Training Strategies for Binary/Ternary Neural Networks

Robin Kiemes | Qing Wang | Braden Refalo

1. Introduction

- Deep neural networks need gigabytes of memory and billions of FLOPs [1,2]
- Binary {−1, +1} and Ternary {-1, 0, +1} weights replaces FP multiply with XNOR+popcount [3]
- Result: huge memory & compute savings for edge/resource-constrained devices [4]
- Challenge: quantization functions are non-differentiable (gradient is zero a.e.) [3,5]
- Quantized nets are efficient but hard to train gradient approximations are needed; different training strategies are employed [5,6]

2. Research Question

How do STE variants, weight clipping, and batch normalization affect training stability, convergence, and final accuracy of weight-quantized networks?

3. Key Concepts

Straight Through Estimators (STEs) [5]

- Quantization function Q(w) has zero gradient everywhere [4]
- STE enables backpropagation by substituting the zero gradient with a surrogate $h(w)$

$$\frac{\partial \mathcal{L}}{\partial w} = \frac{\partial \mathcal{L}}{\partial \hat{w}} \cdot \frac{\partial \hat{w}}{\partial w} = \frac{\partial \mathcal{L}}{\partial \hat{w}} \cdot \underbrace{\frac{d}{dw} Q(w)}_{=0 \text{ a.e.}} \qquad \frac{\partial \mathcal{L}}{\partial w} \approx \frac{\partial \mathcal{L}}{\partial \hat{w}} \cdot h(w)$$

- Where $\mathcal{L}$ is our loss objective, w is our latent weights (full-precision, and $\hat{w}$ is our quantized weights.
- **Eleven** STE variants were tested; different surrogate gradient effects were measured.

Weight Clipping

- Constrains latent weights to [−c, c] after each update, Keeps latent weights near quantization boundaries.

Batch normalization (BN)

- Standardizes activations before quantization [7]
- Placement (pre/post) changes which distribution the quantizer sees

4. Experimental Setup

Four ablation studies, each isolating one training design choice.

- Model: ResNet-20, ~270K params
- Dataset: CIFAR-10 (60K images, 10 classes)
- Quantization Scheme: weights only, TTQ (ternary) [4], XNOR-Net (binary) [3]
- Optimizer: SGD + cosine annealing, 160 epochs, 5 seeds each
- FP32 baseline: 91.61%, serves as our accuracy ceiling

EXP 1 — **STE Ablation** — 11 variants × 5 seeds

EXP 2 — **Weight Clipping** — f ∈ {0.75, 1.0, 2.0, 4.0, ∞}

EXP 3 — **BN Placement** — Pre / Post / None

EXP 4 — **STE Instability** — Fine-tune from checkpoint

5. STE Ablation

Does the choice of STE significantly affect final accuracy?

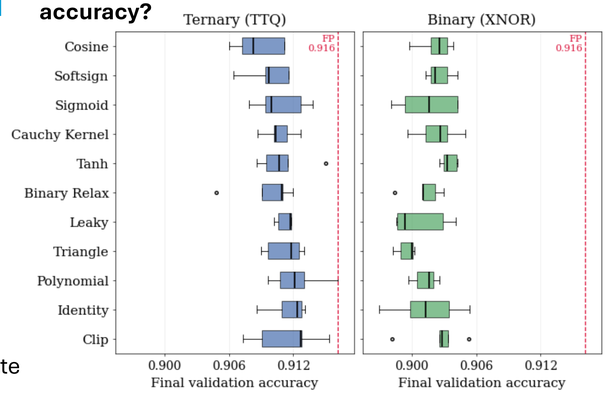

Figure 1: Final-epoch validation accuracy for 11 STE variants under TTQ and XNOR quantization on ResNet-20/CIFAR-10.

- Results (Figure 1) do not support that STE choice influences final accuracy
- Spread is only 0.35 pp (ternary) and 0.40 pp (binary), no variant is statistically significant
- STE choice is a secondary effect of the quantization scheme

6. STE Instability

Does STE choice cause instability when fine-tuning a pre-trained model?

MOST STABLE (TTQ) — **polynomial_ste** — Std = 3.17 · Total variation = 105

LEAST STABLE (TTQ) — **identity_ste** — Std = 3.98 · Total variation = 165

MOST STABLE (XNOR) — **sigmoid_ste** — Std = 3.58 · Total variation = 187

LEAST STABLE (XNOR) — **leaky_ste** — Std = 4.66 · Total variation = 255

- Results show varying standard deviation and total variation based on STE choice
- Leaky STE is most volatile for binary; Sigmoid STE is smoothest.
- Identity STE is most volatile for ternary; Polynomial STE is smoothest.
- Smooth STEs suppress gradients near discrete states
- Smooth STEs are more stable near convergence

7. Weight clipping

Does constraining latent weights improve quantized accuracy?

- f is a multiplier on the weight scale α — e.g. f=4.0 clips latent weights to [−4α, +4α]
- Results (Figure 2) show increasing validation accuracy as clipping boundary increases.
- f=4.0 is optimal and achieves 0.5 pp and 0.26 pp increase in accuracy on ternary and binary over the no clip baseline , respectively.

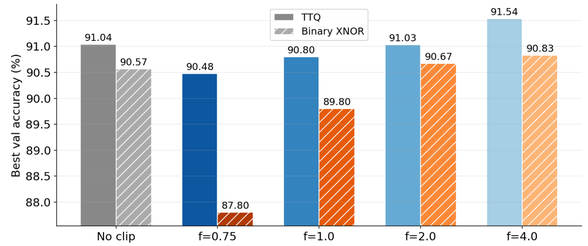

Figure 2: Peak validation accuracy across weight clipping values for binary (XNOR) and ternary (TTQ) networks on ResNet-20/CIFAR-10.

8. Batch normalization

Is batch normalization needed, and does placement (pre/post) matter?

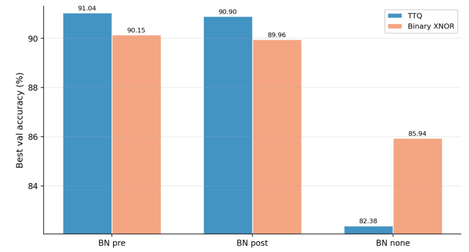

Figure 3: Peak validation accuracy across batch normalization configurations for binary (XNOR) and ternary (TTQ) networks on ResNet-20/CIFAR-10.

- Results show removing BN drops accuracy by up to 8.66 pp. Pre-BN beats Post-BN by 0.14–0.19 pp centring before quantization reduces skew [7]

9. Conclusion

- **STE choice**: minor impact - 0.35 pp spread, no significance. Match STE to quantizer structure [5,9]
- **Weight clipping** [3]: f=4.0 optimal. Prevents drift; too tight causes underfitting (−2.77 pp at f=0.75)
- **Batch normalization**: essential. Removal costs up to 8.66 pp. Pre-BN placement is best
- **Coherence matters**: align gradient approx. [10], weight distribution [5], and normalization

Combining best settings from all 4 experiments (5 seeds, 160 epochs):

Model	STE	Clip f	BN	Acc.	Gain
FP32 ceiling	—	—	—	**91.61%**	—
TTQ baseline [8]	identity	none	post	91.12%	—
Optimal TTQ ★	**polynomial**	**4.0**	**pre**	**91.52%**	**+0.40 pp**
XNOR baseline [6]	identity	none	post	90.14%	—
Optimal XNOR ★	**tanh**	**4.0**	**pre**	**90.78%**	**+0.64 pp**

TERNARY (TTQ) — **91.52%** — polynomial · f=4.0 · Pre-BN — +0.40 pp · std: 0.19→0.10

BINARY (XNOR) — **90.78%** — tanh · f=4.0 · Pre-BN — +0.64 pp · std: 0.32→0.28

[1] (Ren et al., 2015), [2] (Collobert & Weston, 2008), [3] (Rastegari et al., 2016), [4] (Zhu et al., 2016), [5] (Hubara et al., 2016), [6] (Courbariaux et al., 2015), [7] (Sari & Partovi Nia, 2020), [8] (Wu et al., 2023), [9] (Yin et al., 2019), [10] (Bengio et al., 2013)

Code: github.com/robin-kronos/research-project-efficient-learning