



Exploring Complexity Reduction in Deep Learning

Sourya Dey

PhD Candidate, University of Southern California

Advisors: Peter A. Beerel and Keith M. Chugg

B. Tech, Instrumentation Engineering, IIT KGP, 2014

January 3, 2020

USC
Viterbi

School of Engineering
*Ming Hsieh Department
of Electrical and
Computer Engineering*

Outline

Pre-Defined Sparsity

Reduce complexity of neural networks with minimal performance degradation

Analysis and Applications

Deep dive into pre-defined sparsity for MLPs, and a corresponding application

Model Search

Automate the design of CNNs with good performance and low complexity

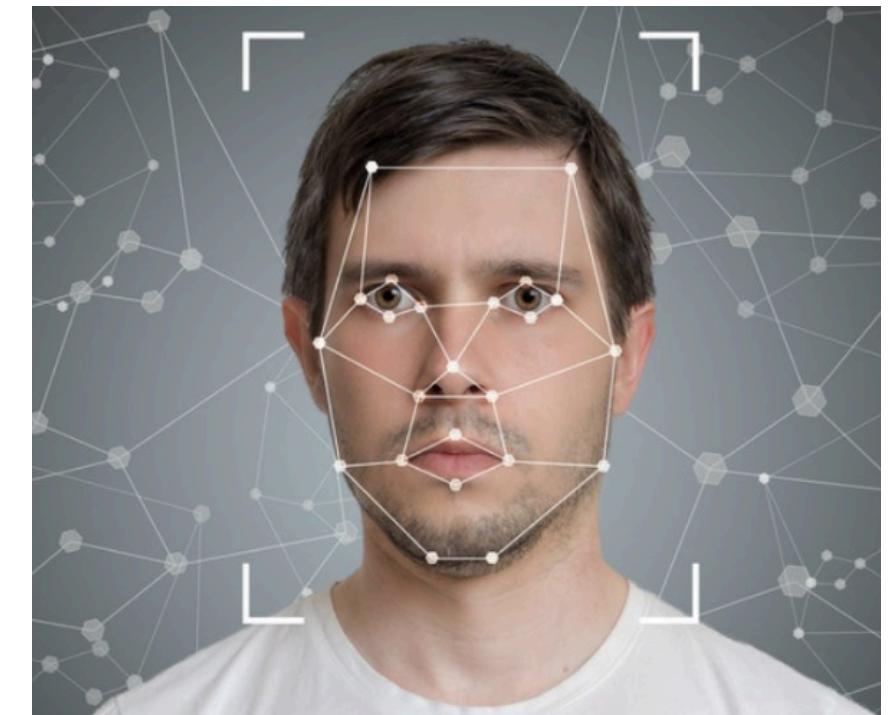
Pre-Defined Sparsity

Reduce complexity of neural networks with minimal performance degradation

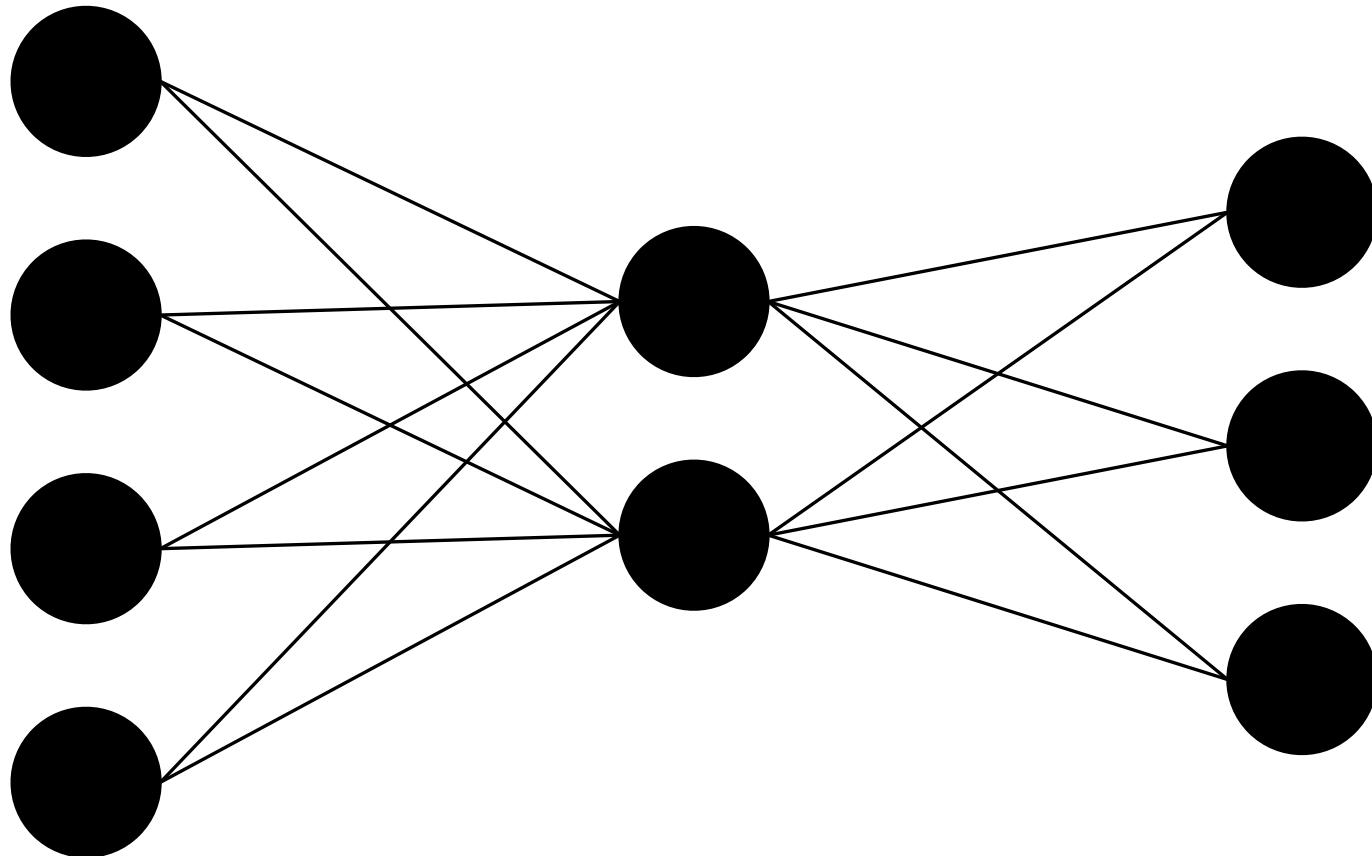
Overview

Neural networks (NNs) are key machine learning technologies

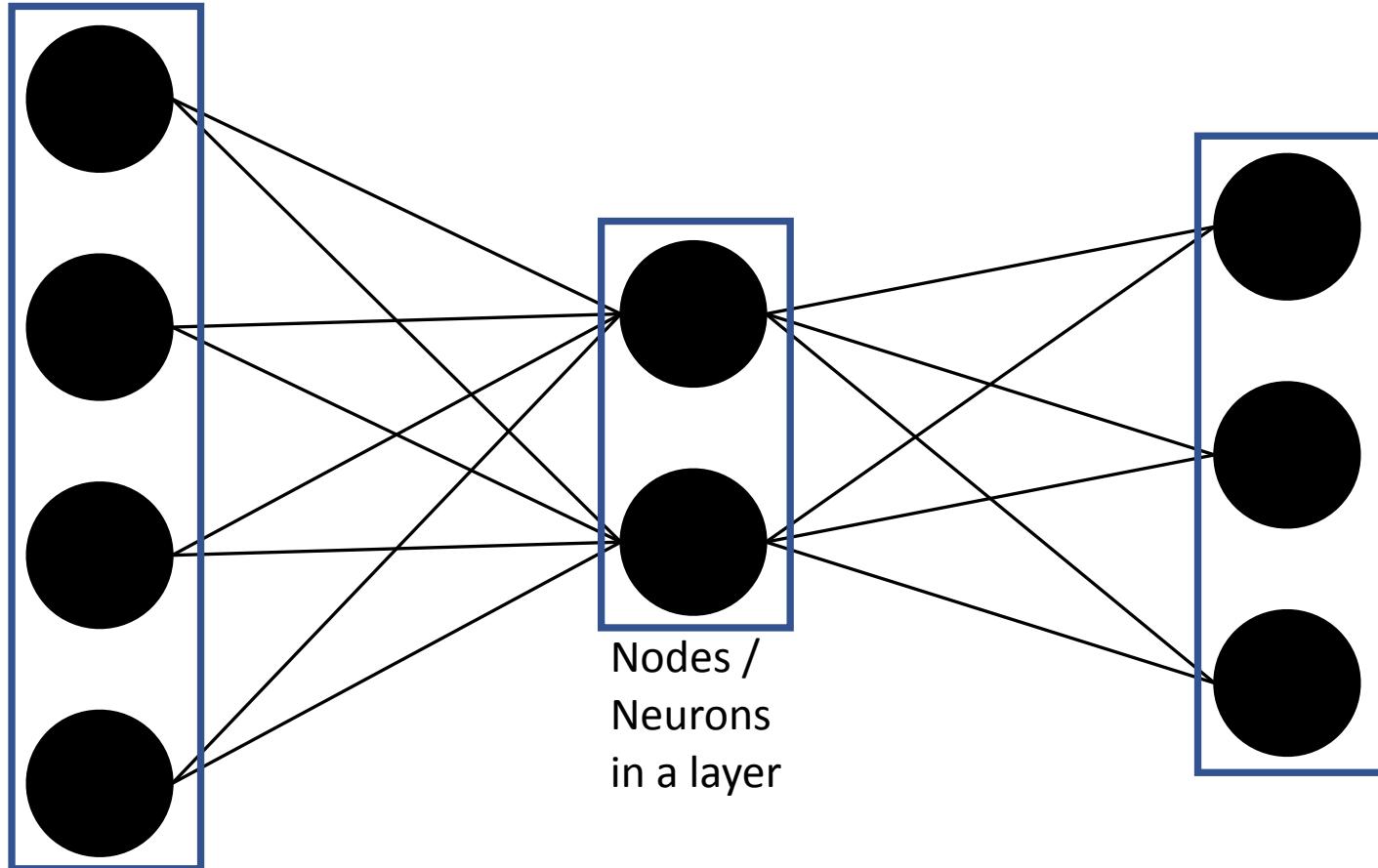
- Artificial intelligence
- Self-driving cars
- Speech recognition
- Face ID
- and more smart stuff ...



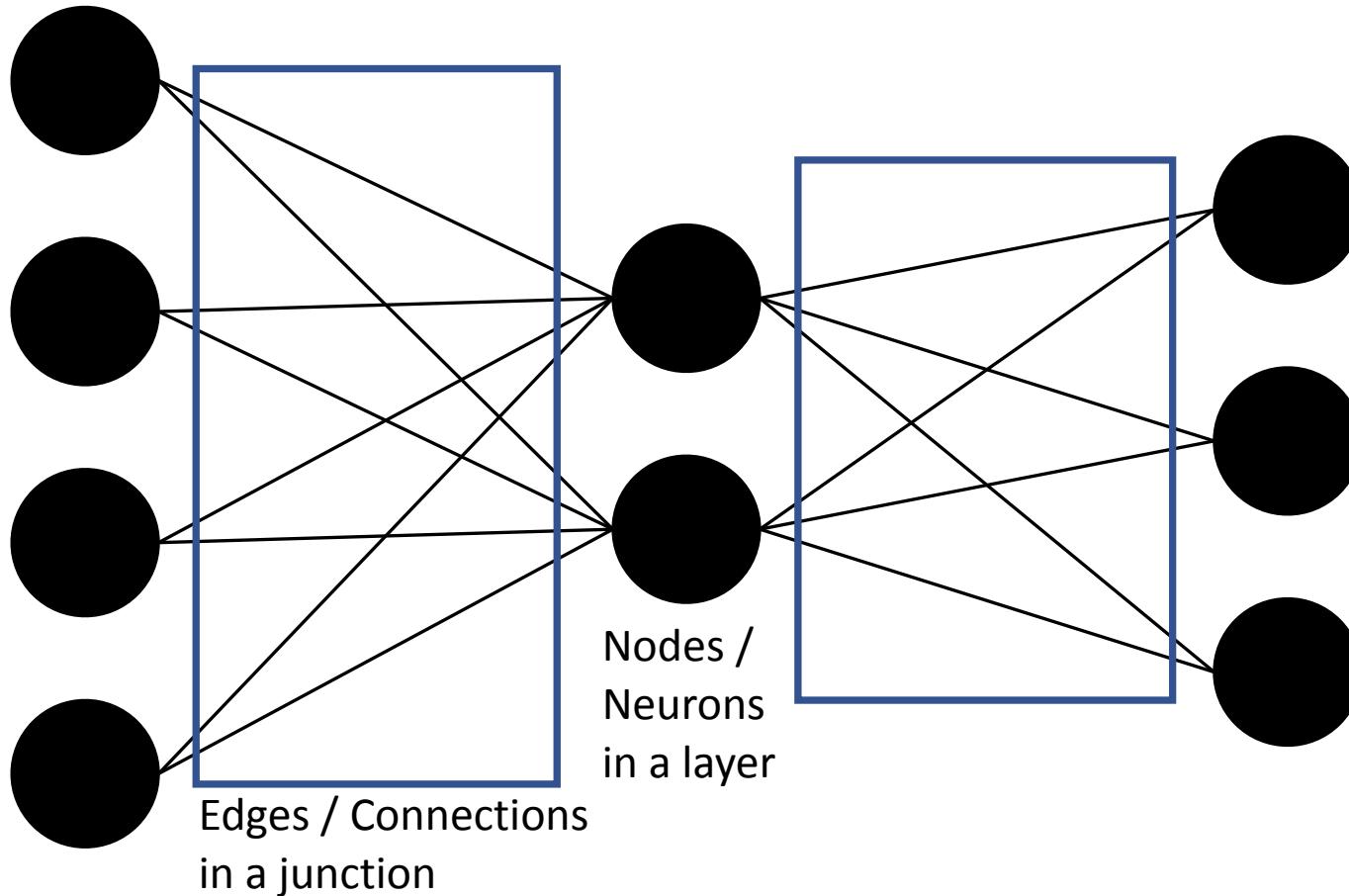
Basic working of an artificial neural network



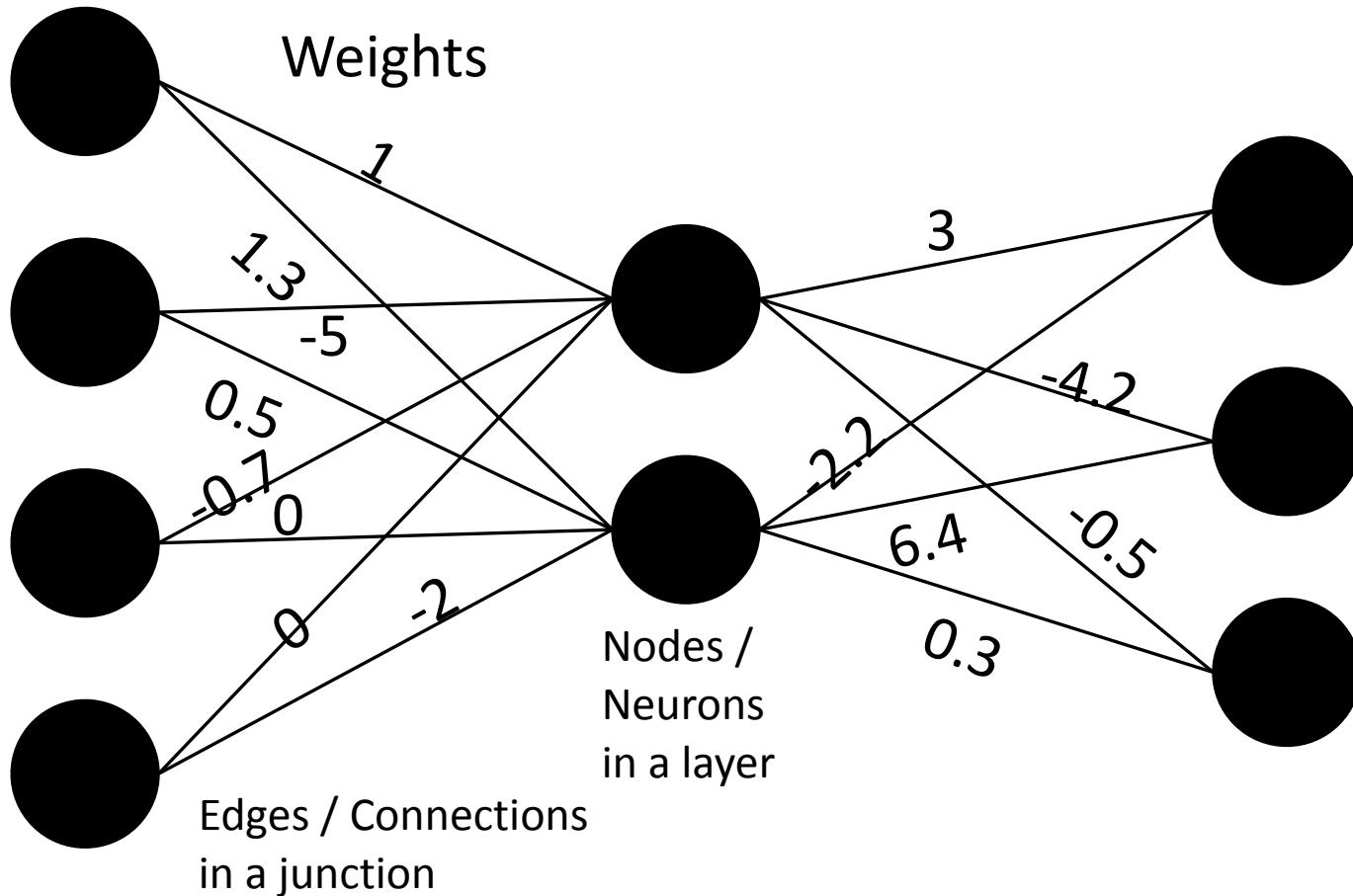
Basic working of an artificial neural network



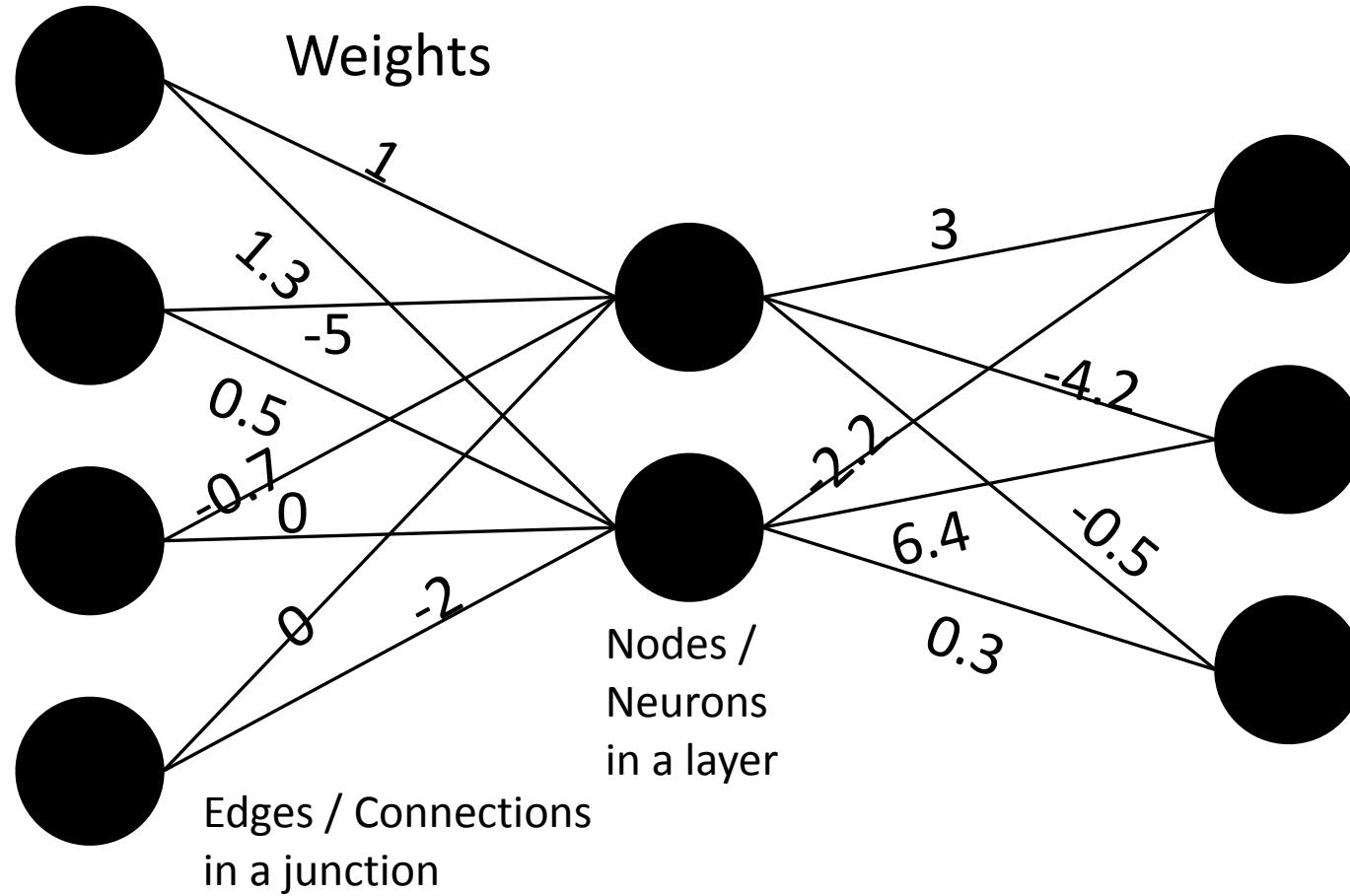
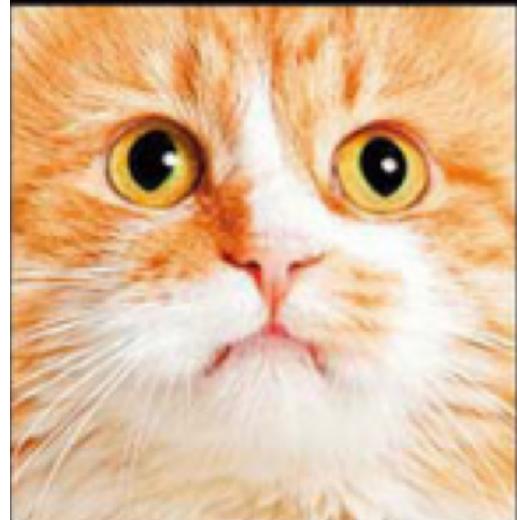
Basic working of an artificial neural network



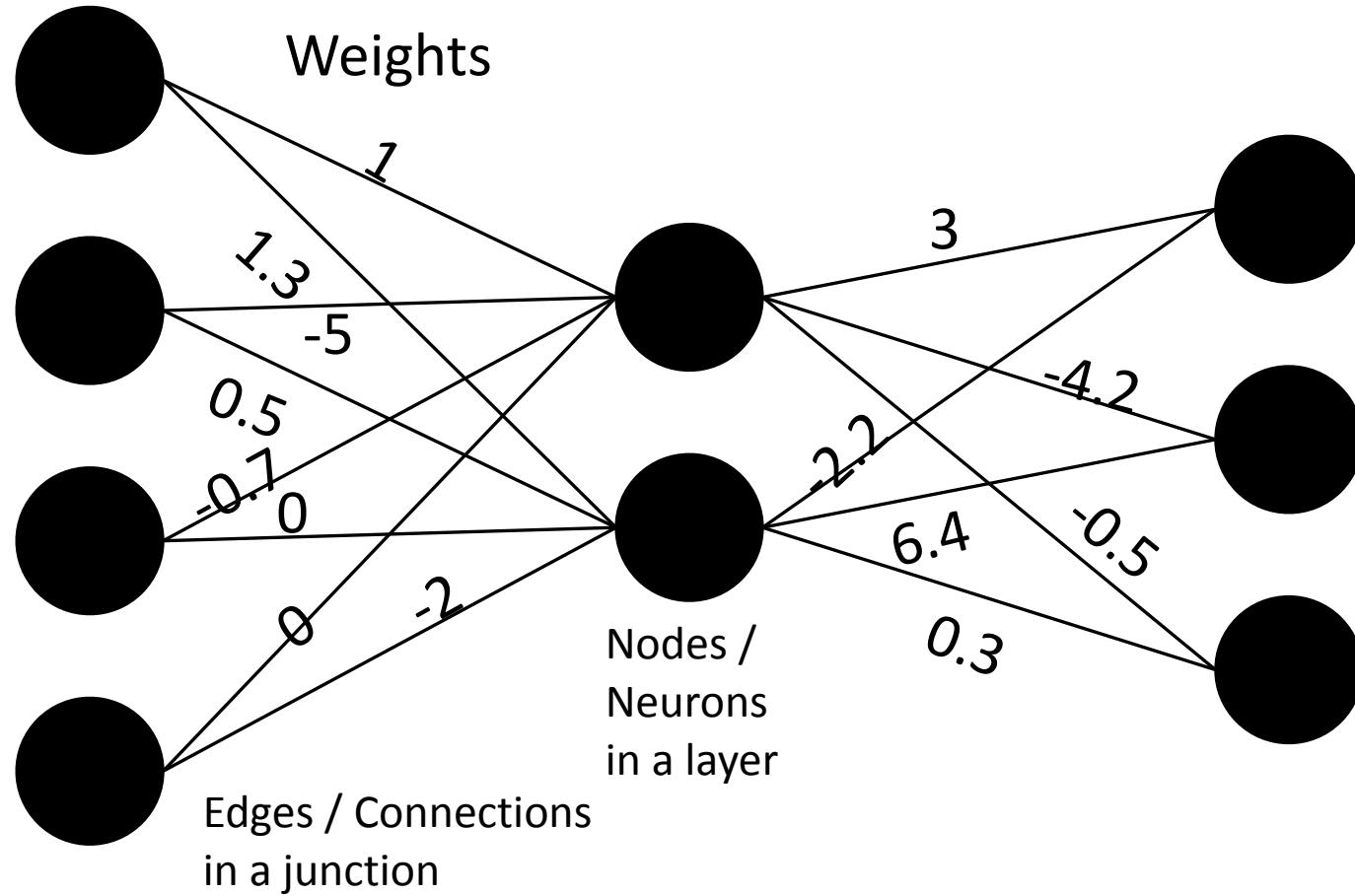
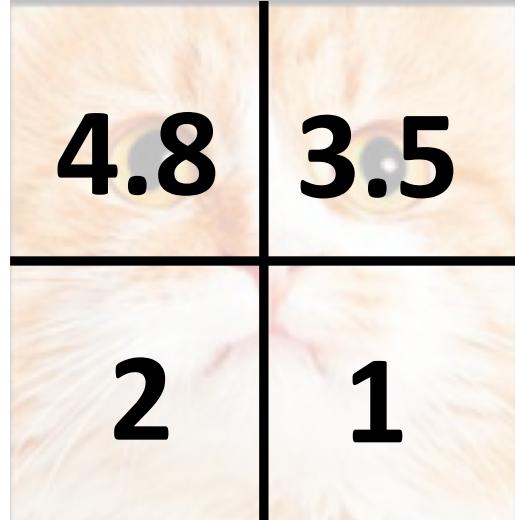
Basic working of an artificial neural network



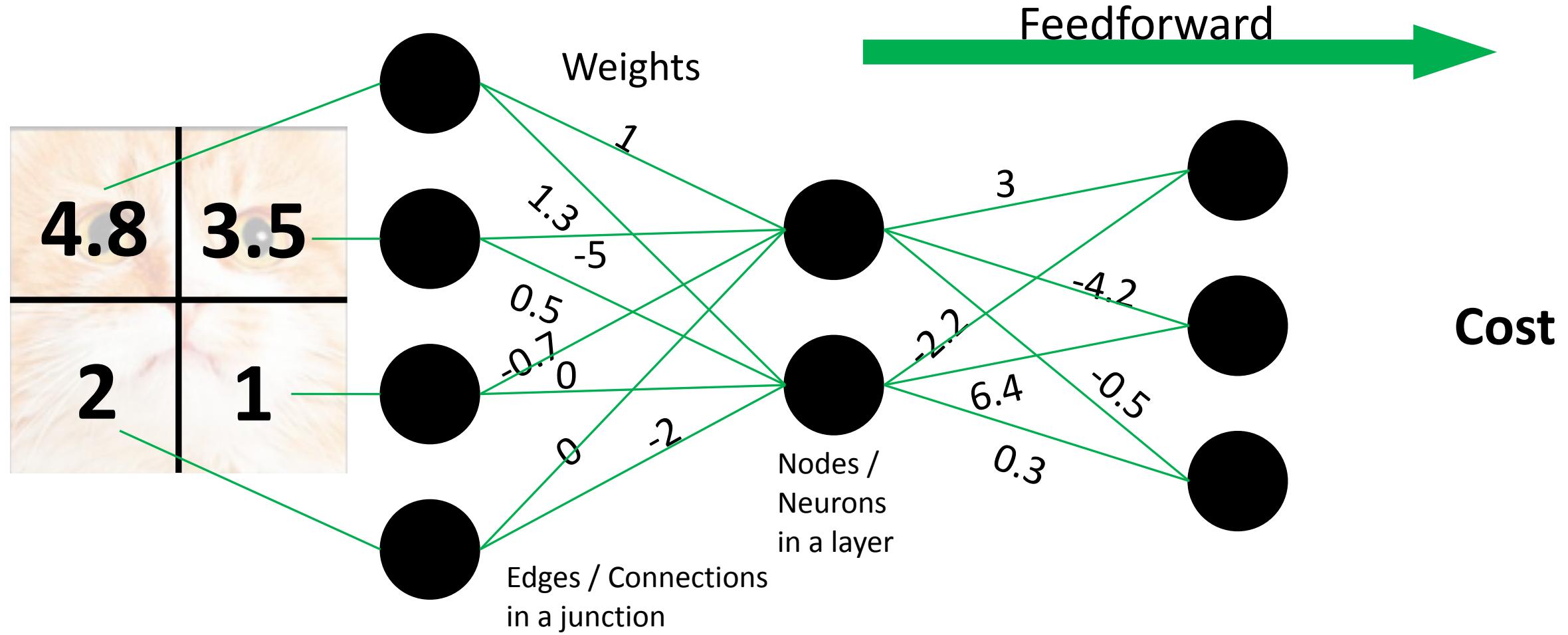
Basic working of an artificial neural network



Basic working of an artificial neural network



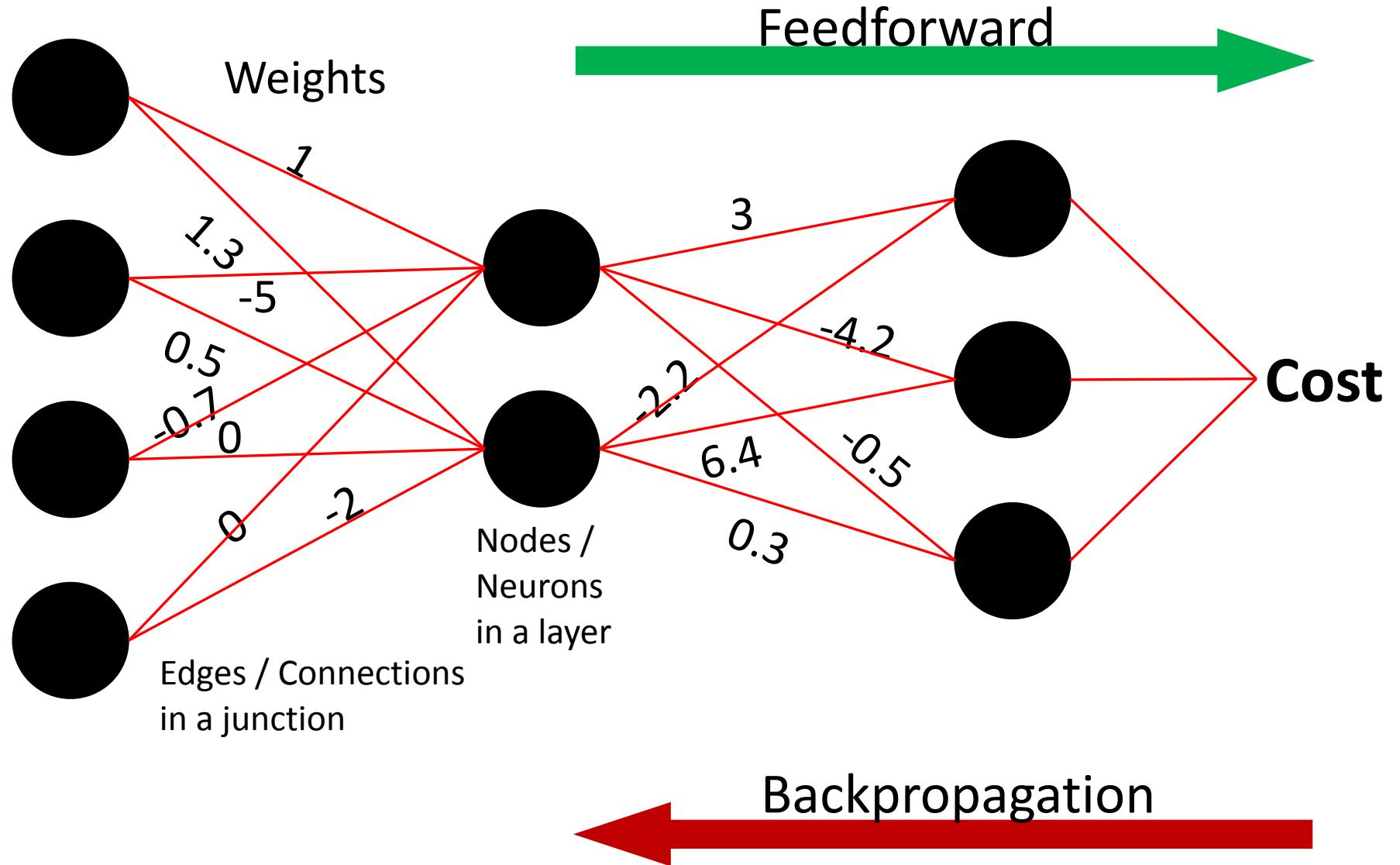
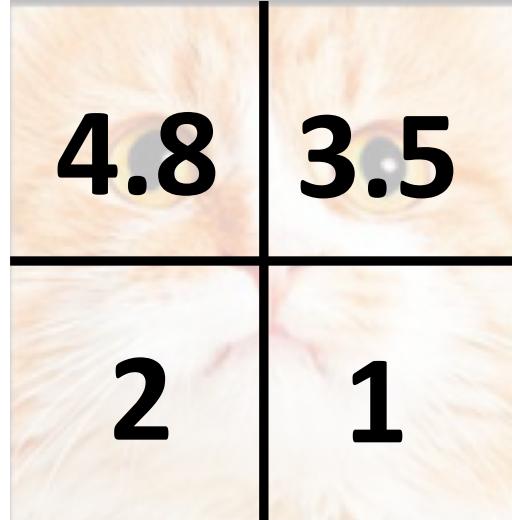
Basic working of an artificial neural network



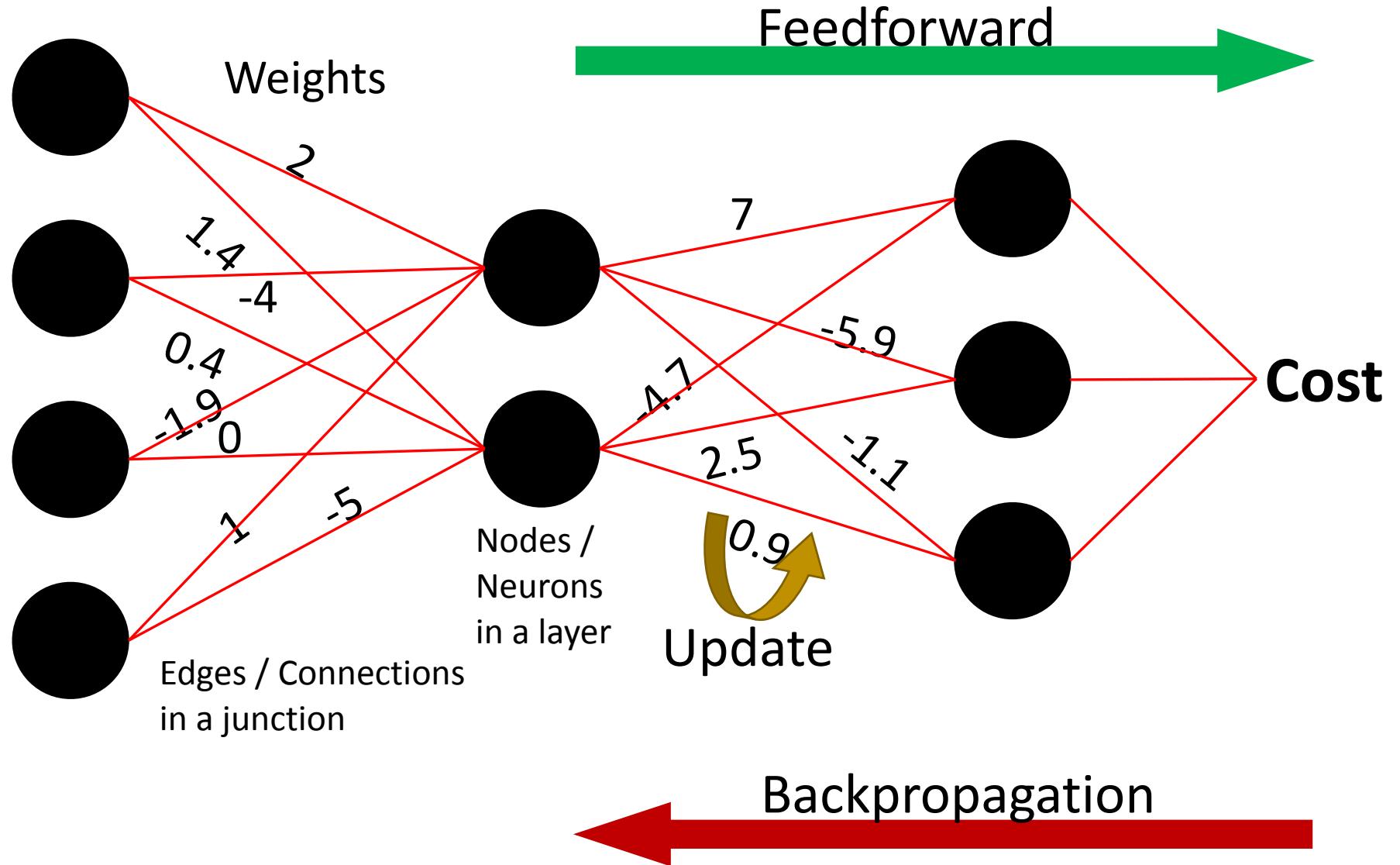
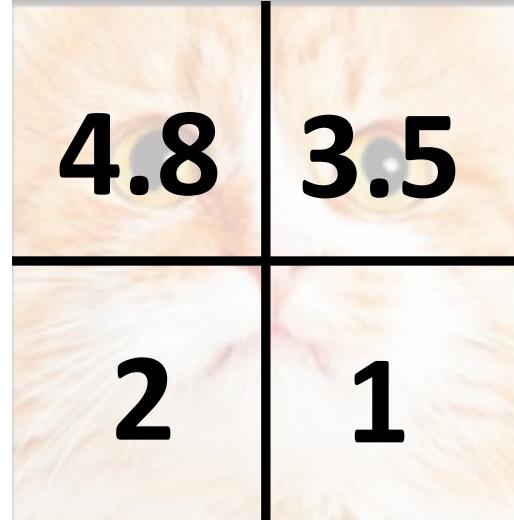
Training

Inference

Basic working of an artificial neural network

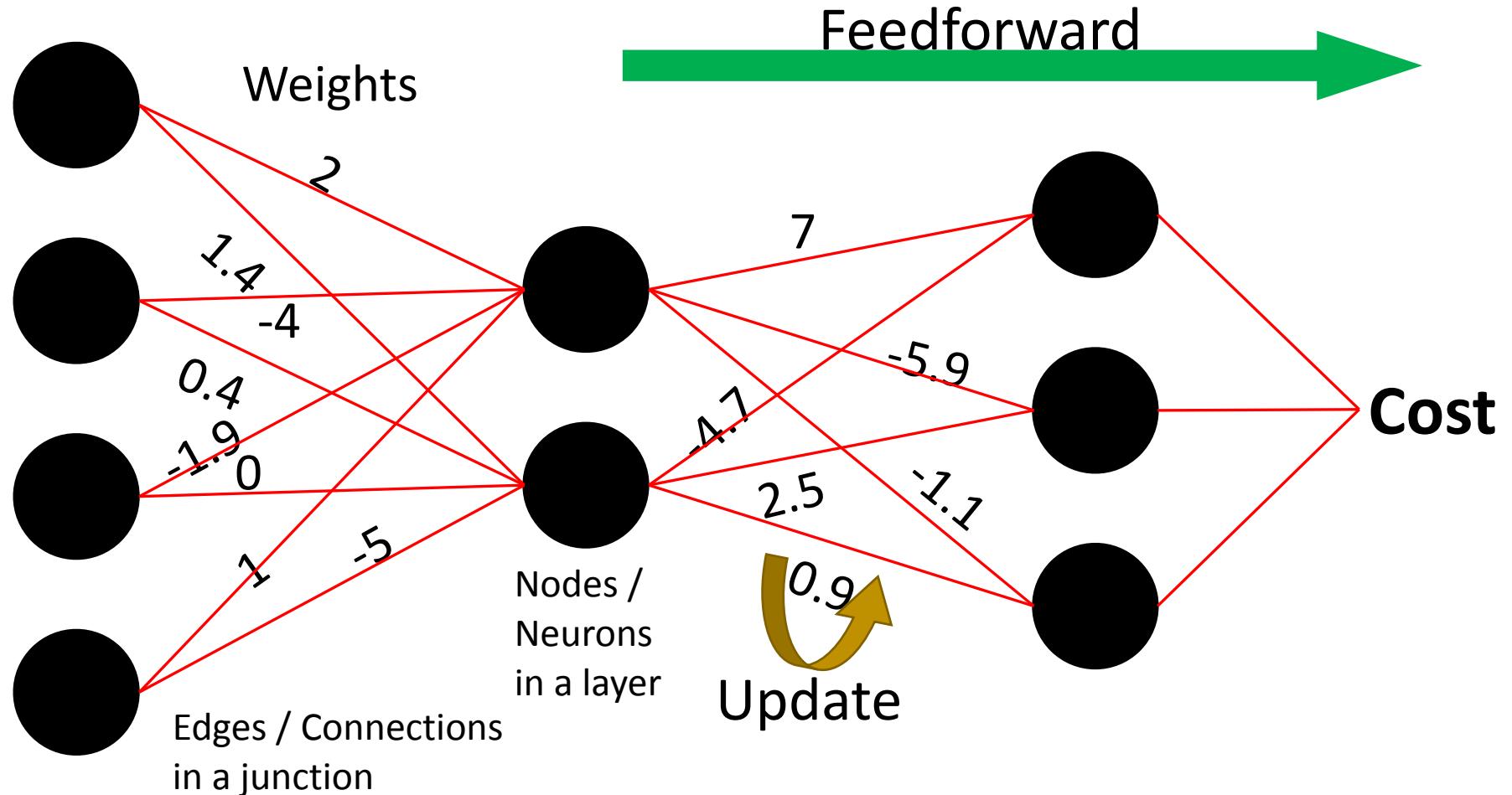
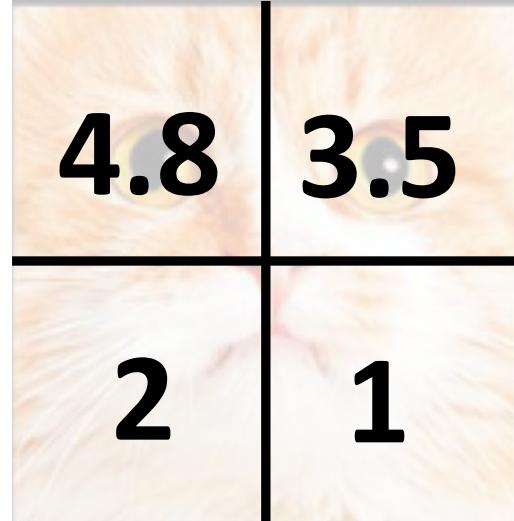


Basic working of an artificial neural network



Training

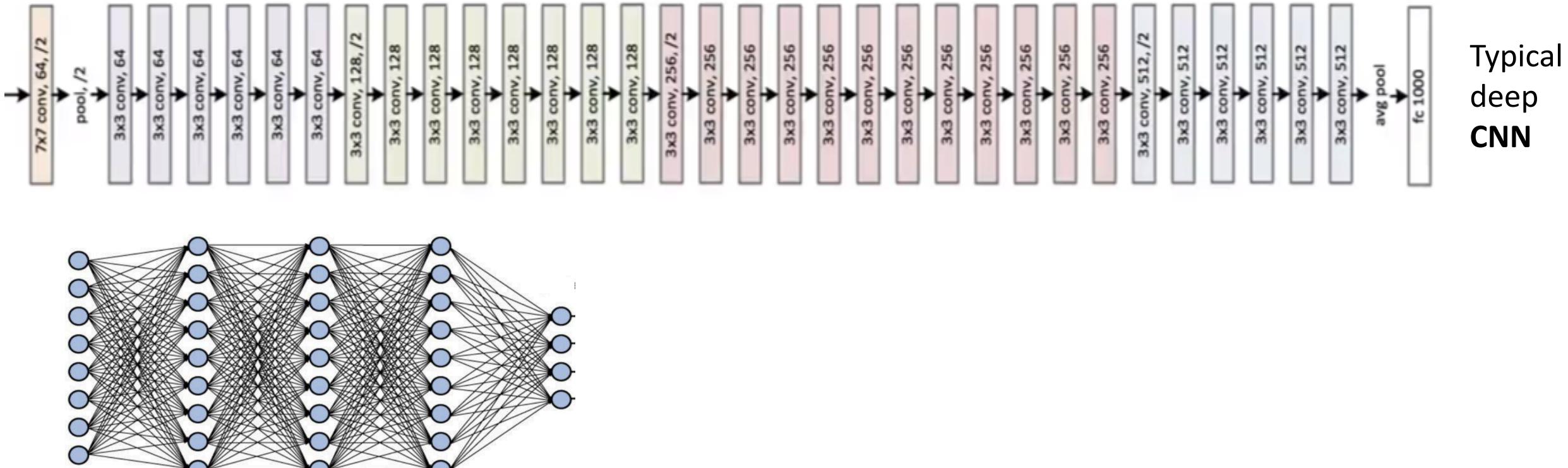
Basic working of an artificial neural network



*Weights dominate complexity –
they are all used in all 3 operations*

Motivation behind our work

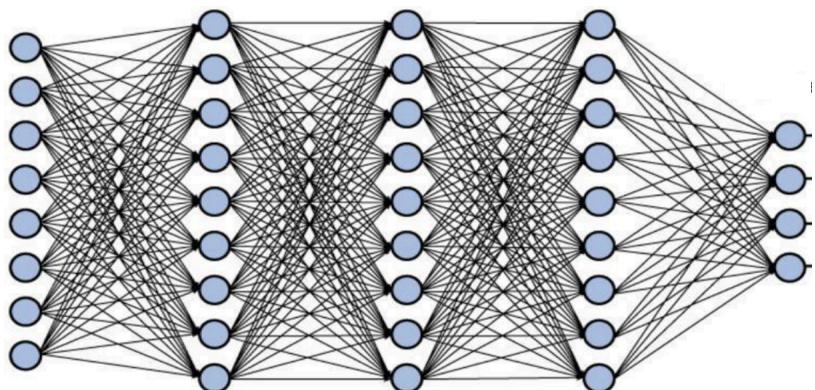
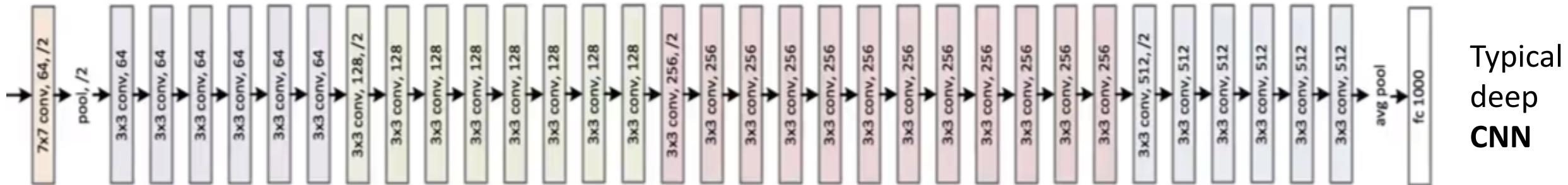
Modern neural networks suffer from parameter explosion



Fully connected (**FC**) Multilayer Perceptron (**MLP**)

Motivation behind our work

Modern neural networks suffer from parameter explosion



Fully connected (**FC**) Multilayer Perceptron (**MLP**)

Training can take weeks on CPU
Cloud GPU resources are expensive



Our Work: Pre-defined Sparsity

Pre-define a sparse connection
pattern **prior to training**

Use this sparse network for both
training and inference

Our Work: Pre-defined Sparsity

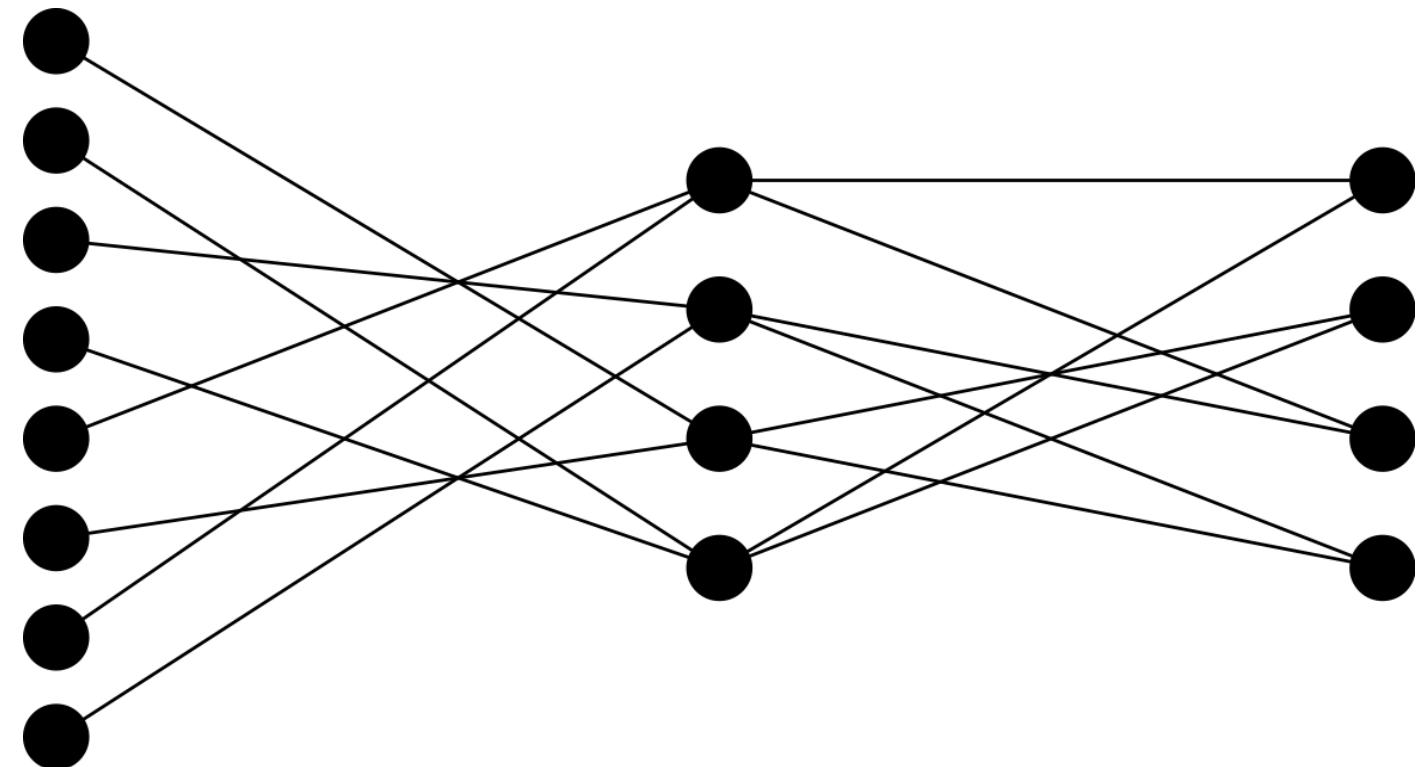
Pre-define a sparse connection pattern **prior to training**

Use this sparse network for both training and inference

$$\mathbf{N}_{\text{net}} = (8, 4, 4)$$

$$\mathbf{d}_{\text{net}}^{\text{out}} = (1, 2)$$

$$\mathbf{d}_{\text{net}}^{\text{in}} = (2, 2)$$



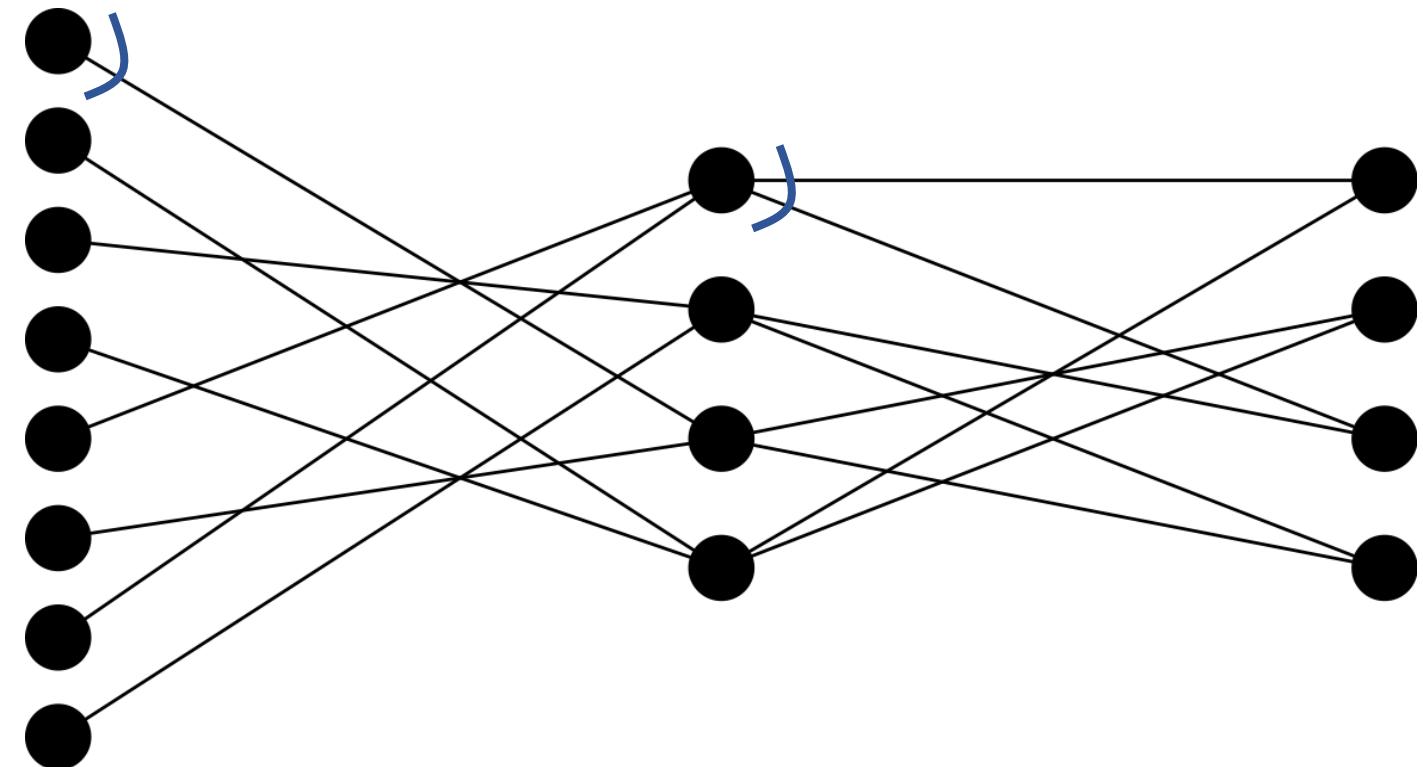
Our Work: Pre-defined Sparsity

Pre-define a sparse connection pattern **prior to training**
Use this sparse network for both training and inference

$$N_{\text{net}} = (8, 4, 4)$$

$$d_{\text{net}}^{\text{out}} = (1, 2)$$

$$d_{\text{net}}^{\text{in}} = (2, 2)$$



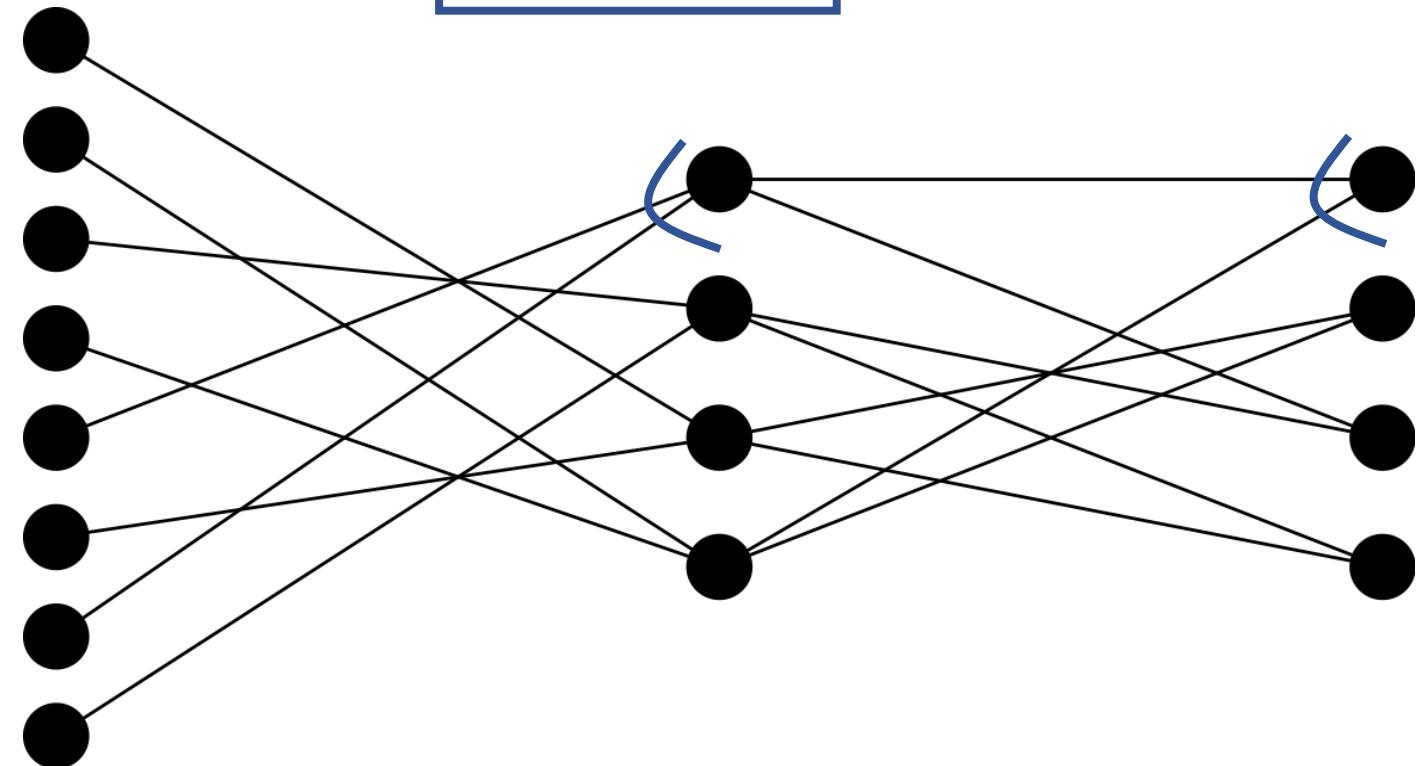
Our Work: Pre-defined Sparsity

Pre-define a sparse connection pattern **prior to training**
Use this sparse network for both training and inference

$$\mathbf{N}_{\text{net}} = (8, 4, 4)$$

$$\mathbf{d}_{\text{net}}^{\text{out}} = (1, 2)$$

$$\mathbf{d}_{\text{net}}^{\text{in}} = (2, 2)$$



Our Work: Pre-defined Sparsity

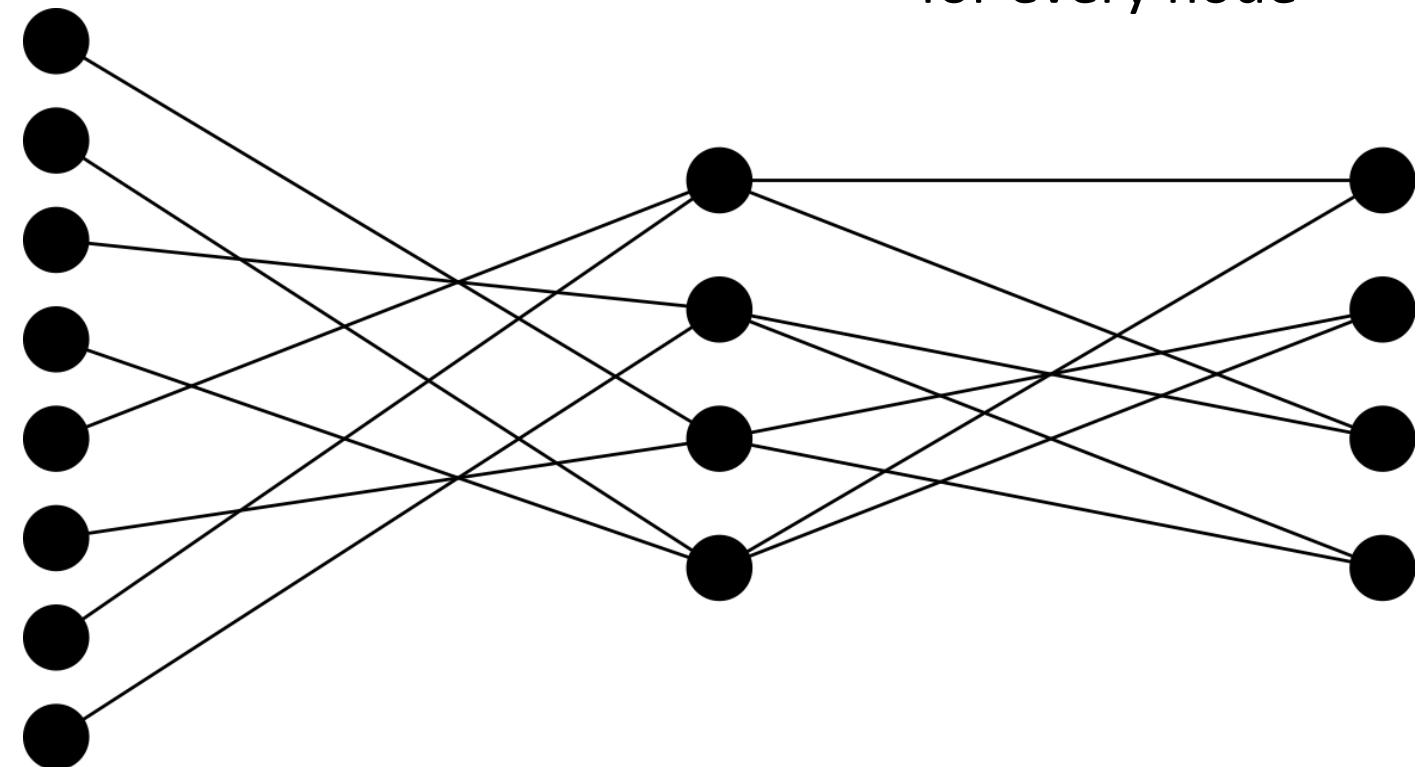
Pre-define a sparse connection pattern **prior to training**

Use this sparse network for both training and inference

$$\mathbf{N}_{\text{net}} = (8, 4, 4)$$

$$\begin{aligned} \mathbf{d}_{\text{net}}^{\text{out}} &= (1, 2) \\ \mathbf{d}_{\text{net}}^{\text{in}} &= (2, 2) \end{aligned}$$

Structured Constraints:
Fixed in-, out-degrees
for every node



Our Work: Pre-defined Sparsity

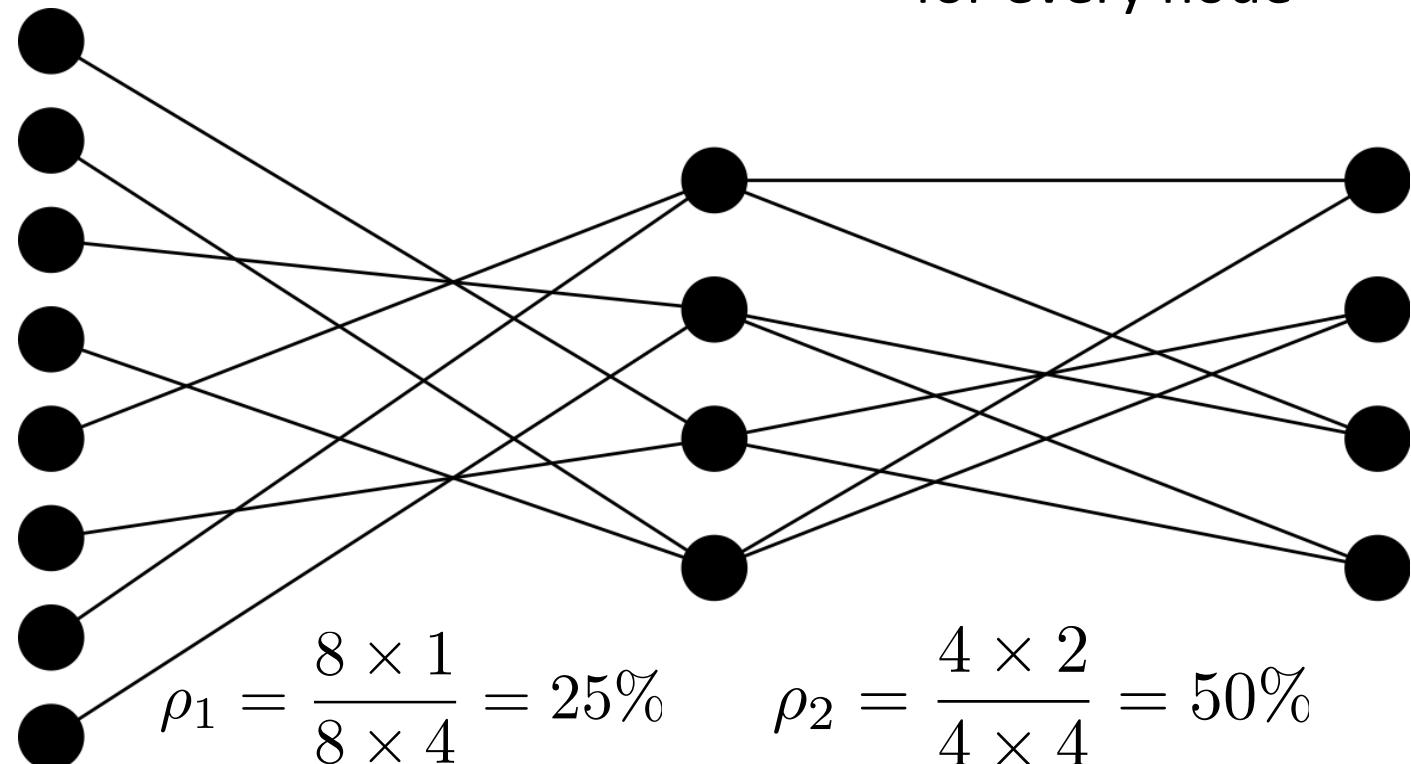
Pre-define a sparse connection pattern **prior to training**

Use this sparse network for both training and inference

$$N_{\text{net}} = (8, 4, 4)$$

$$\begin{aligned} d_{\text{net}}^{\text{out}} &= (1, 2) \\ d_{\text{net}}^{\text{in}} &= (2, 2) \end{aligned}$$

Structured Constraints:
Fixed in-, out-degrees
for every node



$$\rho_{\text{net}} = \frac{8 + 8}{32 + 16} = 33\%$$

Overall Density
compared to FC

Our Work: Pre-defined Sparsity

Pre-define a sparse connection pattern **prior to training**

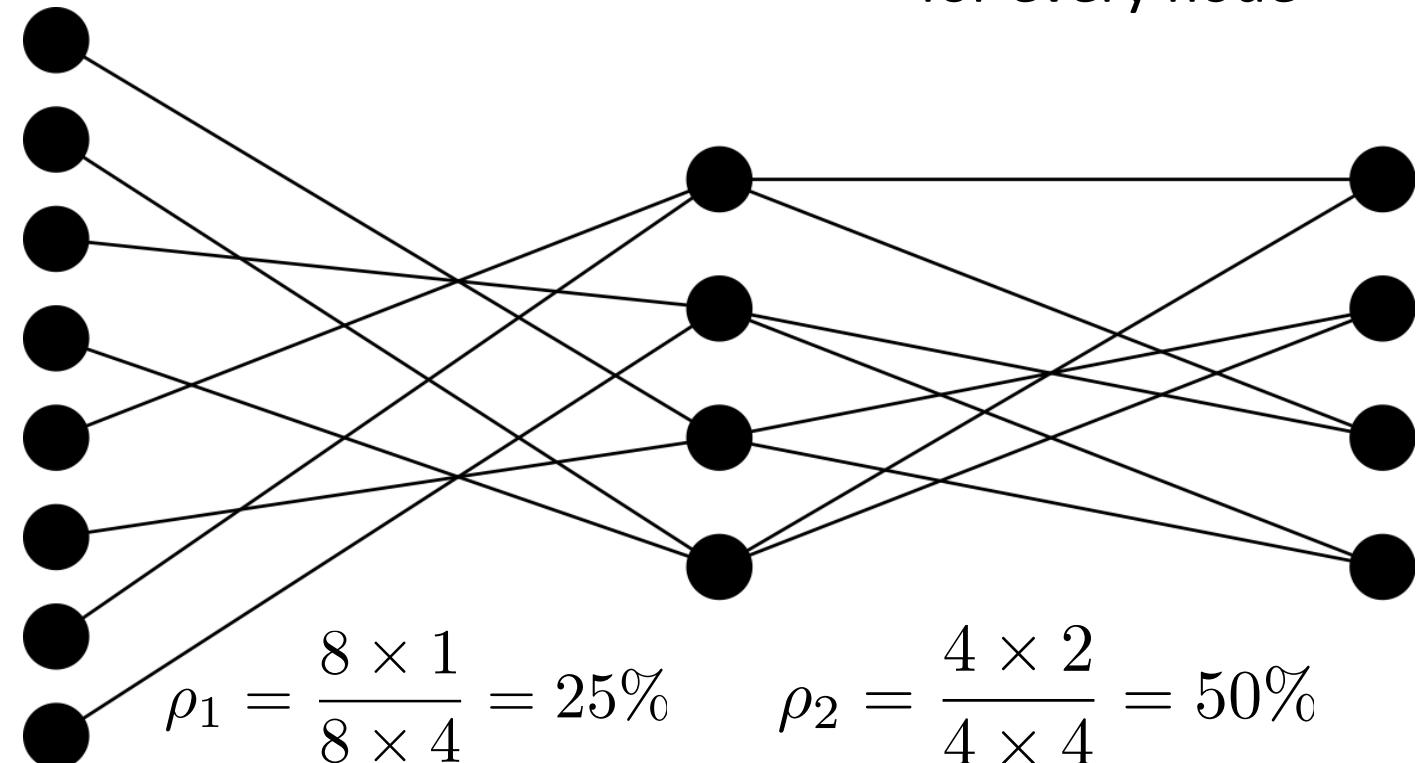
Use this sparse network for both training and inference

Reduced training
and inference
complexity

$$N_{\text{net}} = (8, 4, 4)$$

$$\begin{aligned} d_{\text{net}}^{\text{out}} &= (1, 2) \\ d_{\text{net}}^{\text{in}} &= (2, 2) \end{aligned}$$

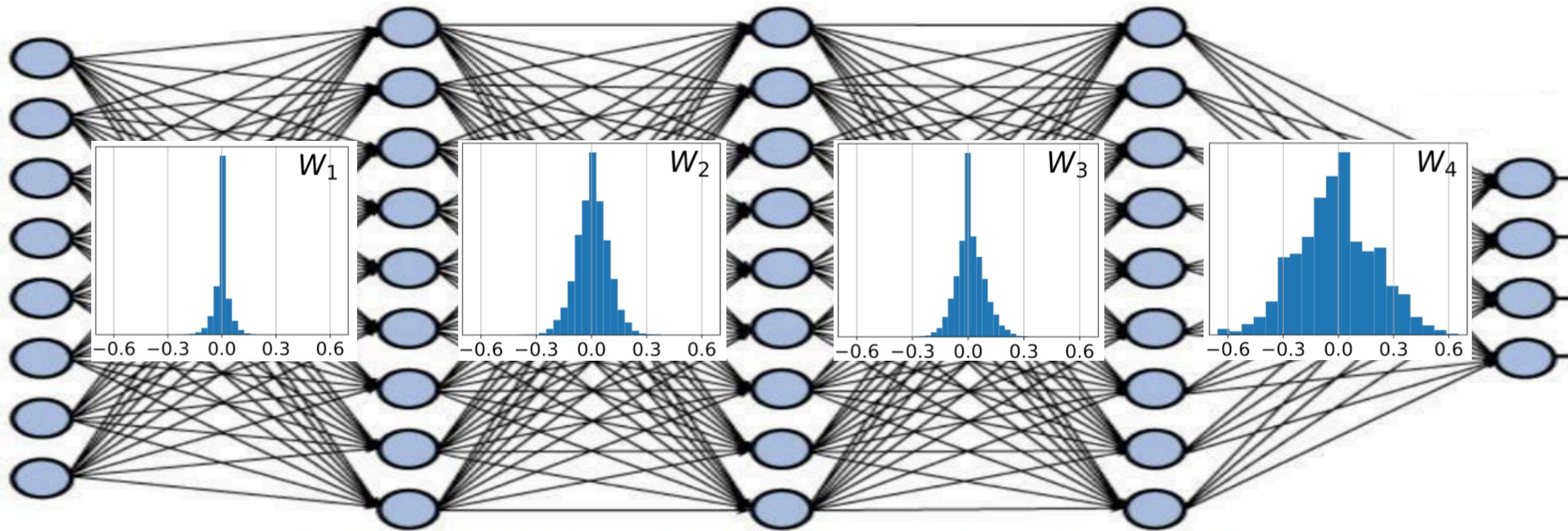
Structured Constraints:
Fixed in-, out-degrees
for every node



$$\rho_{\text{net}} = \frac{8 + 8}{32 + 16} = 33\%$$

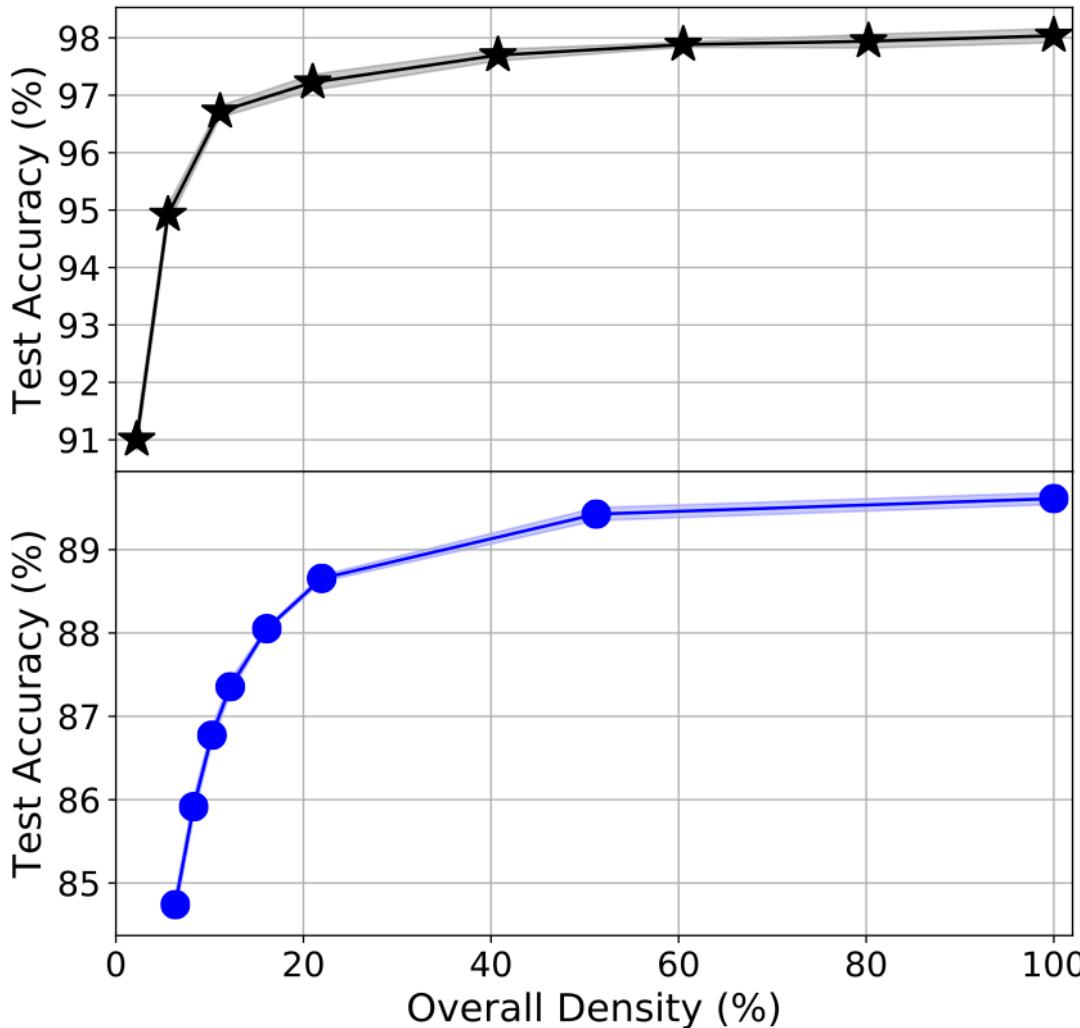
Overall Density
compared to FC

Motivation behind pre-defined sparsity

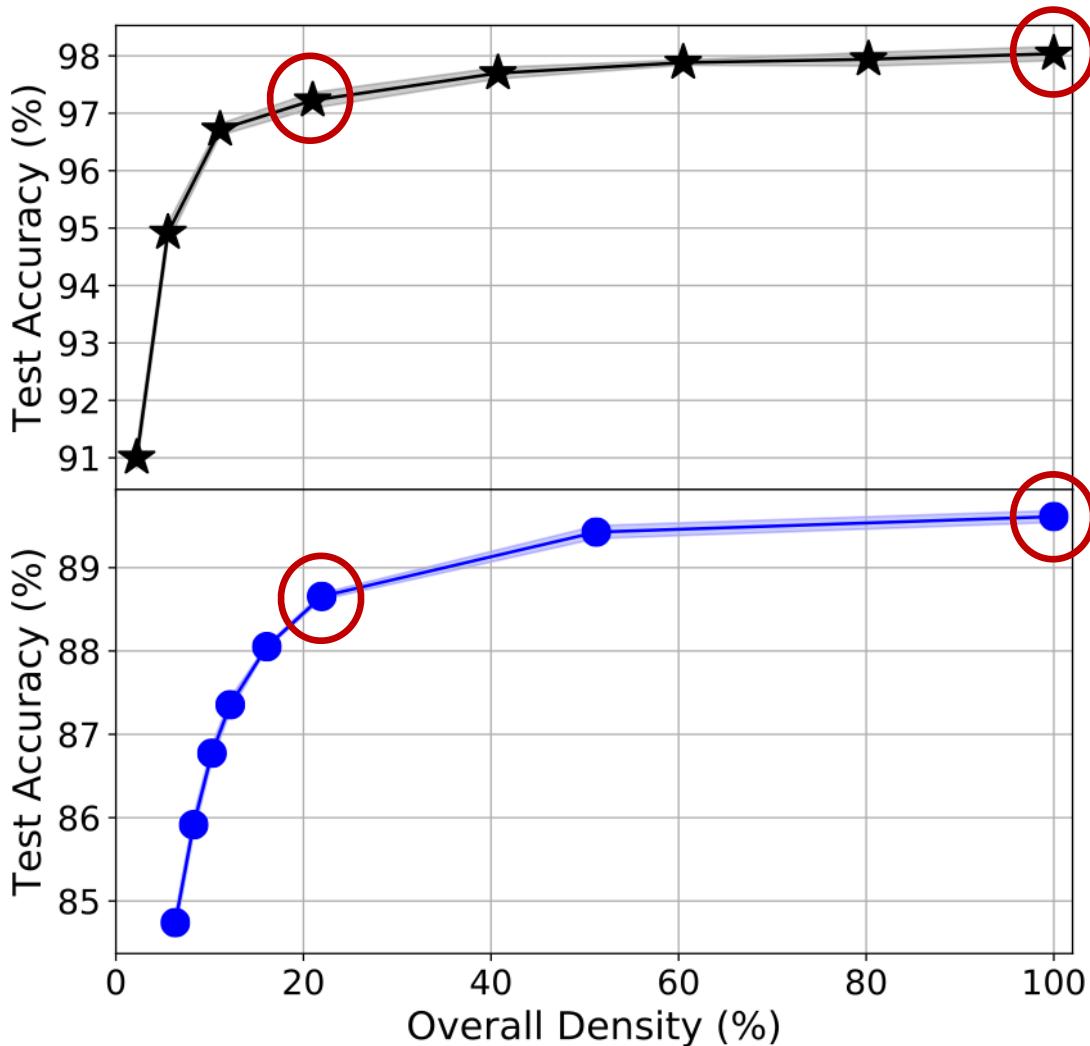


In a FC network, most weights are very small in magnitude after training

Pre-defined sparsity performance on MLPs

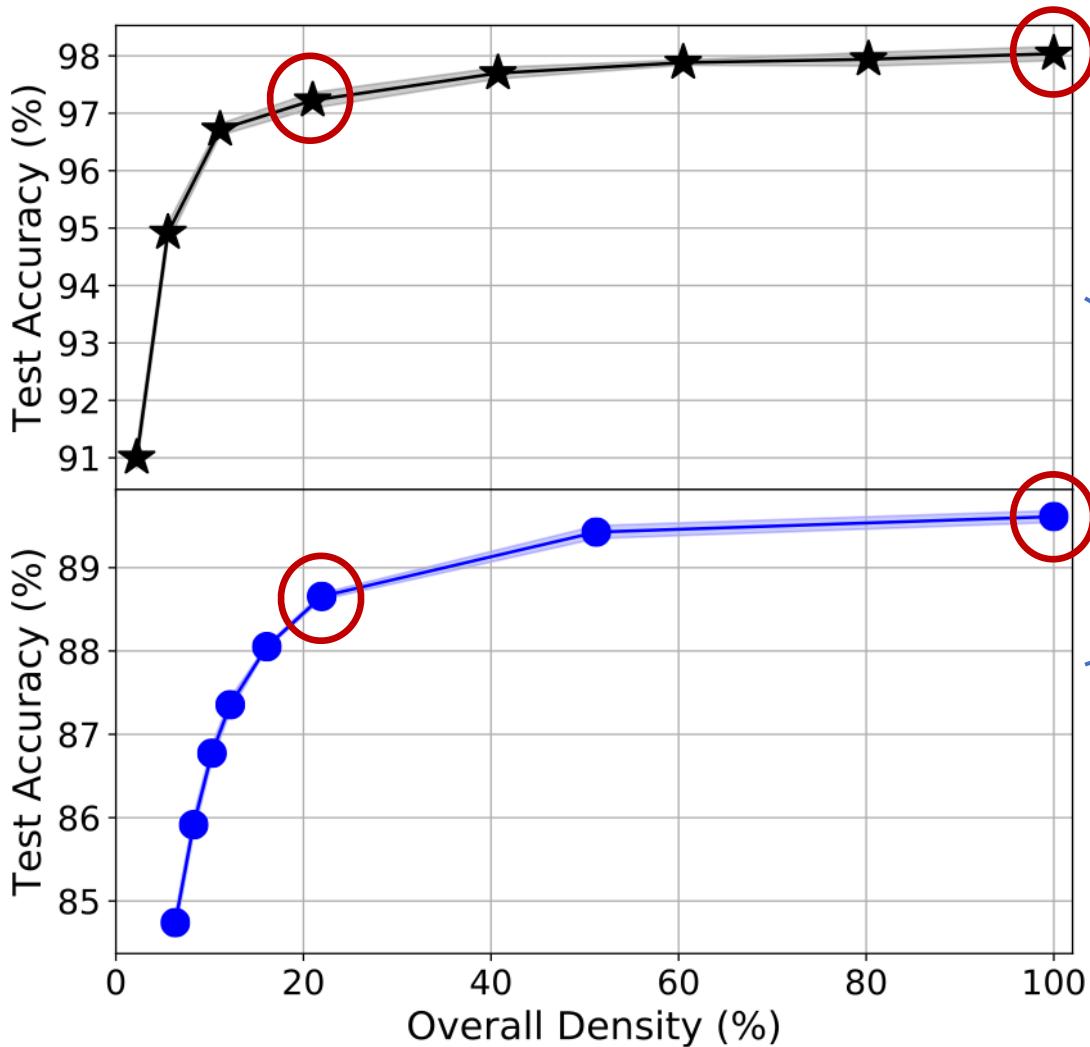


Pre-defined sparsity performance on MLPs



*Starting with only 20%
of parameters reduces
test accuracy by just 1%*

Pre-defined sparsity performance on MLPs



*Starting with only 20%
of parameters reduces
test accuracy by just 1%*

MNIST handwritten digits

Reuters news articles

TIMIT phonemes

CIFAR images

Morse symbols

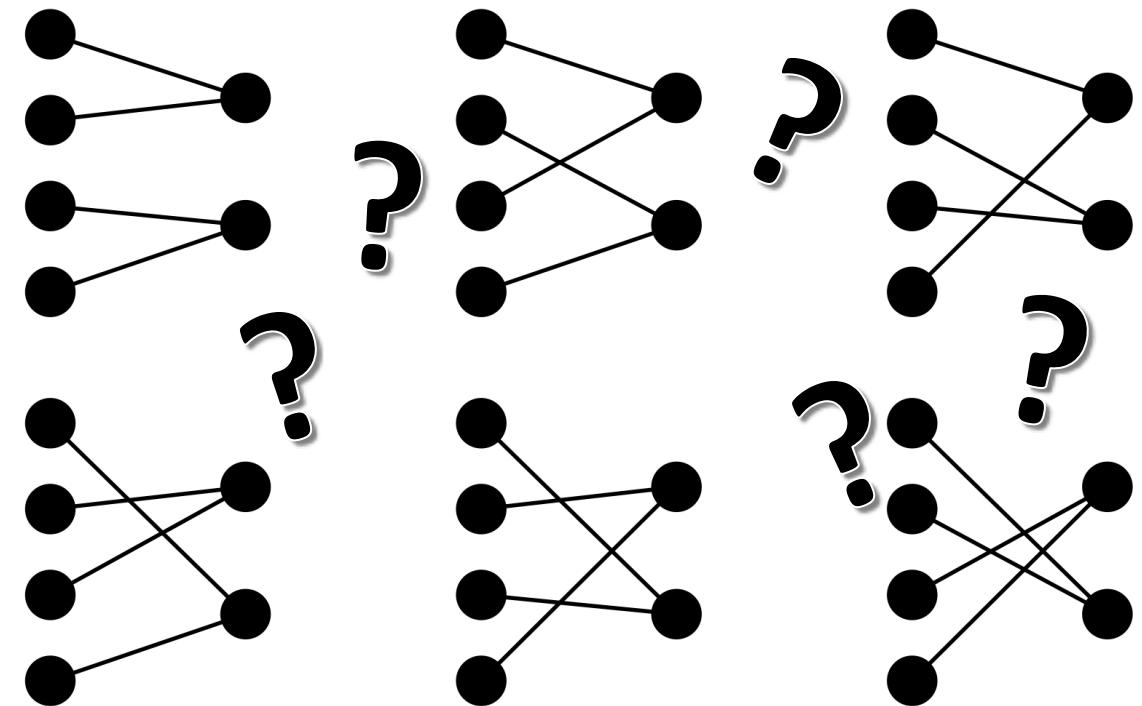
S. Dey, K. M. Chugg and P. A. Beerel, "Morse Code Datasets for Machine Learning," in ICCCNT 2018.
Won Best Paper award.
<https://github.com/usc-hal/morse-dataset>

Analysis and Applications

Deep dive into pre-defined sparsity
for MLPs, and a corresponding
application

Designing pre-defined sparse networks

*A pre-defined sparse connection pattern is a **hyperparameter** to be set prior to training*

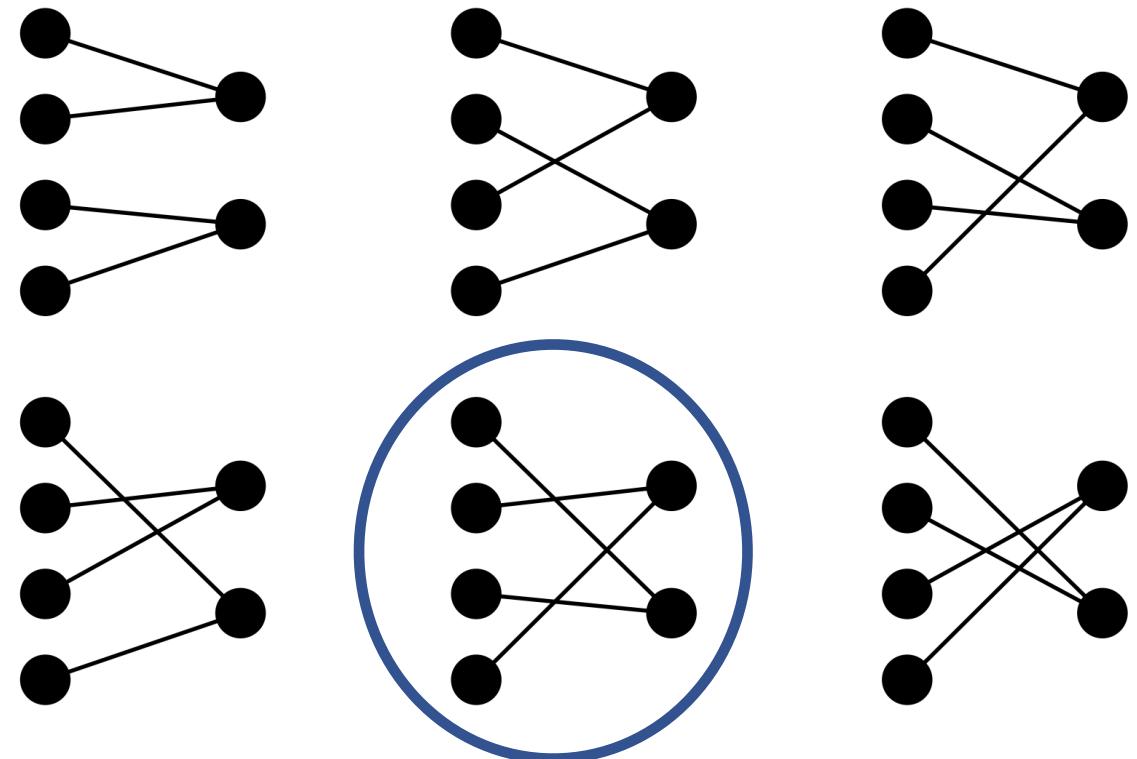


Designing pre-defined sparse networks

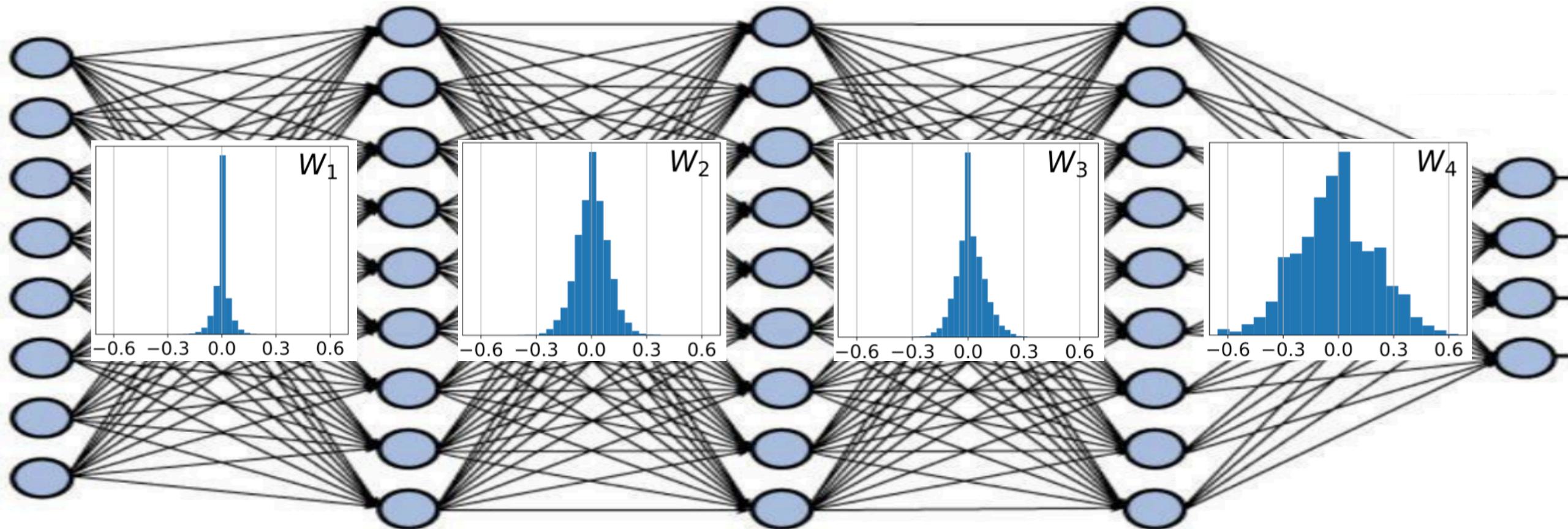
*A pre-defined sparse connection pattern is a **hyperparameter** to be set prior to training*

Find trends and guidelines to optimize pre-defined sparse patterns

S. Dey, K. Huang, P. A. Beerel and K. M. Chugg, "Pre-Defined Sparse Neural Networks with Hardware Acceleration," in *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 2, pp. 332-345, June 2019.



Individual junction densities

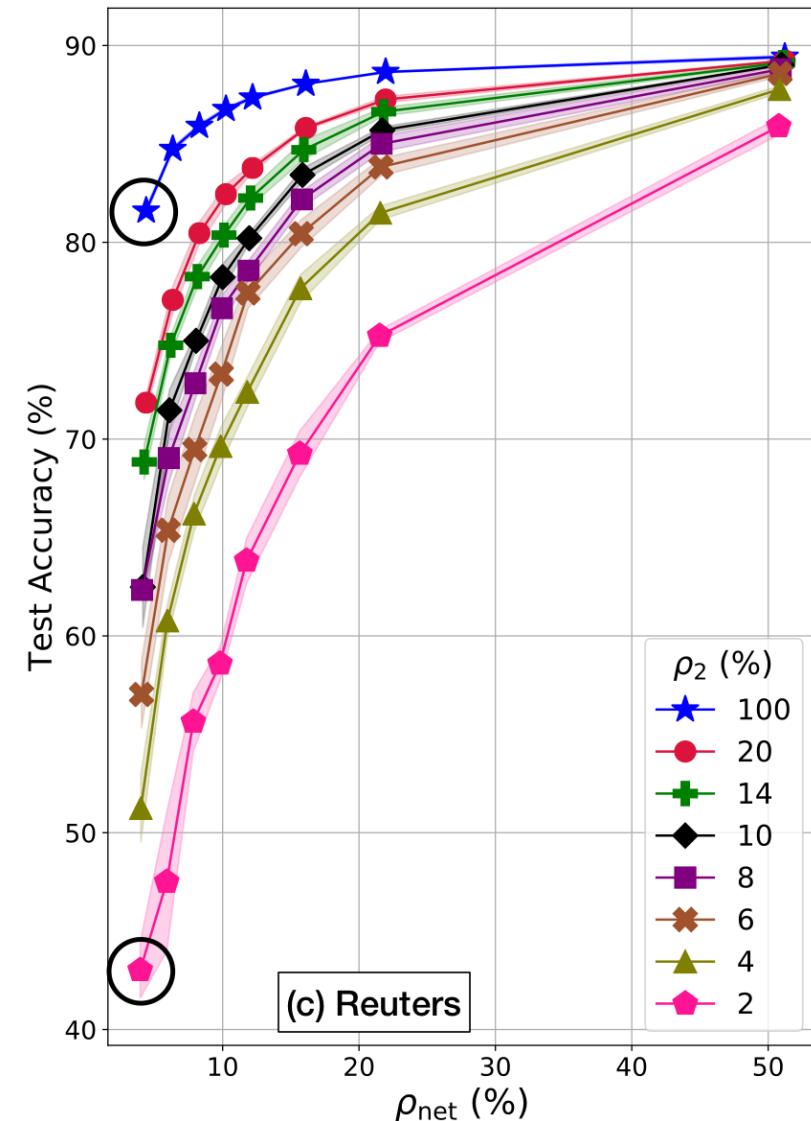
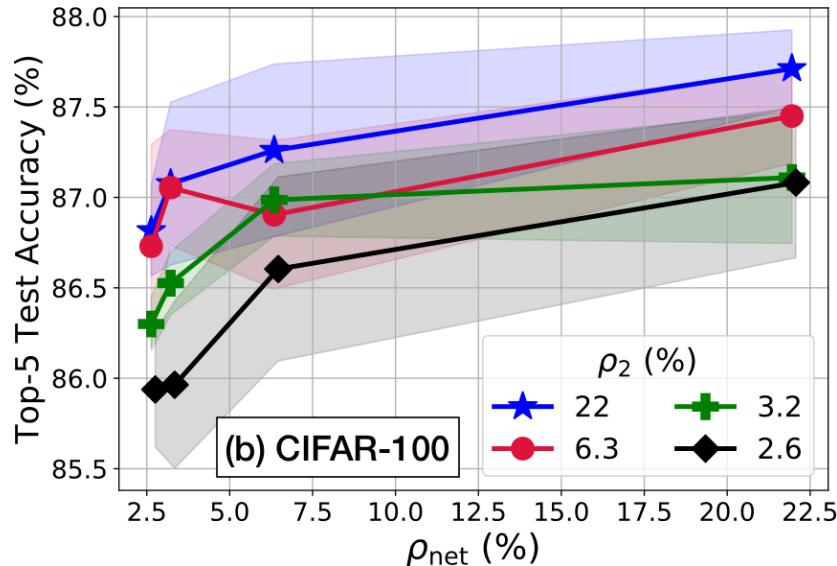
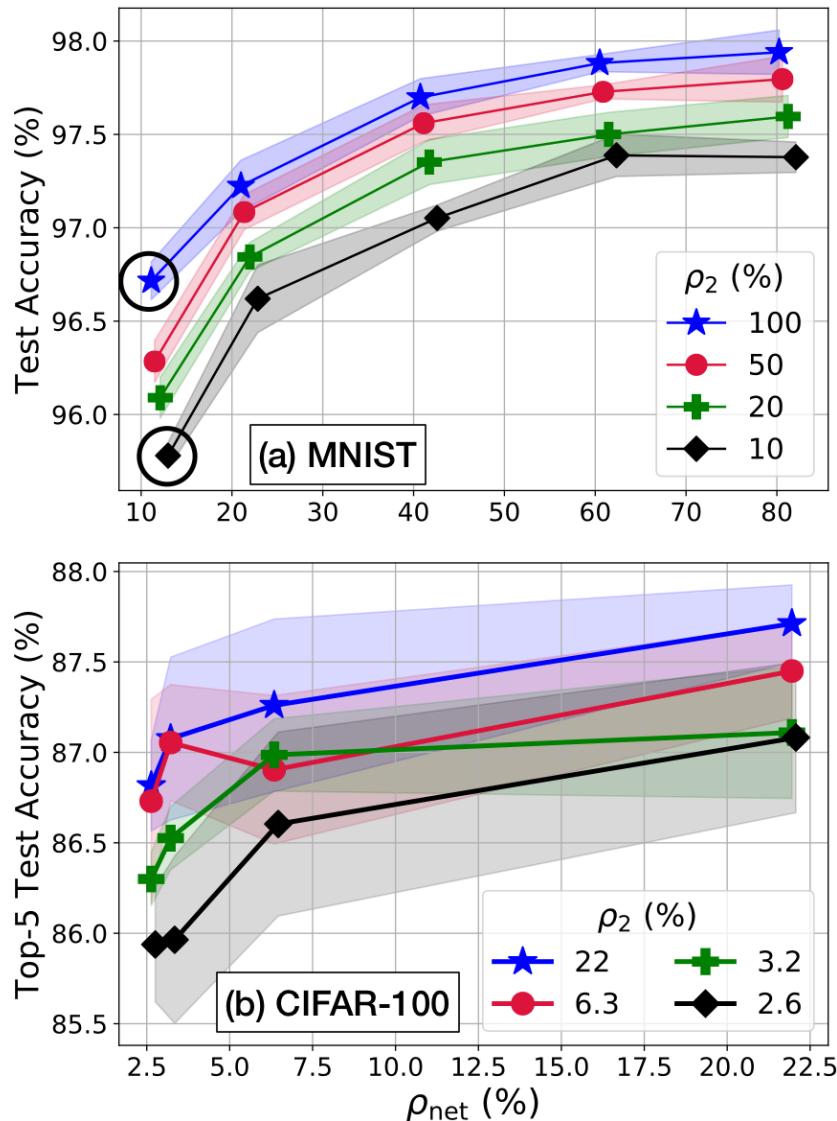


Latter junctions (closer to the output) need to be denser

Individual junction densities

Each curve keeps ρ_2 fixed and varies ρ_{net} by varying ρ_1

For the same ρ_{net} , $\rho_2 > \rho_1$ improves performance

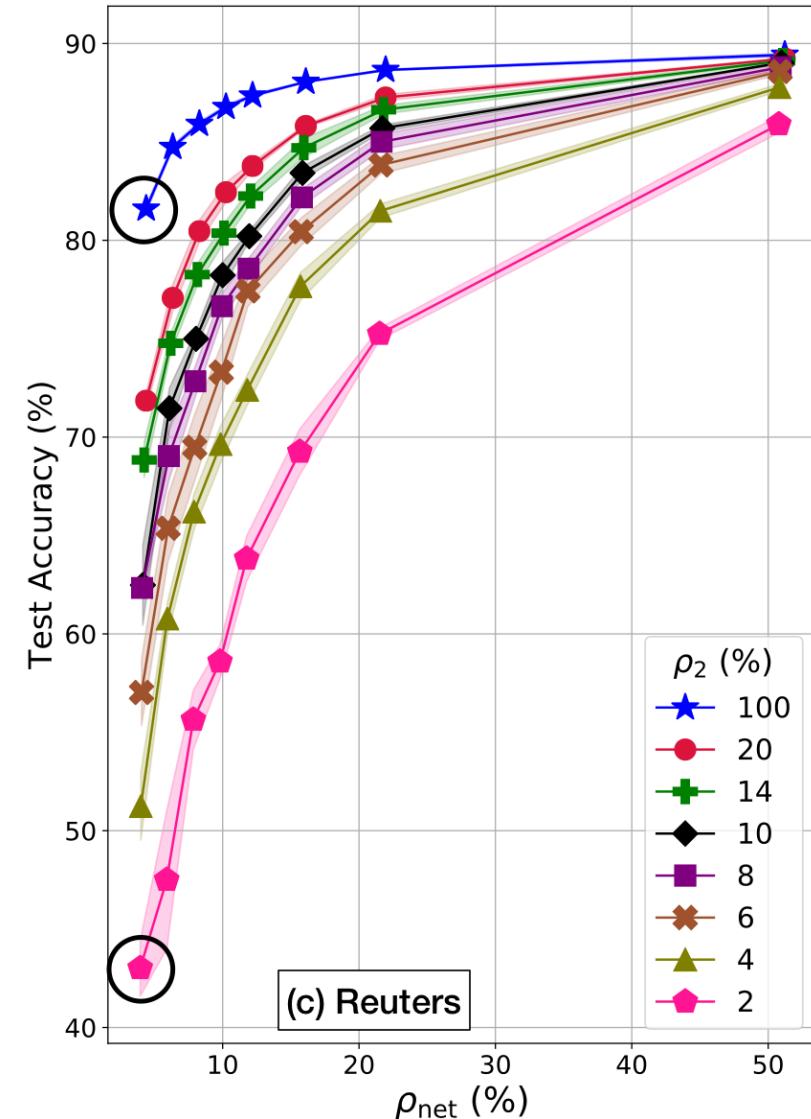
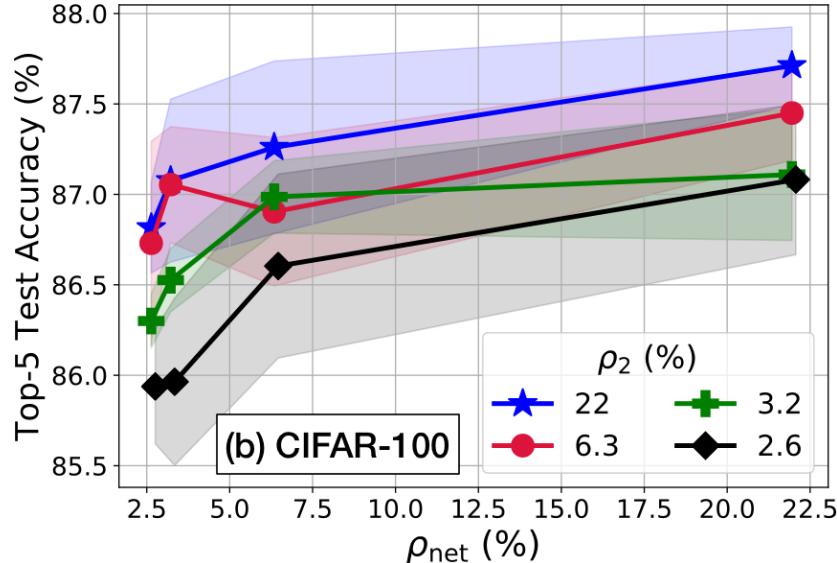
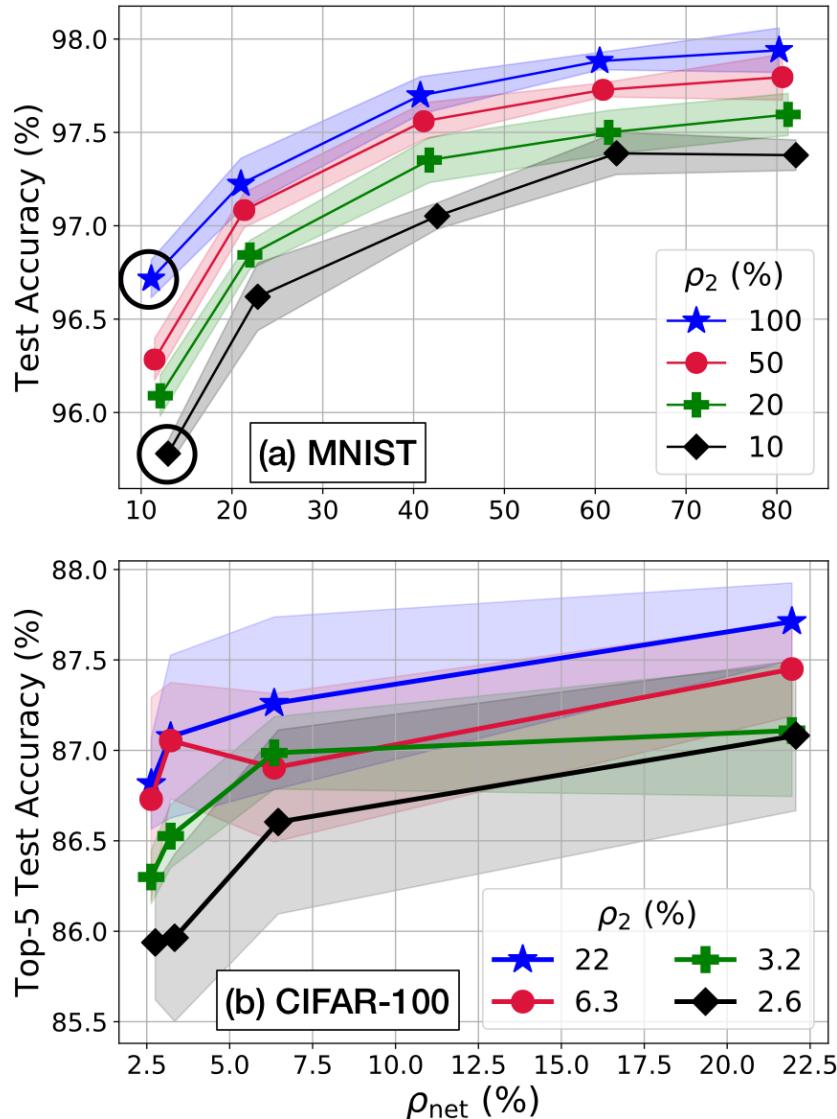


Individual junction densities

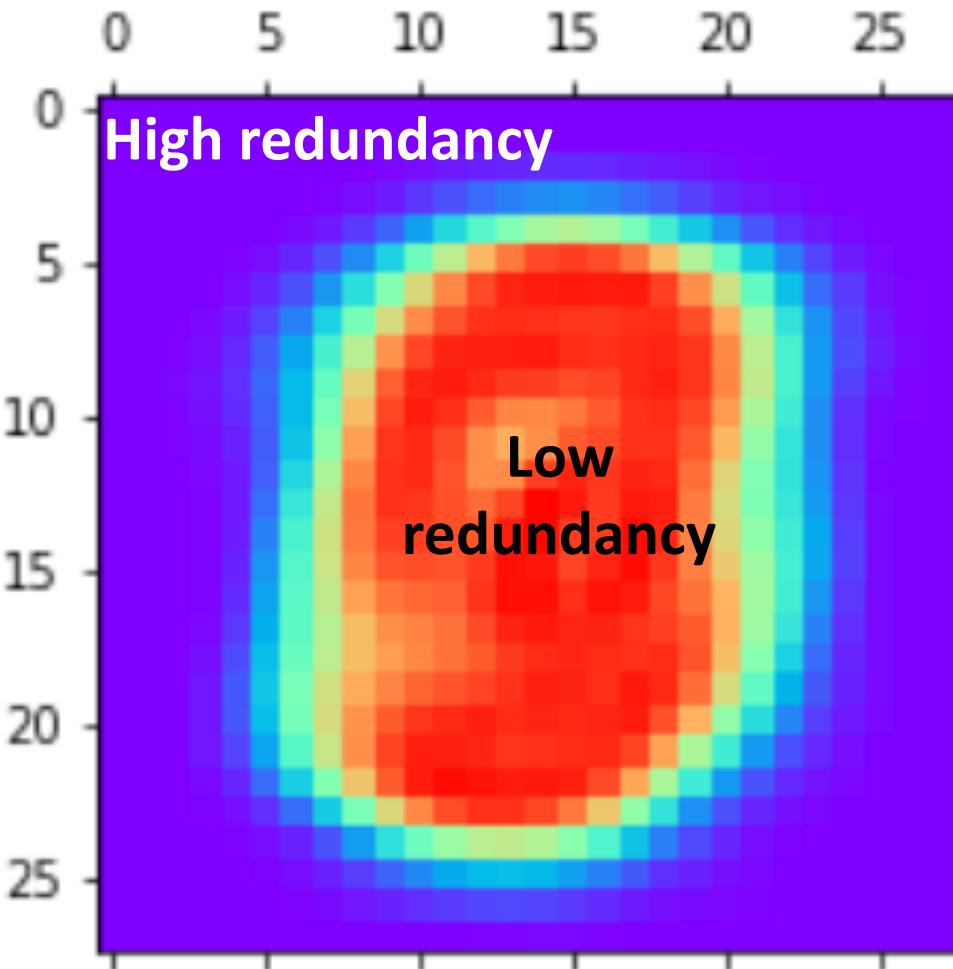
Each curve keeps ρ_2 fixed and varies ρ_{net} by varying ρ_1

For the same ρ_{net} , $\rho_2 > \rho_1$ improves performance

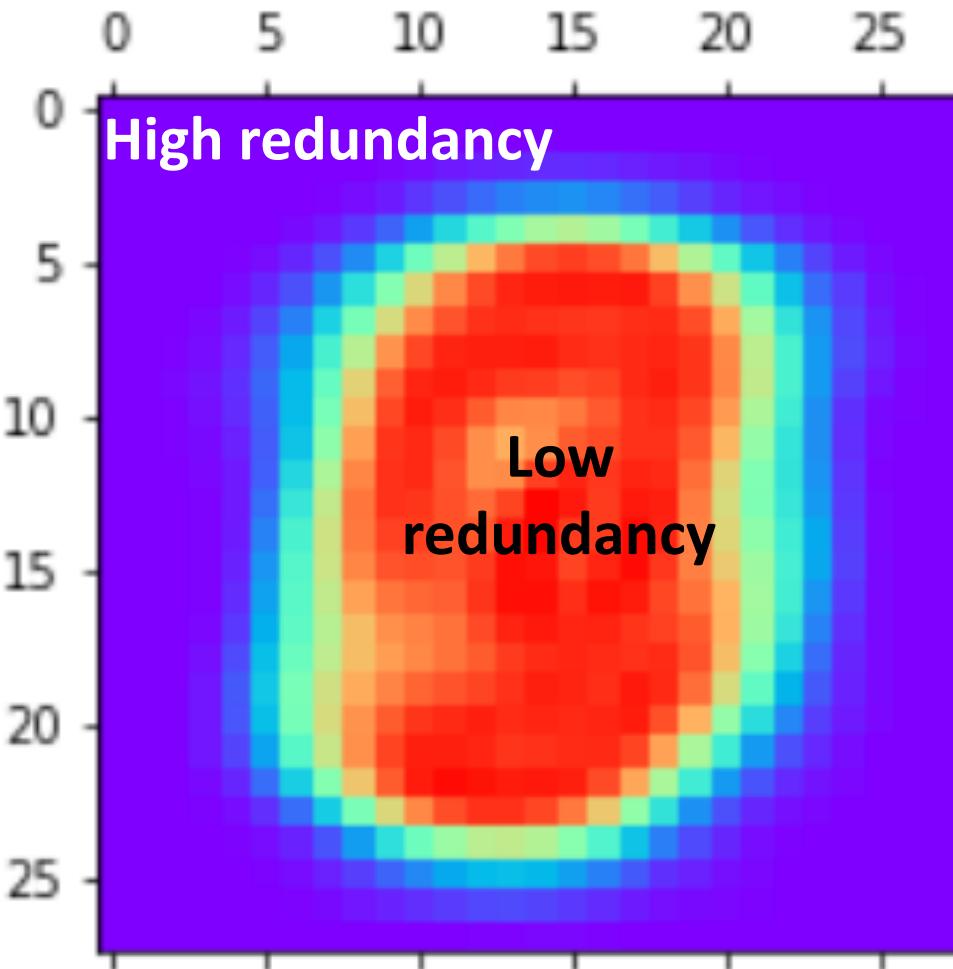
Mostly similar trends observed for deeper networks



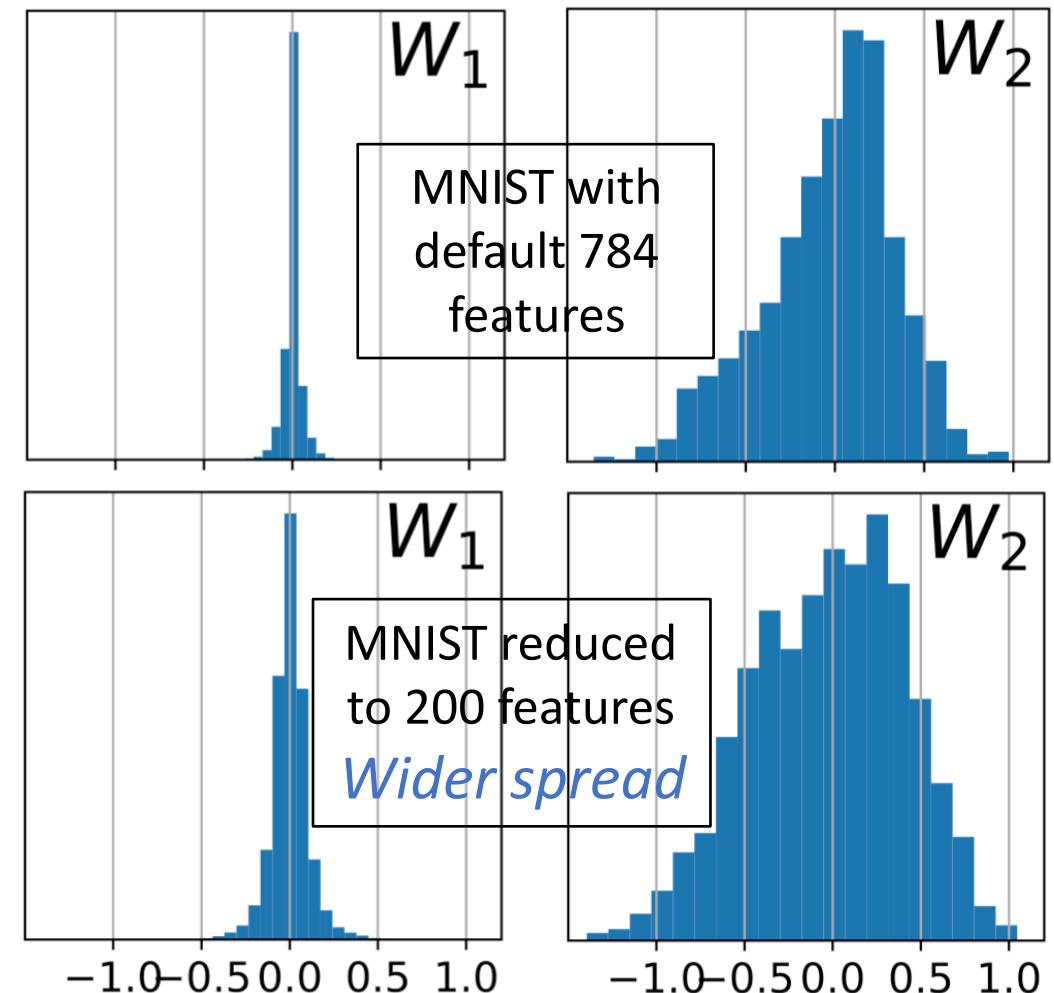
Dataset redundancy



Dataset redundancy



Sourya Dey



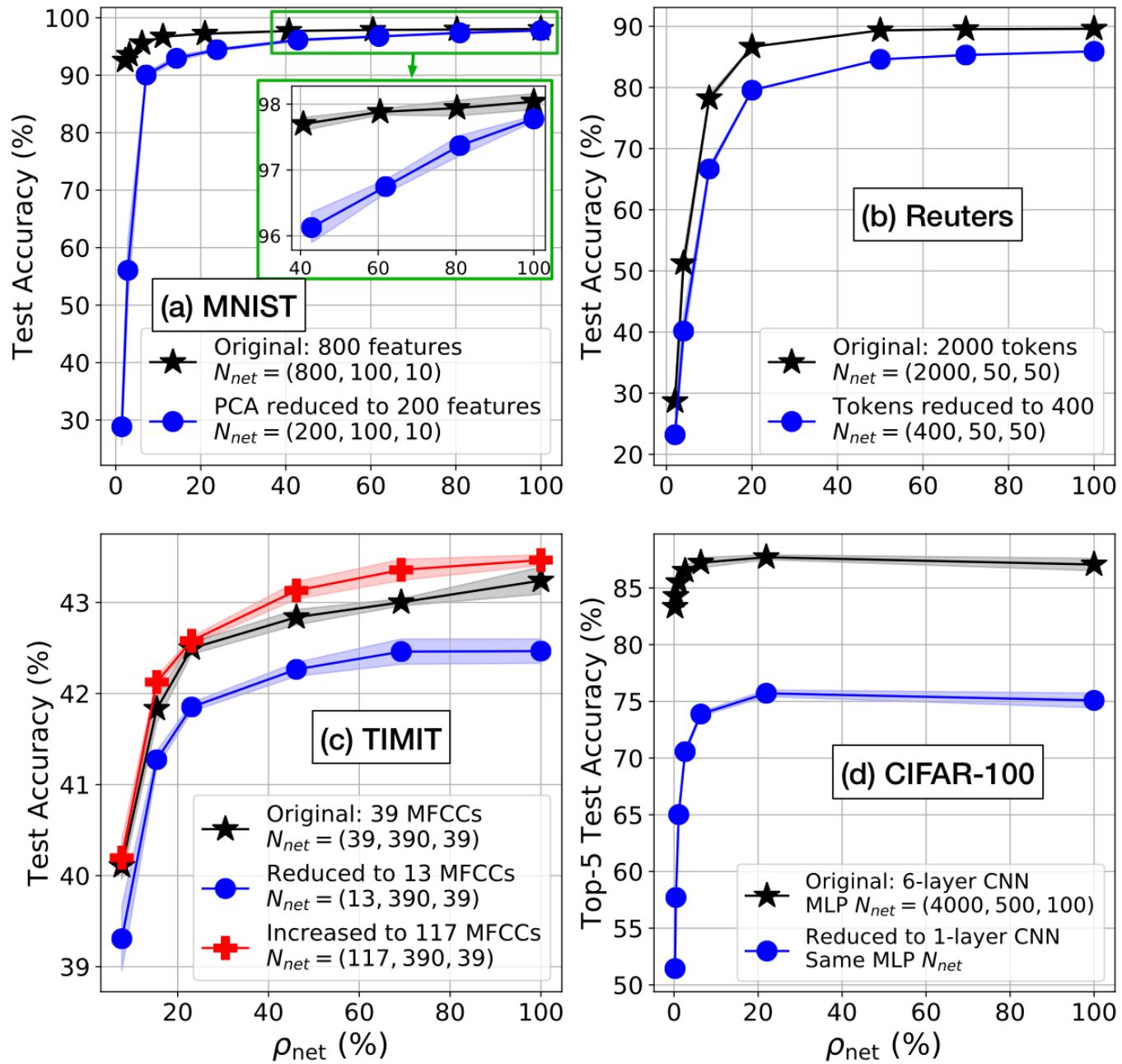
*Less redundancy => Less
sparsification possible*

ifornia

14

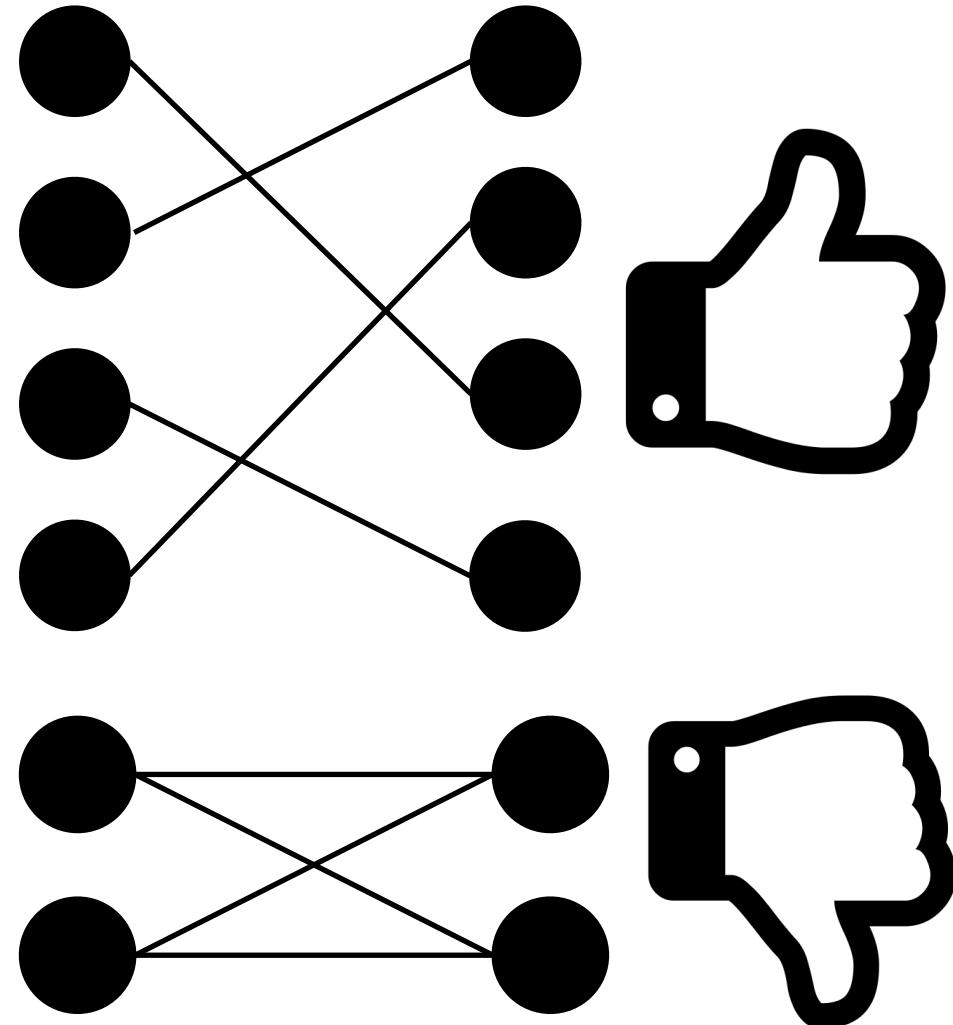
Effect of redundancy on sparsity

Reducing redundancy leads to increased performance degradation on sparsification



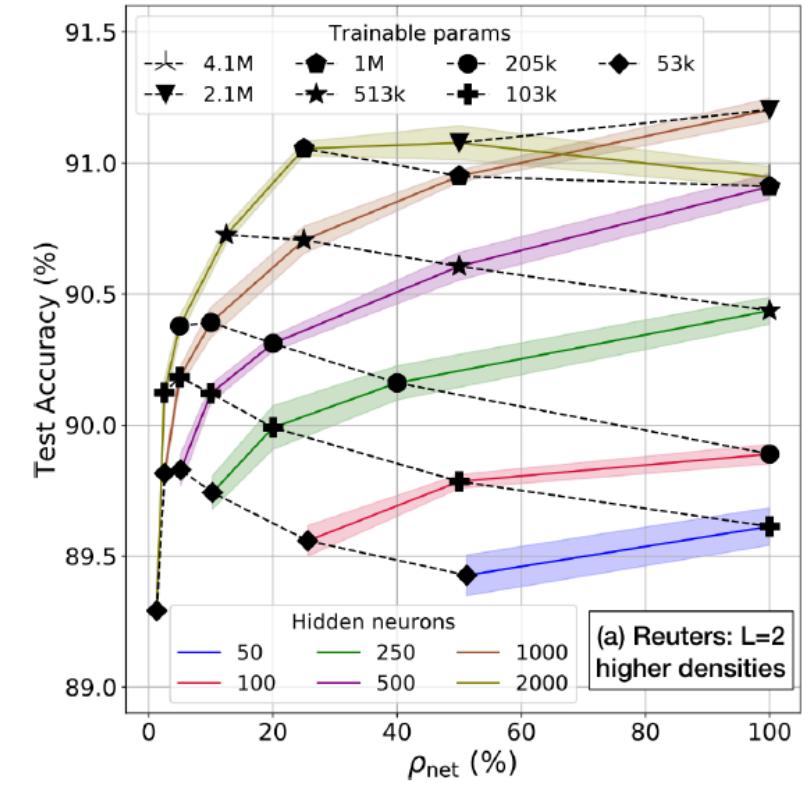
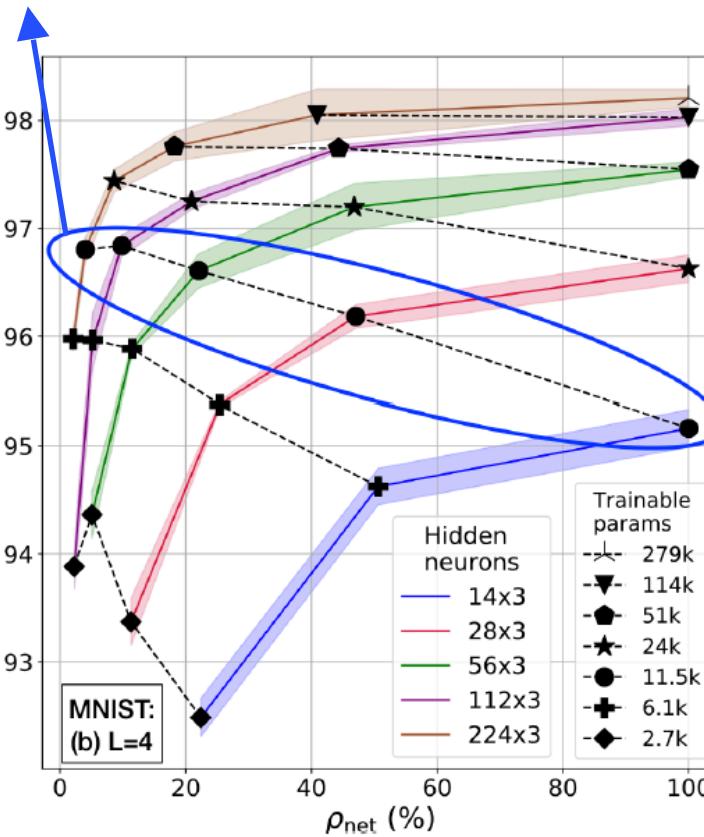
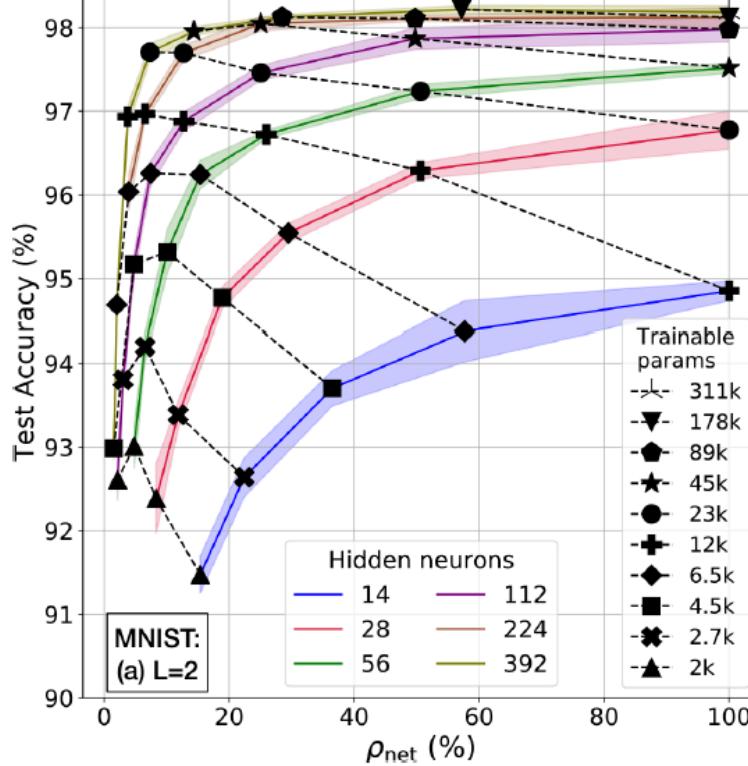
'Large sparse' vs 'small dense' networks

A sparser network with more hidden nodes will outperform a denser network with less hidden nodes, when both have same number of weights



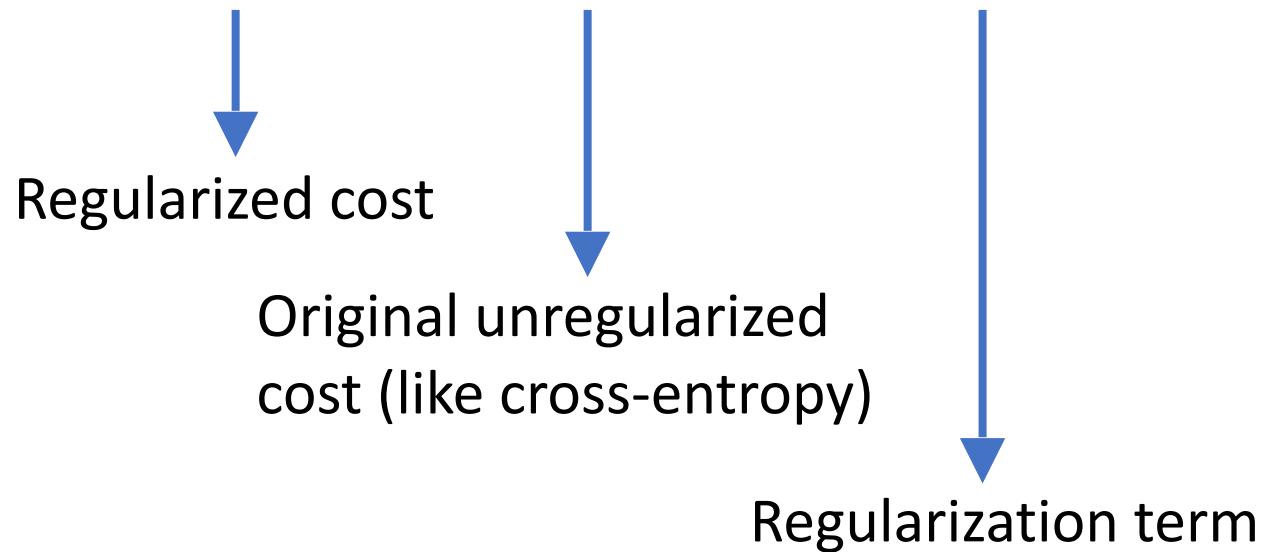
‘Large sparse’ vs ‘small dense’ networks

Networks with same number of parameters go from bad to good as #nodes in hidden layers is increased



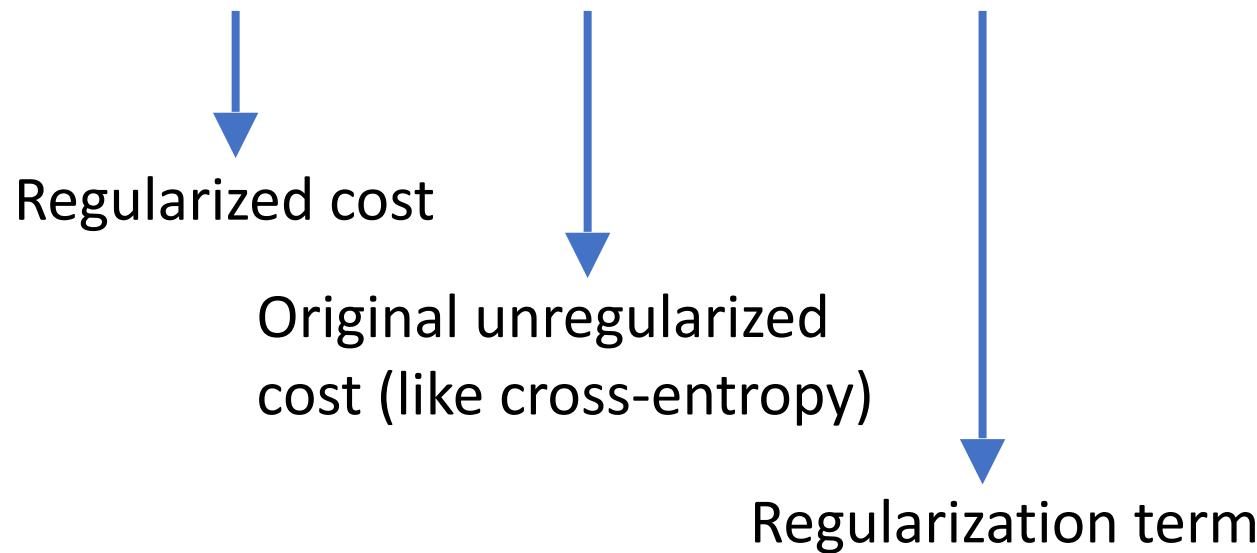
Regularization

$$C(\mathbf{w}) = C_0(\mathbf{w}) + \lambda \|\mathbf{w}\|_2^2$$



Regularization

$$C(\mathbf{w}) = C_0(\mathbf{w}) + \lambda \|\mathbf{w}\|_2^2$$



Pre-defined sparse networks need smaller λ (as determined by validation)

Overall Density	λ
100 %	1.1×10^{-4}
40 %	5.5×10^{-5}
11 %	0

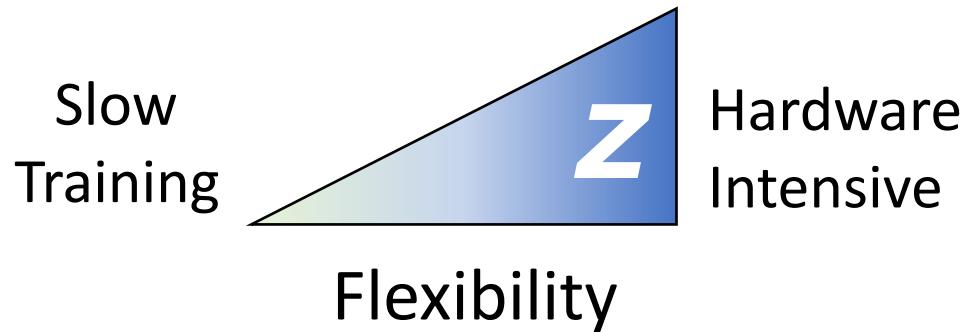
Example for MNIST 2-junction networks

Pre-defined sparsity reduces the overfitting problem stemming from over-parametrization in big networks

Application: A hardware architecture for
on-device training and inference

Application: A hardware architecture for on-device training and inference

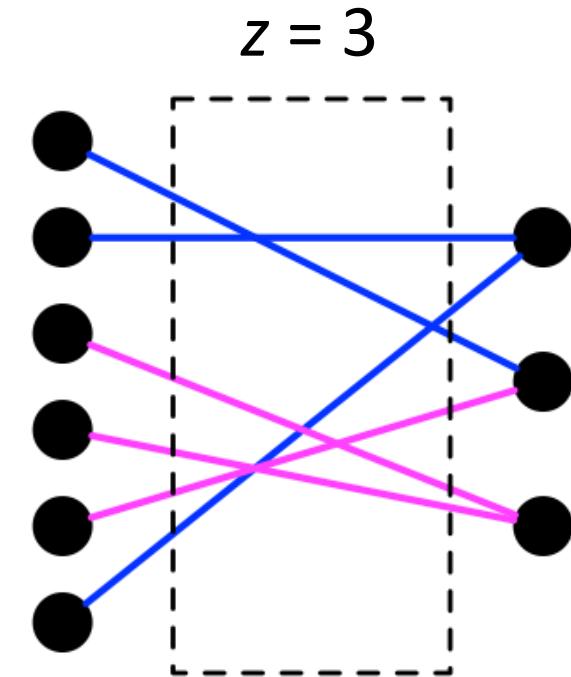
Degree of parallelism (z) = Number of weights processed in parallel in a junction



Application: A hardware architecture for on-device training and inference

Degree of parallelism (z) = Number of weights processed in parallel in a junction

Connections designed for clash-free memory accesses to prevent stalling

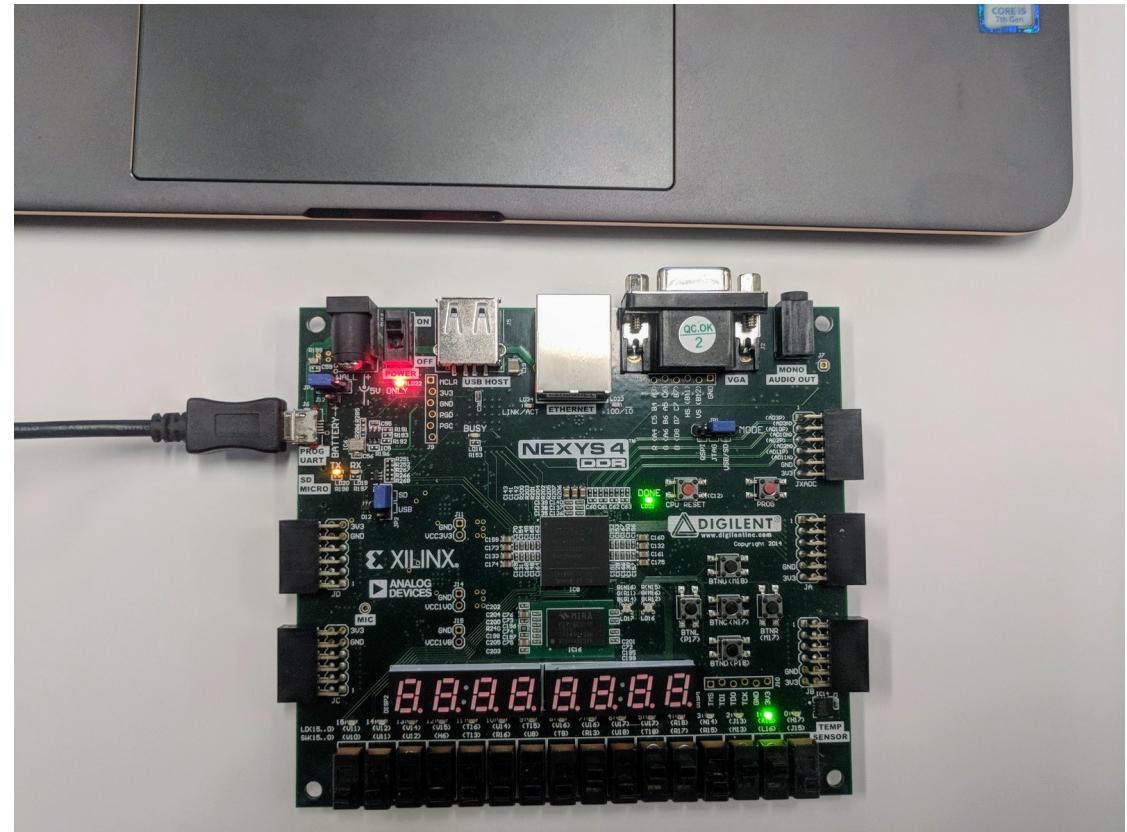


Application: A hardware architecture for on-device training and inference

Degree of parallelism (z) = Number of weights processed in parallel in a junction

Connections designed for clash-free memory accesses to prevent stalling

Prototype implemented on FPGA



S. Dey, D. Chen, Z. Li, S. Kundu, K. Huang, K. M. Chugg and P. A. Beerel, "A Highly Parallel FPGA Implementation of Sparse Neural Network Training," in 2018 International Conference on Reconfigurable Computing and FPGAs (ReConFig), pp.1-4, Dec 2018. Expanded pre-print version available at arXiv:1806.01087.

Model Search

Automate the design of CNNs
with good performance and
low complexity

Model search is ongoing
research, hence currently
not available publicly

Thank you!

<https://souryadey.github.io/>

