ClusterQ: Semantic Feature Distribution Alignment for Data-Free Quantization

Yangcheng Gao, Zhao Zhang, *Senior Member, IEEE,* Richang Hong, *Senior Member, IEEE,* Haijun Zhang, Jicong Fan, Shuicheng Yan, *Fellow, IEEE,* and Meng Wang, *Fellow, IEEE*

***Abstract*—Network quantization has emerged as a promising method for model compression and inference acceleration. However, tradtional quantization methods (such as quantization aware training and post training quantization) require original data for the fine-tuning or calibration of quantized model, which makes them inapplicable to the cases that original data are not accessed due to privacy or security. This gives birth to the data-free quantization with synthetic data generation. While current DFQ methods still suffer from severe performance degradation when quantizing a model into lower bit, caused by the low inter-class separability of semantic features. To this end, we propose a new and effective data-free quantization method termed ClusterQ, which utilizes the semantic feature distribution alignment for synthetic data generation. To obtain high inter-class separability of semantic features, we cluster and align the feature distribution statistics to imitate the distribution of real data, so that the performance degradation is alleviated. Moreover, we incorporate the intra-class variance to solve class-wise mode collapse. We also employ the exponential moving average to update the centroid of each cluster for further feature distribution improvement. Extensive experiments across various deep models (e.g., ResNet-18 and MobileNet-V2) over the ImageNet dataset demonstrate that our ClusterQ obtains state-of-the-art performance.**

***Index Terms*—Model compression, data-free quantization, data generation, semantic feature distribution alignment, DNNs.**

I. Introduction

Deep neural network (DNN)-based models have obtained remarkable progress on computer vision tasks due to its strong representation ability [1]–[5]. However, DNN models usually suffer from high computational complexity and massive parameters, and large DNN models require frequent memory access, which will lead to much more energy consumption and inference latency [6]. Moreover, it is still challenging to deploy them on the edge devices due to the limited memory bandwidth, inference ability and energy consumption.

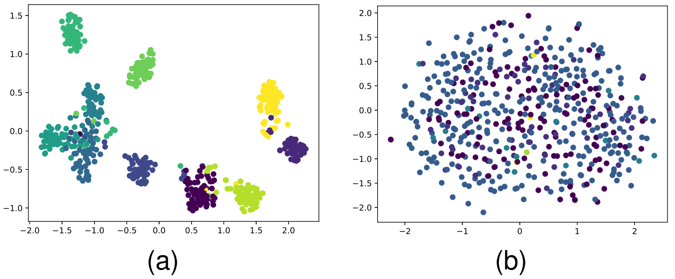


Fig. 1. t-SNE visualization comparison of the 19-th layer features of ResNet-20 [2] inferring on the CIFAR10 dataset [39] (a), and the synthetic data generated by ZeroQ [19] (b).

To solve aforementioned issues, massive model compression methods have emerged to improve the efficiency of DNN models, e.g., pruning [7]–[12], quantization [13]–[25], lightweight architecture design [26]–[29], low-rank factorization [30]–[34] and knowledge distillation [35]–[38]. Different from other model compression methods, model quantization can be implemented in real-scenario model deployment, with the low-precision computation supported on general hardware. Briefly, model quantization paradigm converts the floating-point values into low-bit integers for model compression [13]. As such, less memory access will be needed and computation latency will be reduced in model inference, which make it possible to deploy large DNN model on edge devices for real-time applications.

Due to the limited representation ability over low-bit values, model quantization usually involves noise, which potentially results in the performance degradation in reality. To recover the quantized model performance, Quantization Aware Training (QAT) performs backward propagation to retrain the quantized model [15]–[18]. However, QAT is usually time-consuming and hard to implement, so Post Training Quantization (PTQ), as an alternative method, aims at adjusting the weights of quantized model without training [14], [22], [23]. Note that QAT and PTQ need the original training data for quantization, whereas training data may be prohibited severely from access due to privacy or proprietary rules in real scenario, e.g., user data, military information, or medical images. As a result, real-world applications of QAT and PTQ may be restricted.

Recently, Data-Free Quantization (DFQ) have came into being as a more promising method for the practical applications without access to any training data, which aims at restoring the performance of quantized model by generating synthesis data, similar to the data-free knowledge distillation [37]. Current DFQ methods can be roughly divided into two categories,

Y. Gao, Z. Zhang, R. Hong and M. Wang are with the School of Computer Science and Information Engineering; also with the Key Laboratory of Knowledge Engineering with Big Data (Ministry of Education); also with the Intelligent Interconnected Systems Laboratory of Anhui Province, Hefei University of Technology, Hefei 230601, China (e-mails: gaoyangcheng576@gmail.com, cszzhang@gmail.com, hongrc.hfut@gmail.com, eric.mengwang@gmail.com).

H. Zhang is with the Department of Computer Science, Harbin Institute of Technology, Shenzhen, Shenzhen 518055, China (e-mail: fanjicong@cuhk.edu.cn).

J. Fan is with the School of Data Science, The Chinese University of Hong Kong, Shenzhen, China (e-mail: hjzhang@hit.edu.cn).

S. Yan is with the Sea AI Lab (SAIL), Singapore; also with the National University of Singapore, Singapore 117583. (e-mail: yansc@sea.com).

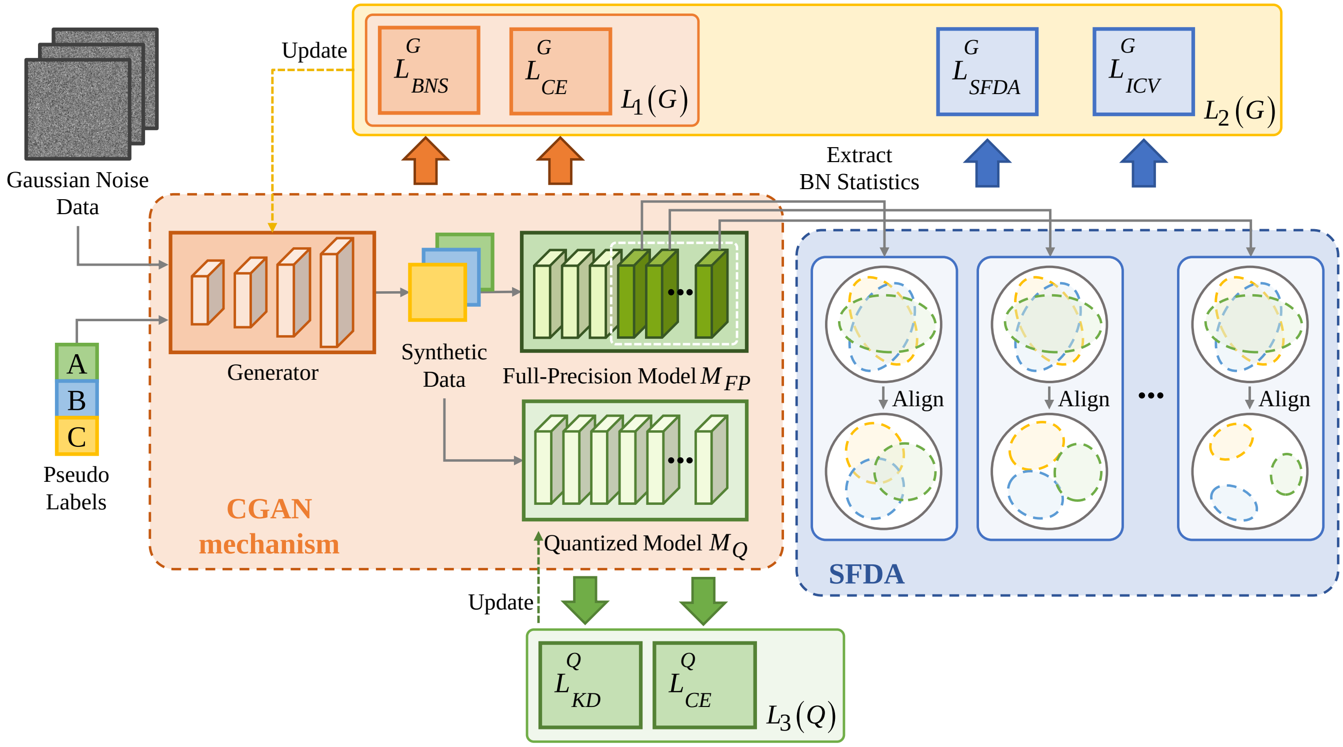


Fig. 2. Overview of the proposed ClusterQ scheme. Based on the Conditional Generative Adversarial Network (CGAN) [40] mechanism, we perform clustering and alignment on the batch normalization statistics of semantic features to obtain high inter-class separability.

i.e., without fine-tuning and with fine-tuning. Pioneer work on DFQ without fine-tuning, like ZeroQ [19], generate the calibration data that matches the batch normalization (BN) statistics of model to clip the range of activation values. However, compressed models by this way often have significant reduction in accuracy when quantizing to lower precision. In contrast, DFQ with fine-tuning applies generator to produce synthetic data and adjusts the parameters of quantized model to retain higher performance. For example, GDFQ [21] learns a classification boundary and generates data with a Conditional Generative Adversarial Network (CGAN) mechanism [40].

Although recent studies have witnessed lots of efforts on the topic of DFQ, the obtained improvements are still limited compared with PTQ, due to the existed gap between the synthetic data and real-world data. As such, how to make the generated synthetic data closer to the real-world data for fine-tuning will be a crucial issue to be solved. To close the gap, we explore the pre-trained model information at a fine-grained level. According to [41], [42], during the DNN model inferring on real data, the distributions of semantic features can be clustered for classification, i.e., inter-class separability property of semantic features. This property has also widely used in domain adaption to align the distributions of different domains. However, synthetic data generated by current DFQ methods (such as ZeroQ [19]) cannot produce semantic features with high inter-class separability in the quantized model, as shown in Figure 1. Based on this phenomenon, we can hypothesize that high inter-class separability will reduce the gap between synthetic data and real-world data. Note that this property has also been explored by FDDA [22], which augments the calibration dataset of real data for PTQ. However, there still does not exist data-free quantization method that imitates the real data distribution with inter-class separability.

From this perspective, we will propose effective strategies to generate synthetic data to obtain features with high inter-class separability and maintain the generalization performance of the quantized model for data-free case. In summary, the major contributions of this paper are described as follows:

1) Technically, we propose a new and effective data-free quantization scheme, termed ClusterQ, via feature distribution clustering and alignment, as shown in Figure 2. As can be seen, ClusterQ formulates the DFQ problem as a data-free domain adaption task to imitate the distribution of original data. To the best of our knowledge, ClusterQ is the first DFQ scheme to utilize feature distribution alignment with clusters.
2) This study also reveals that high inter-class separability of the semantic features is critical for synthetic data generation, which impacts the quantized model performance directly. We quantize and fine-tune the DNN model with a novel synthetic data generation approach without any access to original data. To achieve high inter-class separability, we propose a Semantic Feature Distribution Alignment (SFDA) method, which can cluster and align the feature distribution into the centroids for close-to-reality data generation. For further performance improvement, we introduce the intra-class variance [43] to enhance data diversity and exponential moving average

(EMA) to update the cluster centroids.

3) Based on the clustered and aligned semantic feature distributions, our ClusterQ can effectively alleviate the performance degradation, and obtain state-of-the-art results on a variety of popular deep models.

The rest of this paper is organized as follows. In Section II, we review the related work. The details of our method are elaborated in Section III. In Section IV and V, we present experiment results and analysis. The conclusion and perspective on future work are finally discussed in Section VI.

II. Related Work

We briefly review the low-bit quantization methods that are close to our study. More details can be referred to [44] that provides a comprehensive overview for model quantization.

A. Quantization Aware Training (QAT)

To avoid performance degradation of the quantized model, QAT is firstly proposed to retrain the quantized model [15]–[18]. With full training dataset, QAT performs floating-point forward and backward propagations on DNN models and quantizes them into low-bit after each training epoch. Thus, QAT can quantize model into extremely low precision while retaining the performance. In particular, PACT [15] optimizes the clipping ranges of activations during model retraining. LSQ [17] learns step size as a model parameter and MPQ [18] exploits retraining-based mix-precision quantization. However, high computational complexity of QAT will lead to restrictions on the implementation in reality.

B. Post Training Quantization (PTQ)

PTQ is proposed for efficient quantization [14], [22], [23]. Requiring for a small amount of training data and less computation, PTQ methods have ability to quantize models into low-bit precision with little performance degradation. In particular, [14] propose a clipping range optimization method with bias-correction and channel-wise bit-allocation for 4-bit quantization. [23] explore the interactions between layers and propose layer-wise 4-bit quantization. [22] explore calibration dataset with synthetic data for PTQ. However, above methods require more or less original training data, and they are inapplicable for the cases without access to original data.

C. Data-Free Quantization (DFQ)

For the case without original data, recent studies made great efforts on DFQ to generate the close-to-reality data for model fine-tuning or calibration [19]–[21], [24], [25]. Current DFQ methods can be roughly divided into two categories, i.e., without fine-tuning and with fine-tuning. Pioneer work on DFQ without fine-tuning, like ZeroQ [19], generate the calibration data that matches the batch normalization (BN) statistics. DSG [25] discovers homogenization of synthetic data and enhances the diversity of generated data. However, these methods lead to significant reduction in accuracy when quantizing to lower precision. In contrast, DFQ with fine-tuning applies generator to produce synthetic data and adjusts the parameters of quantized model to retain higher performance. For example, GDFQ [21] employs a Conditional Generative Adversarial Network (CGAN) [40] mechanism and generates dataset for fine-tuning. AutoReCon [24] enhances the generator architecture by neural architecture search. Qimera [20] exploits boundary supporting samples to enhance the classification boundary, whereas it tends to lead to mode collapse and reduce the generalization ability of quantized model.

III. ClusterQ: Semantic Feature Distribution Alignment for DFQ

For easy implementation on hardware, our ClusterQ scheme employs a symmetric uniform quantization, which maps and rounds the floating-point values of full-precision model to low-bit integers. Given a floating-point value x in a tensor $\boldsymbol{x}$ to be quantized, it can be defined as follows:

$$x_q = round(x/\Delta),\ \Delta = \frac{2\alpha}{2^N - 1}, \tag{1}$$

where x_q is the quantized value, N is the quantization bit width, α denotes the clipping range, Δ is the scaling factor to map floating-point value x within clipping range into the range of $[0,\ 2^N - 1]$ and $round(\cdot)$ represents the rounding operation. For most symmetric uniform quantization, α is defined by the maximum of absolute values ,i.e, $\alpha = max(|\boldsymbol{x}|)$, so that all of the values can be represented. Then, we can easily obtain the dequantized value x_d as follows:

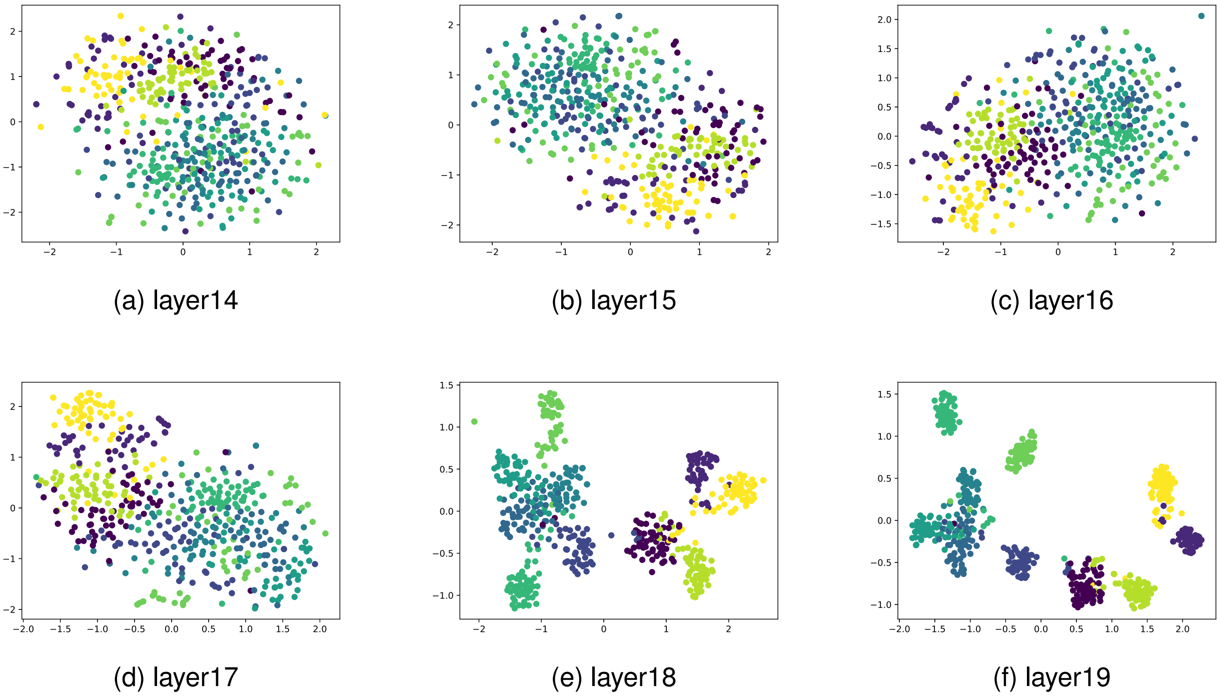
$$x_d = x_q \cdot \Delta\ . \tag{2}$$

Due to the poor representation ability of limited bit width, there exists quantization error between the dequantized value x_d and the original floating-point value x, which may involve quantization noise and lead to accuracy loss.

To recover the quantized model performance, there exist two challenges for DFQ methods: (1) For statistic activation quantization, clipping range of activation values should be determined without access to the training data. (2) To recover the degraded performance, fine-tuning is used to adjust the weights of quantized models without training data. To solve these challenges, current DFQ methods try to generate synthetic data which are similar to the original training data. For example, GDFQ [21] employs a CGAN-based mechanism for fake samples generation. Given a fixed original full-precision model M_{FP} as the discriminator, a generator G is trained to produce synthetic data that are close to the original training data. More details can be referred to [21].

However, without clustering and alignment of the semantic feature distributions, generated synthetic data used for fine-tuning the quantized model will lead to limited performance recovery. According to [41], traits of data domain are contained in the semantic feature distributions. The knowledge of the full-precision pre-trained model can be further used for synthetic data generation by clustering the semantic feature distributions. From our perspective, this will be the most critical factor for the performance recovery of quantized model.

To utilize the distribution of semantic features, we further exploit the Batch Normalization (BN) statistics [45] to imitate



(a) layer14

(b) layer15

(c) layer16

(d) layer17

(e) layer18

(f) layer19

Fig. 3. t-SNE visualization results of the deep layer features in ResNet-20 model inferring on CIFAR-10. From (a) to (f) correspond to the features from 14th layer to 19th layer. The inter-class separability is enhanced as the layer gets deeper.

the original distribution. Next, we briefly review the BN layer in DNN models, which is designed to alleviate the internal covariate shifting. Formally, with a mini-batch input $\boldsymbol{X}_B = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_m\}$ of batch size m, the BN layer will transfer the input $\boldsymbol{X}_B$ into the following expression:

$$\begin{aligned} \hat{\boldsymbol{x}}_i &\leftarrow \frac{\boldsymbol{x}_i - E[\boldsymbol{X}_B]}{\sqrt{Var[\boldsymbol{X}_B] + \epsilon}}, \\ \boldsymbol{y}_i &\leftarrow \gamma_i \hat{\boldsymbol{x}}_i + \beta_i, \end{aligned} \tag{3}$$

where x_i and y_i denote the input and output of BN layer respectively, γ_i and β_i denote the parameters learned during training. After training, the distribution of input in each layer will be stable across training data.

A. *Proposed Framework*

The overview of our proposed ClusterQ is presented in Figure 2, which is based on the CGAN mechanism. Specifically, ClusterQ employs the fixed full-precision model M_{FP} as a discriminator. The generator G is trained by the loss $L_2(G)$ to produce fake data to fine-tune the quantized model M_Q by computing the loss $L_3(M_Q)$.

The loss $L_2(G)$ contains $L_1(G)$ for classification and global distribution information matching. More importantly, $L_2(G)$ introduces the $L^G_{SFDA}(G)$ for distribution clustering and alignment to achieve inter-class separability in semantic layer. Thus, the synthetic data can imitate the distributions of real data in feature level of pre-trained model. To adapt the distribution change during generator training, we implement the dynamic centroid update by EMA. Moreover, to avoid mode collapse, we still introduce the intra-class variance loss $L^G_{ICV}(G)$ to improve the diversity of synthetic data.

To highlight our motivation on the inter-class separability of semantic features, we conduct some pilot experiments on the DNN features to observe the dynamic transformation of this separability over different layers, as illustrated in Figure 3. As the layer getting deeper, the feature distributions are more separable and can be easily clustered or grouped. Specifically, we can easily distinguish the features of the 18th and 19th layers (see Figure 3(e) and 3(f)), while the boundaries of clusters become blurred in the 16th and 17th layers (see Figure 3(c) and 3(d)). For more shallow layers (see Figure 3(a) and 3(b)), almost no boundary exists.

Based on high inter-class separability of semantic features, and we can model the semantic feature distribution as a Gaussian distribution [14]. That is, the semantic feature statistics for different classes will also be clustered into groups. As such, we directly utilize the Batch Normalization statistics that save running statistics for feature clustering and alignment.

The structure of SFDA is shown in Figure 4. In the finetuning process of quantized model, the running BN statistics corresponding to the given pseudo labels are extracted and aligned to the centroids in each layer. The distance between running statistics and centroids is computed to update the generator G. The SFDA process is elaborated below.

1) First, after the generator G warms up, with a given pretrained full-precision model, we initialize the centroids for each class in each semantic layer. Note that the warm-up process is prerequisite for the centroids initialization to generate the synthetic data with diversity.

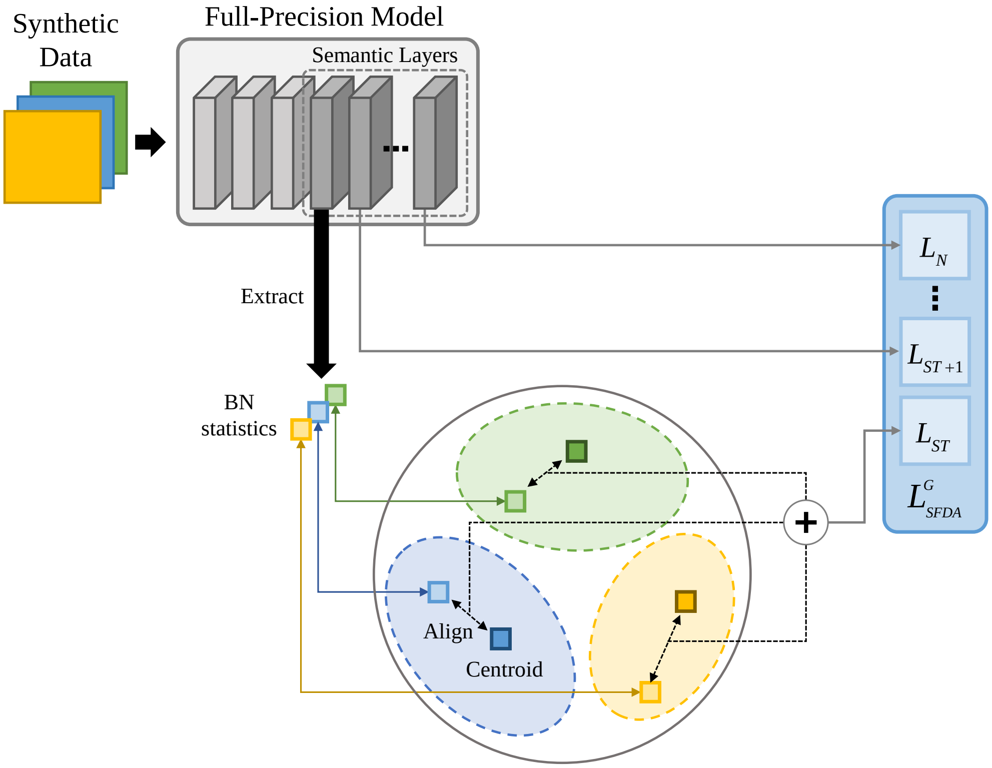


Fig. 4. The structure of SFDA method. BN statistics in each semantic layer are class-wisely extracted, clustered and aligned to the corresponding centroids. The SDFA loss is computed to update the generator. The pseudo labels of centroids, statistics and synthetic data are represented by different colors.

To initialize the centroids, we pass the pseudo label of each class to the generator, infer full-precision model on the synthetic data and extract the corresponding BN statistics in each semantic layer.

2) Then, we formulate the problem as a domain adaption task, and treat the centroids and running BN statistics as target distribution and source distribution. As such, we perform distribution alignment in each semantic layer. The Euclidean distance between running BN statistics and centroids can be calculated by the following SFDA loss function $L^G_{SFDA}(G)$ to align them:

$$L^G_{SFDA}(G) = \sum_{C=0}^{N_C} \sum_{l=l_{st}}^{L} \left\| \hat{\mu}_l^C - \mu_l^C \right\|_2^2 + \left\| \hat{\sigma}_l^C - \sigma_l^C \right\|_2^2, \tag{4}$$

where $\hat{\mu}_l^C$ and $\hat{\sigma}_l^C$ are mean and standard deviation for class C at the lth layer in the full-precision model computed in the process of generator training, μ_l^C and σ_l^C represent the corresponding mean and standard deviation of the centroids, respectively. l_{st} denotes the starting layer that contains semantic features. And N_C denotes the number of classes. To avoid imbalance among categories caused by the random labels, we traverse all categories by employing the pseudo labels, and compute the SFDA loss $L^G_{SFDA}(G)$ independently.

Specifically, according to our experiment results, the SFDA process can significantly promote the generator to produce synthetic data with high inter-class separability of semantic features. During the fine-tuning process, the learned classification boundary will be further enhanced. In addition, to avoid misclassification caused by the pre-trained model, or the gap between synthetic data and real data, we discard the BN statistics obtained by misclassified synthetic data during the generator training process.

B. Centroids Updating

The initialization of centroids may be unstable for SFDA. First, the initialization of centroids is based on the assumption that the semantic feature distributions obtained by synthetic data and real data are close. However, due to the intrinsic limitation of generator, even if the generator G has been warming up, there still remains a gap to the real data which may lead to centroids mismatch and limit further distribution alignment. Specifically, the inter-class separability may be more obvious along with further generator training, and the original centroids will be no longer appropriate to the situation.

For these reasons, we need to update the centroids during generator training to release the negative effects. Thus, we update the centroids by the running BN statistics during generator training. Considering the SFDA method as a clustering method, we apply exponential moving average (EMA) directly on it to update the centroids as follows:

$$\begin{cases} \mu_l^C = (1-\beta_{SFDA})\mu_l^C + \beta_{SFDA}\hat{\mu}_l^C \\ \sigma_l^C = (1-\beta_{SFDA})\sigma_l^C + \beta_{SFDA}\hat{\sigma}_l^C \end{cases}, \tag{5}$$

where $\hat{\mu}_l^C$ and $\hat{\sigma}_l^C$ denote the running mean and standard deviation corresponding to class C, respectively. β_{SFDA} is the

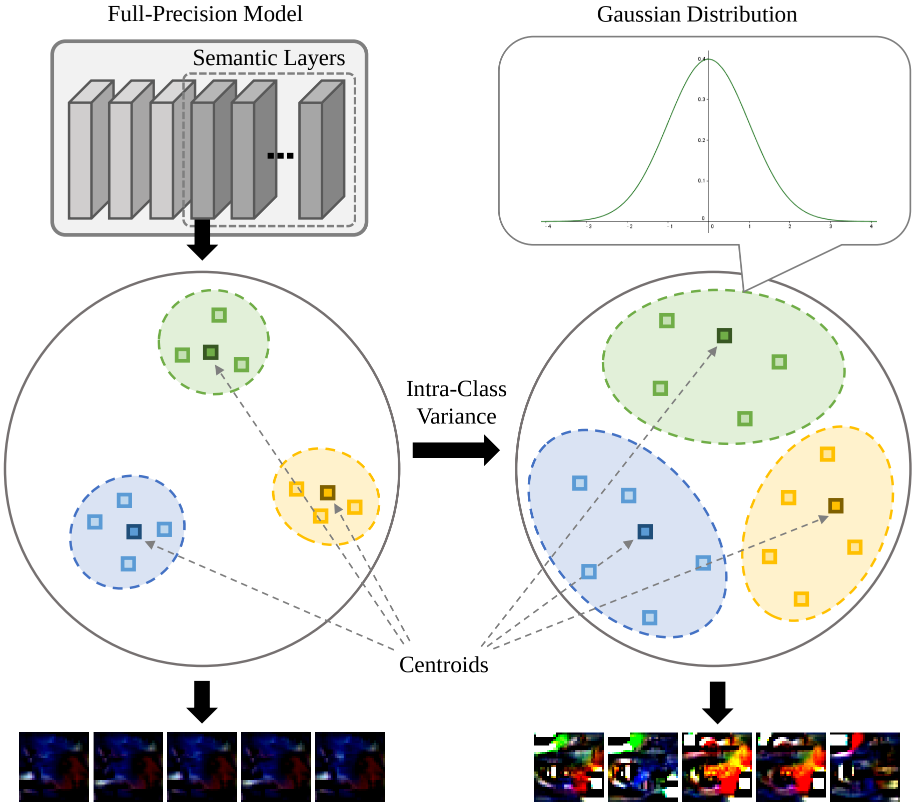


Fig. 5. The effect of Intra-Class Variance. With introduction of intra-class variance loss L_{ICV}, the BN statistic distribution is allowed to shift around the centroids and follow Gaussian distribution. As result, the mode collapse is mitigated in data generation.

decay rate of EMA, which trades off the importance of previous and current BN statistics. Thus, BN centroids can make the SFDA process a grouping method with decentralization property. We will provide experimental results to demonstrate the performance promotion via centroids updating.

C. Intra-Class Variance

Although our proposed ClusterQ can obtain high inter-class separability of semantic features, the distribution alignment may also cause vulnerability of mode collapse which will also degrade the generalization performance of quantized model. That is, the distribution of real data cannot be covered by the synthetic data. For example, given Gaussian input, some generators produce data in fixed mode.

To expand mode coverage, we employ a simple method following [22] to shift the BN statistic distribution around the cluster. Specifically, due to the semantic feature distribution approximately following Gaussian distribution, we introduce Gaussian noise to increase the intra-class discrepancy within clusters and define the intra-class variance loss L_{ICV} as

$$\begin{aligned} L_{ICV}(G) = \sum_{C=0}^{N_C} \sum_{l=l_{st}}^{L} & \left\| \hat{\mu}_l^C - \mathcal{N}(\mu_l^C, \lambda_\mu) \right\|_2^2 \\ & + \left\| \hat{\sigma}_l^C - \mathcal{N}(\sigma_l^C, \lambda_\sigma) \right\|_2^2, \end{aligned} \tag{6}$$

where $\mathcal{N}(\cdot,\cdot)$ denotes Gaussian noise, λ_μ and λ_σ denote the distortion levels to control intra-class variance. In this way, we can allow the running mean $\hat{\mu}_l^C$ and standard deviation $\hat{\sigma}_l^C$ for each class C to shift within a dynamic range around the cluster centroids μ_l^C and σ_l^C respectively. As shown in Figure 5, semantic feature distribution space cannot be covered without intra-class variance, therefore generated data will encounter mode collapse and lead to poor performance. In contrast, diversity images can be produced with introduction of intra-class variance loss L_{ICV}. Experiments have verified the effect of intra-class variance loss L_{ICV} to mitigate the mode collapse in synthetic data generation.

D. Training Process

For better understanding of our quantization scheme, we summarize the whole training process in Algorithm 1. With the low-bit model M_Q quantized by Eq.(1) and the full-precision model M_{FP} as discriminator, our ClusterQ scheme trains the generator G to produce synthetic data and updates the parameters of the quantized model M_Q alternately. Note that our implementation is based on the framework of GDFQ [21].

At the beginning of the generator G training, i.e., warm-up process, we fix the weights and BN statistics of quantized model M_Q to avoid being updated, because the generated synthetic data lack of diversity and textures. The loss function $L_1(G)$ is denoted as follows:

$$L_1(G) = L_{CE}^G(G) + \alpha_1 L_{BNS}^G(G), \tag{7}$$

where α_1 is a trade-off parameter. The term $L^G_{CE}(G)$ utilizes cross-entropy loss function $CE(\cdot, \cdot)$ with given Gaussian noise z and pseudo labels y to update the generator G:

$$L^G_{CE}(G) = \mathbb{E}_{z \sim y}[CE(M_{FP}(G(z|y)),\ y)]. \tag{8}$$

And the term $L^G_{BNS}(G)$ denotes the loss to match BN statistics in each layer, denoted as follows:

$$L^G_{BNS}(G) = \sum_{l=1}^{L} \left\| \hat{\mu}_l - \mu_l \right\|_2^2 + \left\| \hat{\sigma}_l - \sigma_l \right\|_2^2, \tag{9}$$

where $\hat{\mu}_l$ and $\hat{\sigma}_l$ are the running mean and standard deviation in the lth layer, while μ_l and σ_l are original mean and standard deviation stored in BN layer at the lth layer of full-precision model M_{FP}. Note that $L^G_{BNS}(G)$ is totally different from the SFDA loss $L^G_{SFDA}(G)$, even if they look somewhat similar.

After finishing the warm-up process, we utilize the synthetic data to fine-tune the quantized model, and initialize the BN statistic centroids. Then, the SFDA loss $L^G_{SFDA}(G)$ and the intra-class variance loss $L^G_{ICV}(G)$ will be added into the loss function $L_2(G)$ for generator training, formulated as

$$\begin{aligned} L_2(G) = L^G_{CE}(G) + \alpha_1 L^G_{BNS}(G) \\ + \alpha_2 L^G_{SFDA}(G) + \alpha_3 L^G_{ICV}(G), \end{aligned} \tag{10}$$

where α_2 and α_3 is trade-off parameters. After that, the centroids will be updated by EMA.

To fine-tune the quantized model M_Q, we use the following loss function $L_3(M_Q)$:

$$L_3(M_Q) = L^Q_{CE}(M_Q) + \gamma L^Q_{KD}(M_Q), \tag{11}$$

where γ is a trade-off parameter. With the synthetic data and corresponding pseudo label y, term $L^Q_{CE}(M_Q)$ utilizes the cross-entropy loss function $CE(\cdot, \cdot)$ to update the parameters of quantized model as follows:

$$L^Q_{CE}(M_Q) = \mathbb{E}_{\hat{\boldsymbol{x}} \sim y}[CE(M_Q(\hat{\boldsymbol{x}}),\ y)]. \tag{12}$$

And the knowledge distillation loss function $L^Q_{KD}(M_Q)$ via Kullback-Leibler divergence loss $KLD(\cdot, \cdot)$ is employed to compare the outputs of quantized model M_Q and full-precision model M_{FP}, which is formulated as follows:

$$L^Q_{KD}(M_Q) = \mathbb{E}_{\hat{\boldsymbol{x}}}[KLD(M_Q(\hat{\boldsymbol{x}}), M_{FP}(\hat{\boldsymbol{x}}))]. \tag{13}$$

Note the parameters of full-precision model M_{FP} are fixed during the whole training process to avoid modification.

IV. Experiments

A. Experimental Setting

We compare each method on several popular datasets, including CIFAR10, CIFAR100 [39] and ImageNet (ILSVRC12) [46]. With 60 thousand images of pixels 32×32, CIFAR10 and CIFAR100 datasets contain 10 categories for classification. ImageNet has 1000 categories for classification with 1.2 million training images and 150 thousand images for validation.

For experiments, we perform quantization on ResNet-18 [2], MobileNet-V2 [26] on ImageNet, and also ResNet-20 on CIFAR10 and CIFAR100. All experiments are conducted on an NVIDIA RTX 2080Ti GPU with PyTorch [47]. Note that all of the pre-trained model implementations and weights are provided by Pytorchcv[1].

Algorithm 1 ClusterQ Training

Input: Generator G with random initialization, pre-trained full-precision model M_{FP}.

Parameter: Number of training epoch N, number of warm-up epoch N_w and number of fine-tuning step N_{ft}

Output: Trained generator G and quantized model M_Q.

```
Quantize $M_{FP}$ and obtain the quantized model $M_Q$.
Fix BN statistics of quantized model $M_Q$.
for epoch $\leftarrow 1$ to $N$ do
    if epoch $< N_w$ then
        Train generator $G$ with $L_1(G)$ in Eq.(7).
    else
        if epoch $= N_w$ then
            Initialize the centroids.
        else
            for step $\leftarrow 1$ to $N_{ft}$ do
                Generate synthetic data with Gaussian noise $z$
                and pseudo labels $y$.
                Train generator $G$ with $L_2(G)$ in Eq.(10).
                Update the centroids with EMA in Eq.(5).
                Fine-tune $M_Q$ with $L_3(Q)$ in Eq.(11).
            end for
        end if
    end if
end for
```

For implementation, we follow some hyperparameter settings of GDFQ [21]. The number of training epoch is set to 400 and the number of fine-tuning epoch is set to 200. We set 50 epochs for the warm-up process and the rest epochs to update generator G and quantized model M_Q alternately. For the trade-off parameters in Eqs.(10) and (10), we set 0.1 for α_1, 0.9 for α_2, 0.6 for α_3 and 1.0 for γ. For EMA, we set the decay rate β_{SFDA} to 0.2. In L_{ICV}, the distortion levels of Gaussian noise λ_μ and λ_σ are set to 0.5 and 1.0, respectively. For the sake of implementation on hardware, we choose the fixed precision quantization for experiments.

B. Comparison Results

To demonstrate the performance of our ClusterQ, we compare it with several closely-related methods, i.e., ZeroQ [19], GDFQ [21], Qimera [20], DSG [25] and AutoReCon [24]. The comparison results based on ImageNet, CIFAR100 and CIFAR10 are described in Tables I, II and III, respectively. Note that WnAm stands for the quantization bit-width with n-bit weight and m-bit activation. The baseline with W32A32 denotes the full-precision model accuracy. The character † means that the result is obtained by ourselves. By considering the practical applications, we also conduct quantization experiments with different precision settings. Moreover, we choose the bit number with power of two in all experiments for facilitating the deployment.

[1] Computer vision models on PyTorch: https://pypi.org/project/pytorchcv/

TABLE I
COMPARISON RESULT OF EACH METHOD ON IMAGENET DATASET.

DNN Model	Precision	Quantization Method	Top1 Accuracy
ResNet-18	W32A32	Baseline	71.470%
	W4A4	ZeroQ	20.770%
		GDFQ	60.704%
		DSG	34.530%
		Qimera	63.840%
		AutoReCon	61.600%
		Ours	**64.390%**
	W4A8	ZeroQ†	51.176%
		GDFQ†	64.810%
		Qimera†	65.784%
		Ours	**67.826%**
	W8A8	GDFQ†	70.788%
		Qimera†	70.664%
		Ours	**70.838%**
MobileNet-V2	W32A32	Baseline	73.084%
	W4A4	ZeroQ	10.990%
		GDFQ	59.404%
		Qimera	61.620%
		AutoReCon	60.020%
		Ours	**63.328%**
	W4A8	ZeroQ†	13.955%
		GDFQ†	64.402%
		Qimera†	66.486%
		Ours	**68.200%**
	W8A8	GDFQ†	72.814%
		Qimera†	72.772%
		Ours	**72.82%**

TABLE II
COMPARISON RESULTS ON CIFAR100 DATASET.

DNN Model	Precision	Quantization Method	Top1 Accuracy
ResNet-20	W32A32	Baseline	70.33%
	W4A4	ZeroQ	45.20%
		GDFQ	63.91%
		Qimera	65.10%
		Ours	**67.09%**
	W4A8	ZeroQ†	58.606%
		GDFQ†	67.33%
		Qimera†	68.89%
		Ours	**69.68%**
	W8A8	ZeroQ†	70.128%
		GDFQ†	70.39%
		Qimera†	70.40%
		Ours	**70.43%**

1) Results on ImageNet: As can be seen in Table I, with the same precision setting based on the ResNet-18 and MobileNet-V2, our method performs better than its competitors. Specifically, our method performs beyond the most closely-related GDFQ method a lot, especially for the case of lower precision. By comparing with the current state-of-the-art method Qimera, our method still outperforms it 1.708% for MobileNet-V2 that is, in fact, more difficult to be compressed due to smaller weights. One can also note that, with the reduction of precision bits, the presentation ability of the quantized value becomes limited and leads to more performance degradation. In this case, our ClusterQ retains the performance of quantized model better than other compared competitors.

TABLE III
COMPARISON RESULTS ON CIFAR10 DATASET.

DNN Model	Precision	Quantization Method	Top1 Accuracy
ResNet-20	W32A32	Baseline	93.89%
	W4A4	ZeroQ	73.53%
		GDFQ	86.23%
		Qimera	91.23%
		Ours	**92.06%**
	W4A8	ZeroQ†	90.845%
		GDFQ†	93.74%
		Qimera†	93.63%
		Ours	**93.84%**
	W8A8	ZeroQ†	93.94%
		GDFQ†	93.98%
		Qimera†	93.93%
		Ours†	**94.07%**

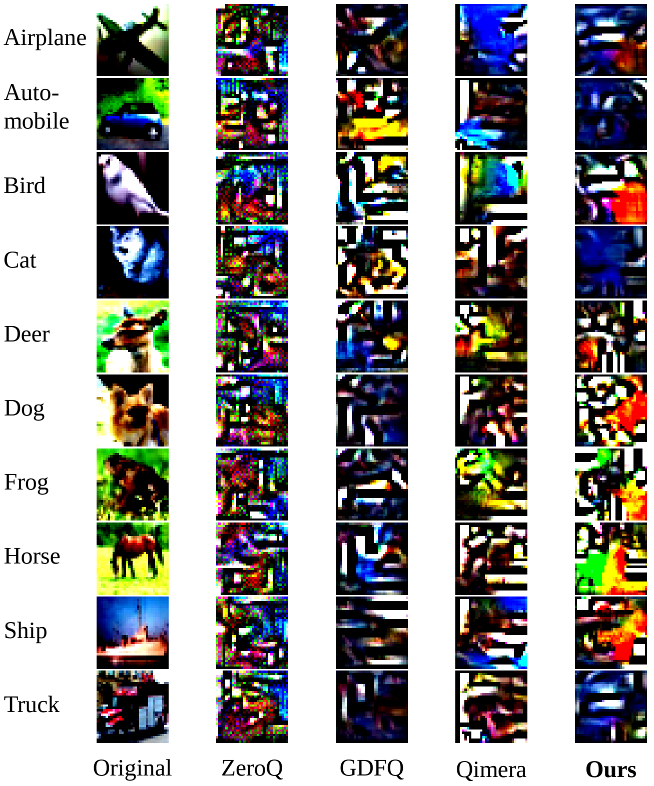


Fig. 6. Synthetic data generated by the pre-trained ResNet-20 model on CIFAR10 dataset. Each row denotes different classes, except for ZeroQ, since it generates data without labels.

2) Results on CIFAR10 and CIFAR100: From the results in Tables II and III based on ResNet-20, similar conclusions can be obtained. That is, our method surpasses the current state-of-the-art methods in terms of accuracy loss in this investigated case. In other words, the generalization performance of our method on different models and datasets can be verified.

C. Visual Analysis

In addition to the above numerical results, we also would like to perform the visual analysis on the generated synthetic data, which will directly impact the performance recovery of each quantized model. In Figure 6, we visualize the synthetic

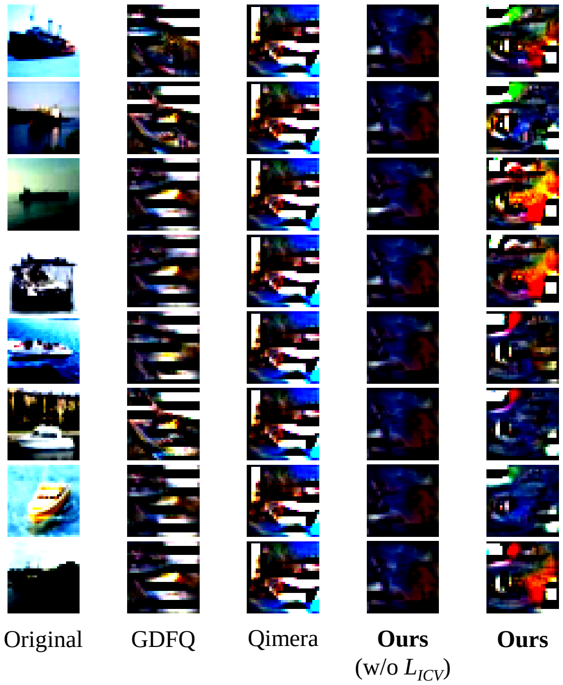


Fig. 7. Randomly selected synthetic data (label="ship") with the pre-trained ResNet-20 model on CIFAR10 dataset. Note that "w/o L_{ICV}" denotes the results from ClusterQ without intra-class variance loss L_{ICV}.

TABLE IV
ABLATION STUDY RESULTS OF RESNET-18 ON THE IMAGENET DATASET WITH THE PRECISION OF W4A4.

Model	L_{ICV}	EMA	Top1
ResNet-18	√	√	64.390%
	√	-	63.646%
	-	√	63.590%
	-	-	63.068%

data with labels generated by existing methods (i.e., ZeroQ, GDFQ and Qimera) based on the ResNet-20 over CIFAR10. We select the synthetic data with label "ship" as an example and show the results in Figure 7.

As shown in Figure 6, due to lack of label information, the data generated by ZeroQ have less class-wise discrepancy. For GDFQ, the generated data can be distinguished into different classes, but containing less detailed textures. Based on the SFDA, our ClusterQ can produce the synthetic data with more useful information. With abundant color and texture, the data generated by Qimera are similar to that of ours. However, as shown in Figure.7, the little variance of the images within each class indicates that they encounter class-wise mode collapse . In contrast, by simultaneously considering the contribution of intra-class variance, the generated synthetic data of the same class by ClusterQ can maintain variety on color, texture and structure. To illustrate the effect of intra-class variance, in Figure.7 we also visual the synthetic data produced by ClusterQ without L_{ICV} which lead to class-wise mode collapse.

D. Ablation Studies

We first evaluate the effectiveness of each component in our ClusterQ, i.e., intra-class variance and EMA. We conduct experiments to quantize the ResNet-18 into W4A4 on ImageNet dataset, and describe the results in Table IV. We see that without the intra-class variance or EMA, the performance improvement of quantized model is limited. That is, both intra-class variance or EMA are important for our method.

Then, we also analyze the sensitivity of our method to the decay rate β_{SFDA} in Figure 8. According to III-B, we set the decay rate β_{SFDA} to control the centroid updating and trade It is clear that the quantized model achieves the best result, when β_{SFDA} equals to 0.2. The performance is reduced when the decay rate is lower than 0.2, since in such cases the centroids cannot adapt to the distribution changing. Moreover, if β_{SFDA} is increased beyond 0.2, the centroids will fluctuate. The above situations lead to performance degradation.

In addition, to explore the effect of the trade-off parameter α_3, we conduct a series of experiments with different settings of α_3. As shown in Figure 9, when α_3 goes up to 0.6, the performance of quantized model will increase. It demonstrates that intra-class variance can improve the quality of synthetic data and lead to performance promotion. However, the performance of quantized model falls down, when α_3 goes above 0.6. Higher trade-off hyperparameter α_3 will enhance the effect of L_{ICV} and broke the classification boundary. In summary, we should set α_3 with consideration of model representation ability and the distribution of original dataset.

V. DISCUSSION

A. On Prior Information

It may be easy to misunderstand that our proposed ClusterQ method depends on the prior information that are provided by the pseudo labels. As such, we want to clarify the classification labels are presented as one-hot vectors and described by the class indices during the whole quantization process. Thus, the only thing our framework needs is the number of classes rather than specific classes. In fact, the number of classes can be obtained by the dimension of the weights in the last layer, even if we have no idea about the class information of dataset. Then, we can create the pseudo labels with class indices and compute the loss function with the output.

B. About Privacy and Secrecy

Prohibition of access to the original data is one of the most important motivations for DFQ methods. Someone may worry the generator-based mechanism or by synthetic data generation will violate the privacy. However, in fact, due to the black box computing manner of deep learning and the limitation of current intelligent technologies, the synthetic images generated by our method still cannot be interpreted by human beings, as shown in Figures 6 and 7.

C. Limitations of our ClusterQ

The proposed scheme utilizes the property of class-wise separability of feature distribution and performs class-wise

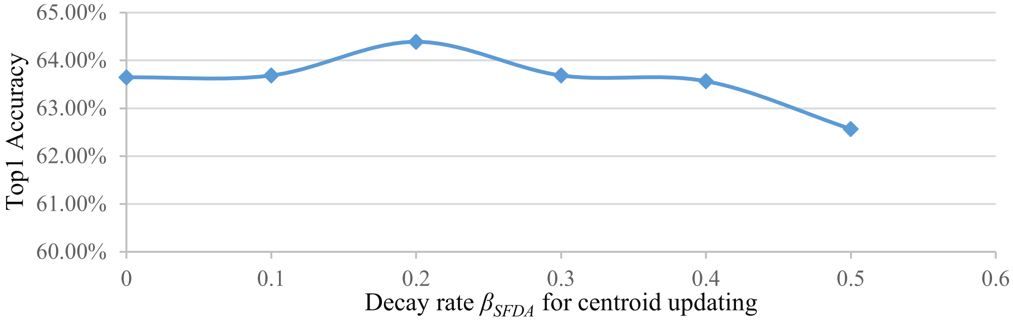


Fig. 8. Sensitivity analysis of the decay rate of EMA for centroid updating. We conduct the experiments by quantizing ResNet-18 on ImageNet dataset. The quantized model performs the best at the point of $\beta_{SFDA} = 0.2$.

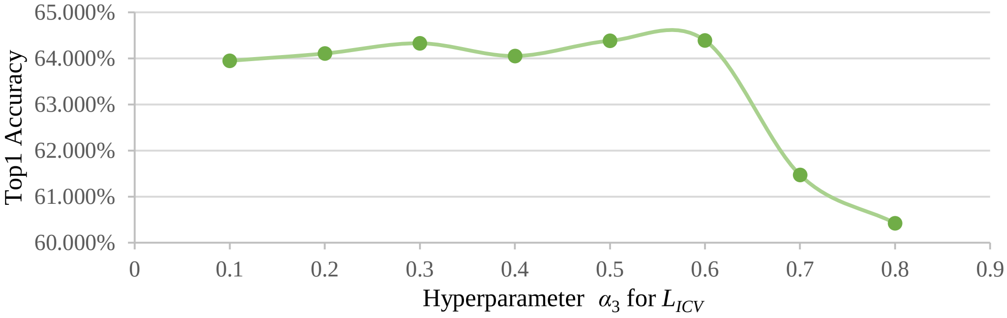


Fig. 9. Sensitivity analysis of the α_3 for intra-class variance. We conduct the experiments by quantizing ResNet-18 on ImageNet dataset. As α_3 goes up to 0.6, performance of quantized model will increase. But the performance of quantized model falls down while α_3 goes above 0.6.

statistic alignment by CGAN-like mechanism to improve the diversity of synthetic data. However, compared with those methods without fine-tuning, such as ZeroQ, generator-based methods always require for time and computation resources to train the generator. What's more, for different computer vision tasks, we have to design new generator with the embedding capability of the corresponding label format. For deep models without BN layer, e.g., ZeroDCE [48], generative DFQ method can not distill the statistics directly from pre-trained model.

VI. Conclusion

We have investigated the problem of alleviating the performance degradation when quantizing a model, by enhancing the inter-class separability of semantic features. Technically, a new and effective data-free quantization method referred to as ClusterQ is proposed. The setting of ClusterQ presents a new semantic feature distribution alignment for synthetic data generation, which can obtain high class-wise separability and enhance the diversity of the generated synthetic data. To further improve the feature distribution and the performance of data-free quantization, we also incorporate the ideas of intra-class variance and exponential moving average, so that the feature distributions are more accurate. Extensive experiments based on different DNN models and datasets demonstrate that our method can achieve state-of-the-art performance among current data-free quantization methods, especially for smaller network architectures. In future work, we will focus on exploring how to extend our ClusterQ to other vision tasks. The deployment of our proposed data-free quantization method into edge devices will also be investigated.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[5] Y. Wei, Z. Zhang, Y. Wang, M. Xu, Y. Yang, S. Yan, and M. Wang, "Deraincyclegan: Rain attentive cyclegan for single image deraining and rainmaking," *IEEE Transactions on Image Processing*, vol. 30, pp. 4788–4801, 2021.

[6] S. Han, X. Liu, H. Mao, J. Pu, A. Pedram, M. A. Horowitz, and W. J. Dally, "Eie: Efficient inference engine on compressed deep neural network," *ACM SIGARCH Computer Architecture News*, vol. 44, no. 3, pp. 243–254, 2016.

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

[8] F. Esposito, D. Malerba, G. Semeraro, and J. Kay, "A comparative analysis of methods for pruning decision trees," *IEEE transactions on pattern analysis and machine intelligence*, vol. 19, no. 5, pp. 476–491, 1997.

[9] J. Markel, "Fft pruning," *IEEE transactions on Audio and Electroacoustics*, vol. 19, no. 4, pp. 305–311, 1971.

[10] X. Ruan, Y. Liu, C. Yuan, B. Li, W. Hu, Y. Li, and S. Maybank, "Edp: An efficient decomposition and pruning scheme for convolutional neural network compression," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 10, pp. 4499–4513, 2020.

[11] Z. Chen, T.-B. Xu, C. Du, C.-L. Liu, and H. He, "Dynamical channel pruning by conditional accuracy change for deep neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 2, pp. 799–813, 2020.

[12] V. N. Ioannidis, S. Chen, and G. B. Giannakis, "Efficient and stable graph scattering transforms via pruning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[13] B. Jacob, S. Kligys, B. Chen, M. Zhu, M. Tang, A. Howard, H. Adam, and D. Kalenichenko, "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2704–2713.

[14] R. Banner, Y. Nahshan, E. Hoffer, and D. Soudry, "Post-training 4-bit quantization of convolution networks for rapid-deployment," *arXiv preprint arXiv:1810.05723*, 2018.

[15] J. Choi, Z. Wang, S. Venkataramani, P. I.-J. Chuang, V. Srinivasan, and K. Gopalakrishnan, "Pact: Parameterized clipping activation for quantized neural networks," *arXiv preprint arXiv:1805.06085*, 2018.

[16] M. Courbariaux, Y. Bengio, and J.-P. David, "Binaryconnect: Training deep neural networks with binary weights during propagations," in *Advances in neural information processing systems*, 2015, pp. 3123–3131.

[17] S. K. Esser, J. L. McKinstry, D. Bablani, R. Appuswamy, and D. S. Modha, "Learned step size quantization," *arXiv preprint arXiv:1902.08153*, 2019.

[18] N. Kim, D. Shin, W. Choi, G. Kim, and J. Park, "Exploiting retraining-based mixed-precision quantization for low-cost dnn accelerator design," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 7, pp. 2925–2938, 2020.

[19] Y. Cai, Z. Yao, Z. Dong, A. Gholami, M. W. Mahoney, and K. Keutzer, "Zeroq: A novel zero shot quantization framework," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 13 169–13 178.

[20] K. Choi, D. Hong, N. Park, Y. Kim, and J. Lee, "Qimera: Data-free quantization with synthetic boundary supporting samples," in *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[21] S. Xu, H. Li, B. Zhuang, J. Liu, J. Cao, C. Liang, and M. Tan, "Generative low-bitwidth data free quantization," in *European Conference on Computer Vision*. Springer, 2020, pp. 1–17.

[22] Y. Zhong, M. Lin, M. Chen, K. Li, Y. Shen, F. Chao, Y. Wu, F. Huang, and R. Ji, "Fine-grained data distribution alignment for post-training quantization," *arXiv preprint arXiv:2109.04186*, 2021.

[23] Y. Nahshan, B. Chmiel, C. Baskin, E. Zheltonozhskii, R. Banner, A. M. Bronstein, and A. Mendelson, "Loss aware post-training quantization," *Machine Learning*, pp. 1–18, 2021.

[24] B. Zhu, P. Hofstee, J. Peltenburg, J. Lee, and Z. Alars, "Autorecon: Neural architecture search-based reconstruction for data-free compression," *arXiv preprint arXiv:2105.12151*, 2021.

[25] X. Zhang, H. Qin, Y. Ding, R. Gong, Q. Yan, R. Tao, Y. Li, F. Yu, and X. Liu, "Diversifying sample generation for accurate data-free quantization," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 658–15 667.

[26] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[27] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan *et al.*, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1314–1324.

[28] N. Ma, X. Zhang, H.-T. Zheng, and J. Sun, "Shufflenet v2: Practical guidelines for efficient cnn architecture design," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 116–131.

[29] H. Liu, K. Simonyan, and Y. Yang, "Darts: Differentiable architecture search," *arXiv preprint arXiv:1806.09055*, 2018.

[30] E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus, "Exploiting linear structure within convolutional networks for efficient evaluation," *Advances in neural information processing systems*, vol. 27, 2014.

[31] M. Jaderberg, A. Vedaldi, and A. Zisserman, "Speeding up convolutional neural networks with low rank expansions," *arXiv preprint arXiv:1405.3866*, 2014.

[32] V. Lebedev, Y. Ganin, M. Rakhuba, I. Oseledets, and V. Lempitsky, "Speeding-up convolutional neural networks using fine-tuned cp-decomposition," *arXiv preprint arXiv:1412.6553*, 2014.

[33] X. Yu, T. Liu, X. Wang, and D. Tao, "On compressing deep models by low rank and sparse decomposition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7370–7379.

[34] Y. Gao, Z. Zhang, H. Zhang, M. Zhao, Y. Yang, and M. Wang, "Dictionary pair-based data-free fast deep neural network compression," in *2021 IEEE International Conference on Data Mining (ICDM)*, 2021, pp. 121–130.

[35] J. Yim, D. Joo, J. Bae, and J. Kim, "A gift from knowledge distillation: Fast optimization, network minimization and transfer learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4133–4141.

[36] X. Cheng, Z. Rao, Y. Chen, and Q. Zhang, "Explaining knowledge distillation by quantifying the knowledge," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 12 925–12 935.

[37] R. G. Lopes, S. Fenu, and T. Starner, "Data-free knowledge distillation for deep neural networks," *arXiv preprint arXiv:1710.07535*, 2017.

[38] N. Wang, W. Zhou, Y. Song, C. Ma, and H. Li, "Real-time correlation tracking via joint model compression and transfer," *IEEE Transactions on Image Processing*, vol. 29, pp. 6123–6135, 2020.

[39] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[40] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[41] Y. Li, N. Wang, J. Shi, J. Liu, and X. Hou, "Revisiting batch normalization for practical domain adaptation," *arXiv preprint arXiv:1603.04779*, 2016.

[42] W. Wan, Y. Zhong, T. Li, and J. Chen, "Rethinking feature distribution for loss functions in image classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9117–9126.

[43] Y. Wang, W. Zhou, T. Jiang, X. Bai, and Y. Xu, "Intra-class feature variation distillation for semantic segmentation," in *European Conference on Computer Vision*. Springer, 2020, pp. 346–362.

[44] A. Gholami, S. Kim, Z. Dong, Z. Yao, M. W. Mahoney, and K. Keutzer, "A survey of quantization methods for efficient neural network inference," *arXiv preprint arXiv:2103.13630*, 2021.

[45] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.

[46] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[47] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[48] C. Guo, C. Li, J. Guo, C. C. Loy, J. Hou, S. Kwong, and R. Cong, "Zero-reference deep curve estimation for low-light image enhancement," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 1780–1789.