Deep Learning with Low Precision by Half-wave Gaussian Quantization

Zhaowei Cai† Xiaodong He‡ Jian Sun§ Nuno Vasconcelos†

† University of California San Diego ‡Microsoft Research §Megvii Inc.

I. Introduction

- Motivations:
  - The state-of-the-art neural networks have high computational complexity (billions of FLOPs) and large model size (hundreds of MB).
  - Binary neural network (BNN) theoretically enables 64× model size reduction and 64× computational speeds-up.
  - BNN has significant accuracy drop from the full-precision neural network.
- Contributions:
  - An half-wave Gaussian quantization (HWGQ) is proposed as forward approximation of the effective non-linear ReLU function.
  - HWGQ has efficient implementation, by exploiting the statistics of of network activations and batch normalization.
  - To overcome the problem of gradient mismatch, due to the use of different forward and backward functions, several effective piece-wise backward approximators are investigated and they successfully suppress the mismatch.
  - HWGQ-Net achieves much closer performance to full precision networks, such as AlexNet, ResNet, GoogLeNet and VGG-Net, than previously available low-precision networks, with 1-bit binary weights and 2-bit quantized activations.

II. Binary Neural Network

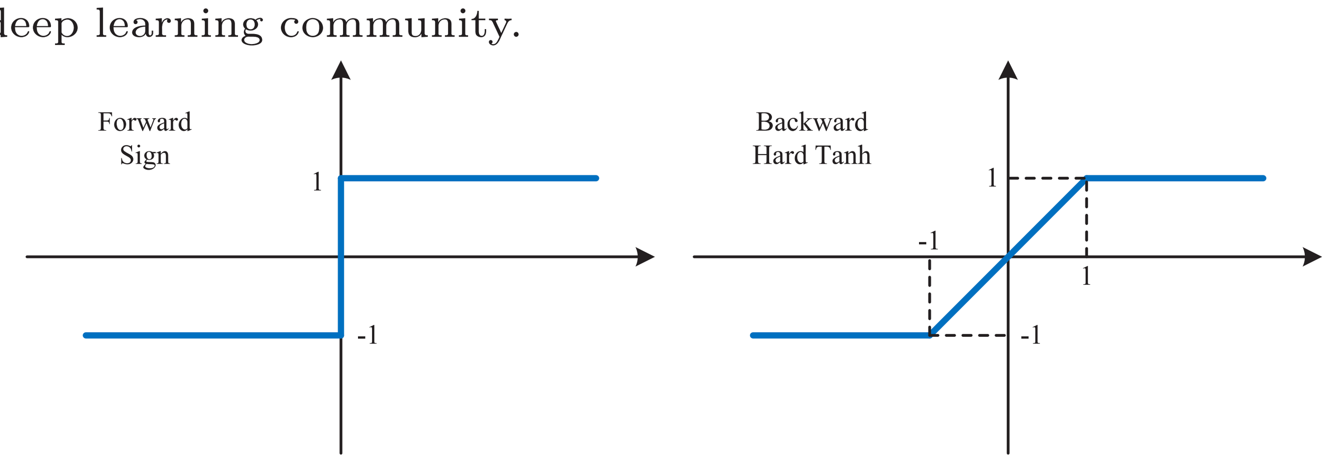
- Goals
  - unit dot-product

$$z = g(\mathbf{w}^T\mathbf{x})$$

  - large memory footprint required to store weights $\mathbf{w}$.
  - high computational complexity required to compute large numbers of full-precision dot-products $\mathbf{w}^T\mathbf{x}$.
- Weight Binarization
  - multiplication-free convolution

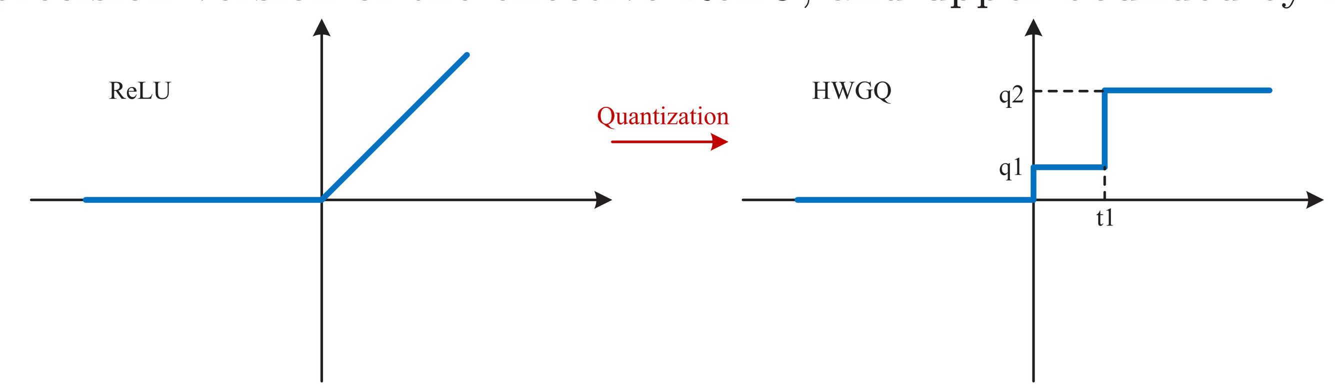
$$\mathbf{I} * \mathbf{W} \approx \alpha(\mathbf{I} \oplus \mathbf{B})$$

  where $\mathbf{B}^* = sign(\mathbf{W})$ and $\alpha^* = \frac{1}{cwh}\|\mathbf{W}\|_1$
  - tremendous reduction in the memory footprint of the model, but the problem of computational complexity is not fully solved, since $\mathbf{I}$ is still with full-precision.
- Binary Activation Quantization
  - substantial complexity reductions can be obtained by the binarization of $\mathbf{I}$, by implementing the dot products with logical and bit-counting operations.
  - it is a much harder problem than weight binarization, and the main reason responsible for accuracy drop of binary neural network.
  - *sign* function is non-differentiable, and *hard-tanh* function is used as backward approximation such that the gradients can be back-propagated.
  - it is upper-bounded by *hard-tanh* function, which is close to abandoned in the deep learning community.



III. HWGQ Network

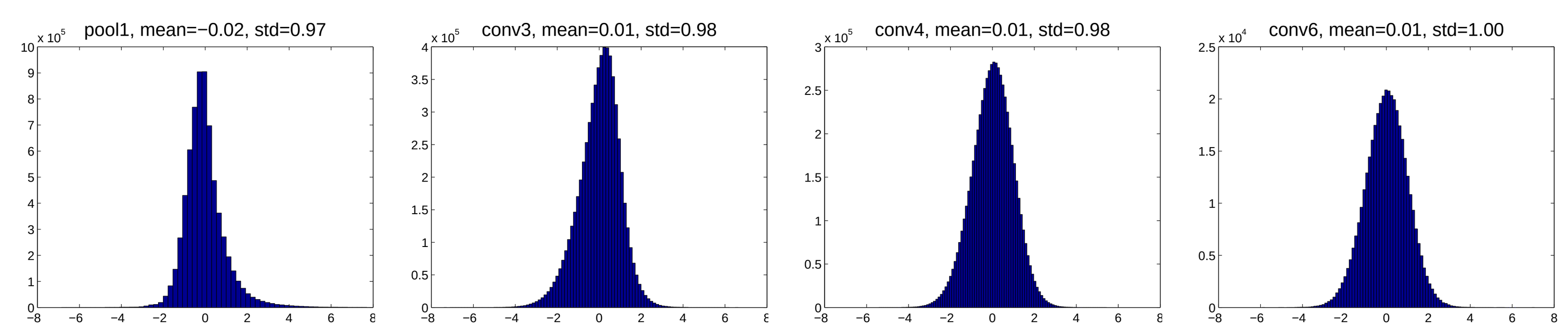
- Half-wave Gaussian Quantization
  - low-precision version of the effective ReLU, and upper-bounded by it.



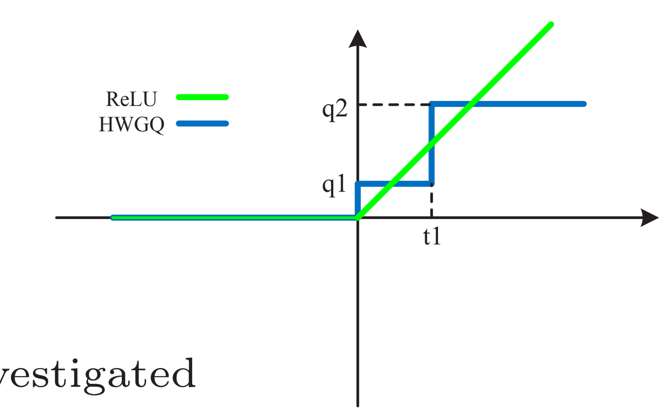
- Efficient Quantization Optimization
  - the optimal quantizer depends on the data distribution

$$Q^*(x) = \arg\min_Q \int p(x)(Q(x) - x)^2 dx$$

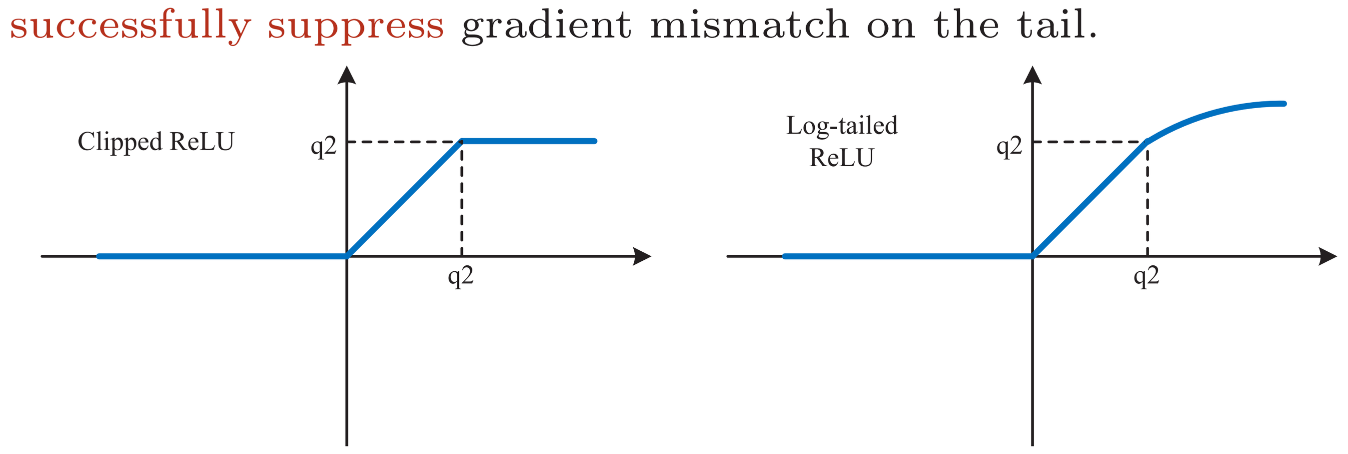
  - dot-product distributions are approximately Gaussians.
    - quantization optimization is an iterative algorithm.
    - the optimal quantizer varies across units, layers and back-propagation iterations.
  - Batch Normalization
    - all Gaussians have zero mean and unit variance.
    - the optimal quantization parameters are universal.
    - apply the quantization algorithm only once, and use the optimal quantizer parameters to parametrize a single HWGQ for all layers.



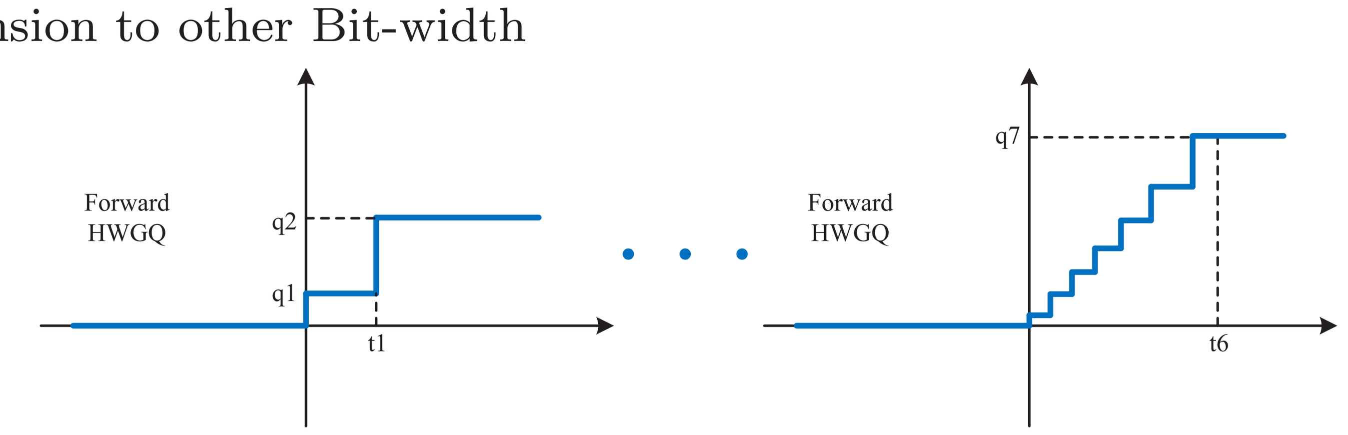
- Gradient Mismatch
  - gradient mismatch between forward HWGQ and backward ReLU functions, unbounded on the tail.
  - it drives the learning unstable, especially for deeper networks.



  - alternative backward approximations are investigated
    - successfully suppress gradient mismatch on the tail.

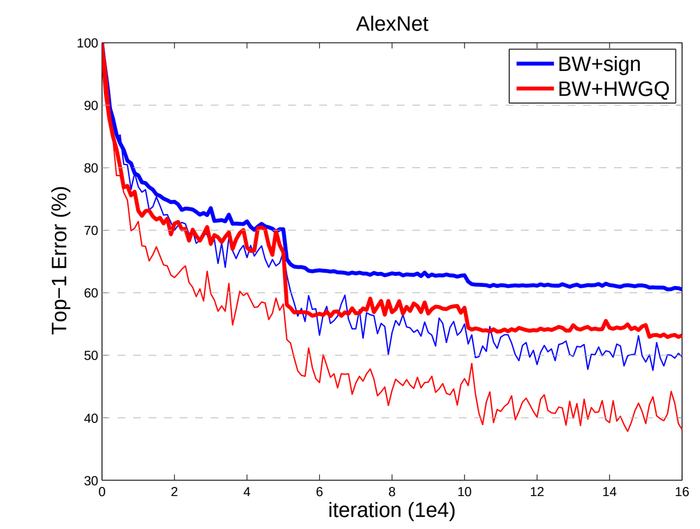


- Extension to other Bit-width

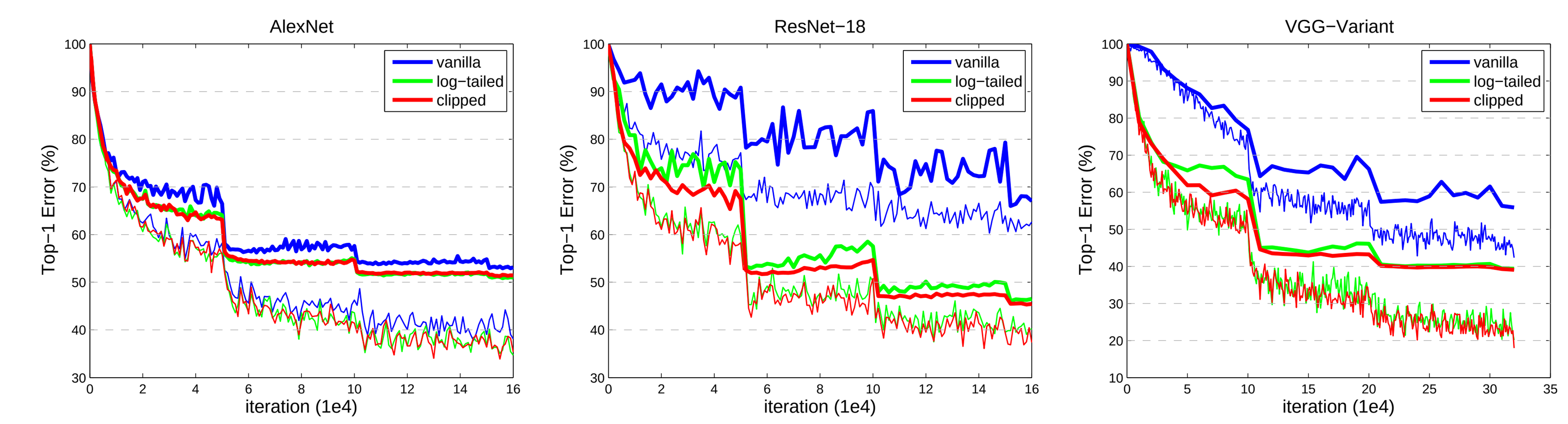


IV. Experimental Results

- Low-precision Activation Quantization Comparison
  - $sign(x)$ is not a good choice for forward quantization function, but HWGQ has a higher upper bound.
  - the much lower training error of HWGQ suggests that it enables a much better approximation of the full precision activations than $sign(x)$.
  - the gradient mismatch makes the optimization somewhat instable.



- Backward Approximations Comparison
  - vanilla ReLU creates instable optimization for all networks, more serious for deeper networks.
  - both of the clipped and log-tailed ReLU enable more stable learning, and in general the clipped ReLU is slightly better than the log-tailed ReLU.



- Comparison with the state-of-the-art
  - closer performance to the full-precision counterparts than the state-of-the-art low-precision networks.
  - enables theoretically 64× model size reduction and 32× speeds-up.

Model	AlexNet			ResNet-18	
	XNOR	DOREFA	HWGQ	XNOR	HWGQ
Top-1	44.2	47.7	52.7	51.2	59.6
Top-5	69.2	-	76.3	73.2	82.2
Top-1 gap	-12.4	-8.2	-5.8	-18.1	-7.7

- Good generalization on various popular networks
  - good performance regardless of model size, depth, complexity, etc.
  - no hyper-parameter needs to be tuned.

Model		Reference	Full	HWGQ
AlexNet	Top-1	57.1	58.5	52.7
	Top-5	80.2	81.5	76.3
ResNet-18	Top-1	69.6	67.3	59.6
	Top-5	89.2	87.9	82.2
ResNet-34	Top-1	73.3	69.4	64.3
	Top-5	91.3	89.1	85.7
ResNet-50	Top-1	76.0	71.5	64.6
	Top-5	93.0	90.5	85.9
VGG-Variant	Top-1	-	69.8	64.1
	Top-5	-	89.3	85.6
GoogLeNet	Top-1	68.7	71.4	63.0
	Top-5	88.9	90.5	84.9

- Reproducible research
  - https://github.com/zhaoweicai/hwgq

Computer Vision and Pattern Recognition, 2017