17th International Conference on Machine Vision Applications (MVA)
Fully Online, July 25–27, 2021.

Lossless AI: Toward Guaranteeing Consistency between Inferences Before and After Quantization via Knowledge Distillation

Tomoyuki Okuno, Yohei Nakata, Yasunori Ishii, Sotaro Tsukizawa
Panasonic Corporation
1006 Kadoma, Kadoma City, Osaka 571-8508, Japan
okuno.tomoyuki@jp.panasonic.com

Abstract

Deep learning model compression is necessary for real-time inference on edge devices, which have limited hardware resources. Conventional methods have only focused on suppressing degradation in terms of accuracy. Even if a compressed model has almost equivalent accuracy to its reference model, the inference results may change when we focus on individual samples or objects. Such a change is a crucial challenge for the quality assurance of embedded products because of unexpected behavior for specific applications on edge devices. Therefore, we propose a concept called "Lossless AI" to guarantee consistency between the inference results of reference and compressed models. In this paper, we propose a training method to align inference results between reference and quantized models by applying knowledge distillation that batch normalization statistics are frozen at moving average values from the middle of training. We evaluated the proposed method on several classification datasets and network architectures. In all cases, our method suppressed the inferred class mismatch between reference and quantized models whereas conventional quantization-aware training did not.

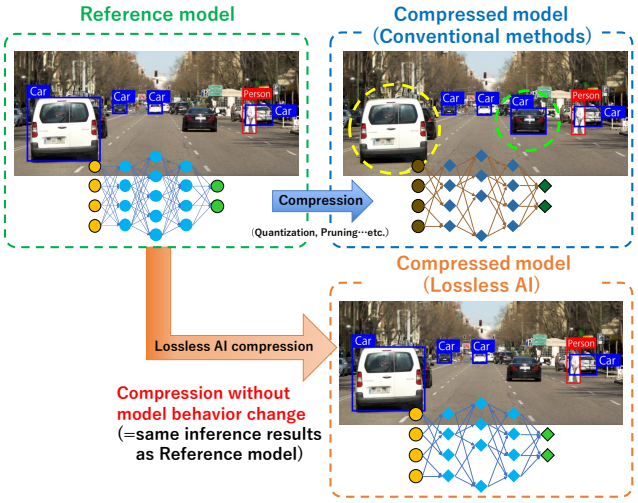

Figure 1. Conceptual examples of conventional compression methods and Lossless AI in car and person detection. Cars surrounded with yellow and green dashed circles are degraded and improved by model compression, respectively.

1 Introduction

Deep neural networks are widely used for recognition in practical applications, such as object detection, face recognition, speech recognition, and translation [1]. For applications that require real-time inference, it is more effective to execute inference processing on edge devices than cloud systems. Because edge devices typically have limited hardware resources, it is necessary to reduce the size of high computational cost models using compression methods, such as pruning and quantization [2, 3], to enable the implementation of models on edge devices.

Previous studies on model compression aimed to suppress accuracy degradation. Even if a compressed model has equivalent accuracy to its reference model, individual sample-by-sample inference results may be different. Figure 1 shows examples of car and person detection. In conventional methods, both reference and compressed models detect four cars, whereas their inference results for individual cars are not the same. As a result, the model behavior on an individual sample changes before and after model compression.

For the quality assurance of embedded products, the behavior differences caused by model compression require a large amount of reworking costs and effort. The model behavior change results in unexpected behavior for specific applications, and additional training and evaluation are required to overcome unexpected degradation. Even if such a modification overcomes degraded samples, it is likely to cause degradation on other samples because it is difficult to reproduce the same inference results before and after model compression. Therefore, the model behavior difference can be a crucial challenge for achieving the required performance on edge devices.

We propose a concept for embedded AI called **"Lossless AI"** to guarantee consistency between the

inference results of reference and compressed models. Consistency between inference results can be evaluated using various criteria for each task, such as the match rate in classification tasks and intersection over union between inferred bounding boxes in object detection tasks. Improving performance based on these criteria, Lossless AI can produce compressed models with inference results closer to those of their reference models. In this paper, we focus on classification tasks and quantization as a model compression method, and propose a knowledge distillation (KD) method that the batch normalization (BN) moving average statistics are frozen. Our contributions are as follows:

- We propose a concept of model compression called "Lossless AI" and a new criterion to guarantee consistency between inference results before and after compression.
- We propose a KD method that BN statistics are frozen at moving average values for inference result alignment.
- We demonstrate the effectiveness of the proposed KD method by evaluating several classification datasets and network architectures.

2 Related works

2.1 Quantization-aware training

Quantization-aware training (QAT) [3] is a method that quantizes a model during training. QAT models quantization and backpropagates gradient approximation with a straight-through estimator [4, 5]. Folding BN [6] is a technique used to stabilize QAT by setting BN statistics (i.e. the mean and variance) to input batch statistics at the beginning of the training phase and freezing them to moving average values after sufficient training. This technique improves accuracy better than post-training quantization, which statically quantizes weights and activations of pretrained models.

2.2 Knowledge distillation

Knowledge distillation (KD) [7] is another approach used to improve small model accuracy. KD is a technique that transfers the knowledge of a large teacher model to a small student model via a soft target in the loss function. KD has been applied to quantization in some studies to improve the quantized model accuracy [8, 9, 10].

2.3 Batch normalization

Batch normalization (BN) [11] is a general technique for improving accuracy in deep networks by normalizing an input tensor $\boldsymbol{x}_i \in \mathbb{R}^{N,H,W}$ (N, H, and W denote the batch size, height, and width, respectively)

Figure 2. Examples of mismatched patterns with different inference results between reference (Ref.) and quantization (Quant.) models: (a) degradation, (b) improvement, and (c) change between the incorrect classes. Ground truth (GT) labels are also shown. The check marks represent correct inferred classes.

for channel i using statistics (mean μ_i and variance σ_i^2). BN is expressed as follows using the $\beta_i, \gamma_i \in \mathbb{R}$ parameters for affine transfer:

$$\boldsymbol{y}_i = \gamma_i \frac{\boldsymbol{x}_i - \mu_i}{\sqrt{\sigma_i^2 + \varepsilon}} + \beta_i, \tag{1}$$

where ε is a small value used to achieve numerical stability. BN batch statistics $\mu_{i,\mathfrak{B}_t}$ and $\sigma^2_{i,\mathfrak{B}_t}$ calculated from the input batch are used for training at the t-th iteration. In the inference phase, BN moving average statistics $\mu_{i,\mathfrak{M}}$ and $\sigma^2_{i,\mathfrak{M}}$ are used. The moving average values are updated with momentum α using all the batch statistics during training.

Adaptive BN (AdaBN) [12] uses domain-specific BN statistics on the basis of the hypothesis that information related to labels and domains are stored in weights and BN statistics, respectively. To improve the inference accuracy in different domains from training, a similar framework was recently applied to several tasks, such as unsupervised domain adaptation [13], adversarial examples [14], and person re-identification [15].

3 Methodology: knowledge distillation with frozen BN statistics

In classification tasks, the match rate of classes indicates the degree of inference result alignment between reference and quantized models. Lossless AI aims to improve the match rate by reducing the number of mismatched samples whose inferred classes differ between the two models. As shown in Figure 2, mismatched samples are categorized into three patterns: degradation, improvement, or change between incorrect classes. In terms of the match rate, conventional methods such as QAT and KD are not sufficient because they can only contribute to accuracy improvement.

Figure 3 shows the proposed knowledge distillation framework. Based on the assumption of AdaBN, BN statistics memorizes domain information. For improving the match rate of classes, we employ fine-tuning only weight parameters which determine classification

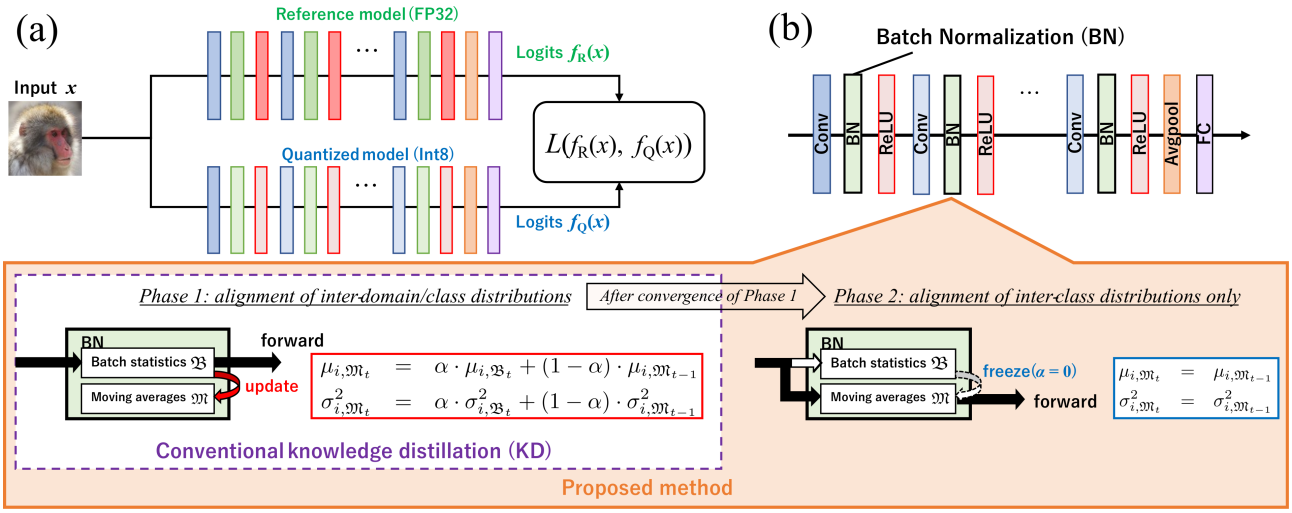

Figure 3. Illustrations of the proposed knowledge distillation framework for alignment between reference and quantized models: (a) overview and (b) training scheme of batch normalization layers.

output on condition that BN statistics are set to frozen moving average values in the same manner as inference mode. Our method of the fine-tuning is carried out after standard KD to optimize both weight and BN moving average parameters. As a result, our quantized model achieves comparable accuracy to the reference model and guarantee consistent model behavior between the two models.

Phase 1: alignment of inter-domain/class distributions
At the beginning of the training, BN batch statistics are used to calculate the forward pass to prevent collapse. BN moving average statistics are updated by all batch statistics in this phase.

Phase 2: alignment of inter-class distributions only
After convergence of Phase 1, BN moving average statistics are used for the forward pass. It is notable that BN moving average statistics should be frozen ($\alpha = 0$) to enable the alignment of inter-class distributions only.

We adopt the mean-square error loss (MSELoss) as the loss function L, and the logits of the two models, f_R and f_Q, as inputs to the loss function to enable the alignment of the inference results including scores.

4 Experiments

In this section, we describe the evaluation of the proposed method on several network architectures and datasets. We performed training and evaluation on five datasets that had different numbers of classes and images: ImageNet [16], Tiny ImageNet [17], CIFAR-100 [18], STL-10 [19], and CIFAR-10 [18].

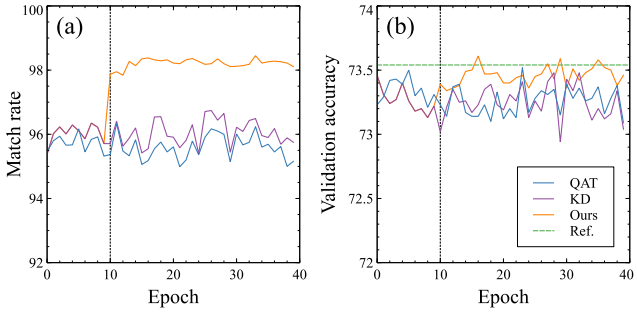

Figure 4. ResNet-50 training procedures of the (a) top-1 match rate and (b) top-1 accuracy in the validation of CIFAR-100. The blue, purple, and orange lines indicate the exp. 1 (QAT), 2 (KD), and 3 (Ours) results, respectively. The green dashed line indicates the validation accuracy of the reference model and the black dotted line indicates the starting epoch for freezing the BN statistics in exp. 3 (Ours).

We performed and compared the following three QATs for 40 epochs after floating point (FP) training for 200 epochs. We started the QATs whose initial weights were from reference models extracted from the epoch with the best accuracy in advance FP training.

exp. 1 QAT: we used the cross-entropy loss between the inferred class and ground truth label.

exp. 2 QAT+KD (KD): we used the MSELoss between the logits of reference and quantized models

exp. 3 QAT+KD (Ours): we used the same loss func-

Table 1. Evaluation results for ResNet-50 trained on CIFAR-100 (unit: %).

	Best epoch – accuracy				Best epoch – match rate		
	FP (Ref.)	QAT	KD	Ours	QAT	KD	Ours
Epochs	170	23	28	16	11	26	33
Top-1 accuracy	73.54	73.52	73.48	**73.61**	73.14	73.18	**73.48**
Top-1 match rate	—	95.79	96.65	**98.38**	96.34	96.74	**98.45**

Table 2. Evaluation results for ResNet-50 trained on several datasets (unit: %).

	Information			Best epoch – match rate		
Datasets	Classes	Images	FP acc.	QAT	KD	Ours
ImageNet	1000	14M	76.13	92.26	97.04	**97.35**
Tiny ImageNet	200	110K	59.45	92.26	94.85	**97.26**
CIFAR-100	100	60K	73.54	96.34	96.74	**98.45**
STL-10	10	13K	93.12	97.28	97.63	**99.19**
CIFAR-10	10	60K	93.12	98.96	99.31	**99.64**

Table 3. Evaluation results for several architectures trained on CIFAR-100 (unit: %).

Architectures	FP acc.	QAT	KD	Ours
ResNet-50	73.54	96.34	96.74	**98.45**
ResNet-20	68.32	95.03	96.34	**98.09**
MobileNet	64.84	94.46	95.10	**97.36**

tion as exp. 2, and BN statistics were frozen at moving average values updated for 10 epochs.

For all experiments, we trained the models in the Distiller [20] environment, which is Intel's open-source package based on a PyTorch implementation. We basically set the learning conditions and hyperparameters according to default settings of Distiller environment. The batch size and learning rate settings were different between FP training and QAT. We set the batch size to 128 for FP training and 64 for QAT, except for the STL-10 dataset. In the STL-10 experiments, we set the batch size to 64 for FP training and 32 for QAT. In FP training, we initially set the learning rate to 0.1 and used a step learning rate scheduler, which multiplied it by 0.1, 0.1, and 0.2 at 80, 120, and 160 epochs, respectively. In QAT, we initially set the learning rate to 0.0001 for exp. 1 (QAT) and 5×10^{-6} for exp. 2 (KD) and 3 (Ours), and used a step learning rate scheduler, which multiplied it by 0.1 at 20 and 30 epochs. We determined these hyperparameters empirically.

4.1 Evaluation and ablation study

Figure 4 shows the training procedures for top-1 match rate and top-1 validation accuracy for ResNet-50 [21] on the CIFAR-100 dataset. The match rate was gradually degraded in QAT and was almost constant in exp. 2 (KD). The proposed method significantly improved the match rate and maintained it at the improved level after the BN statistics were frozen. In addition, accuracy of our method was slightly better than that of the other two methods and comparable to that of the reference model. We summarize the evaluation results in Table 1. Regarding the best epochs for accuracy, the top-1 accuracy of all training was almost the same as that of the reference models, and our method suppressed the model behavior difference by 61% compared with QAT. Regarding the best epoch for the match rate, the proposed method suppressed the model behavior difference by 58% compared with QAT, whereas the accuracy degradation from the reference model was suppressed to 0.6%. The model behavior difference was calcurated from the mismatch rate of the QAT model and the match rate difference between QAT and the proposed methods.

Tables 2 and 3 shows the evaluation results for ResNet-50 trained on five datasets and three architectures (ResNet-20, ResNet-50, and MobileNet [22]) trained on the CIFAR-100 dataset, respectively. For all experiments, the proposed method achieved the best match rate in comparison with the best match rate epochs for each training and suppressed the model behavior difference by more than 50% compared with QAT. In the best case (ResNet-50 trained on STL-10), the match rate improved from 97.28% to 99.19% using the proposed method, and that is equivalent to suppressing the model behavior difference by 70.2%.

5 Conclusions

For the quality assurance of embedded products, we proposed a concept for embedded AI called "Lossless AI" to guarantee consistency between the inference results of reference and compressed models. In this paper, we proposed a KD method with BN moving average statistics frozen to align inter-class distributions effectively. We demonstrated the effectiveness of the proposed method by evaluating several network ar-

chitectures and classification datasets. The proposed method suppressed the behavior difference depending on the application within an acceptable level (around 50–70%). We will analyze the mismatched samples remained after our method and improve the method by making counter measures for them. Furthermore, we will adapt Lossless AI to other tasks and compression methods such as object detection and pruning.

References

[1] J. Gu, et al.: "Recent Advances in Convolutional Neural Networks," *Pattern Recognition*, vol.77, pp.354–377, 2018.

[2] Y. Cheng, et al.: "Model Compression and Acceleration for Deep Neural Networks: The Principles, Progress, and Challenges," *IEEE Signal Processing Magazine*, vol.35, no.1, pp.126–136, 2018.

[3] B. Jacob, et al.: "Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.2704–2713, 2018.

[4] G. Hinton: "Neural Networks for Machine Learning," *Coursera*, video lectures, 2012.

[5] Y. Bengio, et al.: "Estimating or Propagating Gradients through Stochastic Neurons for Conditional Computation," *arXiv preprint arXiv:1308.3432*, 2013.

[6] R. Krishnamoorthi: "Quantizing Deep Convolutional Networks for Efficient Inference: A Whitepaper," *arXiv preprint arXiv: 1806.08342*, 2018.

[7] G. Hinton, et al.: "Distilling the Knowledge in a Neural Network," *arXiv preprint arXiv: 1503.02531*, 2015.

[8] A. Mishra, et al.: "Apprentice: Using Knowledge Distillation Techniques to Improve Low-Precision Network Accuracy," *International Conference on Learning Representations*, 2018.

[9] A. Polino, et al.: "Model Compression via Distillation and Quantization," *International Conference on Learning Representations*, 2018.

[10] J. Kim, et al.: "QKD: Quantization-Aware Knowledge Distillation," *arXiv preprint arXiv: 1911.12491*, 2019.

[11] S. Ioffe, et al.: "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, pp.448–456, 2015.

[12] Y. Li, et al.: "Adaptive Batch Normalization for Practical Domain Adaptation," *Pattern Recognition*, vol.80, pp.109–117, 2018.

[13] W.-G. Chang, et al.: "Domain-Specific Batch Normalization for Unsupervised Domain Adaptation," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.7354–7362, 2019.

[14] C. Xie, et al.: "Adversarial Examples Improve Image Recognition," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.819–828, 2020.

[15] Z. Zhuang, et al.: "Rethinking the Distribution Gap of Person Re-Identification with Camera-Based Batch Normalization," *European Conference on Computer Vision*, pp.140–157, 2020.

[16] J. Deng, et al.: "ImageNet: A Large-Scale Hierarchical Image Database," *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp.248–255, 2009.

[17] Tiny ImageNet: `https://tiny-imagenet.herokuapp.com/`

[18] A. Krizhevsky, et al.: "Learning Multiple Layers of Features from Tiny Images," *CiteSeer*, 2009.

[19] A. Coates, et al.: "An Analysis of Single Layer Networks in Unsupervised Feature Learning," *AISTATS*, 2011.

[20] N. Zmora, et al.: "Neural Network Distiller: A Python Package for DNN Compression Research," *arXiv preprint arXiv:1910.12232*, 2019.

[21] K. He, et al.: "Deep Residual Learning for Image Recognition," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp.770–778, 2016.

[22] A. G. Howard, et al.: "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," *arXiv preprint arXiv:1704.04861*, 2017.