Title

CSQ: Growing Mixed-Precision Quantization Scheme with Bi-level Continuous Sparsification

Permalink

https://escholarship.org/uc/item/73h9v5mn

Authors

Xiao, Lirui
Yang, Huanrui
Dong, Zhen
et al.

Publication Date

2023-07-13

DOI

10.1109/dac56929.2023.10247982

CSQ: Growing Mixed-Precision Quantization Scheme with Bi-level Continuous Sparsification

Lirui Xiao[*,1], Huanrui Yang[*,2], Zhen Dong[2], Kurt Keutzer[2], Li Du[⊠,1], Shanghang Zhang[⊠,3]
[1]Nanjing University, [2]University of California, Berkeley, [3]Peking University
{lirxiao,ldu}@nju.edu.cn {huanrui,zhendong,keutzer}@berkeley.edu
shanghang@pku.edu.cn

***Abstract*—Mixed-precision quantization has been widely applied on deep neural networks (DNNs) as it leads to significantly better efficiency-accuracy tradeoffs compared to uniform quantization. Meanwhile, determining the exact precision of each layer remains challenging. Previous attempts on bit-level regularization and pruning-based dynamic precision adjustment during training suffer from noisy gradients and unstable convergence. In this work, we propose Continuous Sparsification Quantization (CSQ), a bit-level training method to search for mixed-precision quantization schemes with improved stability. CSQ stabilizes the bit-level mixed-precision training process with a bi-level gradual continuous sparsification on both the bit values of the quantized weights and the bit selection in determining the quantization precision of each layer. The continuous sparsification scheme enables fully-differentiable training without gradient approximation while achieving an exact quantized model in the end. A budget-aware regularization of total model size enables the dynamic growth and pruning of each layer's precision towards a mixed-precision quantization scheme of the desired size. Extensive experiments show CSQ achieves better efficiency-accuracy tradeoff than previous methods on multiple models and datasets.**

***Index Terms*—Quantization, continuous sparsification, efficient neural network**

I. Introduction

With the wide application of deep neural networks (DNNs) in mobile and edge applications [1], [2], improving the efficiency of DNNs has been extensively researched. Quantization, which converts the weight and activation of the DNN model from high-precision floating point values to low-precision fixed-point representations, has been widely used to improve DNN efficiency [3]–[5]. Besides largely reducing the number of bits to store the model, the fixed-point representation achieved by linear quantization also enables the use of fixed-point arithmetic units, which largely reduces the area and energy cost, and leads to significant speedup compared to the floating-point counterparts [6].

Nevertheless, the quantization process introduces perturbation to the optimal weight value, which hinders the performance of quantized DNNs. To improve quantized model performance, previous research identifies that not all layers in a DNN are equally sensitive to quantization [7], [8], which leads to the idea of mixed-precision quantization: Sensitive layers are allowed to keep higher precision, while less sensitive layers are quantized to lower precision, therefore reaching a better model size-performance tradeoff.

Due to the large and discrete design space, the difficulty of mixed-precision quantization lies in determining the exact precision of each layer. Previous work tackles the precision assignment problem via reinforcement learning-based search [9] or utilizes higher-order sensitivity statistics computed on the pretrained model [7], [10]. However, the search-based method is costly to run, and the statistics in the pretrained model do not capture the potential sensitivity changes during the model training process. Dynamically achieving a mixed-precision quantization scheme during training is also attempted through the lens of bit-level structural sparsity [8], yet the bit-level training process and the periodic precision adjustment in training both lead to an unstable convergence [8].

In this work, we aim to improve the stability of bit-level training and precision adjustment to achieve a better convergence toward a mixed-precision quantized DNN. We locate two main factors of the instability: 1) the binary selection of bit value, and 2) the binary selection of using a certain bit or not in determining the precision of each layer. Previous methods approximate the gradient of these discrete selections via straight-through estimator (STE) [11], which can be noisy and hinders convergence. Instead, this work proposes Continuous Sparsification Quantization (CSQ). CSQ utilizes the idea of continuous sparsification [12], [13] to relax both levels of discrete selection with a series of smooth parameterized gate functions. The smoothness enables fully differentiable training of the bit-level model without gradient approximation, while proper scheduling of the gate function parameter enables the model to converge to an exact quantized form without additional rounding. We further integrate budget-aware regularization on the bit selection into the pipeline, in order to induce a mixed-precision quantization scheme under the budget constraint through an end-to-end differentiable training process. To the best of our knowledge, CSQ is the first to make the following contributions:

- Utilize continuous sparsification technique to improve bit-level training of quantized DNN;
- Relax precision adjustment in the search of mixed-precision quantization scheme into smooth gate functions;
- Combine the bi-level continuous sparsification into effectively inducing high-performance mixed-precision DNNs.

[*] Equal contribution.
[⊠] Corresponding Author.

The effectiveness of CSQ is well supported by extensive empirical results. For instance, on the CIFAR-10 dataset, CSQ achieves 1.5× further compression over BSQ [8] for ResNet-20 model under similar accuracy, and achieves 16× lossless compression for the VGG19BN model. On ImageNet, CSQ achieves a lossless 10.7× compression for the ResNet-18 model, which is 0.43% higher top-1 accuracy than the same-sized LQ-Net [5]. For ResNet-50, CSQ leads to a 17% further compression than BSQ at the same accuracy.

II. Related work

A. Mixed-precision quantization

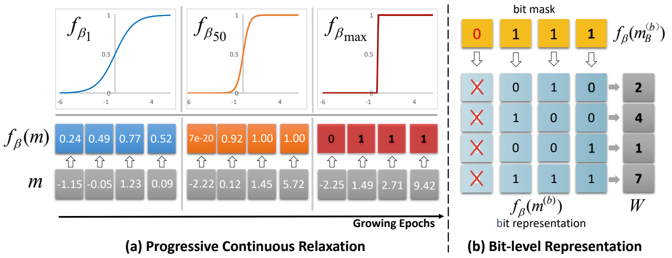
The key research question of mixed-precision quantization has been how to design a set of bit schemes that achieve the best performance-size tradeoff. Early attempts use manual design heuristics such as keeping the first and last layer at a higher precision [14], [15]. Searching-based methods like HAQ [9] utilize reinforcement learning to determine the quantization scheme, yet the search cost is often high, especially for deeper models with an exponentially large search space. Another line of methods directly measures the sensitivity of each layer with metrics like Hessian eigenvalue [7] or Hessian trace [10]. However, such methods only incorporate the sensitivity of the pretrained full-precision model, without considering the change of sensitivity when the weights are being quantized or being updated in the quantization-aware training process. Bit-level sparsity quantization (BSQ) [8] makes the first attempt to simultaneously induce mixed-precision quantization scheme and train the quantized DNN model within a single round of training. BSQ considers each bit of the quantized model as independent trainable variables, and achieves mixed-precision quantization scheme by inducing bit-level structural sparsity. The bit-level representation of layer weight W can be formulated as:

$$W = \frac{s}{2^n - 1} \text{Round} \left[\sum_{b=0}^{n-1} \left(W_p^{(b)} - W_n^{(b)} \right) 2^b \right], \tag{1}$$

where s is the scaling factor, $W_p^{(b)}$ and $W_n^{(b)}$ are the b-th bit of the positive and negative values in W respectively, and n is the quantization precision of the layer. Though BSQ leads to good empirical results, the rounding in the bit-level representation requires straight-through gradient estimation [11] on the bit variables, which can be inaccurate. Also, the hard precision adjustment performed via bit pruning during training hinders the convergence stability. In this work, we relax both bit-level training and precision adjustment with continuous sparsification, leading to improved stability and performance over BSQ.

B. Sparse optimization and continuous sparsification

The difficulty of optimizing discrete values isn't only faced by quantization research. Research on DNN pruning also needs to accommodate the binary mask of selecting a weight element/filter or not into the model training process. Minimizing the ℓ_0 regularization, which is the sum of the binary weight selection mask, has been identified as a straightforward and unbiased method to induce sparse neural network [16], but is difficult due to the discrete nature of the mask. Therefore attempts have been made to relax the binary constraint on the mask to enable gradient-based training.



(a) Progressive Continuous Relaxation

(b) Bit-level Representation

Figure 1: Illustrating the bi-level continuous sparsification used in CSQ. (a) The change of temperature sigmoid gating function throughout the training process. (b) Representing quantized model with bit mask and bit representation.

Louizos et al. first propose to consider the binary mask as stochastic gates, whose distribution can then be relaxed into "Hard concrete distribution" [16] or "Scale mixture of Gaussian" [17] with learnable parameters. However, the gradient estimation required in stochastic optimization leads to high variance and performs poorly on larger models. Continuation methods, on the other hand, approximate the binary constraint by relaxing it with a smooth gating function, while gradually making closer approximations to the binary gate as training progresses. For instance, continuous sparsification [12], [13] relax the binary gate $I(x \geq 0)$ as a Sigmoid function with temperature as formulated in Equation (2).

$$I(x \geq 0) \sim f_\beta(x) = \sigma(\beta x) = \frac{1}{1 + e^{-\beta x}}. \tag{2}$$

Temperature β controls the smoothness of the relaxed gate, which grows exponentially with the training epochs. A smaller β is used at early epochs to enable a smooth optimization. A larger β is used in later epochs to better approximate the discrete binary gate. As continuous sparsification is mainly explored under the DNN pruning setting, this work serves as the first attempt to apply bi-level continuous sparsification on both bit-level training and bit selection to induce a mixed-precision quantized DNN model.

III. Method

This section introduces our method of growing mixed-precision quantized models. Section III-A formulates the bi-level continuous sparsification of the bit-level representation for smooth optimization. Section III-B describes the budget-aware model size regularization that controls the growth and prune of layer precision. The overall algorithm of CSQ is provided in Section III-C.

A. Bi-level continuous sparsification of quantized DNN model

To represent a quantized DNN model, we need: 1) the quantization precision of each layer, and 2) the quantized value of each weight element. Both properties take discrete values, which prevent direct gradient-based updates. This work aims to relax the discrete optimization of these properties with a continuous differentiable function, enabling a smooth and differentiable optimization.

Specifically, inspired by previous attempts on continuous sparsification for DNN pruning [13], we utilize the temperature Sigmoid function $f_\beta(\cdot)$ defined in Equation (2) to relax the binary representation, where f_β can directly replace the bit-level weight $W_p^{(b)}$ and $W_n^{(b)}$ defined in Equation (1). For example, consider a layer with weight tensor W under a linear symmetric n-bit quantization, the quantized weight can be relaxed as

$$W = \frac{s}{2^n - 1} \sum_{b=0}^{n-1} \left[\left(f_\beta \left(m_p^{(b)} \right) - f_\beta \left(m_n^{(b)} \right) \right) 2^b \right], \quad (3)$$

where $f_\beta(m_p^{(b)})$ and $f_\beta(m_n^{(b)})$ are the relaxed bit-level representation of the positive and negative values in W respectively, with $m_p^{(b)}$ and $m_n^{(b)}$ taking any real values.

In training, we consider $s, m_p^{(b)}, m_n^{(b)}$ instead of W as trainable variables. We exponentially increase the value of β with the number of epochs. In this way, the trainable variables can be optimized smoothly in the early training stage, while gradually converging to an exact quantized model as $f_\beta(\cdot)$ converges to a unit step function with $\beta \to +\infty$, as illustrated in Figure 1(a). Since no rounding is applied, there's no approximation of gradient required.

With the quantized model representation in hand, the precision of each layer can be controlled as selecting the number of bits to be used. This can be formulated as having a binary bit mask $q_B \in \{0, 1\}^n$ in each layer as

$$W = \frac{s}{2^n - 1} \sum_{b=0}^{n-1} \left[\left(f_\beta \left(m_p^{(b)} \right) - f_\beta \left(m_n^{(b)} \right) \right) 2^b q_B^{(b)} \right], \quad (4)$$

where $q_B^{(b)} = 1$ if the bit is selected and 0 otherwise. Thus the precision of the layer can be computed as $\sum_b q_B^{(b)}$.

Since q_B is binary, it can also be relaxed with continuous sparsification. The resulting quantized model is formulated in Equation (5), and demonstrated in Figure 1(b).

$$W = \frac{s}{2^n - 1} \sum_{b=0}^{n-1} \left[\left(f_\beta \left(m_p^{(b)} \right) - f_\beta \left(m_n^{(b)} \right) \right) 2^b f_\beta \left(m_B^{(b)} \right) \right]. \quad (5)$$

Here we can use the same temperature scheduling for both bit masks and bit representations of each layer.

B. Budget-aware growing of mixed-precision quantization scheme

Now that we have relaxed both bit representation and bit selection of a quantized DNN in Equation (5), the next step is to adjust the precision of each layer so that it can achieve a mixed-precision quantization scheme within the available budget. This can be achieved with an ℓ_1 regularization over the bit-mask of each layer, as

$$R(m_B) = \sum_b f_\beta \left(m_B^{(b)} \right). \quad (6)$$

With the regularization, the final training objective is

$$\min_{s, m_p, m_n, m_B} \mathcal{L}(W) + \lambda \Delta_S \sum_{\text{Layer}} R(m_B), \quad (7)$$

where $\mathcal{L}(\cdot)$ is the original loss of DNN training, W is the model weight parameterized as Equation (5), λ is the base regularization strength, and Δ_S is the budget-aware scaling factor. Δ_S is added to encourage more pruning when that current model size is significantly large than the budget, less pruning if the current model size is close to the budget, and growing (have a negative regularization) when the current model size is smaller than the budget. Therefore we define Δ_S as the average quantization precision of all elements in the current model minus the targeted average precision of the budget. We count the precision of each layer during the training process based on the value of m_B, where the precision is determined as $\sum_b \left[m_B^{(b)} \geq 0 \right]$. The training objective is end-to-end differentiable with respect to the scaling factor s, bit representation m_p, m_n, and bit mask m_B of each layer, without the need to apply gradient approximation via straight-through estimation.

C. Overall training algorithm

Putting everything together, performing stochastic gradient descent with the derived objective in Equation (7) leads to the CSQ algorithm, as illustrated in detail in Algorithm 1. Bit representations and bit masks are trained simultaneously, leading to a joint optimization for both model weight values and quantization precision. The sigmoid temperatures β is scheduled to grow exponentially with the training epochs, gradually converting the model with smooth gating functions into an exactly quantized model, where f_β converges to a unit-step Sign function $I(m \geq 0)$ at $\beta_{\max}$.

For complicated tasks like ImageNet, we can further boost the final model performance by applying an additional finetuning of the achieved mixed-precision quantized model. During the finetuning process, we fix the quantization scheme of each layer, while only tuning the bit representation s, m_p, m_n of the selected bits in each layer. We rewind the temperature β for the bit representation back to 1, and redo the exponential temperature scheduling with the number of finetuning epochs. This process gives the bit representations adequate flexibility to further improve the model performance under the mixed-precision quantization scheme found by CSQ.

IV. Evaluation

This section summarizes the empirical results of CSQ. We evaluate CSQ using ResNet-20 [18] and VGG19BN [19] on CIFAR-10 [20], and using ResNet-18 and ResNet-50 [18] on ImageNet [21]. We compare the results of our method with existing uniform [3]–[5] and mix-precision [7]–[9], [22] quantization methods. Ablation studies are also provided.

A. Experimental Setup

We use the same set of hyperparameters for experiments conducted on the same model. All models are trained with SGD with an initial learning rate of 0.1 and a cosine annealing

Algorithm 1 Bi-level continuous sparsification

```
Input: Data $\boldsymbol{X} = (x_i)_{i=1}^n$, labels $\boldsymbol{Y} = (y_i)_{i=1}^n$
Output: mixed-precision model $G$
  Initialize: $s, m_p, m_n, m_B$ in $G$
  Initialize temperature: $\beta_0 = 1$, set $\beta_{\max} = 200$
  # CSQ Training
  for $epoch = 0, \dots, T$ do
      Temperature scheduling $\beta = \beta_0 \beta_{\max}^{epoch/T}$
      for $i = i, \dots, n$ do
          Sample mini-batch $x_i, y_i$ from $\boldsymbol{X}, \boldsymbol{Y}$
          Compute model weight $W$ using Eq. (5)
          Update trainable parameters with Eq. (7)
  # Mixed-precision finetuning (optional)
  Fix bit selection $q_B^{(b)} = I(q_B^{(b)} \geq 0)$
  Temperature rewind $\beta = \beta_0$
  for $epoch = 0, \dots, T'$ do
      Temperature scheduling $\beta = \beta_0 \beta_{\max}^{epoch/T'}$
      for $i = i, \dots, n$ do
          Sample mini-batch $x_i, y_i$ from $\boldsymbol{X}, \boldsymbol{Y}$
          Compute model weight $W$ using Eq. (4)
          Update $s, m_p, m_n$ with $\mathcal{L}(W)$
  return $G$
```

schedule. We use a linear learning rate warm-up for the first 5 epochs for ImageNet experiments. Weight decay is set to $5e-4$ for CIFAR-10 and $1e-4$ for ImageNet. Momentum is set to 0.9. All models are trained from scratch. On CIFAR-10, ResNet-20 model is trained with CSQ for 600 epochs, and VGG for 300 epochs, without finetuning. ImageNet models are trained with 200 CSQ epochs plus 100 epochs of finetuning after finalizing the quantization scheme, as introduced in Algorithm 1. These training epochs are comparable with the total training epochs (pretraining + finetuning) used in previous methods like BSQ [8] and HAWQ [7].

For the CSQ training process, we set the shape of the bit representation and bit mask to uniform 8-bit in each layer, as in most cases 8-bit is adequate for a lossless quantization. Since CSQ does not control activation quantization, we quantize the activation uniformly throughout the training process, whose precision is reported in the “A-Bits” column in the tables. We set the base regularization strength λ as 0.01 for training all models. The maximum temperature of the soft gate f_β function for both bit representation weight and bit mask is set as 200, which will be reached in the last epoch. At the last epoch f_β will turn into a steep step function, where all of its output should be either 0 or 1. Additionally, to ensure an exactly quantized model at the end of the training, we set all gate functions to the unit-step function before the final validation.

B. Experimental Results

This section compares our results with previous quantization methods. In all tables “FP” refers to the full-precision model; “MP” denotes mixed-precision weight quantization; and “T” denotes the target precision of CSQ. Weight compression ratio “Comp” is computed with respect to the full-precision model.

Table I: Quantization results of ResNet-20 models on the CIFAR-10 dataset.

A-Bits	Method	W-Bits	Comp(×)	Acc(%)
32	FP	32	1.00	92.62
	LQ-Nets [5]	3	10.67	92.00
	BSQ [8]	MP	19.24	91.87
	CSQ T1	**MP**	**26.67**	**91.70**
	CSQ T2	**MP**	**16.00**	**92.68**
3	LQ-Nets	3	10.67	91.60
	PACT [4]	3	10.67	91.10
	DoReFa [3]	3	10.67	89.90
	BSQ	MP	11.04	92.16
	CSQ T2	**MP**	**16.93**	**92.14**
	CSQ T3	**MP**	**10.49**	**92.42**
2	LQ-Nets	2	16.00	90.20
	PACT	2	16.00	89.70
	DoReFa	2	16.00	88.20
	BSQ	MP	18.85	90.19
	CSQ T1	**MP**	**22.86**	**90.08**
	CSQ T2	**MP**	**16.41**	**90.33**

Table II: Quantization results of VGG19BN models on the CIFAR-10 dataset.

A-Bits.	Method	W-Bits.	Comp(×)	Acc(%)
32	FP	32	1.00	94.22
	LQ-Nets [5]	3	10.67	93.80
	CSQ T2	**MP**	**16.00**	**94.10**
8	ZeroQ [24]	4	8.00	92.69
	ZAQ [25]	4	8.00	93.06
	CSQ T3	**MP**	**10.67**	**93.90**
4	QUANOS [26]	MP	7.11	90.70
	CSQ T3	**MP**	**10.67**	**93.62**
3	LQ-Nets [5]	3	10.67	93.80
	Non-Linear [23]	3	9.14	93.40
	CSQ T2	**MP**	**16.00**	**93.58**

CIFAR-10 results. For ResNet-20 on CIFAR-10, we compare CSQ with various previous methods. As shown in Table I, CSQ outperforms previous methods under all activation precision. Notably, the CSQ-T2 model with 3-bit activation enables an 1.5× further compression vs. BSQ [8] under the same accuracy. Similarly, CSQ also demonstrated superior performance on the VGG19BN model, as shown in Table II. Under full-precision activation CSQ enables a nearly lossless 16× compression, largely pushing the frontier of previous methods. Furthermore, CSQ method even surpasses non-linear quantizers, which are generally powerful but unfriendly for implementation. Comparing to non-linear quantization methods LQ-Nets [5] and [23], CSQ achieves both higher accuracy and higher compression ratio up to 1.8×.

ImageNet results. To evaluate the scalability of CSQ, we perform experiments on the large-scale ImageNet dataset on deeper models. Table III shows the results of ResNet18 and ResNet-50 obtained by different quantization methods [3]–[5], [8], [9], [22], among which CSQ consistently shows strong performance. For ResNet-18, the model obtained by CSQ-T3 with an average of 3-bit weight precision and 8-bit activation precision achieves almost the same accuracy as the full-precision baseline. CSQ-T2, with 4-bit activation, achieves a higher compression rate (15.23×) with a tiny accuracy drop. This result significantly outperforms the W4A4 uniformly

Table III: Quantization results of ResNet-18 and ResNet-50 models on the ImageNet dataset.

	ResNet-18			ResNet-50		
Method	W-Bits	Comp(×)	Acc(%)	W-Bits	Comp(×)	Acc(%)
FP	32	1.00	69.76	32	1.00	76.13
DoReFa [3]	5	6.40	68.4	3	10.67	69.90
PACT [4]	4	8.00	69.2	3	10.67	75.30
LQ-Nets [5]	3	10.67	69.30	3	10.67	74.20
HAWQ-V3 [22]	4	8.00	68.45	4	8.00	74.24
HAQ [9]	\	\	\	MP	10.57	75.30
BSQ [8]	\	\	\	MP	13.90	75.16
CSQ T2	**MP**	**15.23**	**69.11**	**MP**	**14.54**	**75.25**
CSQ T3	**MP**	**10.67**	**69.73**	**MP**	**10.67**	**75.47**

quantized model reported by HAWQ-V3 [22] with 1.9× further compression. For ResNet-50, CSQ also achieves better efficiency-accuracy tradeoff, beating strong mixed-precision quantization baseline HAQ [9] and BSQ [8].

C. Ablation study

In this section, we discuss the key designs of the CSQ algorithm, including the comparison of the proposed continuous sparsification vs. STE in quantization-aware training (QAT), the effectiveness of the budget-aware model size regularization, and the control of accuracy-model size tradeoff. The quantization schemes obtained by CSQ under different model sizes are also demonstrated. All experiments in this section are conducted using ResNet-20 models [18] with 3-bit activation on the CIFAR-10 dataset [20].

Effectiveness of continuous sparsification. In this work, we replace the commonly used STE-based [11] QAT with the proposed bit-level continuous sparsification in training quantized models. Table IV compares the QAT performance for a uniformly-quantized model trained with STE (STE-Uniform) and continuous sparsification (CSQ-Uniform). All models are trained from scratch with fixed-weight precision. STE-Uniform follows the implementation in [27] where the floating-point latent weight is linearly quantized in the forward pass, and accumulates gradients in the backward pass with STE. For CSQ-Uniform, we utilize the weight parameterization in Equation (3), where no bit mask is applied and only bit representations are trained. Under all precision, CSQ-Uniform outperforms STE-Uniform significantly, showing the effectiveness of bit-level continuous sparsification in leading to better convergence of quantized models. Additionally, as we propose to apply another level of continuous sparsification on the bit masks, the resulted bi-level continuous sparsification enables the proposed CSQ to find a better mixed-precision quantization scheme (CSQ-MP), which further boosts the performance over uniformly quantized counterparts.

Budget-aware model size regularization. As proposed in Section III-B, CSQ utilizes a budget-aware regularization to encourage the quantization scheme to meet a target precision. In this section, we explore the influence of the two hyperparameters in the regularization: base strength λ and the target precision, on the final averaged precision achieved by CSQ.

Table IV: CSQ vs. STE-based QAT performance. STE-Uniform training is implemented following [27].

W-Bits	QAT method	Accuracy (%)
4	STE-Uniform [27]	88.89
	CSQ-Uniform	91.93
	CSQ-MP	**92.68**
3	STE-Uniform [27]	87.68
	CSQ-Uniform	91.74
	CSQ-MP	**92.62**
2	STE-Uniform [27]	84.35
	CSQ-Uniform	91.67
	CSQ-MP	**92.34**

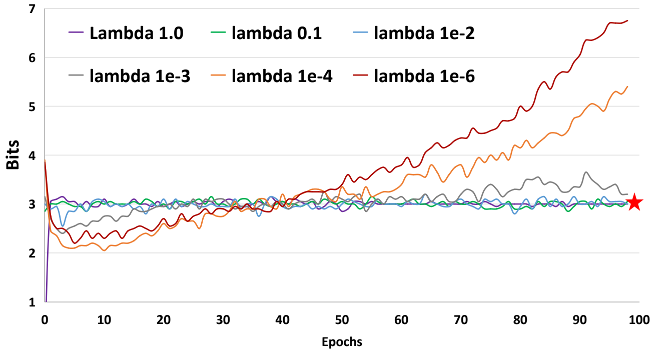


Figure 2: Effect of base regularization strength λ on the averaged model precision during training. All experiments are done with a target of 3-bit, as indicated by the "red star".

Figure 2 visualizes the changes in averaged precision of CSQ models trained with different initial values of λ. Note that since the bit masks are not exactly binary, we record the precision of each layer during training as $\sum_b \left[m_B^{(b)} \geq 0\right]$, as if the bit mask is gated with a step function. The target precision is set to be 3-bit for all trails. We note that the final model precision is not sensitive to the choice of λ in a large range between 1e-3 and 1, where the model consistently converges to the desired target precision. λ being too small (e.g. 1e-4 and 1e-6) resulted in less regularization strength to control the model precision effectively, which is as expected. To this end, we set $\lambda = 0.01$ for all the experiments, which works well across different model architectures and datasets.

Figure 3 shows the change of averaged model precision

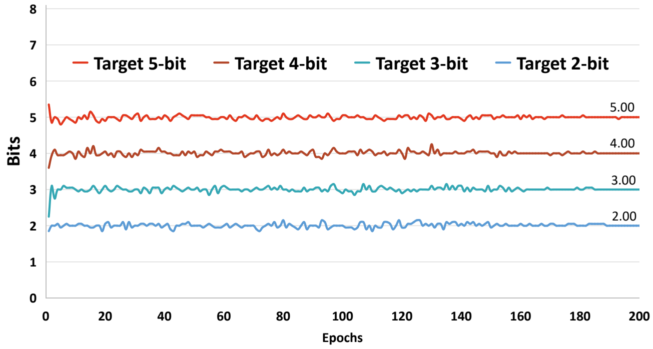


Figure 3: Effect of different target precision on the averaged model precision during training.

Table V: Accuracy-size trade-off under different target bits.

Target	1-bit	2-bit	3-bit	4-bit	5-bit	FP
Ave. prec.	1.00	1.97	3.05	4.00	5.05	32
Comp($\times$)	32	16.24	10.49	8.00	6.34	1
CSQ acc.	90.33	91.70	92.42	92.51	92.61	92.62

during CSQ training with different target precision. It can be seen that the proposed budget-aware regularization controls the model precision to be close to the target throughout the training process, and accurately converges to the target precision at the end of CSQ training. The effectiveness of the regularization enables us to explicitly control the outcome of CSQ by setting an exact model size budget, mitigating the heuristic search of proper regularization strength for a specific model size required by previous methods [8], [12]. The stability of model precision throughout the training process also enables better convergence of the model, effectively leading to a mixed-precision model that is within the target budget while enjoying a preeminent accuracy.

Accuracy-model size trade-off. Performing CSQ training with different target precision effectively controls the size of the resulted quantization scheme, therefore exploring the trade-off between model size and accuracy. Table V summarizes the quantization results under different target bits obtained by CSQ using ResNet-20 on the CIFAR-10 dataset. Averaged precision ("Avg. prec.") is computed across all model layers, and "Comp($\times$)" indicates the compression rate compared to the 32-bit floating-point model. The floating-point performance is also provided under the "FP" column for reference. As observed previously, the final average precision achieved by CSQ is fairly precise compared to the target. CSQ enables a lossless 5-bit quantization, while lower precision can be effectively achieved with the cost of a small accuracy drop.

Layer-wise quantization schemes achieved with CSQ. Figure 4 shows the final precision of each layer in the mixed-precision quantization scheme obtained by CSQ under different target bits. Comparing among each other, the trends of quantization precision for the layers are generally consistent under different target bits, which echoes observations in previous mixed-precision quantization work [7], [8]. Interestingly, the importance ranking produced by CSQ is somewhat different from that of the previous work [7], [8]. Our results show a roughly rising trend in precision from the input to the output layers, whereas the results from [7], [8] show a declining trend in precision, with the lowest precision in the final stage. Given the superior efficiency-accuracy tradeoff achieved by CSQ comparing to these methods, it shows that the heuristic-based importance criteria used in previous work may not accurately reflect the quantized model performance, while CSQ discovers better mixed-precision quantization schemes.

V. Conclusion

In this work, we propose CSQ, a novel method for bit-level mixed-precision DNN training using continuous sparsification. We improve the stability of training quantized DNNs with the bi-level continuous sparsification relaxation, and obtain mix-quantization schemes with explicit target precision utilizing the budget-aware model size regularization. Extensive experiments show the effectiveness of CSQ, where we achieve both higher accuracy and smaller quantization precision on various models and datasets comparing to state of the arts.

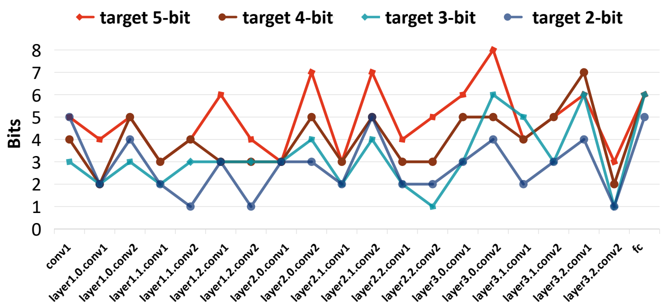


Figure 4: Layer-wise precision comparison of the quantization schemes produced by CSQ under different target bits;

References

[1] Mark Sandler et al. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the CVPR*, pages 4510–4520, 2018.

[2] Jie Hu et al. Squeeze-and-excitation networks. In *Proceedings of the CVPR*, pages 7132–7141, 2018.

[3] Shuchang Zhou et al. Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients. *arXiv preprint arXiv:1606.06160*, 2016.

[4] Jungwook Choi et al. Pact: Parameterized clipping activation for quantized neural networks. *arXiv preprint arXiv:1805.06085*, 2018.

[5] Dongqing Zhang et al. Lq-nets: Learned quantization for highly accurate and compact deep neural networks. In *ECCV*, pages 365–382, 2018.

[6] Mark Horowitz. 1.1 computing's energy problem (and what we can do about it). In *ISSCC*, 2014.

[7] Zhen Dong et al. Hawq: Hessian aware quantization of neural networks with mixed-precision. In *ICCV*, pages 293–302, 2019.

[8] Huanrui Yang et al. Bsq: Exploring bit-level sparsity for mixed-precision neural network quantization. *arXiv preprint arXiv:2102.10462*, 2021.

[9] Kuan Wang et al. Haq: Hardware-aware automated quantization with mixed precision. In *Proceedings of the CVPR*, pages 8612–8620, 2019.

[10] Zhen Dong et al. Hawq-v2: Hessian aware trace-weighted quantization of neural networks. *NeurIPS*, 33:18518–18529, 2020.

[11] Yoshua Bengio et al. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

[12] Pedro Savarese et al. Winning the lottery with continuous sparsification. *NeurIPS*, 33:11380–11390, 2020.

[13] Xin Yuan et al. Growing efficient deep networks by structured continuous sparsification. *arXiv preprint arXiv:2007.15353*, 2020.

[14] Mohammad Rastegari et al. Xnor-net: Imagenet classification using binary convolutional neural networks. In *ECCV*, pages 525–542. Springer, 2016.

[15] Fengfu Li et al. Ternary weight networks. *arXiv preprint arXiv:1605.04711*, 2016.

[16] Christos Louizos et al. Learning sparse neural networks through l_0 regularization. *arXiv preprint arXiv:1712.01312*, 2017.

[17] Christos Louizos et al. Bayesian compression for deep learning. In *NeurIPS*, pages 3288–3298, 2017.

[18] Kaiming He et al. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on CVPR*, pages 770–778, 2016.

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

[20] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[21] Jia Deng et al. Imagenet: A large-scale hierarchical image database. In *CVPR 2009*, pages 248–255. Ieee, 2009.

[22] Zhewei Yao et al. Hawq-v3: Dyadic neural network quantization. In *ICML*, pages 11875–11886. PMLR, 2021.

[23] Marcelo Gennari do Nascimento et al. Finding non-uniform quantization schemes using multi-task gaussian processes. In *ECCV*, pages 383–398. Springer, 2020.

[24] Yaohui Cai et al. Zeroq: A novel zero shot quantization framework. In *Proceedings of the CVPR*, pages 13169–13178, 2020.

[25] Yuang Liu et al. Zero-shot adversarial quantization. In *Proceedings of the CVPR*, pages 1512–1521, June 2021.

[26] Priyadarshini Panda. Quanos: adversarial noise sensitivity driven hybrid quantization of neural networks. In *Proceedings of the ACM/IEEE International Symposium on Low Power Electronics and Design*, pages 187–192, 2020.

[27] A. Polino et al. Model compression via distillation and quantization. *arXiv preprint arXiv:1802.05668*, 2018.