Feature Affinity Assisted Knowledge Distillation and Quantization

Penghang Yin

Department of Math and Stats, SUNY Albany

CBMS Conference: Deep Learning and Numerical PDEs

June 19-23, 2023

Outline

Knowledge Distillation

Quantization

Experiments

Knowledge Distillation

- Knowledge distillation is the process of transferring knowledge from a large model (teacher) to a smaller one (student) to improve the performance of student model.
- The so-called distillation loss [Hinton et al.'15]

$$L_{\mathrm{KD}}(\theta; x) = CE(\mathrm{softmax}(f_t(x)/\tau), \mathrm{softmax}(f_s(\theta; x)/\tau))$$

$f_s(\theta; x)$ and $f_t(x)$ are logits of the student and teacher resp., $\tau > 0$ is a hyperparameter called temperature,

$$\mathrm{softmax}(\mathbf{z})_i = \frac{\exp(z_i)}{\sum_{j=1}^{k} \exp(z_j)},$$

CE is the cross entropy $CE(p, q) = -\sum_{i=1}^{k} p_i \log q_i$

Knowledge Distillation (Cont'd)

When the (one-hot) label y is available, the regular training loss on the student network is

$$L_{\mathrm{True}}(\theta; x) = \lambda \, CE(\mathrm{softmax}(f_s(\theta; x)), y)$$

The knowledge distillation framework[Hinton et al.'15] requires to solve

$$\min_{\theta} \; \frac{1}{N} \sum_{i=1}^{N} L_{\mathrm{KD}}(\theta; x_i) + \lambda \, L_{\mathrm{True}}(\theta; x_i)$$

where $\lambda > 0$ is a regularization param.

Feature Affinity Matrix

- Consider a (reshaped) feature matrix

$$\mathbf{F} = [\mathbf{f}_1, \ldots, \mathbf{f}_{wh}] \in \mathbb{R}^{c \times wh},$$

the feature affinity matrix $\mathbf{H} \in \mathbb{R}^{wh \times wh}$ is given by the pairwise cosine similarity between two (pixel-wise) feature vectors $\mathbf{f}_i, \mathbf{f}_j \in \mathbb{R}^c$:

$$\mathbf{H}_{ij} := \frac{\langle \mathbf{f}_i, \mathbf{f}_j \rangle}{\|\mathbf{f}_i\| \|\mathbf{f}_j\|} = \cos \theta_{ij}$$

where θ_{ij} is the angle between $\mathbf{f}_i$ and $\mathbf{f}_j$.

Feature Affinity Loss

- At a given pair of layers from student and teacher networks resp., let $\mathbf{F}_s(\theta; x) \in \mathbb{R}^{c_s \times wh}$ and $\mathbf{F}_t(x) \in \mathbb{R}^{c_t \times wh}$ be the features with $c_s < c_t$, and let $\mathbf{H}_s(\theta; x)$, $\mathbf{H}_t(x)$ be the induced feature affinity matrices.
- The feature affinity loss given by l pairs of intermediate features is

$$L_{FA}(\theta; x) = \sum_{j=1}^{l} \frac{1}{w_j^2 h_j^2} \|\mathbf{H}_s^j(\theta; x) - \mathbf{H}_t^j(x)\|_F^2$$

- Feature affinity assisted knowledge distillation gives the sample loss:

$$L(\theta; x) = L_{\mathrm{KD}}(\theta; x) + \lambda_1 \, L_{\mathrm{True}}(\theta; x) + \lambda_2 \, L_{FA}(\theta; x)$$

Drop the second term if labels are unavailable (label-free distillation).

Knowledge Distillation Framework

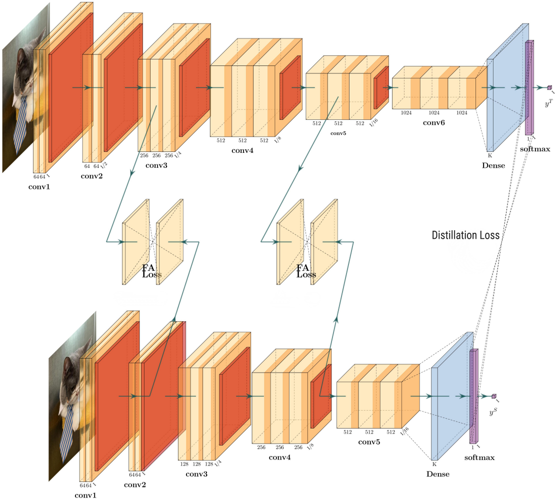


Figure 1: Feature affinity assisted knowledge distillation framework by comparing two sets of feature pairs from the student and teacher.

Existence of Low-Dimensional Feature Embeddings

Denote the cosine similarity by $\|f - g\|_{\cos} := \frac{\langle f, g \rangle}{\|f\| \, \|g\|}$.

Proposition (Johnson-Lindenstrauss-like Lemma)

Given any $\epsilon \in (0,1)$, a feature map $\mathbf{F} = [\mathbf{f_1}, \ldots, \mathbf{f}_n] \in \mathbb{R}^{d \times n}$, with $k = O(\epsilon^{-2} \log n)$, there exists a linear map $T : \mathbb{R}^d \to \mathbb{R}^k$, such that

$$(1-\epsilon)\|\mathbf{f}_i - \mathbf{f}_j\|_{\cos} \leq \|T(\mathbf{f}_i) - T(\mathbf{f}_j)\|_{\cos} \leq (1+\epsilon)\|\mathbf{f}_i - \mathbf{f}_j\|_{\cos}, \ \forall i, j$$

Fast Feature Affinity Loss

- Note that

$$L_{FA}(\theta) := \mathbb{E}_{\mathbf{z}\sim\mathcal{N}(\mathbf{0},\mathbf{I})} \|(\mathbf{H}_s(\theta; x) - \mathbf{H}_t(x))z\|^2$$

- Consider the fast FA loss with k ensembles ($k \ll wh$):

$$L_{fFA,k}(\theta) = \frac{1}{k}\sum_{i=1}^{k} \|(\mathbf{H}_s(\theta; x) - \mathbf{H}_t(x))z_i\|^2$$

where $z_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \in \mathbb{R}^{wh}$

- Concentration inequality:

$$\mathbb{P}\big(|L_{fFA,k}(\theta) - L_{FA}(\theta)| > \epsilon\big) \leq \frac{C}{\epsilon^2 k},$$

where $C = O(L_{FA}(\theta)^4)$.

Outline

Floating Point Representation

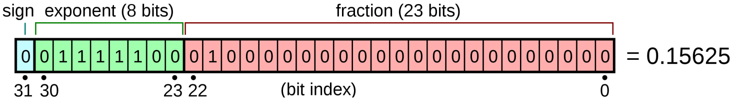
There are 3 elements in a floating point (FP) representation.

- sign
- exponent
- mantissa/fraction

Take FP32 (32-bit) as an example:

$$(-1)^{b_{31}} \cdot 2^{(b_{30}b_{29}\ldots b_{23})_2 - 127} \cdot \left(1 + \frac{b_{22}}{2} + \frac{b_{21}}{2^2} + \cdots + \frac{b_0}{2^{23}}\right)$$

with each $b_i \in \{0, 1\}$.



Integer Representation

For 8-bit integer (INT8):

- signed integer: $-127, -126 \ldots, 127$

$$(-1)^{b_7} \cdot (b_6 b_5 \ldots b_0)_2$$

- unsigned integer: $0, 1, \ldots, 255$

$$(b_7 b_6 \ldots b_0)_2$$

Integer Representation

For 8-bit integer (INT8):

- signed integer: $-127, -126 \ldots, 127$

$$(-1)^{b_7} \cdot (b_6 b_5 \ldots b_0)_2$$

- unsigned integer: $0, 1, \ldots, 255$

$$(b_7 b_6 \ldots b_0)_2$$

INT is more efficient than FP in terms of speed, but lacks of precision. Instead consider the scaled INT:

$$\delta \cdot (-1)^{b_7} \cdot (b_6 b_5 \ldots b_0)_2 \quad \text{or} \quad \delta \cdot (b_7 b_6 \ldots b_0)_2$$

allowing some multiplicative FP scalar $\delta > 0$.

INT Quantization

- Learn (scaled) low-bit INT representation (e.g., INT8) for the **weights** and **activation functions** of neural networks. (both the FP scalars and integers)

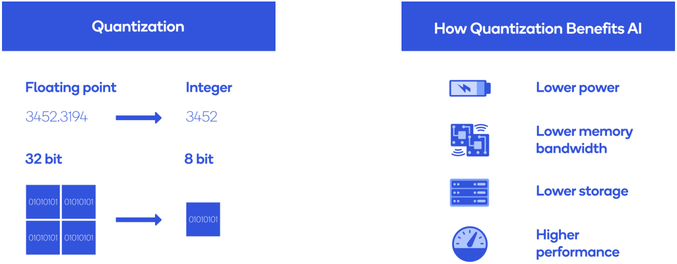
INT Quantization

- Learn (scaled) low-bit INT representation (e.g., INT8) for the **weights** and **activation functions** of neural networks. (both the FP scalars and integers)
- In inference phase, accelerate the forward propagation through linear layers:

$$\begin{aligned} W * A &= (\delta \cdot W^{\text{int}}) * (\alpha \cdot A^{\text{int}}) \\ &= (\delta \cdot \alpha) \cdot (W^{\text{int}} * A^{\text{int}}) \end{aligned}$$

  where W and A are quantized weights and activations, resp.
- The FP scalars $\delta, \alpha > 0$ are shared by the whole linear layer and activation layer, resp. (so-called layer-wise quantization).

- Empirically see an up to 16× increase in energy efficiency and a 4× memory savings by going from FP32 to INT8 quantization.



Hardware Implementation

World's first on-device demonstration of Stable Diffusion on an Android phone

Qualcomm AI Research deploys a popular 1B+ parameter foundation model on an edge device through full-stack AI optimization

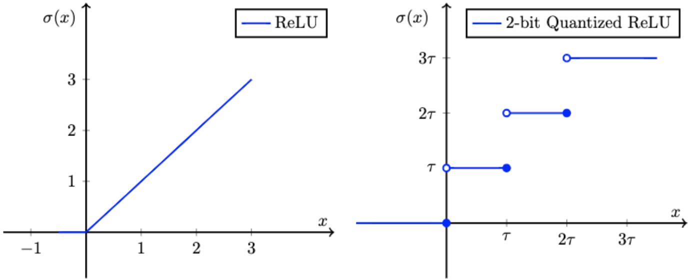
FEB 23, 2023 | Snapdragon and Qualcomm branded products are products of Qualcomm Technologies, Inc. and/or its subsidiaries.

Figure 2: Running INT8 Stable Diffusion model (1B+ params) on Android phones powered by Snapdragon mobile platform takes comparable inference time to that of FP32 model on cloud.

Computational Challenges for Quantization

Solve an optimization problem with

- highly non-convex objective of high dimension
- discrete constraint (quantized weights)
- piecewise constant objective with inapplicable gradient a.e. zero (quantized activations)



Goal: design simple algorithm that

- search along non-gradient based descent direction.
- compute projection efficiently (weight quantization).
- effectively avoid bad local minima.

Training Fully Quantized Neural Networks

$$\min_{\mathbf{w},\boldsymbol{\alpha}} \; f(\mathbf{w},\boldsymbol{\alpha}) := \frac{1}{N}\sum_{i=1}^{N} \ell_i(\mathbf{w},\boldsymbol{\alpha}) \quad \text{subject to} \quad \mathbf{w} \in \mathcal{W}.$$

- $\ell_i(\mathbf{w},\boldsymbol{\alpha}) = \ell(w_L * \sigma(\cdots\sigma(w_1 * x_i, \alpha_1), \dots, \alpha_{L-1}); y_i)$ is the sample loss with or without knowledge distillation.
- $\sigma(x,\alpha)$: unsigned INTq activation function.

$$\sigma(x,\alpha) = \sum_{k=1}^{2^q-2} k\alpha \cdot 1_{\{(k-1)\alpha < x \leq k\alpha\}} + (2^q-1)\alpha \cdot 1_{\{x > (2^q-2)\alpha\}}$$

- $\mathcal{W} = \mathbb{R}_+ \times \{\pm 1\}^n$ for INT1 (a single sign bit), and $\mathcal{W} = \mathbb{R}_+ \times \{0, \pm 1, \dots, \pm(2^{b-1}-1)\}^n$ for signed INTb, $b \geq 2$.

Weight Quantization

Given weights $\mathbf{w}^{(l)} \in \mathbb{R}^n$ (FP32) at Layer l, obtain the INTb quantization by solving

$$\min_{\delta,\mathbf{q}} \|\delta^{(l)} \cdot \mathbf{q}^{(l)} - \mathbf{w}^{(l)}\|^2$$
$$\text{s.t. } \delta^{(l)} > 0,\ \mathbf{q}^{(l)} \in \{0, \pm 1, \ldots, \pm(2^{b-1} - 1)\}^n.$$

Weight Quantization

Given weights $\mathbf{w}^{(l)} \in \mathbb{R}^n$ (FP32) at Layer l, obtain the INTb quantization by solving

$$\min_{\delta,\mathbf{q}} \|\delta^{(l)} \cdot \mathbf{q}^{(l)} - \mathbf{w}^{(l)}\|^2$$
$$\text{s.t. } \delta^{(l)} > 0,\ \mathbf{q}^{(l)} \in \{0, \pm 1, \dots, \pm(2^{b-1} - 1)\}^n.$$

Solve by alternating minimization for $b \geq 2$.

- For $b = 1$, $\mathbf{q}^{(l)} \in \{\pm 1\}^n$, it has closed-form solution [Rastegari et al.'16].

Overcoming Vanished Gradient

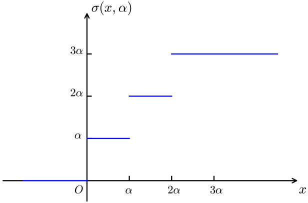
In chain rule, replace $\frac{\partial \sigma}{\partial x}$ with the proxy $\frac{\partial \tilde{\sigma}}{\partial x}$ (so-called straight through estimator [Bengio et al.'13; Yin et al.'19]).

$$\frac{\partial \ell_i(\mathbf{w},\boldsymbol{\alpha})}{\partial \mathbf{w}_{L-1}} \approx \sigma(\mathbf{X}_{L-2},\alpha_{L-2}) \circ \frac{\partial \tilde{\sigma}}{\partial x}(\mathbf{X}_{L-1},\alpha_{L-1}) \circ \mathbf{w}_L^\top \circ \nabla \ell(\mathbf{X}_L; u_i)$$

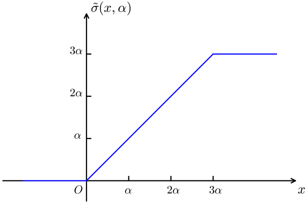
$$\frac{\partial \ell_i(\mathbf{w},\boldsymbol{\alpha})}{\partial \alpha_{L-2}} \approx \frac{\partial \sigma}{\partial \alpha}(\mathbf{X}_{L-2},\alpha_{L-2}) \circ \mathbf{w}_{L-1}^\top \circ \frac{\partial \tilde{\sigma}}{\partial x}(\mathbf{X}_{L-1},\alpha_{L-1}) \circ \mathbf{w}_L^\top \circ \nabla \ell(\mathbf{X}_L; u_i).$$

with $\mathbf{X}_l = \mathbf{w}_l * \sigma(\mathbf{X}_{l-1}, \alpha_{l-1})$ the output from the l-th linear layer.

2-bit quantized ReLU σ



clipped ReLU $\tilde{\sigma}$



- require no extra cost compared with standard gradient computation.

Analysis of Straight Through Estimator

- Given input $\mathbf{x} \in \mathbb{R}^d$ and class label $y \in \{1, \dots, k\}$, consider the two-layer netowrk with output

  $$\mathbf{o}(\mathbf{x}; \mathbf{W}) = \mathbf{V}\sigma(\mathbf{W}\mathbf{x}) \in \mathbb{R}^k$$

  with weights $\mathbf{V} \in \mathbb{R}^{k \times m}$ in the second layer fixed and known. σ is general b-bit activation function:

  $$\sigma(x) = \begin{cases} 0 & \text{if} \quad x \leq 0, \\ \text{ceil}(x) & \text{if} \quad 0 < x < 2^b - 1, \\ 2^b - 1 & \text{if} \quad x \geq 2^b - 1. \end{cases}$$

- $\arg\max_{1 \leq i \leq k} o(x; W)_i$ is the predicted class for x.

- multi-class hinge loss:

$$\ell(\mathbf{W};\mathbf{x},y) = \max\left\{0,\ 1-\left(o(\mathbf{x};\mathbf{W})_y - \max_{i\neq y} o(\mathbf{x};\mathbf{W})_i\right)\right\}$$

- solve the population risk minimization

$$\min_{\mathbf{W}\in\mathbb{R}^{m\times d}}\ f(\mathbf{W}) := \mathbb{E}_{\{\mathbf{x},y\}\sim\mathcal{D}}\left[\ell\left(\mathbf{W};\mathbf{x},y\right)\right],$$

- chain rule to compute partial gradient w.r.t. the j-th row $\mathbf{w}_j^\top$ of $\mathbf{W}$:

$$\begin{aligned}\nabla_{\mathbf{w}_j}\ell(\mathbf{W};\mathbf{x},y) &= (v_{\xi,j}-v_{y,j})\,1_{\{\ell(\mathbf{W};\{\mathbf{x},y\})>0\}}(\mathbf{x})\,\sigma'(\mathbf{w}_j^\top\mathbf{x})\mathbf{x}\\ &= \mathbf{0},\ \text{a.e.}\end{aligned}$$

where $\xi = \text{argmax}_{i\neq y}\, o(x;W)_i$.

Convergence Result

- use (partial) coarse gradient by replacing σ' with μ'

$$\tilde{\nabla}^{\mu}_{\mathbf{w}_j}\ell(\mathbf{W};\mathbf{x},y) := (v_{\xi,j} - v_{y,j})\, 1_{\{\ell(\mathbf{W};\{\mathbf{x},y\})>0\}}(\mathbf{x})\, \mu'(\mathbf{w}_j^\top \mathbf{x})\mathbf{x}.$$

- train the network by coarse gradient algorithm:

$$\mathbf{W}^{t+1} = \mathbf{W}^t - \eta\, \mathbb{E}_{\{\mathbf{x},y\}\sim\mathcal{D}} \tilde{\nabla}^{\mu}\ell(\mathbf{W}^t;\mathbf{x},y)$$

Convergence Result

- use (partial) coarse gradient by replacing σ' with μ'

$$\tilde{\nabla}^{\mu}_{\mathbf{w}_j}\ell(\mathbf{W};\mathbf{x},y) := (v_{\xi,j} - v_{y,j})\, 1_{\{\ell(\mathbf{W};\{\mathbf{x},y\})>0\}}(\mathbf{x})\, \mu'(\mathbf{w}_j^\top \mathbf{x})\mathbf{x}.$$

- train the network by coarse gradient algorithm:

$$\mathbf{W}^{t+1} = \mathbf{W}^t - \eta\, \mathbb{E}_{\{\mathbf{x},y\}\sim\mathcal{D}} \tilde{\nabla}^{\mu}\ell(\mathbf{W}^t;\mathbf{x},y)$$

Theorem (Long, Yin, Xin'21)

Suppose the data from different classes are located in orthogonal subspaces of $\mathbb{R}^d$. *Choose surrogate function* $\mu : \mathbb{R} \to \mathbb{R}$ *satisfying*

1. $\mu(x) = 0$ *for* $x \leq 0$.
2. $\mu'(x) \in [\delta, \tilde{\delta}]$ *for* $x > 0$ *with constants* $0 < \delta < \tilde{\delta} < \infty$.

Then $\lim_{t\to\infty} f(\mathbf{W}^t) = 0$, *leading to perfect classification.*

Full Quantization Algorithm

Algorithm 1 One iteration of Blended Coarse Gradient Descent

Input: mini-batch empirical loss function $f_t(\mathbf{w}, \boldsymbol{\alpha})$, blending parameter $\rho = 10^{-5}$, learning rate $\eta_{\mathbf{w}}^t$ for the weights $\mathbf{w}$, learning rate $\eta_{\boldsymbol{\alpha}}^t$ for the resolutions $\boldsymbol{\alpha}$ (one component per activation layer).

Do:

Evaluate the mini-batch coarse gradient $(\tilde{\nabla}_{\mathbf{w}} f_t, \tilde{\nabla}_{\boldsymbol{\alpha}} f_t)$ at $(\mathbf{w}_Q^t, \boldsymbol{\alpha}^t)$.

$\mathbf{w}^{t+1} = (1-\rho)\mathbf{w}^t + \rho \mathbf{w}_Q^t - \eta_{\mathbf{w}}^t \tilde{\nabla}_{\mathbf{w}} f_t(\mathbf{w}_Q^t, \boldsymbol{\alpha}^t)$ // blended gradient update for weights

$\boldsymbol{\alpha}^{t+1} = \boldsymbol{\alpha}^t - \eta_{\boldsymbol{\alpha}}^t \tilde{\nabla}_{\boldsymbol{\alpha}} f_t(\mathbf{w}_Q^t, \boldsymbol{\alpha}^t)$ // $\eta_{\boldsymbol{\alpha}}^t = 0.01 \cdot \eta_{\mathbf{w}}^t$

$\mathbf{w}_Q^{t+1} = \mathrm{proj}_{\mathcal{W}}(\mathbf{w}^{t+1})$ // quantize the weights

Remark: $\{\mathbf{w}^t\}$ is a sequence of FP-precision auxiliary parameters.

Outline

Experiments

Bitwidth	1W	2W	4W
Cifar-10			
ResNet20 (FP): 92.21 %, Teacher ResNet110			
label-free	89.88%	91.23%	92.19%
with supervision	90.56%	91.65%	92.43%
Cifar-100			
ResNet56 (FP): 72.96%, Teacher ResNet164			
label-free	72.78%	74.35%	74.90%
with supervision	73.35%	74.40%	75.31%
Tiny ImageNet			
ResNet18 (FP): 64.23%, Teacher ResNet34			
label-free FAQD	64.37%	65.05%	65.40%
FAQD with Supervision	65.13%	65.67%	65.92%

Table 1: Wight Quantization (1-bit, 2-bit, or 4-bit) with Feature Affinity Assisted Knowledge Distillation

CIFAR-10		
Pretrained ResNet20: 32W1A-91.89%, 32W4A-92.01%		
Model	1W1A	4W4A
ResNet20	89.70%	92.53%
CIFAR-100		
Pretrained ResNet56: 32W1A-70.96%, 32W4A-71.42%		
Model	1W1A	4W4A
ResNet56	68.18	73.53%
Tiny ImageNet		
Pretrained ResNet18: 32W1A-63.82%, 32W4A-64.15%		
Model	1W1A	4W4A
ResNet18	65.01	65.55%

Table 2: Full quantization on CIFAR-10, CIFAR-100 and Tiny ImageNet, with teacher networks.

Acknowledgement

Zhijian Li, Biao Yang, Jack Xin (UC Irvine)

Shuai Zhang, Jiancheng Lyu, Yingyong Qi (Qualcomm)

Ziang Long (Meta)

Stanley Osher (UCLA)

Partially supported by NSF grants DMS-1924935 and DMS-2208126.

References

P. Yin, J. Lyu, S. Zhang, S. Osher, Y. Qi, J. Xin, **Understanding Straight-Through Estimator in Training Activation Quantized Neural Nets**, *ICLR 2019*.

P. Yin, S. Zhang, J. Lyu, S. Osher, Y. Qi, J. Xin, **Blended Coarse Gradient Descent for Full Quantization of Deep Neural Networks**, *Research in the Mathematical Sciences*, 2019.

Z. Long, P. Yin, J. Xin, **Learning Quantized Neural Nets by Coarse Gradient Method for Non-linear Classification**, *Research in the Mathematical Sciences*, 2021.

Z. Li, B. Yang, P. Yin, Y. Qi, J. Xin, **Feature Affinity Assisted Knowledge Distillation and Quantization of Deep Neural Networks on Label-Free Data**, *arXiv:2302.10899*.

Thank you for your attention!