MetaQuant: Learning to Quantize by Learning to Penetrate Non-differentiable Quantization

Chen Shangyu

October 15, 2019

Overview

Motivation

- Traditional training-based quantization replies on Straight-Through-Estimator (STE) en enable training:
  Forward:
  $$\ell = \mathrm{Loss}(f(Q(\mathbf{W}); \mathbf{x}), y) \tag{1}$$
  Backward:
  $$\frac{\partial Q(r)}{\partial r} = \begin{cases} 1 & \text{if} \quad |r| \leq 1 \\ 0 & \text{else.} \end{cases}. \tag{2}$$
- Gradient Mismatch: the gradients of the weights are not generated using the value of weights, but rather its quantized value.

Motivation (Cont.)

- We propose to learn $\frac{\partial Q(\mathbf{W})}{\partial \mathbf{W}}$ by a neural network ($\mathcal{M}$) during quantization training.
- $\mathcal{M}$ is called *meta quantizer* and is trained together with the base quantized model.
- $\mathcal{M}$ takes $\frac{\partial L}{\partial Q(\mathbf{W})}$ and $\mathbf{W}$ as inputs in a coordinate-wise manner, then its output is assigned to $\frac{\partial L}{\partial \mathbf{W}}$ for weights update using common optimization methods such as SGD and Adam.

MetaQuant

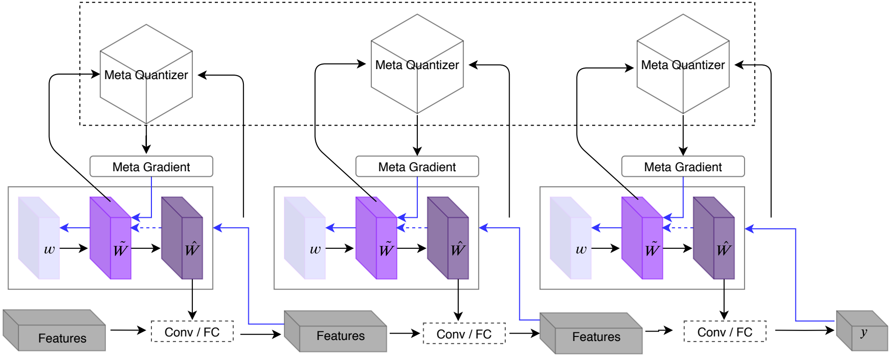


Figure: Overflow of MetaQuant

MetaQuant (Cont.)

dorefa:

$$\tilde{\mathbf{W}} = \mathcal{A}(\mathbf{W}) = \frac{\tanh(\mathbf{W})}{2\max(|\tanh(\mathbf{W})|)} + \frac{1}{2}, \quad (3)$$

$$\hat{\mathbf{W}} = Q(\tilde{\mathbf{W}}) = 2\frac{\text{round}\left[(2^k - 1)\tilde{\mathbf{W}}\right]}{2^k - 1} - 1. \quad (4)$$

BWN:

$$\tilde{\mathbf{W}} = \mathcal{A}(\mathbf{W}) = \mathbf{W}, \quad (5)$$

$$\hat{\mathbf{W}} = Q(\tilde{\mathbf{W}}) = \frac{1}{n}||\tilde{\mathbf{W}}||_{l_1} \times \text{sign}(\tilde{\mathbf{W}}). \quad (6)$$

Incorporation of Meta Quantizer in Base Training

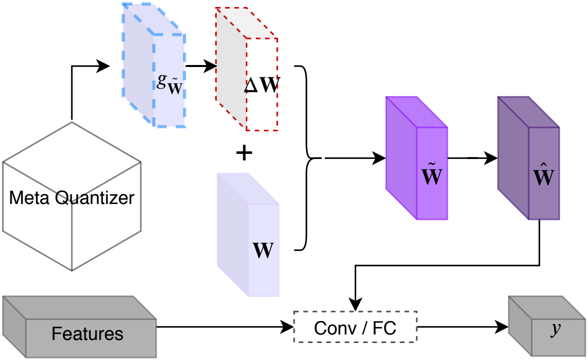


Figure: Incorporation of meta quantizerinto quantization training

Training of Meta Quantizer

Forward:

$$\begin{aligned} \tilde{\mathbf{W}}^t &= \mathcal{A}(\mathbf{W}^t) = \mathcal{A}\left[\mathbf{W}^{t-1} - \alpha \times \pi(\mathcal{M}_\phi(g_{\hat{\mathbf{W}}}^{t-1}, \tilde{\mathbf{W}}^{t-1})\frac{\partial \tilde{\mathbf{W}}^{t-1}}{\partial \mathbf{W}^{t-1}})\right], \\ \ell &= \text{Loss}\left\{f\left[Q(\tilde{\mathbf{W}}^t); \mathbf{x}\right], y\right\}, \end{aligned} \tag{7}$$

Backward:

$$\frac{\partial L}{\partial \phi^t} = \frac{\partial L}{\partial \tilde{\mathbf{W}}^t}\frac{\partial \tilde{\mathbf{W}}^t}{\partial \phi^t} = \mathcal{M}_\phi(g_{\hat{\mathbf{W}}^t}, \tilde{\mathbf{W}}^t)\frac{\partial \tilde{\mathbf{W}}^t}{\partial \phi^t}. \tag{8}$$

Design of Meta Quantizer

$$\texttt{FCGrad}: \qquad \mathcal{M}_\phi(g_{\hat{\mathbf{W}}}) = \mathsf{FCs}(\phi, \sigma, g_{\hat{\mathbf{W}}}),$$

$$\texttt{MultiFC}: \qquad \mathcal{M}_\phi(g_{\hat{\mathbf{W}}}, \tilde{\mathbf{W}}) = g_{\hat{\mathbf{W}}} \cdot \mathsf{FCs}(\phi, \sigma, \tilde{\mathbf{W}}).$$

$$\texttt{LSTMFC}: \qquad \mathcal{M}_\phi(g_{\hat{\mathbf{W}}}, \tilde{\mathbf{W}}) = g_{\hat{\mathbf{W}}} \cdot \mathsf{FCs}(\phi_{FCs}, \sigma, (\mathsf{LSTM}(\phi_{LSTM}, \tilde{\mathbf{W}}))).$$

Experiments: CIFAR10

Network	Forward	Backward	Optimization	Test Acc (%)	FP Acc (%)
ResNet20	dorefa	STE	SGD	80.745(2.113)	91.5
		`MultiFC`		**88.942(0.466)**	
		`LSTMFC`		88.305(0.810)	
		`FCGrad`		88.840(0.291)	
		STE	Adam	89.782(0.172)	
		`MultiFC`		89.941(0.068)	
		`LSTMFC`		**89.979(0.103)**	
		`FCGrad`		89.962(0.068)	
	BWN	STE	SGD	75.913(3.495)	
		`LSTMFC`		**89.289(0.212)**	
		`FCGrad`		88.949(0.231)	
		STE	Adam	89.896(0.182)	
		`LSTMFC`		90.036(0.109)	
		`FCGrad`		**90.042(0.098)**	

Experiments: CIFAR100

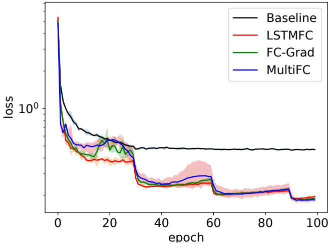
Network	Forward	Backward	Optimization	Test Acc (%)	FP Acc (%)
ResNet56	dorefa	STE	SGD	42.265(8.143)	71.22
		MultiFC		**65.791(0.415)**	
		LSTMFC		63.645(2.183)	
		FCGrad		64.351(0.935)	
		STE	Adam	66.419(0.533)	
		MultiFC		**66.588(0.375)**	
		LSTMFC		66.483(0.793)	
		FCGrad		66.564(0.351)	
	BWN	STE	SGD	34.479(11.737)	
		LSTMFC		63.346(2.253)	
		FCGrad		**64.402(1.434)**	
		STE	Adam	64.297(1.309)	
		LSTMFC		66.584(0.349)	
		FCGrad		**67.018(0.329)**	

Experiments: ImageNet

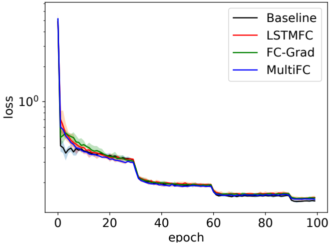
Network	Forward	Backward	Optimization	FP Top1/Top5(%)	Quant Top1/Top5 (%)
ResNet18	dorefa	STE	Adam	67.756/88.224*	58.349(2.072)/81.477(1.567)
		`MultiFC`			59.472(0.025)/82.410(0.010)
		`FCGrad`			**59.835(0.359)/82.671(0.232)**
	BWN	STE	SGD-M		59.426/83.000
		MetaQuant			**61.436/83.912**

Table: Experimental result of MetaQuant and STE using dorefa, BWN on ImageNet. *: Full-Precision accuracy is slightly lower than the reported number in `torchvision` for we use lmdb format to read in data for efficient process, which brings in a slight drop.

Convergence



(a) SGD



(b) Adam

Figure: Convergence speed of MetaQuant V.S STE using SGD/Adam in ResNet20, CIFAR10, dorefa.

Conclusion

- To learn the gradient for penetration of the non-differentiable quantization function in training-based quantization by a meta quantizer .
- This meta network is general enough to be incorporated into various base models and can be updated using the loss of the base models.