True Gradient-Based Training of Deep Binary Activated Neural Networks via Continuous Binarization

Charbel Sakr*,#, Jungwook Choi†, Zhuo Wang†, Kailash Gopalakrishnan†, Naresh Shanbhag*

* University of Illinois at Urbana-Champaign
† IBM T.J. Watson Research Center
# work done at IBM

Acknowledgment:
• This work was supported in part by Systems on Nanoscale Information fabriCs (SONIC), one of the six SRC STARnet Centers, sponsored by MARCO and DARPA.
• This work is supported in part by IBM-ILLINOIS Center for Cognitive Computing Systems Research (C3SR) - a research collaboration as part of the IBM AI Horizons Network.
The Binarization Problem

- Binarization of Neural Networks is an promising direction to complexity reduction.

- Binary activation functions are unfortunately non-continuous.

- Training networks with binary activations cannot use gradient-based learning.
Current Approach

• Treat binary activations as stochastic units (Bengio, 2013).
• Use a straight through estimator (STE) of the gradient.
• Was shown to enable training of binary networks (Hubara et al., Rastegari et al., etc...).
• Often comes at the cost of accuracy loss compared to floating point.
Proposed Method

- Start with a *clipping* activation function and *learn* it to become binary.

\[ \text{actFn}(x) = \text{Clip} \left( \frac{x}{m} + \frac{\alpha}{2}, 0, \alpha \right) \]

- Smaller \( m \) means steeper slope. The activation function naturally approaches a binarization function.
Caveat: Bottleneck Effect

- Activations cannot be learned simultaneously due to the *bottleneck effect* in backward computations.

- Hence we learn slopes *one layer at a time.*
Justification via induction

Step 1: The base case, a baseline network with clipping activation function

The arrows mean layer-wise operation of the input
Justification via induction

Step 1: The base case, a baseline network with clipping activation function

Step 2: Replace first layer activation with binarization and stop learning first layer

The arrows mean layer-wise operation of the input
Justification via induction

Step 1: The base case, a baseline network with clipping activation function

Step 2: Replace first layer activation with binarization and stop learning first layer

Equivalently, we have a new network with binary inputs

The arrows mean layer-wise operation of the input
Justification via induction

Step 1: The base case, a baseline network with clipping activation function

Input features

```
| Input features | Clipping | Clipping | Clipping | Output |
```

Step 2: Replace first layer activation with binarization and stop learning first layer

```
| Binary inputs | Clipping | Clipping | Output |
```

Step $L - 1$: Binary inputs – Even shorter network

```
| Binary inputs | Clipping | Output |
```

The arrows mean layer-wise operation of the input
Justification via induction

Step 1: The base case, a baseline network with clipping activation function

Step 2: Replace first layer activation with binarization and stop learning first layer

Step $L - 1$: Binary inputs – Even shorter network

Step $L$: Binary inputs – Very short network

The arrows mean layer-wise operation of the input
Justification via induction

Step 1: The base case, a baseline network with clipping activation function

Step 2: Replace first layer activation with binarization and stop learning first layer

Step $L-1$: Binary inputs – Even shorter network

Step $L$: Binary inputs – Very short network

The arrows mean layer-wise operation of the input
Analysis

• The mean squared error of approximating the PCF by and SBAF decreases linearly in $m$

\[ MSE = k \int_{-\frac{m \alpha}{2}}^{\frac{m \alpha}{2}} \left( \frac{x}{m} + \frac{\alpha}{2} - \alpha \cdot 1_{x>0} \right)^2 dx = c \cdot m \]

Hence, perturbation magnitude decreases with $m$
Analysis

• The mean squared error of approximating the PCF by and SBAF decreases linearly in $m$

\[ MSE = k \int_{-\frac{m \alpha}{2}}^{-\frac{m \alpha}{2}} \left( \frac{x}{m} + \frac{\alpha}{2} - \alpha \cdot 1_{x>0} \right)^2 dx = c \cdot m \]

Hence, perturbation magnitude decreases with $m$

• There is no mismatch when using the SBAF or the PCF provided a bounded perturbation magnitude

\[ \|q_a\| < \min_{j=1...M, \; j \neq i} \frac{f_i(a_o) - f_j(a_o)}{\|\nabla_{a_o} f_j(a_o) - \nabla_{a_o} f_i(a_o)\|} \]

Backtracking: small $m \rightarrow$ small perturbation $\rightarrow$ less mismatch
Analysis

• The mean squared error of approximating the PCF by and SBAF decreases linearly in $m$

$$MSE = k \int_{-m\alpha/2}^{-m\alpha} \left( \frac{x}{m} + \frac{\alpha}{2} - \alpha \cdot 1_{x>0} \right)^2 dx = c \cdot m$$

Hence, perturbation magnitude decreases with $m$

• There is no mismatch when using the SBAF or the PCF provided a bounded perturbation magnitude

$$\|q_a\| < \min_{j=1...M, j \neq i} \frac{f_i(a_o) - f_j(a_o)}{\|\nabla_{a_o} f_j(a_o) - \nabla_{a_o} f_i(a_o)\|}$$

Backtracking: small $m \rightarrow$ small perturbation $\rightarrow$ less mismatch

How to make $m$ small?
Regularization

• We add a regularization term $\lambda$ to $m$ when learning it ($L_2$ and/or $L_1$).
• The optimal $\lambda$ value is usually found by tuning (common problem with regularization).
• We have some empirical guidelines from our experiments:
  – Layer type 1: fully connected
  – Layer type 2: convolution preceding convolution
  – Layer type 3: convolution preceding pooling
  – We have observed that the following is a good strategy
    • $\lambda_1 > \lambda_2 > \lambda_3$
Convergence

• Blue curve: obtained network by binarizing up to layer $l$
• Orange curve: completely binary activated network
• As training evolves, the network becomes completely binary and the two curves meet
• The accuracy is very close to the initial one which is the baseline
Comparison with STE

<table>
<thead>
<tr>
<th></th>
<th>MNIST</th>
<th>CIFAR-10</th>
<th>SVHN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full-precision Baseline</td>
<td>1.45%</td>
<td>9.04%</td>
<td>2.53%</td>
</tr>
<tr>
<td>Binarization via STE</td>
<td>1.54%</td>
<td>14.80%</td>
<td>4.05%</td>
</tr>
<tr>
<td>Continuous Binarization</td>
<td>1.27%</td>
<td>10.41%</td>
<td>3.20%</td>
</tr>
</tbody>
</table>

*summary of test errors*

- Our method consistently outperforms binarization via STE
Conclusion & Future Work

• We presented a novel method for binarizing the activations of deep neural networks
  → The method leverages true gradient based learning
  Consequently, the obtained results consistently outperform conventional binarization via STE

• Future work
  → Experimentations on larger datasets
  → Combining the proposed activation binarization to weight binarization
  → Extension to multi-bit activations
Thank you!