Two-Step Quantization for Low-bit Neural Networks

2018-10-15

Overview

- CVPR 2018
- Peisong Wang, Qinghao Hu, Yifan Zhang, Chunjie Zhang, Yang Liu, and Jian Cheng

- Two-Step量化：
  - 1 使用全精度weight量化activation，类似HWGQ但做了稀疏化
  - 2 抛弃之前全精度weight，量化transformation function（Low-bit）
  - 3 两步中间做了OTWA初始化
  - 4 使用Asymmetric Transformation Function Learning

Two-Step Quantization

$$
\begin{aligned}
\underset{\{W_l\}}{\text{minimize}} \quad & \mathcal{L}(Z_L, y) \\
\text{subject to} \quad & \hat{W}_l = Q_w(W_l) \\
& Z_l = \hat{W}_l \hat{A}_{l-1} \\
& A_l = \psi(Z_l) \\
& \hat{A}_l = Q_a(A_l), \ \text{for } l = 1, 2, \cdots L
\end{aligned} \tag{2}
$$

step2

step1

For the first step, **all weights are full-precision values** and we use the proposed **sparse quantization method to quantize all activations** into low-bit format. After the first step, only the leaned codes Al are kept while **the learned weights are discarded**, hence the name of **code learning.**

For the second step, we will learn the **transformation function** from Ai−1 to Ai, with low-bit constraints.

Two-Step Quantization

3.1. Sparse Quantization for Code Learning

Two-Step: 对重要的参数求解，其余参数稀疏化为0，量化函数变为：

优化方程变为：

$$Q_\epsilon^*(x) = \underset{Q_\epsilon}{\operatorname{argmin}} E_{x \sim \mathcal{N}(0,1), x > \epsilon}[(Q_\epsilon(x) - x)^2] \quad (6)$$

参数的稀疏化在batch normalization之后，使用给定的θ让分布满足：

$$\Phi(\epsilon) = P(x <= \epsilon) = \theta \quad (7)$$

where $\Phi(x)$ is the cumulative distribution function of standard normal distribution.

Two-Step Quantization

3.2. The Transformation Function Learning

$$\begin{aligned}&\underset{\Lambda,\hat{W}}{\text{minimize}} \quad \| Y - Q_\epsilon(\Lambda \hat{W} X) \|_F^2 \\ =&\underset{\{\alpha_i\},\{\hat{w}_i^T\}}{\text{minimize}} \quad \sum_i \| y_i^T - Q_\epsilon(\alpha_i \hat{w}_i^T X) \|_2^2\end{aligned} \tag{8}$$

non-linear least square regression problem

分解成多个子问题

$$\underset{\alpha,\hat{w}}{\text{minimize}} \quad \| y - Q_\epsilon(\alpha X^T \hat{w}) \|_2^2 \tag{9}$$

为求解子问题引入辅助变量z和惩罚系数λ（常量，趋近无穷时结果收敛于子问题的结果）将原函数松弛

$$\underset{\alpha,\hat{w},z}{\text{minimize}} \quad \| y - Q_\epsilon(z) \|_2^2 + \lambda \| z - \alpha X^T \hat{w} \|_2^2 \tag{10}$$

Two-Step Quantization

Solving α and ˆ w with z fixed.

Solving α and $\hat{w}$ with z fixed. When z is fixed, the optimization problem of 10 becomes to

$$\underset{\alpha,\hat{w}}{\text{minimize}} \quad J(\alpha,\hat{w}) = \| z - \alpha X^T \hat{w} \|_2^2 \tag{11}$$

By expanding Eq. 11, we have

$$J(\alpha,\hat{w}) = z^T z - 2\alpha z^T X^T \hat{w} + \alpha^2 \hat{w}^T X X^T \hat{w} \tag{12}$$

By setting $\partial J/\partial\alpha = 0$, the optimal value of α is given by

$$\alpha^* = \frac{z^T X^T \hat{w}}{\hat{w}^T X X^T \hat{w}} \tag{13}$$

Substituting α^* in equation 12, we can get

$$\hat{w}^* = \underset{\hat{w}}{\text{argmax}} \frac{(z^T X^T \hat{w})^2}{\hat{w}^T X X^T \hat{w}} \tag{14}$$

将z视为常量，则10变为11

11展开变成一二次函数

求J对于a的极小值点（求导）

带入a*，得到w*并使得右侧函数最大化，在这里n-bit
量化需要遍历2^mn个可行解，文章用了近似求解代替，
只需要查找m2^n个点（在这里n很小）

Two-Step Quantization

Solving z with α and ˆ w fixed.

$$\underset{z_i}{\text{minimize}} \quad (y_i - Q_\epsilon(z_i))^2 + \lambda(z_i - v_i)^2 \qquad (15)$$

w, a视为常量（上一轮已解出）， **one-dimensional problems**，10变为15

松弛Qx：

$$\tilde{Q}_\epsilon(x) = \begin{cases} M & x > M, \\ x & 0 < x \leq M, \\ 0 & x \leq 0 \end{cases} \qquad (16)$$

$$z_i^{(0)} = min(0, v_i) \qquad (17)$$

$$z_i^{(1)} = min(M, max(0, \frac{\lambda v_i + y_i}{1+\lambda})) \qquad (18)$$

$$z_i^{(2)} = max(M, v_i) \qquad (19)$$

按照Qx分三段讨论得出z

Two-Step Quantization

Initialization of α and ˆ w using Optimal Ternary Weights Approximation (OTWA).

在code learning之后用0 1 -1对weight和alpha进行2-bit初始化

$$\begin{aligned} \underset{\alpha, \hat{w}}{\text{minimize}} \quad & \| w - \alpha \hat{w} \|_2^2 \\ \text{subject to} \quad & \alpha > 0 \\ & \hat{w} \in \{-1, 0, +1\}^m. \end{aligned} \tag{20}$$

所以在这里保留绝对值最大的top r个weight进行sign得到w

$$\hat{w}_j = \begin{cases} sign(w_j) & abs(w_j) \text{ in top } r \text{ of } abs(w) \\ 0 & \text{others} \end{cases} \tag{22}$$

$$\begin{aligned} \alpha^* &= \frac{w^T \hat{w}}{\hat{w}^T \hat{w}} \\ \hat{w}^* &= \underset{\hat{w}}{\text{argmax}} \frac{(w^T \hat{w})^2}{\hat{w}^T \hat{w}} \end{aligned} \tag{21}$$

在这里可以确定a，w在22确定（这里是让w尽可能少但尽可能大）

Two-Step Quantization

Initialization of α and ˆ w using Optimal Ternary Weights Approximation (OTWA).

Algorithm 1 OTWA weight approximation

Input: weight matrix $W \in R^{k \times m}$
Output: $\hat{W} \in \{+1, 0, -1\}^{k \times m}$
Output: k floating point scaling factors $\{\alpha_i\}_{i=1}^{k}$

```
J(r) ← 0, for r = 1, ··· , m
for i = 1, ··· , k do
    v ← abs(w_i)
    sort v in decreasing order
    for r = 1, ··· , m do
        s ← Σ_{j=1}^{r} v(j)
        J(r) ← s^2/r
    end for
    r* ← argmax_r J(r)
    get ŵ_i* according to equation 22
    get α_i* according to equation 21
end for
```

Asymmetric Transformation Function Learning.

$$\underset{\Lambda, \hat{W}}{\text{minimize}} \quad \| Y - Q_{\epsilon}(\Lambda \hat{W} \tilde{X}) \|_F^2 \qquad (23)$$

为了消除因为各层之间独立量化造成的误差的累积，文章将transformation function learning里面的X换成了quantized network中上一层的activation

Two-Step Quantization

4. Experiments   On ImageNet

Table 1. The optimal thresholds and step values given different sparse ratios for 2-bit sparse quantization.

θ	50%	56.25%	62.5%	68.75%	75%
ϵ	0.00	0.16	0.32	0.49	0.68
Δ	0.5388	0.5914	0.6487	0.7139	0.7889

56.25%-75%

Table 2. Two-bit activation quantization comparison. Our sparse quantization method, denoted by SQ, is conducted under different sparsity.

Model	Sparsity (%)	Top-1 (%)	Top-5 (%)
AlexNet	50.00	58.5	81.5
HWGQ [2]	50.00	55.8	78.7
SQ-1	56.25	58.2	80.7
SQ-2	62.50	59.0	81.3
SQ-3	68.75	58.9	80.8
SQ-4	75.00	57.9	79.8

4.2. Sparse Quantization Results

Table 3. The accuracy of our two-bit activation and ternary weight quantization models before and after fine-tuning. Results are reported under different activation sparsity.

Model	Before Fine-tune		After Fine-tune	
	Top-1	Top-5	Top-1	Top-5
TFL-SQ-1	52.7	76.6	-	-
TFL-SQ-2	55.1	78.4	58.0	80.5
TFL-SQ-3	54.7	78.1	-	-
TFL-SQ-4	54.3	77.4	56.7	79.0

Two-Step Quantization

4.3. Transformation Function Learning Results

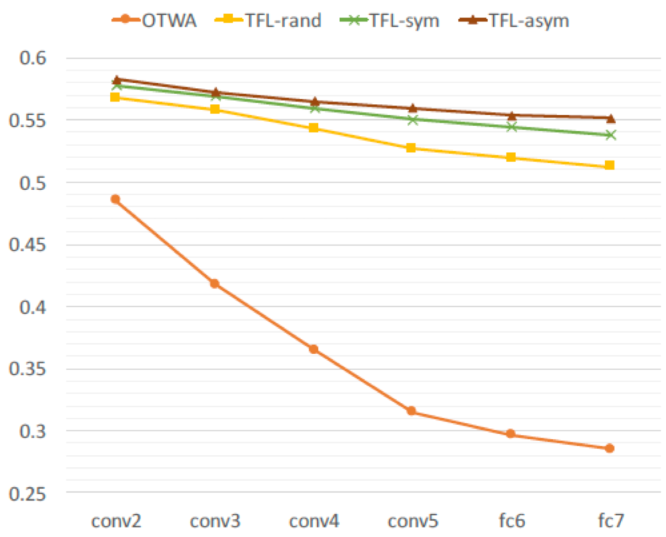

Figure 1. Top-1 accuracy after each layer is quantized for different transformation function learing settings. The results are conducted based on the SQ-2 AlexNet mode without fune-tuning.

- **OTWA**: Weight ternarization using OTWA;
- **TFL-rand**: Asymmetric transformation function learning initialized by random ternary variables;
- **TFL-sym**: Symmetric transformation function learning, initialized by OTWA;
- **TFL-asym**: Asymmetric transformation function learning, initialized by OTWA;

Two-Step Quantization

4.4. Comparison with the stateoftheart

Table 4. Comparison with the state-of-the-art low-bit quantization methods on AlexNet. The accuracy gap to the full-precision model is also reported.

Model	Top-1	Top-5	Top-1 gap	Top-5 gap
AlexNet[2]	58.5	81.5	0	0
XNOR[24]	44.2	69.2	-12.4	-12.3
BNN[30]	46.6	71.1	-11.9	-10.4
DOREFA[38]	47.7	-	-8.2	-
HWGQ[2]	52.7	76.3	-5.8	-5.2
TSQ (ours)	58.0	80.5	-0.5	-1.0

Table 5. Comparison between our quantized VGG-16 model and the full-precison counterparts.

Model	Top-1	Top-5	Top-1 gap	Top-5 gap
VGG-16 [29]	71.1	89.9	0	0
VGG-16-BN	69.6	89.6	-1.5	-0.3
TSQ (ours)	69.1	89.2	-2.0	-0.7

4.5. Results on CIFAR10

Table 6. Comparison with the state-of-the-art low-bit quantization methods on CIFAR-10. The bit-width for activations and weights are given.

Activation	Weights	Method	error (%)
Full	Full	VGG-Small	6.82
Full	Binary	BinaryConnect [6]	8.27
Full	Ternary	TWN [23]	7.44
Binary	Binary	BNN [14]	10.15
2-bit	Binary	HWGQ [2]	7.49
2-bit	Ternary	TSQ (ours)	6.51