# Towards Accurate Binary Convolutional Neural Network

2018-11-05

# Overview

- NIPS 2017

- 做了weight和activation两者的二值化

- 1. approximating full-precision weights with the linear combination of multiple binary weight bases;
- 2. employing multiple binary activations to alleviate information loss.

# ABC-Net (A ccurate-B inary-C onvolutional)

- Weight approximation

$$\min_{\boldsymbol{\alpha},\boldsymbol{B}} J(\boldsymbol{\alpha},\boldsymbol{B}) = ||\boldsymbol{w} - \boldsymbol{B}\boldsymbol{\alpha}||^2, \quad \text{s.t. } \boldsymbol{B}_{ij} \in \{-1,+1\}, \tag{1}$$

where $\boldsymbol{B} = [\text{vec}(\boldsymbol{B}_1), \text{vec}(\boldsymbol{B}_2), \cdots,$ where $\bar{\boldsymbol{W}} = \boldsymbol{W} - \text{mean}(\boldsymbol{W})$ $= [\alpha_1, \alpha_2, \cdots, \alpha_M]^{\text{T}}$.

$$\boldsymbol{B}_i = F_{u_i}(\boldsymbol{W}) := \text{sign}(\bar{\boldsymbol{W}} + u_i \text{std}(\boldsymbol{W})), i = 1, 2, \cdots, M, \tag{2}$$

$$\min_{\boldsymbol{\alpha}} J(\boldsymbol{\alpha}) = ||\boldsymbol{w} - \boldsymbol{B}\boldsymbol{\alpha}||^2, \tag{3}$$

- 做weight的近似组合，使用alpha逼近

# ABC-Net (A ccurate-B inary-C onvolutional)

Forward: $\boldsymbol{B}_1, \boldsymbol{B}_2, \cdots, \boldsymbol{B}_M = F_{u_1}(\boldsymbol{W}), F_{u_2}(\boldsymbol{W}), \cdots, F_{u_M}(\boldsymbol{W}),$ (4)

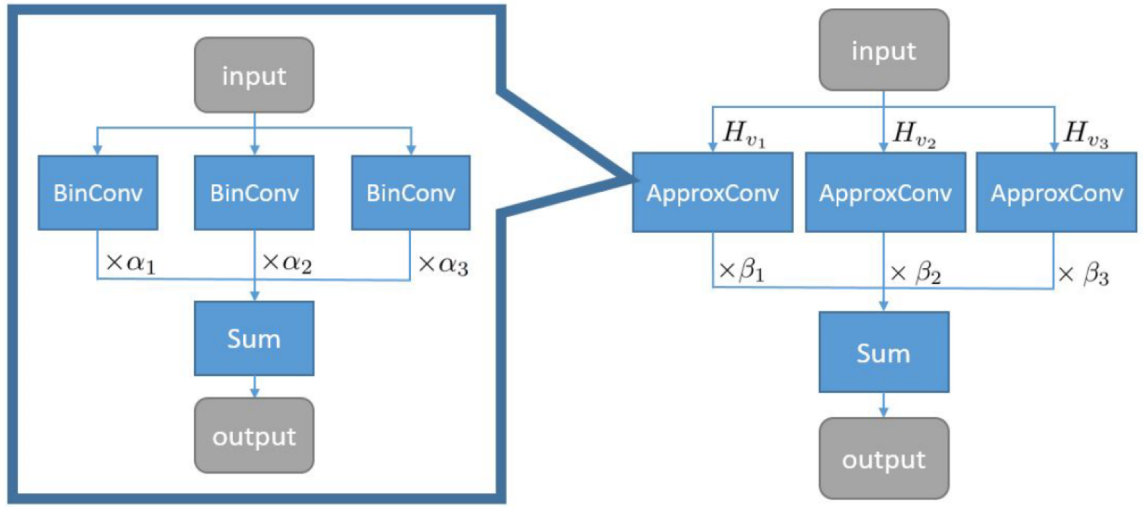
Solve (3) for $\boldsymbol{\alpha}$, (5)

$$\boldsymbol{O} = \sum_{m=1}^{M} \alpha_m \mathrm{Conv}(\boldsymbol{B}_m, \boldsymbol{A}). \quad (6)$$

Backward: $$\frac{\partial c}{\partial \boldsymbol{W}} = \frac{\partial c}{\partial \boldsymbol{O}} \left( \sum_{m=1}^{M} \alpha_m \frac{\partial \boldsymbol{O}}{\partial \boldsymbol{B}_m} \frac{\partial \boldsymbol{B}_m}{\partial \boldsymbol{W}} \right) \overset{\mathrm{STE}}{=} \frac{\partial c}{\partial \boldsymbol{O}} \left( \sum_{m=1}^{M} \alpha_m \frac{\partial \boldsymbol{O}}{\partial \boldsymbol{B}_m} \right) = \sum_{m=1}^{M} \alpha_m \frac{\partial c}{\partial \boldsymbol{B}_m}. \quad (7)$$

**Weight部分**

For/Back-ward



# Experiment

Table 2: Prediction accuracy (Top-1/Top-5) for ImageNet with different choices of $M$ and $N$ in a ABC-Net (approximate weights as a whole). "res18", "res34" and "res50" are short for Resnet-18, Resnet-34 and Resnet-50 network topology respectively. $M$ and $N$ refer to the number of weight bases and activations respectively.

| Network | $M$-weight base | $N$-activation base | Top-1 | Top-5 | Top-1 gap | Top-5 gap |
|---|---|---|---|---|---|---|
| res18 | 1 | 1 | 42.7% | 67.6% | 26.6% | 21.6% |
| res18 | 3 | 1 | 49.1% | 73.8% | 20.2% | 15.4% |
| res18 | 3 | 3 | 61.0% | 83.2% | 8.3% | 6.0% |
| res18 | 3 | 5 | 63.1% | 84.8% | 6.2% | 4.4% |
| res18 | 5 | 1 | 54.1% | 78.1% | 15.2% | 11.1% |
| res18 | 5 | 3 | 62.5% | 84.2% | 6.8% | 5.0% |
| res18 | 5 | 5 | 65.0% | 85.9% | **4.3%** | **3.3%** |
| res18 | Full Precision | | 69.3% | 89.2% | - | - |
| res34 | 1 | 1 | 52.4% | 76.5% | 20.9% | 14.8% |
| res34 | 3 | 3 | 66.7% | 87.4% | 6.6% | 3.9% |
| res34 | 5 | 5 | 68.4% | 88.2% | **4.9%** | **3.1%** |
| res34 | Full Precision | | 73.3% | 91.3% | - | - |
| res50 | 5 | 5 | 70.1% | 89.7% | **6.0%** | **3.1%** |
| res50 | Full Precision | | 76.1% | 92.8% | - | - |

# Experiment

Table 1: Top-1 (left) and Top-5 (right) accuracy of ABC-Net on ImageNet, using full-precision activation and different choices of the number of binary weight bases $M$.

| | BWN | $M = 1$ | $M = 2$ | $M = 3$ | $M = 5$ | FP |
|---|---|---|---|---|---|---|
| Top-1 | 60.8% | 62.8% | 63.7% | 66.2% | 68.3% | 69.3% |
| Top-5 | 83.0% | 84.4% | 85.2% | 86.7% | 87.9% | 89.2% |

Table 3: Classification test accuracy of CNNs trained on ImageNet with Resnet-18 network topology. 'W' and 'A' refer to the weight and activation bitwidth respectively.

| Model | W | A | Top-1 | Top-5 |
|---|---|---|---|---|
| Full-Precision Resnet-18 [full-precision weights and activation] | 32 | 32 | 69.3% | 89.2% |
| BWN [full-precision activation] Rastegari et al. [2016] | 1 | 32 | 60.8% | 83.0% |
| DoReFa-Net [1-bit weight and 4-bit activation] Zhou et al. [2016] | 1 | 4 | 59.2% | 81.5% |
| XNOR-Net [binary weight and activation] Rastegari et al. [2016] | 1 | 1 | 51.2% | 73.2% |
| BNN [binary weight and activation] Courbariaux et al. [2016] | 1 | 1 | 42.2% | 67.1% |
| ABC-Net [5 binary weight bases, 5 binary activations] | 1 | 1 | 65.0% | 85.9% |
| ABC-Net [5 binary weight bases, full-precision activations] | 1 | 32 | 68.3% | 87.9% |

# Future Work

这一部分比较重要，**硬件友好的运算优化**

### 5.3 Further computation reduction in run-time

On specialized hardware, the following operations in our scheme can be integrated with other operations in run-time and further reduce the computation requirement.

(1) Shift operations. The existence of shift parameters seem to require extra additions/subtractions (see (2) and (8)). However, the binarization operation with a shift parameter can be implemented as a comparator where the shift parameter is the number for comparison, e.g., $H_v(\boldsymbol{R}) = \begin{cases} \mathbf{1}, & \boldsymbol{R} \geq 0.5 - v; \\ \mathbf{-1}, & \boldsymbol{R} < 0.5 - v. \end{cases}$ ($0.5 - v$ is a constant), so no extra additions/subtractions are involved.

(2) Batch normalization. In run-time, a batch normalization is simply an affine function, say, $\mathrm{BN}(\boldsymbol{R}) = a\boldsymbol{R} + b$, whose scale and shift parameters $a, b$ are fixed and can be integrated with $v_n$'s. More specifically, a batch normalization can be integrated into a binarization operation as follow: $H_v(\mathrm{BN}(\boldsymbol{R})) = \begin{cases} \mathbf{1}, & a\boldsymbol{R} + b \geq 0.5 - v; \\ \mathbf{-1}, & a\boldsymbol{R} + b < 0.5 - v. \end{cases} = \begin{cases} \mathbf{1}, & \boldsymbol{R} \geq (0.5 - v - b)/a; \\ \mathbf{-1}, & \boldsymbol{R} < (0.5 - v - b)/a. \end{cases}$ Therefore, there will be no extra cost for the batch normalization.