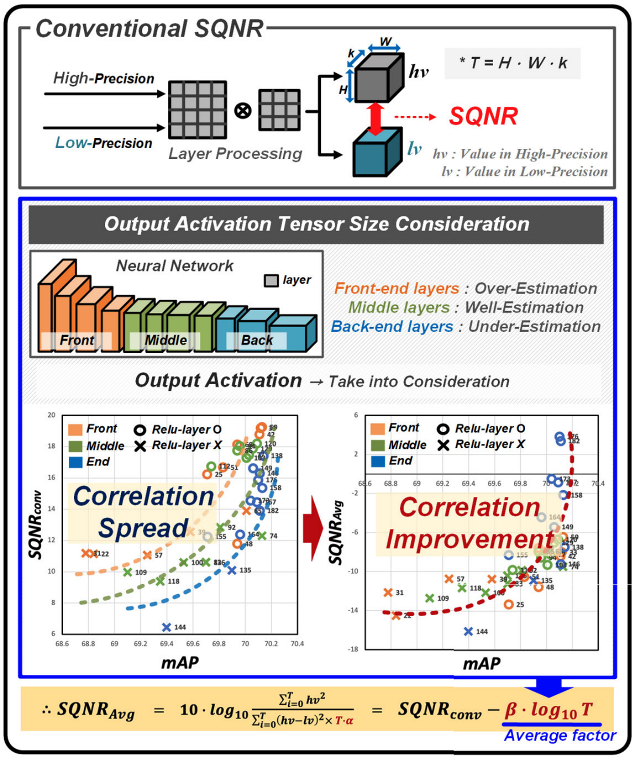

Conventional SQNR
High-Precision
Low-Precision
Layer Processing
W
k
H
hv
lv
* T = H · W · k
SQNR
hv : Value in High-Precision
lv : Value in Low-Precision
Output Activation Tensor Size Consideration
Neural Network
layer
Front
Middle
Back
Front-end layers : Over-Estimation
Middle layers : Well-Estimation
Back-end layers : Under-Estimation
Output Activation → Take into Consideration
Front
Middle
End
Relu-layer O
Relu-layer X
Correlation Spread
SQNRconv
mAP
Correlation Improvement
SQNRAvg
mAP
∴ SQNR_Avg = 10 · log10 (Σ_{i=0}^T hv²)/(Σ_{i=0}^T (hv−lv)² × T·α) = SQNR_conv − β · log10 T
Average factor

FIGURE 3. Proposed SQNR-based sensitivity estimation.

STEP-3: With the low-precision layer selection algorithm, find the low-precision layers progressively while keeping the model accuracy better than target lower-bound *mAP*. If the network accuracy or complexity reaches the target value, the layer-wise mixed-precision is configured with the result of layer selection.

B. FAST ANALYSIS WITH THE PROPOSED $SQNR_{AVG}$

In STEP-1 of the proposed scheme, the layer sensitivity to low-precision of each layer, S_{layer}, can be evaluated with the quality factor, Q, which are either *mAP* or *SQNR*. Due to analyze the effects of low-precision compared to high-precision, the baseline quality factor, Q_{base}, is set to the scenario where all layers are in high-precision. For each layer, low-precision (4-bit) is applied and the quality factor, $Q_{layer}[n]$ for n-th layer, is evaluated. Accordingly, the S_{layer} is determined by the following (1).

$$S_{layer}[n] = Q_{base} - Q_{layer}[n] \tag{1}$$

When *mAP* is used as the quality factor(Q), the layer-wise mixed-precision scheme can provide good overall results in terms of accuracy. However, it is too time-consuming due to its n times iterative process and the number of computations to obtain *mAP* values. The deeper the neural network, the longer the analysis time to find optimal layer-wise mixed-precision takes much longer. Therefore, as shown in Fig. 3, we propose a method of measuring the sensitivity by using the *SQNR*. Generally, *SQNR* is "Signal to Quantization Noise Ratio", which means the ratio between the original signal and noise of applying quantization. In this instance, the quality factor, Q, is defined as *SQNR*. The layer sensitivity, S_{layer}, is determined by subtracting $Q_{layer}[n]$ (which is the *SQNR* when low-precision is applied in the n-th layer) from Q_{base} (which is the *SQNR* when high-precision is applied in each layer). As expressed in (2), we use $SQNR_{dB}$ as layer quality values. fpv is the value in 32-bit floating point mode and qv is the value in quantization mode. But, in this paper, we want to compare the effects of applying quantization with high-precision or low-precision. Therefore, the proposed $SQNR_{conv}$ is defined with the output activations in high-precision mode and low-precision mode as expressed in (3). The top portion of the figure depicts the conventional *SQNR* determined by equation (3).

$$\begin{aligned} SQNR_{dB} &= 10 \cdot log_{10} \frac{P_{signal}}{P_{noise}} \\ &= 10 \cdot log_{10} \frac{\sum fpv}{\sum (fpv - qv)^2} \end{aligned} \tag{2}$$

$$SQNR_{conv} = 10 \cdot log_{10} \frac{\sum hv^2}{\sum (hv - lv)^2} \tag{3}$$

where hv is the value in high-precision mode and lv is the value in low-precision mode.

SQNR-based analysis has much lower computation than *mAP*-based analysis. The *SQNR*-based analysis is obtained by each layer during only one inference of neural network whereas *mAP*-based analysis uses accuracy which is obtained by inference of the whole neural network. Therefore, each layer's *mAP* has to be obtained by total n times inference. Although the computation of $SQNR_{conv}$ is much simpler than that of *mAP*, the use of $SQNR_{conv}$ as a quality factor does not show consistent correlations between layers with *mAP*-based approach as shown in Fig. 3, which leads to suboptimal layer-wise mixed-precision results. For the front-end layers, $SQNR_{conv}$ was measured to be relatively higher than that of *mAP* (Over-Estimation), and in the case of the back-end layers, *mAP* was measured to be higher than $SQNR_{conv}$ (Under-Estimation).

In order to improve the correlation with conventional *mAP*-based approach, we propose the $SQNR_{Avg}$ where the magnitude of the output activation, T, is considered in (4).

$$\begin{aligned} SQNR_{Avg} &= 10 \cdot log_{10} \frac{\sum hv^2}{\sum (hv - lv)^2 \cdot T \cdot \alpha} \\ &= 10 \cdot log_{10} \frac{\sum hv^2}{\sum (hv - lv)^2} - \beta log_{10} T \\ &= SQNR_{conv} - \beta \cdot log_{10} T \end{aligned} \tag{4}$$

where α and β are the parameters that adjust the quantity of output activation. Those two parameters, alpha and beta, can be empirically determined between 1 to 10 to ensure the correlation improvement between *mAP* and the proposed $SQNR_{Avg}$. Although the parameters are empirically determined, the proposed analysis time is much smaller than the conventional *mAP*-based method and the total

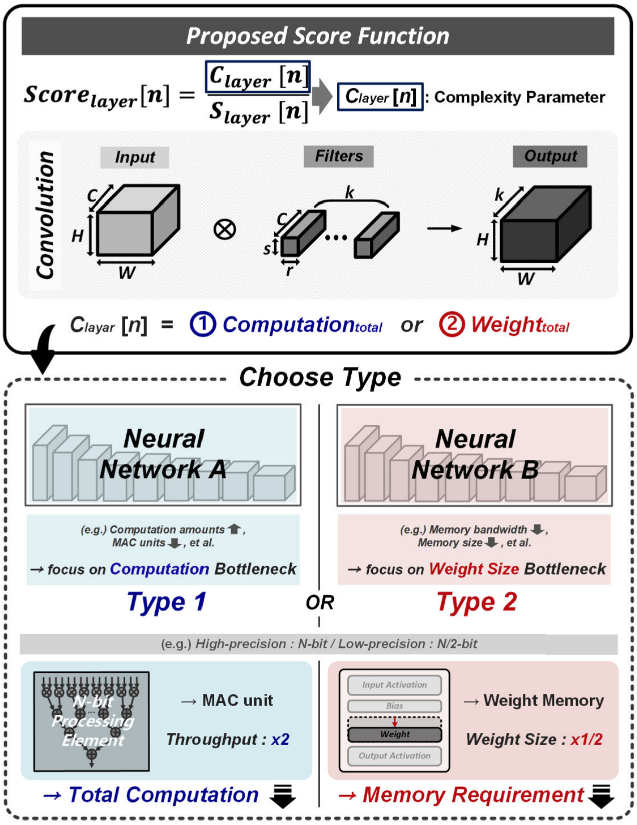

Proposed Score Function
$Score_{layer}[n] = \frac{\boxed{C_{layer}[n]}}{S_{layer}[n]}$
$\boxed{C_{layer}[n]}$: Complexity Parameter
Convolution
Input
Filters
Output
C
H
W
k
s
r
k
H
W
$C_{layar}[n]$ = ① $Computation_{total}$ or ② $Weight_{total}$
Choose Type
Neural Network A
(e.g.) Computation amounts ⬆, MAC units ⬇, et al.
→ focus on Computation Bottleneck
Type 1
OR
Neural Network B
(e.g.) Memory bandwidth ⬇, Memory size ⬇, et al.
→ focus on Weight Size Bottleneck
Type 2
(e.g.) High-precision : N-bit / Low-precision : N/2-bit
N-bit Processing Element
→ MAC unit
Throughput : x2
Input Activation
Bias
Weight
Output Activation
→ Weight Memory
Weight Size : x1/2
→ Total Computation ⬇
→ Memory Requirement ⬇

FIGURE 4. Proposed neural network complexity consideration.

time for determining β is about a few minutes. In Fig. 3, the blue box highlights the *SQNR* with output activation consideration and the impact of the proposed $SQNR_{Avg}$. The output activation consideration is shown as the average factor of $SQNR_{Avg}$ as shown in yellow box of the figure. As illustrated in the figure, the proposed $SQNR_{Avg}$ demonstrates a good correlation with *mAP* compared to the conventional $SQNR_{conv}$ method. Furthermore, the analysis time required for the proposed layer-wise mixed-precision scheme can be significantly reduced when compared to the traditional *mAP*-based approach. This is due to the fact that the proposed analysis scheme utilizes only the intermediate results of inference, namely the output activation.

C. COMPLEXITY CONSIDERATION FOR EACH LAYER

In order to reduce the hardware complexity, which is critical especially in mobile/edge devices, the proposed layer-wise mixed-precision scheme can consider the hardware complexity in $Score_{layer}[n]$ evaluation in Fig. 2. The top of Fig. 4 shows the proposed score function for mixed-precision, which takes into account the layer-wise complexity. The complexity of neural networks refers to the extent of intricacy in processing inference, which is affected by the operation method of layers and the amount of data. There are two types of complexity considerations and the type of complexity is determined based on the network bottleneck: computation bottleneck and weight size bottleneck. Mixed-precision involves using low-precision in some layers to reduce the complexity in computation and weight size. Since both complexity and accuracy are critical for neural networks, the value of mixed-precision comes from effectively managing the trade-off between reducing complexity and maintaining accuracy. To achieve the optimal trade-off, it is more effective to select specific layers for low-precision rather than uniformly applying to all layers. Although applying low-precision to certain layers can reduce both the computational intensity and weight size requirement, the primary bottleneck source in these networks varies based on conditions specific to the neural network, including computation types, layer configurations, memory bandwidth, and other factors. For instance, when the neural network has substantial computational requirements, the computation bottleneck should be resolved for better inference speed. Likewise, for the given memory bandwidth, excessive weight sizes increase the weight preparation time required for fetching weight values from DRAM to the internal buffer before computation. In such cases, the weight size reduction should have a higher priority in optimization consideration. In the proposed mixed-precision score function, we can choose which complexity parameter is more considered when selecting the low-precision layers for the given neural network conditions. Fig. 4 depicts *Type 1* (in blue color) and *Type 2* (in red color) to illustrate the purposes and effects of each complexity type, respectively, where the complexity parameter of n-th layer, $C_{layer}[n]$, can be set according to the optimization goals: *Type 1* is mainly for computational complexity reduction and *Type 2* is defined to reduce the memory requirement for weight values where the total amount of computation and the total weights size defined in (5) and (6), respectively, can be used the complexity parameter.

$$Computation_{total} = k \cdot r \cdot s \cdot C \cdot W \cdot H \tag{5}$$

$$Weight_{total} = k \cdot r \cdot s \cdot C \tag{6}$$

where k is the number of weight kernels, the width and height of the kernel are r and s, respectively, C is the number of input activation channels, and the width and height of output activation are W and H. The total computation means the computation amount required for convolution calculation, where the computation amount is the number of operation cycles in the computation unit. In other words, it is the number of calculations until the convolution of the one layer is finished. As illustrated in Fig. 4, the computational complexity can be decreased with the low-precision operations although the total number of operations is the same. Also, by applying the multi-mode Multiply-Accumulate (MAC) unit which can support multiple precisions, the overall throughput in neural network inference can be increased since multiple low-precision operations can execute in parallel [45], [46], [47]. On the other hand, the total weight size means the memory requirements for storing the weight parameters.

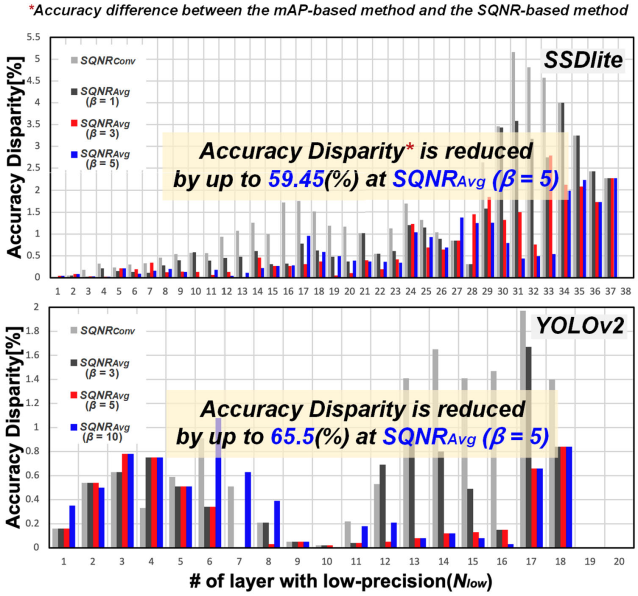


FIGURE 5. Accuracy disparity of $SQNR_{Conv}$ and $SQNR_{Avg}$.

TABLE 1. Correlation ratio of $SQNR_{CONV}$ and $SQNR_{AVG}$.

Network	SSDlite				YOLOv2			
SQNR Type	$SQNR_{Conv}$	$SQNR_{Avg}$			$SQNR_{Conv}$	$SQNR_{Avg}$		
		β=1	β=3	β=5		β=3	β=5	β=10
Average Disparity[a][%]	1.46	1.05	0.67	**0.59**	0.70	0.44	**0.24**	0.37

[a] Average accuracy difference between the mAP-based method(*Baseline*) and the proposed SQNR-based method.

The total amounts of filters in the figure is the required total weight size. If the low-precision is applied for the weights, the memory requirements for weights are reduced. If the target network has computation bottleneck, *Type 1* can be chosen, and the number of Ops can be reduced with the proposed layer-wise mixed-precision scheme while maintaining model accuracy. Whereas, if the target network has weight size bottleneck, *Type 2* can be chosen, and the proposed layer-wise precision scheme leads to the memory requirement reduction. In other words, by taking the complexity into account for the score that decides the layers to apply low-precision, much more complexity reduction is possible.

III. EXPERIMENTAL RESULTS

The proposed complexity-aware layer-wise mixed-precision schemes are applied to two object detection model, SSDlite network where MobileNet-v2 is applied as backbone to SSD network, and YOLOv2 model and evaluated using Pascal-VOC dataset with uniform quantization. Also, per channel quantization is applied to weights, and per layer quantization is applied to activations.

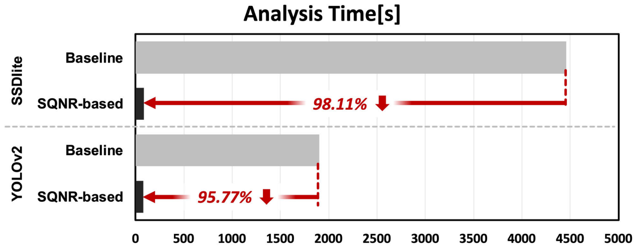


FIGURE 6. Analysis time with the proposed scheme.

TABLE 2. Analysis time comparison.

Network	SSDlite		YOLOv2	
Method	Baseline	SQNR-based	Baseline	SQNR-based
Analysis Time [s]	4456.7 (100%)[a]	84.41 **(1.89%)**[b]	1901.6 (100%)	80.42 **(4.23%)**

[a] Baseline for the comparison with the SQNR-based method.

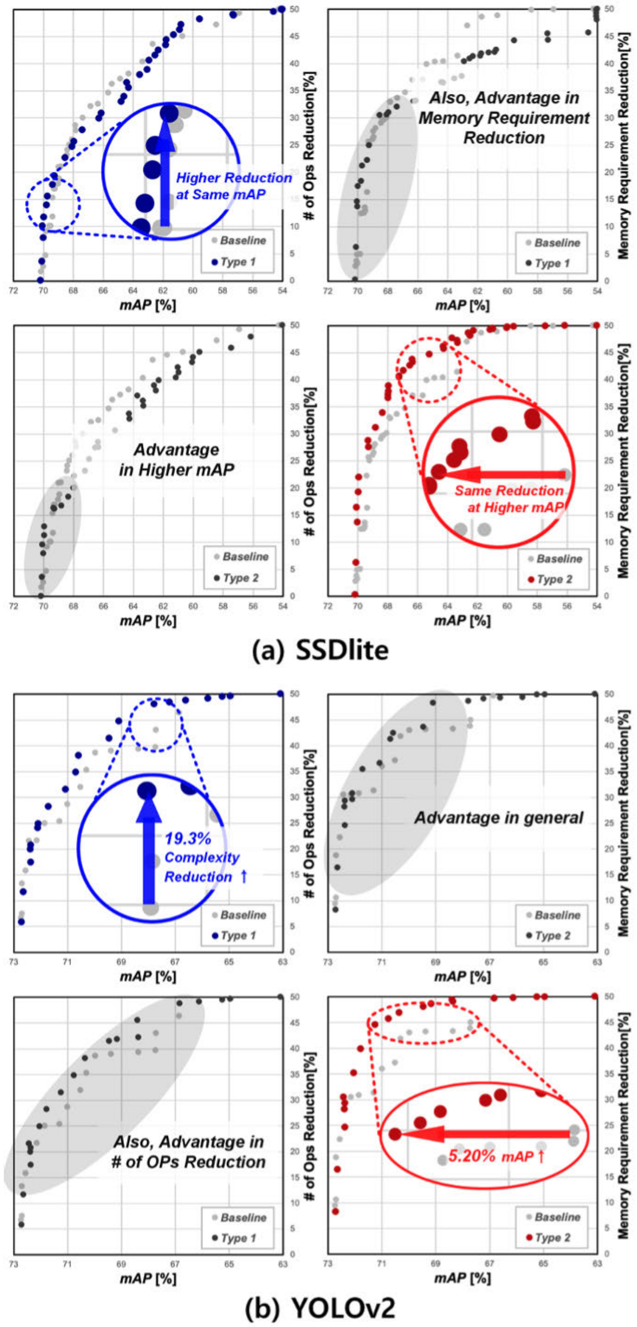
[b] Reduction results of the proposed method compared to baseline.

*OS: Ubuntu 18.04/ CPU: Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz/ DRAM: 32GB/ GPU: RTX-2060, 6GB GDDR6.

A. EFFECTIVENESS OF PROPOSED $SQNR_{AVG}$

In this study, mean Average Precision(*mAP*) provided by the object detection network is used as an indicator for accuracy evaluation. Fig. 5 shows the accuracy disparity of *SQNR*-based scheme, which is the degree of accuracy degradation compared to *mAP*-based scheme. In other words, the accuracy disparity is calculated as the difference in accuracy(*mAP*[%]) between the *mAP*-based analysis and the *SQNR*-based scheme. The figure displays the impact of β on $SQNR_{Avg}$ for two object detection models, SSDlite and YOLOv2. The yellow box indicates the lowest accuracy disparity for each model. As shown in Section II-B, with the empirically determined constant, $SQNR_{Avg}$ shows small accuracy disparity compared to $SQNR_{conv}$ due to the consideration of output activation size. Table 1 denotes the average disparity according to different β values for SSDlite and YOLOv2 networks. The more the average disparity smaller, the more similar results to *mAP*. $SQNR_{conv}$ shows the largest average disparity. In the case of the $SQNR_{Avg}$, the average disparity can be reduced from 1.46% of $SQNR_{conv}$ to 0.59% in SSDlite networks. In the YOLOv2 network, it can be reduced from 0.70% to 0.24%. As a result, β is 5 in both SSDlite and YOLOv2 networks.

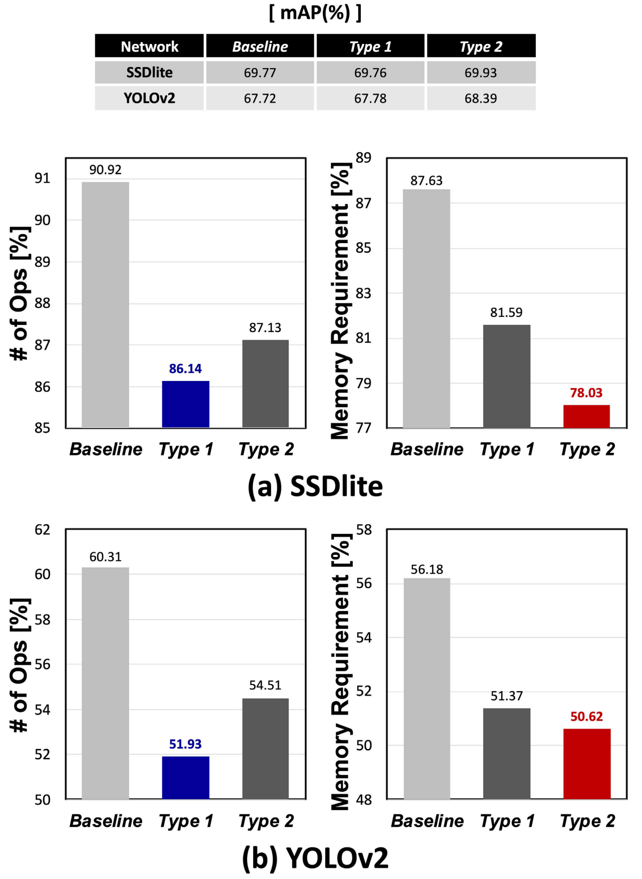
Fig. 6 show the analysis runtime comparison between the proposed $SQNR_{Avg}$-based scheme and the conventional *mAP*-based scheme, *Baseline*, described in Fig. 2. The analysis time is measured in SSDlite and YOLOv2. The experiment environment and details of the analysis time can be found in Table 2. In this study, analysis runtime is measured in the experiment environment where 16.04 Ubuntu was used as operating system, with an Intel(R) Core(TM) i7-8750H CPU, 32GB DRAM and RTX-2060 6GB GDDR6. The analysis



(a) SSDlite

(b) YOLOv2

FIGURE 7. Complexity reduction and accuracy comparison between baseline and proposed scheme for (a) SSDlite network and (b) YOLOv2 network.


[mAP(%)]

Network	Baseline	Type 1	Type 2
SSDlite	69.77	69.76	69.93
YOLOv2	67.72	67.78	68.39

(a) SSDlite

(b) YOLOv2

FIGURE 8. Complexity reduction comparison for each type.

runtime is the time interval between the start the inference and the completion of the computation, which includes task such as inference in *mAP*-based analysis or convolution in *SQNR*-based analysis. The reduction ratio in Table 2 means how much the analysis time is reduced through the use of *SQNR*-based analysis compared to *mAP*-based analysis. For SSDlite network, *mAP*-based method takes 4456.7s whereas proposed scheme takes only 84.41s with 98.11% reduction. Also, for YOLOv2 network, it shows 95.77% reduction in analysis time similarly. If this method is applied to a different new neural network model, the optimal beta can be determined by evaluating *mAP* accuracy, computation reduction, or weight memory reduction for several possible values ranging from 1 to 10. As the analysis time for a given beta value is still very fast similar to Table 2, within 5% of the time compared to the existing method, it has the advantage of still being very fast.

B. EFFECTIVENESS OF COMPLEXITY-AWARENESS

In order to evaluate the effectiveness of the complexity consideration in the proposed scheme, Fig. 7 and Fig. 8 shows that the proposed scheme can reduce the hardware complexity according to the target optimization: *Type 1* for the number of Ops and *Type 2* for the weight memory requirement, where baseline indicates all layers are in high-precision mode and conventional *mAP*-based scheme is considered without complexity-awareness. At this experiment process,the amount of complexity reduction considers the total quantity of parameters applied with low-precision or high-precision. These factors are evaluated through (5) and (6) in Section II-C. As illustrated in Fig. 7 and Fig. 8, itBaseline, represent the results with high-precision whereas *Type 1 or Type 2* show the results with target optimization. Fig. 7

TABLE 3. Comparison with prior works using mixed-precision.

Type	[28]	[29]	[31]	This work	
Data Set	COCO	NWPU-RESISC45	Pascal-VOC	Pascal-VOC	
Training	Yes	Yes	Yes	**No**	
Quantization Type	N/A[a]	Nonlinear	linear	**Uniform**	
Additional Scheme[b]	Retrain	NAS-based	Tanh(.) function	**UNNECESSARY**	
Network	YOLOv3	ResNet-34	SSD-300	SSDlite	YOLOv2
Precision[c] [bits]	1.412	5	1.22	4.74	4.43
Difference of *mAP*[d] [%]	-0.46	+0.16	-12.89	-4.16	-1.90
Compression Ratio	×22.66	×6.46	×26.22	×6.75	×7.22

[a] Unknown information is indicated by N/A.

[b] The additional scheme required for target quantization of the model.

[c] Precision of weights.

[d] Accuracy difference from the baseline using 32-bit floating point.

shows the evaluation results with all proposed schemes including $SQNR_{Avg}$-based fast analysis and the complexity consideration. In the figure, itType 1 has a higher complexity reduction at the same model accuracy, *mAP* result, compared to the baseline for both SSDlite and YOLOv2 networks. For the case of the same computation reduction, *Type 1* shows higher *mAP* results compared to the baseline. Similarly, *Type 2* shows better model accuracy for the same weight size compared to the baseline. Also, for the same model accuracy, the proposed scheme with *Type 2* can reduce the memory requirement significantly, for both SSDlite and YOLOv2 networks. Fig. 8 presents the best performance of results shown in Fig. 7, demonstrating performance comparison of complexity reduction while maintaining the same accuracy. For the comparison, we define the complexity of the network composed only of high-precision layers as 100%. And the results show the percentage of reduced complexity achieved by applying mixed-precision. Specifically, compared to the *Baseline*, the number of operations of *Type 1* is reduced from 90.92% to 86.14% in SSDlite and from 60.31% to 51.93% in YOLOv2. Additionally, the memory requirement of *Type 2* is reduced from 87.63% to 78.03% and from 56.18% to 50.62% for each network, respectively. The *mAP*(%) results for each type are displayed at the top of Fig. 8. The accuracy is similar across all types, but the proposed method shows a greater reduction in complexity compared to the conventional method, *Baseline*. Therefore, the proposed complexity-aware layer-wise mixed-precision scheme shows better model accuracy at the same complexity requirement or provides intended complexity reduction according to the optimization target at the same target model accuracy. In our proposed method, the degree of optimization can slightly affect the level of accuracy and complexity reduction within a certain range. However, our method consistently shows good results in all aspects of accuracy and complexity reduction. Therefore, our approach demonstrates the ability to select layers that achieve complexity reduction while minimizing accuracy degradation.

C. COMPARISON WITH PRIOR WORKS

As denoted in Table 3, the proposed work is compared with other papers using mixed-precisions. In this work, a uniform quantization type is used and any additional quantization scheme isn't required. The final precision, accuracy drops, and compression ratio are shown in the table. The simulation network includes 38 layers for SSDlite and 20 layers for YOLOv2. In the end, 4-bit precision is used in 21 out of the 38 layers of the SSDlite network, resulting in a precision level of 4.93 bits. In a similar fashion, low precision is used in 10 out of the 20 layers of the YOLOv2 network, resulting in a precision level of 4.37 bits. In other papers, training and special quantization methods are basically necessary to configure the network in low-precision. Additional schemes for quantization incurs hardware implementation overhead. Also, despite the application of a special scheme, the model accuracy degradation was greater than the proposed model as shown in comparison with [31]. In compared with [29], the compression ratio is smaller than in this work. Although the accuracy degradation and comparison ratio are better in [28] than in this work, they have difficulty in that it has to retrain for quantization. On the other hand, the proposed scheme has better performance without any other special scheme and provides fast analysis as well. It is possible to apply low-precision of symmetric and uniform quantization without a separate training process.

IV. CONCLUSION

In this paper, we proposed efficient layer-wise mixed-precision scheme. Instead of using the conventional time-consuming *mAP*-based analysis, the proposed scheme uses simple $SQNR_{Avg}$-based analysis to reduce the time amount required for layer-wise mixed-precision determination. Also, by considering the complexity of each layer, the proposed scheme provides the way to optimize the hardware complexity in terms of either the number of operations or the weight memory size. The proposed scheme shows more than 95% reduction in analysis runtime and dramatic hardware complexity reduction both in the number of operations and weight memory size while maintaining the model accuracy. Also, with the proposed method, users can readily apply it to inference tasks and leverage pre-trained neural network models without requiring any further training or additional hardware units.

REFERENCES

[1] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[2] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot MultiBox detector," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2016, pp. 21–37.

[3] L. Liu, W. Ouyang, X. Wang, P. Fieguth, J. Chen, X. Liu, and M. Pietikäinen, "Deep learning for generic object detection: A survey," *Int. J. Comput. Vis.*, vol. 128, no. 2, pp. 261–318, Feb. 2020.

[4] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, 2017.

[5] D. Lu and Q. Weng, "A survey of image classification methods and techniques for improving classification performance," *Int. J. Remote Sens.*, vol. 28, no. 5, pp. 823–870, Mar. 2007.

[6] A. Graves, A.-R. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 6645–6649.

[7] M. Malik, M. K. Malik, K. Mehmood, and I. Makhdoom, "Automatic speech recognition: A survey," *Multimedia Tools Appl.*, vol. 80, no. 6, pp. 9411–9457, 2021.

[8] S. Kim, S. Na, B. Y. Kong, J. Choi, and I.-C. Park, "Real-time SSDLite object detection on FPGA," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 29, no. 6, pp. 1192–1205, Jun. 2021.

[9] H.-H. Chin, R.-S. Tsay, and H.-I. Wu, "A high-performance adaptive quantization approach for edge CNN applications," 2021, *arXiv:2107.08382*.

[10] A. Trusov, E. Limonova, D. Slugin, D. Nikolaev, and V. V. Arlazarov, "Fast implementation of 4-bit convolutional neural networks for mobile devices," in *Proc. 25th Int. Conf. Pattern Recognit. (ICPR)*, Jan. 2021, pp. 9897–9903.

[11] A. N. Mazumder, J. Meng, H.-A. Rashid, U. Kallakuri, X. Zhang, J.-S. Seo, and T. Mohsenin, "A survey on the optimization of neural network accelerators for micro-AI on-device inference," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 11, no. 4, pp. 532–547, Dec. 2021.

[12] S. S. A. Zaidi, M. S. Ansari, A. Aslam, N. Kanwal, M. Asghar, and B. Lee, "A survey of modern deep learning based object detection models," *Digit. Signal Process.*, vol. 126, Jun. 2022, Art. no. 103514.

[13] S. Dong, P. Wang, and K. Abbas, "A survey on deep learning and its applications," *Comput. Sci. Rev.*, vol. 40, May 2021, Art. no. 100379.

[14] R. Mishra, H. P. Gupta, and T. Dutta, "A survey on deep neural network compression: Challenges, overview, and solutions," 2020, *arXiv:2010.03954*.

[15] Z. Liu, M. Sun, T. Zhou, G. Huang, and T. Darrell, "Rethinking the value of network pruning," 2018, *arXiv:1810.05270*.

[16] Z. Liu, H. Mu, X. Zhang, Z. Guo, X. Yang, K.-T. Cheng, and J. Sun, "MetaPruning: Meta learning for automatic neural network channel pruning," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 3295–3304.

[17] L. Bondi, S. Lameri, D. Güera, P. Bestagini, E. J. Delp, and S. Tubaro, "Tampering detection and localization through clustering of camera-based CNN features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 1855–1864.

[18] M. Xu, S. Qi, Y. Yue, Y. Teng, L. Xu, Y. Yao, and W. Qian, "Segmentation of lung parenchyma in CT images using CNN trained with the clustering algorithm generated dataset," *Biomed. Eng.*, vol. 18, no. 1, pp. 1–21, 2019.

[19] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *Comput. Sci.*, vol. 14, no. 7, pp. 38–39, 2015.

[20] J. Gou, B. Yu, S. J. Maybank, and D. Tao, "Knowledge distillation: A survey," *Int. J. Comput. Vis.*, vol. 129, no. 6, pp. 1789–1819, Jun. 2021.

[21] K. Khalil, O. Eldash, A. Kumar, and M. Bayoumi, "Economic LSTM approach for recurrent neural networks," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 66, no. 11, pp. 1885–1889, Nov. 2019.

[22] K. Khalil, A. Kumar, and M. Bayoumi, "Reconfigurable hardware design approach for economic neural network," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 69, no. 12, pp. 5094–5098, Dec. 2022.

[23] J. Tang, M. Liu, N. Jiang, W. Yu, C. Yang, and J. Zhou, "Knowledge distillation based on positive-unlabeled classification and attention mechanism," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2021, pp. 1–5.

[24] M. Nagel, R. A. Amjad, M. Van Baalen, C. Louizos, and T. Blankevoort, "Up or down? Adaptive rounding for post-training quantization," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 7197–7206.

[25] B. Ron, N. Yury, and H. Elad, "Post training 4-bit quantization of convolution networks for rapid-deployment," in *Proc. Adv. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, 2019, pp. 7948–7956.

[26] M. Nagel, M. V. Baalen, T. Blankevoort, and M. Welling, "Data-free quantization through weight equalization and bias correction," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1325–1334.

[27] Y. Choukroun, E. Kravchik, F. Yang, and P. Kisilev, "Low-bit quantization of neural networks for efficient inference," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*, Oct. 2019, pp. 3009–3018.

[28] D. T. Nguyen, H. Kim, and H.-J. Lee, "Layer-specific optimization for mixed data flow with mixed precision in FPGA design for CNN-based object detectors," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 6, pp. 2450–2464, Jun. 2021.

[29] X. Wei, H. Chen, W. Liu, and Y. Xie, "Mixed-precision quantization for CNN-based remote sensing scene classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 18, no. 10, pp. 1721–1725, Oct. 2021.

[30] C. Tang, K. Ouyang, Z. Wang, Y. Zhu, Y. Wang, W. Ji, and W. Zhu, "Mixed-precision neural network quantization via learned layer-wise importance," 2022, *arXiv:2203.08368*.

[31] T. Chu, Q. Luo, J. Yang, and X. Huang, "Mixed-precision quantized neural networks with progressively decreasing bitwidth," *Pattern Recognit.*, vol. 111, Mar. 2021, Art. no. 107647.

[32] Z. Dong, Z. Yao, A. Gholami, M. Mahoney, and K. Keutzer, "HAWQ: Hessian AWare quantization of neural networks with mixed-precision," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 293–302.

[33] A. Bulat and G. Tzimiropoulos, "Bit-mixer: Mixed-precision networks with runtime bit-width selection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 5168–5177.

[34] K. Wang, Z. Liu, Y. Lin, J. Lin, and S. Han, "HAQ: Hardware-aware automated quantization with mixed precision," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 8604–8612.

[35] D. Das, N. Mellempudi, D. Mudigere, D. Kalamkar, S. Avancha, K. Banerjee, S. Sridharan, K. Vaidyanathan, B. Kaul, E. Georganas, A. Heinecke, P. Dubey, J. Corbal, N. Shustrov, R. Dubtsov, E. Fomenko, and V. Pirogov, "Mixed precision training of convolutional neural networks using integer operations," 2018, *arXiv:1802.00930*.

[36] P. V. Kotipalli et al., "AMPT-GA: Automatic mixed precision floating point tuning for GPU applications," in *Proc. ACM Int. Conf. Supercomput.*, 2019, pp. 160–170.

[37] N. Mellempudi, S. Srinivasan, D. Das, and B. Kaul, "Mixed precision training with 8-bit floating point," 2019, *arXiv:1905.12334*.

[38] H. Zhang, D. Chen, and S.-B. Ko, "Efficient multiple-precision floating-point fused multiply-add with mixed-precision support," *IEEE Trans. Comput.*, vol. 68, no. 7, pp. 1035–1048, Jul. 2019.

[39] C. Chen, T. Zhang, Z. Yu, A. Raghuraman, S. Udayan, J. Lin, and M. M. S. Aly, "Scalable hardware acceleration of non-maximum suppression," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2022, pp. 96–99.

[40] M. Shi, P. Ouyang, S. Yin, L. Liu, and S. Wei, "A fast and power-efficient hardware architecture for non-maximum suppression," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 66, no. 11, pp. 1870–1874, Nov. 2019.

[41] C. Fang, H. Derbyshire, W. Sun, J. Yue, H. Shi, and Y. Liu, "A sort-less FPGA-based non-maximum suppression accelerator using multi-thread computing and binary max engine for object detection," in *Proc. IEEE Asian Solid-State Circuits Conf. (A-SSCC)*, Nov. 2021, pp. 1–3.

[42] K. Sun, Z. Li, Y. Zheng, H.-W. Kuo, K.-P. Lee, and K.-T. Tang, "An area-efficient accelerator for non-maximum suppression," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 70, no. 6, pp. 2251–2255, Jun. 2023.

[43] O. J. Al-Furaiji, N. A. Tuan, and V. Y. Tsviatkou, "A new fast efficient non-maximum suppression algorithm based on image segmentation," *Indonesian J. Elect. Eng. Comput. Sci.*, vol. 19, pp. 1062–1070, Aug. 2020.

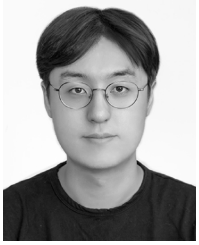
[44] D. Oro, C. Fernández, X. Martorell, and J. Hernando, "Work-efficient parallel non-maximum suppression for embedded GPU architectures," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, Mar. 2016, pp. 1026–1030.

[45] W. Kim, H. Kim, J. Lee, H. Kim, and J.-H. Kim, "Multi-mode SpMV accelerator for transprecision PageRank with real-world graphs," *IEEE Access*, vol. 11, pp. 6261–6272, 2023.

[46] J.-S. Park, C. Park, S. Kwon, T. Jeon, Y. Kang, H. Lee, D. Lee, J. Kim, H.-S. Kim, Y. Lee, S. Park, M. Kim, S. Ha, J. Bang, J. Park, S. Lim, and I. Kang, "A multi-mode 8k-MAC HW-utilization-aware neural processing unit with a unified multi-precision datapath in 4-nm flagship mobile SoC," *IEEE J. Solid-State Circuits*, vol. 58, no. 1, pp. 189–202, Jan. 2023.

[47] Q. Cheng, M. Huang, C. Man, A. Shen, L. Dai, H. Yu, and M. Hashimoto, "Reliability exploration of system-on-chip with multi-bit-width accelerator for multi-precision deep neural networks," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 70, no. 10, pp. 3978–3991, Oct. 2023.

HANA KIM (Graduate Student Member, IEEE) received the B.S. degree in electronic and electrical engineering from Ewha Womans University, Seoul, South Korea, in 2020, and the M.S. degree from the Digital System Architecture Laboratory, Ewha Womans University, where she is currently pursuing the Ph.D. degree with the Digital System Architecture Laboratory. Her current research interests include data type, deep neural network (DNN) accelerator, system-on-chip (SoC), and digital system architecture design.



HYUN EUN received the B.S. and M.S. degrees in electrical and electronic engineering from Sungkyunkwan University, Suwon, South Korea, in 2009 and 2011, respectively.

From 2011 to 2012, he worked on the Android OS media framework with Pixtree, Seoul, South Korea. From 2013 to 2018, he was a Video Firmware Developer with Chips&Media, Seoul. Since 2019, he has been a SDK Developer with the NPU Team, OPENEDGES Technology, Inc., Seoul. His research interest includes neural network quantization.

JUNG HWAN CHOI received the B.S. and M.S. degrees in electrical engineering from Seoul National University, Seoul, South Korea, and the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, USA, in 2009.

From 2009 to 2014, he was with the DMC Research and Development Center, Samsung Electronics, Suwon, South Korea, where he worked on SoC design and low-power design methodology. From 2015 to 2018, he was with the CAE Group, SK Hynix, Icheon, South Korea. In 2018, he joined OPENEDGES Technology, Inc., and currently leads the NPU Design Team for edge devices. His research interests include power-efficient NPU design and neural network quantization.

JI-HOON KIM (Senior Member, IEEE) received the B.S. (summa cum laude) and Ph.D. degrees in electrical engineering and computer science from KAIST, Daejeon, South Korea, in 2004 and 2009, respectively.

In 2009, he joined Samsung Electronics, Suwon, South Korea, as a Senior Engineer, and worked on next-generation architecture for 4G communication modem system-on-chip (SoC). He was an Associate Professor with the Department of Electronics Engineering, Chungnam National University, Daejeon, from 2010 to 2016. In 2018, he joined the Department of Electronic and Electrical Engineering, Ewha Womans University, Seoul, South Korea, as a Faculty Member, where he is currently a Professor. His current interests include CPU microarchitecture, domain-specific SoC, and deep neural network accelerators.

Dr. Kim served on the Technical Program Committee and Organizing Committee for various international conferences, including the IEEE International Conference on Computer Design (ICCD), the IEEE Asian Solid-State Circuits Conference (A-SSCC), and the IEEE International Solid-State Circuits Conference (ISSCC). He was a co-recipient of the Distinguished Design Award at the 2019 IEEE A-SSCC, and a recipient of the Best Design Award at 2007 Dongbu HiTek IP Design Contest, the First Place Award at 2008 International SoC Design Conference (ISOCC) Chip Design Contest, and the IEEE/IEIE Joint Award for Young Scientist and Engineer.