

MEC-Quant: Maximum Entropy Coding for Extremely Low Bit Quantization-Aware Training

Junbiao Pang, Tianyang Cai, Baochang Zhang

Abstract—Quantization-Aware Training (QAT) has driven much attention to produce efficient neural networks. Current QAT still obtains inferior performances compared with the Full Precision (FP) counterpart. In this work, we argue that quantization inevitably introduce biases into the learned representation, especially under the extremely low-bit setting. To cope with this issue, we propose Maximum Entropy Coding Quantization (MEC-Quant), a more principled objective that explicitly optimizes on the structure of the representation, so that the learned representation is less biased and thus generalizes better to unseen in-distribution samples. To make the objective end-to-end trainable, we propose to leverage the minimal coding length in lossy data coding as a computationally tractable surrogate for the entropy, and further derive a scalable reformulation of the objective based on Mixture Of Experts (MOE) that not only allows fast computation but also handles the long-tailed distribution for weights or activation values. Extensive experiments on various tasks on computer vision tasks prove its superiority. With MEC-Qaunt, the limit of QAT is pushed to the x-bit activation for the first time and the accuracy of MEC-Quant is comparable to or even surpass the FP counterpart. Without bells and whistles, MEC-Qaunt establishes a new state of the art for QAT. Our code is available at this [https URL](https://url) and has been integrated into MQBench (this [https URL](https://url))

Index Terms—Quantization, Maximum Entropy Principle, Long-tailed Distribution, Mixed of Expert

I. INTRODUCTION

WITH the rapid development of deep learning and the rapid growth of massive data, artificial intelligence has concentrated on the two poles of edge computing model and cloud model. However, the high training and storage costs have hindered the production and deployment of deep learning models, and model compression has therefore attracted great attention. Model compression techniques are further divided into neural architecture search [1], network pruning [2], and quantization [3], [4]. Among these methods, quantization plays an important role in reducing memory consumption and accelerating inference due to low bit weights and activations.

According to the different optimization objects, quantization can be divided into Post-Training Quantization (PTQ) [5]–[7] and Quantization-Aware Training (QAT) [8], [9]. PTQ only uses a calibration set composed of a small portion of the training set for fine-tuning, without the need to retrain the entire training set. Therefore, PTQ is very concise and efficient, but

J. Pang, T. Cai and are with the Faculty of Information Technology, Beijing University of Technology, Beijing 100124, China (e-mail: junbiao_pang@bjut.edu.cn).

B. Zhang is with the University of Chinese Academy of Sciences, Chinese Academy of Sciences (CAS), Beijing 100049, China, and the Institute of Computing Technology, CAS, Beijing 100190, China (email: bczhang@139.com).

it also faces serious accuracy loss at extremely low bits. QAT uses the entire training set for retrain, simulating errors caused by quantization during inference while optimizing weights and quantization parameters. It still maintains good performance at extremely low bit. Therefore, at extremely low bit, QAT is widely used in both academia and industry.

However, current QAT still obtains inferior performances compared with the Full Precision (FP) counterpart. In this work, we argue that quantization inevitably introduce biases into the learned representation, especially under the extremely low-bit setting. In this work, we believe that this is because of the quantization operation itself causing bias in the learned representation, leading to performance degradation. This deviation will accumulate in the backbone, resulting in cumulative errors that are reflected in downstream tasks. To solve this problem, we propose Maximum Entropy Coding Quantization (MEC-Quant), a more principled objective that explicitly optimizes on the structure of the representation, so that the learned representation is less biased and thus generalizes better to unseen in-distribution samples.

However, due to the fact that features in deep learning are high-dimensional vectors, it is very difficult to directly calculate feature entropy and the computational complexity is relatively high. To solve this problem, we propose to leverage the minimal coding length in lossy data coding as a computationally tractable surrogate for the entropy [10]. The log-determinant term costs the most computation in the coding length function. By using Taylor expansion on the matrix, we further accelerate and estimate it. As shown in Fig, high dimensional features usually have the characteristic of long-tailed distribution, and Taylor expansion at a certain point alone cannot meet the accuracy of estimation. To address this issue, we propose using Mixture Of Experts (MOE) for multi-point expansion, which not only ensures estimation accuracy but also speeds up operations. Our contributions are as follow:

- We believe that the quantization operation itself during the quantization process can cause bias in the learned representation, leading to performance degradation. We propose Maximum Entropy Coding Quantization (MEC-Quant), a more principled objective that explicitly optimizes on the structure of the representation, so that the learned representation is less biased and thus generalizes better to unseen in-distribution samples.
- To accelerate the calculation and accuracy of feature entropy, we propose to leverage the minimal coding length in lossy data coding as a computationally tractable surrogate for the entropy. To address the issue of inaccurate estimation caused by the long tail distribution

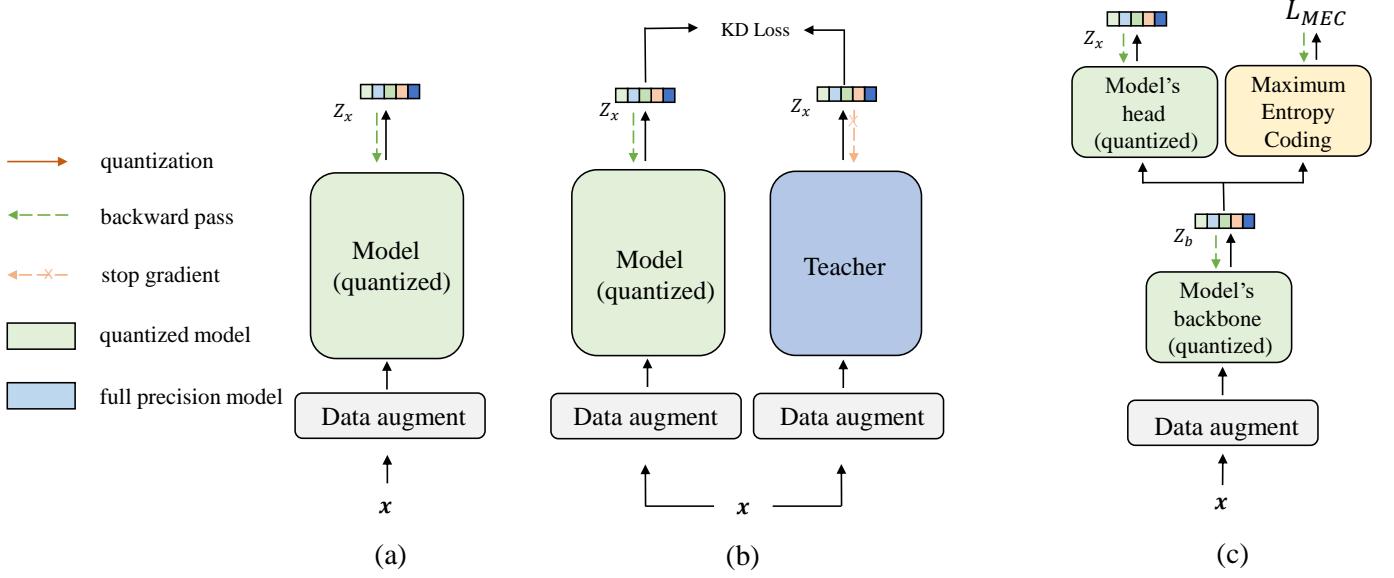


Fig. 1. Comparison between vanilla QAT, quantization with KD [11] and our method. (a) the overview of vanilla QAT. (b) the overview of quantization with KD. (c) the overview of our method.

phenomenon in high-dimensional features, we derive a scalable reformulation of the objective based on MOE that not only allows fast computation but also handles the long-tailed distribution for weights or activation values.

- We propose a training paradigm for QAT based on MEC named MEC-Qaunt. The proposed approach, a simple, novel, yet powerful method, is easily adapted to different neural networks. MEC-Qaunt pushes the performances of QAT models towards, and even surpasses that of FP32 counterparts. Extensive experiments on different models prove that our method set up a new State-Of-The-Art (SOTA) method for QAT.

II. RELATED WORK

Quantization-Aware Training. The idea of QAT is to minimize quantization errors by the complete training dataset, enabling the network to consider task-related loss. There are two research directions in QAT: 1) one primarily focuses on addressing quantization-specific issues, such as gradient backpropagation caused by rounding operations [12], [13] and weight oscillations during the training stage [9]; 2) the other integrates quantization with other training paradigms.

For the first category, STE [12] employed the expected probability of stochastic quantization as the derivative value in backpropagation. STE assumes the derivative of the rounding operation is defaulted to 1, failing to accurately reflect the actual quantization error. EWGS [13] adaptively scaled quantized gradients based on quantization error, compensating for the gradient. PSG [14] scaled gradients based on the position of the weight vector, essentially providing gradient compensation. DiffQ [15] found that STE can cause weight oscillations during training and thus employed additive Gaussian noise to simulate quantization noise. It is overcoming Oscillations Quantization [9] addressed oscillation issues by introducing a regularization term that encourages latent weights to be close

to the center of the bin. ReBNN [16] introduced weighted reconstruction loss to establish an adaptive training objective. The balance parameter associated with the reconstruction loss controls the weight oscillations.

In addition to addressing quantization-specific issues, some approaches integrate quantization with other training paradigms. SSQL [17] unified self-supervised learning and quantization to learn quantization-friendly visual representations during pre-training. [18] combined unsupervised learning and binary quantization to improve network compression and maintain model performance. [19] effectively combined quantization and distillation, inducing the training of lightweight networks with solid performance.

Maximum Entropy for Quantization. Q-ViT [20] believes that if one wants to restore the feature representation of FP model, the mutual information between the quantized model and the FP model should be maximized, so the quantized model should have the maximum information entropy. This work aligns with the goal of our work, but they assume that the output is Gaussian distribution and directly optimize the maximum entropy of the Gaussian distribution. This is clearly suboptimal, as high-dimensional features are not Gaussian distributions and are closer to long-tailed distributions.

N2UQ [21] also believes that more information are preserved when quantified quantized contain higher entropy. It applies a regularization term to the weights and optimizes them to give the quantized weights greater information entropy. The optimization object of this work is quantized weights, while our optimization object is the learned representation

III. METHODOLOGY

A. Notation and Background

Basic Notations. In this paper, notation \mathbf{X} represents a matrix (or tensor), the vector is denoted as \mathbf{x} , labeled data

is \mathbf{x}_l , and unlabeled data is \mathbf{x}_u . $S(\mathbf{x}; \mathbf{w})$ represents the model $S(\cdot; \cdot)$ with the parameter \mathbf{w} and the input \mathbf{x} .

For a neural network with activation, we denote the loss function as $L(\mathbf{w}, \mathbf{x})$, where \mathbf{w} and \mathbf{x} represent the network's weights and input, respectively. Note that we assume \mathbf{x} is sampled from the training set \mathcal{D}_t , thus the final loss is defined as $\mathbb{E}_{\mathbf{x} \sim \mathcal{D}_t} [L(\mathbf{w}, \mathbf{x})]$.

Quantization. Quantization parameters steps s and zero points z serve as a bridge between floating-point and fixed-point representations. Given the input tensor \mathbf{x} , the quantization operation is as follows:

$$\begin{aligned}\mathbf{x}_{int} &= clip\left(\lfloor \frac{\mathbf{x}}{s} \rfloor + z, 0, 2^q - 1\right) \\ \hat{\mathbf{x}} &= (\mathbf{x}_{int} - \mathbf{z}) s,\end{aligned}\quad (1)$$

where $\lfloor \cdot \rfloor$ represents the rounding-to-nearest operator, q is the predefined quantization bit-width, s denotes the step size between two subsequent quantization levels. \mathbf{z} stands for the zero-points. The s and z is initialized through forward propagation by a calibration set \mathcal{D}_c ($\mathcal{D}_c \in \mathcal{D}_t$) from the training dataset \mathcal{D}_t .

$$s = \frac{\mathbf{x}_{max} - \mathbf{x}_{min}}{2^q - 1}. \quad (2)$$

Different from LSQ [8], the quantization parameter s is calibrated by samples. Therefore, the loss function of a quantized model is given as follows:

$$\arg \min_{\hat{\mathbf{w}}} \mathbb{E}_{\mathbf{x} \sim \mathcal{D}_t} [L(\hat{\mathbf{w}}, \mathbf{x})], \quad (3)$$

where $\hat{\mathbf{w}}$ is the quantized weight obtained from the latent weight \mathbf{w} after quantization by (1).

B. Feature Collapse for low bit quantization

Singular value decomposition has been widely used to measure the collapse phenomenon (Jing et al., 2021). In Figure xxx, we have shown that the learned feature matrices Z of the last layer are approximately low-rank with ??? some extremely small singular values. To determine the degree of collapse for such matrices with low-rank tendencies, we propose the rectified eigenvalue entropy as a generalized quantification as follows:

$$H(x) = -\frac{rank(Z)}{N_{rank}} \sum_i \frac{\sigma_i}{\|\sigma_i\|} \log \frac{\sigma_i}{\|\sigma_i\|} \quad (4)$$

the feature matrix should be diverse [] (full rank) and representative [] (the max eigenvalue / min eigenvalue = 1).

We conduct experiments on the CIFAR-10 dataset with the Quant-mec method, LSQ method, and full-precision ResNet-18 model. We randomly use 500 images from the CIFAR-10 dataset as test samplesWe use (4) to measure the numerical values of feature collapse for Quant-MEC, LSQ, and full-precision models. The experimental results are shown in Tab. I. The results indicate that the eigenvalues of features in the full-precision model are more evenly distributed, implying better feature representation. The eigenvalue distribution of the LSQ model is more concentrated, meaning only a few eigenvalues play a more critical role, facing the phenomenon of feature collapse. However, Quant-MEC alleviates this phenomenon.

TABLE I
NUMERICAL COMPARISON OF FEATURE COLLAPSE AMONG QUANT-MEC, LSQ AND FULL-PRECISION MODEL.

	Quant-MEC	LSQ
Full Prec.	8.598 +- 0.000406	
W4A4	7.812+-0.000008	6.220+-0.000067
W2A4	7.163+-0.000089	6.404+-0.000139
W2A2	7.804+-0.000124	6.497+-0.000174

Intuitively, a matrix with high information abundance demonstrates a balanced distribution in vector space since it has similar singular values. In contrast, a matrix with low information abundance suggests that the components corresponding to smaller singular values can be compressed without significantly impacting the result. yet it is applicable for non-strictly low-rank matrices, especially for fields with $D_i >> K$ which is possibly of rank K . We calculate the information abundance of embedding matrices for the enlarged DCNv2 (Wang et al., 2021) and compare it with that of randomly initialized matrices, shown in Figure 2. It is observed that the information abundance of learned embedding matrices is extremely low, indicating the embedding collapse phenomenon.

C. Maximum Entropy Coding

The framework of MEC-Quant is shown in Fig.1(c). The model $S(\mathbf{x}; \mathbf{w})$ is the ones that we aim to quantize. It is first initialized with the calibration dataset.

In information theory, entropy for continuous variables is usually defined as: $H(\mathbf{z}) = -\int p(\mathbf{z}) \log p(\mathbf{z}) dz$. In deep learning, the learned representation are usually high-dimensional vectors: $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_m]$. However, it is very difficult to estimate the true distributions [22], [23] $p(\mathbf{Z})$ from the learned representation \mathbf{Z} . A handy fact is that entropy is conceptually equivalent to the minimal number of bits required to encode the data losslessly, so the minimal lossless coding length could be used to represent the entropy. However, lossless coding of continuous random variables is infeasible in our case since it often requires an infinite number of bits, breaking the numerical stability. Instead, we exploit the coding length in lossy data coding [10] as a computationally tractable surrogate for the entropy of continuous random variables. For the learned representation Z , the minimal number of bits needed to encode Z subject to a distortion ϵ is given by the following coding length function:

$$L = \left(\frac{m+d}{2} \right) \log \det \left(\mathbf{I}_m + \frac{d}{m\epsilon^2} \mathbf{Z}^\top \mathbf{Z} \right), \quad (5)$$

where \mathbf{I}_m denotes the identity matrix with dimension m , and ϵ is the upper bound of the expected decoding error between $\mathbf{z} \in \mathbf{Z}$ and the decoded $\hat{\mathbf{z}}$, i.e., $\mathbb{E}[\|\mathbf{z} - \hat{\mathbf{z}}\|_2] \leq \epsilon$.

Due to the high computational cost of the log-determinant of the high-dimensional matrix in (5) and the potential for numerical instability of the ill conditioned matrix, we need to

rewrite it. According to [24], (5) can be rewrite by Taylor expansion as:

$$L = \text{Tr} \left(\mu \sum_{k=1}^{\infty} \frac{(-1)^{k+1}}{k} (\lambda \mathbf{Z}^\top \mathbf{Z})^k \right), \quad (6)$$

where $\mu = \frac{m+d}{2}$ and $\lambda = \frac{d}{me^2}$, with convergence condition: $\|\lambda \mathbf{Z}^\top \mathbf{Z}\|_2 < 1$.

D. MoE and Gating

We believe that in deep learning, features typically exhibit a long-tailed distribution, and for (6), it only unfolds at zero and cannot satisfy convergence conditions. Therefore, in order to address the problems faced by long-tailed distributions, we propose using the MoE mechanism to perform multi-point expansion on (6).

In our work, each expert corresponds to the Taylor expansion of (6) at a certain point, and we assume that $f(\mathbf{X}) = \log(\mathbf{I}_m + \mathbf{X})$, where $\mathbf{X} = \frac{d}{me^2} \mathbf{Z}^\top \mathbf{Z}$, $\mathbf{X} \in \mathbb{R}^{m \times m}$, \mathbf{A} is the unit matrix with dimension $m \times m$. The n-th derivative of $f(\mathbf{X})$ is: $f^{(n)}(\mathbf{X}) = (-1)^{n+1} \frac{(n-1)!}{(1+x)^n}$. The Taylor expansion of $f(\mathbf{X})$ at $\mathbf{X} = a \cdot \mathbf{A}$ is:

$$f(\mathbf{X}) = f(a \cdot \mathbf{A}) + \sum_{k=1}^{\infty} \frac{f^{(k)}(a)}{k!} (\mathbf{X} - a \cdot \mathbf{A})^k. \quad (7)$$

Therefore, (6) can be rewritten as:

$$L_i(\mathbf{X}; a_i, k) = \text{Tr} \left(\mu \left(f(a_i \cdot A) + \sum_{k=1}^{\infty} \frac{f^{(k)}(a_i)}{k!} (X - a_i \cdot A)^k \right) \right), \quad (8)$$

where L_i represents the loss calculated by the i-th expert, and $\mathbf{X} = a \cdot \mathbf{A}$ represents the expansion point responsible by the i-th expert.

How to allocate the corresponding weights for each expert. We introduce the Gating Network mechanism, which is responsible for processing the weight coefficients of each expert.

$$\begin{aligned} L_{MEC}(\mathbf{Z}) &= \sum_{i=1}^n (G(\mathbf{Z})_i L_i(\mathbf{X}; a_i, k)) \\ G(\mathbf{Z}) &= \text{Softmax}(W_g(\mathbf{Z})), \end{aligned} \quad (9)$$

where $G(\cdot)$ represents the gating network, W_g is the corresponding parameter of the gated network, n represents the number of experts in MoE, $\mathbf{X} = \frac{d}{me^2} \mathbf{Z}^\top \mathbf{Z}$. After Softmax processing, \mathbf{Z} is assigned weights to each expert. The final calculation result of MoE is the weighted sum of n expert network outputs.

E. Training Processing of MEC-Quant

In this paper, we focus on the classification task. Therefore, the whole loss consists of two components as follows:

$$L_{total}(\hat{\mathbf{w}}, \mathbf{x}) = \text{task_loss}(\mathbf{Z}_o) + \lambda L_{MEC}(\mathbf{Z}_b), \quad (10)$$

where \mathbf{Z}_o presents the output from the model for the labeled data (\mathbf{x}, y) , \mathbf{Z}_b presents the output from the backbone of model, weight λ ($\lambda > 0$) balances between the CE loss and MEC loss in (10).

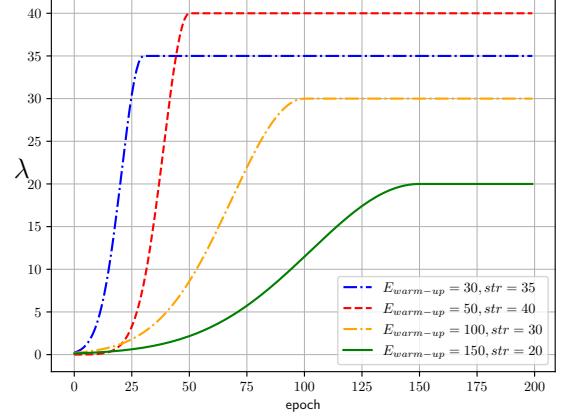


Fig. 2. Different settings of the strength (λ) of MEC

In this work, we focus on classification tasks, and there are two possible case for task loss:

$$\text{task_loss}(\mathbf{Z}_o) = CE(\mathbf{Z}_o, y), \quad (11)$$

where $CE(\cdot, \cdot)$ denotes the cross-entropy loss for classification tasks, y represents the annotation information of the data. In this case, we use the annotations of the data as supervised information. Another case is:

$$\text{task_loss}(\mathbf{Z}_o) = KL(\mathbf{Z}_o, \mathbf{Z}_t), \quad (12)$$

where $KL(\cdot, \cdot)$ represents the calculation of KL divergence, \mathbf{Z}_t is the output of the FP model. In this case, we use the features of the full precision model as supervised information. In this case, MEC-Quant is a label free quantification.

In order to avoid trivial solutions, the model should have a certain prior information as a guarantee that the model learns in the direction related to downstream tasks, or in other words, the model has its own main driving force. In (11), the annotation information y is our prior information, and annotation y ensures that the quantization model learns in the correct direction for classification. In (12), the feature representation of the FP model is our prior information, and the fully accurate feature representation ensures that the quantized model learns from the FP feature space. Therefore, MEC constraints can be seen as an auxiliary regularization.

The weight λ is very important in (10). We hope that in the early stage of QAT training, the strength of MEC constraints should be as small as possible, with the aim of allowing the model to fully learn the priors related to downstream tasks. In the later stage of QAT training, the strength of MEC constraints should increase in order to enable the model to have greater feature entropy while maintaining performance. In summary, the intensity of MEC constraints should gradually increase with the training rounds, reaching its maximum intensity in the middle stage and remaining constant in the later stages of training. In this paper, λ is progressively increased as follows:

$$\lambda = \text{str} \times e^{-5 \times \left[1 - \left(\frac{\beta}{E_{\text{warm-up}}} \right)^2 \right]}, \quad (13)$$

Algorithm 1 MEC-Quant

- 1: **Input:** labeled data \mathbf{x} ; quantized model $S(\mathbf{x}; \mathbf{w})$; iteration T .
 - 2: **for** $i = 1$ **to** T **do**
 - 3: Preprocess the input data;
 - 4: Feed the data into the quantized model, get the final output of the model \mathbf{Z}_o and the output of the backbone \mathbf{Z}_b ;
 - 5: Calculate the MEC Loss and the task loss based on (9) and (10);
 - 6: Update for student model's weights parameter and quantization parameter;
 - 7: **end for**
 - 8: **Output:** quantized model.
-

where str represents the intensity of MEC, $E_{\text{warm-up}}$ represents the predetermined hyperparameter of warm-up epoch, and β is:

$$\beta = \text{clip}(t, 0, E_{\text{warm-up}}), \quad (14)$$

where t represents the current epoch.

As illustrated in Fig. 2, a small λ value in (13) can ensure that the model learn from the ground truth in the early stages of training, in order to prevent the MEC constraints in the later stages from obtaining the trivial solution. The λ value gradually increases, and while ensuring that task loss is dominant, the entropy of the features gradually increases. Ultimately, a stable λ ensures that MEC constraints play a role correctly. Our overall training algorithm is shown in algorithm 1. The overall algorithm is very simple and effective.

IV. EXPERIMENTS

Experimental Protocols. Our code is based on PyTorch [25] and relies on the MQBench [26] package. By default, we set the β increase linearly from 0 to 5 as the number of epochs increases unless explicitly mentioned otherwise. The default data augmentation includes random horizontal flipping and random translation. As for quantization, we use the default method of asymmetric quantization. For the cifar-10 and cifar-100 datasets, we use 100 images for calibration. We also keep the first and last layers with 8-bit quantization, which is same as QDrop [7]. Additionally, we employ per-channel quantization for weight quantization. We use WXAX to represent X-bit weight and activation quantization.

Two experimental settings are evaluated as follows:

- (A) We use annotation information y as supervised information. At this time, the annotation information y is the prior information of MEC-Quant, as illustrated in (11).
- (B) We use the output feature \mathbf{Z}_t of the FP model as supervised information. At this time, the output feature \mathbf{Z}_t of the FP model is the prior information of MEC-Quant, as illustrated in (12).

Training Details. We use SGD as the optimizer, with a batch size of 256 and a base learning rate of 0.01. The default learning rate (LR) scheduler follows the cosine annealing method. The weight decay is 0.0005, and the SGD momentum

is 0.9. We train for 200 epochs on CIFAR-10 and CIFAR-100 unless otherwise specified.

A. Ablation Study

TABLE II
ABLATION STUDIES OF THE CHOICE OF MEC-QUANT (ACCURACY %) ON CIFAR-10.

MEC	MoE	ResNet-18	MobileNetV2
		88.36	78.15
✓		88.90	81.03
✓	✓	88.98	81.20

Effectiveness of MEC: To verify the effectiveness of MEC and MoE, we used W2A4 quantization configuration on CIFAR-10 dataset. In Tab. II, we used LSQ as our baseline, with ResNet-18 [27] having an accuracy of 88.36% and MobileNetV2 [28] having an accuracy of 78.15%. When using MEC validation experiments, we use (6) as our regularization constraint and use 4th order Taylor expansion. When using MoE in combination, we reduced the Taylor expansion of MEC (9) to 2th order to reduce computational complexity and address issues arising from long-tailed distributions.

B. Parameter Configuration

In this section, We investigate the effect of hyperparameters on MEC-Quant experiments.

TABLE III
HYPERPARAMETER COMPARATIVE EXPERIMENT OF THE STRENGTH OF MEC ON CIFAR-10.

The strength of MEC.	Val Acc(%)
Constant 1 (baseline)	87.85
(13), where str=0.5, $E_{\text{warm-up}} = 50$	88.07
(13), where str=5, $E_{\text{warm-up}} = 50$	88.26
(13), where str=10, $E_{\text{warm-up}} = 50$	88.17
(13), where str=5, $E_{\text{warm-up}} = 100$	88.02
(13), where str=5, $E_{\text{warm-up}} = 30$	88.17

The Strength of MEC. MEC constraint strength builds a bridge between task loss and MEC constraints. We quantized the ResNet18 model using W2A4 on the CIFAR-10 dataset, and the results are shown in Tab. III. As mentioned in section III-E, MEC constraints can be seen as an auxiliary regularization. In the early stage of QAT training, the strength of MEC constraints should be as small as possible, with the aim of allowing the model to fully learn downstream task related priors. In the later stage of QAT training, the strength of MEC constraints should increase in order to enable the model to have greater feature entropy while maintaining performance.

Taylor expansion order. We quantized the ResNet18 model using W2A2 on the CIFAR-10 dataset, and the results are shown in Tab. IV. When using (6), as the order of Taylor expansion increases, the model performance improves, but at the same time, it is accompanied by a larger computational load. When using (9), we found that using only the Taylor expansion with k=2 can achieve the effect of k=4 in (6).

TABLE IV
HYPERPARAMETER COMPARATIVE EXPERIMENT OF TAYLOR EXPANSION METHOD ON CIFAR-10.

Taylor expansion method.	Val Acc(%)
(6), where $k = 4$	89.85
(6), where $k = 2$	89.67
(9), where $k = 2$	89.98

C. Literature Comparison

TABLE V
COMPARISON AMONG DIFFERENT QAT STRATEGIES REGARDING ACCURACY ON CIFAR-10.

Labeled data	Methods	W/A	Res18	Res50	MBV1	MBV2
50000	Full Prec.	32/32	88.72	89.95	85.52	85.81
	PACT [29]	4/4	88.15	85.27	80.77	79.88
	LSQ [8]	4/4	88.69	90.01	82.39	84.45
	LSQ+ [30]	4/4	88.40	90.30	84.32	84.30
	KD [31]	4/4	88.86	90.34	84.77	83.79
	MEC-Quant (Setting (A))	4/4	90.22	91.25	85.44	85.52
	MEC-Quant (Setting (B))	4/4	90.18	90.88	85.32	85.57
50000	PACT [29]	2/4	87.55	85.24	69.04	67.18
	LSQ [8]	2/4	88.36	90.01	78.15	78.15
	LSQ+ [30]	2/4	87.76	89.62	81.26	77.00
	KD [31]	2/4	88.83	90.18	78.84	75.56
	MEC-Quant (Setting (A))	2/4	89.76	91.11	80.79	81.20
	MEC-Quant (Setting (B))	2/4	89.98	91.23	80.55	80.96
	PACT [29]	2/2	76.90	64.94	11.71	10.58
	LSQ [8]	2/2	87.60	87.79	75.29	70.32
	LSQ+ [30]	2/2	87.60	86.10	74.22	72.18
	KD [31]	2/2	88.06	89.20	68.62	67.15
	MEC-Quant (Setting (A))	2/2	88.51	89.15	75.86	72.87
	MEC-Quant (Setting (B))	2/2	88.26	89.32	76.02	73.77

CIFAR-10. We selected ResNet-18 and -50 [27] with normal convolutions, MobileNetV1 [32] and V2 [28] with depthwise separable convolutions as our experimental models. In Tab. V, we quantized the weights to 2-bit and the activations to 2-bit. We compared our approach with effective baselines, including LSQ [8], LSQ+ [30], PACT [29], and KD [31].

As shown in Tab. V, in setting (A), MEC-Quant showed significant improvements compared to the baselines. In W4A4 quantization, MEC-Quant achieved about 1~2% accuracy improvements over LSQ. In W2A4 quantization, MEC-Quant achieved nearly 3% accuracy improvements over LSQ of MobilenetV2 model. Furthermore, to explore the ability of MEC-Quant, we conducted W2A2 quantization experiment. In W2A2 quantization, the advantages of MEC-Quant are more obvious. It achieved 3.4% accuracy improvements over LSQ of MobilenetV2 model. A similar situation occurred in setting (B). The accuracy of setting (B) is very close to that of setting (A).

CIFAR-100 In this section, we investigated the effectiveness of MEC-Quant on the CIFAR-100 dataset. We employed ResNet-18, ResNet-50, MobileNetV1 and MobileNetV2 as the experimental models. In Tab. VI, we quantized the weights to 2-bit and the activations to 2-bit. Similar to CIFAR-10, We compared our approach with effective baselines, including LSQ [8], LSQ+ [30], PACT [29], and KD [31].

Similar to CIFAR-10, MEC-Quant showed significant improvements compared to the baselines. As shown in Tab6, In W4A4 quantization, MEC-Quant achieved about $x \sim x\%$ accuracy improvements over LSQ. In W2A4 quantization,

TABLE VI
COMPARISON AMONG DIFFERENT QAT STRATEGIES REGARDING ACCURACY ON CIFAR-100.

Labeled data	Methods	W/A	Res18	Res50	MBV1	MBV2
50000	Full Prec.	32/32	75.40	78.94	70.22	71.30
	PACT [29]	4/4	74.17	74.78	64.65	64.06
	LSQ [8]	4/4	75.30	78.20	68.63	69.01
	LSQ+ [30]	4/4	74.50	77.39	67.89	68.25
	KD [31]	4/4	74.70	78.80	70.96	71.66
	MEC-Quant (Setting (A))	4/4	75.65	78.65	68.75	68.51
	MEC-Quant (Setting (B))	4/4	75.98	79.50	70.11	70.33
50000	PACT [29]	2/4	73.77	74.72	49.98	57.90
	LSQ [8]	2/4	74.93	77.82	65.13	66.15
	LSQ+ [30]	2/4	73.90	76.61	65.28	66.24
	KD [31]	2/4	74.35	77.34	66.90	63.77
	MEC-Quant (Setting (A))	2/4	75.2	77.72	65.97	65.00
	MEC-Quant (Setting (B))	2/4	75.79	79.22	66.75	64.61
	PACT [29]	2/2	65.16	4.26	3.25	8.39
	LSQ [8]	2/2	71.80	62.79	55.53	31.08
	LSQ+ [30]	2/2	71.25	84.21	55.56	30.08
	KD [31]	2/2	73.13	63.16	55.37	28.56
	MEC-Quant (Setting (A))	2/2	73.57	64.8	58.98	34.54
	MEC-Quant (Setting (B))	2/2	75.15	71.54	62.99	42.54

MEC-Quant achieved nearly $x \sim x\%$ accuracy improvements over LSQ of MobilenetV2 model. Furthermore, to explore the ability of MEC-Quant, we conducted W2A2 quantization experiment. In W2A2 quantization, the advantages of MEC-Quant are more obvious. It achieved 3.4% accuracy improvements over LSQ of MobilenetV2 model.

D. Generalization of MEC-Quant

TABLE VII
A COMPARISON OF THE HESSIAN EIGENVALUES OF THE MODELS TRAINED BY THE MEC-QUANT AND LSQ METHODS UNDER W2A2.

method	max eigenvalue	mean eigenvalue
MEC-Quant	155.73	18.7
LSQ	520.24	61.65

It has been empirically pointed out that the dominant eigenvalue of $\nabla^2 L_{\text{val}}(\mathbf{w})$ (spectral norm of Hessian) is highly correlated with the generalization quality of QAT solutions [33] [34]. In standard QAT training, the Hessian norm is usually great, which leads to deteriorating (test) performance of the solutions. We calculated the eigenvalues of the Hessian matrices of MEC-Quant and the comparison method LSQ under W2A2, and obtained the results shown in Tab. VII. The maximum eigenvalue of MEC-Quant is 155.73, which is much lower than 520.24 of LSQ. The average value of the eigenvalues of MEC-Quant is 18.7, which is also much lower than 61.65 of LSQ. Both phenomena indicate the advantage of MEC-Quant in enhancing generalization.

V. CONCLUSION

In this paper, we propose MEC-Quant, a simple, novel, yet compelling paradigm for QAT. MEC-Quant is a more principled objective that explicitly optimizes on the structure of the representation, so that the learned representation is less biased and thus generalizes better to unseen in-distribution samples. MEC-Quant pushes the performances of QAT models towards and surpasses that of FP32 counterparts, especially at the

low bit widths. Our approach demonstrates promising results across various neural network models. Extensive experiments show that our method successfully enhances the generalization ability of the quantized model and outperforms the SOTA approaches in recent QAT research.

REFERENCES

- [1] B. Zoph and Q. V. Le, “Neural architecture search with reinforcement learning,” *arXiv preprint arXiv:1611.01578*, 2016.
- [2] S. Han, H. Mao, and W. J. Dally, “Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding,” *arXiv preprint arXiv:1510.00149*, 2015.
- [3] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, “Quantization and training of neural networks for efficient integer-arithmetic-only inference,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2704–2713.
- [4] S. Xu, Y. Li, M. Lin, P. Gao, G. Guo, J. Lü, and B. Zhang, “Q-det: An efficient low-bit quantized detection transformer,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 3842–3851.
- [5] M. Nagel, R. A. Amjad, M. Van Baalen, C. Louizos, and T. Blankevoort, “Up or down? adaptive rounding for post-training quantization,” in *International Conference on Machine Learning*. PMLR, 2020, pp. 7197–7206.
- [6] Y. Li, R. Gong, X. Tan, Y. Yang, P. Hu, Q. Zhang, F. Yu, W. Wang, and S. Gu, “Brecq: Pushing the limit of post-training quantization by block reconstruction,” *arXiv preprint arXiv:2102.05426*, 2021.
- [7] X. Wei, R. Gong, Y. Li, X. Liu, and F. Yu, “Qdrop: Randomly dropping quantization for extremely low-bit post-training quantization,” *arXiv preprint arXiv:2203.05740*, 2022.
- [8] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha, “Learned step size quantization,” *arXiv preprint arXiv:1902.08153*, 2019.
- [9] M. Nagel, M. Fournarakis, Y. Bondarenko, and T. Blankevoort, “Overcoming oscillations in quantization-aware training,” in *International Conference on Machine Learning*. PMLR, 2022, pp. 16318–16330.
- [10] T. M. Cover, *Elements of information theory*. John Wiley & Sons, 1999.
- [11] A. Polino, R. Pascanu, and D. Alistarh, “Model compression via distillation and quantization,” *arXiv preprint arXiv:1802.05668*, 2018.
- [12] Y. Bengio, N. Léonard, and A. Courville, “Estimating or propagating gradients through stochastic neurons for conditional computation,” *arXiv preprint arXiv:1308.3432*, 2013.
- [13] J. Lee, D. Kim, and B. Ham, “Network quantization with element-wise gradient scaling,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 6448–6457.
- [14] J. Kim, K. Yoo, and N. Kwak, “Position-based scaled gradient for model quantization and pruning,” *Advances in neural information processing systems*, vol. 33, pp. 20415–20426, 2020.
- [15] A. Défossez, Y. Adi, and G. Synnaeve, “Differentiable model compression via pseudo quantization noise,” *arXiv preprint arXiv:2104.09987*, 2021.
- [16] S. Xu, Y. Li, T. Ma, M. Lin, H. Dong, B. Zhang, P. Gao, and J. Lu, “Resilient binary neural network,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, 2023, pp. 10620–10628.
- [17] Y.-H. Cao, P. Sun, Y. Huang, J. Wu, and S. Zhou, “Synergistic self-supervised and quantization learning,” in *European Conference on Computer Vision*. Springer, 2022, pp. 587–604.
- [18] P. Wang, X. He, Q. Chen, A. Cheng, Q. Liu, and J. Cheng, “Unsupervised network quantization via fixed-point factorization,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 6, pp. 2706–2720, 2021.
- [19] Y. Wei, X. Pan, H. Qin, W. Ouyang, and J. Yan, “Quantization mimic: Towards very tiny cnn for object detection,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 267–283.
- [20] Y. Li, S. Xu, B. Zhang, X. Cao, P. Gao, and G. Guo, “Q-vit: Accurate and fully quantized low-bit vision transformer,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 34451–34463, 2022.
- [21] Z. Liu, K.-T. Cheng, D. Huang, E. P. Xing, and Z. Shen, “Nonuniform-to-uniform quantization: Towards accurate quantization via generalized straight-through estimation,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 4942–4952.
- [22] J. Beirlant, E. J. Dudewicz, L. Györfi, E. C. Van der Meulen *et al.*, “Nonparametric entropy estimation: An overview,” *International Journal of Mathematical and Statistical Sciences*, vol. 6, no. 1, pp. 17–39, 1997.
- [23] L. Paninski, “Estimation of entropy and mutual information,” *Neural computation*, vol. 15, no. 6, pp. 1191–1253, 2003.
- [24] X. Liu, Z. Wang, Y.-L. Li, and S. Wang, “Self-supervised learning via maximum entropy coding,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 34091–34105, 2022.
- [25] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, “Pytorch: An imperative style, high-performance deep learning library,” *Advances in neural information processing systems*, vol. 32, 2019.
- [26] Y. Li, M. Shen, J. Ma, Y. Ren, M. Zhao, Q. Zhang, R. Gong, F. Yu, and J. Yan, “Mqbenc: Towards reproducible and deployable model quantization benchmark,” *arXiv preprint arXiv:2111.03759*, 2021.
- [27] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [28] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.
- [29] J. Choi, Z. Wang, S. Venkataramani, P. I.-J. Chuang, V. Srinivasan, and K. Gopalakrishnan, “Pact: Parameterized clipping activation for quantized neural networks,” *arXiv preprint arXiv:1805.06085*, 2018.
- [30] Y. Bhalgat, J. Lee, M. Nagel, T. Blankevoort, and N. Kwak, “Lsq+: Improving low-bit quantization through learnable offsets and better initialization,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 696–697.
- [31] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” *arXiv preprint arXiv:1503.02531*, 2015.
- [32] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “Mobilenets: Efficient convolutional neural networks for mobile vision applications,” *arXiv preprint arXiv:1704.04861*, 2017.
- [33] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, “On large-batch training for deep learning: Generalization gap and sharp minima,” in *International Conference on Learning Representations*, 2017.
- [34] Y. Wen, K. Luk, M. Gazeau, G. Zhang, H. Chan, and J. Ba, “An empirical study of stochastic gradient descent with structured covariance noise,” in *International Conference on Artificial Intelligence and Statistics*, 2020.