Cluster-Promoting Quantization with Bit-Drop for Minimizing Network Quantization Loss

Jung Hyun Lee[1]*, Jihun Yun[1]*, Sung Ju Hwang[1,2], Eunho Yang[1,2]
[1]Korea Advanced Institute of Science and Technology (KAIST), [2]AITRICS
{onliwad101, arcprime, sjhwang, eunhoy}@kaist.ac.kr

Abstract

Network quantization, which aims to reduce the bit-lengths of the network weights and activations, has emerged for their deployments to resource-limited devices. Although recent studies have successfully discretized a full-precision network, they still incur large quantization errors after training, thus giving rise to a significant performance gap between a full-precision network and its quantized counterpart. In this work, we propose a novel quantization method for neural networks, Cluster-Promoting Quantization (CPQ) *that finds the optimal quantization grids while naturally encouraging the underlying full-precision weights to gather around those quantization grids cohesively during training. This property of CPQ is thanks to our two main ingredients that enable differentiable quantization: i) the use of the categorical distribution designed by a specific probabilistic parametrization in the forward pass and ii) our proposed multi-class straight-through estimator (STE) in the backward pass. Since our second component, multi-class STE, is intrinsically biased, we additionally propose a new bit-drop technique,* DropBits, *that revises the standard dropout regularization to randomly drop bits instead of neurons. As a natural extension of DropBits, we further introduce the way of learning heterogeneous quantization levels to find proper bit-length for each layer by imposing an additional regularization on DropBits. We experimentally validate our method on various benchmark datasets and network architectures, and also support a new hypothesis for quantization: learning heterogeneous quantization levels outperforms the case using the same but fixed quantization levels from scratch.*

*Equal contribution

1. Introduction

Deep neural networks have achieved great success in various computer vision applications. However, the state-of-the-art neural network architectures including ResNet [12] generally require too much computation and memory to be deployed to resource-limited devices. Therefore, researchers have explored diverse approaches to compress them to reduce memory usage and computation cost.

Among them, network quantization aims to reduce the bit-width of network parameters while maintaining competitive performance of a full-precision counterpart. One of the simplest methods is to round a weight or an activation of a network x to $\widehat{x} = \alpha\lfloor \frac{x}{\alpha} + \frac{1}{2}\rfloor$ where α controls the grid interval size. However, this naïve approach incurs severe performance degradation mainly due to the *quantization loss*. Given that if the underlying full-precision weights x are clustered well around the optimal quantization grids, the performance difference between before and after the quantization can be marginal so that the performance of full-precision network can be preserved even with the quantized parameters. Hence, we focus on jointly finding the optimal quantization grids and clustering the underlying full-precision weights x around those quantization grids cohesively.

Some recent studies in fact have experimentally confirmed that their methods can partially give a clustering effect in the quantization process. VNQ [2] clusters the underlying full-precision weights x around quantization grids using multi-spike-and-slab prior, but it is restricted only to ternary precision. RQ [19] experimentally shows some clustering effects around several modes in low bit-width, but it does not equip any algorithm that explicitly encourages clustering around quantization grids. As a result, both methods incur a considerable performance gap between a full-precision network and its quantized counterpart.

In order to preserve the performance of a full-precision network in low bit-width, we propose the *Cluster-Promoting Quantization (CPQ)* that not only finds the optimal quantization grids but also encourages the underlying full-precision weights x to gather around those quantization grids cohesively in low bit-length regimes. Although CPQ does not have any explicit regularization or loss for clustering, the combination of the following two key components results in better clustering effect (and thus final performance) both theoretically and experimentally: i) choosing the mode of the categorical distribution parametrized by a particular probabilistic approach in the forward pass and ii) taking advantage of our multi-class straight-through estimator (STE) in the backward pass.

As our multi-class STE is biased like the original STE for the binary case [3], we present a novel bit-drop technique named *DropBits* to reduce the bias of the multi-class STE in CPQ. Motivated from Dropout [27], DropBits drops bits rather than neurons/filters to train low-bit neural networks under CPQ framework.

In addition, DropBits allows *heterogeneous quantization*, which learns different bit-width per parameter/channel/layer by dropping redundant bits. DropBits with learnable bit-drop rates adaptively finds out the optimal bit-width for each group of parameters, possibly further reducing the overall bits. In contrast to recent studies [30, 29] in heterogeneous quantization that exhibit almost all layers have *at least* 4-bit, up to 10-bit, our method yields much more resource-efficient low-bit neural networks with *at most* 4 bits for all layers.

With trainable bit-widths, we also articulate a *new hypothesis for quantization* where one can find the learned bit-width network (termed a 'quantized sub-network') which can perform better than the network with the same but fixed bit-widths from scratch.

Our contribution is threefold:

- We propose a new quantization method, **C**luster-**P**romoting **Q**uantization (CPQ) that not only finds the optimal quantization grids but also encourages the underlying full-precision weights to congregate around those quantization grids cohesively in low bit-width regimes by the combination of a particular probabilistic parametrization for discretization and our multi-class straight-through estimator. We further present a novel bit-drop technique coined **DropBits** to reduce the bias of the multi-class straight-through estimator in CPQ.

- Extending DropBits technique, we propose a more resource-efficient heterogeneous quantization algorithm to curtail redundant bit-widths across groups of weights and/or activations (e.g. across layers) and verify that our method is able to find out 'quantized sub-networks'.

- We conduct extensive experiments on several benchmark datasets to demonstrate the effectiveness of our method. We accomplish new **state-of-the-art** results for ResNet-18 and MobileNetV2 on the ImageNet dataset when *all* layers are uniformly quantized.

2. Related Work

BinaryConnect [6] first attempted to binarize weights to ± 1 by employing deterministic or stochastic operation. To obtain better performance, various studies [23, 17, 2, 25] have been conducted in binarization and ternarization. Although these works effectively decrease the model size and raise the accuracy, they are limited to quantizing weights with activations remaining in full-precision. To take full advantage of quantization at run-time, it is necessary to quantize activations as well.

Researchers have recently focused more on simultaneously quantizing both weights and activations [34, 31, 4, 32, 11, 15, 8]. XNOR-Net [23] exploits the efficiency of XNOR and bit-counting operations. QIL [15] also quantizes weights and activations by introducing parametrized learnable quantizers that can be trained jointly with weight parameters. [8] recently presented a simple technique to approximate the gradients with respect to the grid interval size to improve QIL. Nevertheless, these methods do not quantize the first or last layer, which leaves a room to improve power-efficiency.

For ease of deployment in practice, it is inevitable to quantize weights and activations of all layers, which is the most challenging. [2] proposed multi-spike-and-slab prior to allow multiple modes at quantization grids, but it is limited to ternary precision. [19] proposed to use the Gumbel-Softmax trick [14, 21], but it does not cluster weights around quantization grids well. [13] presented efficient fixed-point implementations by formulating the grid interval size to the power of two, but they quantized the first and last layer to at least 8-bit. [33] proposed to quantize the grid interval size and network parameters in batch normalization for the deployment of quantized models on low-bit integer hardware, but it requires a specific accelerator only for this approach.

As another line of work, [10] proposed a heterogeneous binarization given pre-defined bit-distribution. HAWQ [7] determines the bit-width for each block heuristically based on the top eigenvalue of Hessian. Unfortunately, both of them do not learn optimal bit-widths for heterogeneity. Toward this, [30] and [29] proposed a layer-wise heterogeneous quantization by exploiting reinforcement learning and learning dynamic range of quantizers, respectively. However, their results exhibit that almost all layers have up to 10-bit (at least 4-bit), which would be suboptimal. [18] presented a channel-wise heterogeneous quantization by exploiting hierarchical reinforcement learning, but channel-wise precision limits the structure of accelerators, thereby restricting the applicability of the model.

3. Cluster-Promoting Quantization

In this section, we first summarize the notations used in this paper and then present an overview of our method.

The variable x denotes a weight or an activation of a full-precision network and $\widehat{x}$ indicates the quantized value of x. Here, we consider the following quantization grids for x: For a weight x, $\widehat{\mathcal{G}} \coloneqq [g_0, \dots, g_{2^b-1}] = \alpha[-2^{b-1}, \dots, 0, \dots, 2^{b-1}-1]$ where b is the given bit-width and $\alpha > 0$ is a learnable parameter that controls the interval of quantization grids. For an activation x, quantization grids in $\widehat{\mathcal{G}}$ start from zero since the output of ReLU is always non-negative. Lastly, $[n]$ denotes the set $\{0, 1, \cdots, n-1\}$ for a positive integer n.

Our main goal is to design a quantization algorithm that both finds the optimal α and clusters the underlying full-precision weights x around quantization grids $\widehat{\mathcal{G}}$ cohesively in low bit-width regimes. As a neural network is over-parametrized, there may exist a parameter such that the underlying full-precision weights x crowd around some discrete values without performance degradation. Toward this, we propose the Cluster-Promoting Quantization (CPQ) that not only finds the optimal α but also helps the underlying full-precision weights x congregate around quantization grids $\widehat{\mathcal{G}}$ cohesively. The proposed CPQ consists of two components: (i) a certain probabilistic parametrization for discretization (Section 3.1) and (ii) our multi-class STE (Section 3.2). Surprisingly, CPQ does not require any penalty or loss for clustering thanks to the combination of these two components as shown in Proposition 1 introduced in Section 3.2.

3.1. Probabilistic Parametrization for Quantization

To permit gradient-based optimization, we assume that x is perturbed by noise ϵ as $\widetilde{x} = x + \epsilon$. The variable ϵ represents random noise for variational optimization [28] that can follow any distribution with zero mean and standard deviation σ. Here, let ϵ follow the logistic distribution $p(\epsilon) = \text{Logistic}(0, \sigma)$ so that $p(\widetilde{x})$ is governed by $\text{Logistic}(x, \sigma)$. Under such $p(\widetilde{x})$, the *unnormalized* probability of $\widetilde{x}$ being quantized to each quantization grid g_i can be easily computed in a closed form as below:

$$\pi_i = \text{Sigmoid}\Big(\frac{g_i + \frac{\alpha}{2} - x}{\sigma}\Big) - \text{Sigmoid}\Big(\frac{g_i - \frac{\alpha}{2} - x}{\sigma}\Big), \tag{1}$$

where the cumulative distribution function of the logistic distribution is a sigmoid function. Note that under (1), x, α, and σ are trainable parameters. Given unnormalized categorical probabilities $\boldsymbol{\pi} = \{\pi_i\}_{i=0}^{2^b-1}$ for quantization grids $\widehat{\mathcal{G}} = \{g_i\}_{i=0}^{2^b-1}$, depending on how to design where x is quantized according to $\boldsymbol{\pi}$, an algorithmic detail is determined. For instance, [19] employed the Gumbel-Softmax trick [14, 21] based on $\boldsymbol{\pi}$. In this paper, we adopt such a probabilistic parametrization (1) from [19] for our method.

3.2. Multi-Class Straight-Through Estimator

Given a probabilistic model for quantization like (1), we define a new straight-through estimator (STE) as follows:

$$\textbf{Forward: } y = \texttt{one\_hot}[\operatorname*{argmax}_i \pi_i] \tag{2}$$

$$\textbf{Backward: } \frac{\partial \mathcal{L}}{\partial \pi_{i_{\max}}} = \frac{\partial \mathcal{L}}{\partial y_{i_{\max}}} \text{ and } \frac{\partial \mathcal{L}}{\partial \pi_i} = 0 \text{ for } \forall\, i \neq i_{\max}, \tag{3}$$

where y_i is the i-th entry of the one-hot vector y and $\mathcal{L}$ is the cross entropy between the true label and the prediction made by a quantized neural network by the forward pass (2). We dub (2) and (3) the '*multi-class STE*'. That is, in the forward pass, we directly select the mode (or the most likely grid) of the categorical distribution parametrized by the probabilistic model. On the other hand, in the backward pass, it is required to handle the non-differentiable argmax operator in computing $i_{\max}$. To allow gradient-based optimization, we enable backpropagation through a non-differentiable *categorical* sample by backpropagating only through the path corresponding to $i_{\max}$.

Note that the proposed multi-class STE can be thought of as a natural extension of binary case [3], but this work is the first in-depth study on multi-class STE in terms of network quantization. Although [14] proposed slightly different heuristic estimator, ST GS, that uses the Gumbel-softmax trick with another straight-through estimator to bypass the non-differentiability of discrete random variables, ST GS does not have any justification in the context of network quantization. On the other hand, our multi-class STE theoretically and empirically demonstrates the superiority of clustering by the following proposition, when π_i is computed as in (1), even without any regularization or loss for clustering.

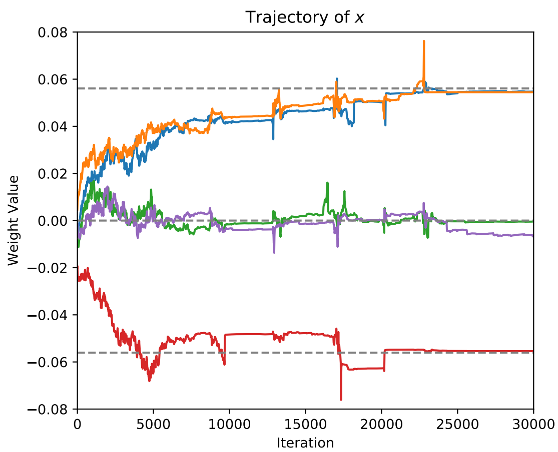


Figure 1. Trajectories of five random weights in the second layer when training LeNet-5 on MNIST in 3-bit. The x-axis indicates the number of training iterations, and the y-axis represents the value of weight. The horizontal dashed lines (gray) denote quantization grids after training.

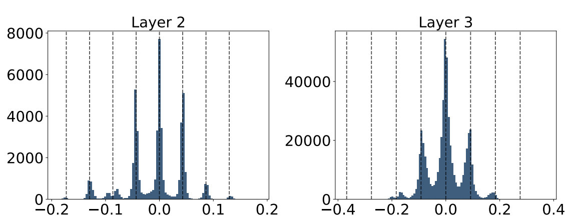


Figure 2. Weight distributions for 3-bit quantized LeNet-5 by our method, CPQ. The x-axis and y-axis indicate weight values and their frequencies, respectively. The vertical dashed lines denote quantization grids.

Proposition 1. *Let $\mathcal{L}$ be a loss function calculated from a quantized neural network using* (1) *and* (2)*. Under the assumption that* $|\frac{\partial \mathcal{L}}{\partial y_{i_{max}}}|$ *is bounded, the gradient of $\mathcal{L}$ with respect to full precision variable x from* (3)*,* $\frac{\partial \mathcal{L}}{\partial x}$*, converges to zero as a weight x approaches its nearest quantization grid $g_{i_{max}}$.*

By Proposition 1, once x is trained to become near $g_{i_{\max}}$, x can be kept to stay around $g_{i_{\max}}$ as seen in Figure 1, thus making it possible to cluster the underlying full-precision weights around quantization grids cohesively as seen in Figure 2. As our multi-class STE with (1) can be qualitatively distinct from other unjustified estimators from this perspective, we call the combination of (1) and the multi-class STE 'Cluster-Promoting Quantization (CPQ)'. The overall procedure of CPQ is described in Algorithm 1.

One might wonder that the almost zero gradient near quantization grids may make a network untrainable, which would not be a gradient-based learning. Although $\frac{\partial \mathcal{L}}{\partial x}$ is almost zero when x is close to $g_{i_{\max}}$, α is still trained to find the better grid points. After α is updated, if the gap between x and α is widened , then x is trained accordingly. Hence, a network will continue to train until it finds the optimal α. Such a training procedure is illustrated in Figure 3.

In addition to Proposition 1, our multi-class STE has another strength: it makes the variance of gradients become indeed zero, which has to do with what [19] highlighted to train a network with low bit-widths successfully. As our multi-class STE always chooses the mode of the categorical distribution paramterized by a probabilistic model (i.e., there is no randomness in the forward pass (2)) and the gradient of $\mathcal{L}$ with respect to the individual categorical probabilities is exactly *zero* everywhere except for the coordinate corresponding to the mode in the backward pass (3), the variance of our gradient estimator becomes zero.

4. DropBits and Its Extension to Heterogeneous Quantization

We propose a novel bit-drop technique named *DropBits* to reduce the bias of the multi-class STE (Section 4.1). We also impose an extra regularization on DropBits to permit heterogeneous quantization (Section 4.2) and put forward a new hypothesis for quantization (Section 4.3).

4.1. DropBits

Although our multi-class STE enjoys zero variance of gradients, it is biased to the mode as the binary one in [3]. To reduce the bias of STE, [5] propose the slope annealing trick, but this strategy is only applicable to the binary case. To address this limitation, we propose a novel bit-drop method, *DropBits*, to decrease the bias of our multi-class STE. Inspired by dropping neurons in Dropout [27], we *drop an arbitrary number of grid points at random every iteration*, where in effect the probability of being quantized to dropped grid points becomes zero.

However, the design policy that each grid point has its own binary mask would make the number of masks increase exponentially with bit-width. Taking into consideration appropriate noise levels with a less aggressive design, the following two examples are available: **(a)** endpoints in the grids share the same binary mask, and **(b)** the grid points in the same bit-level share the same binary mask (see Figure 4). Hereafter, we consider **(b)** bitwise-sharing masks for groups of grid points, unless otherwise specified.

Now, we introduce how to formulate binary masks. Unlike Dropout implementation through dividing activations

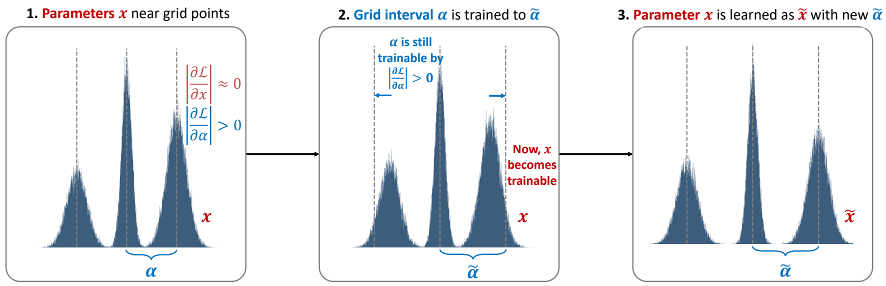
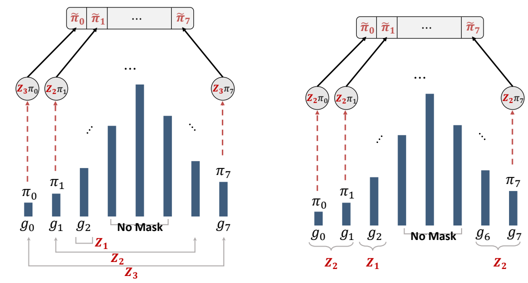


Figure 3. Training procedure when weights are close to quantization grids.

Algorithm 1 Cluster-Promoting Quantization (CPQ)

Input: Training data $\mathcal{D}$, network parameters $\{W_l, b_l\}_{l=1}^{L}$, layer-wise grid interval parameters and the standard deviations of a logistic distribution in the l-th layer $\{\alpha_l, \sigma_l\}_{l=1}^{L}$.
Output: A low bit-width model with quantized network parameters $\{\widehat{W}_l, \widehat{b}_l\}_{l=1}^{L}$ after deployment procedure.
Initialize: Bit-width b and parameters $\{W_l, b_l, \alpha_l, \sigma_l\}_{l=1}^{L}$. Initialize layer-wise grid $\widehat{\mathcal{G}}_l := [g_{l,0}, g_{l,1}, \cdots, g_{l,2^b-1}] = \alpha_l[-2^{b-1}, \cdots, 2^{b-1}-1]$ for $l \in \{1, \cdots, L\}$.
procedure TRAINING
  // Forward pass
  for $l = 1, \cdots, L$ **do**
    $x \leftarrow$ Each entry of W_l or b_l
    $I_l = \widehat{\mathcal{G}}_l - \alpha/2$ ▷ Shift the grid by $-\alpha/2$
    $F = \text{Sigmoid}\left(\frac{I_l - x}{\sigma_l}\right)$ ▷ Compute CDFs
    $\pi_i = F[i+1] - F[i]$ for $i \in [2^b - 1]$ ▷ Eq. (1)
    $y = \texttt{one\_hot}[\text{argmax}_i\ \pi_i]$ ▷ Eq. (2)
    $\widehat{x} = y \odot \widehat{\mathcal{G}}_l$ ▷ Quantization
    Activation can be quantized in the same way
  end for
  // Backward pass
  for $l = L, \cdots, 1$ **do**
    Compute gradients $\left(\frac{\partial\mathcal{L}}{\partial W_l}, \frac{\partial\mathcal{L}}{\partial b_l}, \frac{\partial\mathcal{L}}{\partial \alpha_l}, \frac{\partial\mathcal{L}}{\partial \sigma_l}\right)$ ▷ Eq. (3)
    Update parameters $(W_l, b_l, \alpha_l, \sigma_l)$
  end for
end procedure
procedure DEPLOYMENT
  for $l = 1, \cdots, L$ **do**
    $\widehat{W}_l = \min(\max(\alpha_l \cdot \text{Round}(W_l/\alpha_l), g_{l,0}), g_{l,2^b-1})$
    $\widehat{b}_l = \min(\max(\alpha_l \cdot \text{Round}(b_l/\alpha_l), g_{l,0}), g_{l,2^b-1})$
  end for
end procedure

by $1-p$ (here, p is a dropout probability), we employ an explicit binary mask Z whose probability Π can be optimized jointly with model parameters. The Bernoulli random variable being non-differentiable, we relax a binary mask via the *hard concrete* distribution [20]. While the binary concrete distribution [21] has its support $(0,1)$, the hard concrete distribution stretches it slightly at both ends, thus concentrating more mass on exact 0 and 1. Assuming disjoint masks, we describe the construction of a binary mask Z_k for the k-th bit-level using the hard concrete distribution as follows.



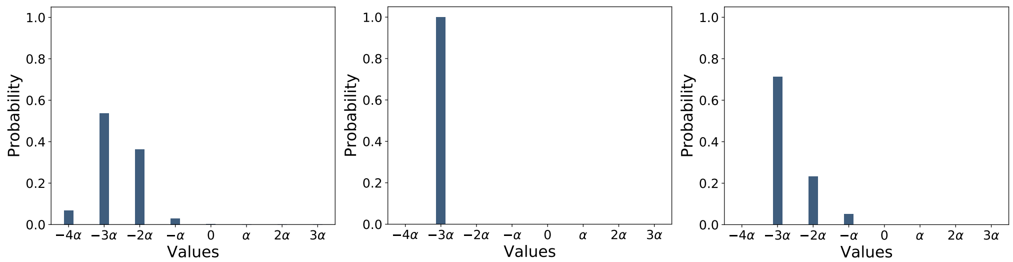
(a) Endpoints-sharing mask (b) Bitwise-sharing mask

Figure 4. Designs of two masks for 3-bit

$$U_k \sim \text{Uniform}(0,1), \tag{4}$$
$$S_k = \text{Sigmoid}\Big(\big(\log U_k - \log(1-U_k) + \log\frac{\Pi_k}{1-\Pi_k}\big)/\tau'\Big)$$
$$\bar{S}_k = S_k(\zeta - \gamma) + \gamma \quad \text{and} \quad Z_k = \min(\max(\bar{S}_k, 0), 1)$$

where τ' is a temperature for the hard concrete distributions with $\gamma < 0$ and $\zeta > 0$ reflecting stretching level. For $i = 2^{b-1}-1, 2^{b-1}$ and $2^{b-1}+1$, we do not sample from the above procedure but fix $Z = 1$ to prohibit all the binary masks from becoming zero (see 'No Mask' in Figure 4).

With the value of each mask generated from the above procedure, the probability of being quantized to any grid point is re-calculated by multiplying π_i's by their corresponding binary masks Z_k's (e.g. $\widetilde{\pi}_0 = Z_2 \cdot \pi_0$ in Figure 4 (b)) and then normalizing them (to sum to 1). As seen in Figure 5, the sampling distribution of CPQ is biased to the mode, -3α. For an appropriate value of Π_k, the sampling distribution of CPQ + DropBits can more resemble the original categorical distribution than that of CPQ by adjusting π_i's to $\widetilde{\pi}_i$'s based on Z_k's via DropBits, which means that DropBits is able to reduce the bias of the multi-class straight-through estimator in CPQ effectively. Not only that, DropBits does not re-



(a) Categorical distribution (b) CPQ (c) CPQ + DropBits

Figure 5. The illustration of the effect of DropBits on CPQ. For a certain weight, **(a)** the categorical distribution indicates $\pi_i/\Sigma_{j=0}^{7}\pi_j$ for each grid $(i = 0, \cdots, 7)$, **(b)** the distribution of CPQ is a sampling distribution after taking the argmax of π_i, and **(c)** the distribution of CPQ + DropBits is a sampling distribution after taking the argmax of $\widetilde{\pi}_i$. Here, Π_k's are initialized to 0.7 for clear understanding.

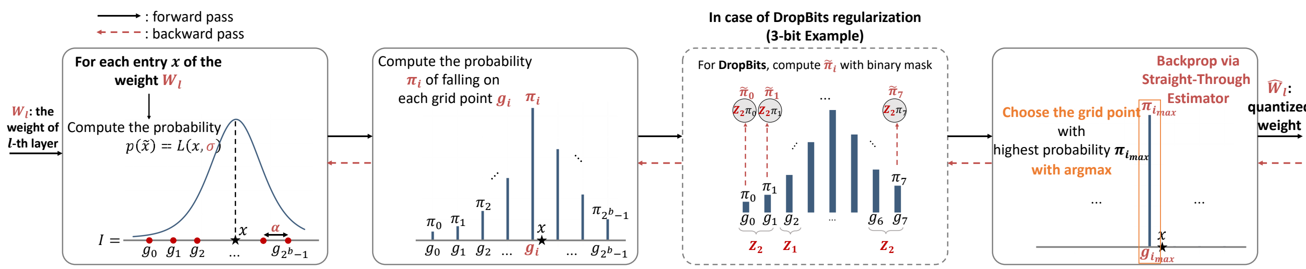


Figure 6. Illustration of Cluster-Promoting Quantization (CPQ) framework with DropBits technique.

quire any hand-crafted scheduling at all due to the learnable characteristic of Π_k, whereas such scheduling is vital for Gumbel-Softmax [14, 21] and slope annealing trick [5].

Although quantization grids for weights are symmetric with respect to zero, those for activations start from zero, which makes it difficult to exploit symmetrically-designed DropBits for activations. Therefore, DropBits is applied only for weights in our experiments. Assuming that Z_k's are shared across all weights of each layer, the overall procedure is described in Figure 6. The overall algorithm of CPQ + DropBits is deferred to Appendix due to space limit.

4.2. Learning Bit-width towards Resource-Efficiency

As noted in Section 1 and 2, recent studies on heterogeneous quantization use at least 4-bit, up to 10-bit, which leaves much room for saving energy and memory. Towards more resource-efficient method, we introduce an additional regularization on DropBits to drop redundant bits.

As the mask design in Figure 4-(b) reflects the actual bit-level and the probability of each binary mask in DropBits is learnable, we can penalize the case where we use higher bit-levels via a sparsity encouraging regularizer like ℓ_1. As [20] proposed a relaxed ℓ_0 regularization using the hard concrete binary mask, we adopt this continuous version of ℓ_0 as a sparsity inducing regularizer. Following (4), we define the smoothed ℓ_0-norm as $\mathcal{R}(Z;\Pi) = \mathrm{Sigmoid}(\log\frac{\Pi}{1-\Pi} - \tau' \log\frac{-\gamma}{\zeta})$. One caveat here is that we do not have to regularize masks for low bit-level if a higher bit-level is still alive (in this case such a high bit-level is still necessary for quantization). We thus design a regularization in such a specific way as only to permit the probability of a binary mask at the current highest bit-level to approach zero. More concretely, for bit-level binary masks $\{Z_k\}_{k=1}^{b-1}$ as in Figure 4-(b) and the corresponding probabilities $\{\Pi_k\}_{k=1}^{b-1}$, our regularization term to learn the bit-width is

$$\mathcal{R}\big(\{Z_k\}_{k=1}^{b-1}, \{\Pi_k\}_{k=1}^{b-1}\big) = \sum_{k=1}^{b-1} \mathbb{I}(Z_k > 0)\Big(\prod_{j=k+1}^{b-1} \mathbb{I}(Z_j = 0)\Big)\mathcal{R}(Z_k;\Pi_k). \quad (5)$$

Note that $\{Z_k\}_{k=1}^{b-1}$ is assigned to each group (e.g. all weights or activations in a layer or channel for instance). Hence, every weight in a group shares the same sparsity pattern (and bit-width as a result), and learned bit-widths across groups are allowed to be heterogeneous.

Assuming the l-th *layer* shares binary masks $\boldsymbol{Z}^l := \{Z_k^l\}_{k=1}^{b-1}$ associated with probabilities $\boldsymbol{\Pi}^l := \{\Pi_k^l\}_{k=1}^{b-1}$, our final objective function for a L-layer neural network becomes $\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\alpha}, \boldsymbol{\sigma}, \boldsymbol{Z}, \boldsymbol{\Pi}) + \lambda \sum_{l=1}^{L} \mathcal{R}(\boldsymbol{Z}^l, \boldsymbol{\Pi}^l)$, where $\boldsymbol{\alpha} = \{\alpha_l\}_{l=1}^{L}$ and $\boldsymbol{\sigma} = \{\sigma_l\}_{l=1}^{L}$ represent the layer-wise grid interval parameters and standard deviations of logis-

Table 1. Test error (%) for LeNet-5 on MNIST and VGG-7 on CIFAR-10. “Ann.” stands for annealing the temperature of the Gumbel-Softmax trick in RQ.

Dataset	# Bits W./A.	RQ	RQ + Ann.[2]	CPQ	CPQ + DropBits
MNIST	4/4	0.58	0.62	0.59	**0.53**
	3/3	0.69	0.74	0.67	**0.58**
	2/2	0.76	–	0.72	**0.63**
CIFAR-10	4/4	8.43	8.47	7.15	**6.85**
	3/3	9.56	10.78	7.08	**6.94**
	2/2	11.75	–	7.68	**7.51**

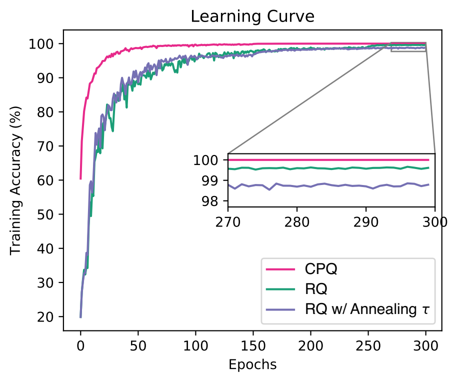


Figure 7. Learning curves of VGG-7 quantized by RQ, RQ with annealing τ, and CPQ in 3-bit.

tic distributions, $\boldsymbol{Z} = \{\boldsymbol{Z}^l\}_{l=1}^L$, $\boldsymbol{\Pi} = \{\boldsymbol{\Pi}^l\}_{l=1}^L$, and λ is a regularization parameter. In inference phase, we just drop unnecessary bits based on the values of $\boldsymbol{\Pi}$.

4.3. New Hypothesis for Quantization

[9] articulated the ‘lottery ticket hypothesis’, stating that one can find some sparse sub-networks, ‘winning tickets’, from randomly-initialized, dense neural networks that are easier to train than sparse networks resulting from pruning. In this section, we define a new hypothesis for quantization with slightly different (opposite in some sense) perspective from the original one.

Notation. $a \succ_{\text{bit}} b$ and $a =_{\text{bit}} b$ denote that a has strictly higher bit-width than b for at least one of all groups (e.g. channels or layers), and a has the same bit-precision as b across all groups, respectively.

Definition. For a network $f(x;\theta)$ with randomly-initialized parameters θ, let $f(x;\theta')$ be a quantized network from $f(x;\theta)$ such that $\theta \succ_{\text{bit}} \theta'$. If the accuracy of $f(x;\theta')$ is higher than that of $f(x;\theta'')$ where $f(x;\theta'')$ is trained from scratch with fixed bit-widths such that $\theta' =_{\text{bit}} \theta''$, $f(x;\theta')$ is referred to as a *quantized sub-network* of $f(x;\theta)$.

[2]We cannot reproduce the results of RQ in 2-bit, so we experiment only on 3-bit and 4-bit RQ

Table 2. Top-1/Top-5 error (%) with ResNet-18 and MobileNetV2 on the ImageNet dataset.

Method	# Bits W./A.	ResNet-18 Top-1/Top-5	MobileNetV2 Top-1/Top-5
Full-precision	32/32	30.24/10.92	28.12/9.71
RQ [19]	4/4	38.48/16.01	−/−
RQ ST [19]	4/4	37.54 / 15.22	– / –
QIL[3] [15]	4/4	31.05/11.23	32.77/12.51
LLSQF [33]	4/4	30.60/11.28	32.63/12.01
	3/3	33.33/12.58	−/−
TQT [13, 29]	4/4	30.49/−	32.21/−
CPQ + DropBits	4/4	**30.37/10.96**	**30.83/11.26**
	3/3	**32.79/12.57**	**35.71/14.36**

This hypothesis implies learning bit-width would be superior to pre-defined bit-width. To the best of our knowledge, our study is the first attempt to delve into this hypothesis.

5. Experiments

As popular deep learning libraries such as TensorFlow [1] and PyTorch from v1.3 [22] already provide their own 8-bit quantization functionalities, we focus on low bit-width regimes (2$\sim$4-bit). In contrast to some other quantization papers, our method uniformly quantizes weights and activations of *all* layers containing both the *first* and *last* layers. We first show that CPQ and DropBits have its own contribution, none of which is negligible. Then, we evaluate CPQ + DropBits on a large-scale dataset with deep networks. Finally, we demonstrate our heterogeneous quantization method yields promising results even if all layers have at most 4-bit and validate a new hypothesis for quantization in Section 4.3.

5.1. Ablation Studies

To validate the efficacy of CPQ and DropBits, we successively apply each piece of our method to LeNet-5 [16] on MNIST and VGG-7 [26] on CIFAR-10. Table 1 shows that CPQ outperforms RQ in most cases. One might wonder that the performance of RQ can be improved by an annealing schedule of the temperature in the Gumbel-Softmax trick. Unfortunately, RQ with an annealing schedule suffers from high variance of gradients due to low temperatures at the end of training as shown in Figure 7, thus giving rise to worse performance than RQ as shown in Table 1. Finally, it can be clearly identified that DropBits consistently improves CPQ by decreasing the bias of our multi-class STE in CPQ.

[3]Our own implementation with all layers quantized by using pretrained models available from PyTorch

Table 3. Test error (%) for quantized sub-networks using LeNet-5 on MNIST, VGG-7 on CIFAR-10,and ResNet-18 on ImageNet. Here, an underline means the learned bit-width and "T" stands for ternary precision.

Model	Initial # Bits W/A	Test Error	Trained W. Bits per layer	Test Error (Fixed)	Test Error (Reg.)
LeNet-5	4/4	0.53	$4/4/\underline{3}/4$	0.55	**0.52**
	3/3	0.58	$3/\underline{2}/3/3$	0.65	**0.55**
	2/2	0.63	$2/2/2/\underline{\mathrm{T}}$	0.68	**0.59**
VGG-7	4/4	6.77	$4/4/4/4/4/\underline{3}/\underline{3}/4$	6.74	**6.65**
	3/3	6.82	$3/3/3/3/3/\underline{2}/3/3$	6.81	**6.77**
	2/2	7.49	$2/2/2/2/2/2/2/\underline{\mathrm{T}}$	7.43	**7.36**
ResNet-18	4/4	33.20	$4/\underline{3}/\underline{3}/\underline{3}/\underline{3}/\underline{3}/\underline{3}/\underline{3}/\underline{3}/\underline{3}/\underline{3}/\underline{3}/\underline{3}/\underline{3}/\underline{3}/4/4/\underline{3}/4/4/4$	34.58	**34.30**
	3/3	37.80	$3/3/\underline{2}/3/\underline{2}/3/3/3/3/3/3/3/\underline{2}/3/3/3/3/3/3/3/3$	41.01	**40.30**

5.2. ResNet-18 and MobileNetV2 on ImageNet

To verify the effectiveness of our algorithm on the ImageNet dataset, we quantize the ResNet-18 [12] and MobileNetV2 [24] architectures initialized with each pre-trained full-precision network available from the official PyTorch repository. In Table 2, our method is only compared to the state-of-the-art algorithms that quantize both weights and activations of *all* layers for fair comparisons. The extensive comparison against recent works that remain the first or last layer in the full-precision is given in Appendix.

Table 2 illustrates how much better our model performs than the latest quantization methods. In ResNet-18, CPQ + DropBits outdoes RQ, QIL, LLSQF, and TQT, even achieving the top-1 and top-5 errors in 4-bit nearly close to those of the full-precision network. In MobileNetV2, CPQ + DropBits with 4-bit surpasses all existing studies by more than one percentage point. Moreover, we quantize MobileNetV2 to 3-bit, obtaining competitive performance, which is remarkable due to the fact that none of previous works successfully quantizes MobileNetV2 to 3-bit.

5.3. Finding Quantized Sub-networks

In this experiment, we validate a *new hypothesis for quantization* by training the probabilities of binary masks using the regularizer in Section 4.2 to learn the bit-width of each layer. For brevity, only weights are heterogeneously quantized, and the bit-width for activations remains fixed.

In Table 3, the fourth column represents the bit-width per layer learned by our regularizer, and the fifth and last columns indicate the test error when fixing the bit-width of each layer same as trained bit-widths (fourth column) from scratch and when using our regularization approach, respectively. Table 3 shows that a learned structure by our heterogeneous quantization method (last column) is superior to the fixed structure with learned bit-widths from scratch (fifth column) for all cases. It might be doubtful whether our regularizer is able to recognize which layer is really redundant or not. This may be indirectly substantiated by the observation that the fixed structure with trained bit-widths from scratch (fifth column) outperforms the uniform quantization (third column) on CIFAR-10. More experiments on different values of the regularization parameter λ are deferred to Appendix.

6. Conclusion

We proposed *Cluster-Promoting Quantization (CPQ)*, which not only finds the optimal quantization grids but also encourages the underlying full-precision weights to cluster around those quantization grids cohesively in low bit-width regimes. To reduce the bias of the multi-class STE in CPQ, we also proposed a novel bit-drop technique, *DropBits*. We showed that both CPQ and DropBits possess its own value, thereby leading CPQ + DropBits to achieve the state-of-the-art performance for ResNet-18 and MobileNetV2 on ImageNet. Furthermore, we took one step forward to consider heterogeneous quantization by simply penalizing binary masks in DropBits, which enables us to find out quantized sub-networks. As future work, we plan to extend our heterogeneous quantization method to activations and its application to other quantizers.

Acknowledgements

This work was supported by the National Research Foundation of Korea (NRF) grants (2018R1A5A1059921, 2019R1C1C1009192) and Institute of Information & Communications Technology Planning & Evaluation (IITP) grants (No. 2017-0-01779, A machine learning and statistical inference framework for explainable artificial intelligence, and No.2019-0-00075, Artificial Intelligence Graduate School Program(KAIST)) funded by the Korea government(MSIT).

References

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: A system for large-scale machine learning. In *USENIX Symposium on Operating Systems Design and Implementation*, pages 265–283, 2016. 7

[2] Jan Achterhold, Jan Mathias Koehler, Anke Schmeink, and Tim Genewein. Variational network quantization. In *International Conference on Learning Representations*, 2018. 1, 2

[3] Yoshua Bengio, Nicholas Leonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. 2, 3, 4

[4] Jungwook Choi, Zhuo Wang, Swagath Venkataramani, Pierce I-Jen Chuang, Vijayalakshmi Srinivasan, and Kailash Gopalakrishnan. PACT: parameterized clipping activation for quantized neural networks. *CoRR*, abs/1805.06085, 2018. 2

[5] Junyoung Chung, Sungjin Ahn, and Yoshua Bengio. Hierarchical multiscale recurrent neural networks. In *International Conference on Learning Representations*, 2016. 4, 6

[6] Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3123–3131. Curran Associates, Inc., 2015. 2

[7] Zhen Dong, Zhewei Yao, Amir Gholami, Michael W. Mahoney, and Kurt Keutzer. Hawq: Hessian aware quantization of neural networks with mixed-precision. In *The IEEE International Conference on Computer Vision (ICCV)*, 2019. 2

[8] Steven K. Esser, Jeffrey L. McKinstry, Deepika Bablani, Rathinakumar Appuswamy, and Dharmendra S. Modha. Learned step size quantization. In *International Conference on Learning Representations*, 2020. 2

[9] Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019. 7

[10] Joshua Fromm, Shwetak Patel, and Matthai Philipose. Heterogeneous bitwidth binarization in convolutional neural networks. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 4006–4015. Curran Associates, Inc., 2018. 2

[11] Ruihao Gong, Xianglong Liu, Shenghu Jiang, Tianxiang Li, Peng Hu, Jiazhen Lin, Fengwei Yu, and Junjie Yan. Differentiable soft quantization: Bridging full-precision and low-bit neural networks. In *The IEEE International Conference on Computer Vision (ICCV)*, 2019. 2

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 1, 8

[13] Sambhav R Jain, Albert Gural, Michael Wu, and Chris H Dick. Trained quantization thresholds for accurate and efficient fixed-point inference of deep neural networks. *arXiv preprint arXiv:1903.08066*, 2019. 2, 7

[14] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017. 2, 3, 6

[15] Sangil Jung, Changyong Son, Seohyung Lee, Jinwoo Son, Jae-Joon Han, Youngjun Kwak, Sung Ju Hwang, and Changkyu Choi. Learning to quantize deep networks by optimizing quantization intervals with task loss. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4350–4359, 2019. 2, 7

[16] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 7

[17] Fengfu Li, Bo Zhang, and Bin Liu. Ternary weight networks. In *NIPS Workshop on EMDNN*, 2016. 2

[18] Qian Lou, Feng Guo, Minje Kim, Lantao Liu, and Lei Jiang. Autoq: Automated kernel-wise neural network quantization. In *International Conference on Learning Representations*, 2020. 3

[19] Christos Louizos, Matthias Reisser, Tijmen Blankevoort, Efstratios Gavves, and Max Welling. Relaxed quantization for discretized neural networks. In *International Conference on Learning Representations*, 2019. 1, 2, 3, 4, 7

[20] Christos Louizos, Max Welling, and Diederik P. Kingma. Learning sparse neural networks through l0 regularization. In *International Conference on Learning Representations*, 2018. 5, 6

[21] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017. 2, 3, 5, 6

[22] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019. 7

[23] Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 525–542, Cham, 2016. Springer International Publishing. 2

[24] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted

residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 8

[25] Oran Shayer, Dan Levi, and Ethan Fetaya. Learning discrete weights using the local reparameterization trick. In *International Conference on Learning Representations*, 2018. 2

[26] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 7

[27] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014. 2, 4

[28] Joe Staines and David Barber. Variational optimization. *arXiv preprint arXiv:1212.4507*, 2012. 3

[29] Stefan Uhlich, Lukas Mauch, Fabien Cardinaux, Kazuki Yoshiyama, Javier Alonso Garcia, Stephen Tiedemann, Thomas Kemp, and Akira Nakamura. Mixed precision dnns: All you need is a good parametrization. In *International Conference on Learning Representations*, 2020. 2, 7

[30] Kuan Wang, Zhijian Liu, Yujun Lin, Ji Lin, and Song Han. Haq: Hardware-aware automated quantization with mixed precision. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8612–8620, 2019. 2

[31] Penghang Yin, Shuai Zhang, Jiancheng Lyu, Stanley Osher, Yingyong Qi, and Jack Xin. Blended coarse gradient descent for full quantization of deep neural networks. *arXiv preprint arXiv:1808.05240*, 2018. 2

[32] Dongqing Zhang, Jiaolong Yang, Dongqiangzi Ye, and Gang Hua. Lq-nets: Learned quantization for highly accurate and compact deep neural networks. In *European Conference on Computer Vision (ECCV)*, 2018. 2

[33] Xiandong Zhao, Ying Wang, Xuyi Cai, Cheng Liu, and Lei Zhang. Linear symmetric quantization of neural networks for low-precision integer hardware. In *International Conference on Learning Representations*, 2020. 2, 7

[34] Shuchang Zhou, Yuxin Wu, Zekun Ni, Xinyu Zhou, He Wen, and Yuheng Zou. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *CoRR*, abs/1606.06160, 2016. 2