Learning Accurate Low-bit Deep Neural Networks with Stochastic Quantization

Yinpeng Dong[1], Renkun Ni[2], Jianguo Li[3], Yurong Chen[3], Jun Zhu[1], Hang Su[1]

[1] Department of CST, Tsinghua University

[2] University of Virginia

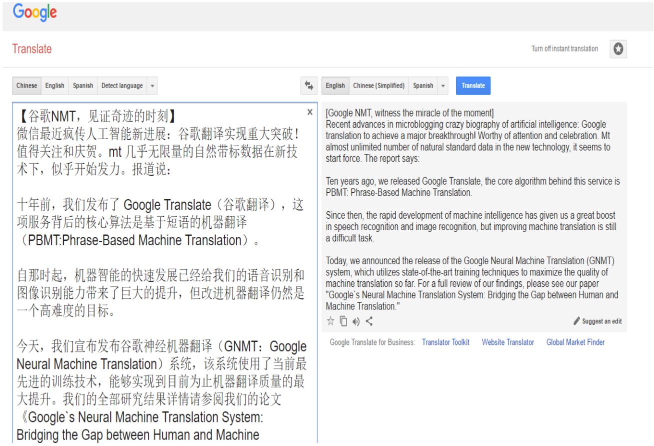
[3] Intel Labs China

Deep Learning is Everywhere

Self-Driving

Alpha Go

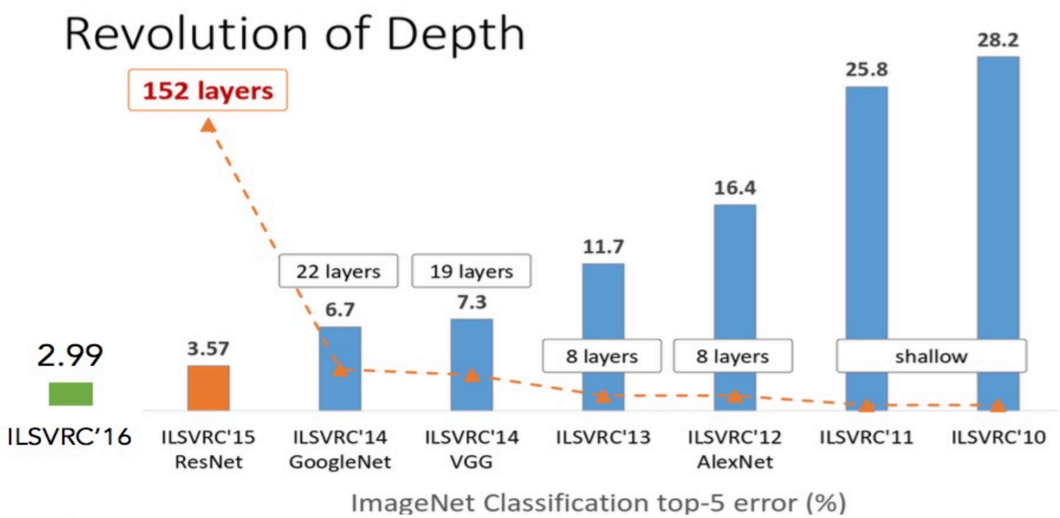
Machine Translation



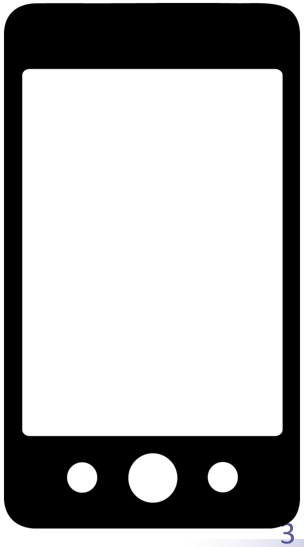
Dota

Limitations

- More data + deeper models → more FLOPs + lager memory



- Computation Intensive
- Memory Intensive
- Hard to deploy on mobile devices



Low-bit DNNs for Efficient Inference

- High Redundancy in DNNs;
- Quantize full-precision(32-bits) weights to binary(1 bit) or ternary(2 bits) weights;
- Replace multiplication(convolution) by addition and subtraction;

Typical Low-bit DNNs

- BinaryConnect:

$$B_i = \begin{cases} +1 & \text{with probability } p = \sigma(W_i) \\ -1 & \text{with probability } 1-p \end{cases}$$

- BWN: minimize $\|W - \alpha B\|$

$$B_i = sign(W_i), \qquad \alpha = \frac{\sum_{i=1}^{d} |W_i|}{d}$$

- TWN: minimize $\|W - \alpha T\|$

$$T_i = \begin{cases} +1 & \text{if } W_i > \Delta \\ 0 & \text{if } |W_i| < \Delta \\ -1 & \text{if } W_i < -\Delta \end{cases}, \qquad \alpha = \frac{\sum_{i \in I_\Delta} |W_i|}{|I_\Delta|}$$

$$I_\Delta = \{i \,|\, |W_i| > \Delta\}, \qquad \Delta = \frac{0.7}{d} \sum_{i=1}^{d} |W_i|$$

Training & Inference of Low-bit DNN

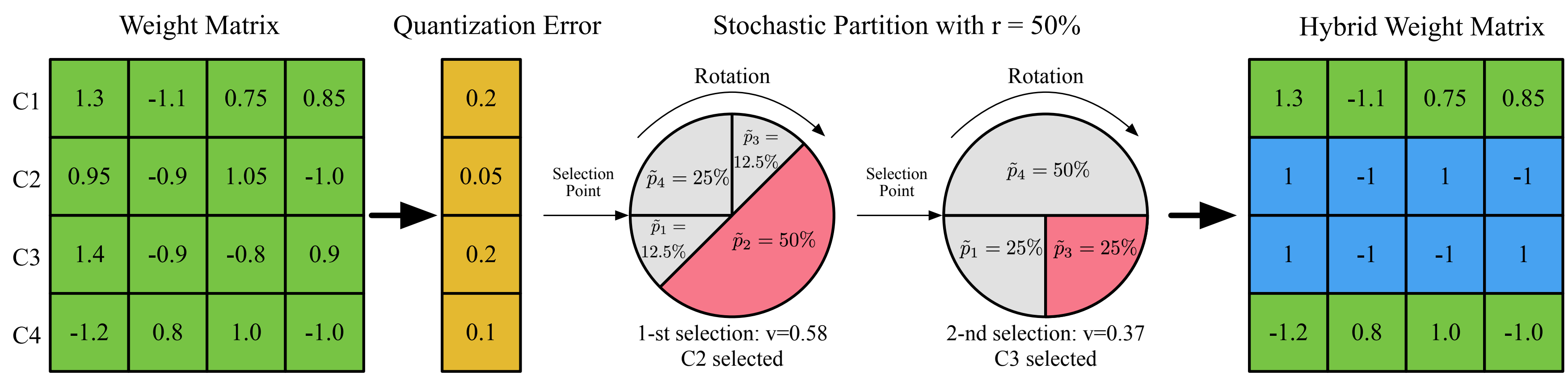
- Let W be the full-precision weights, Q be the low-bit weights (B, T, αB, αT).
- Forward propagation: quantize W to Q and perform convolution or multiplication
- Backward propagation: use Q to calculate gradients
- Parameter update: $W^{t+1} = W^{t} - \eta^{t} \frac{\partial L}{\partial Q^{t}}$
- Inference: only need to keep low-bit weights Q

Motivations

- Quantize all weights simultaneously;
- Quantization error $||W - Q||$ may be large for some elements/filters;
- Induce inappropriate gradient directions.

- **Quantize a portion of weights**
- **Stochastic selection**
- **Could be applied to any low-bit settings**

Roulette Selection Algorithm



Quantization Error: $$e_i = \frac{\|W_i - Q_i\|_1}{\|W_i\|_1}$$

Quantization Probability: Larger quantization error means smaller quantization probability, *e.g.* $p_i \propto \frac{1}{e_i}$

Quantization Ratio r: Gradually increase to 100%

Training & Inference

- Hybrid weight matrix $\tilde{Q}$

$$\tilde{Q}_i = \begin{cases} Q_i & \text{if channel i being selected} \\ W_i & \text{else} \end{cases}$$

- Parameter update

$$W^{t+1} = W^t - \eta^t \frac{\partial L}{\partial \tilde{Q}^t}$$

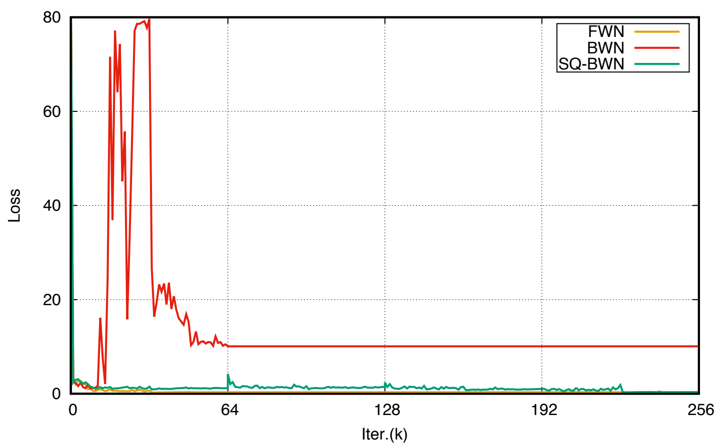
- Inference: all weights are quantized; use Q to perform inference

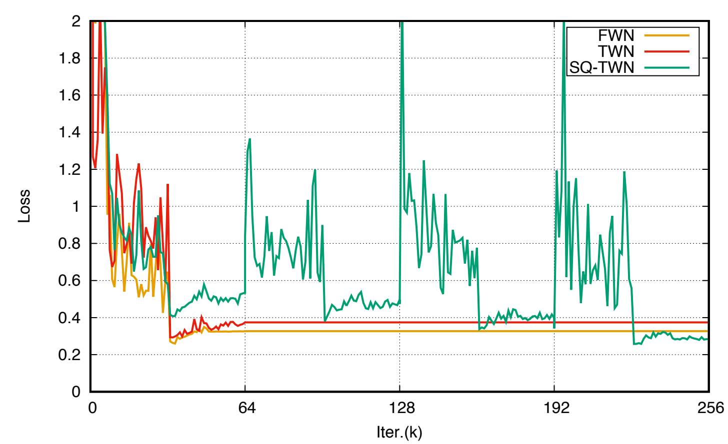
Ablation Studies

- Selection Granularity:
  - **Filter-level** > Element-level
- Selection/partition algorithms
  - **Stochastic** (roulette) > deterministic (sorting) ~ fixed (selection only at first iteration)
- Quantization functions
  - **Linear** > Sigmoid > Constant ~ Softmax
    - $p_i = \exp(f_i)/\sum\exp(f_i)$, where $f = \frac{1}{e}$
- Quantization Ratio Update Scheme
  - **Exponential** > Fine-tune > Uniformly
    - 50% → 75% → 87.5% → 100%

Results -- CIFAR

	Bits	CIFAR-10		CIFAR-100	
		VGG-9	ResNet-56	VGG-9	ResNet-56
FWN	32	9.00	6.69	30.68	29.49
BWN	1	10.67	16.42	37.68	35.01
SQ-BWN	1	9.40	7.15	35.25	31.56
TWN	2	9.87	7.64	34.80	32.09
SQ-TWN	2	**8.37**	**6.20**	34.24	**28.90**





Results -- ImageNet

	Bits	AlexNet-BN		ResNet-18	
		top-1	top-5	top-1	top-5
FWN	32	44.18	20.83	34.80	13.60
BWN	1	51.22	27.18	45.20	21.08
SQ-BWN	1	48.78	24.86	41.64	18.35
TWN	2	47.54	23.81	39.83	17.02
SQ-TWN	2	44.70	21.40	36.18	14.26

Conclusions

- We propose a stochastic quantization algorithm for Low-bit DNN training
- Our algorithm can be flexibly applied to all low-bit settings;
- Our algorithm help to consistently improve the performance;
- We release our codes to public for future development
  - https://github.com/dongyp13/Stochastic-Quantization

Q & A