This ICCV paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

# Lossy and Lossless ($L^2$) Post-training Model Size Compression

Yumeng Shi[1,2] Shihao Bai[2] Xiuying Wei[1,2] Ruihao Gong[1,2*] Jianlei Yang[1*]
[1]Beihang University [2]SenseTime Research
{senbei,jianlei}@buaa.edu.cn {baishihao,gongruihao}@sensetime.com weixiuying966@gmail.com

## Abstract

*Deep neural networks have delivered remarkable performance and have been widely used in various visual tasks. However, their huge sizes cause significant inconvenience for transmission and storage. Many previous studies have explored model size compression. However, these studies often approach various lossy and lossless compression methods in isolation, leading to challenges in achieving high compression ratios efficiently. This work proposes a post-training model size compression method that combines lossy and lossless compression in a unified way. We first propose a unified parametric weight transformation, which ensures different lossy compression methods can be performed jointly in a post-training manner. Then, a dedicated differentiable counter is introduced to guide the optimization of lossy compression to arrive at a more suitable point for later lossless compression. Additionally, our method can easily control a desired global compression ratio and allocate adaptive ratios for different layers. Finally, our method can achieve a stable* $10\times$ *compression ratio without sacrificing accuracy and a* $20\times$ *compression ratio with minor accuracy loss in a short time. Our code is available at https://github.com/ModelTC/L2_Compression.*

## 1. Introduction

In recent years, deep neural networks (DNNs), especially convolutional neural networks (CNNs) [1, 2, 3, 4], have achieved attractive performance in various computer vision tasks such as image classification, detection, and segmentation. However, as their performance improves, their parameter counts also significantly increase, which is very storage-consuming. Therefore, despite their excellent performance, it is difficult to deploy models with a large number of parameters, particularly on mobile or edge devices with limited storage resources.

Model compression [5, 6, 7, 8, 9, 10] is a common solution to reduce the model size, including lossless and lossy compression. Common lossless compression methods such as Huffman coding and Range coding are both entropy coding methods. They can leverage redundant information in data to achieve distortion-free compression. However, for data with weak spatiotemporal coherence, high compression ratios are often difficult to achieve. Compared with lossless methods, lossy methods such as pruning [11] and quantization [12, 13] have attracted more attention recently. Pruning reduces the model size by removing extraneous weights, and quantization replaces weights in a low-bit format. Both pruning and quantization are trade-offs between model distortion and compression ratio. Previous studies have primarily focused on individual compression methods or have merely combined different compression techniques without considering the interaction between them, resulting in multiple isolated trade-offs in successive stages. Hence, they can hardly achieve a higher compression ratio with a small amount of data and little training time.

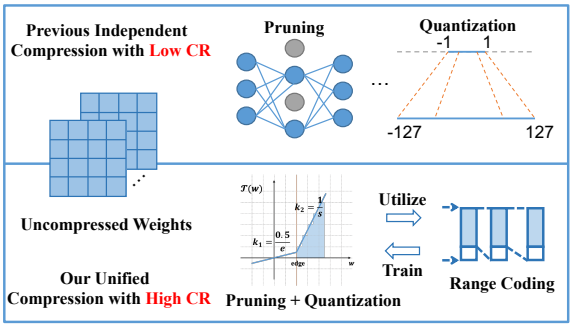


Figure 1. Comparison between previous compression methods and our unified post-training compression.

To address this issue, as shown in Figure 1, this paper proposes to mix different compression methods and optimize them under the recent popular post-training setting, which slightly adjusts weights for better performance. We build an optimization objective that introduces an entropy regularization term, making the global compression ratio controllable. Based on it, a unified parametric weight transformation is first designed to integrate different lossy com-

*Corresponding Author

pression techniques together, allowing us to jointly explore various compression strategies and determine unique ones for each layer. Second, we devise a novel differentiable counter to make our entropy regularization term differentiable by leveraging kernel functions. This differentiable way imposes constraints on the distribution of compressed weights during optimization, contributing to more compatible optimized weights with later lossless compression. Consequently, our method combining lossy and lossless compression can achieve a consistently superior compression ratio with satisfying accuracy performance in an efficient manner.

To the best of our knowledge, this is the first work that presents a unified modeling approach for various lossy compression methods while leveraging the characteristics of lossless compression to optimize the process of lossy compression. Extensive experiments on various networks verify the efficacy of our method (e.g., stable $10\times$ compression ratio without sacrificing accuracy and up to $20\times$ compression ratio with minor accuracy loss).

Our main contributions can be summarized as follows:

- We propose a pioneering post-training model size compression method that combines lossy and lossless compression with a new optimization objective.
- We design a unified parametric weight transformation approach for lossy compression methods, integrating techniques such as pruning and quantization into a single stage and determining each layer's unique compression scheme.
- We introduce a dedicated differentiable counter to estimate the entropy of compressed weights. This counter ensures that the distribution of the optimized weights is more amenable for lossless compression.
- Extensive experiments conducted on various architectures, including classification and object detection tasks, demonstrate that our method achieves high compression ratios with negligible accuracy drops.

## 2. Related Work

In the past, numerous researchers have explored model compression techniques. Knowledge distillation [14] is one such technique that aims to transfer knowledge from a complex teacher model to a simplified student model. By utilizing the soft target probability distribution from the teacher model, the student model achieves comparable performance with a smaller size. Matrix factorization [15, 16] is another such technique that breaks down neural network weight matrices into smaller ones, reducing parameters and making the model more resource-efficient. Common methods include SVD and QR decomposition. Hereafter, we focus on introducing network pruning, weight quantization, and entropy encoding.

**Network pruning.** This technique achieves model compression by selectively removing unimportant weights. Network pruning can be categorized into structured pruning and unstructured pruning. Structured pruning [17] targets specific structures, like rows, columns, channels, or filters, which can accelerate computation. Unstructured pruning [11] solely considers the importance of weight elements, disregarding their position. Our objective is to compress the model size, so we focus on unstructured pruning. Previous studies [18, 19] have explored various methods to measure weight importance, including magnitude-based and derivative-based methods. Additionally, some studies [20, 21, 22] have concentrated on determining the sparsity ratio for each layer.

**Weight quantization.** Another common model compression technique is weight quantization, which reduces the number of bits needed for weight storage by discretizing weight values. Many studies [12, 23] have explored the effects of quantization intervals, clips, and rounding methods. There are also studies [24, 25] focusing on non-uniform quantization, which utilizes clustering to group weights instead of using a uniform formula for calculation. Other studies [26, 27, 28, 29, 30, 31, 32] have investigated mixed precision quantization, suggesting that the importance of network layers can influence the required storage bit-width for each layer.

**Entropy coding.** Entropy coding, as a lossless data compression technique, aims to achieve data compression without any loss of information. It re-encodes data based on its probability distribution, using shorter codes for high-frequency data and longer codes for low-frequency data, effectively saving storage space. Common entropy coding methods encompass Huffman coding, and arithmetic coding, among others. It is frequently integrated with other methods to further reduce weight storage [6, 33] or reduce the memory footprint and transmission bandwidth of feature maps during inference [34, 35].

Regarding model compression, using a single technique often fails to achieve satisfactory performance. Certain studies [36, 37, 6] have explored using multiple methods sequentially. However, they treated them in isolation without considering their mutual impacts. Some studies [7] have attempted to apply in-parallel pruning and quantization but overlooked the influence of lossless compression on lossy compression. Furthermore, these approaches typically required extensive training to achieve desirable results. Addressing these challenges involves contemplating the integration of lossless compression effects and the unification

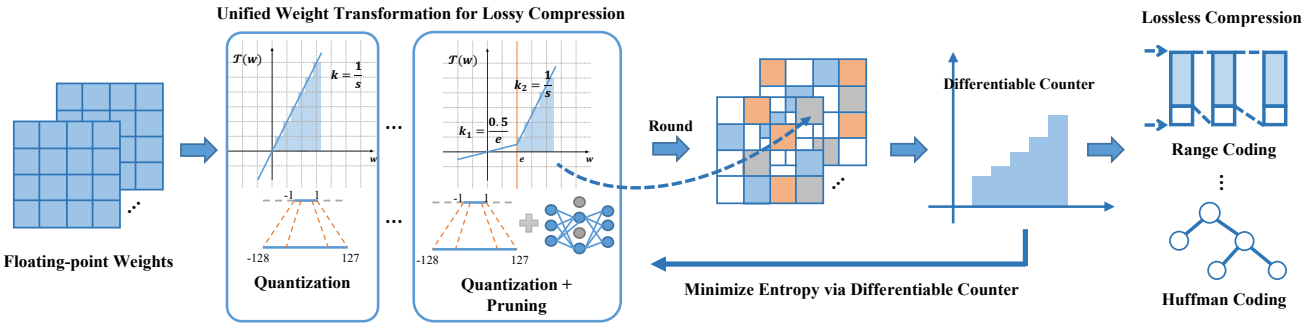


Figure 2. Overview of the proposed post-training compression method.

of modeling for lossy compression methods. We propose an efficient post-training approach that achieves high compression ratios while maintaining superior accuracies.

## 3. Preliminaries

In this section, some basic concepts of model compression methods will be introduced. Model compression reduces the model size, which can be divided into lossy compression and lossless compression.

**Basic notation.** In this paper, we mark vectors and flattened matrices $\boldsymbol{w}$. Scalar multiplication is defined as $\cdot$. We take $CR$ as an abbreviation for compression ratio, $\mathcal{L}(\cdot)$ as the loss function, $\mathcal{P}(\cdot)$ as probability mass function, and $\mathcal{T}(\cdot)$ as an element-wise transformation that converts dense or real signals to sparse or discrete ones.

**Lossless compression.** Lossless compression utilizes redundancy in data to achieve compression without introducing distortion, allowing for complete data recovery during decompression. The compression ratio is positively related to the amount of data redundancy. In information theory, Shannon's source coding theorem states that information entropy measures the average information content of given data, representing the average shortest coding length.

Since the model weights do not exhibit spatial and temporal locality in our task, we treat the $m$ elements of each layer's weights $\boldsymbol{w}$ as independent and identically distributed random variables $w$. The self-information of $w$ can be expressed as:

$$\mathcal{I}(w) = -log_b \mathcal{P}(w), \tag{1}$$

where $\mathcal{P}(\cdot)$ represents the probability mass function of $w$. When $b$ equals 2, the unit of self-information is bits. The entropy of $w$ is the expected value of the self-information:

$$\mathcal{H}(w) = \mathbb{E}(\mathcal{I}(w)) = -\sum_{\tilde{w}\in\tilde{\boldsymbol{w}}} \mathcal{P}(\tilde{w}) \log_2 \mathcal{P}(\tilde{w}), \tag{2}$$

where $\tilde{\boldsymbol{w}}$ represents the symbol set of $\boldsymbol{w}$. Considering $\mathcal{P}(\cdot)$ as the sampling distribution function, the total shortest coding length for a layer is:

$$\begin{aligned} \mathcal{S}(\boldsymbol{w}) &= m\mathcal{H}(w) = -\sum_{\tilde{w}\in\tilde{\boldsymbol{w}}} m\mathcal{P}(\tilde{w}) \log_2 \mathcal{P}(\tilde{w}) \\ &= -\sum_{\tilde{w}\in\tilde{\boldsymbol{w}}} num(\tilde{w}) \log_2 \mathcal{P}(\tilde{w}) \\ &= -\sum_{w\in\boldsymbol{w}} \log_2 \mathcal{P}(w), \end{aligned} \tag{3}$$

where $num(\tilde{w})$ represents the number of elements in $\boldsymbol{w}$ that are equal to $\tilde{w}$. The entire neural network's shortest encoding length is the sum of that of each layer.

**Lossy compression.** Pruning and quantization are prevalent lossy compression methods that compress the original weights $\boldsymbol{w}$ to $\hat{\boldsymbol{w}}$ by removing unimportant parameters or converting floating-point values to low-bit fixed-point numbers. In the post-training setting, some studies [38, 12] on pruning and quantization also utilize a small amount of calibration data to quickly fine-tune the weights. Most studies aim to minimize the mean squared error between the model outputs before and after compression, formulating their optimization goal as follows:

$$\min_{\hat{\boldsymbol{w}}} \mathbb{E}\left[\|\mathcal{F}(\boldsymbol{x}, \hat{\boldsymbol{w}})) - \mathcal{F}(\boldsymbol{x}, \boldsymbol{w})\|_F^2\right], \tag{4}$$

where $\boldsymbol{x}$ is extracted from a calibration dataset with about hundreds of images and $\mathcal{F}(\cdot)$ produces outputs. Here, we use the output of a neural network.

## 4. Method

In this section, we will first present the optimization objective for both lossy and lossless compression methods in the post-training setting. Subsequently, two novel techniques will be introduced: the unified weight transformation and the differentiable counter. These techniques serve to optimize lossy compression uniformly and enhance collaboration with later lossless compression. Using our tech-

niques shown in Figure 2, it is now possible to efficiently achieve a highly compressed yet accurate model.

### 4.1. Optimization Objective

In this part, a novel optimization objective is introduced, targeting superior compressed results in the post-training setting.

To pursue high accuracy with limited data and GPU effort, we follow previous studies [11, 12] to slightly tune the weights, minimizing the distance between the output after compression and its original counterpart in Eq. 4. Besides, another regularization term is added to encourage small compressed models.

$$\min_{\hat{\boldsymbol{w}}} \mathbb{E}\left[\|\mathcal{F}(\boldsymbol{x}, \hat{\boldsymbol{w}})) - \mathcal{F}(\boldsymbol{x}, \boldsymbol{w})\|_F^2 + \lambda \cdot \mathcal{L}_r(\hat{\boldsymbol{w}})\right], \quad (5)$$

where $\mathcal{L}_r(\cdot)$ defines the new regularization term, and $\lambda$ is the balance factor.

Motivated by [5], we incorporate entropy into the regularization term and take $\mathcal{L}_r(\cdot) = \mathcal{S}(\cdot)$, rather than directly calculating the compressed model size. By explicitly building the relationship between compressed model size with lossless techniques and entropy term, we can guide the optimization to pursue a better weight distribution, which is more appropriate for later lossless compression. In this way, the potential of the subsequent lossless compression can be well unleashed.

$$\min_{\hat{\boldsymbol{w}}} \mathbb{E}\left[\|\mathcal{F}(\boldsymbol{x}, \hat{\boldsymbol{w}})) - \mathcal{F}(\boldsymbol{x}, \boldsymbol{w})\|_F^2 + \lambda \cdot S(\hat{\boldsymbol{w}})\right]. \quad (6)$$

By substituting Eq. 3 into the above equation, the following one can be deduced:

$$\min_{\hat{\boldsymbol{w}}} \mathbb{E}\left[\|\mathcal{F}(\boldsymbol{x}, \hat{\boldsymbol{w}})) - \mathcal{F}(\boldsymbol{x}, \boldsymbol{w})\|_F^2 - \lambda \cdot \sum_{\hat{w} \in \hat{\boldsymbol{w}}} log_2 \mathcal{P}(\hat{w})\right]. \quad (7)$$

**Compression ratio control.** Furthermore, considering our objective is to achieve high accuracy while meeting the compression ratio requirement, we introduce $CR_{\text{target}}$ into the term to control the whole compression ratio as shown in Eq. 8. Once the $CR_{\text{target}}$ is attained, the lossy compression will be subject only to its original objective:

$$\mathcal{L}_r(\hat{\boldsymbol{w}}) = ReLU(-\sum_{\hat{w} \in \hat{\boldsymbol{w}}} log_2 \mathcal{P}(\hat{w}) - \frac{32 \cdot numel(\boldsymbol{w})}{CR_{target}}), \quad (8)$$

where $32 \cdot numel(\boldsymbol{w})$ is the size of the original weights $\boldsymbol{w}$.

Optimizing the above equations is not straightforward. Specifically, to attain our optimization objective, we put forward a method that can jointly optimize various lossy compression methods with a unified weight transformation in Sec. 4.2. Additionally, a method is presented to ensure the differentiability of the probability mass function by leveraging a kernel function in Sec. 4.3.

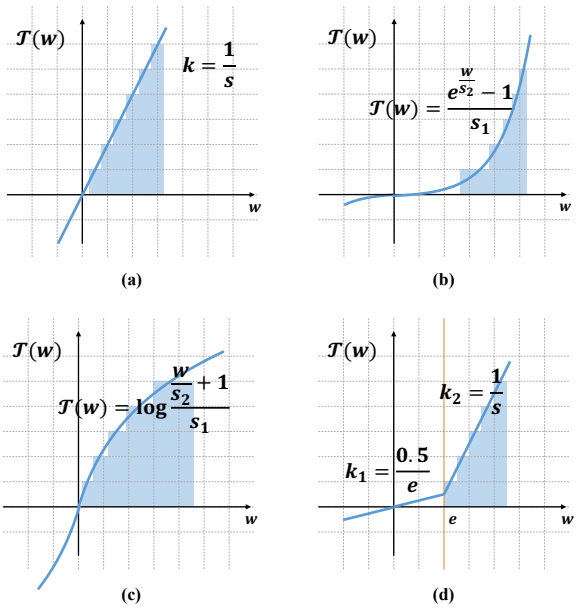


Figure 3. Some forms of transformations: (a) refers to linear quantization, (b) refers to exponential quantification, (c) refers to logarithmic quantization, (d) refers to joint pruning and quantization.

### 4.2. Unified Weight Transformation

Previous studies [6, 7] treated different compression methods separately without considering how they might affect each other. Or they lacked a unified approach to constrain compression methods according to the optimization objective. Consequently, they heavily relied on retraining the model's weights to achieve desired results, making it challenging to obtain satisfactory outcomes in a post-training setting. To address these issues, we abstract different lossy compression methods into a unified weight transformation.

**Unified representation.** Here, we propose to present quantization and pruning functions in a unified way. Quantization usually first quantizes weights into integers and then recovers them to floating-points, as defined in Eq. 9.

$$\hat{w} = \mathcal{T}^{-1}(\lfloor \mathcal{T}(w) \rceil), \quad (9)$$

where $\lfloor \cdot \rceil$ is the round-to-nearest operation, the element-wise function $\mathcal{T}(\cdot)$ encodes the weight transformation, and $\mathcal{T}^{-1}(\cdot)$ is its reverse function. Note that $\mathcal{T}(\cdot)$ always serves as a continuous function under different quantization settings, thus $\mathcal{T}^{-1}(\cdot)$ always exists. Especially, we can achieve uniform quantization by setting $\mathcal{T}(w) = \frac{w}{s}$, where $s$ stands for the interval of compressed data, as shown in Figure 3 (a). Non-uniform quantization like logarithmic or exponential ones can also be obtained by taking some non-linear $\mathcal{T}(\cdot)$ functions, as illustrated in Figure 3 (b) and (c).

Motivated by the quantization process Eq. 9, we propose to denote the pruning in the same way but with different $\mathcal{T}(\cdot)$ and $\mathcal{T}^{-1}(\cdot)$ functions. The $\mathcal{T}(\cdot)$ transformation for pruning can be devised as below:

$$\mathcal{T}(w) = \begin{cases} \frac{0.5}{e} \cdot w, & |w| < e \\ w, & |w| \geq e \end{cases}, \tag{10}$$

where $e$ is the pruning threshold. In Eq. 10, it can be observed that elements inside $(-e, e)$ are projected into $(-0.5, 0.5)$, which will become zero after rounding and reverse functions. Otherwise, values are kept intact. Therefore, unstructured pruning can be practiced by taking the same Eq. 9 like quantization and assigning $\mathcal{T}(\cdot)$ to Eq. 10. That is to say, unstructured pruning and different kinds of quantization can be clarified under the same concept.

**Joint pruning and quantization.** The unified representation Eq. 9 for quantization and pruning brings us a way to integrate and optimize them together. By designing a continuous piece-wise function as shown in Eq. 11 and Figure 3 (d), we can pursue a joint objective.

$$\mathcal{T}(w) = \begin{cases} \frac{0.5}{e} \cdot w, & |w| < e \\ sign(w) \cdot (\frac{|w| - e}{s} + 0.5), & |w| \geq e \end{cases}, \tag{11}$$

where weights whose absolute values are smaller than $e$ will be pruned, and rest non-pruned weights will be operated with $s$ to achieve quantization.

Eq. 11 establishes a joint form for lossy compression. By using such a $\mathcal{T}(\cdot)$ and Eq. 9 in Eq. 5, joint pruning and quantization optimization can be realized. Note that apart from weight tuning for our objective, parameters of pruning and quantization methods (i.e, $e$ and $s$) are also optimized together. Consequently, by adopting $\mathcal{T}_i(\cdot)$ for each layer, we can explore various compression methods for each layer and determine their unique strategies. Thus, the model can be effectively compressed with enhanced outcomes in a post-training manner.

### 4.3. Differentiable Counter

To minimize Eq. 7, the differentiability of $\mathcal{P}(\hat{w})$ is required, as it needs to guide the learning of parameters in $\mathcal{T}(\cdot)$. Since $\hat{w} = \mathcal{T}^{-1}(\lfloor \bar{w} \rceil)$ and $\mathcal{T}^{-1}(\cdot)$ is bijective, we can deduce that $\mathcal{P}(\hat{w}) = \mathcal{P}(\lfloor \bar{w} \rceil)$. We model $\mathcal{P}(\lfloor \bar{w} \rceil)$ instead of $\mathcal{P}(\hat{w})$ as the interval remains fixed during the rounding operation.

In comparison with complex probability models, directly counting the frequencies of weights provides a more accurate representation of the weight distribution. Based on this, a novel differentiable counter is designed to estimate the probability mass function, facilitating the learning. We initially define the cumulative distribution function $f_{\bar{\boldsymbol{w}}}(x)$ for weights $\bar{\boldsymbol{w}}$ as follows:

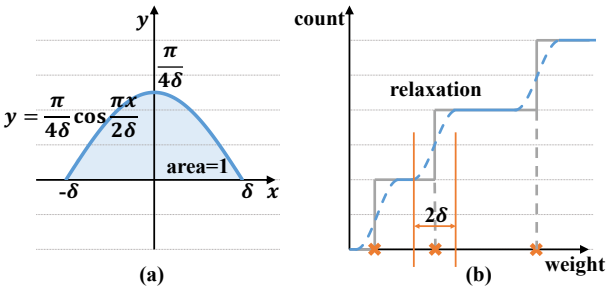


Figure 4. (a) refers to kernel function for relaxation, (b) refers to the relaxed count function.

$$f_{\bar{\boldsymbol{w}}}(x) = \frac{\mathcal{C}(\bar{\boldsymbol{w}}, x)}{numel(\bar{\boldsymbol{w}})}, \tag{12}$$

$$\mathcal{C}(\bar{\boldsymbol{w}}, x) = \sum_{\bar{w} \in \bar{\boldsymbol{w}}} \epsilon(x - \bar{w}), \tag{13}$$

$$\epsilon(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}, \tag{14}$$

where $numel(\bar{\boldsymbol{w}})$ represents the number of weights in $\bar{\boldsymbol{w}}$, and $\mathcal{C}(\bar{\boldsymbol{w}}, x)$ is a counter that counts the number of weights less than $x$ in $\bar{\boldsymbol{w}}$.

To ensure the differentiability of $\mathcal{C}(\bar{\boldsymbol{w}}, x)$, we leverage a kernel function and replace the step function $\epsilon(x)$. In detail, the relaxation function is defined as below:

$$\mathcal{C}(\bar{\boldsymbol{w}}, x) \cong \sum_{\bar{w} \in \bar{\boldsymbol{w}}} \int_{-\infty}^{x - \bar{w}} \mathcal{K}(u) du, \tag{15}$$

$$\mathcal{K}(x) = \begin{cases} 0, & x \in (-\infty, \delta) \cup (\delta, \infty) \\ \frac{\pi}{4\delta} \cdot cos(\frac{\pi x}{2\delta}), & x \in [-\delta, \delta] \end{cases}, \tag{16}$$

where $\mathcal{K}(\cdot)$ is the kernel function with a relaxation factor $\delta$ as shown in Figure 4. When $\delta$ approaches zero, the function simplifies to Eq. 13. Considering the rounding operation, we estimate the probability mass of $\lfloor \bar{w} \rceil$ based on the difference in cumulative probability function between $\lfloor \bar{w} \rceil + 0.5$ and $\lfloor \bar{w} \rceil - 0.5$:

$$\mathcal{P}(\lfloor \bar{w} \rceil) = f_{\bar{\boldsymbol{w}}}(\lfloor \bar{w} \rceil + 0.5) - f_{\bar{\boldsymbol{w}}}(\lfloor \bar{w} \rceil - 0.5). \tag{17}$$

With the differentiable counter, obtaining the differentiable probability mass of the weights $\lfloor \bar{\boldsymbol{w}} \rceil$ becomes easy. Besides, we utilize the Straight-Through Estimator (STE) [40] to obtain gradients for rounding operations. Hence, the entropy term in Eq. 7 can be minimized to achieve $\lfloor \bar{\boldsymbol{w}} \rceil$ more suitable for later lossless compression during the weight transformation process, enabling a more effective combination of lossy and lossless compression.

| Model | Method | Top1-acc | Top5-acc | Target CR | CR |
|---|---|---|---|---|---|
| ResNet-18 | uncompressed | 71.06% | 89.90% | - | 1.0× |
| | DFQ | 61.50% | 83.56% | - | 9.8× |
| | DeepCABAC | 70.68% | 89.67% | - | 7.6× |
| | AdaRound | 70.0% | 89.26% | - | 9.7× |
| | Data-Aware PNMQ* | 69.21% / 69.76% | 88.76% / 89.08% | - | 7.4× |
| | Ours | 70.79% | 89.94% | 12× | 12.0× |
| | Ours (higher CR) | 70.20% | 89.47% | 15× | 15.0× |
| ResNet-50 | uncompressed | 76.63% | 93.07% | - | 1.0× |
| | DFQ | 75.47% | 92.36% | - | 6.1× |
| | AdaRound | 75.87% | 92.66% | - | 9.4× |
| | Data-Aware PNMQ* | 75.5% / 76.13% | 92.74% / 92.86% | - | 7.8× |
| | Ours | 76.68% | 93.13% | 10× | 10.0× |
| | Ours (higher CR) | 75.95% | 92.82% | 15× | 15.0× |
| MobileNetV2 | uncompressed | 72.62% | 90.62% | - | 1.0× |
| | DFQ | 68.70% | 88.30% | - | 5.0× |
| | DeepCABAC | 72.00% | 90.39% | - | 4.8× |
| | AdaRound | 69.01% | 88.86% | - | 7.2× |
| | Data-Aware PNMQ* | 71.68% / 71.88% | 90.2% / 90.29% | - | 4.9× |
| | Ours | 72.29% | 90.58% | 8× | 7.9× |
| | Ours (higher CR) | 71.53% | 90.18% | 10× | 9.9× |
| RegNet-600m | uncompressed | 73.55% | 91.57% | - | 1.0× |
| | DFQ | 73.32% | 91.52% | - | 5.6× |
| | DeepCABAC | 70.76% | 90.49% | - | 5.3× |
| | AdaRound | 71.92% | 90.65% | - | 9.7× |
| | Ours | 73.08% | 91.25% | 12× | 11.9× |
| RegNet-3200m | uncompressed | 78.36% | 94.16% | - | 1.0× |
| | DFQ | 78.06% | 94.02% | - | 5.8× |
| | DeepCABAC | 77.02% | 93.40% | - | 5.5× |
| | AdaRound | 77.32% | 93.57% | | 10.4× |
| | Ours | 78.32% | 94.13% | 10× | 10.0× |
| | Ours (higher CR) | 77.82% | 93.89% | 15× | 15.0× |
| MNasNet | uncompressed | 76.56% | 93.15% | - | 1.0× |
| | DFQ | 76.20% | 92.97% | - | 5.1× |
| | DeepCABAC | 74.04% | 92.02% | - | 5.1× |
| | AdaRound | 74.41% | 91.99% | | 8.4× |
| | Ours | 76.04% | 92.77% | 12× | 12.0× |

* means using the results published in the original paper [39].

Table 1. Comparison results with state-of-the-arts on various networks. Our unified compression method achieves the best performance.

## 5. Experiment

This section offers a comprehensive evaluation of our method. We compare and analyze the results of the method with state-of-the-art methods on the classification task, and present results on the detection task. In addition, ablation studies are conducted, accompanied by the analysis of time cost, weight transformation, and compression ratio.

### 5.1. Experimental Setting

To demonstrate the generality of our proposed method, extensive evaluations encompassing both the classification task and the object detection task are performed. For the classification task, we assess its performance across a wide range of convolutional neural networks, including ResNet [1], MobileNet [3], RegNet [2], and MNasNet [4]. The evaluation is carried out on the challenging ImageNet-1k dataset [41], with a calibration set of 1000 images sampled from the training set to train the transformation parameters. Moreover, for the object detection task, we conduct experiments on the YOLOv5 model [42], and the performance is validated on the widely used COCO2017 dataset [43].

We apply Eq. 11 for weight transformation and normal quantization for bias. Weight transformation parameters are

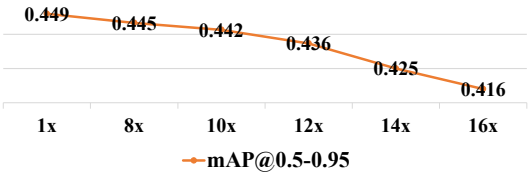


Figure 5. The mAP@0.5-0.95 of YOLOv5 with CR increasing.

initially trained, followed by fine-tuning of the weights. Inspired by knowledge distillation, both the complete network output and specific intermediate layer outputs contribute to the computation of the mean squared error loss. The networks are trained for 5 epochs, with each epoch containing 300 iterations of transformation training and 1000 iterations of weight fine-tuning. Finally, range coding, realized by [44], is used to encode the model weights. More implementation details can be found in the supplementary material.

### 5.2. Classification Task

We compare our method with the state-of-the-art DFQ [13] (only 8-bit weights), DeepCABAC [36, 37], Data-Aware PNMQ [39], and AdaRound [12] (only 4-bit weights) on ImageNet-1k. In the case of AdaRound, the first and last layers are 8-bit weights. Additionally, to ensure a fair comparison, we apply range coding for all methods.

Table 1 lists the performance of all methods. Thanks to the unified modeling of different compression techniques, our method could easily achieve the best compression ratio with little accuracy loss, proving the superiority of our method. For instance, the proposed method reduces the model size by more than 12 times on ResNet-18 with only a 0.3% accuracy drop, achieving about 60% improvement than PNMQ. Moreover, we can find that different networks have different sensitivities to compression. MobileNetV2 is the most sensitive, and RegNet-3200m with a larger number of parameters has a higher compression potential.

### 5.3. Object Detection Task

Besides, we conduct experiments on the obeject detection task, using a YOLOv5 model and the COCO2017 dataset. The final results are shown in Figure 5. In Figure 5, mAP (mean average precision) is a widely-adopted metric, which serves to evaluate the accuracy and recall rate of a model in detecting objects across different classes. A higher mAP implies that the model can better identify objects. It can be observed that our method still performs remarkably well on this task.

### 5.4. Ablation Study

In this part, ablation experiments will be performed on the proposed unified weight transformation $\mathcal{T}(\cdot)$ and the differentiable counter.

| weight transformation $\mathcal{T}(\cdot)$ | CR | Top1-acc |
|---|---|---|
| linear quantization | 14.97× | 70.37% |
| log quantization | 14.91× | 70.24% |
| exp quantization | 14.94× | 70.29% |
| joint pruning and quantization | 14.94× | 70.20% |
| linear quantization | 19.85× | 68.34% |
| log quantization | 19.89× | 68.47% |
| exp quantization | 19.87× | 68.75% |
| joint pruning and quantization | 19.95× | 69.15% |

Table 2. Compression results with different transformations.

| kernel | resolution | CR | Top1-acc | Top5-acc |
|---|---|---|---|---|
| cosine | 16 | 15.10× | 70.14% | 89.62% |
| | 32 | 15.03× | 70.25% | 89.60% |
| | 64 | 14.93× | 70.21% | 89.60% |
| | 128 | 15.00× | 70.04% | 89.63% |
| linear | 16 | 15.29× | 70.15% | 89.57% |
| | 32 | 15.15× | 70.05% | 89.40% |
| | 64 | 14.95× | 70.07% | 89.50% |
| | 128 | 15.00× | 70.20% | 89.57% |
| triangle | 16 | 15.06× | 70.11% | 89.56% |
| | 32 | 15.11× | 70.09% | 89.52% |
| | 64 | 15.09× | 70.14% | 89.61% |
| | 128 | 14.90× | 70.14% | 89.64% |

Table 3. Performance on ResNet-18 at 15× CR for the differentiable counter with different resolutions and kernel functions.

**Choice of weight transformation.** Table 2 presents the results of various transformations described in Figure 3. It is evident that different transformations can achieve comparable performance when the compression ratio is close to 15× on ResNet-18, demonstrating the robustness of our transformation. However, as the compression ratio increases to 20×, the transformation that combines pruning and quantization exhibits better performance. This observation highlights that our unified transformation effectively harnesses diverse compression methods, enabling the possibility of achieving extreme model compression.

**Choice of differentiable counter.** Table 3 provides compression results on ResNet-18 at 15× CR with different relaxation factors $\delta = \frac{max(\boldsymbol{w})-min(\boldsymbol{w})}{resolution}$ and kernel functions. We explored three kernel functions (cosine, linear, and triangle) and four resolutions (16, 32, 64, 128) for the differentiable counter. The results under all settings demonstrate the robustness of our differentiable counter.

### 5.5. Analysis

In this part, we provide some analysis of our method.

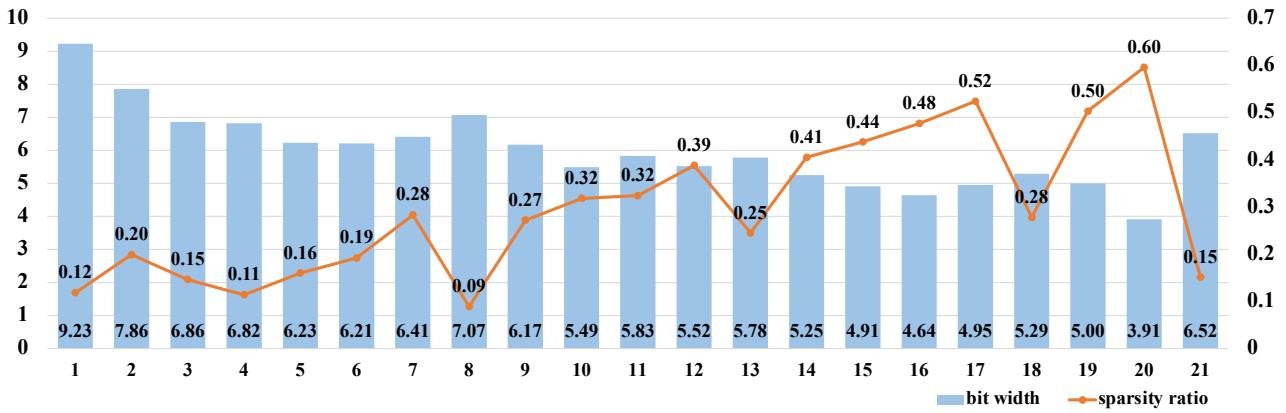


Figure 6. The distribution of bit width and sparsity ratio of ResNet-18 at a compression ratio of 20×.

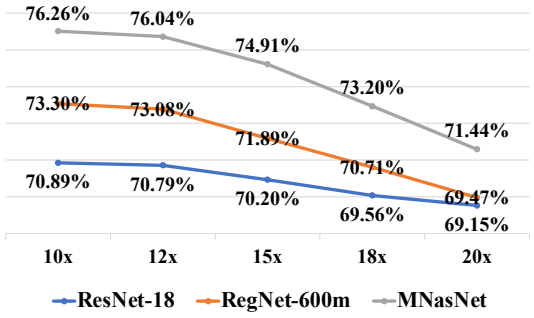


Figure 7. The accuracy of compressed models with CR increasing.

**Analysis of time consumption.** The main time cost of our method during compression comes from the training of transformation parameters and weight fine-tuning. We use a V100 GPU with 16GB memory and set the batch size to 32. For models such as ResNet-18, the training takes around 0.8s per iteration, while fine-tuning takes around 0.08s. Larger models may take more time. Typically, 500 iterations for training and 2000 for fine-tuning can achieve satisfactory results, which take only around 10 minutes.

As for the overhead of decompression during inference, lossless compression is applied to the weights after the complete transformation $\mathcal{T}^{-1}(\lfloor \mathcal{T}(w) \rceil)$. Hence, we only need to decode the entropy coding without dequantization or any other inverse transformations. The time cost of decoding is worth considering, but it can be mitigated by employing various techniques, such as heterogeneous computing.

**Analysis of weight transformation.** The transformation combining quantization and pruning adaptively selects suitable quantization bits and sparsity ratios for each layer, as Figure 6 shows. It helps detect layers with redundant parameters, such as "layer4.1.conv2" (20 in the figure) in ResNet-18, meaningful for network structure optimizing.

**Analysis of compression ratio.** Figure 7 shows the trend of performance with respect to the compression ratio. It can be observed that, upon reaching a certain compression ratio, such as 12× on MNasNet, the model experiences a significant accuracy drop. The controllability of the compression ratio in our method assists us in a more practical trade-off between accuracy and compression ratio.

## 6. Conclusion

In this work, we propose a novel post-training compression method that combines lossy and lossless compression. For lossy compression, we unify the modeling of weight distortion via a unified weight transformation for pruning, quantization, and so on. Moreover, we design a dedicated differentiable counter that accurately computes the information entropy of the compressed weights, which can regulate the weights and adapt to later lossless compression, thus achieving a better compression ratio. Extensive experiments on various networks prove the superiority of our method compared to previous methods. Furthermore, our work provides a meaningful perspective for more extreme model compression in the future by unifying different compression methods.

## Acknowledgment

We sincerely thank the anonymous reviewers for their serious reviews and valuable suggestions to make this better. This work was supported in part by the National Natural Science Foundation of China under Grant 62206010 and 62072019.

## References


[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016.

[2] Ilija Radosavovic, Raj Prateek Kosaraju, Ross B. Girshick, Kaiming He, and Piotr Dollár. Designing network design spaces. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 10425–10433. Computer Vision Foundation / IEEE, 2020.

[3] Mark Sandler, Andrew G. Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4510–4520. Computer Vision Foundation / IEEE Computer Society, 2018.

[4] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V. Le. Mnasnet: Platform-aware neural architecture search for mobile. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 2820–2828. Computer Vision Foundation / IEEE, 2019.

[5] Deniz Oktay, Johannes Ballé, Saurabh Singh, and Abhinav Shrivastava. Scalable model compression by entropy penalized reparameterization. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[6] Song Han, Huizi Mao, and William J. Dally. Deep compression: Compressing deep neural network with pruning, trained quantization and huffman coding. In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[7] Frederick Tung and Greg Mori. CLIP-Q: deep network compression learning by in-parallel pruning-quantization. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7873–7882. Computer Vision Foundation / IEEE Computer Society, 2018.

[8] Berivan Isik, Kristy Choi, Xin Zheng, Tsachy Weissman, Stefano Ermon, H.-S. Philip Wong, and Armin Alaghi. Neural network compression for noisy storage devices. *CoRR*, abs/2102.07725, 2021.

[9] Sein Park, Junhyuk So, Juncheol Shin, and Eunhyeok Park. NIPQ: noise injection pseudo quantization for automated DNN optimization. *CoRR*, abs/2206.00820, 2022.

[10] Pierre Stock, Angela Fan, Benjamin Graham, Edouard Grave, Rémi Gribonval, Hervé Jégou, and Armand Joulin. Training with quantization noise for extreme model compression. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[11] Ivan Lazarevich, Alexander Kozlov, and Nikita Malinin. Post-training deep neural network pruning via layer-wise calibration. In *IEEE/CVF International Conference on Computer Vision Workshops, ICCVW 2021, Montreal, BC, Canada, October 11-17, 2021*, pages 798–805. IEEE, 2021.

[12] Markus Nagel, Rana Ali Amjad, Mart van Baalen, Christos Louizos, and Tijmen Blankevoort. Up or down? adaptive rounding for post-training quantization. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 7197–7206. PMLR, 2020.

[13] Markus Nagel, Mart van Baalen, Tijmen Blankevoort, and Max Welling. Data-free quantization through weight equalization and bias correction. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 1325–1334. IEEE, 2019.

[14] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *CoRR*, abs/1503.02531, 2015.

[15] Lucas Liebenwein, Alaa Maalouf, Dan Feldman, and Daniela Rus. Compressing neural networks: Towards determining the optimal layer-wise decomposition. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 5328–5344, 2021.

[16] Andrey Kuzmin, Mart van Baalen, Markus Nagel, and Arash Behboodi. Quantized sparse weight decomposition for neural network compression. *CoRR*, abs/2207.11048, 2022.

[17] Zejiang Hou, Minghai Qin, Fei Sun, Xiaolong Ma, Kun Yuan, Yi Xu, Yen-Kuang Chen, Rong Jin, Yuan Xie, and Sun-Yuan Kung. CHEX: channel exploration for CNN model compression. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 12277–12288. IEEE, 2022.

[18] Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both weights and connections for efficient neural networks. *CoRR*, abs/1506.02626, 2015.

[19] Shixing Yu, Zhewei Yao, Amir Gholami, Zhen Dong, Sehoon Kim, Michael W. Mahoney, and Kurt Keutzer. Hessian-aware pruning and optimal neural implant. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2022, Waikoloa, HI, USA, January 3-8, 2022*, pages 3665–3676. IEEE, 2022.

[20] Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li-Jia Li, and Song Han. AMC: automl for model compression and acceleration on mobile devices. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part*

*VII*, volume 11211 of *Lecture Notes in Computer Science*, pages 815–832. Springer, 2018.

[21] Sixing Yu, Arya Mazaheri, and Ali Jannesari. Auto graph encoder-decoder for neural network pruning. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 6342–6352. IEEE, 2021.

[22] Sixing Yu, Arya Mazaheri, and Ali Jannesari. Topology-aware network pruning using multi-stage graph embedding and reinforcement learning. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvári, Gang Niu, and Sivan Sabato, editors, *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 25656–25667. PMLR, 2022.

[23] Dongqing Zhang, Jiaolong Yang, Dongqiangzi Ye, and Gang Hua. Lq-nets: Learned quantization for highly accurate and compact deep neural networks. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part VIII*, volume 11212 of *Lecture Notes in Computer Science*, pages 373–390. Springer, 2018.

[24] Karen Ullrich, Edward Meeds, and Max Welling. Soft weight-sharing for neural network compression. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[25] Eunhyeok Park, Junwhan Ahn, and Sungjoo Yoo. Weighted-entropy-based quantization for deep neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 7197–7205. IEEE Computer Society, 2017.

[26] Nilesh Prasad Pandey, Markus Nagel, Mart van Baalen, Yin Huang, Chirag Patel, and Tijmen Blankevoort. A practical mixed precision algorithm for post-training quantization. *CoRR*, abs/2302.05397, 2023.

[27] Yiren Zhou, Seyed-Mohsen Moosavi-Dezfooli, Ngai-Man Cheung, and Pascal Frossard. Adaptive quantization for deep neural network. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pages 4596–4604. AAAI Press, 2018.

[28] Zhewei Yao, Zhen Dong, Zhangcheng Zheng, Amir Gholami, Jiali Yu, Eric Tan, Leyuan Wang, Qijing Huang, Yida Wang, Michael W. Mahoney, and Kurt Keutzer. HAWQ-V3: dyadic neural network quantization. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 11875–11886. PMLR, 2021.

[29] Zhen Dong, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. HAWQ: hessian aware quantization of neural networks with mixed-precision. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 293–302. IEEE, 2019.

[30] Zhen Dong, Zhewei Yao, Daiyaan Arfeen, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. HAWQ-V2: hessian aware trace-weighted quantization of neural networks. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[31] Lirui Xiao, Huanrui Yang, Zhen Dong, Kurt Keutzer, Li Du, and Shanghang Zhang. CSQ: growing mixed-precision quantization scheme with bi-level continuous sparsification. *CoRR*, abs/2212.02770, 2022.

[32] Yuexiao Ma, Taisong Jin, Xiawu Zheng, Yan Wang, Huixia Li, Guannan Jiang, Wei Zhang, and Rongrong Ji. OMPQ: orthogonal mixed precision quantization. *CoRR*, abs/2109.07865, 2021.

[33] Yoojin Choi, Mostafa El-Khamy, and Jungwon Lee. Towards the limit of network quantization. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[34] Chaim Baskin, Brian Chmiel, Evgenii Zheltonozhskii, Ron Banner, Alex M. Bronstein, and Avi Mendelson. CAT: compression-aware training for bandwidth reduction. *J. Mach. Learn. Res.*, 22:269:1–269:20, 2021.

[35] Brian Chmiel, Chaim Baskin, Evgenii Zheltonozhskii, Ron Banner, Yevgeny Yermolin, Alex Karbachevsky, Alex M. Bronstein, and Avi Mendelson. Feature map transform coding for energy-efficient CNN inference. In *2020 International Joint Conference on Neural Networks, IJCNN 2020, Glasgow, United Kingdom, July 19-24, 2020*, pages 1–9. IEEE, 2020.

[36] Simon Wiedemann, Heiner Kirchhoffer, Stefan Matlage, Paul Haase, Arturo Marbán, Talmaj Marinc, David Neumann, Ahmed Osman, Detlev Marpe, Heiko Schwarz, Thomas Wiegand, and Wojciech Samek. Deepcabac: Context-adaptive binary arithmetic coding for deep neural network compression. *CoRR*, abs/1905.08318, 2019.

[37] Simon Wiedemann, Heiner Kirchhoffer, Stefan Matlage, Paul Haase, Arturo Marbán, Talmaj Marinc, David Neumann, Tung Nguyen, Heiko Schwarz, Thomas Wiegand, Detlev Marpe, and Wojciech Samek. Deepcabac: A universal compression algorithm for deep neural networks. *IEEE J. Sel. Top. Signal Process.*, 14(4):700–714, 2020.

[38] Xiuying Wei, Ruihao Gong, Yuhang Li, Xianglong Liu, and Fengwei Yu. Qdrop: Randomly dropping quantization for extremely low-bit post-training quantization. *arXiv preprint arXiv:2203.05740*, 2022.

[39] Vladimir Chikin and Mikhail Antiukh. Data-free network compression via parametric non-uniform mixed precision quantization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 450–459. IEEE, 2022.

[40] Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *CoRR*, abs/1308.3432, 2013.

[41] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255. IEEE Computer Society, 2009.

[42] Glenn Jocher et. al. ultralytics/yolov5: v6.0 - YOLOv5n 'Nano' models, Roboflow integration, TensorFlow export, OpenCV DNN support, October 2021.

[43] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755. Springer, 2014.

[44] Robert Bamler. Understanding entropy coding with asymmetric numeral systems (ans): a statistician's perspective. *arXiv preprint arXiv:2201.01741*, 2022.