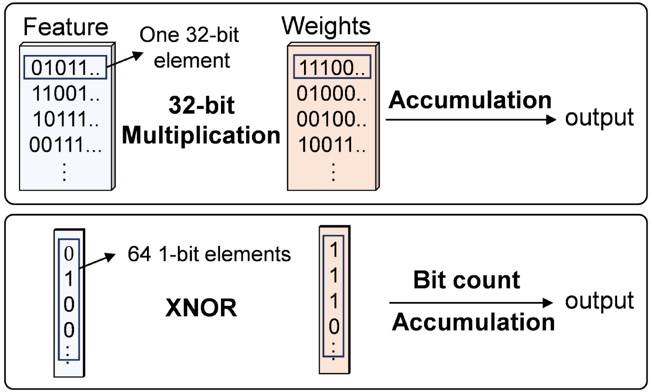
Heterogenous Bitwidth Binarization: Weird Operators with Big Benefits

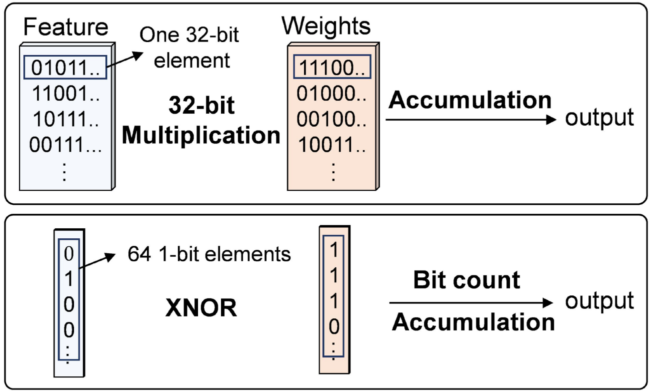
Josh Fromm

Network Binarization

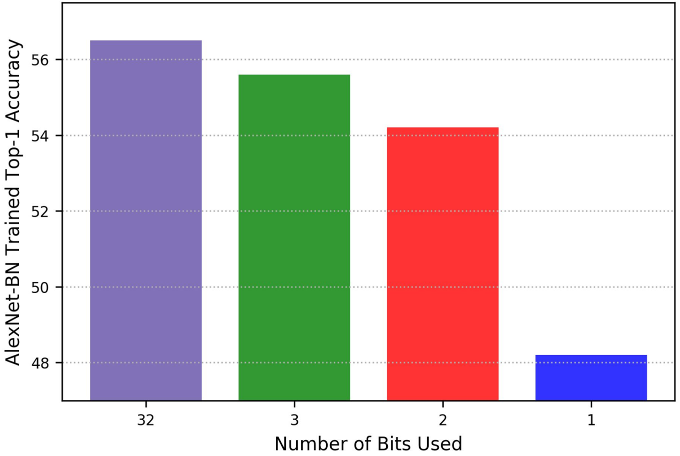


- Multiply-accumulate becomes xnor-popcount.
- 5-30x theoretical speedup.
- 32x weight memory compression.

Network Binarization



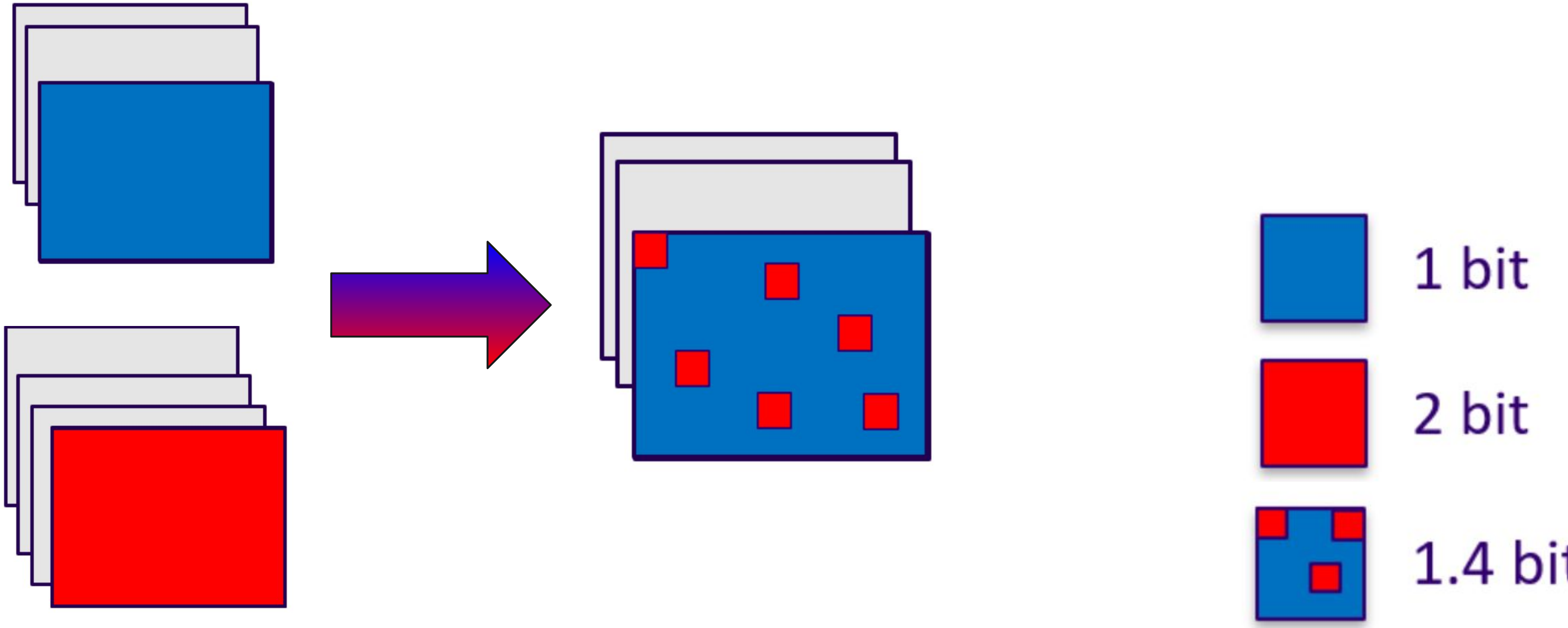
- Multiply-accumulate becomes xnor-popcount.
- 5-30x theoretical speedup.
- 32x weight memory compression.



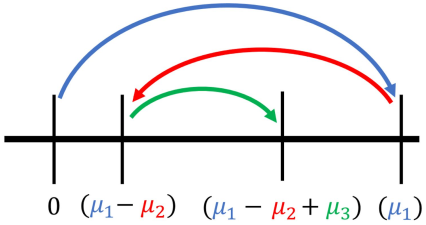
- 1-bit accuracy is too low but fast.
- 2-bit accuracy is high but too slow.
- How to bridge the gap?

Mixed Bitwidth Tensors

Mixed Bitwidth Tensors

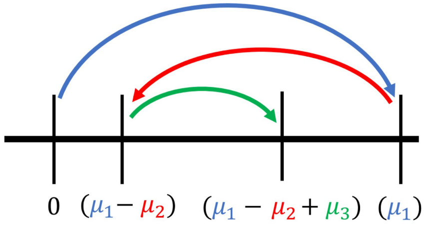


Middle-Out Bit Distribution



$$\boldsymbol{T}^{B}_{[1,2,3]} = [+1, -1, +1]$$
$$\boldsymbol{\mu}_{[1,2,3]} = [\mu_1, \mu_2, \mu_3]$$

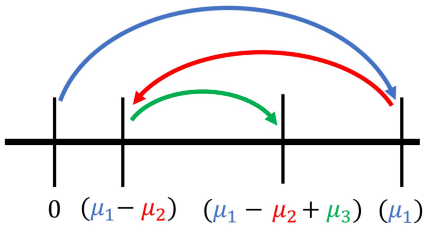
Middle-Out Bit Distribution



$$\boldsymbol{T}^{B}_{[1,2,3]} = [+1, -1, +1]$$
$$\boldsymbol{\mu}_{[1,2,3]} = [\mu_1, \mu_2, \mu_3]$$

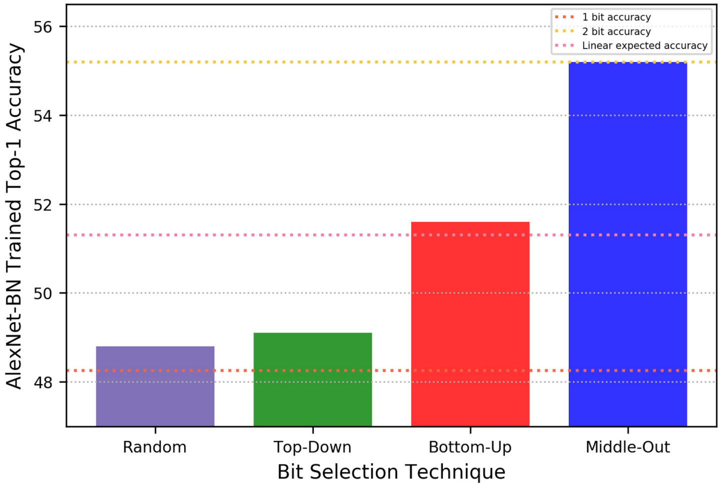
$$\mathrm{TD}(T) = \mathrm{sort}(|T|, \text{descending})$$
$$\mathrm{MO}(T) = \mathrm{sort}(|T| - \mathrm{mean}(|T|), \text{ascending})$$
$$\mathrm{BU}(T) = \mathrm{sort}(|T|, \text{ascending})$$
$$\mathrm{R}(T) = \text{a fixed uniformly random permutation of } T$$

Middle-Out Bit Distribution

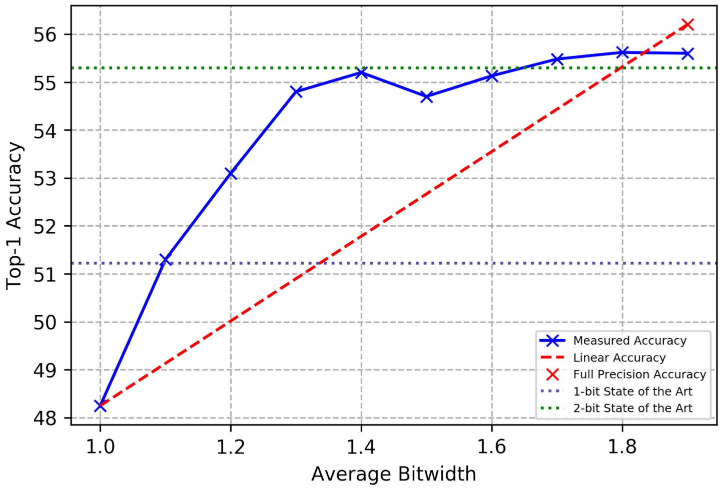


$$\boldsymbol{T}^B_{[1,2,3]} = [+1, -1, +1]$$
$$\boldsymbol{\mu}_{[1,2,3]} = [\mu_1, \mu_2, \mu_3]$$

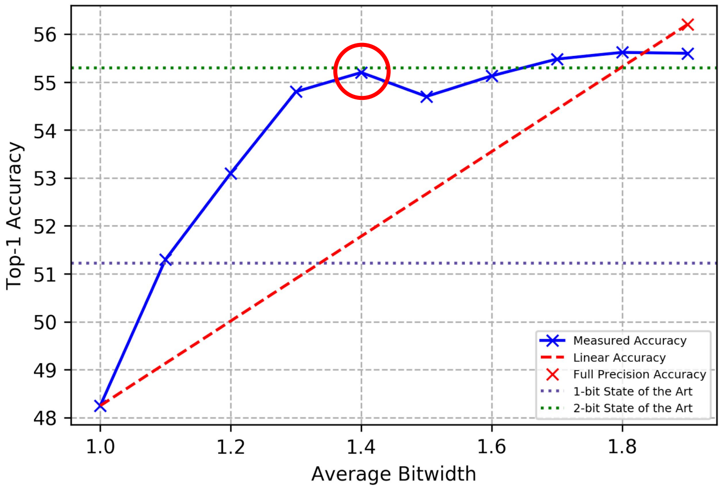
$$\mathrm{TD}(T) = \mathrm{sort}(|T|, \mathrm{descending})$$
$$\mathrm{MO}(T) = \mathrm{sort}(|T| - \mathrm{mean}(|T|), \mathrm{ascending})$$
$$\mathrm{BU}(T) = \mathrm{sort}(|T|, \mathrm{ascending})$$
$$\mathrm{R}(T) = \text{a fixed uniformly random permutation of } T$$



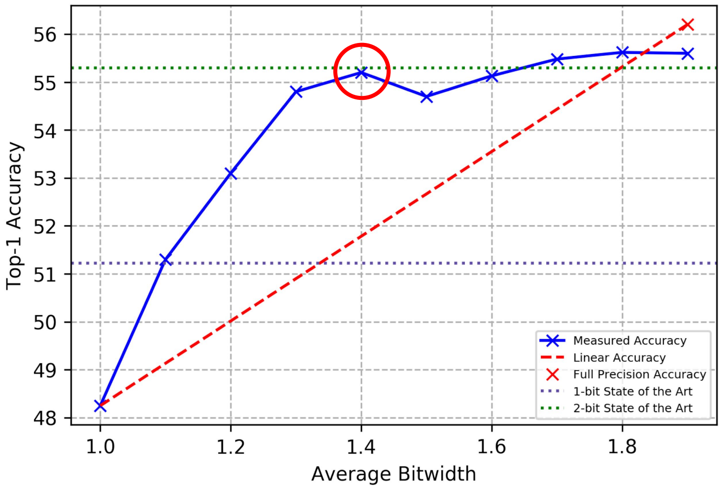
Super-Linear Scaling



Super-Linear Scaling



Super-Linear Scaling



	Model	Name	Binarization (Inputs / Weights)	Top-1	Top-5
		Binarized weights with floating point activations			
1	AlexNet	SQ-BWN (Dong et al., 2017)	full precision / 1-bit	51.2%	75.1%
2	AlexNet	SQ-TWN (Dong et al., 2017)	full precision / 2-bit	55.3%	78.6%
3	AlexNet	TWN (our implementation)	full precision / 1-bit	48.3%	71.4%
4	AlexNet	TWN	full precision / 2-bit	54.2%	77.9%
5	AlexNet	HBNN (our results)	full precision / 1.4-bit	55.2%	78.4%
6	MobileNet	HBNN	full precision / 1.4-bit	65.1%	87.2%
		Binarized weights and activations excluding input and output layers			
7	AlexNet	BNN (Courbariaux et al., 2015)	1-bit / 1-bit	27.9%	50.4%
8	AlexNet	Xnor-Net (Rastegari et al., 2016)	1-bit / 1-bit	44.2%	69.2%
9	AlexNet	DoReFaNet (Zhou et al., 2016)	2-bit / 1-bit	50.7%	72.6%
10	AlexNet	QNN (Hubara et al., 2016)	2-bit / 1-bit	51.0%	73.7%
11	AlexNet	our implementation	2-bit / 2-bit	52.2%	74.5%
12	AlexNet	our implementation	3-bit / 3-bit	54.2%	78.1%
13	AlexNet	HBNN	1.4-bit / 1.4-bit	53.2%	77.1%
14	AlexNet	HBNN	1-bit / 1.4-bit	49.4%	72.1%
15	AlexNet	HBNN	1.4-bit / 1-bit	51.5%	74.2%
16	AlexNet	HBNN	2-bit / 1.4-bit	52.0%	74.5%
17	MobileNet	our implementation	1-bit / 1-bit	52.9%	75.1%
18	MobileNet	our implementation	2-bit / 1-bit	61.3%	80.1%
19	MobileNet	our implementation	2-bit / 2-bit	63.0%	81.8%
20	MobileNet	our implementation	3-bit / 3-bit	65.9%	86.7%
21	MobileNet	HBNN	1-bit / 1.4-bit	60.1%	78.7%
22	MobileNet	HBNN	1.4-bit / 1-bit	62.0%	81.3%
23	MobileNet	HBNN	1.4-bit / 1.4-bit	64.7%	84.9%
24	MobileNet	HBNN	2-bit / 1.4-bit	63.6%	82.2%
		Unbinarized (our implementation)			
25	AlexNet	(Krizhevsky et al., 2012)	full precision / full precision	56.5%	80.1%
26	MobileNet	(Howard et al., 2017)	full precision / full precision	68.8%	89.0%

Hard to Implement!

Implementing on CPU

- Needs efficient sparse tensor library support

Implementing on FPGA

- Gates can be directly laid out for big benefits
- Designing FPGAs is hard, especially for non-uniform computation

TVM can enable these platforms!