# Learning to Generate Diverse Data From a Temporal Perspective for Data-Free Quantization

Hui Luo, Shuhai Zhang, Zhuangwei Zhuang, Jiajie Mai, Mingkui Tan, *Member, IEEE*, and Jianlin Zhang

***Abstract*— Model quantization is a prevalent method to compress and accelerate neural networks. Most existing quantization methods usually require access to real data to improve the performance of quantized models, which is often infeasible in some scenarios with privacy and security concerns. Recently, data-free quantization has been widely studied to solve the challenge of not having access to real data by generating synthetic data, among which generator-based data-free quantization is an important type. Previous generator-based methods focus on improving the performance of quantized models by optimizing the spatial distribution of synthetic data, while ignoring the study of changes in synthetic data from a temporal perspective. In this work, we reveal that generator-based data-free quantization methods usually suffer from the issue that synthetic data show homogeneity in the mid-to-late stages of the generation process due to the stagnation of the generator update, which hinders further improvement of the performance of quantized models. To solve the above issue, we propose introducing the discrepancy between the full-precision and quantized models as new supervision information to update the generator. Specifically, we propose a simple yet effective adversarial Gaussian-margin loss, which promotes continuous updating of the generator by adding more supervision information to the generator when the discrepancy between the full-precision and quantized models is small, thereby generating heterogeneous synthetic data. Moreover, to mitigate the homogeneity of the synthetic data further, we augment the synthetic data with linear interpolation. Our proposed method can also promote the performance of other generator-based data-free quantization methods. Extensive experimental results show that our proposed method achieves superior performances for various settings on data-free quantization, especially in ultra-low-bit settings, such as 3-bit.**

***Index Terms*— Model quantization, data-free quantization, generation process, synthetic data, linear interpolation.**

Manuscript received 24 February 2024; revised 8 April 2024; accepted 30 April 2024. Date of publication 9 May 2024; date of current version 30 October 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62072190; in part by the Frontier Research Fund of the Institute of Optics and Electronics, Chinese Academy of Sciences, under Grant C21K005; and in part by the Key-Area Research and Development Program Guangdong Province under Grant 2018B010107001. This article was recommended by Associate Editor J. Cai. *(Corresponding authors: Jianlin Zhang; Mingkui Tan.)*

Hui Luo and Jianlin Zhang are with the National Key Laboratory of Optical Field Manipulation Science and Technology and the Key Laboratory of Optical Engineering, Institute of Optics and Electronics, Chinese Academy of Sciences, Chengdu 610209, China, and also with the School of Electronic, Electrical and Communication Engineering, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: luohui19@mails.ucas.ac.cn; jlin_zh@163.com).

Shuhai Zhang, Zhuangwei Zhuang, and Mingkui Tan are with the School of Software Engineering and the Key Laboratory of Big Data and Intelligent Robot, Ministry of Education, South China University of Technology, Guangzhou 510641, China (e-mail: mszhangshuhai@mail.scut.edu.cn; z.zhuangwei@mail.scut.edu.cn; mingkuitan@scut.edu.cn).

Jiajie Mai is with the Department of Electrical Engineering, City University of Hong Kong, Hong Kong, China (e-mail: jiajiemai0926@gmail.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSVT.2024.3399311.

Digital Object Identifier 10.1109/TCSVT.2024.3399311

## I. Introduction

DEEP neural networks (DNNs) have shown excellent performance in many fields of computer vision, such as image classification [1], [2], object detection [3], [4], [5], semantic segmentation [6], [7], and video processing [8], [9], [10]. However, the rapid growth of parameters and computational complexity of DNNs hinders their deployment on resource-constrained edge devices. To address this challenge, massive model compression and acceleration methods, such as pruning [11], [12], [13], quantization [14], [15], [16], knowledge distillation [17], [18], [19] and low-rank approximation [20], have emerged to improve the efficiency of DNNs.

Model quantization is a prevalent model compression method that uses low-bit integers to represent floating-point weights and activations to compress the model and accelerate inference. As more and more devices support low-precision computations [23], model quantization is more hardware-friendly than other model compression methods. Existing model quantization methods can be divided into post-training quantization (PTQ) [24], [25], [26] and quantization-aware training (QAT) [27], [28], [29]. PTQ directly quantizes pre-trained full-precision models without fine-tuning or retraining and relies on only a small amount of real training data. Although PTQ is easy to implement, it is prone to significant performance degradation, especially in ultra-low-bit quantization. In contrast, QAT uses sufficient real training data to calibrate the model to adapt to quantization errors, whereby the quantized model shows comparable or even better performance than the pre-trained full-precision model. QAT is hard to implement as it is time-consuming and computationally intensive.

Both PTQ and QAT require real training data to calibrate the quantized model to restore performance. However, in modern

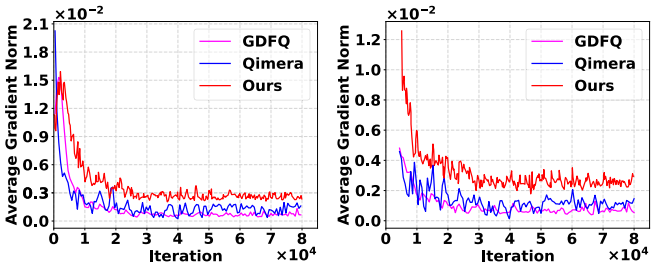



Fig. 1. Comparison of our method with GDFQ [21] and Qimera [22] regarding average gradient norm. When ResNet-20 is quantized to 3-bit on CIFAR-100, the average gradient norms of (a) the generator and (b) the quantized model are reported, respectively. Best viewed in color.

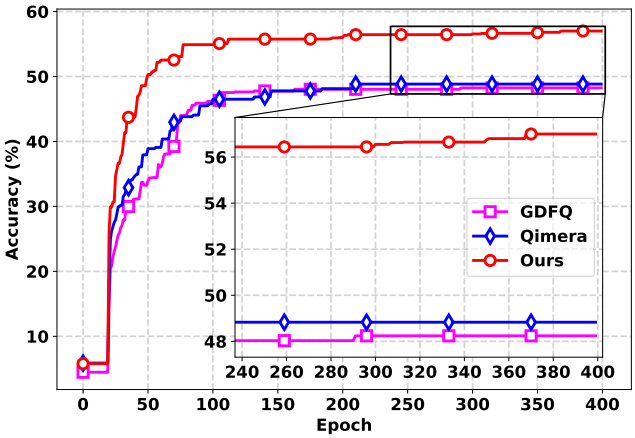



Fig. 2. Comparison of our method with GDFQ and Qimera regarding best test accuracy during training when ResNet-20 is quantized to 3-bit on CIFAR-100. It is easy to observe that in the mid-to-late stages of the training process, GDFQ and Qimera can hardly further improve the best test accuracy, while our method can still improve the best test accuracy. Best viewed in color.

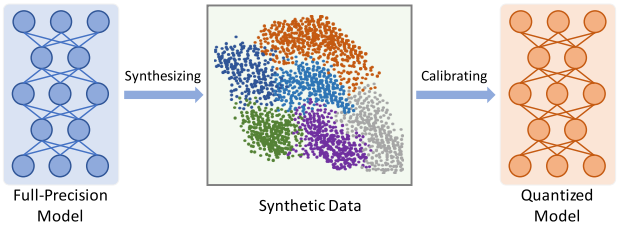
society involving data privacy protection and sensitivity, many scenarios face the inaccessibility of real data. Such practical scenarios have given rise to the emergence of data-free quantization, which quantizes models without access to any real training data.

When only a pre-trained full-precision model is given, in order to achieve data-free quantization, a straightforward solution is to generate synthetic data based on knowledge from full-precision models to replace real data. Methods for generating synthetic data can be broadly categorized into generator-based methods [22], [30], [31] and optimization-based methods [32], [33], [34]. The former is to generate synthetic data by deploying a generator and utilizing the full-precision model as the discriminator to guide the update of the generator. In contrast, the latter is to fit the real data distribution by iteratively optimizing input sampled from a random noise distribution, e.g., Gaussian distribution. Recently, generator-based methods have been attracting much attention due to their excellent performance in capturing the structural and semantic of the data [35]. In this work, we choose the generator-based method to generate synthetic data for data-free quantization.

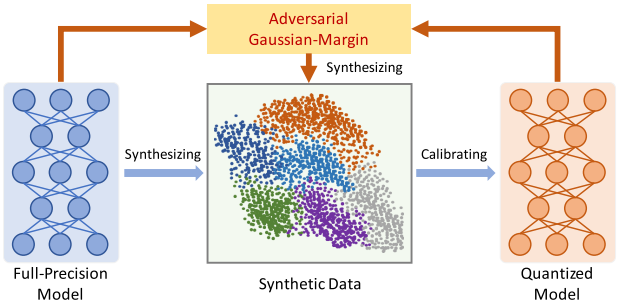
Existing generator-based data-free quantization methods [21], [22], [36] mainly address the gap between synthetic data distribution and real data distribution. For example, Choi et al. [22] proposes to generate synthetic boundary supporting samples better to learn the distribution of real data around decision boundaries. However, these methods only study from the perspective of the spatial distribution of synthetic data and do not consider the changes in synthetic data from a temporal perspective, i.e., the generation process. In this work, to explore the changes in synthetic data during the generation process, we investigate the gradient changes of the generator during the generation process. Specifically, we count the changes in the average gradient norm of the generator during the generation process. From Fig. 1(a), the average gradient norm of GDFQ and Qimera stabilizes in the range close to 0 in the mid-to-late stages of the generation process. Obviously, the average gradient norm of the generator stabilizes around 0, meaning that the update of the generator is stagnant. What prevents the generator from being consistently updated? We argue that it is due to existing generator-based methods only utilizing knowledge from the full-precision model when updating the generator. If a generator only relies on the full-precision model for updating, the distribution of the synthetic data will gradually approach the distribution of the real data represented by the full-precision model. In that case, the supervision information provided by the full-precision model to the generator for updating will gradually become insufficient, thus leading to stagnation in updating the generator. Once the update of the generator stagnates, synthetic data will show homogeneity in the mid-to-late stages of the generation process.

Homogeneous synthetic data may hinder the calibration of the quantized model. Specifically, when the synthetic data becomes homogeneous, as the quantized model is gradually calibrated, the discrepancy between the quantized and full-precision models will gradually decrease, and then the supervision signal provided to the quantized model will decrease. As seen from Fig. 1(b), in the mid-to-late stages of the training process, the average gradient norm used by GDFQ and Qimera to update the quantized model stabilizes around 0. This ultimately limits the performance of the quantized model to further improve in the mid-and-late stages of the training process. As seen in Fig. 2, GDFQ and Qimera hardly further improve the best test accuracy of the quantized model in the mid-to-late stages of the generation process (greater than the 240-th epoch).

In response to the above concerns, in this work, we consider introducing additional supervision information to encourage the generator to be consistently updated in the mid-to-late stages of the generation process. Specifically, as demonstrated in Fig. 3, we introduce the discrepancy between the full-precision and quantized models as additional supervision information to guide the generator update. Furthermore, to take advantage of this additional supervision information, we propose a simple yet effective adversarial Gaussian-margin loss based on the Gaussian kernel. This loss is effective in providing more supervision information for updating the generator to generate heterogeneous synthetic data when the discrepancy between the full-precision and quantized models is small. In the mid-to-late stages of the training process, since generated heterogeneous synthetic data increases the




(a) Common data-free quantization schemes.




(b) Our data-free quantization scheme.

Fig. 3. Illustration of our data-free quantization scheme compared to common data-free quantization schemes. The main difference is that our scheme additionally uses the discrepancy between the full-precision and quantized models for synthesizing the data compared to the common scheme.

discrepancy between the two models, more supervision information is provided to calibrate the quantized model. Thus, the performance of the quantized model can be further improved. Moreover, to mitigate the homogeneity of synthetic data further, we augment the synthetic data with a simple but effective linear interpolation.

We highlight our main contributions as follows:

- For the first time, we revisit generator-based data-free quantization methods from a temporal perspective. Our study reveals that generator-based data-free quantization methods usually suffer from the issue that synthetic data show homogeneity in the mid-to-late stages of the generation process, which harms the performance of the quantized model. To our knowledge, this issue has not been reported in existing studies.
- To address the above issue, we introduce the discrepancy between the full-precision and quantized models as additional supervision information and propose a simple yet effective adversarial Gaussian-margin loss to utilize this additional supervision information to enable the generator to generate heterogeneous synthetic data in the mid-to-late stages of the generation process.
- To give an intuitive explanation of our method, we theoretically analyze the overall optimization for the generator and the quantized model from an adversarial lens. Extensive experiments on CIFAR-10/100, ImageNet, and a variety of popular models demonstrate the superiority of our method over existing data-free quantization methods.

The remaining parts of this paper are organized as follows: We briefly review some relevant existing work in model quantization in Sec. II. Then, in Sec. III, we present details of our proposed method. Next, extensive experimental results and analysis are presented to verify the effectiveness of our method in Sec. IV. We finally conclude in Sec. V.

## II. Related Work

Model quantization is a promising research topic in model compression and acceleration, which aims to store parameters with lower bit-widths for reducing computational and memory costs. Depending on whether the quantization uses data or not, model quantization can be broadly categorized into data-driven quantization and data-free quantization. In this section, we provide a brief review of data-driven quantization and data-free quantization.

### A. Data-Driven Quantization

As an effective method of model compression and acceleration, an important challenge of model quantization is that quantization usually leads to performance degradation, especially in ultra-low-bit settings. To address this challenge, various quantization schemes have been proposed. PTQ is proposed to perform model quantization with limited training data and less computational overhead [26], [37], [38], [39], [40]. Specifically, PTQ first allows the pre-trained full-precision model to perform forward inference on a small amount of training data (a.k.a. calibration data) to obtain the statistical parameters required for quantization and then performs quantization operations based on these statistical parameters to obtain the quantized model. Reference [37] optimizes the clipping range analytically and introduces channel-wise bit allocation and bias-correction to achieve 4-bit PTQ. Reference [24] divides the original quantization range into non-overlapping regions to alleviate the performance degradation of PTQ. Li et al. [26] proposes reconstructing the quantized model in a block level, which for the first time reduces the limit of the bit-width of PTQ to INT2. Reference [39] transforms the linear quantization into the minimum mean square error of the weights and activations to be solved. Zhao et al. [40] proposes outlier channel splitting to reduce channels that contain outliers. Since the differences in the ranges of the outputs of different channels can be very large, PTQ is prone to lead to poor performance of the quantized model.

To obtain a more accurate quantized model, QAT is proposed [28], [29], [41], [42], [43], [44]. QAT first inserts fake quantization layers into the computational graph of the full-precision model to simulate the quantization operation, then fine-tunes the model to adapt to the errors caused by quantization, and finally quantizes the model using the obtained statistical parameters. QAT is more expensive to train than PTQ. Reference [41] proposes learnable step size, which is learned in combination with other network parameters to improve quantization performance. Gong et al. [42] proposes that differentiable soft quantization bridges the gap between the full-precision and quantized models by gradually approximating the standard quantization during training. References [43] and [44] design quantizers to better learn the distribution of weights and activations. Reference [29] introduces a full-precision auxiliary module to update the parameters of the low-precision model, making it easier to back-propagate the gradient on the low-precision model during training. Lin et al. [28] viewed the quantization error from

the perspective of angular bias and proposed a rotated binary neural network to align the angle between the full-precision weight and its binarized version. Methods represented by PTQ and QAT have driven the development of model quantization. However, these model quantization methods mentioned above all require access to part or all of the real training data, which is not applicable to cases without access to real data.

### B. Data-Free Quantization

Recently, data-free quantization [21], [31], [32], [33], [45] has gained widespread attention due to security and privacy concerns, as it can quantize models without access to real data. The early work DFQ [45] equalizes the weight range in the network and corrects the bias caused by quantization to improve the accuracy of the quantized model. To tackle the challenge of inaccessible data, data-free quantization usually generates synthetic data. ZeroQ [32] utilizes the distance between the mean and variance of the synthetic data and the BN statistics of the full-precision model as a supervision signal to optimize Gaussian noise to generate synthetic data. He et al. [46] achieves zero-shot optimization of synthetic data by generative modeling to directly match the distribution of BN statistics. In addition to BN statistics, GDFQ [21] also uses classification boundary knowledge in the pre-trained full-precision model, i.e., category label information, to optimize the generator to generate meaningful synthetic data. Reference [47] proposes three schemes for generating synthetic data to calibrate and fine-tune quantized models without accessing real data. Reference [48] proposes an adversarial knowledge distillation scheme to perform data-free quantization, which minimizes the maximum distance between the outputs of the student model and the teacher model on the adversarial sample. SQuant [49] performs data-free quantization by approximating the three diagonal Hessians. Reference [50] proposes a novel two-level difference modeling to train the generator in an adversarial manner, promoting knowledge transfer from the full-precision model to the quantized model. AdaSG [51] and AdaFQ [31] regard data-free quantization as a zero-sum game, consider the adaptation of synthetic data between the full-precision and quantized models, and propose an adaptive scheme to regulate the adaptation of synthetic data to the quantized model. Here, we want to stress the difference in motivation between our work and these two works. AdaSG and AdaFQ introduce knowledge from the quantized model to address the problem of over-and-under fitting of synthetic data, while our goal is to promote the continuous update of the generator.

Since the quality of synthetic data has a significant impact on the performance of data-free quantization, several works have been devoted to improving the quality of synthetic data. DSG [33] relaxes the alignment loss of the BN statistics to alleviate the homogenization of synthetic data. Considering the distribution of synthetic data around the decision boundaries, Qimera [22] proposes to generate synthetic boundary support samples to reflect the distribution of real data better. IntraQ [34] uses a local object reinforcement and a marginal distance constraint for increasing the intra-class heterogeneity of the synthetic data. Unlike the above methods, which mainly focus on optimizing the synthetic data in terms of its spatial distribution, our proposed method concentrates on mitigating the homogeneity shown by the synthetic data during the training process.

**Algorithm 1** Pipeline of Our Data-Free Quantization Scheme

**Input:** A pre-trained full-precision model $P$; Quantization precision $k$; Warm-up epoch $T_w$; All epochs $T_c$; A generator $G$ with randomly initialized weights.

**Output:** A quantized model $Q$ with $k$-bit precision.

```
Insert fake quantization layers into the pre-trained full-precision model P using Eq. (5) to obtain corresponding quantized model Q.
for t_w = 1, ..., T_w do
    Warm-up the generator G using Eqs. (7 and 8).
end for
for t_c = T_w, ..., T_c do
    Sample random noise z ~ N(0, 1) and corresponding pseudo label y ~ U(0, C − 1).
    Generate synthetic data using Eq. (1).
    Update the generator G by minimizing Eq. (11).
    Obtain augmented synthetic data and corresponding pseudo labels using Eqs. (12 and 13)
    Update the quantized model Q by minimizing Eq. (16).
end for
return A quantized model Q with k-bit precision.
```

## III. Methodology

In this work, we propose an effective data-free quantization scheme to address the problem that generator-based data-free quantization methods are prone to suffer from homogenization of synthetic data in the mid-to-late stages of the generation process. Specifically, as shown in Fig. 4, our scheme contains two components: 1) updating the generator to generate heterogeneous synthetic data and 2) calibrating the quantized model with data augmented by linear interpolation. Data-free quantization is completed by alternately performing the above two components at each iteration. Our general scheme is shown in Algorithm 1.

### A. Preliminaries

Suppose a full-precision model $P$ with full-precision parameters $\theta$. If the real data cannot be accessed, data-free quantization can be employed to obtain a quantized model $Q$ with low-precision parameters $\tilde{\theta}$. To obtain synthetic data, data-free quantization uses a trainable generator $G$ to synthesize data of multiple classes. When assigned a one-hot pseudo label $y$ and a random noise input $z$, the generator can generate synthetic data $\hat{x}$:

$$\hat{x} = G(z \mid y), \quad z \sim \mathcal{N}(0, 1), \tag{1}$$

where $\mathcal{N}(0, 1)$ is a standard Gaussian distribution. To calibrate the quantized model, model quantization reduces the prediction discrepancy between the full-precision $P$ and quantized

models $Q$ on the synthetic data $\hat{x}$ by optimizing the quantized model $Q$:

$$\min_{Q} \mathcal{L}\left[P\left(\hat{x}\right), Q\left(\hat{x}\right)\right], \tag{2}$$

where $\mathcal{L}(\cdot,\cdot)$ is a common loss function, such as Kullback-Leibler (KL) divergence and Mean Squared Error (MSE).

In our work, we focus on the asymmetric uniform quantizer to deploy network quantization, following [21] and [34]. For given full-precision weights $\theta$, the fake quantization layer quantizes them to $k$-bit precision using an asymmetric uniform quantizer as follows:

$$S = \frac{2^k - 1}{u - l}, \tag{3}$$

$$\rho = S \times l + 2^{k-1}, \tag{4}$$

$$\theta_q = \lfloor S \times \theta - \rho \rceil, \tag{5}$$

where $u$ and $l$ are the upper and lower bounds of $\theta$, respectively. $S$ is the scaling factor that converts $\theta$ from the full-precision range to the $k$-bit range $\left[-2^{k-1}, 2^{k-1}-1\right]$. $\rho$ is the zero point of quantization. $\lfloor\cdot\rceil$ is a clipping operation that returns the nearest integer to its input value, $\theta_q$ is the quantized integer. Furthermore, we can compute the dequantized value $\tilde{\theta}$ for forward inference as follows:

$$\tilde{\theta} = \frac{\theta_q + \rho}{S}, \tag{6}$$

it can be seen from Eqs. 5 and 6 that there is a quantization error between $\theta_q$ and $\tilde{\theta}$ due to the clipping operation. QAT is to adapt the model to the quantization error by fine-tuning. Also, since the clipping operation prevents direct backpropagation, we use the Straight Through Estimator (STE) [52] to propagate the gradient bypassing the fake quantization layer. For the quantization of activation, the same operation is performed as for the quantization of weights above.

### B. Updating the Generator

Only a pre-trained full-precision model is provided in settings where real training data is inaccessible. Data-free quantization should fully mine the knowledge from the full-precision model to guide the generator in synthesizing data. Fortunately, in modern deep neural networks, the Batch Normalization (BN) layer serves as a basic component to record the distribution of real training data. The Batch Normalization Statistics (BNS), i.e., the mean and variance, can be used to describe the distribution of the real training data. Therefore, data-free quantization can utilize the BNS stored in the full-precision model to guide the generator update. To this end, we align the mean and variance related to the synthetic data with the mean and variance of the BN layer in the full-precision model:

$$\mathcal{L}_{BNS}\left(\hat{\mathbf{x}}\right) = \sum_{l=1}^{L} \left\| \hat{\boldsymbol{\mu}}_l^P\left(\hat{\mathbf{x}}\right) - \boldsymbol{\mu}_l^P \right\|_2^2 + \left\| \hat{\boldsymbol{\sigma}}_l^P\left(\hat{\mathbf{x}}\right) - \boldsymbol{\sigma}_l^P \right\|_2^2, \tag{7}$$

where $\hat{\mathbf{x}} = \left\{\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n\right\}$ denotes a mini-batch of synthetic data with batch size $n$. $\hat{\boldsymbol{\mu}}_l^P$ and $\hat{\boldsymbol{\sigma}}_l^P$ are the running mean and variance of synthetic data $\hat{\mathbf{x}}$ in the $l$-th BN layer of the pre-trained full-precision model $P$, and $\boldsymbol{\mu}_l^P$ and $\boldsymbol{\sigma}_l^P$ are the mean and variance stored in the $l$-th BN layer of the pre-trained full-precision model $P$, respectively.

In classification tasks, in addition to considering the overall data distribution, it is also necessary to pay attention to the distribution boundaries between categories. Since the decision boundary of the full-precision model can reflect the distribution boundary of the real training data, the decision ability of the full-precision model can be introduced for updating the generator. Specifically, by inputting synthetic data into the full-precision model, the discrepancy between the prediction results of the full-precision model $P$ and the pseudo labels can be used to guide the generator $G$ update, thereby reducing the discrepancy between the distributions of the real training data and the synthetic data. It can be formulated as:

$$\mathcal{L}_{CE}^{G}\left(\hat{x}\right) = -\sum_{i=0}^{C-1} y_i \log \frac{\exp(\hat{l}_i)}{\sum_{j=0}^{C-1}\exp(\hat{l}_j)}, \tag{8}$$

where $y_i$ is the $i$-th value of the assigned one-hot pseudo label $y$ with $C$ values. $\hat{l}_i$ and $\hat{l}_j$ are the $i$-th and $j$-th values of the logits $\hat{l} = P\left(\hat{x}\right) \in \mathbb{R}^C$ output by the full-precision model $P$, respectively.

As mentioned earlier, existing generator-based methods usually suffer from the challenge that the generator update shows stagnation in the mid-to-late stages of the generation process and fails to generate heterogeneous synthetic data to support the entire training process. In order to encourage the generator to be updated in the mid-to-late stages of the generation process to generate heterogeneous synthetic data, thereby providing long-term and rich supervision information for updating the quantized model, we expect to introduce additional supervision information. To this end, we propose introducing the discrepancy between the full-precision and quantized models to guide the generator update. As can be seen from Fig. 3, unlike common data-free quantization schemes, our method additionally introduces knowledge from the quantized model to update the generator, which is expected that the generator still has sufficient supervision signals for updating in the mid-to-late stages of the generation process. In other words, as the synthetic data approaches the real data represented by the full-precision model, the supervision signals that the full-precision model can provide to the generator gradually decrease. At this time, we expect to provide additional supervision information to compensate for the reduced supervision information and keep the generator updated.

How the discrepancy between the full-precision and quantized models is used to guide the generator is critical in determining its effectiveness in promoting the continuous updating of the generator. Since the generator update is stagnant, the discrepancy between the full-precision and quantized models will gradually decrease, thus limiting the improvement of the performance of the quantized model. Therefore, we consider controlling the discrepancy between the full-precision and quantized models to a reasonable range to mitigate the above issue. Specifically, in the mid-to-late stages of the generation process, controlling the discrepancy between the two models to a reasonable range can enable the generator to be updated to generate heterogeneous synthetic data. On the other

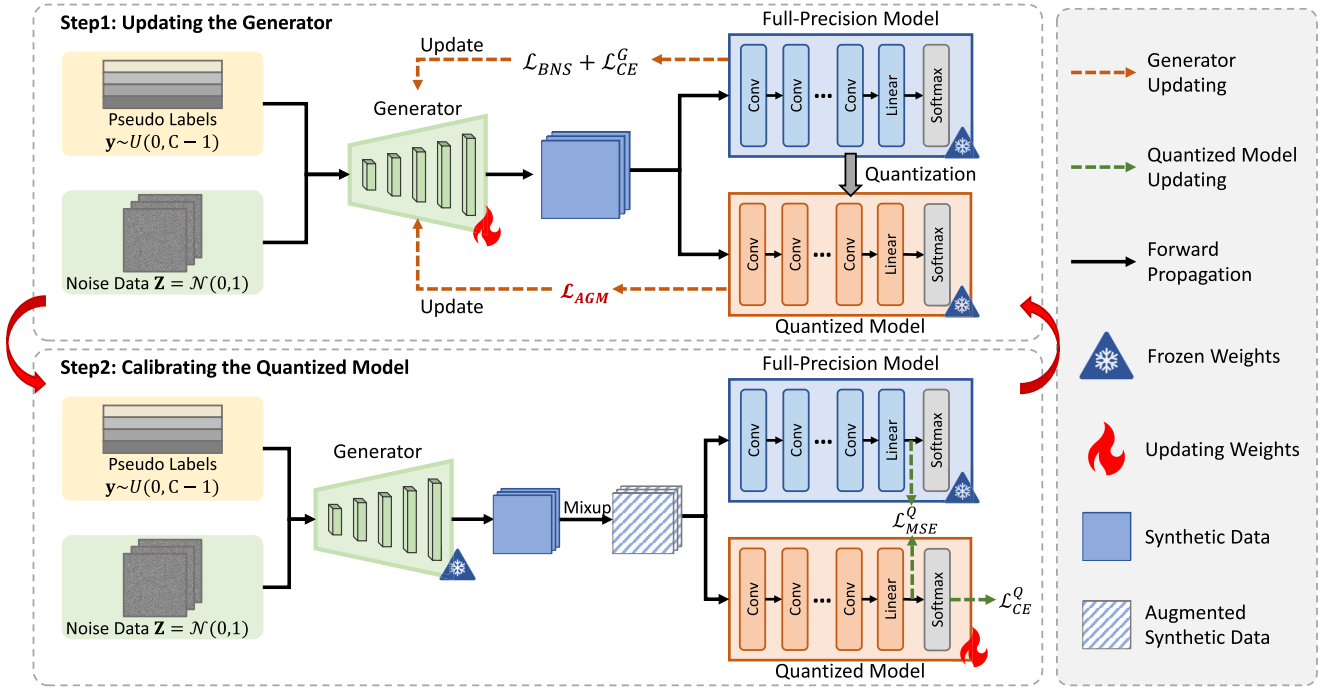



Fig. 4. Illustration of our proposed data-free quantization scheme. Our scheme consists of two steps: 1) updating the generator and 2) calibrating the quantized model. In the first step, Gaussian noise and the corresponding pseudo label are inputted to the generator to generate synthetic data. Then, the synthetic data is fed into the full-precision and quantized models, and the knowledge of these two models is used to update the generator. In the second step, the quantized model learns the knowledge from both the full-precision model and the augmented synthetic data to implement calibration to obtain the quantized model. These two steps are performed alternately during training.

hand, it can give the quantized model sufficient supervision information for updating, thus improving its performance.

*1) Naive Adversarial Margin Loss:* In light of the above analysis, we expect to increase the discrepancy when the discrepancy between the full-precision and quantized models is lower than a threshold, thus keeping the discrepancy around the threshold. To this end, we design the following solution:

$$\min_{G} ReLU\left[\gamma - \left\|P\left(\hat{x}\right) - Q\left(\hat{x}\right)\right\|_2^2\right], \tag{9}$$

where $ReLU$ denotes the ReLU activation function commonly used in neural networks. $\gamma$ is an adjustable threshold. Using $ReLU$ and $\gamma$ ensures that additional supervision signals are transmitted to the generator only when the discrepancy is smaller than a threshold. In the early generation process, the generator is not affected by the loss, thus achieving stable training. On the one hand, this solution helps to control the discrepancy between the two models around a threshold, ensuring that the quantized model always has sufficient supervision signals for updating. On the other hand, the additional supervision signal drives the update of the generator to generate heterogeneous synthetic data.

However, this solution also presents a huge challenge: the setting of threshold $\gamma$ is unfriendly in practical applications. It is usually necessary to know the distribution range of the discrepancy in advance to set an appropriate threshold. If the threshold is set too small, $ReLU\left[\cdot\right]$ will always return 0, which will not provide additional supervision signals to the generator for updating. On the contrary, if the threshold is set too large, providing additional supervision signals to the generator too early will damage the performance of the quantized model. As seen from Fig. 5, this solution is sensitive

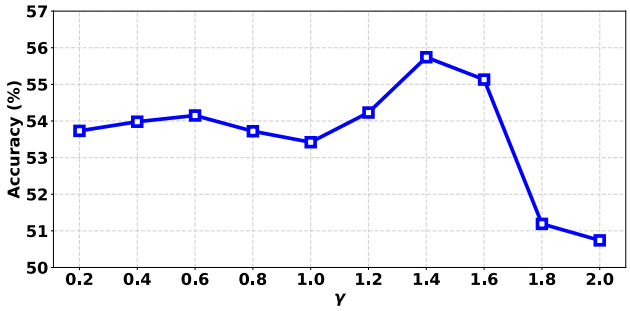



Fig. 5. Accuracy changes of 3-bit quantized ResNet-20 on CIFAR-100 when using naive adversarial margin loss with different thresholds $\gamma$.

to the threshold $\gamma$, and it is difficult to control the threshold to efficiently control the additional supervision signals to tune the quantization performance. For different networks, datasets, and quantization settings, the distribution range of the discrepancy of the two models is different, which further hinders the applicability of the solution. In addition to the difficulty of setting thresholds, this solution may lead to discontinuous loss surfaces, making training unstable [53].

*2) Adversarial Gaussian-Margin Loss:* To further optimize the above solution to control the additional supervision signal flexibly, we propose an adversarial Gaussian-margin loss based on the Gaussian kernel as follows:

$$\mathcal{L}_{AGM}(\hat{x}) = ReLU\left[\exp\left(-\frac{\left\|P\left(\hat{x}\right) - Q\left(\hat{x}\right)\right\|_2^2}{\delta \cdot C}\right) - \tau\right], \tag{10}$$

where $C$ is the number of classes for the classification task. By dividing by $C$, the complexity of the task can

be normalized to ensure that the loss function applies to classification tasks with different numbers of classes. The ReLU function in $\mathcal{L}_{AGM}$ is employed to maintain a non-negative loss value, thereby providing additional supervision signals to the generator. While we choose to utilize the ReLU function to ensure the non-negativity of the loss value, it is important to note that our method is not dependent on any specific activation function. Therefore, the AGM measurement demonstrates broad applicability across various neural network architectures. $\delta$ is a hyper-parameter that controls the smoothing of the prediction discrepancy between the two models. Its role is similar to the kernel width in a Gaussian kernel [54], [55], [56], with smaller kernel widths making the adversarial Gaussian-margin loss more sensitive to the prediction difference, while larger kernel widths make it less sensitive [57]. Since the sensitivity of the adversarial Gaussian-margin loss to the prediction discrepancy can be controlled by adjusting the kernel width, the quantization performance can be flexibly controlled under different datasets, network structures, and bit-width settings. Notably, unlike $\gamma$ in Eq. 9, which is challenging to set, Eq. 10 normalizes the prediction discrepancy, constraining the threshold $\tau \in (0, 1)$, so it is easier to set in practical applications. Going to a more significant level, the form of the Gaussian kernel provides smoothness in measuring prediction discrepancy. Specifically, through the exponential term, the Gaussian kernel produces a gradual response to smaller prediction differences, which helps the loss surface during training to be smoother so that the training after the introduction of adversarial Gaussian-margin loss is still stable.

To update the generator to synthesize high-quality data, we need to train the generator $G$. As in Eq. 1, we start by randomly sampling noise inputs $\mathbf{z} = \{z_1, z_2, \ldots, z_m\}$ from a standard Gaussian distribution $\mathcal{N}(0, 1)$ and pseudo labels $\mathbf{y} = \{y_1, y_2, \ldots, y_m\}$ from a discrete uniform distribution $U(0, C-1)$. These noise inputs and assigned pseudo labels are then fed into the generator $G$ to produce synthetic data $\hat{\mathbf{x}}$. Here, $m$ and $C$ denote the batch size and the number of classes, respectively. Then, we feed the synthetic data $\hat{\mathbf{x}}$ into the pre-trained full-precision model $P$ to compute the loss functions in Eqs. 7 and 8. Also, the synthetic data $\hat{\mathbf{x}}$ is fed into the quantized model $Q$ to compute the loss function in Eq 10. As such, our final loss for updating the generator to synthesize the data can be computed as follows:

$$\mathcal{L}_G\left(\hat{\mathbf{x}}\right) = \mathbb{E}\left[\mathcal{L}_{CE}^{G}\left(\hat{\mathbf{x}}\right)\right] + \beta_1 \mathcal{L}_{BNS}\left(\hat{\mathbf{x}}\right) + \beta_2 \mathbb{E}\left[\mathcal{L}_{AGM}\left(\hat{\mathbf{x}}\right)\right], \tag{11}$$

where $\beta_1$ and $\beta_2$ are hyper-parameters used for trade-offs. $\mathbb{E}[\cdot]$ returns the expected value of the input.

### C. Calibrating the Quantized Model

To mitigate the homogeneity of the synthetic data further and to improve the discriminative power of the quantized model on the synthetic data, we expect to augment the synthetic data with linear interpolation, which gives the quantized model a chance to capture the richer information better. In this work, we use mixup [58] to augment the synthetic data to mitigate the homogeneity of the synthetic data. Furthermore, mixup can also improve the robustness and generalization of the model, which helps mitigate the negative impact of quantization on the performance of the model. Mixup is an effective data augmentation method that mixes two input data in a random linear interpolation manner to construct new training data and the corresponding labels. It can be formulated as:

$$\hat{x}_m = \lambda \hat{x}_i + (1-\lambda)\,\hat{x}_j, \tag{12}$$

$$y_m = \lambda y_i + (1-\lambda)\,y_j, \tag{13}$$

where $\hat{x}_i$ and $\hat{x}_j$ are raw inputs, and $y_i$ and $y_j$ are the corresponding pseudo one-hot labels. $\lambda \in [0, 1]$ is a weighting coefficient. $\hat{x}_m$ and $y_m$ are the augmented synthetic data and pseudo label, respectively. Next, we will use the augmented synthetic data to replace the raw synthetic data for calibrating the quantized model.

In the previous subsection, we encourage the generator to generate heterogeneous synthetic data in the generation process by controlling the discrepancy between the full-precision and quantized models. In this subsection, we will calibrate the model using the augmented synthetic data. In data-free quantization, synthetic data and full-precision models are utilized to guide the calibration of the quantized model, which is usually fine-tuned by a teacher-student framework.

Considering that the BNS in the model generally capture information about the distribution of real training data, we opt to maintain the BNS fixed in both the full-precision and quantized models during the training process. This helps the quantized model retain the distribution information of the real data and stabilize the training process. Besides distribution information, the quantized model should also have benign boundary decision ability like the full-precision model, i.e., the quantized model should be able to classify the synthetic data correctly. In the previous subsection, the decision boundary of the full-precision model was used to align the distribution boundary of the synthetic data to update the generator. Here, we use the distribution boundary of the synthetic data to align the decision boundary of the quantized model. This allows the decision boundaries of the full-precision model and the quantized model to be aligned so that the quantized model has a similar boundary decision ability to the full-precision model. Therefore, we define the following loss function to update the quantized model:

$$\mathcal{L}_{CE}^{Q}\left(\hat{x}_m\right) = -\sum_{i=0}^{C-1} y_{m;i}\,\log\frac{\exp(\tilde{l}_{m;i})}{\sum_{j=0}^{C-1}\exp(\tilde{l}_{m;j})}, \tag{14}$$

where $\hat{x}_m$ is the augmented synthetic data, $y_{m;i}$ is the $i$-th value of the augmented pseudo label $y_m$ with $C$ values. $\tilde{l}_{m;i}$ and $\tilde{l}_{m;j}$ are the $i$-th and $j$-th values of the logits $\tilde{l}_m = Q\left(\hat{x}_m\right) \in \mathbb{R}^C$ output by the quantized model $Q$, respectively.

Although synthetic data plays the role of a bridge between the full-precision and quantized models, helping to transfer knowledge from the full-precision model to the quantized model, it is insufficient to update the quantized model only by aligning the distributions between the synthetic data and the quantized model due to the synthetic data is not equivalent to

the real data. Therefore, to further improve the performance of the quantized model, we use a teacher-student framework to transfer knowledge from the teacher model (full-precision model) to the student model (quantized model). Specifically, for the same synthetic data, the knowledge from the teacher model can be transferred to the student model by reducing the discrepancy between the teacher model and the student model, which can be given by Eq. 2. In this work, inspired by [36], since MSE loss solves the distribution shift problem better than KL loss and has a smaller generalization bound, we use MSE loss as a more strict metric to measure the discrepancy between the full-precision and quantized models on the synthetic data. Thus, the above process can be represented as:

$$\mathcal{L}_{MSE}^{Q}\left(\hat{x}_m\right) = MSE\left[P\left(\hat{x}_m\right), Q\left(\hat{x}_m\right)\right], \tag{15}$$

where $MSE$ denotes Mean Squared Error (MSE). Therefore, the overall loss used for calibrating the quantized model is as follows:

$$\mathcal{L}_Q\left(\hat{\mathbf{x}}_m\right) = \mathbb{E}\left[\mathcal{L}_{CE}^{Q}\left(\hat{\mathbf{x}}_m\right)\right] + \beta_3 \mathbb{E}\left[\mathcal{L}_{MSE}^{Q}\left(\hat{\mathbf{x}}_m\right)\right], \tag{16}$$

where $\hat{\mathbf{x}}_m = \left\{\hat{x}_{1;m}, \hat{x}_{2;m}, \ldots, \hat{x}_{n;m}\right\}$ denotes a mini-batch of augmented synthetic data with batch size $n$, and $\beta_3$ is a hyper-parameter used for trade-offs.

In order to make the training more stable, we start the activity in the form of a warm-up. Specifically, we first stop updating the quantized model $Q$ and first train the generator $G$ alone for some time. After finishing the warm-up process, we update the generator $G$ and the quantized model $Q$ in an alternating training manner, i.e., the generator $G$ and the quantized model $Q$ are optimized alternatively in every iteration.

### D. Theoretical Analysis

To elucidate the optimization process regarding Eqs. 11 and 16, we would like to analyze it from an adversarial mechanism, making it easier to understand the core insights of our method. To achieve this, we provide the following proposition to give an intuitive explanation of the overall optimization, where we do not consider the augmentation operation to the synthetic data for simplicity.

**Proposition 1** *As the discrepancy between the full-precision model $P$ and the quantized model $Q$ decreases during training, especially when $\left\|P\left(\hat{x}\right) - Q\left(\hat{x}\right)\right\|_2^2 < -C\delta\log\tau$ for $\hat{x}=G(z)$, $z \sim \mathcal{N}(0,1)$, $\mathcal{L}_{AGM}$ encourages $G$ to generate synthetic data maximizing the MSE between the output of $P$ and $Q$. Adversarially, $Q$ resists this by minimizing $\mathcal{L}_{MSE}^{Q}(\hat{x})$. This iterative process converges to a Nash equilibrium between $P$ and $Q$.*

*Proof 1: The conclusion is evident. If the discrepancy between the output of $P$ and $Q$ is less than some specific value, $\left\|P\left(\hat{x}\right) - Q\left(\hat{x}\right)\right\|_2^2 < -C\delta\log\tau$, then there exists as least one sample $\left\|P\left(\hat{x}\right) - Q\left(\hat{x}\right)\right\|_2^2 < -C\delta\log\tau$, contributing to $\exp\left(-\frac{\|P(\hat{x})-Q(\hat{x})\|_2^2}{\delta\cdot C}\right) > \tau$. This triggers the effect of the term $\mathcal{L}_{AGM}$. The generator begins to craft synthetic data with a larger discrepancy between the output of $P$ and $Q$ to decrease the loss $\mathcal{L}_G$ by minimizing $\mathcal{L}_{AGM}$. Adversarially, the quantized model $Q$ responds to decrease the alignment discrepancy by minimizing $\mathcal{L}_{MSE}^{Q}(\hat{x})$. Under such circumstances, the synthetic data with either too large or too small MSE fail to benefit $Q$ during the calibration process. Consequently, $P$ and $Q$ gradually reach a Nash equilibrium to strike a better balance until convergence.*

Proposition 1 shows that the adversarial mechanism enforces the MSE loss $\mathcal{L}_{MSE}^{Q}(\hat{x})$ of $Q$ always to be kept within a reasonable range during the training process, ensuring that the quantized model $Q$ consistently updates its parameters based on the currently generated heterogeneous synthetic data. This also implies that the gradient $\nabla_Q\mathcal{L}_{MSE}^{Q}(\hat{x})$ of the discrepancy for $Q$ continuously encourages adjustments to align the model with all heterogeneous synthetic data. This conclusion coincides with the phenomenons in Fig. 1, where the average gradient norm of $P$ and $Q$ are much larger than other baselines. The enduring gradient prompts the generator to consistently create more varied and distinct samples for quantization, mitigating the problem of *synthetic data homogenization*. Simultaneously, this steady sample flow offers richer supervision signals for the quantized model until the final iteration. This encourages the model to obtain superior performance, as evidenced by the continuously increasing best test accuracy in the last 200 iterations in Fig. 2.

## IV. Experiments

### A. Experimental Setup

*1) Datasets and Networks:* We evaluate our method on three datasets, including CIFAR-10, CIFAR-100 [59] and ImageNet [60], which are well-known datasets for evaluating models on image classification tasks. Meanwhile, they are often used in data-free quantization work. CIFAR-10 and CIFAR-100 contain 10 and 100 classes of natural images of $32 \times 32$ pixels, respectively. They both contain 50k images for training and 10k images for testing. ImageNet is a commonly used large-scale image classification dataset. It has around 1.2 million natural images of $224 \times 224$ pixels for training and 50,000 images of $224 \times 224$ pixels for testing. They are all categorized into 1,000 classes. In this work, to keep the data-free setting, all experiments will only use the test set of the above dataset to evaluate the performance of the quantized model.

We choose to quantize ResNet-20 [2] on CIFAR-10/100, ResNet-18, ResNet-50, ShuffleNet [61] and InceptionV3 [62] on ImageNet. All model implementations and weights of pre-trained full-precision models were taken from the pytorch library.[1]

*2) Baseline:* To demonstrate the superiority of our proposed method, we compare it with some state-of-the-art data-free quantization methods, such as DFQ [45], GDFQ [21], ZeroQ [32], ZAQ [50], DSG [33], ARC [30], Qimera [22], IntraQ [34], AIT [63], SQuant [49], KMDFQ [36], AdaSG [51], HAST [64], DSG-QAT [65] and AdaFQ [31]. Besides, we quantize the model with real training data, calibrate the quantized model by Eq. 14, and report its

[1] https://pypi.org/project/pytorchcv/

accuracy, denoted by **Real Data**. We report the Top-1 accuracy of all methods.

*3) Implementation Details:* For CIFAR-10/100, to synthesize the data, we built the generator using the architecture of ACGAN [66] with 100-dimensional noise. The generator is optimized by an Adam [67] optimizer with a momentum of 0.9, the initial learning rate of 1e-3 and a weight decay of 1e-4. The learning rate is decayed by 0.1 for every 100 epochs. Notably, the optimization of the generator during the warm-up does not use Eq. 10, while the optimization after the warm-up uses Eq. 10. The warm-up epoch is set to 20. For ImageNet, we follow SN-GAN [68] by replacing the standard batch normalization layer of the generator with the categorical conditional batch normalization layer. The training settings of the generator on ImageNet are the same as those on CIFAR-10/100. To calibrate the quantized model, The quantized model is calibrated via Eq. 16 using SGD with Nesterov [69] with a momentum of 0.9 and a weight decay of 1e-4. For CIFAR-10/100 and ImageNet, the initial learning rate is set to 1e-5 and decayed by 0.1 for every 100 epochs. There are five important hyper-parameters in this work, including $\beta_1$ and $\beta_2$ in Eq. 11, $\beta_3$ in Eq. 16, $\delta$ and $\tau$ in Eq. 10. They are respectively set to 0.1, 0.04, 3, 8 and 0.8 on CIFAR-10; 0.1, 0.04, 5, 8 and 0.8 on CIFAR-100; 0.1, 5, 5, 8 and 0.8 on ImageNet. Notably, for $\tau$, we select the optimal threshold $\tau$ based on the dataset. Besides, the generator and the quantized model were alternately trained for 400 epochs with 200 iterations per epoch. We implemented all experiments using PyTorch [70] and ran the experiments on one NVIDIA GeForce RTX 3090 GPU.

### B. Comparison Results on CIFAR-10/100

We quantize the weights and activations of ResNet-20 on CIFAR-10/100 to investigate the effectiveness of the proposed method. From Table I, compared to other state-of-the-art methods, the quantized models obtained by our method show a clear superiority on both CIFAR-10 and CIFAR-100, especially on ultra-low-bit, such as W3A3. Specifically, compared to the advanced optimization-based method IntraQ, our method improves the top-1 accuracy of the 3-bit quantized model by 10.11% (77.07% vs 87.18%) on CIFAR-10 and 8.75% (48.25% vs 57.00%) on CIFAR-100. Meanwhile, compared to the advanced generator-based method AIT, our method achieves significant advantages on both 3-bit and 4-bit quantized models. By introducing the knowledge from the quantized model to guide the generator in generating adaptive samples, AdaSG achieves 84.14% and 52.76% 3-bit accuracy on CIFAR-10 and CIFAR-100, respectively. In comparison, our proposed method performs better at 3-bit, 4-bit, and 5-bit settings. In contrast with AdaFQ, even though our method is slightly inferior in the 5-bit setting, it shows significant advantages in the 3-bit and 4-bit settings, which shows that our proposed method is more effective in utilizing the knowledge from the quantized model. In particular, in the 3-bit case of CIFAR-100, the quantized model obtained by our method is better than that obtained using real training data, which indicates that the heterogeneous synthetic data obtained by our method can correct the quantized model more effectively.

TABLE I

COMPARISON OF QUANTIZATION RESULTS OF RESNET-20 ON CIFAR-10/100. "W$k$A$k$" INDICATES THE WEIGHTS AND ACTIVATIONS ARE QUANTIZED TO $k$-BIT. THE BEST AND SECOND-BEST RESULTS ARE HIGHLIGHTED IN **BOLD** AND UNDERLINE, RESPECTIVELY

| Dataset (FP32 Acc.) | Method | Generator | Top-1 Acc.(%) | | |
|---|---|---|---|---|---|
| | | | W5A5 | W4A4 | W3A3 |
| CIFAR-10 (93.89) | Real Data | - | 93.82 | 92.93 | 87.76 |
| | ZeroQ [32] (CVPR'20) | ✗ | 90.08 | 84.68 | 29.32 |
| | GDFQ [21] (ECCV'20) | ✓ | 93.38 | 90.11 | 75.11 |
| | ZAQ [50] (CVPR'21) | ✓ | 93.36 | 92.13 | - |
| | DSG [33] (CVPR'21) | ✗ | - | 88.74 | 32.90 |
| | ARC [30] (IJCAI'21) | ✓ | 92.88 | 88.55 | - |
| | Qimera [22] (NeurIPS'21) | ✓ | 93.46 | 91.26 | 74.43 |
| | IntraQ [34] (CVPR'22) | ✗ | - | 91.49 | 77.07 |
| | AIT [63] (CVPR'22) | ✓ | 92.98 | 91.23 | 80.49 |
| | SQuant [49] (ICLR'22) | - | - | 92.24 | 79.19 |
| | KMDFQ [36] (TCSVT'23) | ✓ | 93.67 | 92.24 | - |
| | AdaSG [51] (AAAI'23) | ✓ | 93.76 | 92.10 | 84.14 |
| | AdaFQ [31] (CVPR'23) | ✓ | **93.81** | 92.31 | 84.89 |
| | **Ours** | ✓ | 93.79 | **92.87** | **87.18** |
| CIFAR-100 (70.33) | Real Data | - | 69.89 | 68.52 | 56.75 |
| | ZeroQ [32] (CVPR'20) | ✗ | 64.36 | 58.42 | 15.38 |
| | GDFQ [21] (ECCV'20) | ✓ | 67.52 | 63.75 | 47.61 |
| | ZAQ [50] (CVPR'21) | ✓ | 68.70 | 60.42 | - |
| | DSG [33] (CVPR'21) | ✗ | - | 62.36 | 25.48 |
| | ARC [30] (IJCAI'21) | ✓ | 68.40 | 62.76 | 40.15 |
| | Qimera [22] (NeurIPS'21) | ✓ | 69.02 | 65.10 | 46.13 |
| | IntraQ [34] (CVPR'22) | ✗ | - | 64.98 | 48.25 |
| | AIT [63] (CVPR'22) | ✓ | 68.40 | 61.05 | 41.34 |
| | SQuant [49] (ICLR'22) | - | - | 63.96 | 40.36 |
| | KMDFQ [36] (TCSVT'23) | ✓ | 69.68 | 67.15 | - |
| | AdaSG [51] (AAAI'23) | ✓ | 69.42 | 66.42 | 52.76 |
| | HAST [64] (CVPR'23) | ✗ | - | 66.68 | 55.67 |
| | AdaFQ [31] (CVPR'23) | ✓ | **69.93** | 66.81 | 52.74 |
| | **Ours** | ✓ | 69.91 | **67.93** | **57.00** |

### C. Comparison Results on ImageNet

In this subsection, we further conduct experiments on the large-scale dataset ImageNet to demonstrate the superiority of our proposed method. The quantized models include ResNet-18, ResNet-50, ShuffleNet and InceptionV3. Similar to CIFAR-10/100, we quantize the weights and activations of these models.

*1) ResNet-18/50:* Table II shows the experimental results of ResNet-18 and ResNet-50. For Resnet-18, in the case of 5-bit, our method is slightly worse than the generator-based method AIT (70.28% vs 70.16%). When it comes to 3-bit and 4-bit, our method significantly outperforms AIT, especially at 3-bit (36.70% vs 41.60%). Compared to AdaSG and AdaFQ, which utilize the knowledge from the quantized model to update the generator, our method maintains significant superiority at 3-bit, while the performance at other bits remains very close, with a gap of less than 0.2%. For ResNet-50, compared to other state-of-the-art methods, our method achieves the best results at each bit setting. Notably, SQuant is a data-free quantization framework. However, in comparison to SQuant, although our method requires more effort, it exhibits clear advantages in the

TABLE II

COMPARISON OF QUANTIZATION RESULTS OF RESNET-18/RESNET-50 ON IMAGENET. "W$k$A$k$" INDICATES THE WEIGHTS AND ACTIVATIONS ARE QUANTIZED TO $k$-BIT. THE BEST AND SECOND-BEST RESULTS ARE HIGHLIGHTED IN **BOLD** AND UNDERLINE, RESPECTIVELY

| Model (FP32 Acc.) | Method | Generator | Top-1 Acc.(%) | | |
|---|---|---|---|---|---|
| | | | W5A5 | W4A4 | W3A3 |
| ResNet-18 (71.47) | Real Data | - | 70.21 | 64.88 | 51.78 |
| | ZeroQ [32] (CVPR'20) | ✗ | 62.51 | 24.77 | - |
| | GDFQ [21] (ECCV'20) | ✓ | 68.49 | 60.67 | 20.23 |
| | ZAQ [50] (CVPR'21) | ✓ | 64.54 | 52.64 | - |
| | ARC [30] (IJCAI'21) | ✓ | 68.88 | 61.32 | 23.37 |
| | Qimera [22] (NeurIPS'21) | ✓ | 69.29 | 63.84 | 1.17 |
| | IntraQ [34] (CVPR'22) | ✗ | 69.94 | 66.47 | - |
| | AIT [63] (CVPR'22) | ✓ | 70.28 | 65.73 | 36.70 |
| | SQuant [49](ICLR'22) | - | 69.52 | 66.14 | 32.21 |
| | KMDFQ [36] (TCSVT'23) | ✓ | 69.93 | 64.39 | - |
| | AdaSG [51] (AAAI'23) | ✓ | **70.29** | 66.50 | 37.04 |
| | AdaFQ [31] (CVPR'23) | ✓ | **70.29** | 66.53 | 38.10 |
| | **Ours** | ✓ | 70.16 | **66.66** | **41.60** |
| ResNet-50 (77.73) | Real Data | - | 76.35 | 72.37 | 31.98 |
| | GDFQ [21] (ECCV'20) | ✓ | 71.63 | 54.16 | - |
| | ZAQ [50] (CVPR'21) | ✓ | 73.38 | 53.02 | - |
| | ARC [30] (IJCAI'21) | ✓ | 74.13 | 64.37 | 1.63 |
| | Qimera [22] (NeurIPS'21) | ✓ | 75.32 | 66.25 | - |
| | AIT [63] (CVPR'22) | ✓ | 76.00 | 68.27 | - |
| | SQuant [49](ICLR'22) | - | 75.79 | 70.80 | 14.67 |
| | AdaSG [51] (AAAI'23) | ✓ | 76.03 | 68.58 | 16.98 |
| | AdaFQ [31] (CVPR'23) | ✓ | 76.03 | 68.38 | 17.63 |
| | **Ours** | ✓ | **76.13** | **71.32** | **20.27** |

TABLE III

COMPARISON OF QUANTIZATION RESULTS OF SHUFFLENET AND INCEPTIONV3 ON IMAGENET. "W$k$A$k$" INDICATES THE WEIGHTS AND ACTIVATIONS ARE QUANTIZED TO $k$-BIT. THE BEST AND SECOND-BEST RESULTS ARE HIGHLIGHTED IN **BOLD** AND UNDERLINE, RESPECTIVELY

| Model (FP32 Acc.) | Method | Generator | Top-1 Acc.(%) | |
|---|---|---|---|---|
| | | | W5A5 | W4A4 |
| ShuffleNet (65.16) | Real Data | - | 57.65 | 33.01 |
| | DFQ [45] (ICCV'19) | ✗ | 1.23 | 0.92 |
| | ZeroQ [32] (CVPR'20) | ✗ | 7.91 | 0.92 |
| | GDFQ [21] (ECCV'20) | ✓ | 45.10 | 22.89 |
| | DSG [33] (CVPR'21) | ✗ | 54.21 | 24.78 |
| | KMDFQ [36] (TCSVT'23) | ✓ | **56.73** | 28.26 |
| | **Ours** | ✓ | 55.85 | **30.83** |
| InceptionV3 (77.63) | Real Data | - | 76.45 | 71.10 |
| | DFQ [45] (ICCV'19) | ✗ | 59.44 | 0.94 |
| | ZeroQ [32] (CVPR'20) | ✗ | 68.17 | 18.55 |
| | GDFQ [21] (ECCV'20) | ✓ | 75.08 | 68.40 |
| | DSG [33] (CVPR'21) | ✗ | 74.45 | 67.36 |
| | KMDFQ [36] (TCSVT'23) | ✓ | 75.54 | 71.22 |
| | **Ours** | ✓ | **76.20** | **72.39** |

performance of quantized models. Furthermore, SQuant relies on approximating the model's Hessian matrix, which may not fully capture the complexity of the model and the diversity of the data distribution. This limitation is particularly evident in complex models or tasks, such as LLM and stable diffusion, where the Hessian approximation method used by SQuant may struggle to adapt to the parameter space of such complex models. This limitation could lead to limited application scenarios compared to our method. These experimental results show that our proposed method is still effective on large-scale datasets.

*2) ShuffleNet and InceptionV3:* Since ShuffleNet and InceptionV3 are susceptible to suffering significant performance degradation when quantized to ultra-low-bit, we quantize them to 4-bit and 5-bit following the settings of most previous methods [21], [33], [36]. In Table III, our method still outperforms most baselines when quantizing lightweight models. For example, our method achieves better performance when quantizing ShuffleNet to 4-bit compared to DSG, even though the performance in the 5-bit setting is slightly worse than DSG. When it comes to InceptionV3, while other methods achieve lower performance, our method obtains 72.39% accuracy on 4-bit and 76.20% accuracy on 5-bit. Experimental results on ShuffleNet and InceptionV3 demonstrate the effectiveness of our proposed method on lightweight models.

*3) Comparison with DSG-QAT:* DSG-QAT [65] is a state-of-the-art data-free quantization method that generates diverse synthetic data at both the distribution level and the sample level to improve the performance of the quantized model. In Tables IV and V, we present the comparison results of our method with DSG-QAT. We directly report the results published in the original paper in Tables IV and V. In particular, for InceptionV3 and ShuffleNet, because our method and DSG-QAT use different versions of baseline models, to ensure fair comparison as much as possible, we adopt the Acc. $\downarrow$ (%) as a criterion, which measures the drop in accuracy of quantized models compared to full-precision models. From Tables IV and V, when using higher bit-widths, our method, though slightly inferior to DSG-QAT, still remains comparable. When using lower bit-widths, our method generally outperforms DSG-QAT. For example, with a 4-bit ResNet-50, our method achieves a 3.02% higher accuracy compared to DSG-QAT (71.32% vs. 68.30%). We speculate that this phenomenon occurs because our method relies on the prediction discrepancy between full-precision and quantized models. With smaller bit-widths, the prediction discrepancy between full-precision and quantized models becomes more noticeable. This can provide rich supervision signals for updating the generator in the mid-to-late stages of the generation process, thus improving the performance of the quantized model. Conversely, with larger bit-widths, the prediction discrepancy between full-precision and quantized models decreases, making it difficult to provide rich supervision signals for updating the generator in the mid-to-late stages of the generation process.

### D. Combination With Other Methods

To explore the potential of our proposed method, we integrate it with some existing state-of-the-art generator-based

TABLE IV

COMPARISON OF THE RESULTS OF OUR METHOD WITH DSG-QAT WHEN QUANTIZING RESNET-18/RESNET-50 ON IMAGENET. "W$k$A$k$" INDICATES THE WEIGHTS AND ACTIVATIONS ARE QUANTIZED TO $k$-BIT. THE BEST RESULTS ARE HIGHLIGHTED IN **BOLD**

| Model (FP32 Acc.) | Method | Generator | Top-1 Acc.(%) | | |
|---|---|---|---|---|---|
| | | | W8A8 | W6A6 | W4A4 |
| ResNet-18 (71.47) | DSG-QAT [65] (TPAMI'23) | ✔ | **71.46** | 71.18 | **66.67** |
| | **Ours** | ✔ | 71.37 | **71.22** | 66.66 |
| ResNet-50 (77.73) | DSG-QAT [65] (TPAMI'23) | ✔ | **77.83** | 77.22 | 68.30 |
| | **Ours** | ✔ | 77.71 | **77.48** | **71.32** |

TABLE V

COMPARISON OF THE RESULTS OF OUR METHOD WITH DSG-QAT WHEN QUANTIZING INCEPTIONV3/SHUFFLENET ON IMAGENET. "W$k$A$k$" INDICATES THE WEIGHTS AND ACTIVATIONS ARE QUANTIZED TO $k$-BIT. THE BEST RESULTS ARE HIGHLIGHTED IN **BOLD**

| Model | Bit | Method | FP32 Acc. (%) | Quantized Acc. (%) | Acc. ↓ (%) |
|---|---|---|---|---|---|
| InceptionV3 | W4A4 | DSG-QAT [65] (TPAMI'23) | 78.80 | 74.02 | **4.78** |
| | | Ours | 77.63 | 72.39 | 5.24 |
| | W6A6 | DSG-QAT [65] (TPAMI'23) | 78.80 | 78.59 | 0.21 |
| | | Ours | 77.63 | 77.54 | **0.09** |
| | W8A8 | DSG-QAT [65] (TPAMI'23) | 78.80 | 78.85 | -0.05 |
| | | Ours | 77.63 | 77.75 | **-0.12** |
| ShuffleNet | W6A6 | DSG-QAT [65] (TPAMI'23) | 65.07 | 61.94 | 3.13 |
| | | Ours | 65.16 | 62.18 | **2.98** |
| | W8A8 | DSG-QAT [65] (TPAMI'23) | 65.07 | 64.97 | **0.10** |
| | | Ours | 65.16 | 64.63 | 0.53 |

quantization techniques to enhance performance. Specifically, we incorporate our method with GDFQ [21] and FDDA [71], respectively. Notably, FDDA is a post-training quantization approach that utilizes a small calibration dataset. As demonstrated in Table VI, we present comparison results for quantizing ResNet-18 to 3-bit and 4-bit on ImageNet, respectively. The experimental results highlight that our method effectively enhances performance when combined with other generator-based quantization approaches.

In Eq. 15, we use the MSE loss to replace the KL divergence loss to align the predictions between the full-precision and quantized models. To verify whether our proposed method continues to enhance the quantized model's performance when employing the KL divergence loss to align predictions between the full-precision and quantized models, we conducted additional experiments. We replaced the MSE loss in Eq. 15 with the KL divergence loss while keeping other parameters and settings unchanged. Table VII presents the results of quantizing ResNet-20 to 3-bit and 4-bit on CIFAR-100. In the table, "KL" and "MSE" indicate the utilization of KL divergence loss and MSE loss, respectively, in Eq. 15 for aligning model predictions when AGM and Mixup are not utilized. "KL+AGM+Mixup" and "MSE+AGM+Mixup" indicate the combination of AGM and Mixup with the corresponding loss functions. As depicted in Table VII, MSE loss proves more effective for data-free quantization than KL divergence loss, consistently improving the quantized model's performance, in line with prior findings [36]. Additionally, AGM+Mixup continues to enhance the quantized model's performance even when employing the KL divergence loss to align predictions between the two models.

TABLE VI

COMPARISON OF THE RESULTS OF OUR METHOD COMBINED WITH GDFQ AND FDDA, RESPECTIVELY, WHEN QUANTIZING RESNET-18 ON IMAGENET. "W$k$A$k$" INDICATES THE WEIGHTS AND ACTIVATIONS ARE QUANTIZED TO $k$-BIT. THE BEST RESULTS ARE HIGHLIGHTED IN **BOLD**

| Bit | Method | Top-1 Acc.(%) |
|---|---|---|
| W3A3 | GDFQ [36] | 20.23 |
| | **GDFQ [36]+Ours** | **21.97** |
| | FDDA [71] | 38.72 |
| | **FDDA [71]+Ours** | **40.15** |
| W4A4 | GDFQ [36] | 60.67 |
| | **GDFQ [36]+Ours** | **61.42** |
| | FDDA [71] | 68.47 |
| | **FDDA [71]+Ours** | **68.91** |

TABLE VII

COMPARISON OF RESULTS FROM DIFFERENT METHODS WHEN QUANTIZING RESNET-20 ON CIFAR-100. "W$k$A$k$" INDICATES THE WEIGHTS AND ACTIVATIONS ARE QUANTIZED TO $k$-BIT. THE BEST RESULTS ARE HIGHLIGHTED IN **BOLD**

| Bit | Method | Top-1 Acc.(%) |
|---|---|---|
| W3A3 | KL | 49.84 |
| | KL+AGM+Mixup | 52.31 |
| | MSE | 53.57 |
| | **Ours (MSE+AGM+Mixup)** | **57.00** |
| W4A4 | KL | 64.57 |
| | KL+AGM+Mixup | 66.29 |
| | MSE | 66.72 |
| | **Ours (MSE+AGM+Mixup)** | **67.93** |

TABLE VIII

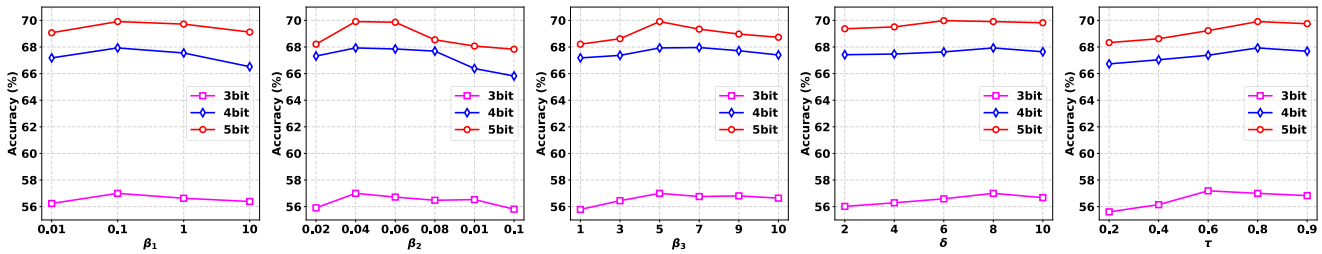
ABLATIONS ON DIFFERENT COMPONENTS OF OUR METHOD. WE PERFORM ABLATION EXPERIMENTS ON RESNET-20 AND CIFAR-100. "W$k$A$k$" INDICATES THE WEIGHTS AND ACTIVATIONS ARE QUANTIZED TO $k$-BIT. THE BEST AND SECOND-BEST RESULTS ARE HIGHLIGHTED IN **BOLD** AND UNDERLINE, RESPECTIVELY

| | Baseline | AGM | Mixup | Top-1 Acc.(%) | | |
|---|---|---|---|---|---|---|
| | | | | W5A5 | W4A4 | W3A3 |
| 1 | ✔ | | | 68.93 | 66.72 | 53.57 |
| 2 | ✔ | ✔ | | <u>69.54</u> | <u>67.57</u> | <u>56.32</u> |
| 3 | ✔ | | ✔ | 69.14 | 67.08 | 54.48 |
| 4 | ✔ | ✔ | ✔ | **69.91** | **67.93** | **57.00** |

### *E. Ablation Study*

In this subsection, we perform extensive ablation experiments to demonstrate the effectiveness of each part of our proposed method.

*1) Effectiveness of the Components of the Proposed Method:* In this work, the components of the proposed method include updating the generator with adversarial Gaussian-margin loss (AGM) and augmenting the synthetic data with mixup. Here, we perform ablations to demonstrate their effectiveness. The experimental results are shown in Table VIII. It should be noted that when neither component is used, it is considered baseline. Specifically, $\beta_2$ in Eq. 11 is set to 0, mixup is not used to augment the synthetic data, and other settings remain unchanged. From the results, when the two components are individually used, the accuracy is improved compared with the baseline under each bit setting. In contrast, AGM brings more significant improvements than mixup under each bit setting. Lastly, when the two components are both




(a) (b) (c) (d) (e)

Fig. 6. Effect of the hyper-parameters on the accuracy of ResNet-20 with different bits on CIFAR-100. From (a) to (e) are the effects of hyper-parameters $\beta_1$, $\beta_2$, $\beta_3$, $\delta$, and $\tau$ on accuracy, respectively. Best viewed in color.

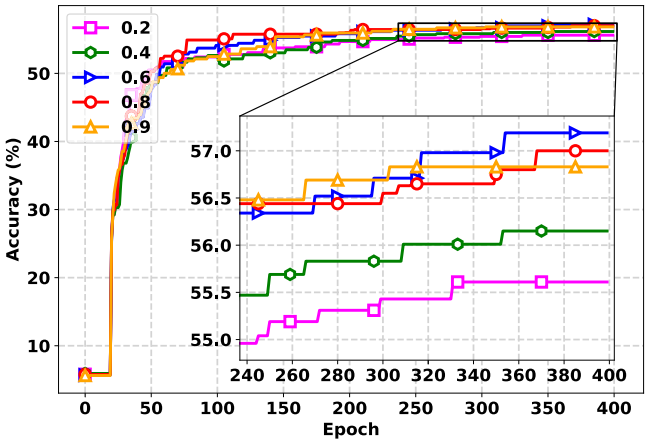
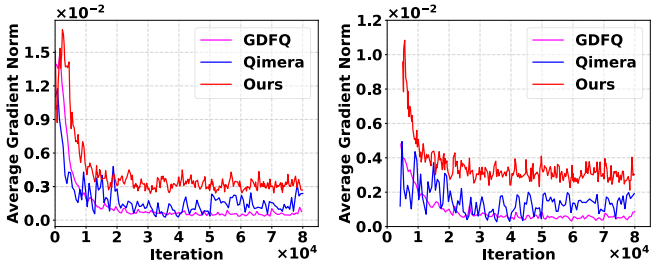



Fig. 7. Comparison of the change in the best test accuracy of the quantized model when quantizing ResNet-20 to 3-bit on CIFAR-100 using different thresholds $\tau$. Best viewed in color.




(a) (b)

Fig. 8. Comparison of our method with GDFQ [21] and Qimera [22] regarding average gradient norm. When ResNet-20 is quantized to 4-bit on CIFAR-100, the average gradient norms of (a) the generator and (b) the quantized model are reported, respectively. Best viewed in color.

applied, the best performance is achieved under each bit setting.

*2) Effects of Different Hyper-parameters:* In this work, the hyper-parameters include $\beta_1$ and $\beta_2$ in Eq. 11, $\beta_3$ in Eq. 16, $\delta$ and $\tau$ in Eq. 10. As shown in Fig. 6, we show the effect of using different hyper-parameters and bits to quantize ResNet-20 on CIFAR-100. It is easy to observe that our method is not very sensitive to the setting of hyper-parameters. Overall, $\beta_1 = 0.1$, $\beta_2 = 0.04$, $\beta_3 = 5$, $\delta = 8$ and $\tau = 0.8$ are optimal settings for CIFAR-100. For CIFAR-10 and ImageNet, we perform similar experiments to search for optimal values of these hyper-parameters.

The threshold $\tau$ serves as a crucial hyper-parameter. To delve deeper into its impact on the quantized model's performance, we investigate how the quantized model's performance evolves with epoch across different thresholds $\tau$. Specifically, we quantize ResNet-20 to 3-bit on CIFAR-100 using varying thresholds $\tau$. In Fig. 7, we compare the changes in the quantized model's best test accuracy over epochs for different thresholds $\tau$. As depicted, the quantized model's performance tends to be subpar with smaller thresholds (e.g., 0.2, 0.4), improving notably with larger thresholds (e.g., 0.6, 0.8, 0.9). However, excessively large thresholds (e.g., 0.9) hinder further improvement in best test accuracy in later stages. We attribute this to our proposed AGM loss potentially exerting early influence with smaller thresholds, which could negatively impact the generator's early updates. Conversely, overly large thresholds pose challenges in providing additional supervision signals to the generator.

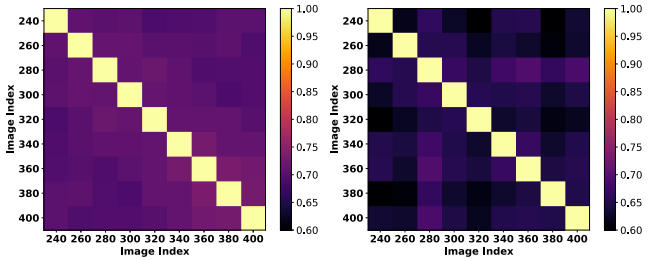
*3) Comparison of Supervision Signals:* In this work, we argue that existing generator-based data-free quantization suffers from the problem of homogeneous data during training caused by insufficient supervision signals. To demonstrate that our proposed method mitigates the problem of insufficient supervision signals, we compare the proposed method with some advanced data-free quantization methods [21], [22] regarding supervision information during the training process. As can be seen from Fig. 1(a), compared with GDFQ and Qimera, our proposed method can provide more supervision signals to the generator. Especially in the mid-to-late stages of the generation process, when the average gradient norm of GDFQ and Qimera is close to 0, the proposed method is still able to provide more supervision signals to update the generator. When the generator has more supervision signals to synthesize the heterogeneous data, the quantized model can also receive more continuous and richer supervision signals for updating during the training process (see Fig. 1(b)). Ultimately, the performance of the quantized model can be further improved. As shown in Fig. 2, our proposed method can make the best accuracy of the quantized model still improve in the mid-to-late stages of the training process. The above experimental results demonstrate that our proposed adversarial Gaussian-margin loss effectively mitigates the problem of insufficient supervision signals suffered by generator-based data-free quantization in the mid-to-late stages of the training process, thus further improving the performance of the quantized model. These results again verify the correctness of our motivation to add more supervision signals.

*4) Comparison of the Two Losses:* We perform experiments to explore the effect of the naive adversarial margin loss with different thresholds $\gamma$ on the quantized model, and the experimental results are shown in Fig. 5. Obviously, we can

(a) Synthetic data obtained without our method.

(b) Synthetic data obtained with our method.

Fig. 9. Visual comparison of synthetic data obtained (a) without and (b) with our method when quantizing ResNet-20 to 3-bit on CIFAR-100. Best viewed in color.




(a) (b)

Fig. 10. Comparison of SSIM heatmaps of synthetic data obtained (a) without or (b) with our method when quantizing ResNet-20 to 3-bit on CIFAR-100. The lower the SSIM score, the more heterogeneous it is. Best viewed in color.

observe from Fig. 5 that the performance of the quantized ResNet-20 is very sensitive to the threshold $\gamma$. For example, when the threshold is less than 1.2, there is no significant improvement in the performance of the quantized model, and there may even be a slight decrease. When the threshold lies between 1.2 and 1.6, there is a significant improvement in the performance. As the threshold continues to increase, the performance decreases significantly. Besides, since the optimal threshold is different for different datasets, networks, and bit settings, this setting is cumbersome. On the contrary, from Fig. 6(e), we can observe that the performance of the quantized model is not very sensitive to the hyper-parameters when using adversarial Gaussian-margin loss. Especially in the setting of the threshold $\tau$, which not only has an insignificant effect on the performance, but its value range is also controllable.

*5) Homogeneity Issue under Varied Bits:* To investigate whether homogeneity arises consistently across various bit-widths, we conducted additional experiments. Specifically, we quantize ResNet-20 to 4-bit on CIFAR-100 and monitor critical indicators' changes during training, such as the average gradient norms of both the generator and the quantized model. Results are presented in Fig. 8. It's evident that during the mid-to-late stages of training, the average gradient norms of the generator and quantized model stabilize around 0. With minimal gradients, the generator receives few updates, leading to homogeneous synthetic data generation. This homogeneity hinders the quantized model's performance improvements by affecting its calibration. Consequently, across different bit-widths, homogeneity issues persist due to the generator's stagnant updates.

*6) Visualization:* To compare the synthetic data obtained without utilizing our method with the synthetic data obtained using our method, we conduct a visual analysis of the synthetic data. In Fig. 9, we present visualizations of synthetic data for both scenarios. While both sets of synthetic data exhibit perceptible features such as edges and contours, distinguishing between them visually proves challenging. To quantitatively assess the similarity between synthetic data, we introduce the Structural Similarity Index Measure (SSIM) [72], a widely used metric for comparing images. Throughout our training process spanning 400 epochs, we extract the first synthetic image generated at specific epochs (e.g., 240, 260, 280, 300, 320, 340, 360, 380, 400) for each scenario and compute the SSIM score between these images. Fig. 10 illustrates the computed SSIM scores. Overall, we observe lower SSIM scores between synthetic data obtained with our method compared to those obtained without it, indicating that the synthetic data generated using our method may exhibit greater heterogeneity.

## V. Conclusion

In this paper, we revisit generator-based data-free quantization from a temporal perspective and demonstrate that the synthetic data generated in the mid-to-late stages of the training process exhibits homogeneity, which hinders the performance improvement of the quantized model. To address this issue, we propose a simple yet effective adversarial Gaussian-margin loss, which can mitigate the homogenization issue by introducing the discrepancy between the quantized and full-precision models as additional supervision information to guide the update of the generator in the mid-to-late stages of the training process. Extensive experimental results demonstrate the effectiveness of our proposed method, especially at 3-bit.

However, our proposed method also has limitations. With the popularity of large models and the emphasis on data privacy, how to do data-free quantization for large models is an urgent requirement. Due to our limited hardware resources, we are unable to implement the proposed method on large models, thus the applicability of the proposed method on large models remains an open challenge. We will address this issue with more efforts in the near future.

## References

[1] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Training very deep networks," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 28, 2015, pp. 2377–2385.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[3] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully convolutional one-stage object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 9627–9636.

[4] C.-Y. Wang, A. Bochkovskiy, and H.-Y.-M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 7464–7475.

[5] X. Xie, G. Cheng, J. Wang, X. Yao, and J. Han, "Oriented R-CNN for object detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 3520–3529.

[6] B. Zhuang, C. Shen, M. Tan, L. Liu, and I. Reid, "Structured binary neural networks for accurate image classification and semantic segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 413–422.

[7] Z. Zhuang, R. Li, K. Jia, Q. Wang, Y. Li, and M. Tan, "Perception-aware multi-sensor fusion for 3D LiDAR semantic segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 16280–16290.

[8] E.-J. Ong, S. S. Husain, M. Bober-Irizar, and M. Bober, "Deep architectures and ensembles for semantic video classification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 12, pp. 3568–3582, Dec. 2019.

[9] D. Tran, H. Wang, M. Feiszli, and L. Torresani, "Video classification with channel-separated convolutional networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 5552–5561.

[10] F. Guo, W. Wang, Z. Shen, J. Shen, L. Shao, and D. Tao, "Motion-aware rapid video saliency detection," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 12, pp. 4887–4898, Dec. 2020.

[11] Z. Zhuang et al., "Discrimination-aware channel pruning for deep neural networks," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 31, 2018, pp. 875–886.

[12] H. Luo, Z. Zhuang, Y. Li, M. Tan, C. Chen, and J. Zhang, "Towards compact and robust model learning under dynamically perturbed environments," *IEEE Trans. Circuits Syst. Video Technol.*, early access, p. 1, 2023.

[13] M. Lin et al., "HRank: Filter pruning using high-rank feature map," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1529–1538.

[14] C. Liu et al., "RB-Net: Training highly accurate and efficient binary neural networks with reshaped point-wise convolution and balanced activation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 9, pp. 6414–6424, Sep. 2022.

[15] W. Xu et al., "Improving extreme low-bit quantization with soft threshold," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 4, pp. 1549–1563, Apr. 2022.

[16] J. Shi, M. Lu, and Z. Ma, "Rate-distortion optimized post-training quantization for learned image compression," *IEEE Trans. Circuits Syst. Video Technol.*, 2023.

[17] F. Tung and G. Mori, "Similarity-preserving knowledge distillation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1365–1374.

[18] K. Zhang, C. Zhang, S. Li, D. Zeng, and S. Ge, "Student network learning via evolutionary knowledge distillation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 4, pp. 2251–2263, Apr. 2022.

[19] S. Li et al., "Distilling a powerful student model via online knowledge distillation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 34, no. 11, pp. 8743–8752, 2022.

[20] A.-H. Phan et al., "Stable low-rank tensor decomposition for compression of convolutional neural network," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Glasgow, U.K.: Springer, 2020, pp. 522–539.

[21] S. Xu et al., "Generative low-bitwidth data free quantization," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2020, pp. 1–17.

[22] K. Choi, D. Hong, N. Park, Y. Kim, and J. Lee, "Qimera: Data-free quantization with synthetic boundary supporting samples," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 34, 2021, pp. 14835–14847.

[23] H. Qin et al., "Forward and backward information retention for accurate binary neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 2250–2259.

[24] J. Fang, A. Shafiee, H. Abdel-Aziz, D. Thorsley, G. Georgiadis, and J. H. Hassoun, "Post-training piecewise linear quantization for deep neural networks," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Glasgow, U.K.: Springer, 2020, pp. 69–86.

[25] P. Wang, Q. Chen, X. He, and J. Cheng, "Towards accurate post-training network quantization via bit-split and stitching," in *Proc. Int. Conf. Mach. Learn.*, 2020, pp. 9847–9856.

[26] Y. Li et al., "BRECQ: Pushing the limit of post-training quantization by block reconstruction," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2021, pp. 1–16.

[27] B. Martinez, J. Yang, A. Bulat, and G. Tzimiropoulos, "Training binary neural networks with real-to-binary convolutions," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2020, pp. 1–11.

[28] M. Lin et al., "Rotated binary neural network," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 33, 2020, pp. 7474–7485.

[29] B. Zhuang, L. Liu, M. Tan, C. Shen, and I. Reid, "Training quantized neural networks with a full-precision auxiliary module," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 1488–1497.

[30] B. Zhu, P. Hofstee, J. Peltenburg, J. Lee, and Z. Alars, "Autorecon: Neural architecture search-based reconstruction for data-free compression," in *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, 2021, pp. 1–7.

[31] B. Qian, Y. Wang, R. Hong, and M. Wang, "Adaptive data-free quantization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 7960–7968.

[32] Y. Cai, Z. Yao, Z. Dong, A. Gholami, M. W. Mahoney, and K. Keutzer, "ZeroQ: A novel zero shot quantization framework," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 13169–13178.

[33] X. Zhang et al., "Diversifying sample generation for accurate data-free quantization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 15658–15667.

[34] Y. Zhong et al., "IntraQ: Learning synthetic images with intra-class heterogeneity for zero-shot network quantization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 12339–12348.

[35] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–35.

[36] S. Xu, S. Zhang, J. Liu, B. Zhuang, Y. Wang, and M. Tan, "Generative data free model quantization with knowledge matching for classification," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 33, no. 12, pp. 7296–7309, 2023.

[37] R. Banner, Y. Nahshan, and D. Soudry, "Post training 4-bit quantization of convolutional networks for rapid-deployment," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 32, 2019, pp. 7950–7958.

[38] T. Chu, Z. Yang, and X. Huang, "Improving the post-training neural network quantization by prepositive feature quantization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 34, no. 4, pp. 3056–3060, Apr. 2024.

[39] 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.

[40] R. Zhao, Y. Hu, J. Dotzel, C. De Sa, and Z. Zhang, "Improving neural network quantization without retraining using outlier channel splitting," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 7543–7552.

[41] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha, "Learned step size quantization," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2020, pp. 1–12.

[42] R. Gong et al., "Differentiable soft quantization: Bridging full-precision and low-bit neural networks," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 4852–4861.

[43] S. Jung et al., "Learning to quantize deep networks by optimizing quantization intervals with task loss," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4350–4359.

[44] Y. Li, X. Dong, and W. Wang, "Additive powers-of-two quantization: An efficient non-uniform discretization for neural networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2020, pp. 1–15.

[45] 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.

[46] X. He, J. Lu, W. Xu, Q. Hu, P. Wang, and J. Cheng, "Generative zero-shot network quantization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2021, pp. 3000–3011.

[47] M. Haroush, I. Hubara, E. Hoffer, and D. Soudry, "The knowledge within: Methods for data-free model compression," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 8494–8502.

[48] Y. Choi, J. Choi, M. El-Khamy, and J. Lee, "Data-free network quantization with adversarial knowledge distillation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2020, pp. 710–711.

[49] C. Guo et al., "SQuant: On-the-fly data-free quantization via diagonal Hessian approximation," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2022, pp. 1–18.

[50] Y. Liu, W. Zhang, and J. Wang, "Zero-shot adversarial quantization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 1512–1521.

[51] B. Qian, Y. Wang, R. Hong, and M. Wang, "Rethinking data-free quantization as a zero-sum game," 2023, *arXiv:2302.09572*.

[52] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," 2013, *arXiv:1308.3432*.

[53] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 249–256.

[54] J. A. Bilmes et al., "A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models," *Int. Comput. Sci. Inst.*, vol. 4, no. 510, p. 126, 1998.
[55] S. Zhang et al., "Detecting adversarial data by probing multiple perturbations using expected perturbation score," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2023, pp. 41429–41451.
[56] S. Zhang, Y. Song, J. Yang, Y. Li, B. Han, and M. Tan, "Detecting machine-generated texts by multi-population aware optimization for maximum mean discrepancy," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2024, pp. 1–36.
[57] C. J. C. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining Knowl. Discovery*, vol. 2, no. 2, pp. 121–167, Jun. 1998.
[58] H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz, "Mixup: Beyond empirical risk minimization," 2017, *arXiv:1710.09412*.
[59] A. Krizhevsky et al., "Learning multiple layers of features from tiny images," M.S. thesis, Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, 2009.
[60] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2009, pp. 248–255.
[61] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 6848–6856.
[62] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2818–2826.
[63] K. Choi et al., "It's all in the teacher: Zero-shot quantization brought closer to the teacher," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 8311–8321.
[64] H. Li et al., "Hard sample matters a lot in zero-shot quantization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 24417–24426.
[65] H. Qin, Y. Ding, X. Zhang, J. Wang, X. Liu, and J. Lu, "Diverse sample generation: Pushing the limit of generative data-free quantization," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 1–18, 2023.
[66] A. Odena, C. Olah, and J. Shlens, "Conditional image synthesis with auxiliary classifier GANs," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 2642–2651.
[67] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.
[68] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–26.
[69] Y. Nesterov, "A method for solving the convex programming problem with convergence rate o $(1/k^2)$," *Doklady Akademii Nauk SSSR*, vol. 269, pp. 543–547, Jan. 1983.
[70] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 32, 2019, pp. 8026–8037.
[71] Y. Zhong et al., "Fine-grained data distribution alignment for post-training quantization," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Tel Aviv-Yafo, Israel: Springer, 2022, pp. 70–86.
[72] W. Zhou, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, pp. 600–612, 2004.

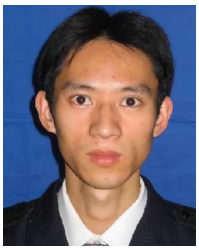
**Hui Luo** is currently pursuing the Ph.D. degree in signal and information processing with the University of Chinese Academy of Sciences, Beijing, China. During his Ph.D. program, he also conducts research with the Institute of Optics and Electronics, Chinese Academy of Sciences, Chengdu, China. His research interests include model compression and acceleration and developing robust and reliable models.

**Shuhai Zhang** is currently pursuing the Ph.D. degree with South China University of Technology, China. He has published papers in *Neural Networks*, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, ICCV, and ICML. His research interests include machine learning, model compression, and adversarial attack.

**Zhuangwei Zhuang** received the bachelor's degree in automation and engineering and the master's degree in software engineering from South China University of Technology, Guangzhou, China, in 2016 and 2018, respectively, where he is currently pursuing the Ph.D. degree with the School of Software Engineering. His research interests include model compression and 3D scene understanding for autonomous driving.

**Jiajie Mai** received the B.S. degree from Beijing University of Posts and Telecommunications in 2020 and the master's degree from King's College London in 2021. His research interests include non-convex optimization and neural networks.

**Mingkui Tan** (Member, IEEE) received the bachelor's degree in environmental science and engineering and the master's degree in control science and engineering from Hunan University, Changsha, China, in 2006 and 2009, respectively, and the Ph.D. degree in computer science from Nanyang Technological University, Singapore, in 2014. From 2014 to 2016, he was a Senior Research Associate of computer vision with the School of Computer Science, The University of Adelaide, Australia. He is currently a Professor with the School of Software Engineering, South China University of Technology, Guangzhou, China. His research interests include machine learning, sparse analysis, deep learning, and large-scale optimization.



**Jianlin Zhang** received the Ph.D. degree in signal and information processing from the University of Chinese Academy of Sciences, Beijing, China, in 2008. From 2014 to 2015, he was a Visiting Scholar with the Department of Biomedical Engineering, Johns Hopkins University, Baltimore, MD, USA. He is currently a Professor with the Institute of Optics and Electronics, Chinese Academy of Sciences, Chengdu, China. He is also a Doctoral Supervisor with the University of Chinese Academy of Sciences. His research interests include machine learning and computer vision.