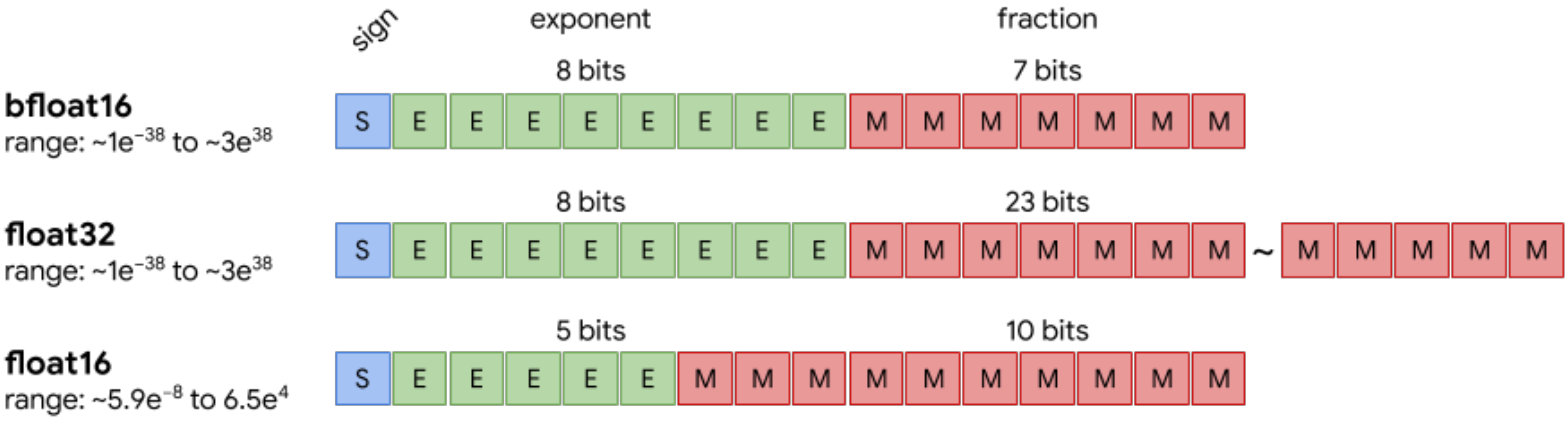
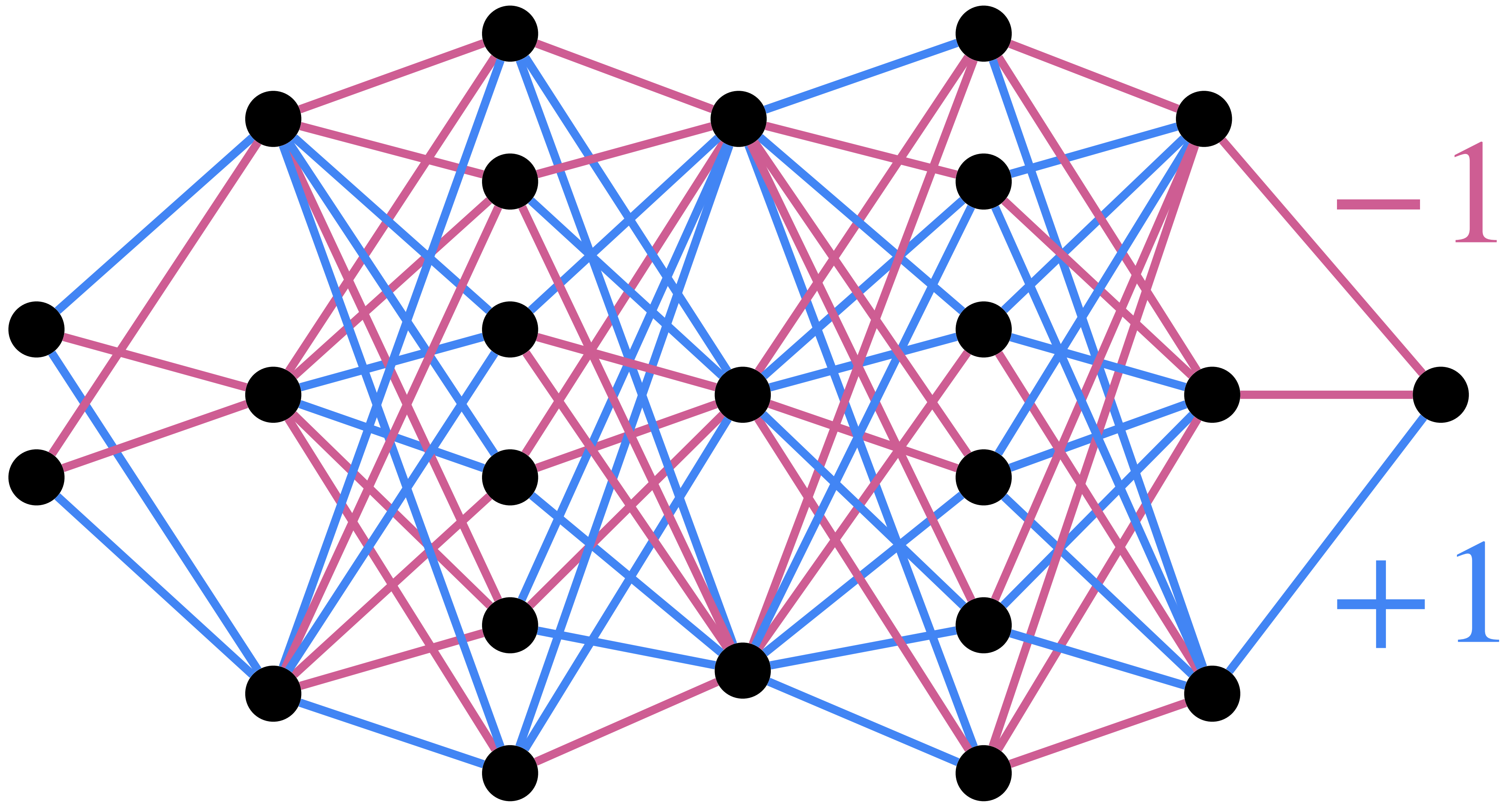
How efficient and expressive are Binary Neural Networks

Standard approach to precision



Binary Neural Networks

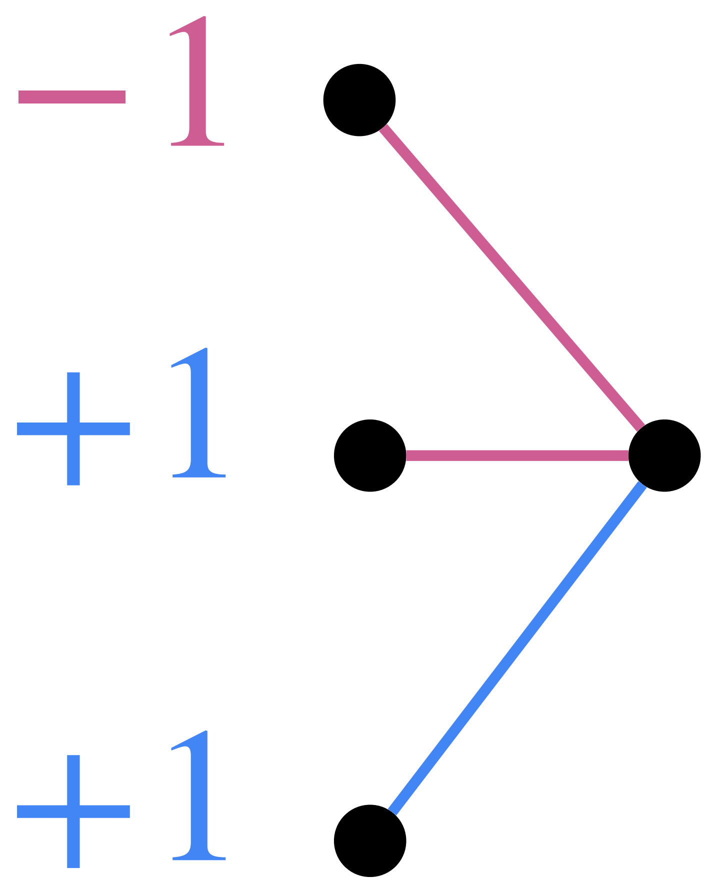
- A lot of recent work since 2016
- Several benefits:
    - Memory/Storage/Comm/Compute
    - Energy



- Typically suffer from accuracy loss
- Learning algorithms are a bit too heuristic
- Theoretical results very very limited (expressivity/algorithmic aspects)

Multiplication => XNOR + bitcount

A	B	XNOR(A,B)
0 (-1)	0 (-1)	1 (+1)
0 (-1)	1 (+1)	0 (-1)
1(+1)	0 (-1)	0 (-1)
1 (+1)	1 (+1)	1 (+1)



$$2 \cdot \textbf{popcount}\ (\ \textbf{XNOR}(-1, -1);\ \ \textbf{XNOR}(1, -1);\ \ \textbf{XNOR}(1,1)\ ) - 3$$

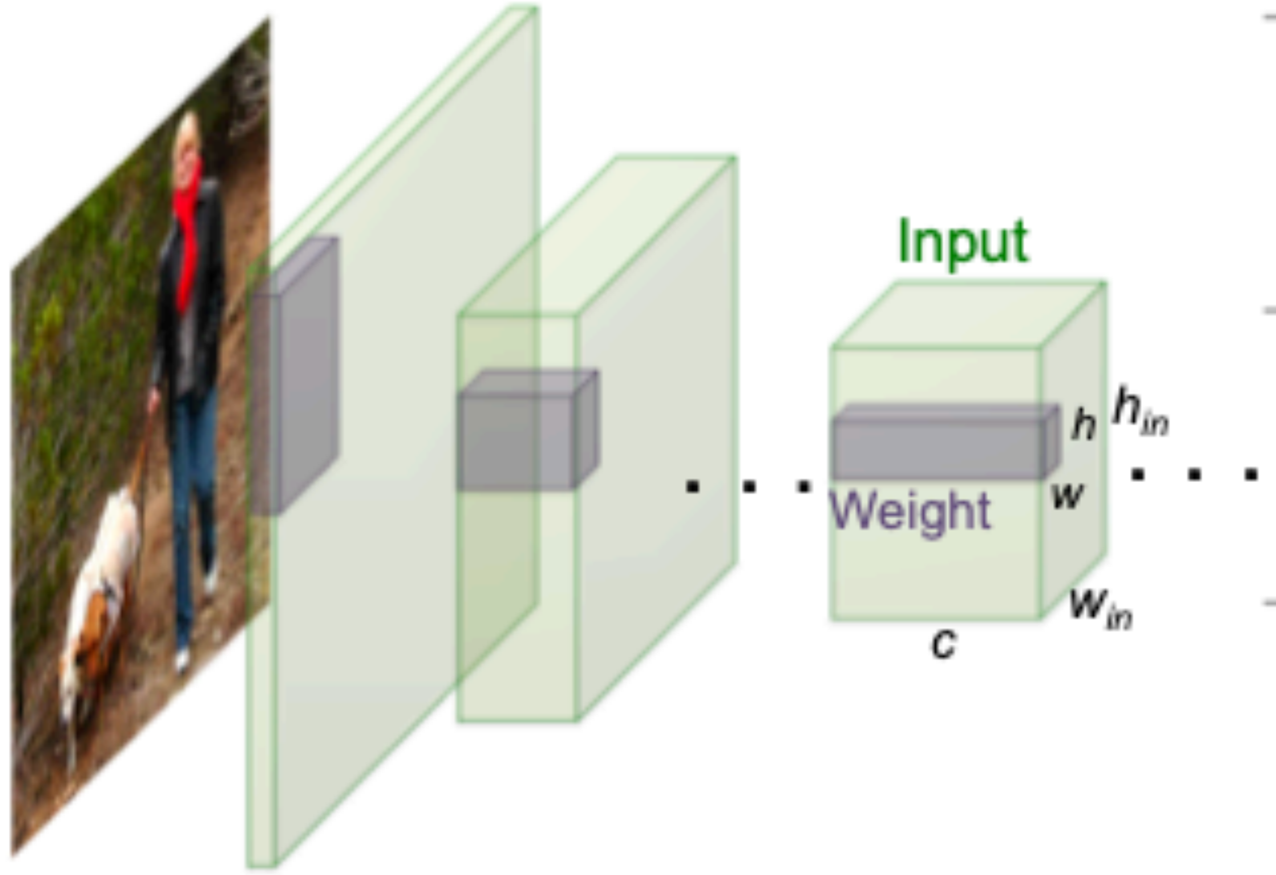
Some ways to Binarize Neural Nets

XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks

Mohammad Rastegari[†], Vicente Ordonez[†], Joseph Redmon[*], Ali Farhadi[†*]

Allen Institute for AI[†], University of Washington[*]
{mohammadr,vicenteor}@allenai.org
{pjreddie,ali}@cs.washington.edu

XNOR-Net



	Network Variations		Operations used in Convolution	Memory Saving (Inference)	Computation Saving (Inference)	Accuracy on ImageNet (AlexNet)
Standard Convolution	Real-Value Inputs 0.11 -0.21 ... -0.34 -0.25 0.61 ... 0.52	Real-Value Weights 0.12 -1.2 ... 0.41 -0.2 0.5 ... 0.68	+ , − , ×	1x	1x	%56.7
Binary Weight	Real-Value Inputs 0.11 -0.21 ... -0.34 -0.25 0.61 ... 0.52	Binary Weights 1 -1 ... 1 -1 1 ... 1	+ , −	~32x	~2x	%56.8
BinaryWeight Binary Input (**XNOR-Net**)	Binary Inputs 1 -1 ... -1 -1 1 ... 1	Binary Weights 1 -1 ... 1 -1 1 ... 1	XNOR , bitcount	~32x	~58x	%44.2

- Binary-Weight-Nets: conv filters are only +1/-1
- XNOR-Nets: input AND filter is binary

How to binarize a network?

Goal:
Find the best binary network that approximates original

- We hope that $\mathbf{W} * \mathbf{X} \approx a \cdot \mathbf{B} * \mathbf{X}$
- For some ± 1 matrix $\mathbf{B}$

How to binarize a network?

- Method: for a given layer, and a given matrix, F, the best binary matrix B is given by
- $$\min_{a\in\mathbb{R},B_{i,j}\in\{-1,1\}} \|\mathbf{W} * \mathbf{X} - \alpha\mathbf{B} * \mathbf{X}\|_F^2 \equiv \min_{a\in\mathbb{R},B_{i,j}\in\{-1,1\}} \|\mathbf{W} - \alpha\mathbf{B}\|_F^2$$

[Rastegari, Ordonez, Redmon, Farhadi, 2016]

How to binarize a network?

$$\min_{a \in \mathbb{R}, B_{i,j} \in \{-1,1\}} \|\mathbf{W} - \alpha \mathbf{B}\|_F^2$$

$$\equiv \min_{a \in \mathbb{R}, B_{i,j} \in \{-1,1\}} \|\mathbf{W}\|_F^2 - 2\alpha \cdot \text{trace}\{\mathbf{W}^T \mathbf{B}\} + \alpha^2 \|\mathbf{B}\|_F^2$$

How to binarize a network?

$$\min_{a \in \mathbb{R}, B_{i,j} \in \{-1,1\}} \|\mathbf{W} - \alpha \mathbf{B}\|_F^2$$

$$\equiv \min_{a \in \mathbb{R}, B_{i,j} \in \{-1,1\}} \|\mathbf{W}\|_F^2 - 2\alpha \cdot \mathrm{trace}\{\mathbf{W}^T \mathbf{B}\} + \alpha^2 \|\mathbf{B}\|_F^2$$

$$\equiv \min_{a \in \mathbb{R}, B_{i,j} \in \{-1,1\}} - 2\alpha \cdot \mathrm{trace}\{\mathbf{W}^T \mathbf{B}\} + \alpha^2 \|\mathbf{B}\|_F^2$$

How to binarize a network?

$$\min_{a\in\mathbb{R}, B_{i,j}\in\{-1,1\}} \|\mathbf{W} - \alpha\mathbf{B}\|_F^2$$

$$\equiv \min_{a\in\mathbb{R}, B_{i,j}\in\{-1,1\}} \|\mathbf{W}\|_F^2 - 2\alpha \cdot \mathsf{trace}\{\mathbf{W}^T\mathbf{B}\} + \alpha^2\|\mathbf{B}\|_F^2$$

$$\equiv \min_{a\in\mathbb{R}, B_{i,j}\in\{-1,1\}} - 2\alpha \cdot \mathsf{trace}\{\mathbf{W}^T\mathbf{B}\} + \alpha^2\|\mathbf{B}\|_F^2$$

$$\equiv \min_{a\in\mathbb{R}} \left(\left\{ \min_{B_{i,j}\in\{-1,1\}} - 2\alpha \cdot \mathsf{trace}\{\mathbf{W}^T\mathbf{B}\} \right\} + \alpha^2 \cdot N \right)$$

How to binarize a network?

$$\min_{a \in \mathbb{R}, B_{i,j} \in \{-1,1\}} \|\mathbf{W} - \alpha \mathbf{B}\|_F^2$$

$$\equiv \min_{a \in \mathbb{R}, B_{i,j} \in \{-1,1\}} \|\mathbf{W}\|_F^2 - 2\alpha \cdot \mathrm{trace}\{\mathbf{W}^T\mathbf{B}\} + \alpha^2 \|\mathbf{B}\|_F^2$$

$$\equiv \min_{a \in \mathbb{R}, B_{i,j} \in \{-1,1\}} - 2\alpha \cdot \mathrm{trace}\{\mathbf{W}^T\mathbf{B}\} + \alpha^2 \|\mathbf{B}\|_F^2$$

$$\equiv \min_{a \in \mathbb{R}} \left(\left\{ \min_{B_{i,j} \in \{-1,1\}} - 2\alpha \cdot \mathrm{trace}\{\mathbf{W}^T\mathbf{B}\} \right\} + \alpha^2 \cdot N \right)$$

$$\equiv \min_{a \in \mathbb{R}} \left(-2\alpha \left\{ \max_{B_{i,j} \in \{-1,1\}} \mathrm{vec}(\mathbf{W})^T \mathrm{vec}(\mathbf{B}) \right\} + \alpha^2 \cdot N \right)$$

How to binarize a network?

$$\min_{a\in\mathbb{R},B_{i,j}\in\{-1,1\}} \|\mathbf{W}-\alpha\mathbf{B}\|_F^2$$

$$\equiv \min_{a\in\mathbb{R},B_{i,j}\in\{-1,1\}} \|\mathbf{W}\|_F^2 - 2\alpha\cdot\mathrm{trace}\{\mathbf{W}^T\mathbf{B}\} + \alpha^2\|\mathbf{B}\|_F^2$$

$$\equiv \min_{a\in\mathbb{R},B_{i,j}\in\{-1,1\}} - 2\alpha\cdot\mathrm{trace}\{\mathbf{W}^T\mathbf{B}\} + \alpha^2\|\mathbf{B}\|_F^2$$

$$\equiv \min_{a\in\mathbb{R}}\left(\left\{\min_{B_{i,j}\in\{-1,1\}} - 2\alpha\cdot\mathrm{trace}\{\mathbf{W}^T\mathbf{B}\}\right\} + \alpha^2\cdot N\right)$$

$$\equiv \min_{a\in\mathbb{R}}\left(-2\alpha\left\{\max_{B_{i,j}\in\{-1,1\}} \mathrm{vec}(\mathbf{W})^T\mathrm{vec}(\mathbf{B})\right\} + \alpha^2\cdot N\right)$$

$$\equiv \min_{a\in\mathbb{R}}\left(-2\alpha\left\{\max_{\mathbf{b}\in\{-1,1\}^N} \mathbf{w}^T\mathbf{b}\right\} + \alpha^2\cdot N\right)$$

Complexity of binarization

- Optimal solution of $\min_{a \in \mathbb{R}, B_{i,j} \in \{-1,1\}} \|\mathbf{W} - \alpha \mathbf{B}\|_F^2$ is

[Rastegari, Ordonez, Redmon, Farhadi, 2016]

Complexity of binarization

- Optimal solution of $\min_{a \in \mathbb{R}, B_{i,j} \in \{-1,1\}} \|\mathbf{W} - \alpha \mathbf{B}\|_F^2$ is

$$\mathbf{B}^* = \mathrm{sign}(\mathbf{W}) \quad \text{and} \quad \alpha^* = \frac{\mathrm{trace}(\mathbf{W}^T \mathbf{B}^*)}{N}$$

[Rastegari, Ordonez, Redmon, Farhadi, 2016]

Complexity of binarization

- Optimal solution of $\min_{a \in \mathbb{R}, B_{i,j} \in \{-1,1\}} \|\mathbf{W} - \alpha \mathbf{B}\|_F^2$ is

$$\mathbf{B}^* = \mathrm{sign}(\mathbf{W}) \quad \text{and} \quad \alpha^* = \frac{\mathrm{trace}(\mathbf{W}^T \mathbf{B}^*)}{N}$$

- Computable in linear time in number of weights.

[Rastegari, Ordonez, Redmon, Farhadi, 2016]

What if you want to Binarize Inputs too?

- We would hope that $\mathbf{W} * \mathbf{X} \approx a \cdot \mathbf{B} * \mathbf{Z}$ where

$$\min_{a \in \mathbb{R}, B_{i,j}, Z_{i,j} \in \{-1,1\}} \|\mathbf{W} * \mathbf{X} - \alpha \mathbf{B} * \mathbf{Z}\|_F^2$$

- Similar, but a bit more involved solution for this too

Backprop for BW-Net

- How do we train?
- Forward pass: binarize weights, and compute loss
- Backward pass: replace grad of $\nabla \text{sign}(w)$ function with $w\mathbf{1}_{|w|<1}$ and follow chain rule
- Parameter update: use floats
- XNOR-net backprop a little trickier but similar

XNOR-Net: Efficiency Experiments

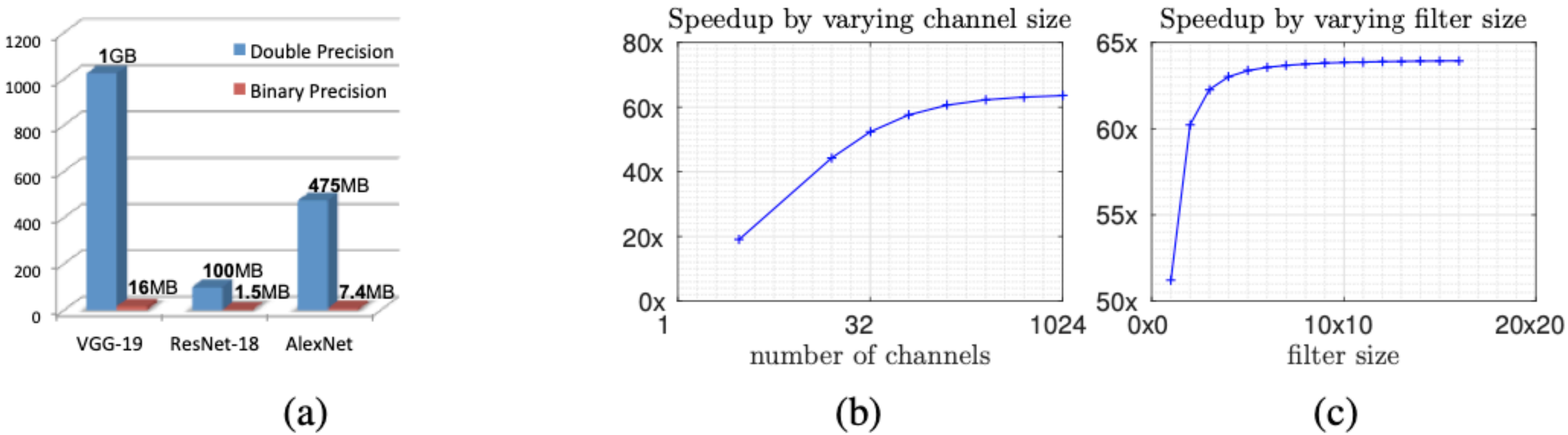
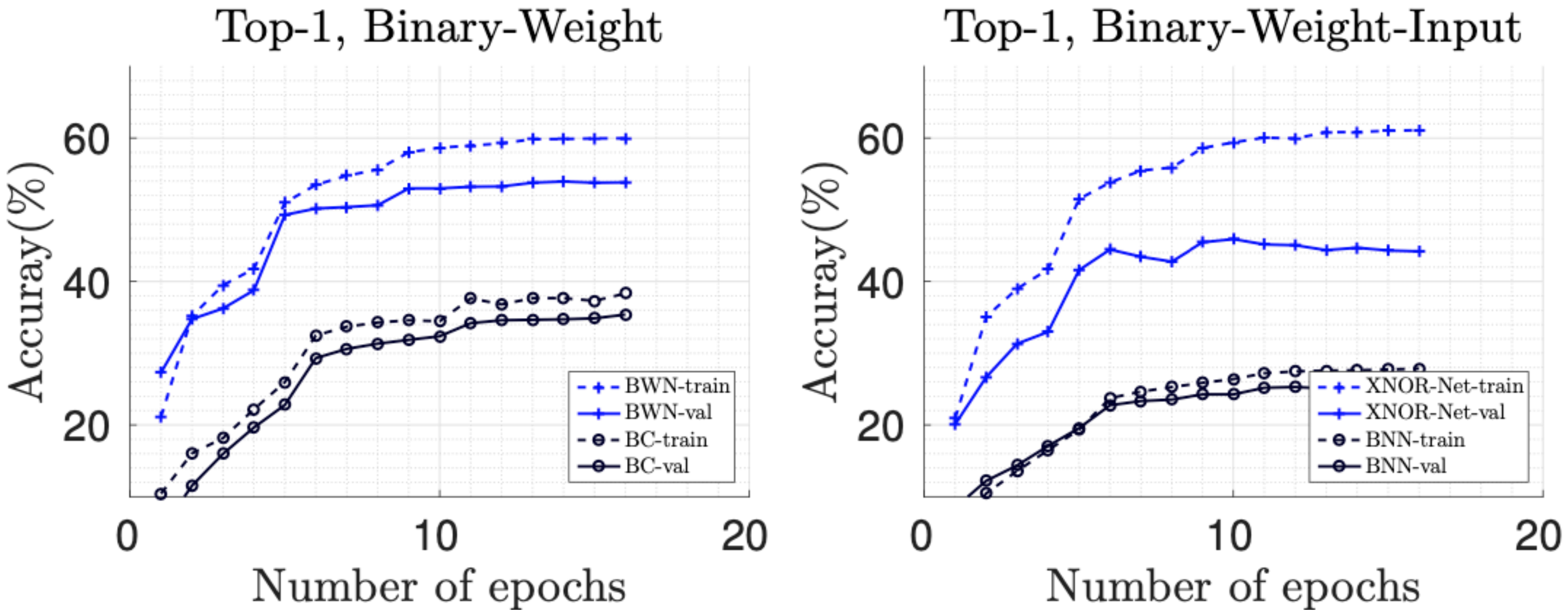


Fig. 4: This figure shows the efficiency of binary convolutions in terms of memory(a) and computation(b-c). (a) is contrasting the required memory for binary and double precision weights in three different architectures(AlexNet, ResNet-18 and VGG-19). (b,c) Show speedup gained by binary convolution under (b)-different number of channels and (c)-different filter size

XNOR-Net: Accuracy Experiments



Alexnet on ImageNet
BC and BNN SOTA at the point.

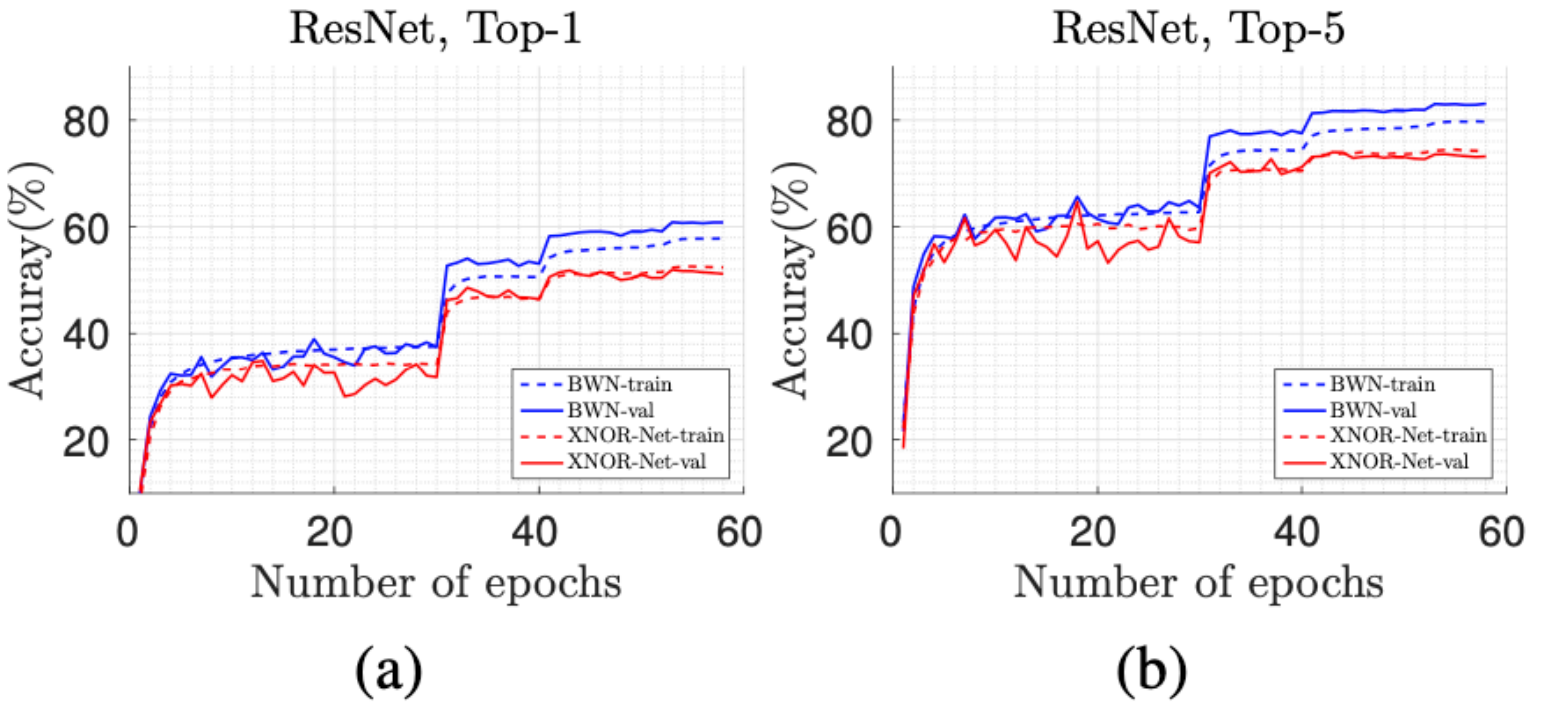
XNOR-Net: Accuracy Experiments

Classification Accuracy(%)									
Binary-Weight				Binary-Input-Binary-Weight				Full-Precision	
BWN		BC[11]		XNOR-Net		BNN[11]		AlexNet[1]	
Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5	Top-1	Top-5
56.8	**79.4**	35.4	61.0	**44.2**	**69.2**	27.9	50.42	56.6	80.2

Table 1: This table compares the final accuracies (Top1 - Top5) of the full precision network with our binary precision networks; Binary-Weight-Networks(BWN) and XNOR-Networks(XNOR-Net) and the competitor methods; BinaryConnect(BC) and BinaryNet(BNN).

Alexnet on ImageNet
BC and BNN SOTA at the point.

XNOR-Net: Accuracy Experiments



ResNet-18 on ImageNet

XNOR-Net: Accuracy Experiments

	ResNet-18		GoogLenet	
Network Variations	top-1	top-5	top-1	top-5
Binary-Weight-Network	60.8	83.0	65.5	86.1
XNOR-Network	51.2	73.2	N/A	N/A
Full-Precision-Network	69.3	89.2	71.3	90.0

XNOR-Net: Experiments

up to 30x speedups (but not for same accuracy)

Easy to binarize algorithm

Networks suitable for edge devices

Semi-current state

electronics

MDPI

Review

A Review of Binarized Neural Networks

Taylor Simons and Dah-Jye Lee *

Electrical and Computer Engineering, Brigham Young University, Provo, UT 84602, USA; taylor.simons@byu.edu

* Correspondence: djlee@byu.edu; Tel.: +1-801-422-5923

check for updates

Received: 14 May 2019; Accepted: 5 June 2019; Published: 12 June 2019

Methodology	Activation	Gain	Multiplicity	Regularization
Original BNN	Sign Function	None	1	None
XNOR-Net	Sign Function	Statistical	1	None
DoReFa-Net	Sign Function	Learned Param.	1	None
Tang et al.	PReLU	Inside PReLU	2	L2
ABC-Net	Sign w/Thresh.	Learned Param.	5	None
BNN+	Sign w/SS_t for STE	Learned Param.	1	L1 and L2

Table 3. Comparison of accuracies on the ImageNet dataset from works presented in this section. Full precision network accuracies are included for comparison as well.

Methodology	Topology	Top-1 Accuracy (%)	Top-5 Accuracy (%)
Original BNN	AlexNet	41.8	67.1
Original BNN	GoogleNet	47.1	69.1
XNOR-Net	AlexNet	44.2	69.2
XNOR-Net	ResNet18	51.2	73.2
DoReFa-Net	AlexNet	43.6	-
Tang et al.	51.4	75.6	
ABC-Net	ResNet18	65.0	85.9
ABC-Net	ResNet34	68.4	88.2
ABC-Net	ResNet50	76.1	92.8
BNN+	AlexNet	46.11	75.70
BNN+	ResNet18	52.64	72.98
Full Precision	AlexNet	57.1	80.2
Full Precision	GoogleNet	71.3	90.0
Full Precision	ResNet18	69.3	89.2
Full Precision	ResNet34	73.3	91.3
Full Precision	ResNet50	76.1	92.8

Recent insights

Published as a conference paper at ICLR 2019

An Empirical study of Binary Neural Networks' Optimisation

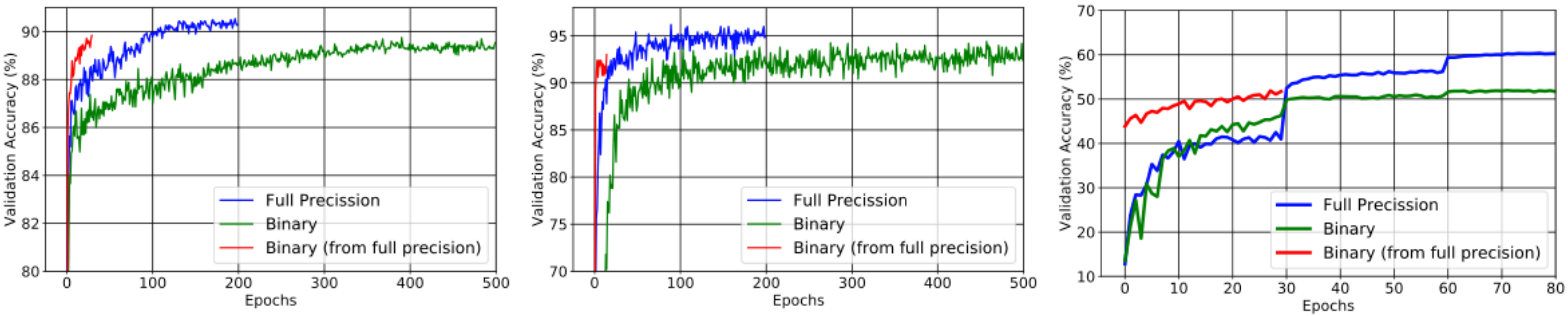
Milad Alizadeh, Javier Fernández-Marqués, Nicholas D. Lane & Yarin Gal
Department of Computer Science
University of Oxford

Binarizing a fully trained model helps

Table 5: Training binary models using pre-trained full-precision models for CIFAR-10 (ResNet-18 and VGG-10) and ImageNet (AlexNet-like) datasets.

	Binarisation	Best Validation Accuracy	Test Accuracy
Binary ResNet-18	end-to-end	94.40% (in epoch 457)	91.16%
	from full-precision	93.60% (in epoch **17**)	91.18%
Binary VGG-10	end-to-end	89.76% (in epoch 391)	89.18%
	from full-precision	90.16% (in epoch **24**)	89.32%
Binary AlexNet-like	end-to-end	51.98% (in epoch 88)	—
	from full-precision	51.85% (in epoch **30**)	—

Binarizing a fully trained model helps



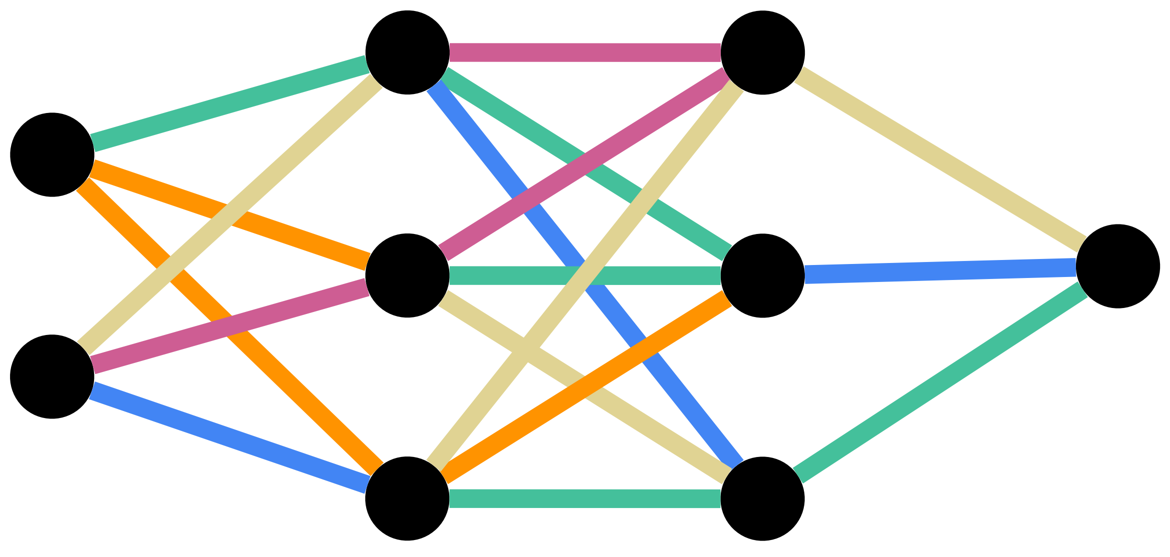
(a) VGG-10 (b) ResNet-18 (c) AlexNet-like on ImageNet

Figure 4: A binary model (red) is initialised from a full precision model (blue) and reaches top accuracy in a fraction of the epochs that would require to train a binary model (green) end-to-end.

Some theoretical bounds

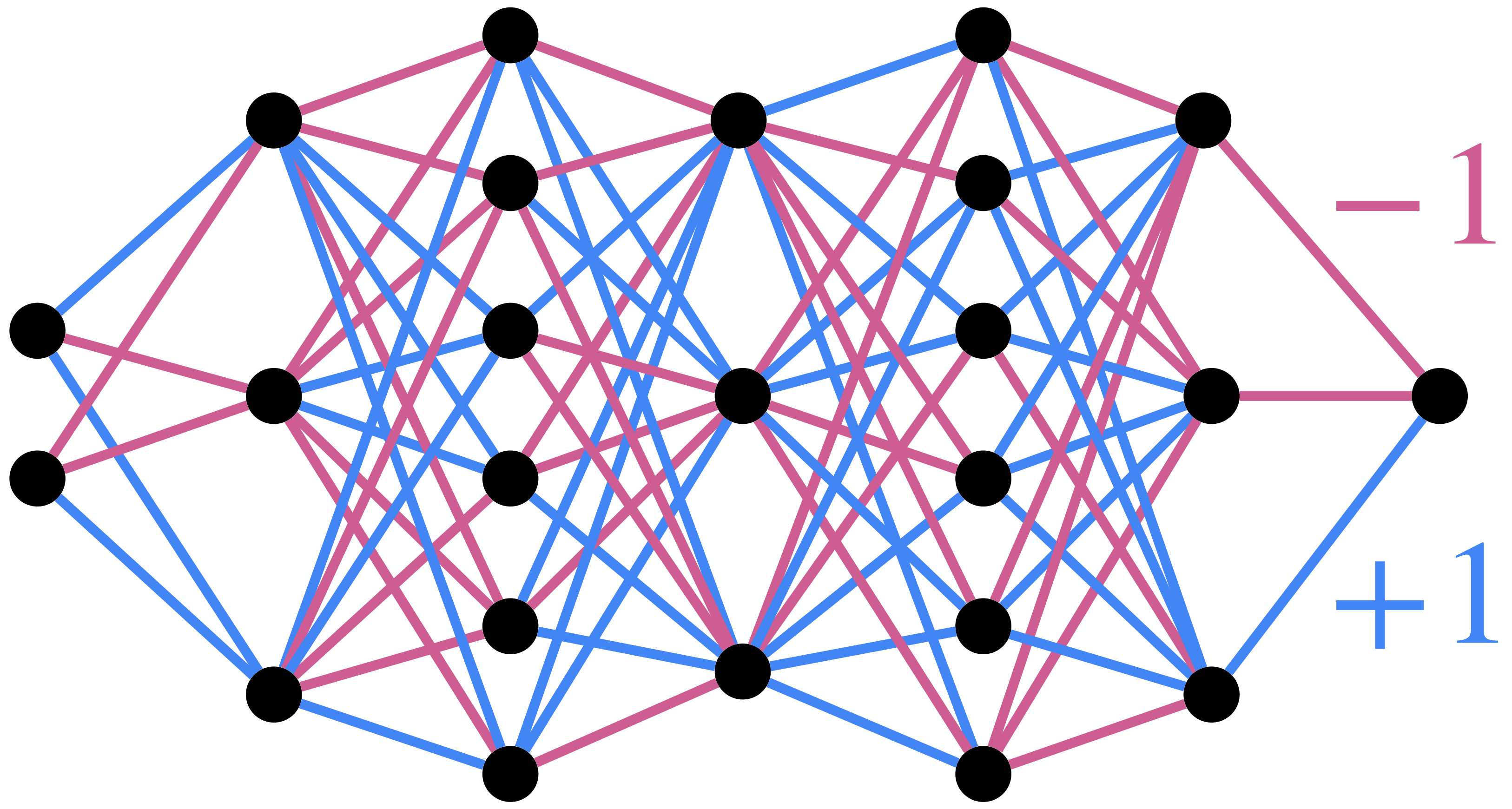
Binary Nets can be Very Expressive

Any network can be approximated
by log-bigger binary network



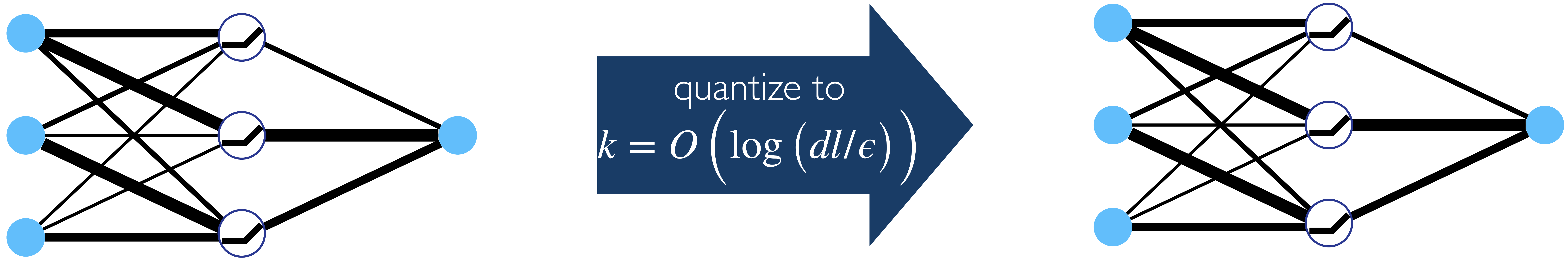
a neural network
with high accuracy

$\approx$



a larger, binary network can approximate it

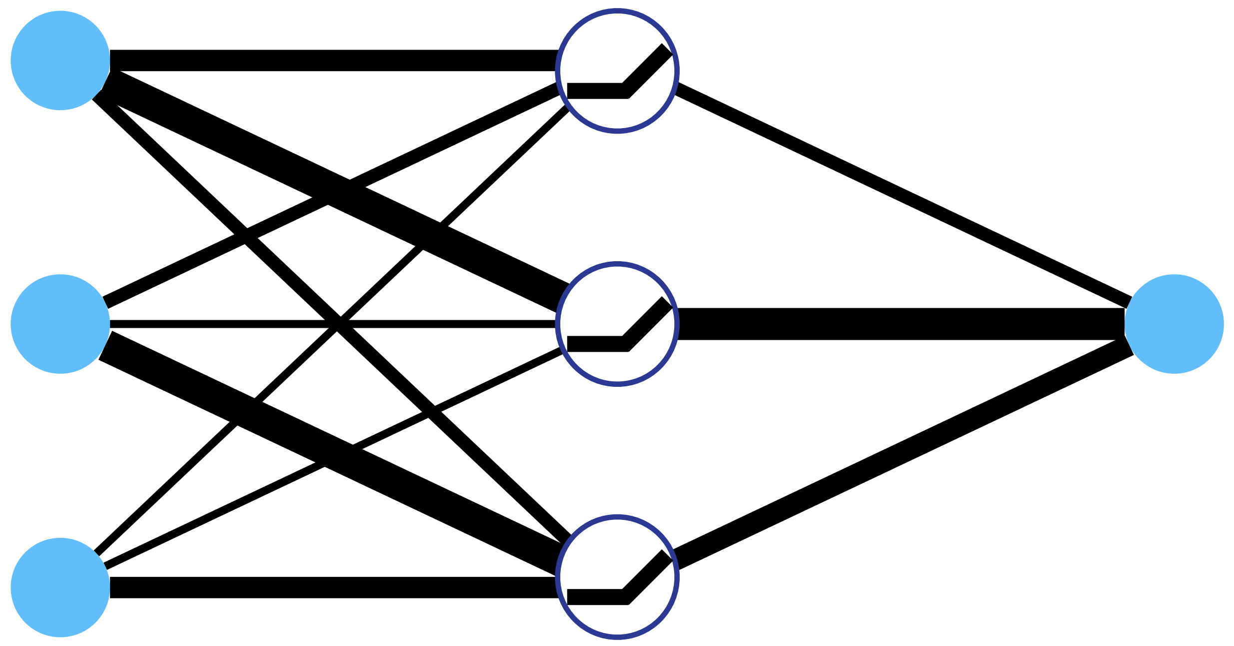
Step 1: Quantizing to Finite Precision



$w = 0.257081639...1... \in [-1,1]$

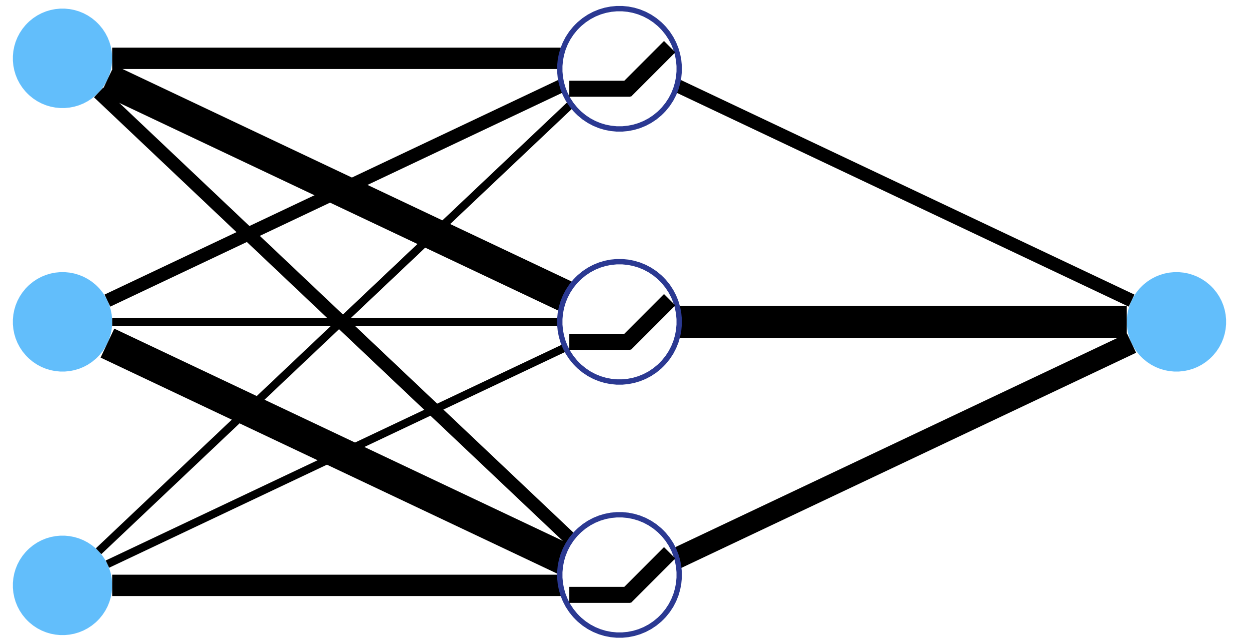
$w = \underbrace{0.257081639...1}_{k \textbf{ digits}} \in \{-1 : 1/\epsilon : 1\}$

Step 1: Quantizing to Finite Precision



$w = 0.257081639...1... \in [-1,1]$

quantize to
$k = O\left(\log\left(dl/\epsilon\right)\right)$



$w = \underbrace{0.257081639...1}_{k\ \textbf{digits}} \in \{-1 : 1/\epsilon : 1\}$

Theorem:
Let $f(x) = \sigma(W_l\sigma(W_{l-1}\ldots\sigma(W_1x)))$ be such that $\|W_i\|_2 \leq 1$. Then, for any $\epsilon > 0$, we can replace the weights of $f(x)$ with a finite precision truncated ones, each represented by $k = O(\log(dl/\epsilon))$ bits such that

$$\max_{\|x\|\leq 1} \|f(x) - f_k(x)\| \leq \epsilon$$

Step 1: Quantizing to Finite Precision

Theorem:
Let $f(x) = \sigma(W_l\sigma(W_{l-1}...\sigma(W_1x)))$ be such that $\|W_i\|_2 \leq 1$. Then, for any $\epsilon > 0$, we can replace the weights of $f(x)$ with a finite precision truncated ones, each represented by $k = O(\log(dl/\epsilon))$ bits such that

$$\max_{\|x\|\leq 1} \|f(x) - f_k(x)\| \leq \epsilon$$

- Proof:
  Let $w \in \mathscr{R}$, $|w| \leq 1$ and w_k be a finite-precision truncation of w with $O(\log(1/\delta))$ digits.
  Then $|w - w_k| \leq \delta$.

  Hence for a "network" $f(x) = \sigma(wx)$, we can get $f_k(x) = \sigma(w_kx)$ s.t. $\max_{|x|\leq 1} |f(x) - f_k(x)| \leq \delta$

Step 1: Quantizing to Finite Precision

Theorem:
Let $f(x) = \sigma(W_l\sigma(W_{l-1}\ldots\sigma(W_1x)))$ be such that $\|W_i\|_2 \leq 1$. Then, for any $\epsilon > 0$, we can replace the weights of $f(x)$ with a finite precision truncated ones, each represented by $k = O(\log(dl/\epsilon))$ bits such that

$$\max_{\|x\|\leq 1} \|f(x) - f_k(x)\| \leq \epsilon$$

- Proof:
  For a single layer, we obtain $\|\sigma(Wx) - \sigma(W_kx)\| \leq \|Wx - W_k\| \leq d^2\delta$

Step 1: Quantizing to Finite Precision

Theorem:
Let $f(x) = \sigma(W_l\sigma(W_{l-1}\ldots\sigma(W_1x)))$ be such that $\|W_i\|_2 \leq 1$. Then, for any $\epsilon > 0$, we can replace the weights of $f(x)$ with a finite precision truncated ones, each represented by $k = O(\log(dl/\epsilon))$ bits such that

$$\max_{\|x\|\leq 1} \|f(x) - f_k(x)\| \leq \epsilon$$

- Proof:
  For a single layer, we obtain $\|\sigma(Wx) - \sigma(W_kx)\| \leq \|Wx - W_k\| \leq d^2\delta$

  For two layers we have

$$\begin{aligned}
\|W_2\sigma(W_1x) - W_{2,k}\sigma(W_{2,k}x)\| &\leq \|W_2y - W_{2,k}(y + \delta r)\| \\
&\leq \|W_2y - W_{2,k}y\| + \|\delta W_{2,k}r\| \\
&\leq \|W_2 - W_{2,k}\|\,\|y\| + \delta \\
&\leq 2\delta
\end{aligned}$$

Step 1: Quantizing to Finite Precision

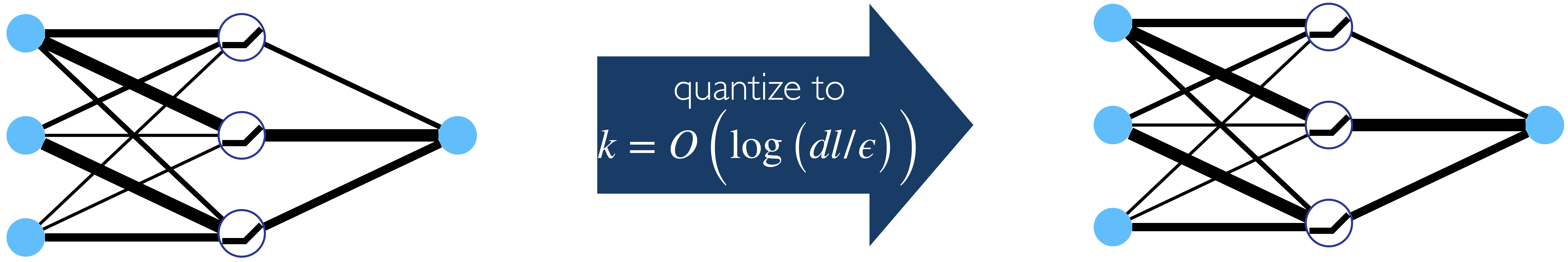
Theorem:
Let $f(x) = \sigma(W_l\sigma(W_{l-1}\ldots\sigma(W_1x)))$ be such that $\|W_i\|_2 \leq 1$. Then, for any $\epsilon > 0$, we can replace the weights of $f(x)$ with a finite precision truncated ones, each represented by $k = O(\log(dl/\epsilon))$ bits such that

$$\max_{\|x\|\leq 1} \|f(x) - f_k(x)\| \leq \epsilon$$

- Proof:
  For l layers we have $\max_{\|x\|\leq 1} \|f(x) - f_k(x)\| \leq d^2 \cdot l \cdot \epsilon$

  Setting $\delta = \dfrac{\epsilon}{d^2 l}$ completes the proof

Step 1: Quantizing to Finite Precision



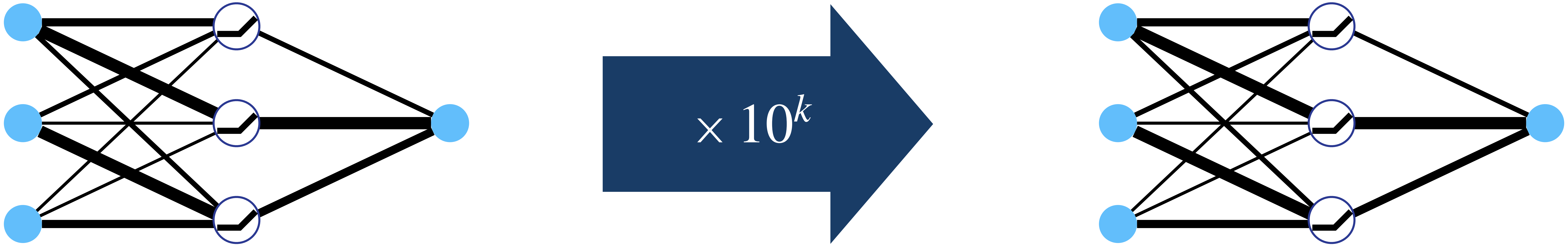
$w = 0.257081639...1... \in [-1,1]$

$w = 0.\underbrace{257081639...1}_{k\ \textbf{digits}} \in \{-1 : 1/\epsilon : 1\}$

Theorem:
Let $f(x) = \sigma(W_l\sigma(W_{l-1}\ldots\sigma(W_1x)))$ be such that $\|W_i\|_2 \leq 1$. Then, for any $\epsilon > 0$, we can replace the weights of $f(x)$ with a finite precision truncated ones, each represented by $k = O(\log(dl/\epsilon))$ bits such that

$$\max_{\|x\|\leq 1} \|f(x) - f_k(x)\| \leq \epsilon$$

Step 2: Mapping to Integer Network



$w = 0.257081639...1 \in \{-1 : 1/\epsilon : 1\}$ $\quad$ $w = 257081639...1 \in \{-1/\epsilon : 1 : 1/\epsilon\}$

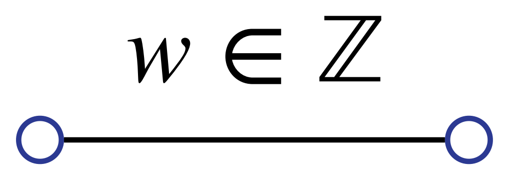
ReLus are positive homogeneous, hence for positive a

$$\sigma(a \cdot x)) = a \cdot \sigma(x)$$

Finite precision network is equivalent to integer network

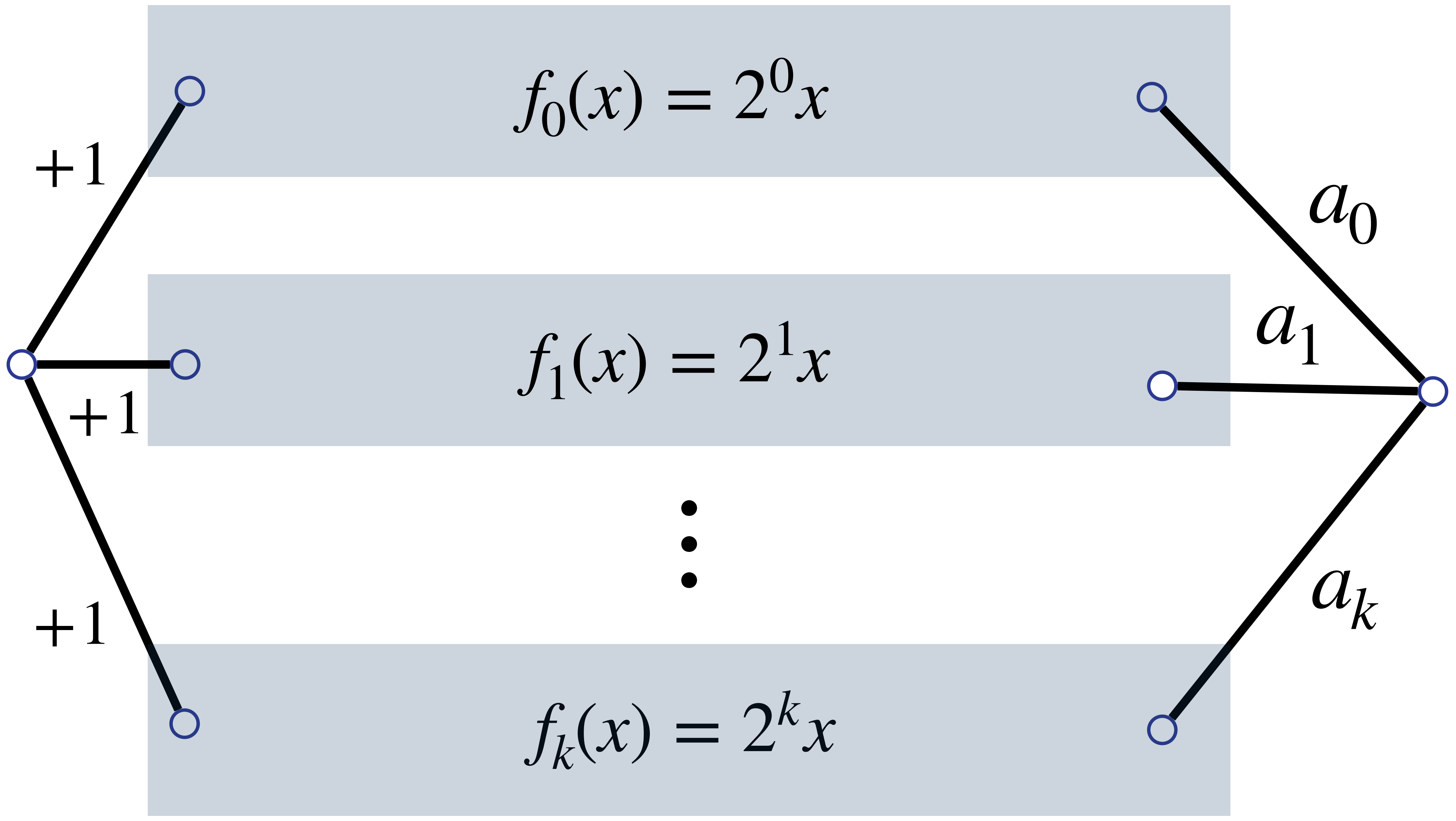
Step 3: From Integers Weights to Binary

target integer weight



$$w = \sum_{i=0}^{\lfloor \log w \rfloor = k} a_i \cdot 2^i$$

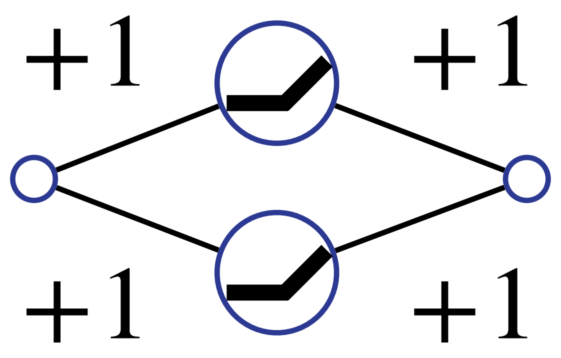
$a_i \in \{-1, 0, 1\}$



Q: How can we build f_i using binary ReLU network?

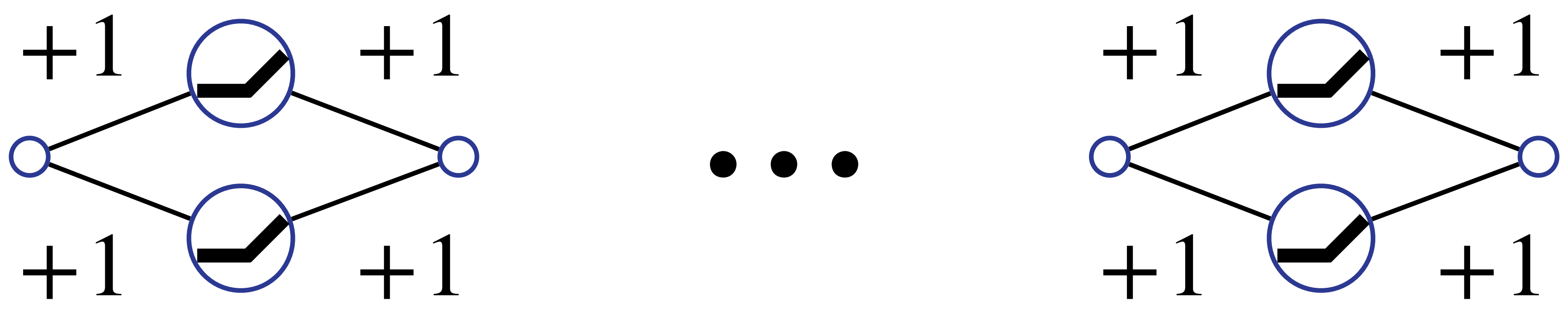
Binary Gadgets

Basic Unit



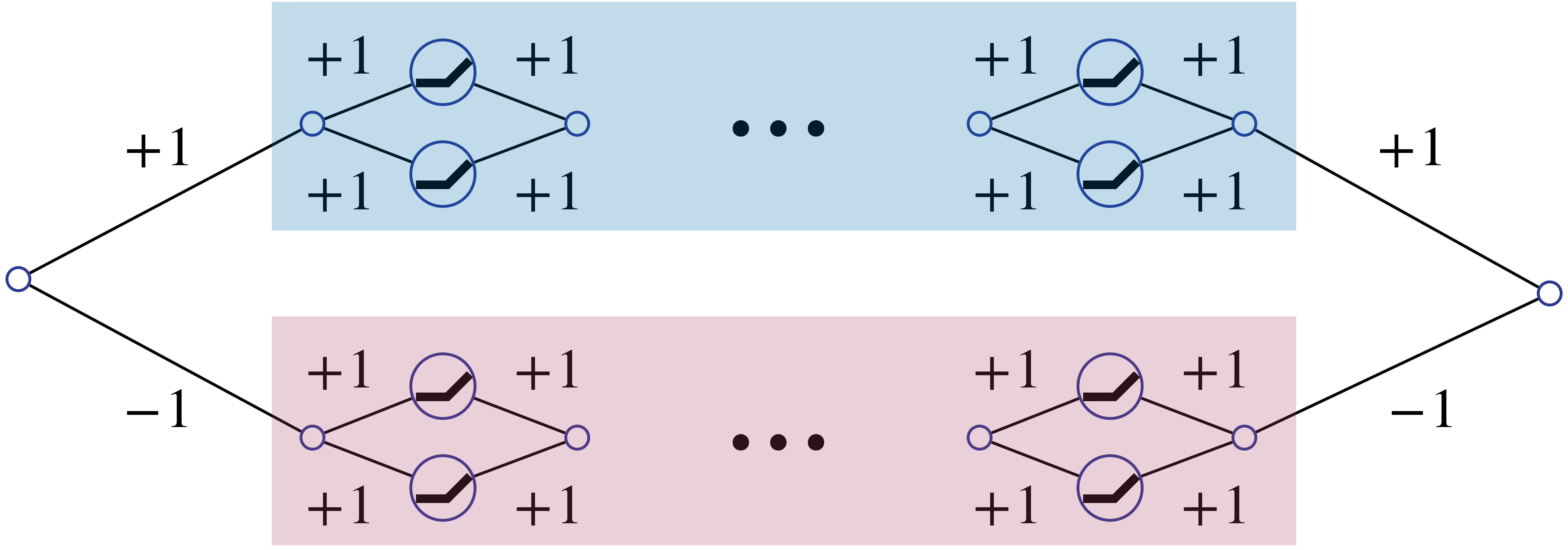
$$g_1(x) = 2 \cdot \max\{x, 0\}$$

Replicate in Serial



$$g_i(x) = 2^i \max\{x, 0\}$$

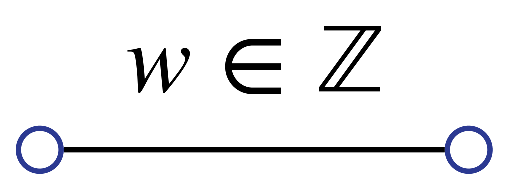
Add/Subtract



$$f_i(x) = g_i(x) - g_i(-x)$$
$$= 2^i \cdot x$$

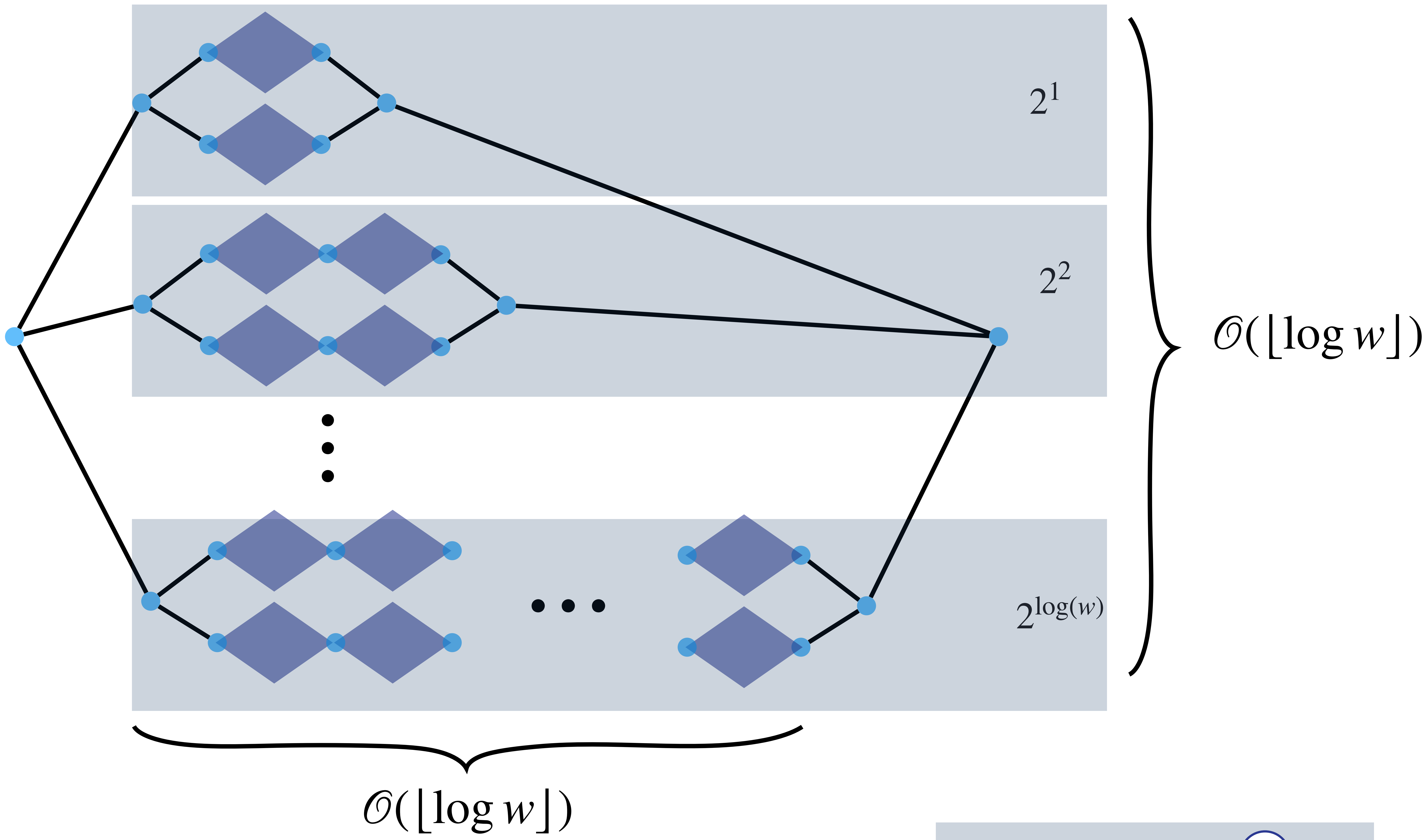
Binary Gadgets

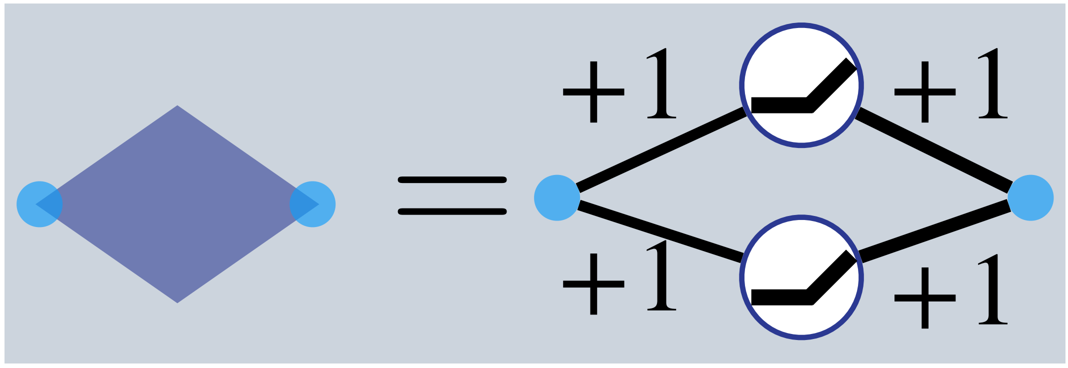
target integer weight



$$w = \sum_{i=0}^{\lfloor \log w \rfloor} a_i \cdot 2^i$$

$$a_i \in \{-1,0,1\}$$





Recap of proof steps

Approximate real net with finite prec

Replace finite prec with integer

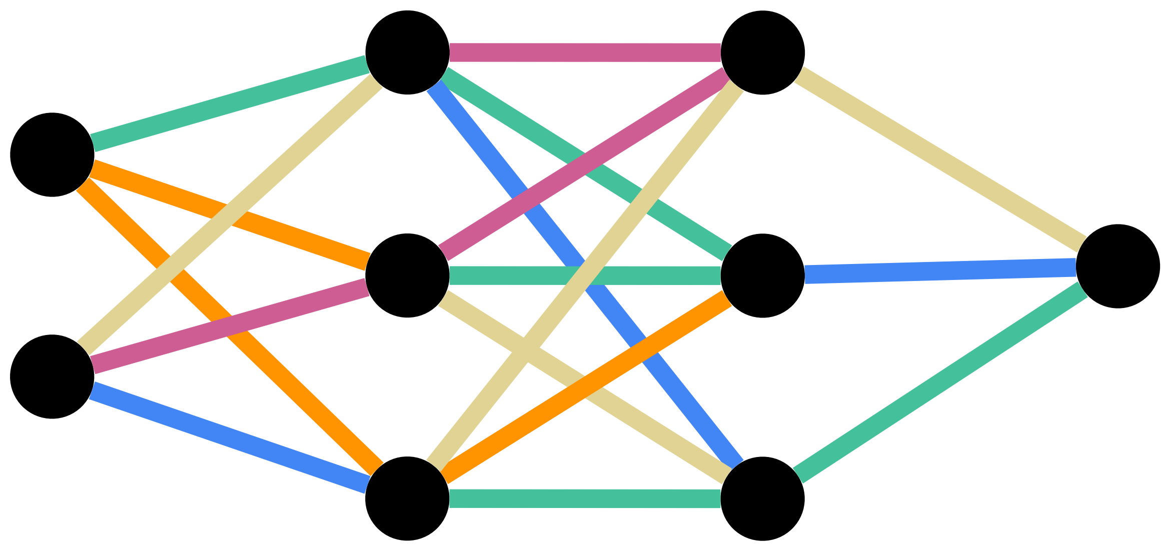
Replace each integer edge with log-bigger FC Relu

Apply on all weight and layers

reminder: this is an existence proof, not an algorithm!

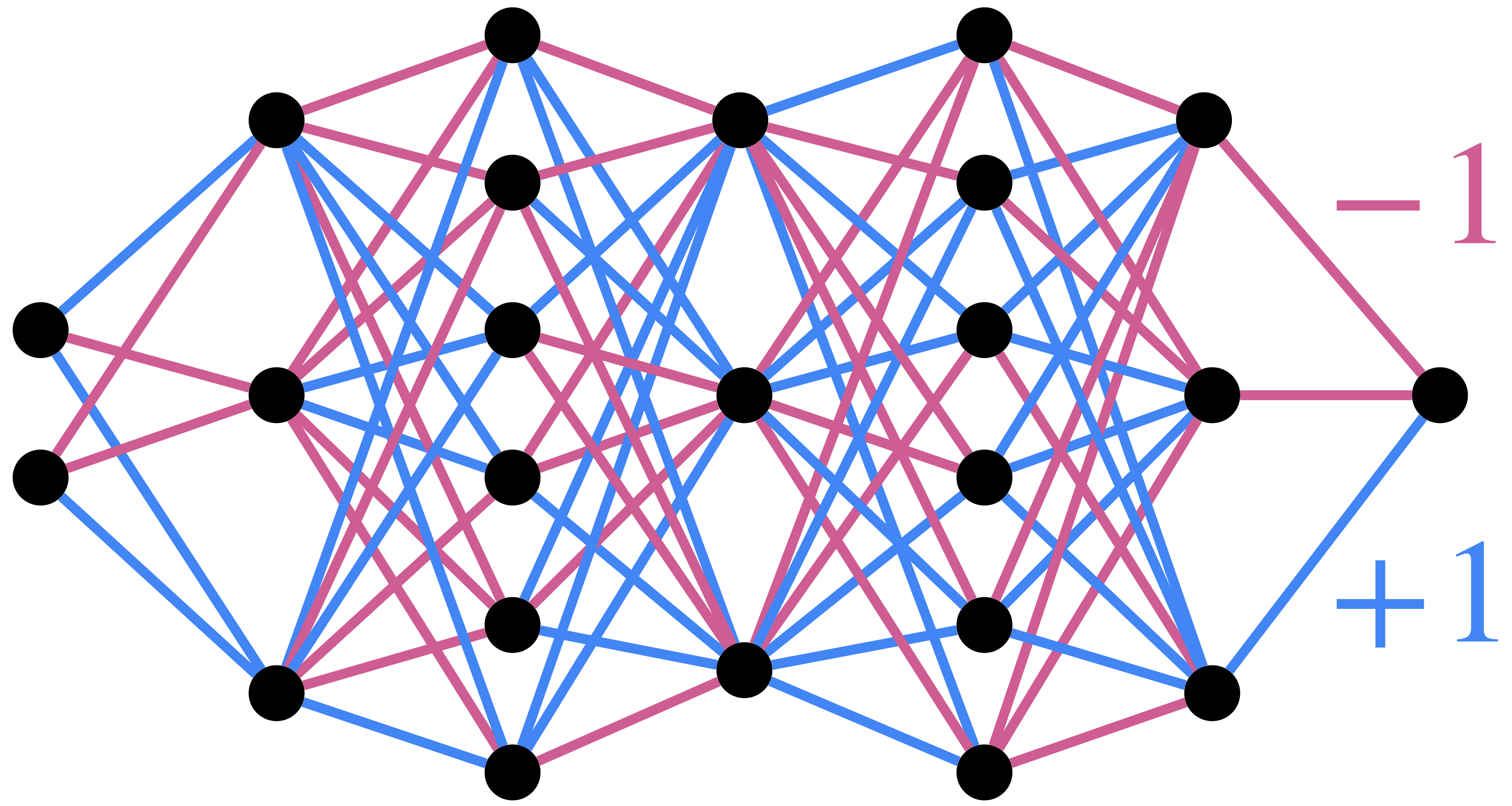
Binary ReLU Nets can be Very Expressive

Any network can be approximated
by $O(\log(dl/\epsilon))$-deeper and $O(\log^2(dl/\epsilon))$-wider binary network



a neural network
with high accuracy

$\approx$



a larger, binary network can approximate it

Conclusion

- Binary networks can be accurate and efficient
- Training algorithms based on simple variants of backprop

<u>Open Questions</u>

- Theoretical analysis on algorithms for training BNNs
- Network architectures amenable to binarization
- Theory for threshold+binary weights?

Reading List

Courbariaux, M., Bengio, Y. and David, J.P., 2015. Binaryconnect: Training deep neural networks with binary weights during propagations. Advances in neural information processing systems, 28.

Rastegari, M., Ordonez, V., Redmon, J. and Farhadi, A., 2016, October. Xnor-net: Imagenet classification using binary convolutional neural networks. In European conference on computer vision (pp. 525-542). Springer, Cham.

Qin, H., Gong, R., Liu, X., Bai, X., Song, J. and Sebe, N., 2020. Binary neural networks: A survey. Pattern Recognition, 105, p.107281.

Alizadeh, M., Fernández-Marqués, J., Lane, N.D. and Gal, Y., 2018, September. An empirical study of binary neural networks' optimisation. In International conference on learning representations.

Gholami, A., Kim, S., Dong, Z., Yao, Z., Mahoney, M.W. and Keutzer, K., 2021. A survey of quantization methods for efficient neural network inference. arXiv preprint arXiv:2103.13630.

Sreenivasan, K., Rajput, S., Sohn, J.Y. and Papailiopoulos, D., 2021. Finding Everything within Random Binary Networks. AISTATS 2022