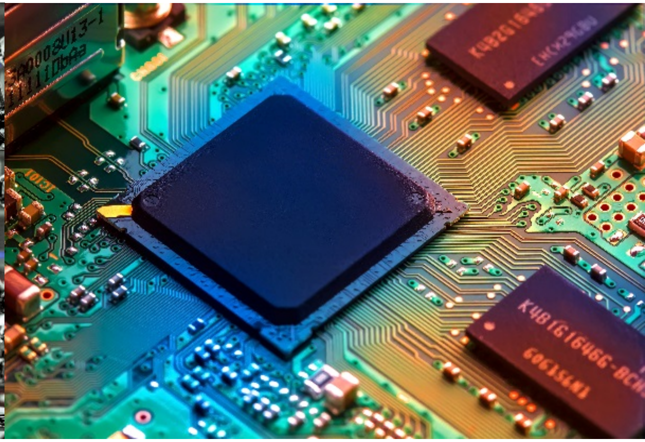
Minimum Precision Requirements of Deep Neural Networks

Numerical Software Verification
July 20, 2020



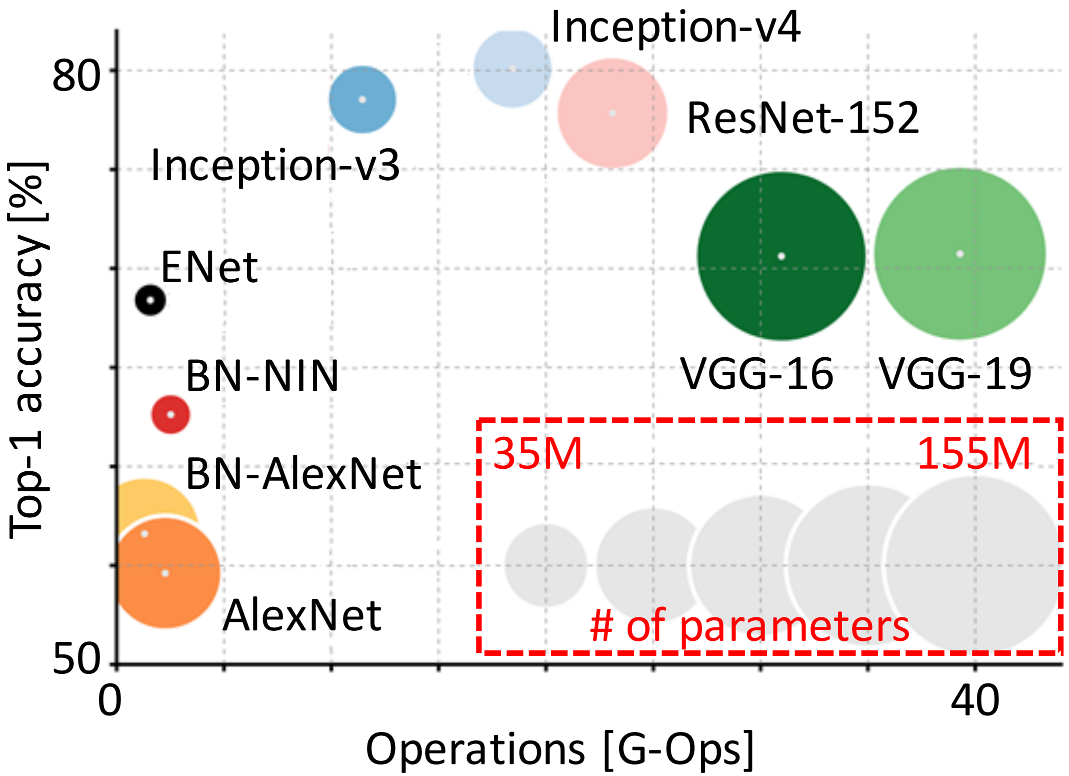
Naresh R. Shanbhag

ILLINOIS
Electrical & Computer Engineering
COLLEGE OF ENGINEERING

Charbel Sakr, Sujan Gonugondla, and Hassan Dbouk

The Efficiency Challenge in Learning

Model Complexity

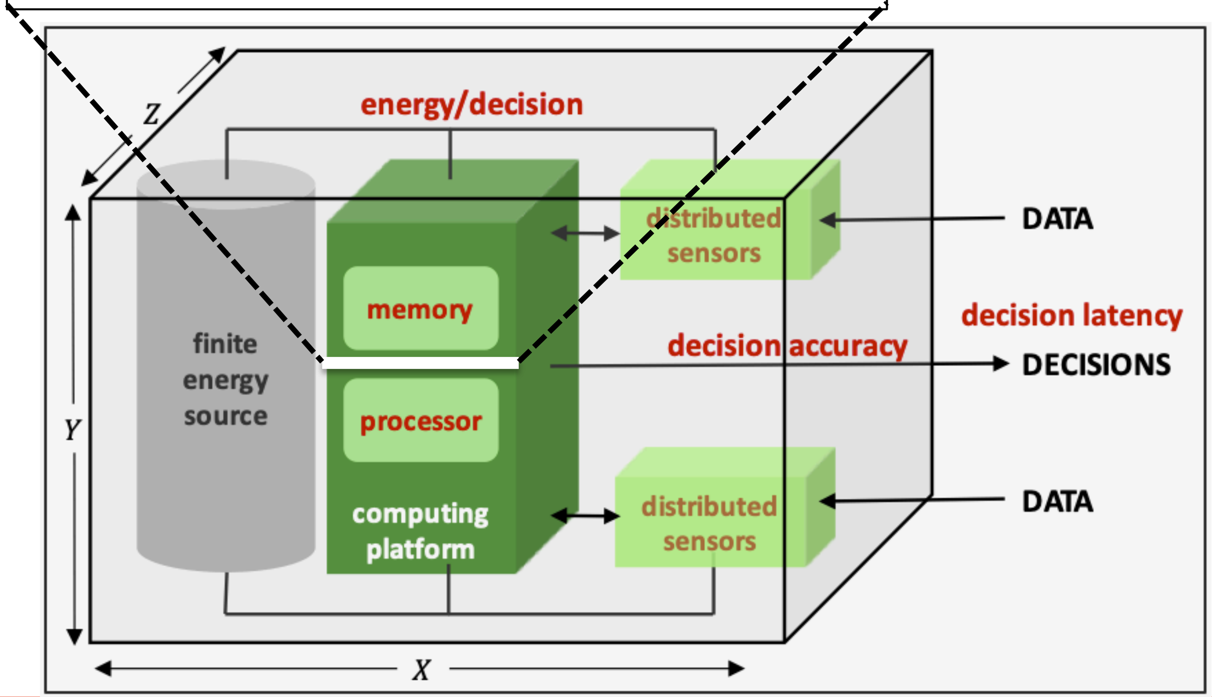


[Canziani, Arxiv'16]

Data Movement Cost

$$\frac{E_{mem}}{E_{mac}} \approx \quad \sim 100\times \text{ (SRAM)} \rightarrow \sim 500\times \text{ (DRAM)}$$

The Memory Wall in von Neumann Architectures



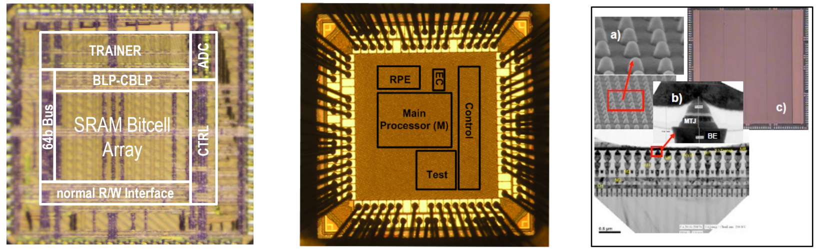
<u>fundamental question</u>

how do we design **learning machines** that
operate at the limits of **accuracy-robustness-energy efficiency** with **guarantees**?

Shanbhag Group Research Vectors

http://shanbhag.ece.Illinois.edu

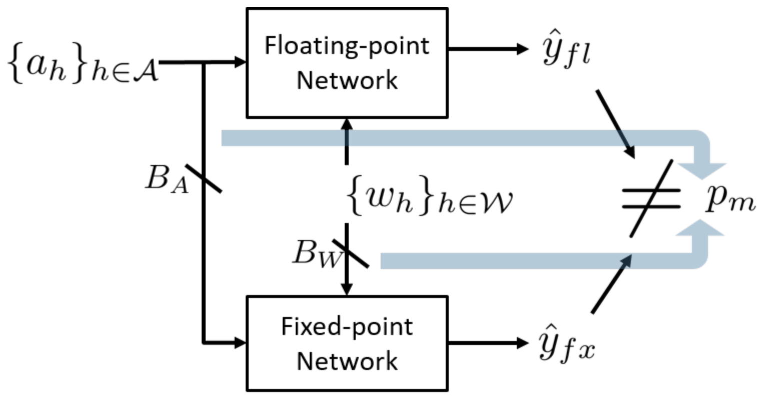
energy efficient circuit architectures



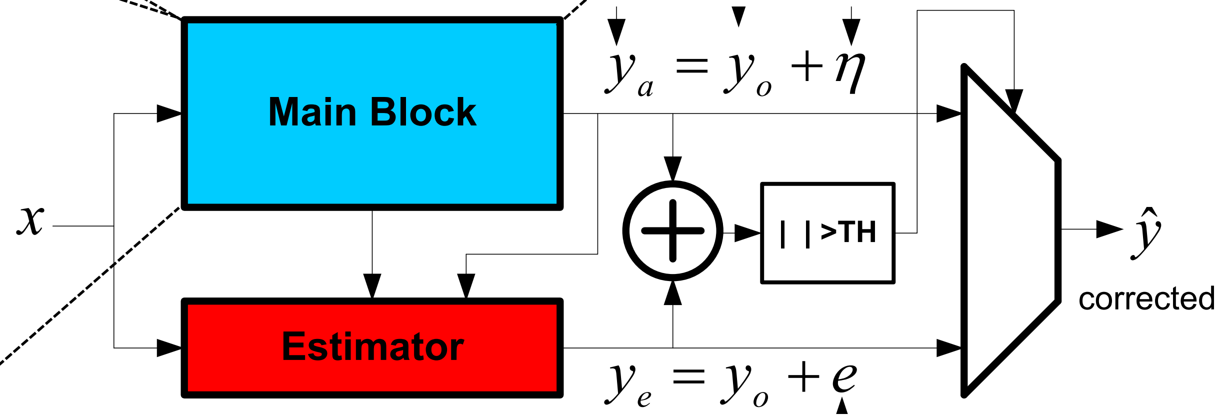
applications

- Computer Vision
- ATR
- RF Signal Processing
- Biomedical

low complexity algorithms

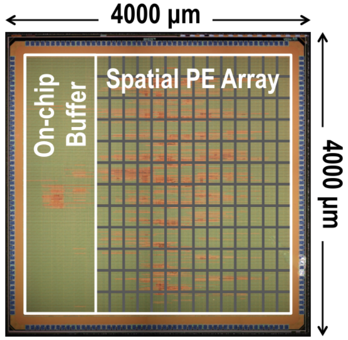
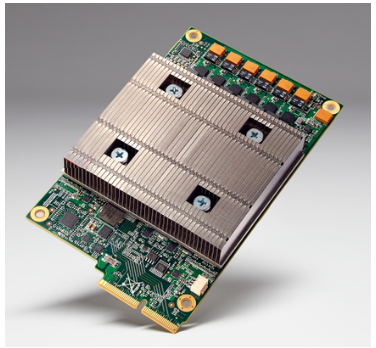
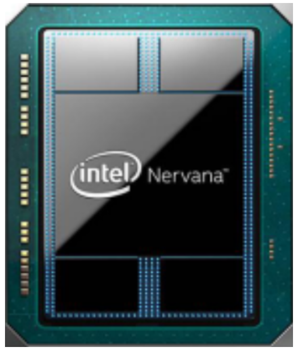
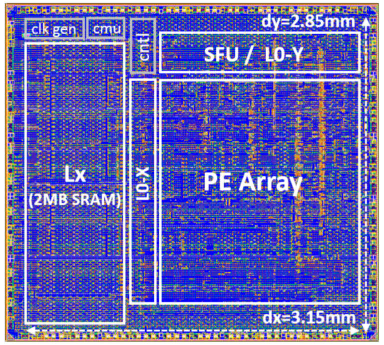


$$p_m \leq \Delta_A^2 E_A + \Delta_W^2 E_W$$



Shannon-inspired model of computation

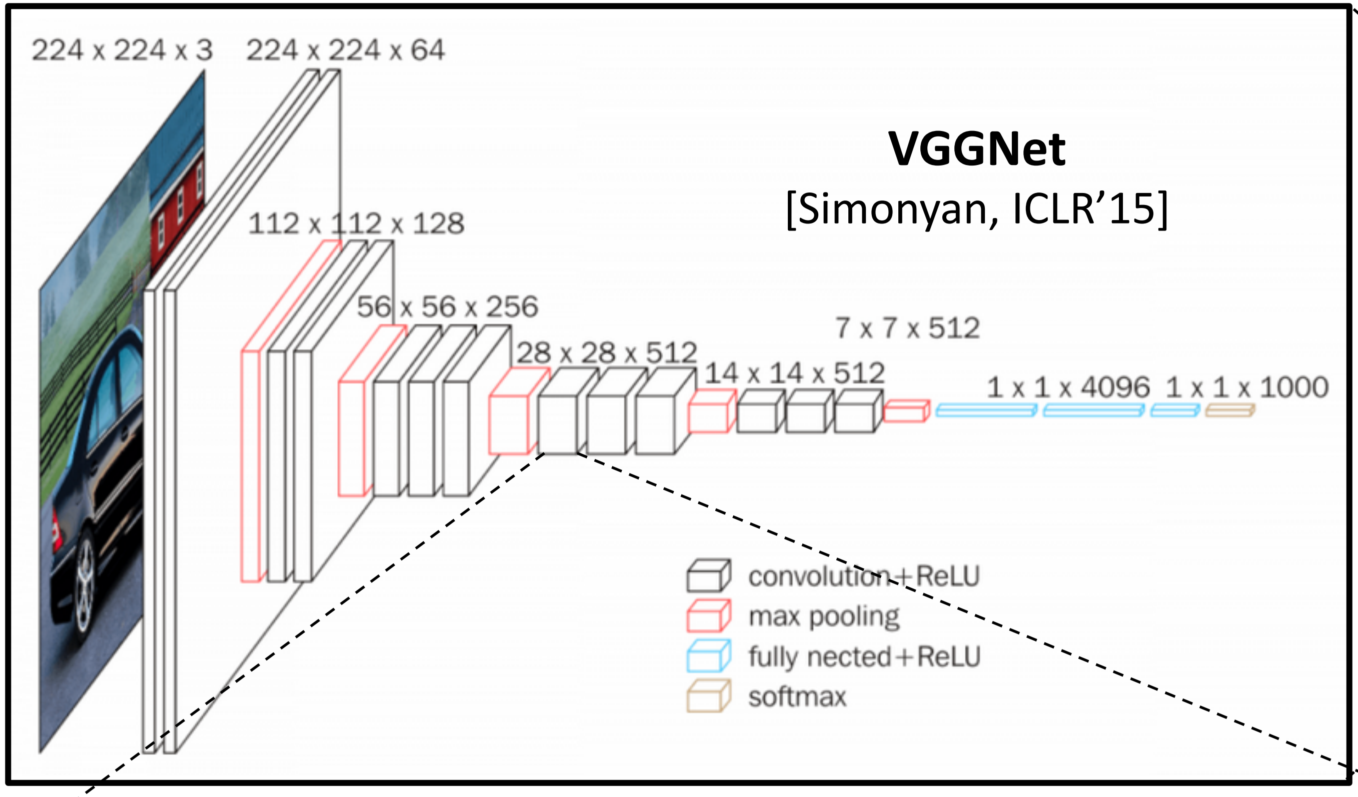
Machine Learning in Reduced Precision

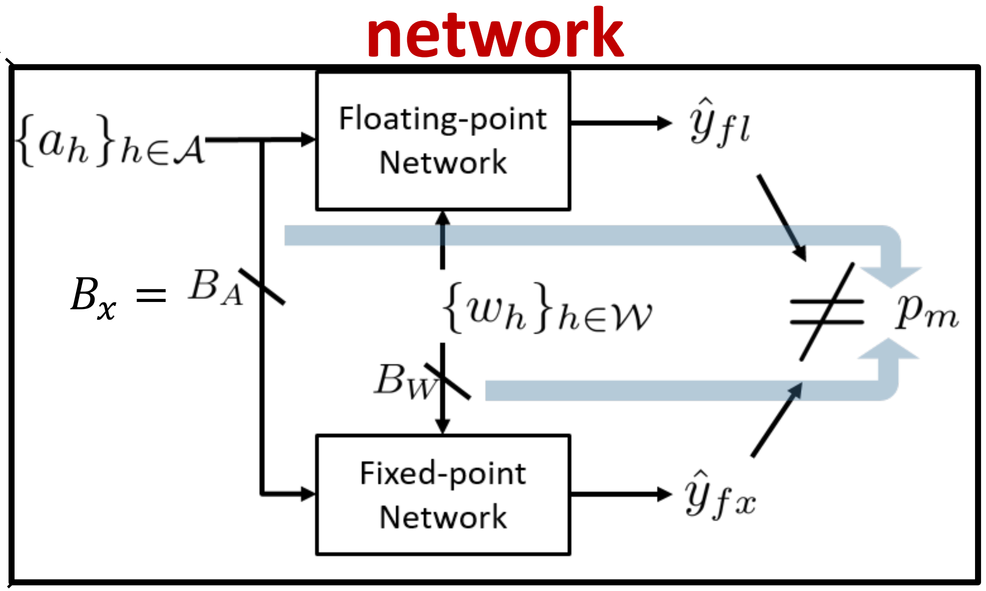
MIT's Eyeriss	Google's TPU	Intel's NNP	IBM's AI Core
			
[ISSCC'16]	**[ISCA'17]**	**[NIPS'17]**	**[VLSI'18]**
16b fixed-point (inference)	**8b fixed-point** (inference) **16b floating-point** (training)	**16b flexpoint** (training)	**16b floating-point** (training)

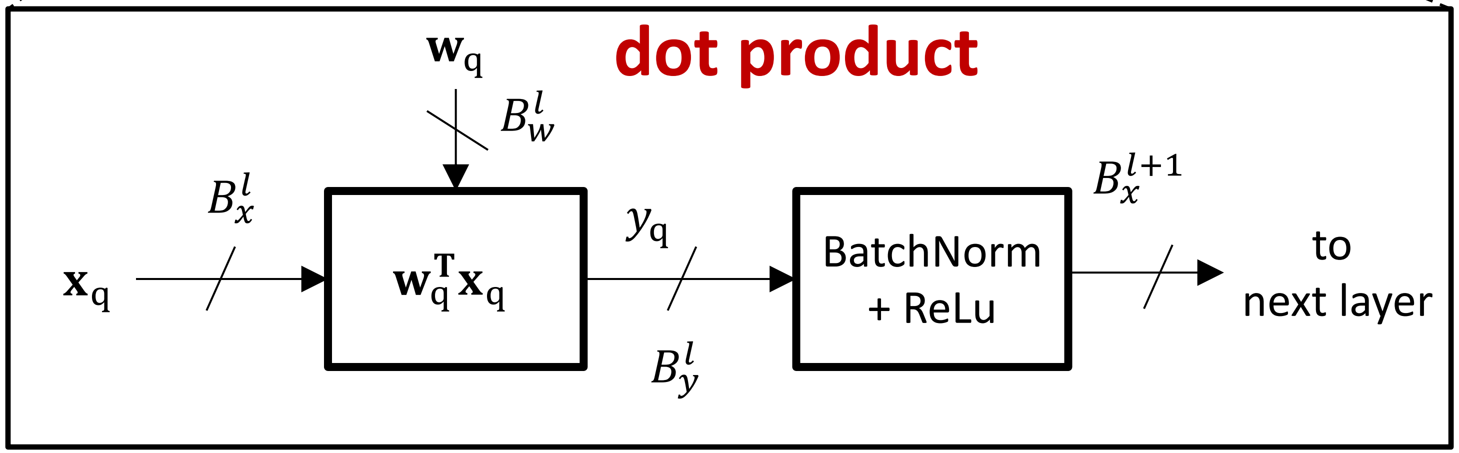
Are these the minimum precisions required?

Can minimum precision requirements be determined analytically?

UIUC (Sakr, Shanbhag) - ICML 2017, ICASSP 2018, ICLR 2019, ICLR 2019 (with IBM)







- what are the minimum values of B_x^l, B_w^l, and B_y^l $\forall\, l$ such that the network accuracy is within a Δ of floating-point network accuracy?

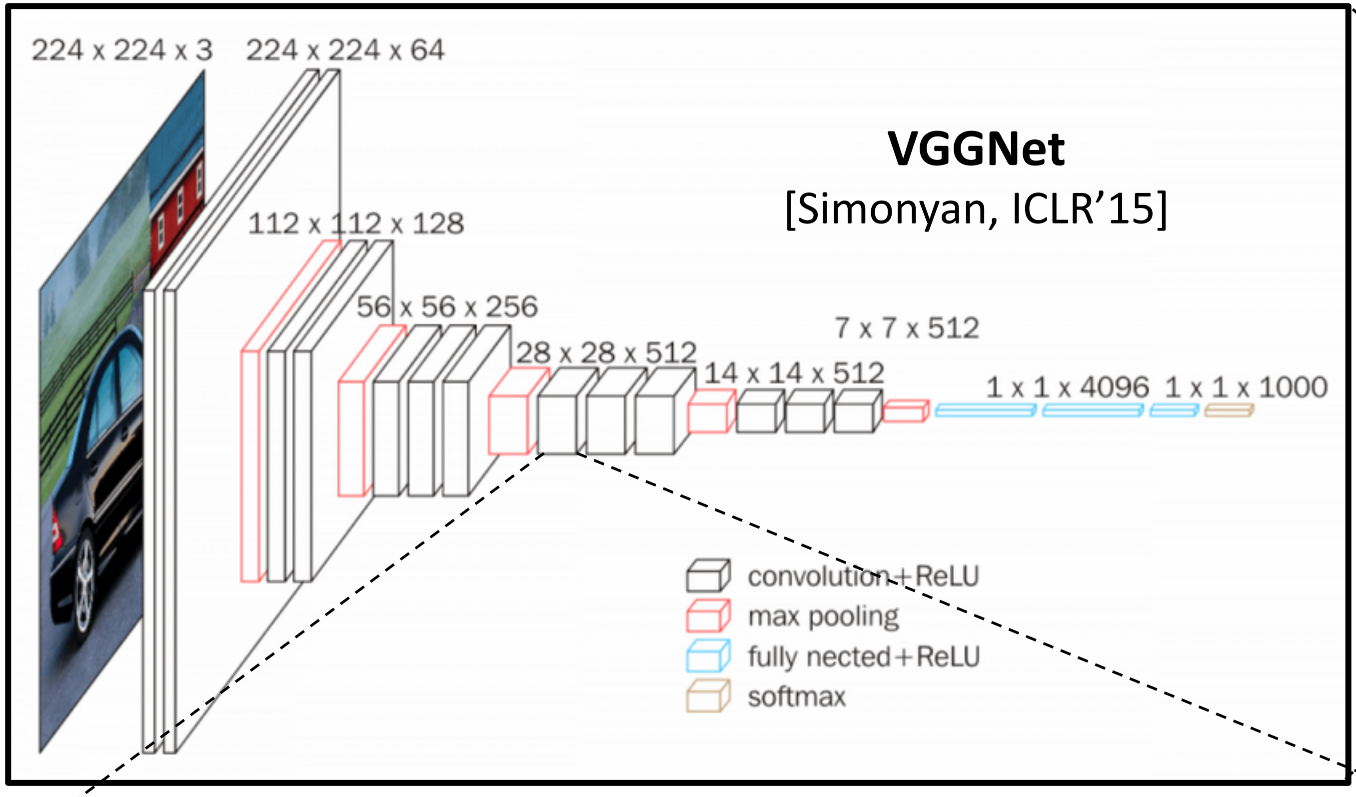
Outline

1) "What are the minimum output precision requirements of a **dot product kernel** in order to meet a specific accuracy requirement at its output **(DP accuracy)**?"

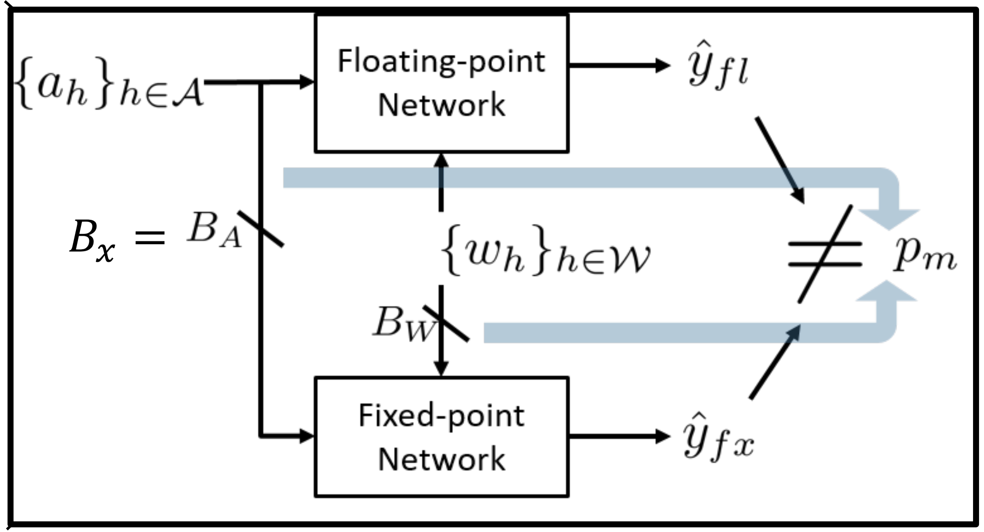
2) "What are the minimum precision requirements of a **DNN** to meet a specific accuracy requirements at its output (**network accuracy**)?"

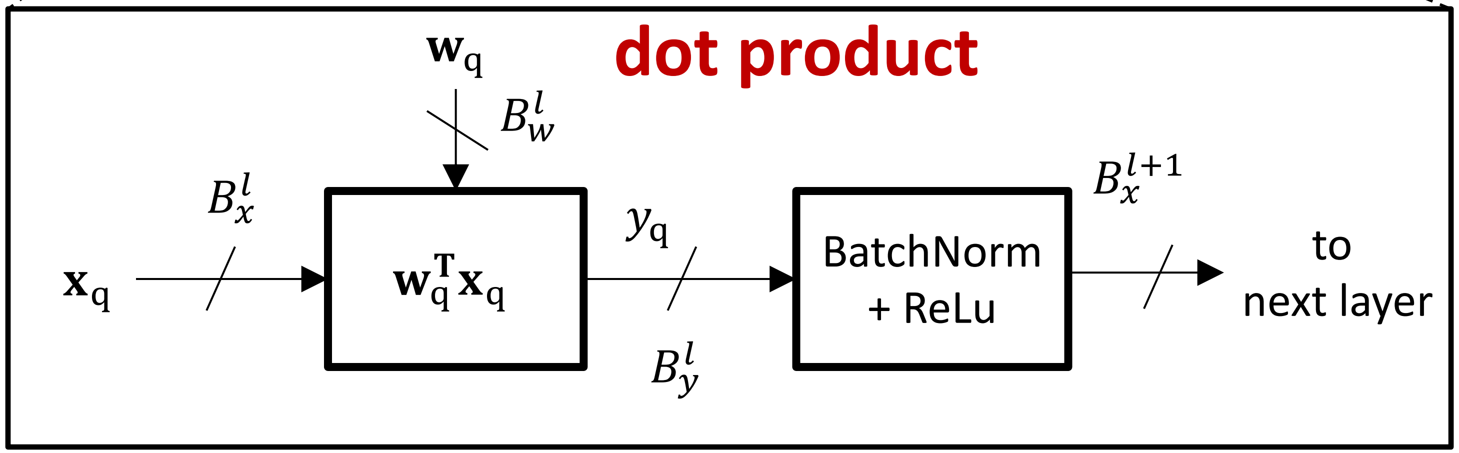
3) employ the above two insights to determine the precision limits of the recently proposed **in-memory computing (IMC) architectures.**

Minimum Output Precision Requirements for Dot Product



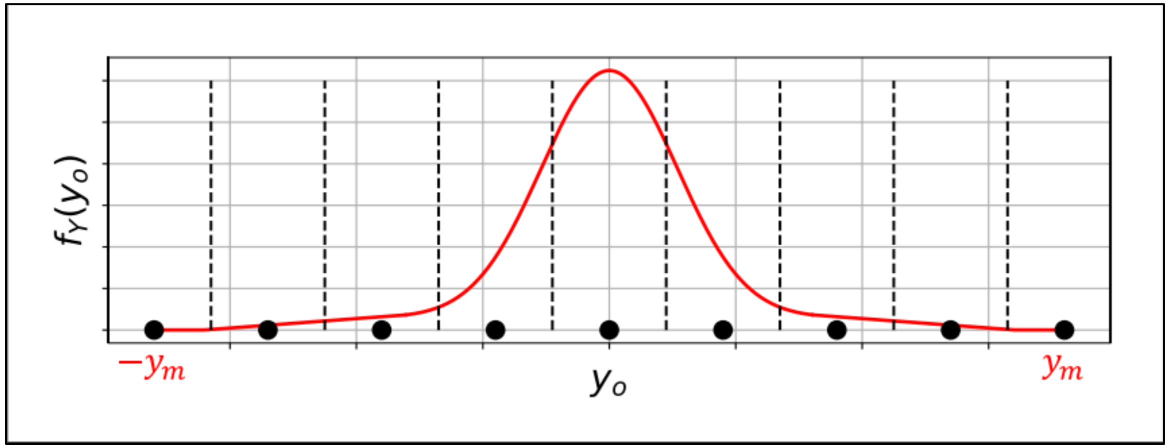
network





- what are the minimum values of B_x^l, B_w^l, and $\boldsymbol{B_y^l}$ $\forall\, l$ such that the network accuracy is within a Δ of floating-point network accuracy?

Why Output Precision B_y?

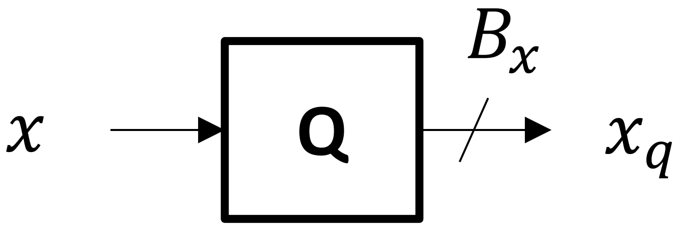


$$y_q = \mathbf{w}^{\mathrm{T}}\mathbf{x} + q_y$$

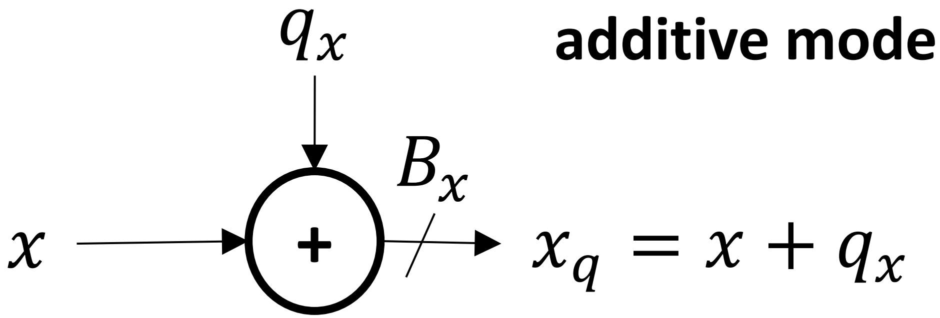
- B_y is the accumulator precision in digital architectures → accumulator complexity dominates power in low-precision DNNs
  - e.g., 32b accumulator 10× more power than a 3×1-b multiplier in 28nm CMOS – hence research on low-resolution accumulation [Sakr ICLR19; Wang NeurIPS'18]
- B_y is the ADC precision in in-memory architectures → ADCs can dominate (~80%) latency and power when implementing DNNs [Kim ISLPED'18, Rekhi DAC'20]

Quantization Noise Model

quantizer symbol



additive model



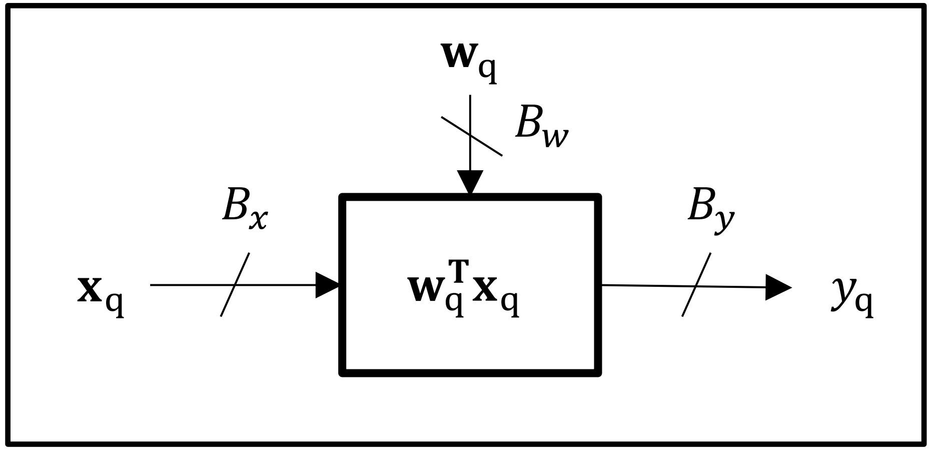
- additive model assumption: q_x is independent of x
- $SQNR$: signal-to-quantization noise ratio → accuracy measure
- ζ : peak-to-average (power) ratio → measure of ‘peakiness’ of signal distribution

$$SQNR_x = 10\log_{10}\left[\frac{\sigma_x^2}{\sigma_{q_x}^2}\right]$$

$$SQNR_x(dB) = 6B_x + 4.78 - \zeta_x\ (dB)$$

$$\zeta_x = \frac{x_m}{\sigma_x}$$

Fixed-Point Dot Product



$$y_q = \underbrace{\overbrace{\mathbf{w}^{\mathrm{T}}\mathbf{x}}^{\overbrace{y_o}^{\text{ideal FL output}}} + q_{iy} + \overbrace{q_y}^{\text{output quantization noise}}}_{\text{input quantization noise \textit{reflected} at the output}}$$

Accuracy Metrics for Quantized Dot Products

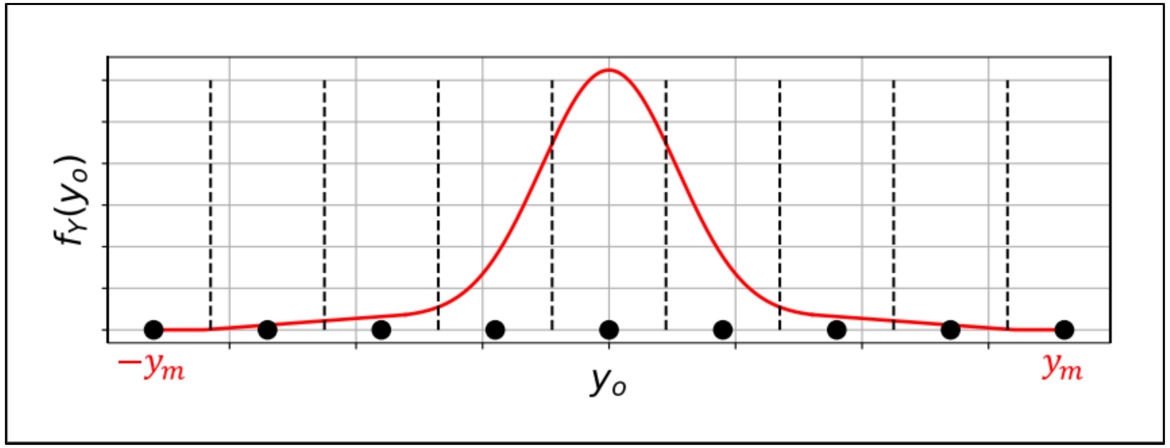
$$\boxed{SQNR_{\mathrm{T}} = \left[\frac{1}{SQNR_{q_{iy}}} + \frac{1}{SQNR_y}\right]^{-1}} \rightarrow \boxed{\text{Limited by } SQNR_{q_{iy}}}$$

- Choose $SQNR_y(dB) \geq SQNR_{q_{iy}}(dB) + 9$ to minimize ($< 0.5dB$) its impact on $SQNR_{\mathrm{T}}$

$$\boxed{SQNR_y(dB) = 6B_y + 4.8 - [\zeta_x(dB) + \zeta_w(dB)] - 10\log_{10}(N)}$$

- But for fixed B_y: $SQNR_y(dB)$ **reduces with N** (N in hundreds in DNNs) → increase B_y
- But large B_y → leads to very large accumulator bit widths
- **How to choose output precision B_y?**

Bit Growth Criterion (BGC) for Choosing B_y

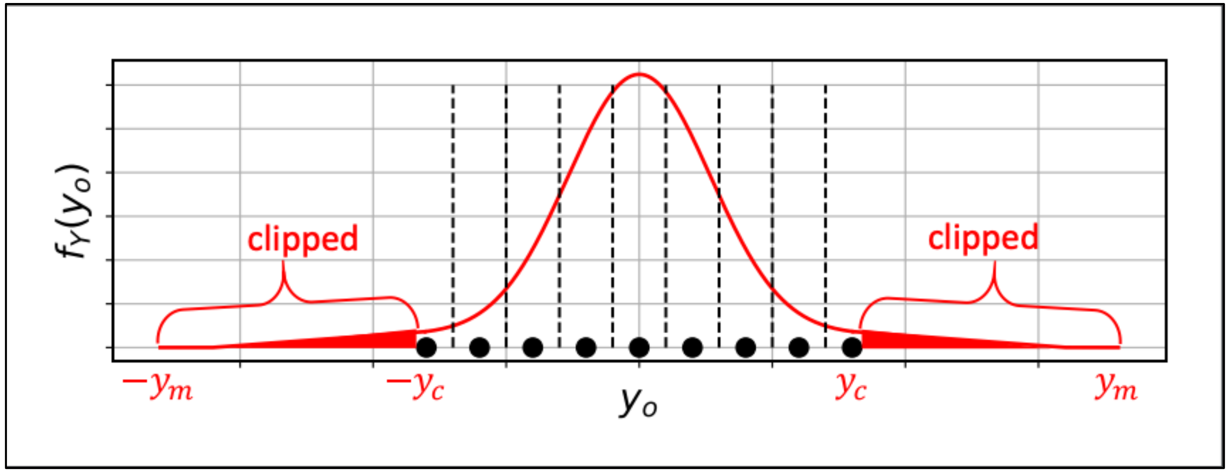


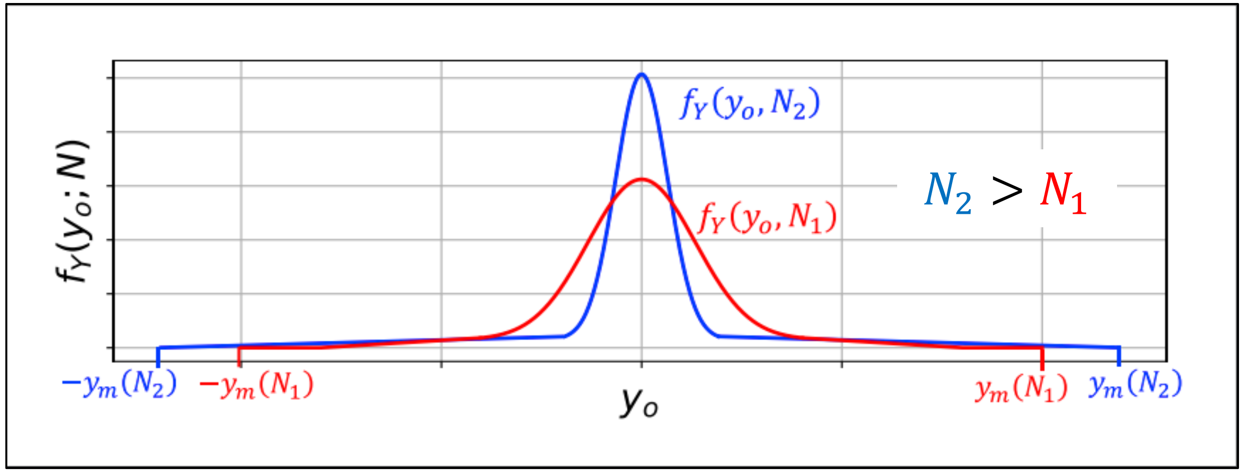
$$\boxed{B_y = B_x + B_w + \log_2(N)}$$

$$\boxed{SQNR_y^{BGC}(dB) = 6(B_x + B_w) + 4.8 - [\zeta_x(dB) + \zeta_w(dB)] + 10\log_{10}(N)}$$

- commonly employed in digital architectures and network design
- B_y (accumulator precision) and $SQNR_y$ both **increase with N**

Proposed - Minimum Precision Criterion (MPC)

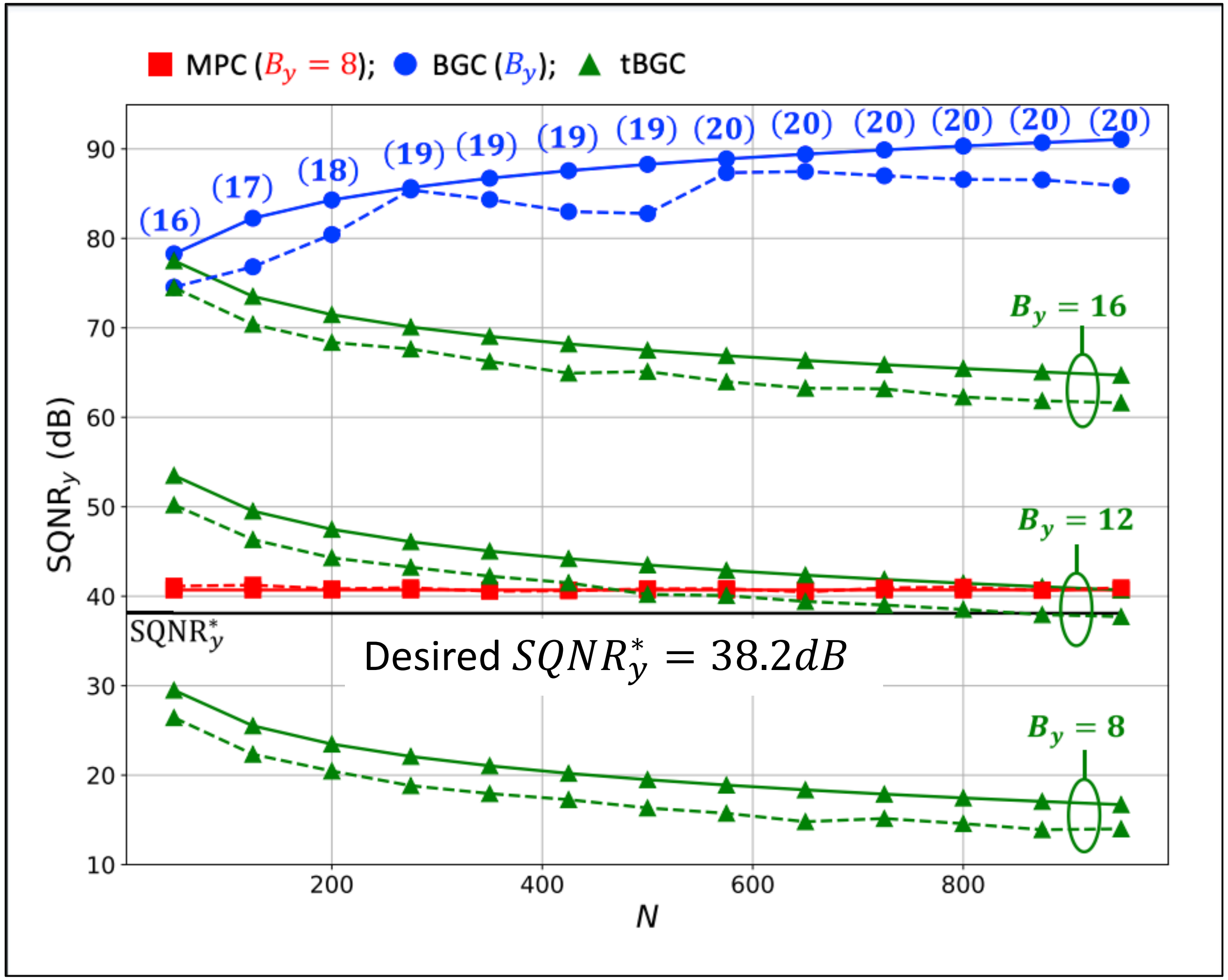




$$SQNR_y^{MPC}(dB) = 6B_y + 4.8 - \zeta_y^{MPC}(dB) - 10\log_{10}\left(1 + p_c \frac{\sigma_{cc}^2}{\sigma_{q_y}^2}\right)$$

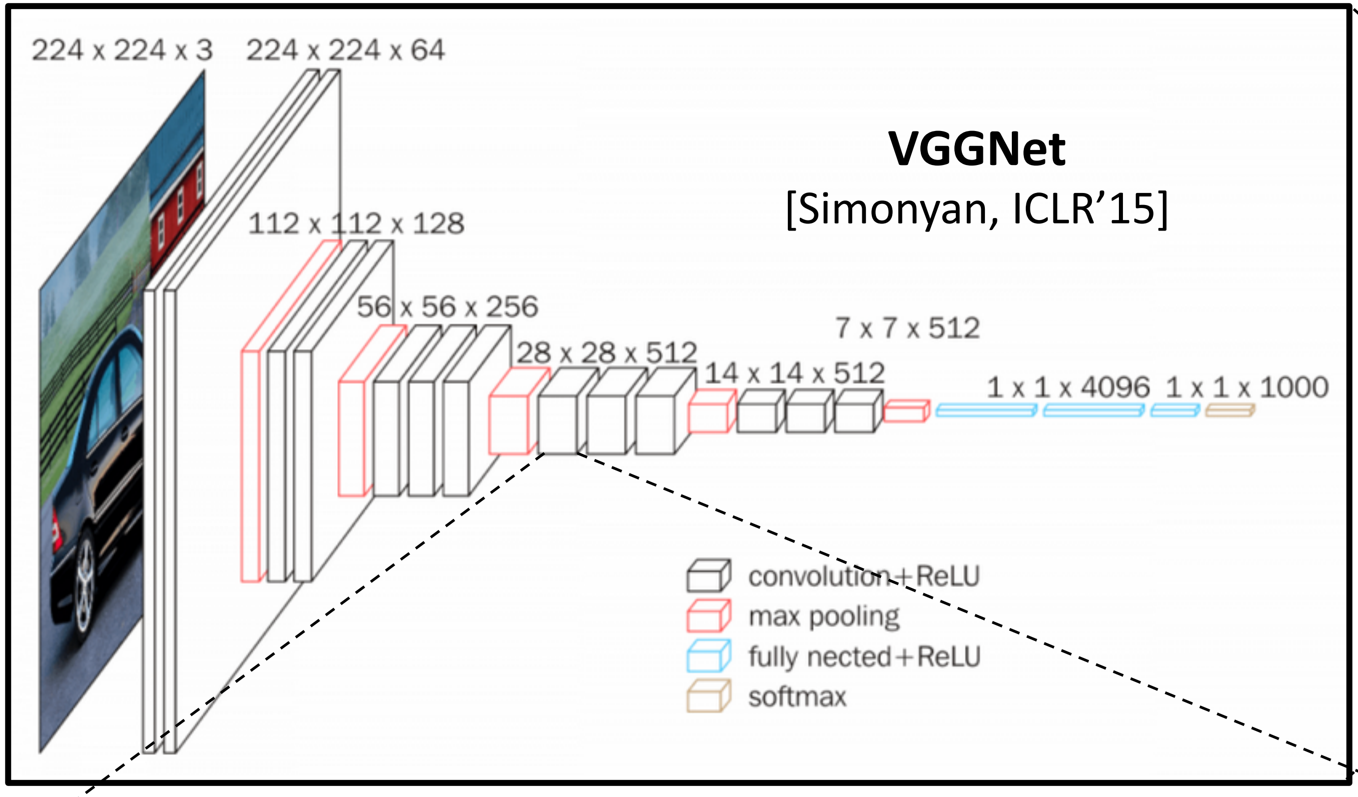
- allow for a non-zero but small probability of clipping (p_c) – BGC avoids clipping
- exploits reduction in $\frac{\sigma}{\mu}$ of y_o with N (Central Limit Theorem)
- exhibits a trade-off between clipping noise and quantization noise

Comparing MPC and BGC

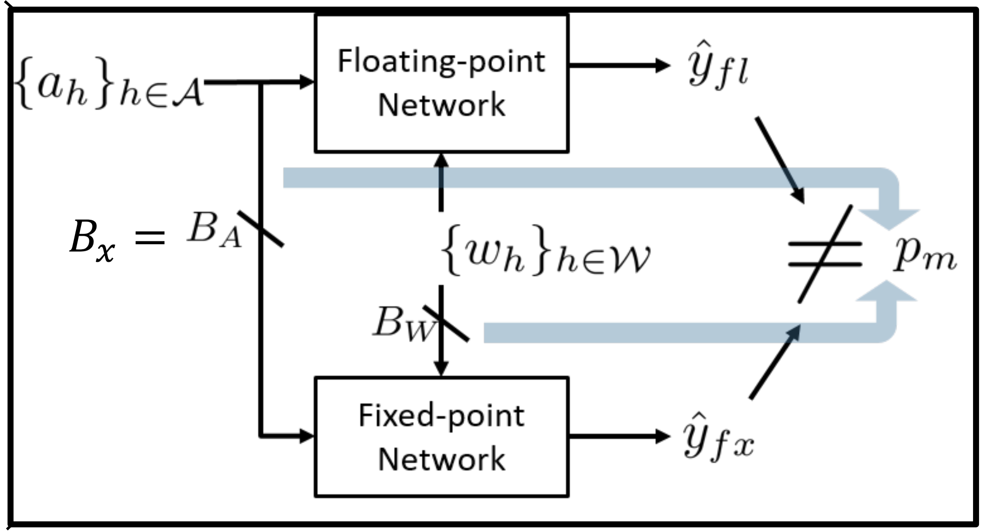


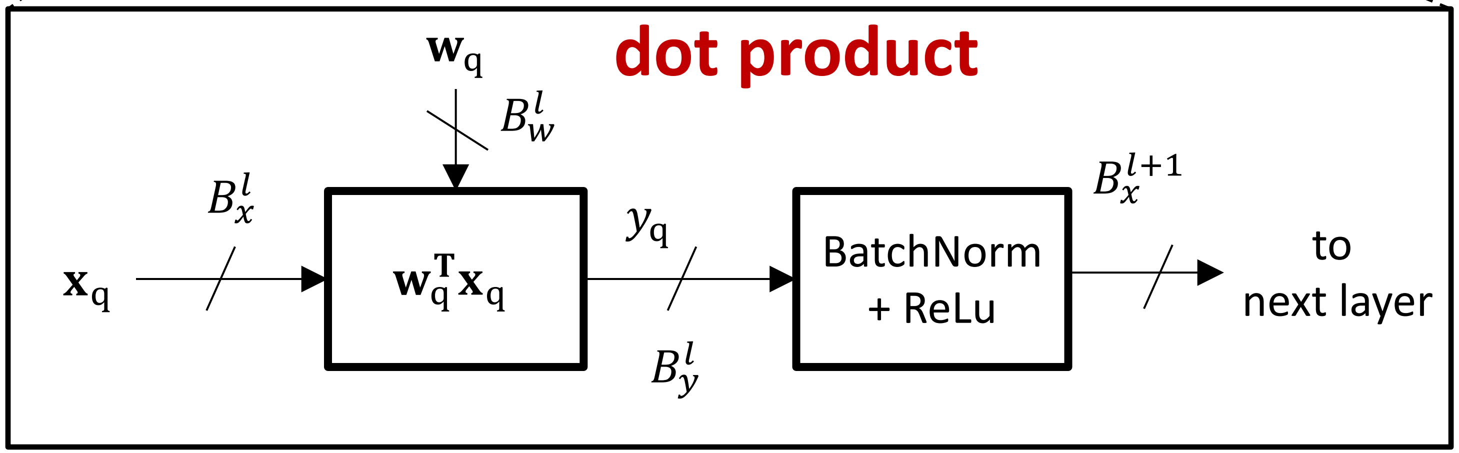
- MPC achieves the desired $SQNR_y^*$ with minimum precision ($B_y = 8$)
- BGC is a huge overkill → leads to very large accumulator bit widths ($B_y = 16$ to 20)
- tBGC (truncated BGC) needs $B_y = 12$ (still significant)
- Use MPC to assign minimum output (accumulator) precision

Input Precision Requirements for DNNs



network





- what are the minimum values of $\boldsymbol{B_x^l}$, $\boldsymbol{B_w^l}$, and B_y^l $\forall\, l$ such that the network accuracy is within a Δ of floating-point network accuracy?

Related Works

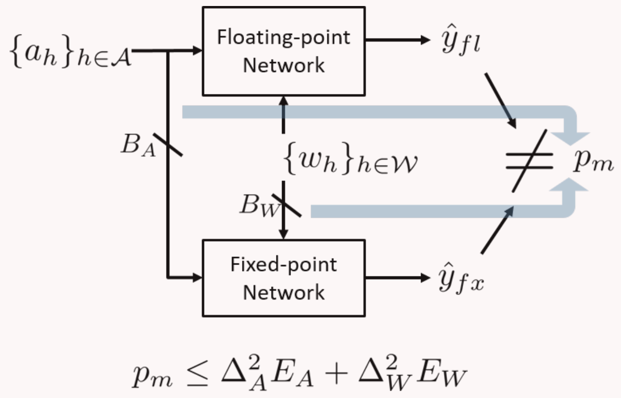
- much work on reduced precision machine learning since 2015 [stochastic rounding, BinaryNets, TernGrad, pruning…]

But……

- largely based on heuristics – relying on the benevolence of SGD
- lacking theoretical guarantees on accuracy – try and hope it works!
- difficult to realize in H/W – complex and irregular arithmetic

Deep Learning in Finite Precision

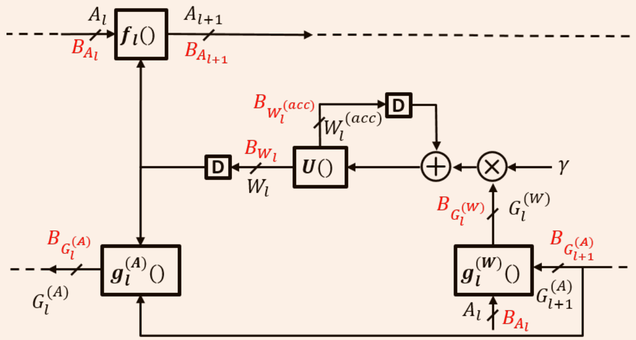
Fixed-point **inference** with theoretical guarantees



Sakr, Kim, Shanbhag
ICML 2017

Sakr & Shanbhag
ICASSP 2018

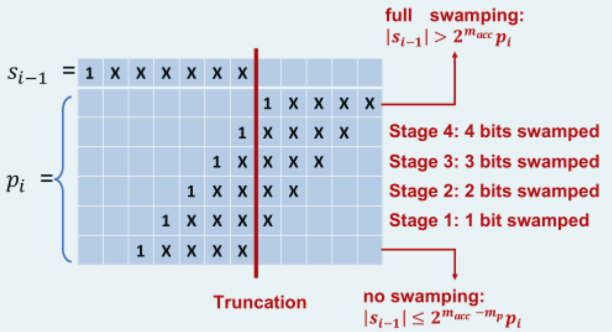
Fixed-point **training** with close-to-minimal precisions



Sakr & Shanbhag

ICLR 2019

Fl.-pt. training with accumulation bit-width scaling



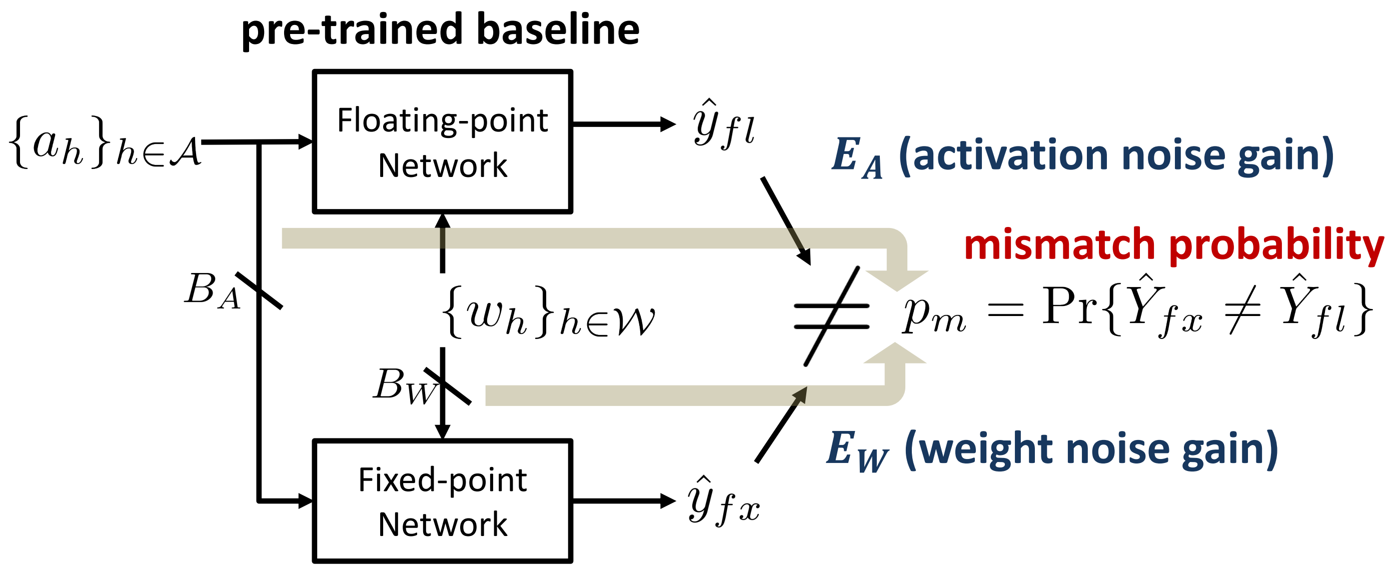
Sakr & Shanbhag

(*with K. Gopalakrishnan [IBM]*)

ICLR 2019

code available at https://github.com/charbel-sakr

Precision Analysis Framework



$$p_m \leq \Delta_A^2 E_A + \Delta_W^2 E_W$$

- no retraining; per-layer precision; activation vs. weight trade-off
- noise gains computed via one standard backprop iteration
- minimizing p_m done via noise equalization (NE)

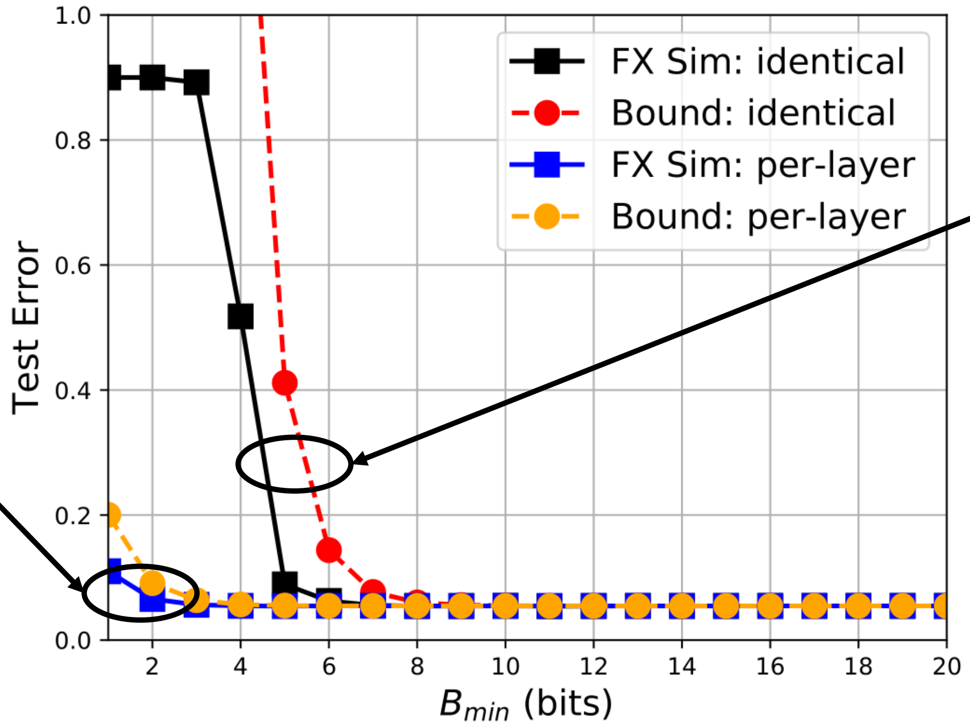
Lesson 1 – Precision Trade-offs Captured Analytically

$$B_{A,l} = \log_2 \sqrt{\frac{E_{A,l}}{E_{\min}}} + B_{\min}$$

&

$$B_{W,l} = \log_2 \sqrt{\frac{E_{W,l}}{E_{\min}}} + B_{\min}$$

CIFAR-10 using ResNet 18



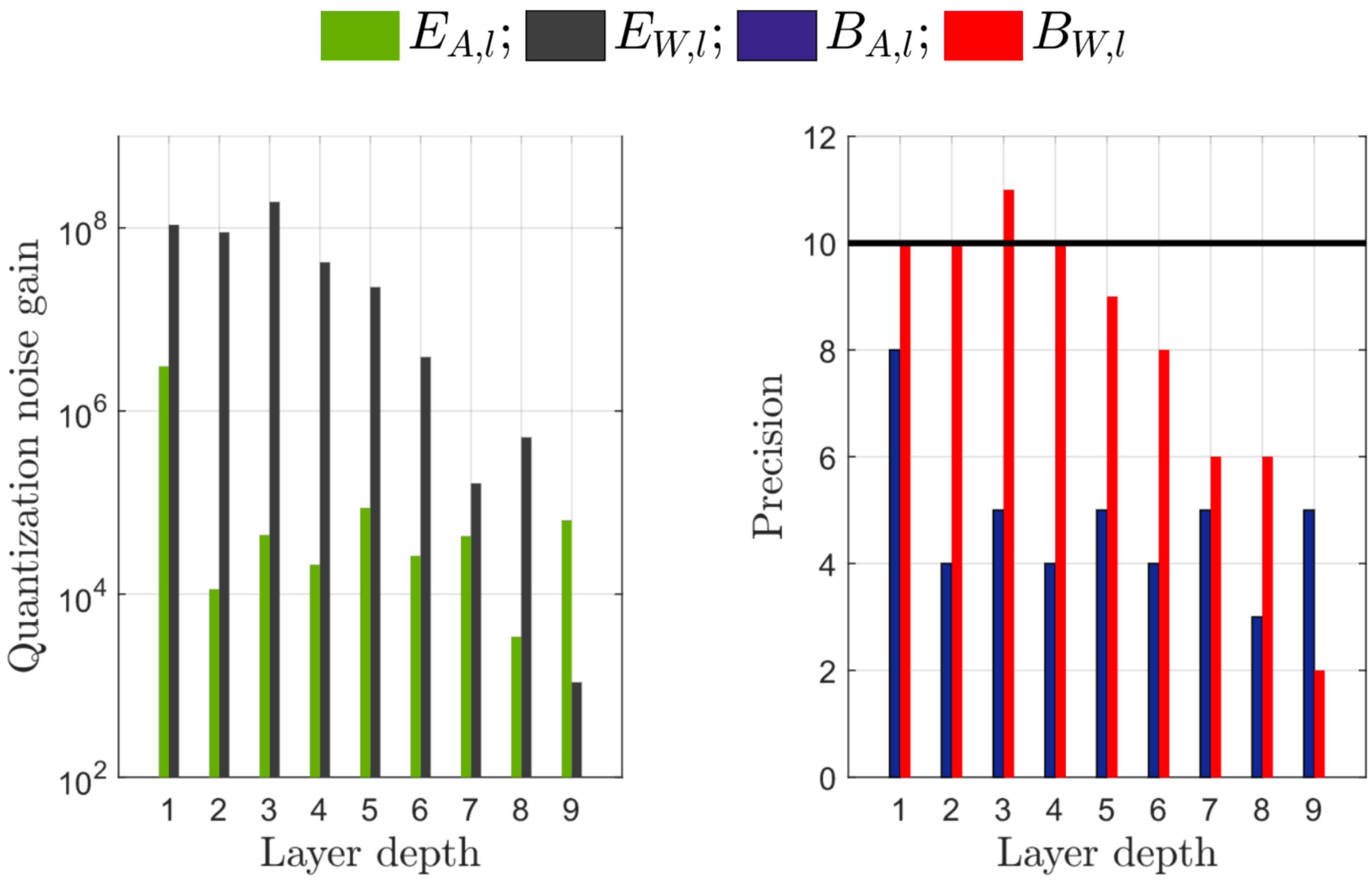
$B_{A,l} = B_{W,l} = B_{\min}$
for $l = 1 \dots L$

- bound predicts minimum precision within 1~2 bits
- bound reveals trade-offs between network precisions
- trade-offs captured by relative values of noise gains

Lesson 2 – Precision Decreases with Depth

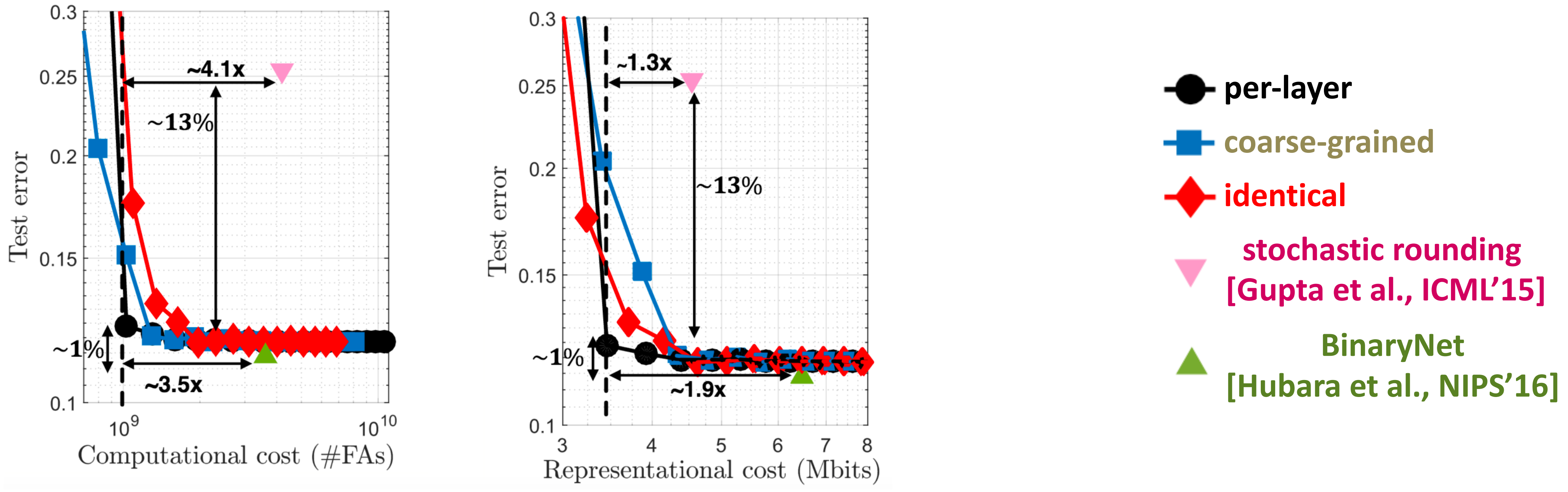
- weights typically require more precision than activations
- precision decreases because early perturbations are most destructive

CIFAR-10 using VGG-9



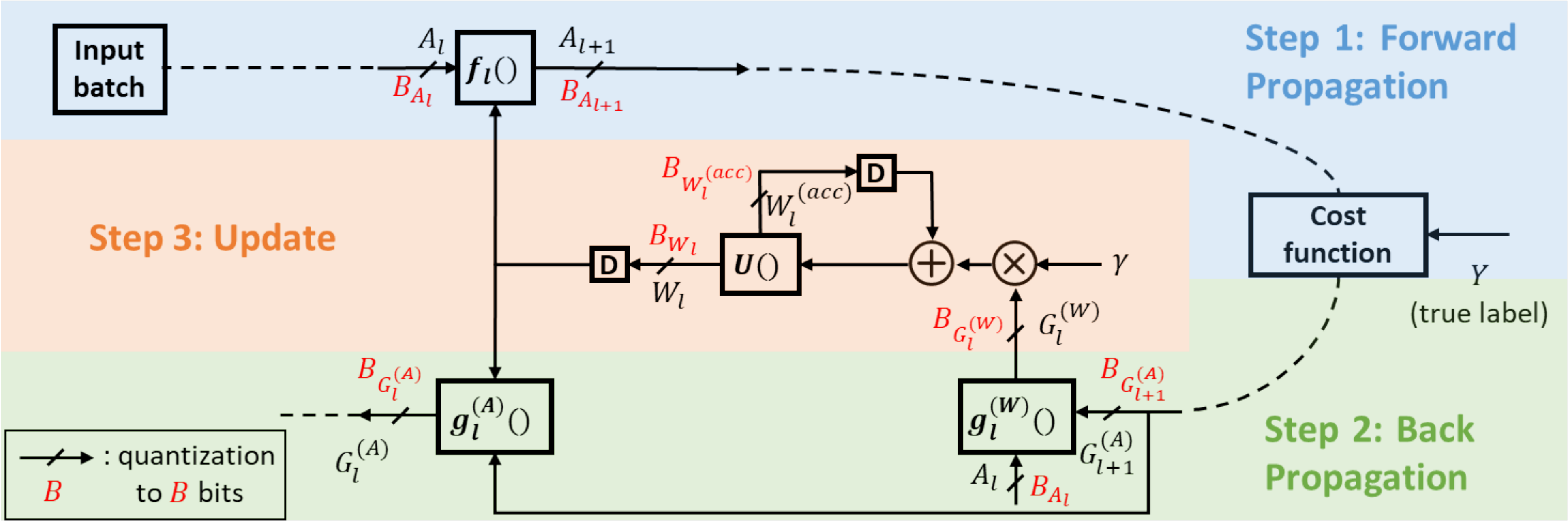
Lesson 3 – BinaryNets are More Complex than Minimum Precision Networks

CIFAR-10 using VGG-9



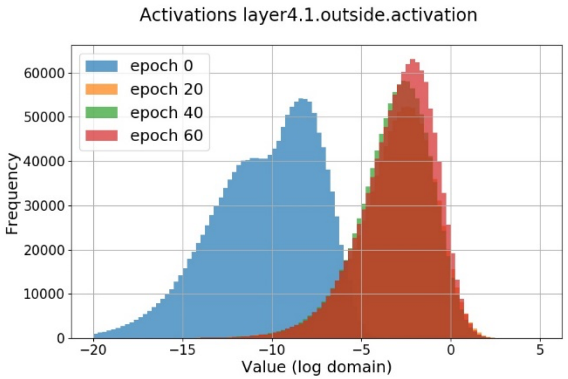
- up to **3.5x lower** complexity & **2X** lower storage over BinaryNet at **iso-accuracy**
- empirically observed by [Moons, Verhelst]

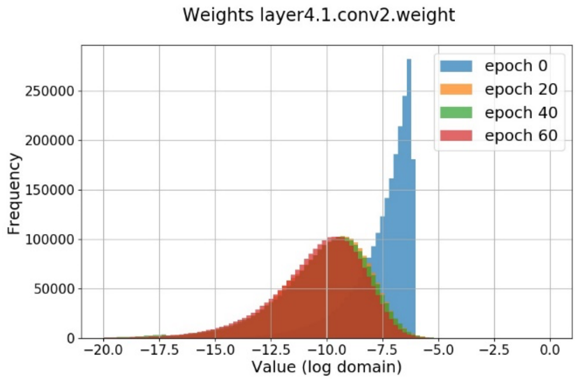
Challenges in Fixed-point Training



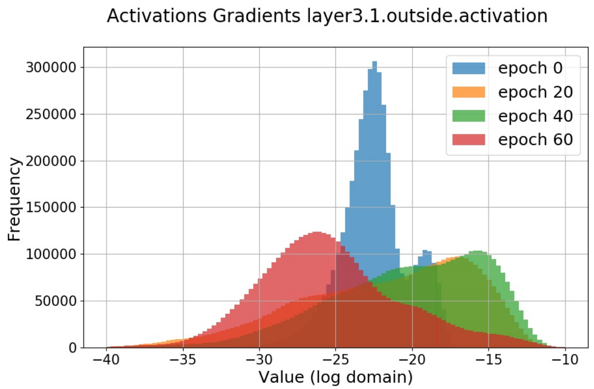
- multiple forward quantization noise sources
- unknown gradient dynamic range
- instability due to quantization noise bias in updates
- back-propagation of quantization noise in activation gradients
- risk of premature stoppage of convergence

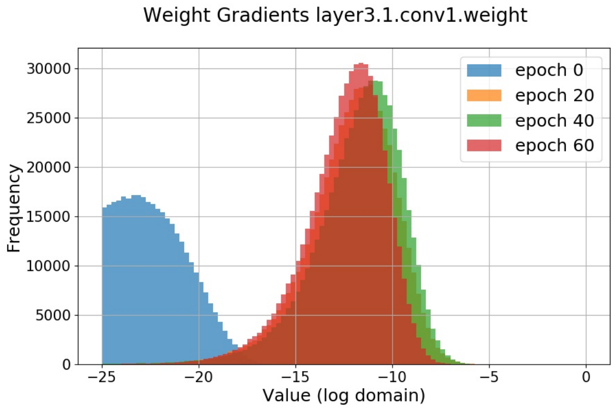
activations & weights
spatio-temporally varying distributions



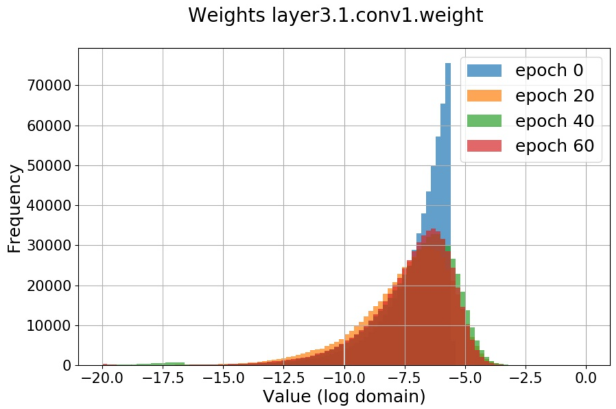


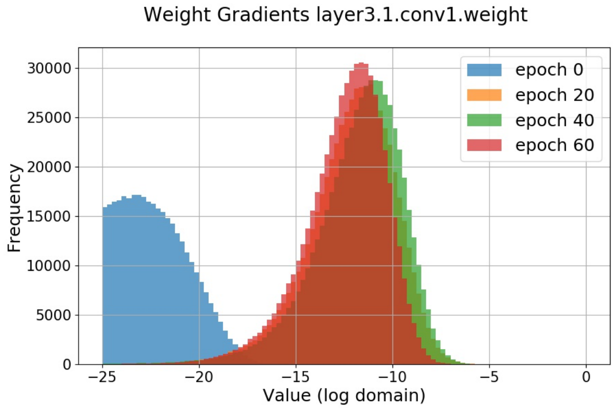
gradients
spatio-temporally varying dynamic range





gradients & weights
huge dynamic range mismatch





[ResNet-18 on CIFAR-10]

The Solution

Criterion 1: equalization of quantization noise gains

$$B_{W_l} = rnd\left(\log_2\left(\sqrt{\frac{E_{W_l \to p_m}}{E^{(\min)}}}\right)\right) + B^{(\min)}$$

$$B_{A_l} = rnd\left(\log_2\left(\sqrt{\frac{E_{A_l \to p_m}}{E^{(\min)}}}\right)\right) + B^{(\min)}$$

Criterion 2: proper gradient clipping

$$r_{G_l^{(W)}} \geq 2\sigma^{(\max)}_{G_l^{(W)}}$$

$$r_{G_{l+1}^{(A)}} \geq 4\sigma^{(\max)}_{G_{l+1}^{(A)}}$$

Criterion 3: quantization bias elimination

$$\Delta_{G_l^{(W)}} < \frac{\sigma^{(\min)}_{G_l^{(W)}}}{4}$$

Criterion 4: back-propagated noise bound

$$\Delta_{G_{l+1}^{(A)}} < \frac{\Delta_{G_l^{(W)}}}{\sqrt{\lambda^{(\max)}_{G_{l+1}^{(A)} \to G_l^{(W)}}}}\left(\frac{\left|G_l^{(W)}\right|}{\left|G_{l+1}^{(A)}\right|}\right)^{1/4}$$

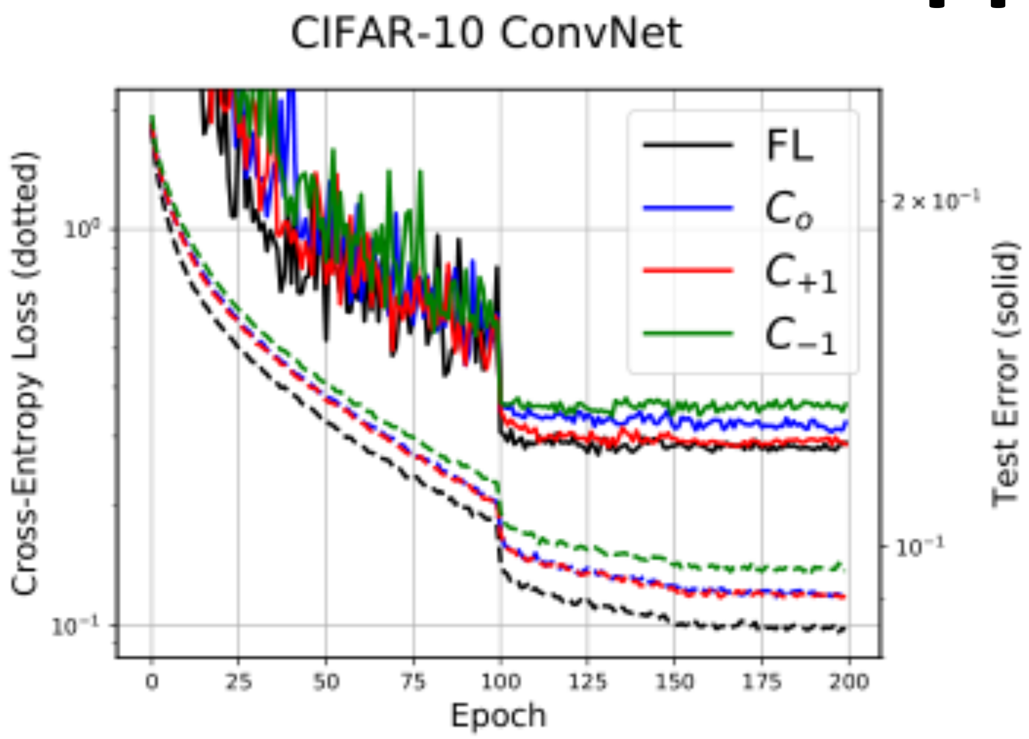
Criterion 5: accumulator stopping condition

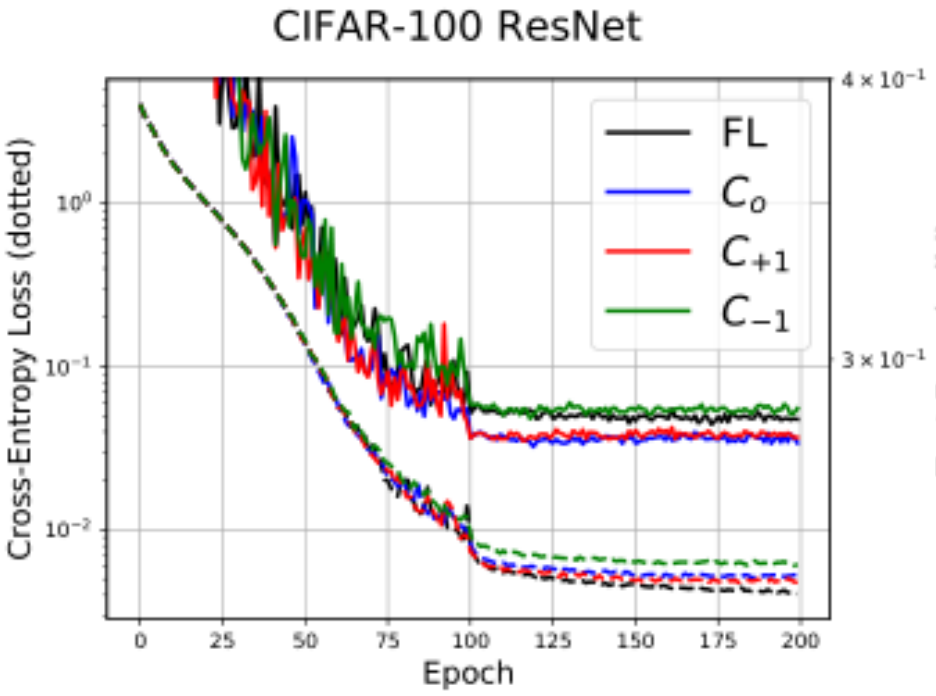
$$r_{W_l^{(acc)}} \geq 2^{-B_{W_l}}$$

$$\Delta_{W_l^{(acc)}} < \gamma^{(\min)}\Delta_{G_l^{(W)}}$$

- analytically guaranteed to provide close-to-minimal precision @ iso-accuracy
- requires a floating-point network to be probed during training

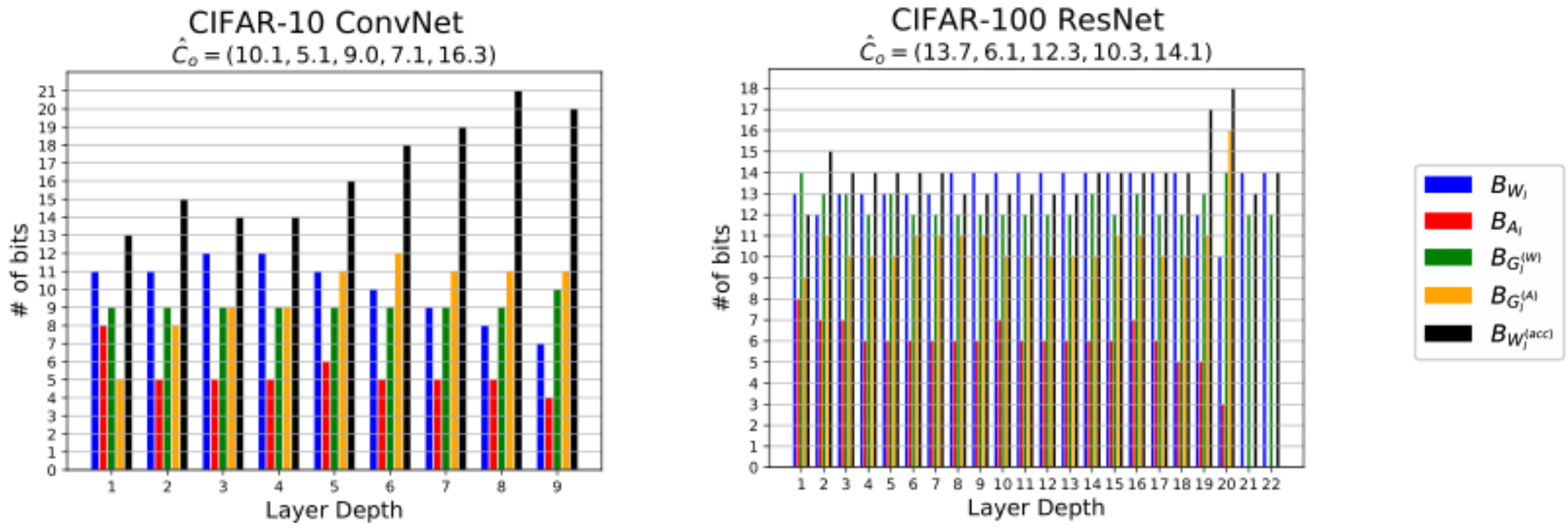
FX Training Converges with Close-to-Minimal Precisions





- FX training was believed to be hard due to dynamic range issues [Koester, NeurIPS’2017]
- proposed FX training is able to match FL training accuracy
- precision assignment found to be nearly minimal

Per-layer Precision Trends Vary



- weight precision decreases from network input to output
- precisions of activation gradients and weight accumulators increase
- ResNets have uniform precision requirements per tensor type

Comparison w.r.t. Hyper-precision Reduction Techniques

	$\mathcal{C}_W$ (10^6b)	$\mathcal{C}_A$ (10^6b)	$\mathcal{C}_M$ (10^9FA)	$\mathcal{C}_C$ (10^6b)	Test Error	$\mathcal{C}_W$ (10^6b)	$\mathcal{C}_A$ (10^6b)	$\mathcal{C}_M$ (10^9FA)	$\mathcal{C}_C$ (10^6b)	Test Error
	CIFAR-10 ConvNet					**CIFAR-100 ResNet**				
FL	148	9.3	94.4	49	12.02%	1789	97	4319	597	28.06%
FX (C_o)	**56.5**	**1.7**	11.9	14	12.58%	**750**	**25**	776	216	**27.43%**
BN	100	4.7	**2.8**	49	18.50%	1211	50	**128**	597	29.35%
SQ	78.8	**1.7**	11.9	14	**11.32%**	1081	**25**	776	216	28.03%
TG	102	9.3	94.4	**3.1**	12.49%	1230	97	4319	**37.3**	30.62%

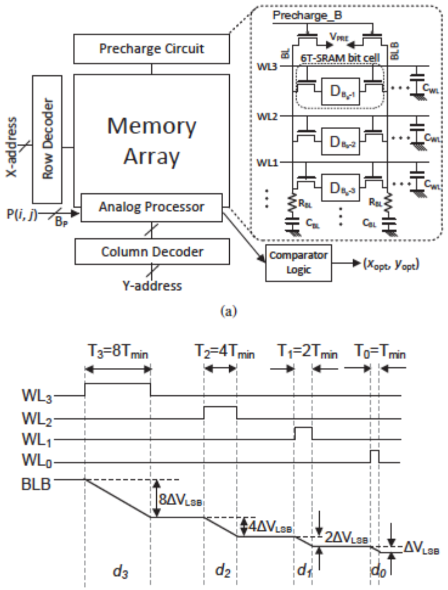
- feedforward binarization (BN) and gradient ternarization (TG) fail to match FL accuracy for same topology
- stochastic quantization (SQ) provides marginal returns
- BN, TG, SQ do not address the fundamental problem of realizing true FX training

Precision Requirements for In-Memory Architectures

UIUC 6T SRAM Deep In-memory IC Prototypes

100X EDP reduction over von Neumann equivalent* @ iso-accuracy

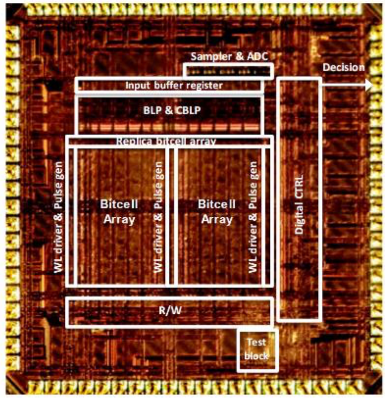
concept



[ICASSP'14] (with Micron)

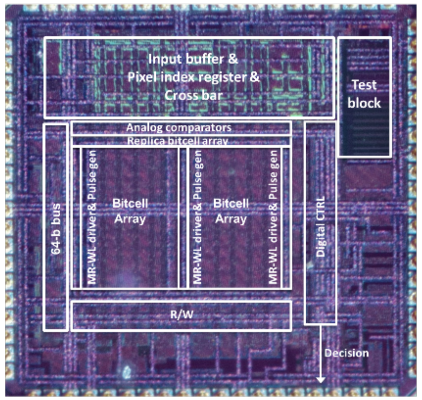
8b compute; 16kB SRAM in 65nm CMOS (UIUC)

multi-functional



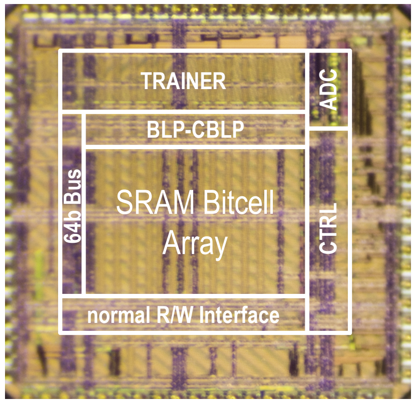
[arxiv'16, JSSC'18]

random forest



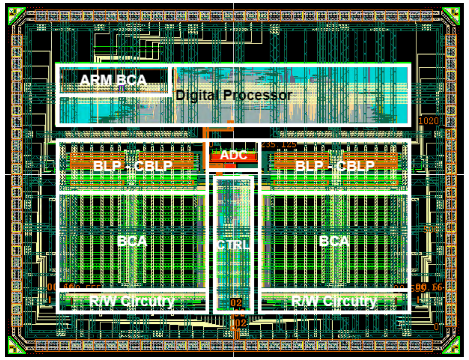
[ESSCIRC'17 JSSC'18]

on-chip learning



[ISSCC'18, JSSC'18]

RNN attention ntwk



[CICC'20]

23 Mar 2018 | 15:00 GMT

To Speed Up AI, Mix Memory and Processing

New computing architectures aim to extend artificial intelligence from the cloud to smartphones

By **Katherine Bourzac**

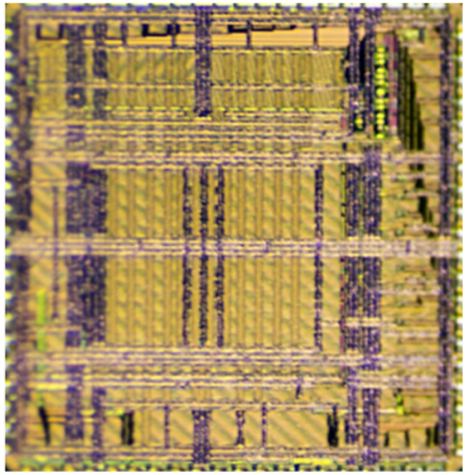


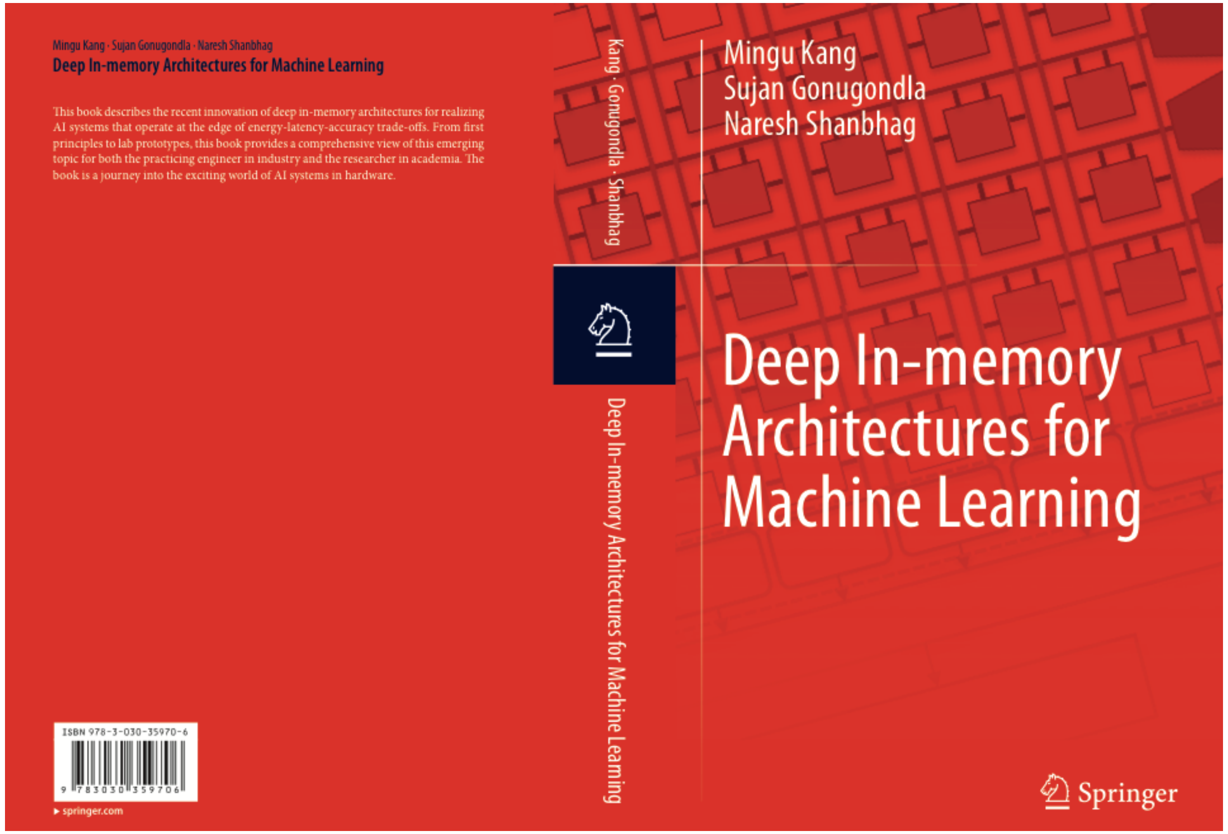
Image: Sujan Gonugondla

Tearing Down Walls: This prototype features a new chip design called deep in-memory architecture.

If John von Neumann were designing a computer today, there's no way he would build a thick wall between processing and memory. At least, that's what computer engineer Naresh Shanbhag of the University of Illinois at Urbana-Champaign believes. The eponymous von Neumann architecture was published in 1945. It enabled the first stored-memory, reprogrammable computers—and it's been the backbone of the industry ever since.

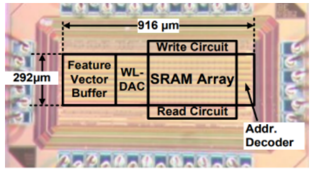
Now, Shanbhag thinks it's time to switch to a design that's better suited for today's data-intensive tasks. In February, at the International Solid-State Circuits Conference (ISSCC), in San Francisco, he and others made their case for a new architecture that brings computing and memory closer together. The idea is not to replace the processor altogether but to add new functions to the memory that will make devices smarter without requiring more power.

The Deep In-memory Architecture (DIMA)

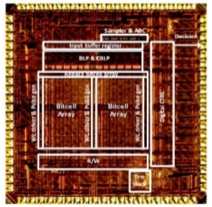


https://spectrum.ieee.org/computing/hardware/to-speed-up-ai-mix-memory-and-processing

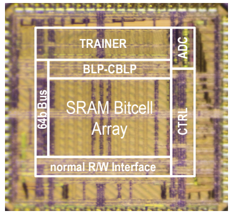
In-memory ICs for Machine Learning is Hot!



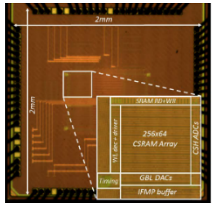
[VLSI'16]



[JSSC'18]

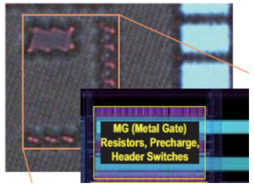


[ISSCC'18]

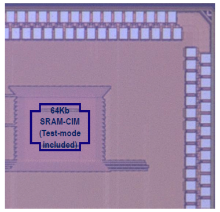


[ISSCC'18]

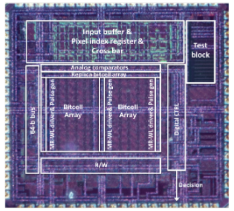
[ISSCC'18]



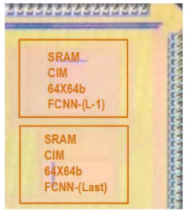
[VLSI '19]



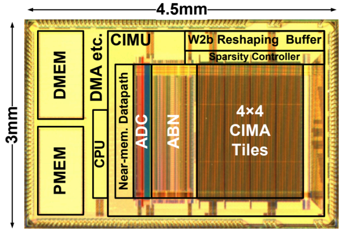
[ISSCC'20]



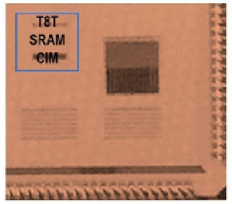
[ESSCIRC'17]



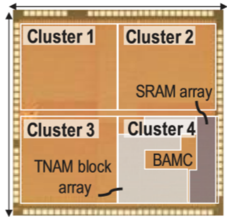
[ISSCC'18]



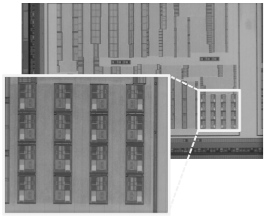
[arxiv'19]



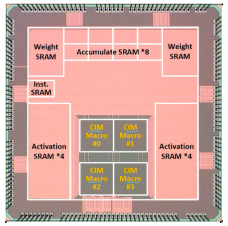
[ISSCC'19]



[VLSI'19]

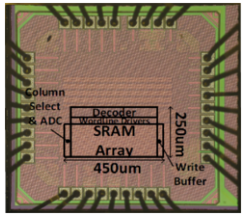


[ISSCC'20]

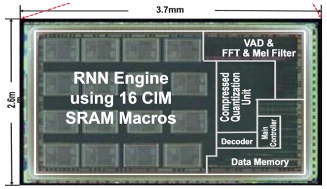


[ISSCC'20]

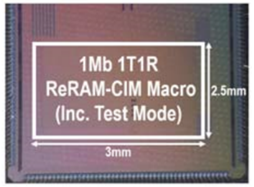
[VLSI'18]



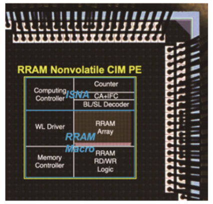
[VLSI'18]



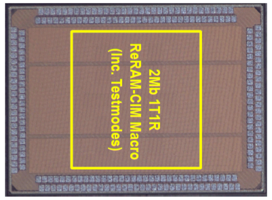
[VLSI '19]



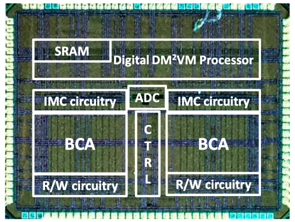
[ISSCC '19]



[VLSI '19]



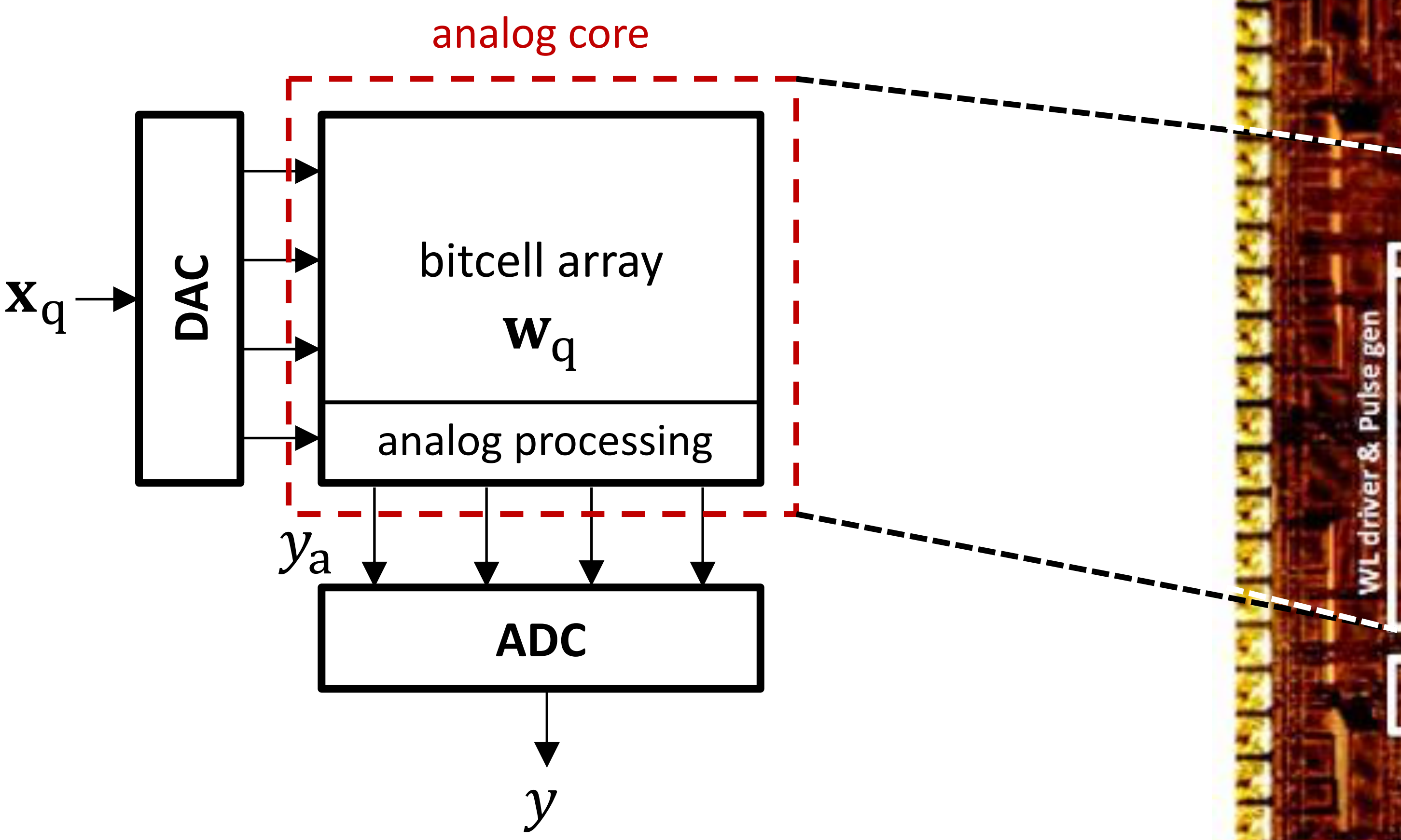
[ISSCC'20]

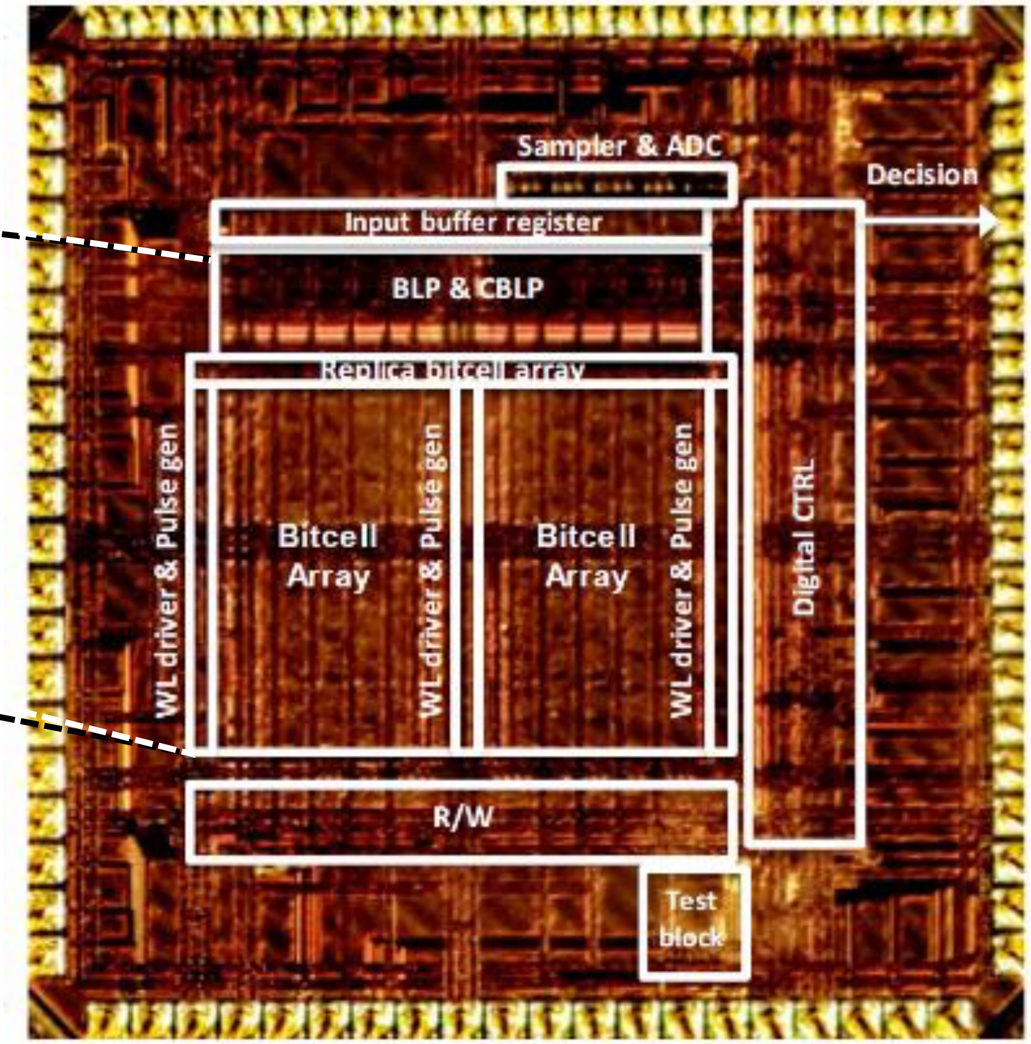


[CICC'20]

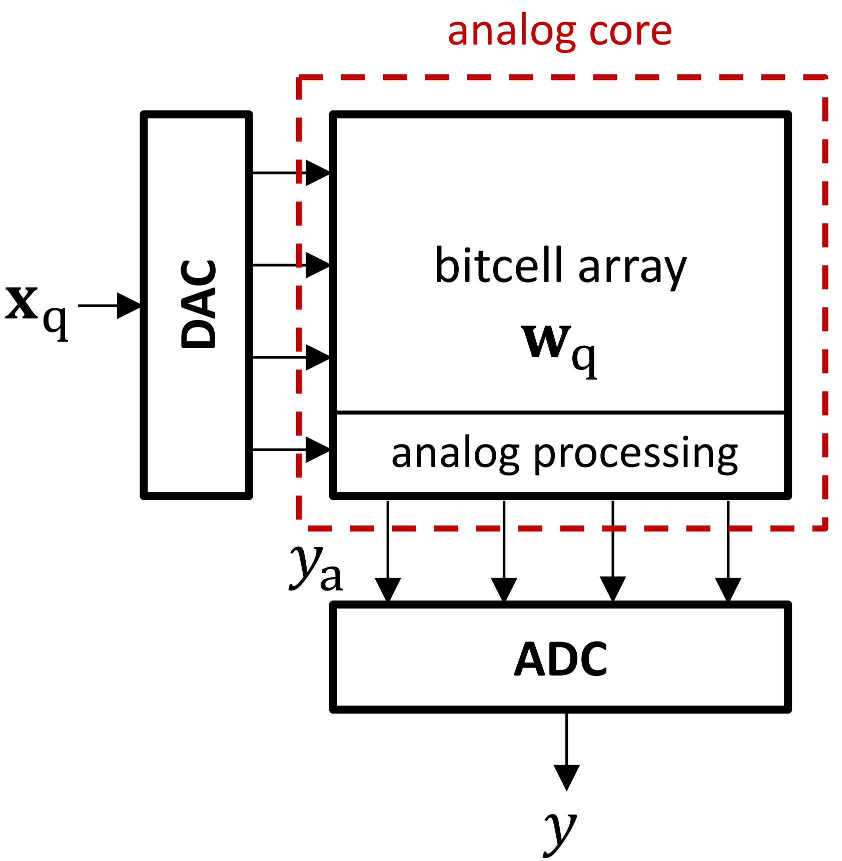
Research Questions

- Is there a common basis for these architectures?
- **What are their precision (compute SNR) limits?** (BIG? for IMCs today)

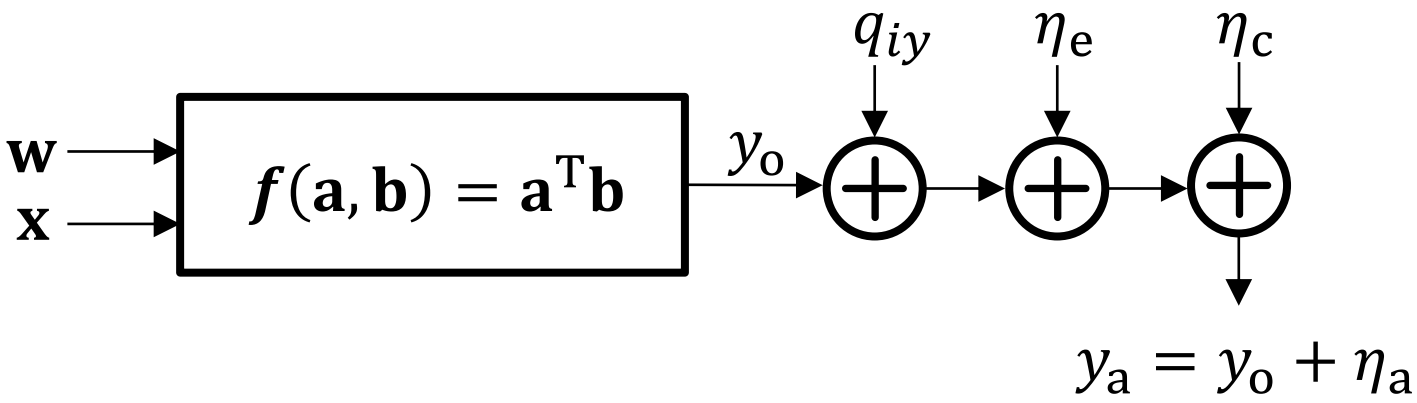

analog core
DAC
$\mathbf{x}_\mathrm{q}$
bitcell array
$\mathbf{w}_\mathrm{q}$
analog processing
y_a
ADC
y


Sampler & ADC
Decision
Input buffer register
BLP & CBLP
Replica bitcell array
WL driver & Pulse gen
Bitcell Array
WL driver & Pulse gen
Bitcell Array
WL driver & Pulse gen
Digital CTRL
R/W
Test block

Fixed-Point Dot Product on IMCs

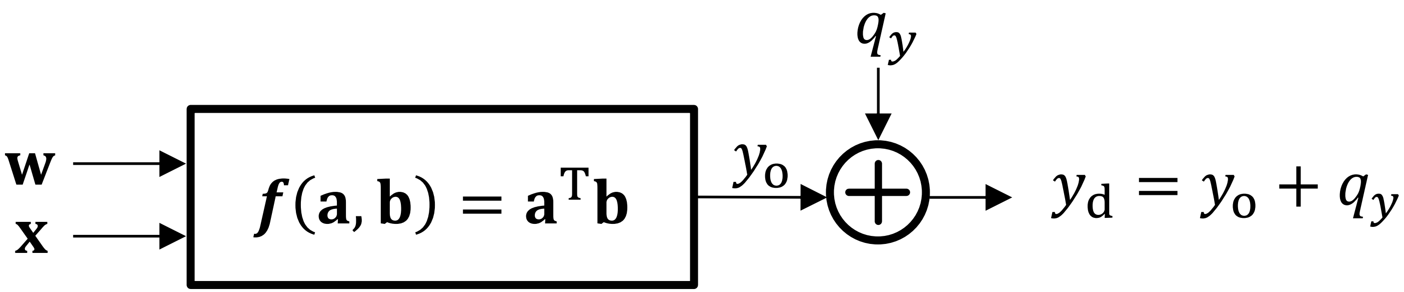


$$y = \overbrace{\overbrace{\mathbf{w}^\mathrm{T}\mathbf{x}}^{y_o}}^{\text{ideal FL output}} + q_{iy} + \underbrace{\eta_e}_{\text{spatio-temporal analog noise sources}} + \overbrace{\eta_c}^{\text{clipping due to headroom limitations}} + q_y$$



$$SNR_{\mathrm{a}} = \frac{\sigma_{y_o}^2}{\sigma_{\eta_a}^2}$$

(analog SNR)



$$SNR_{\mathrm{d}} = \frac{\sigma_{y_o}^2}{\sigma_{\eta_a}^2}$$

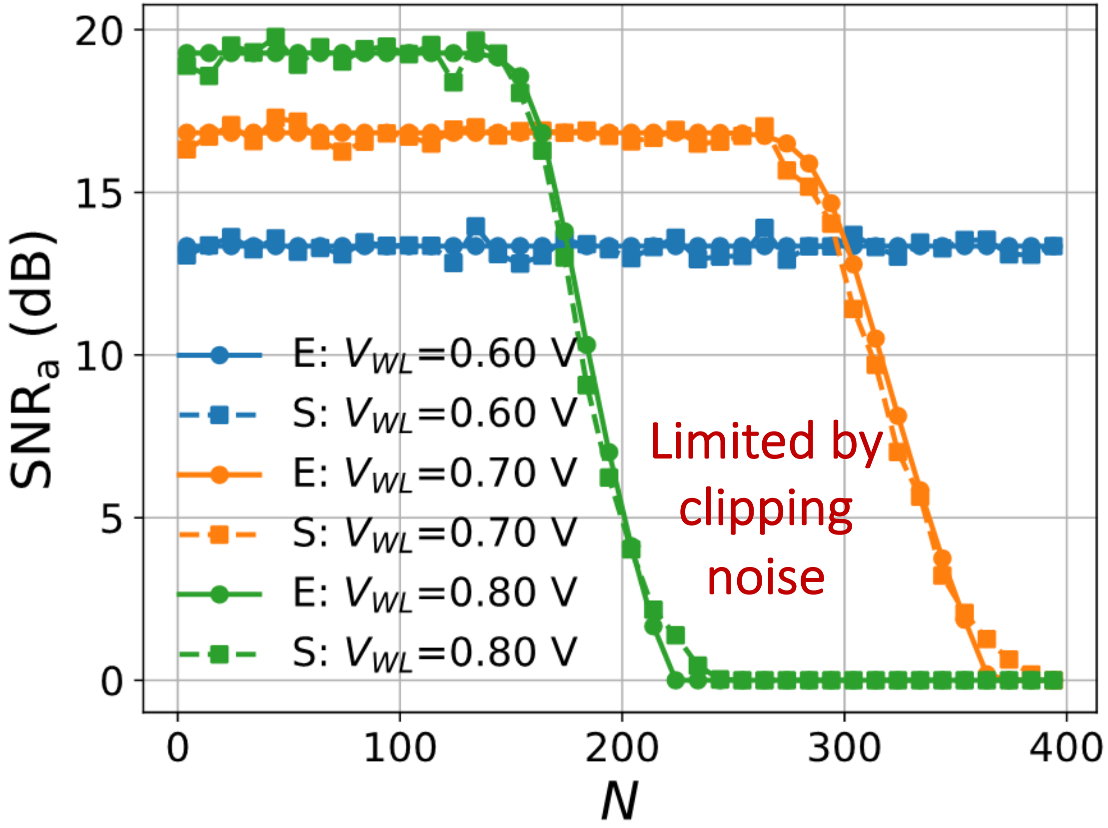
(digitization SNR)

$$\boxed{SNR_{\mathrm{T}} = \left[\frac{1}{SNR_{\mathrm{a}}} + \frac{1}{SNR_{\mathrm{d}}}\right]^{-1}} \longrightarrow \boxed{\text{Limited by } SNR_{\mathrm{a}}}$$

SNR Tradeoffs in Charge Summing Architectures

limited by circuit noise

$B_x = 6, B_w = 6$



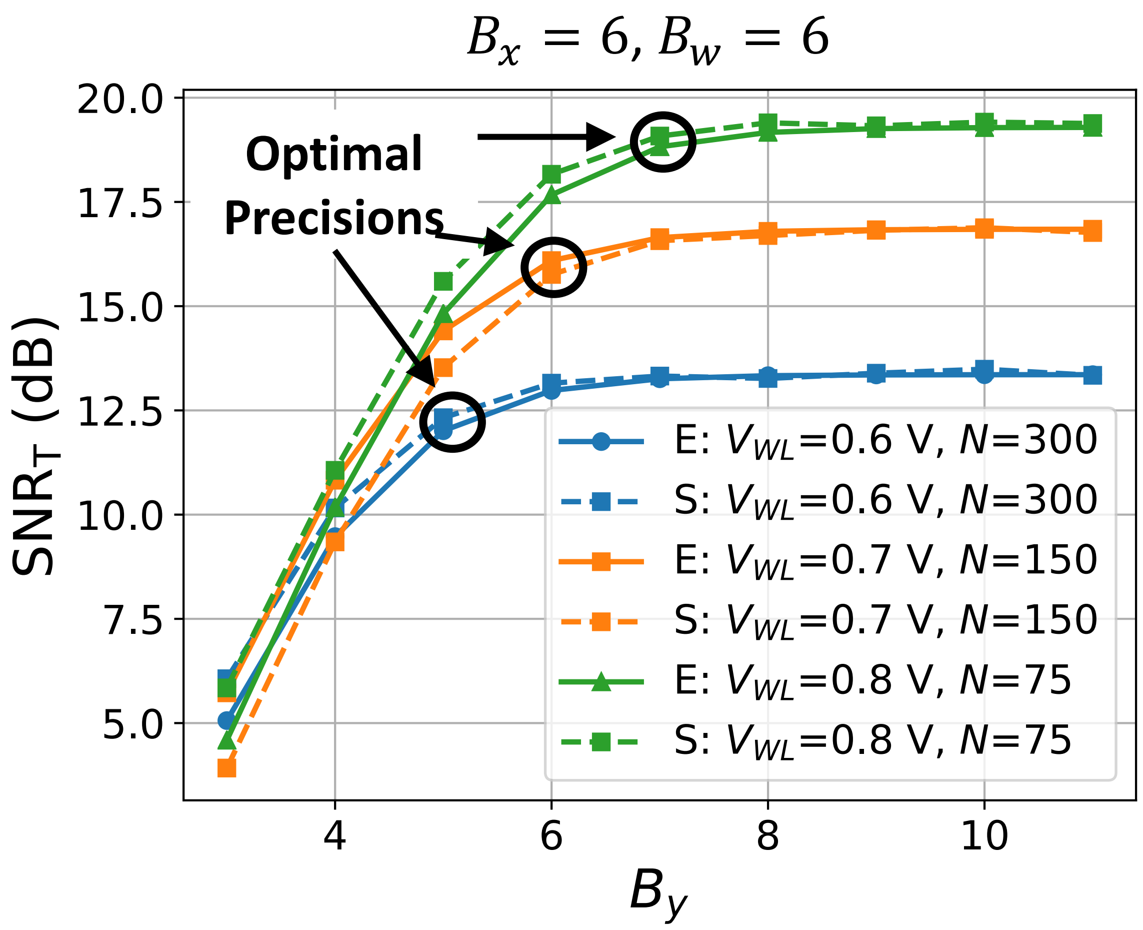
E: analytical expression S: simulations

$$\mathrm{SNR_a} = \frac{12\sigma_w^2 E[x^2]}{E[x^2]\Delta_w^2 + \sigma_w^2\Delta_x^2 + E[x^2]\frac{D_c}{4}\frac{\sigma_I^2}{I_o^2} + \sigma_w^2\frac{D_c}{4}\frac{\sigma_T^2}{T_o^2} + \frac{1}{N}\sigma_{\eta_c}^2}$$

- discharge current and pulse width trades-off with clipping noise
- clipping noise dominates as dimensionality N increases

SNR trades off with N and V_{WL}

ADC Precision Requirements



E: analytical expression S: simulations

- precision limited by $\mathrm{SNR_a}$

$$\mathrm{SNR_a(dB)} - \mathrm{SNR_T(dB)} \leq 0.5\ \mathrm{dB}$$

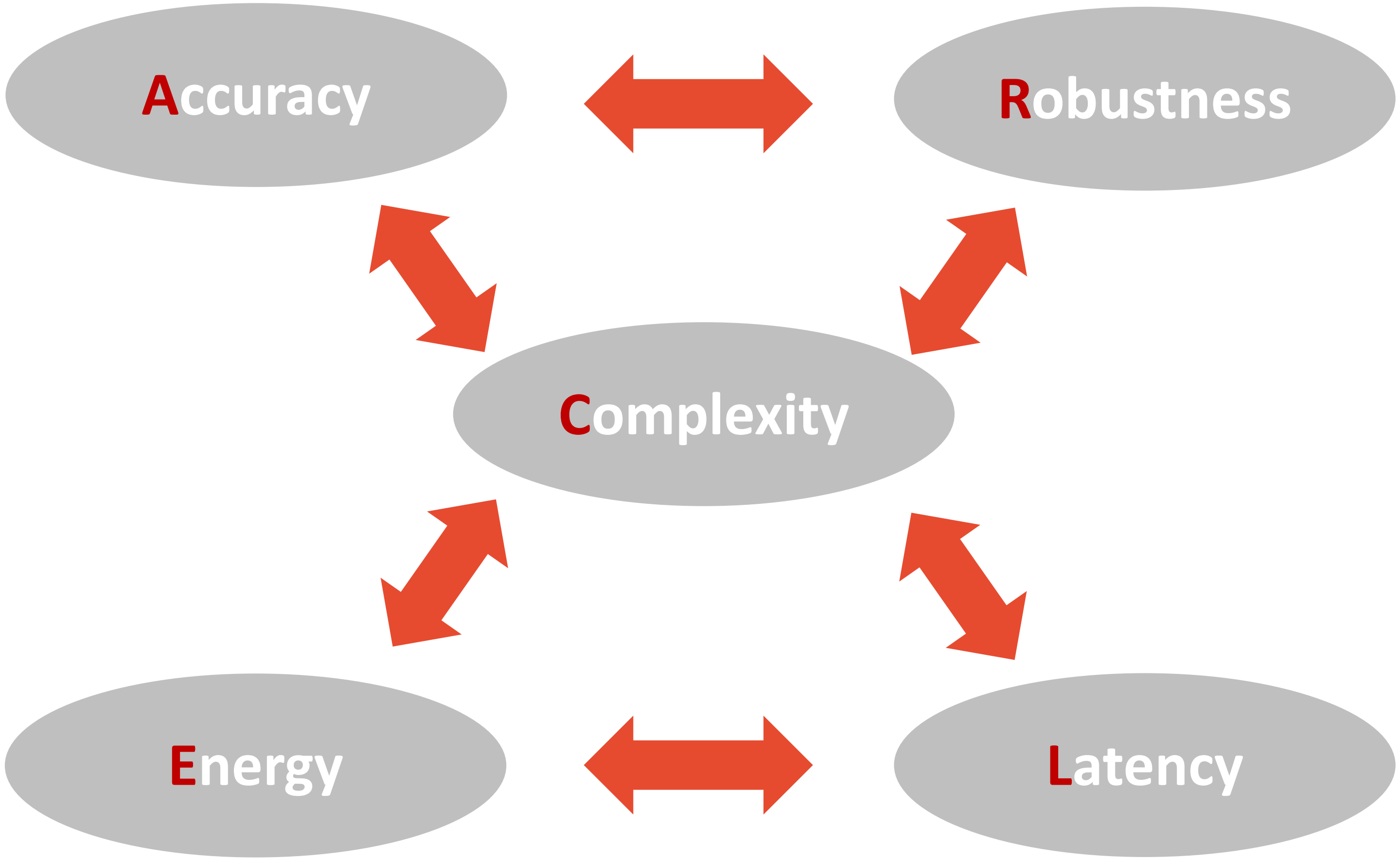
$$B_y > \min\left(\log_2(k_{\text{clip}}), \frac{\mathrm{SNR_a(dB)}+16.6}{6}\right)$$

Summary

- Design of DNNs need not be trial-&-error based - analytical methods exist (for precision assignment) or can be developed
- use MPC & noise gain analysis to determine minimum precisions of DNNs
- parallel considerations for in-memory architectures – interplay between analog and quantization noise sources
- Next: design optimization framework? network accuracy vs. energy vs. latency vs….

Major challenge – engineered design of AI systems →

composability, interpretability, robustness, security, ethics, with guarantees



Machine Learning with Guarantees

Thank You!

http://shanbhag.ece.uiuc.edu