DNN ENGINE: A 16nm Sub-uJ DNN Inference Accelerator for the Embedded Masses

Paul N. Whatmough[1,2]

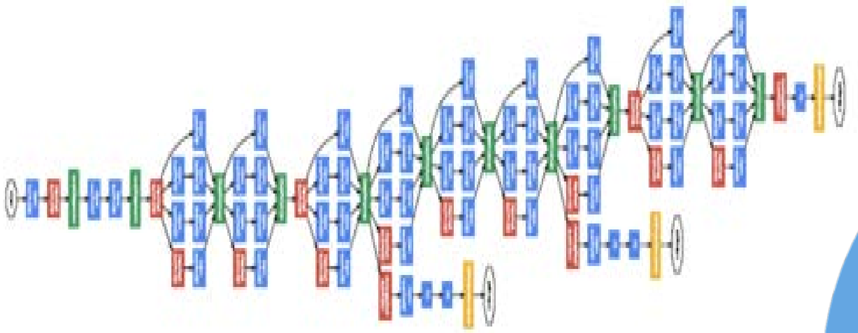
S. K. Lee[2], N. Mulholland[2], P. Hansen[2], S. Kodali[3], D. Brooks[2], G.-Y. Wei[2]

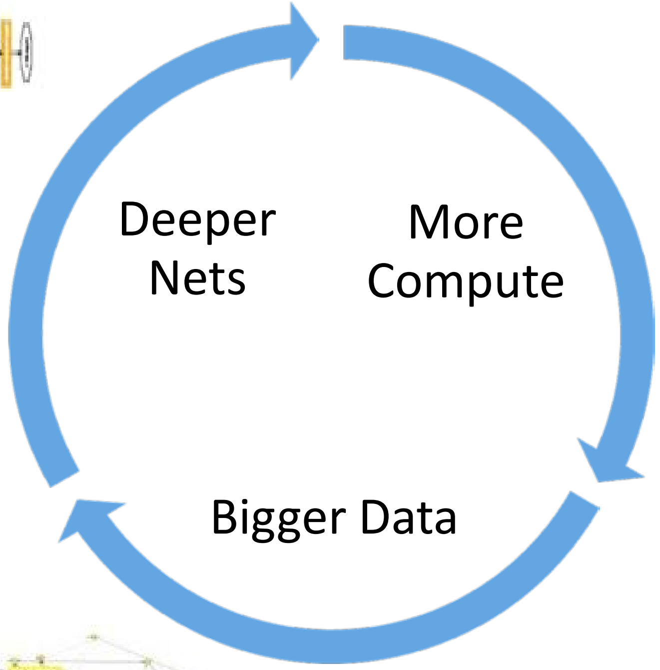
[1]ARM Research, Boston, MA

[2]Harvard University, Cambridge, MA

[3]Princeton University, NJ

The age of deep learning



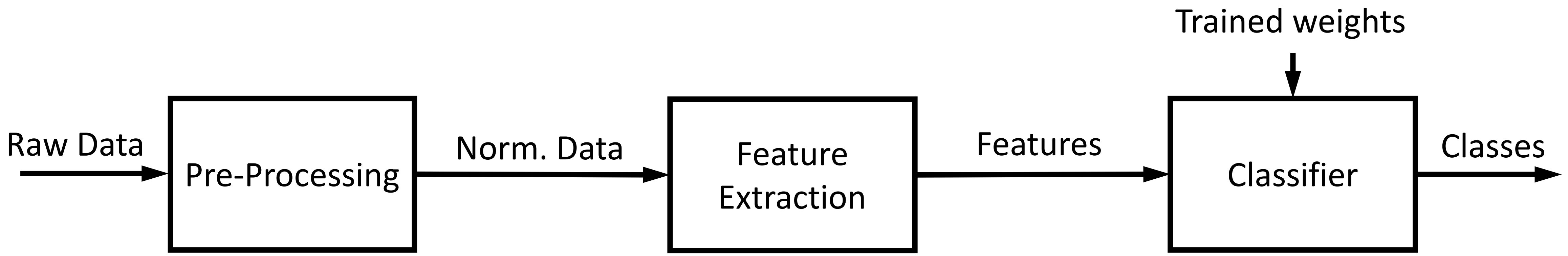


IMAGENET

The embedded masses

- Interpret noisy real-world data from sensor-rich systems
- Keep inference at the edge: always-on sensing
- Large memory footprint and compute load

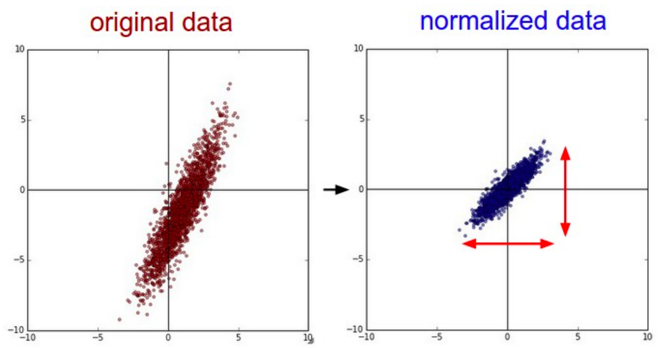
DNN inference at the edge

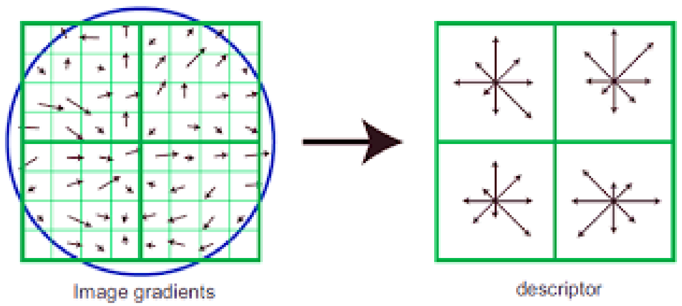


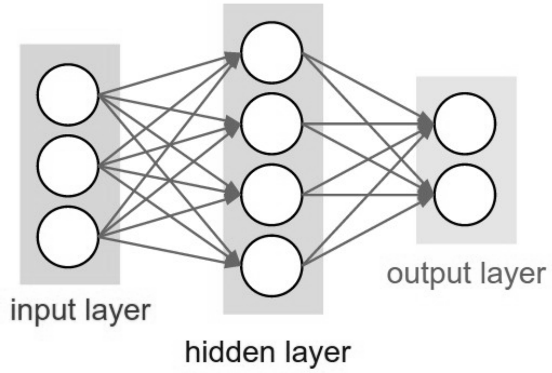
Normalize
mean=0, variance=1

FFT, MFCC,
Sobel, HOG, etc

Fully-connected
Deep neural network
(DNN)





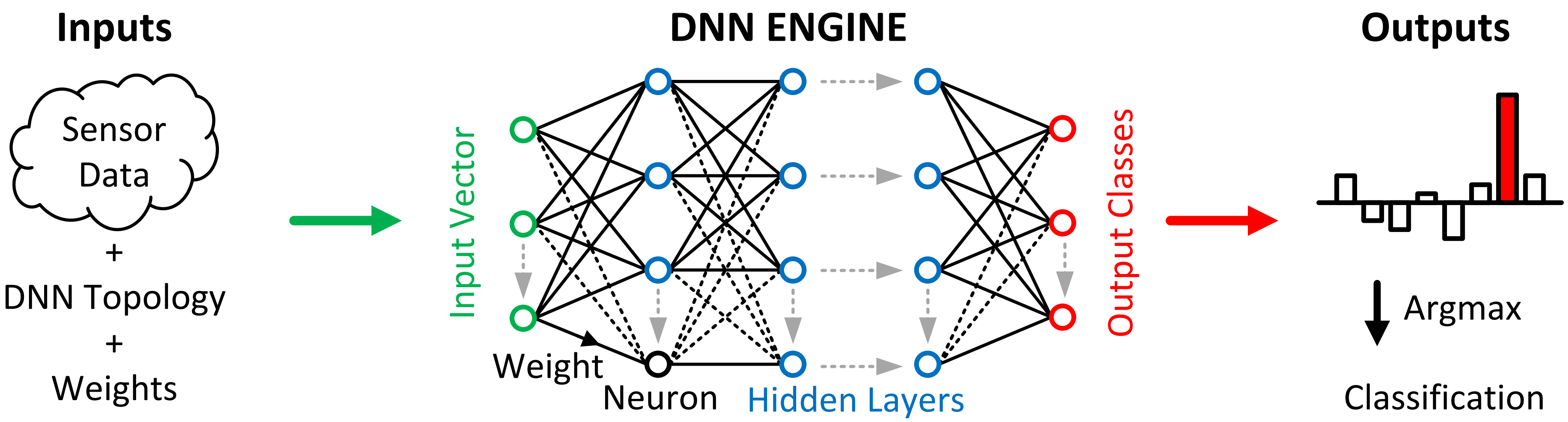


DNN classifier design flow

[Reagen et al., ISCA 2016]

- Data
  - Training, validation, test
- Training
  - Optimize hyper-parameters
  - Minimize size and test error
- Quantization
  - Fixed-point 8-bit / 16-bit
- Inference
  - CPU, DSP, GPU, Accelerator

DNN ENGINE accelerator architecture

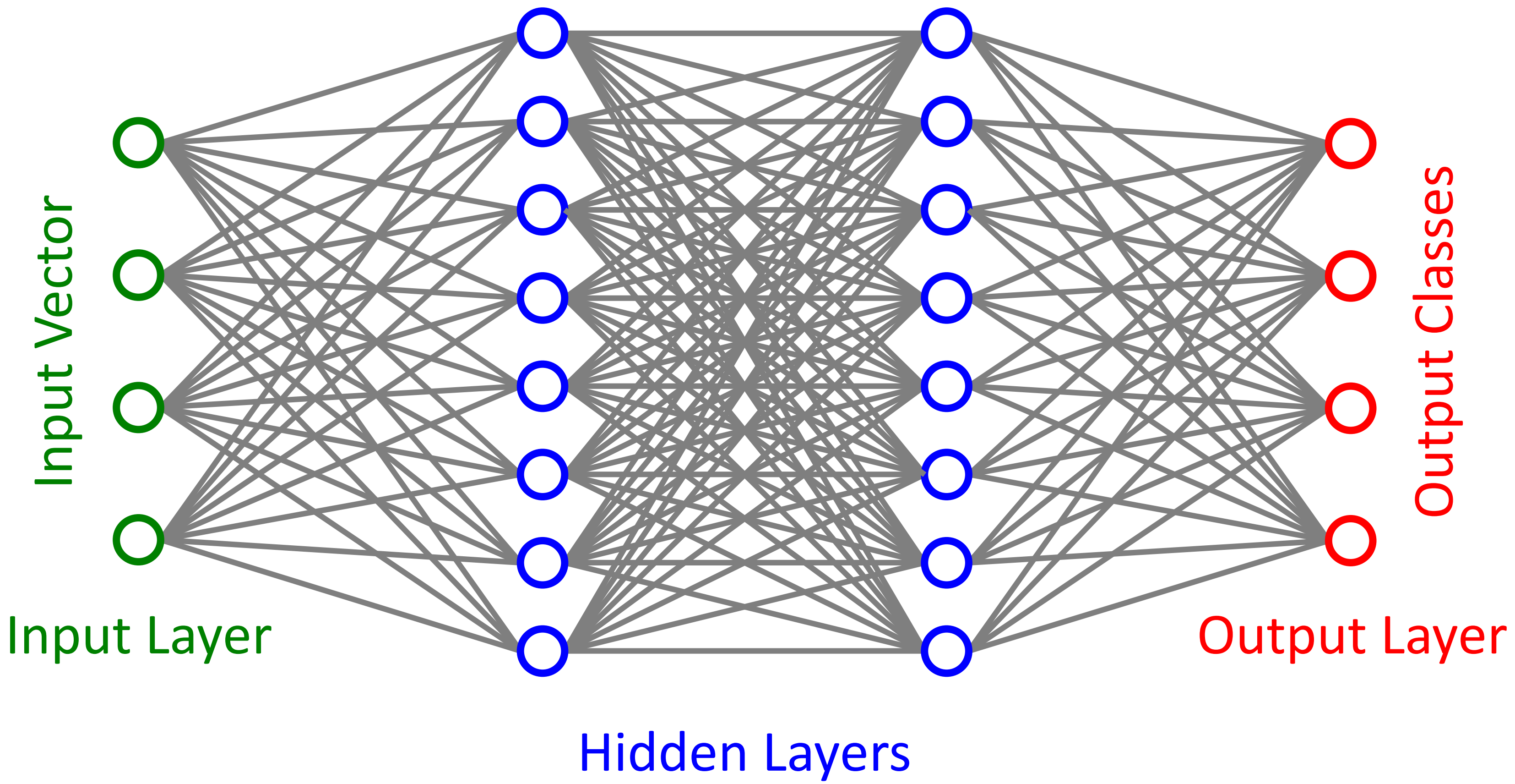


- Parallelism and data reuse
- Sparse data and small data types
- Algorithmic resilience

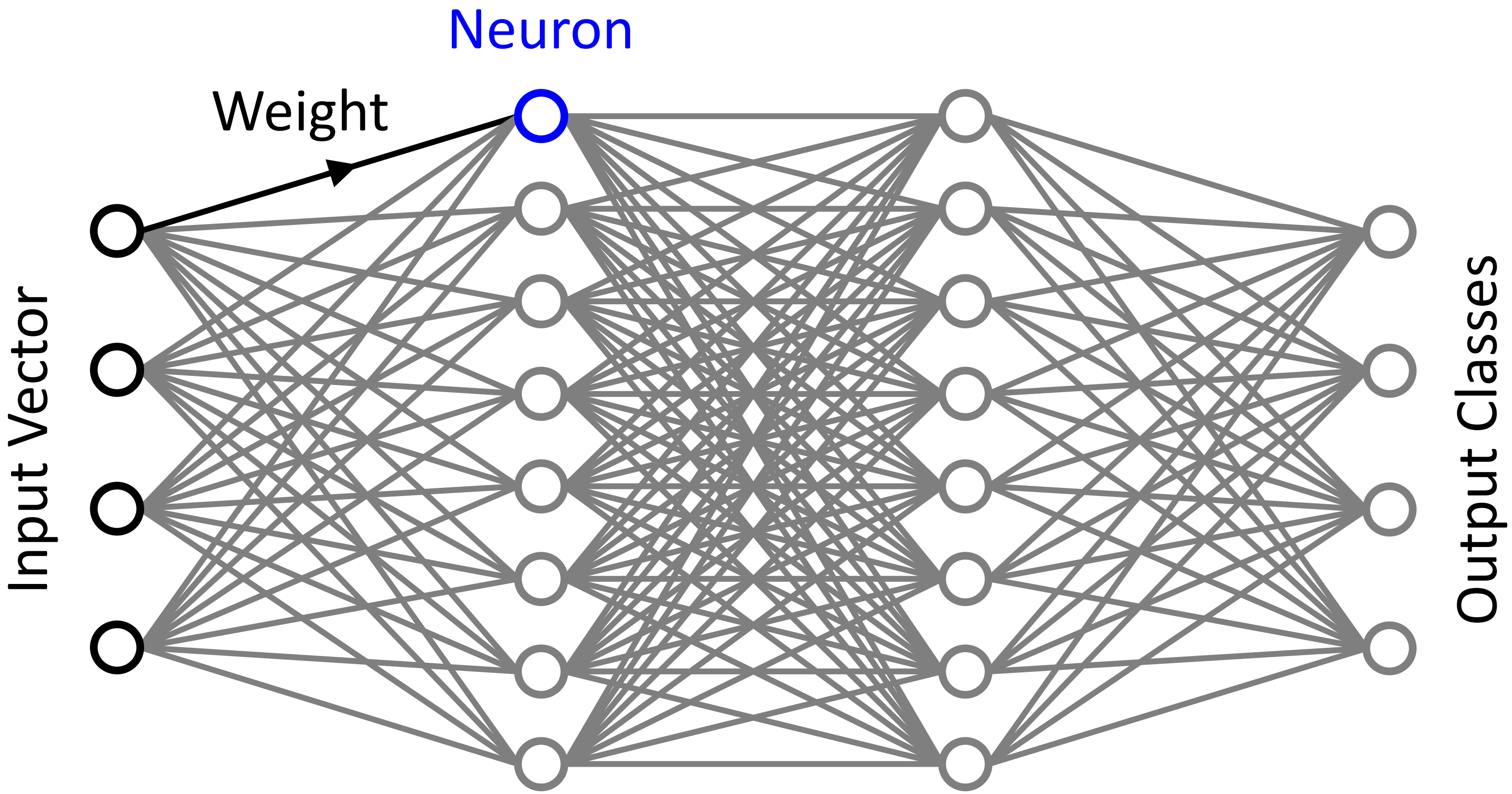
Outline

- Background and motivation
- DNN ENGINE
  - Parallelism and data reuse
  - Sparse data and small data types
  - Algorithmic resilience
- Measurement Results
- Summary

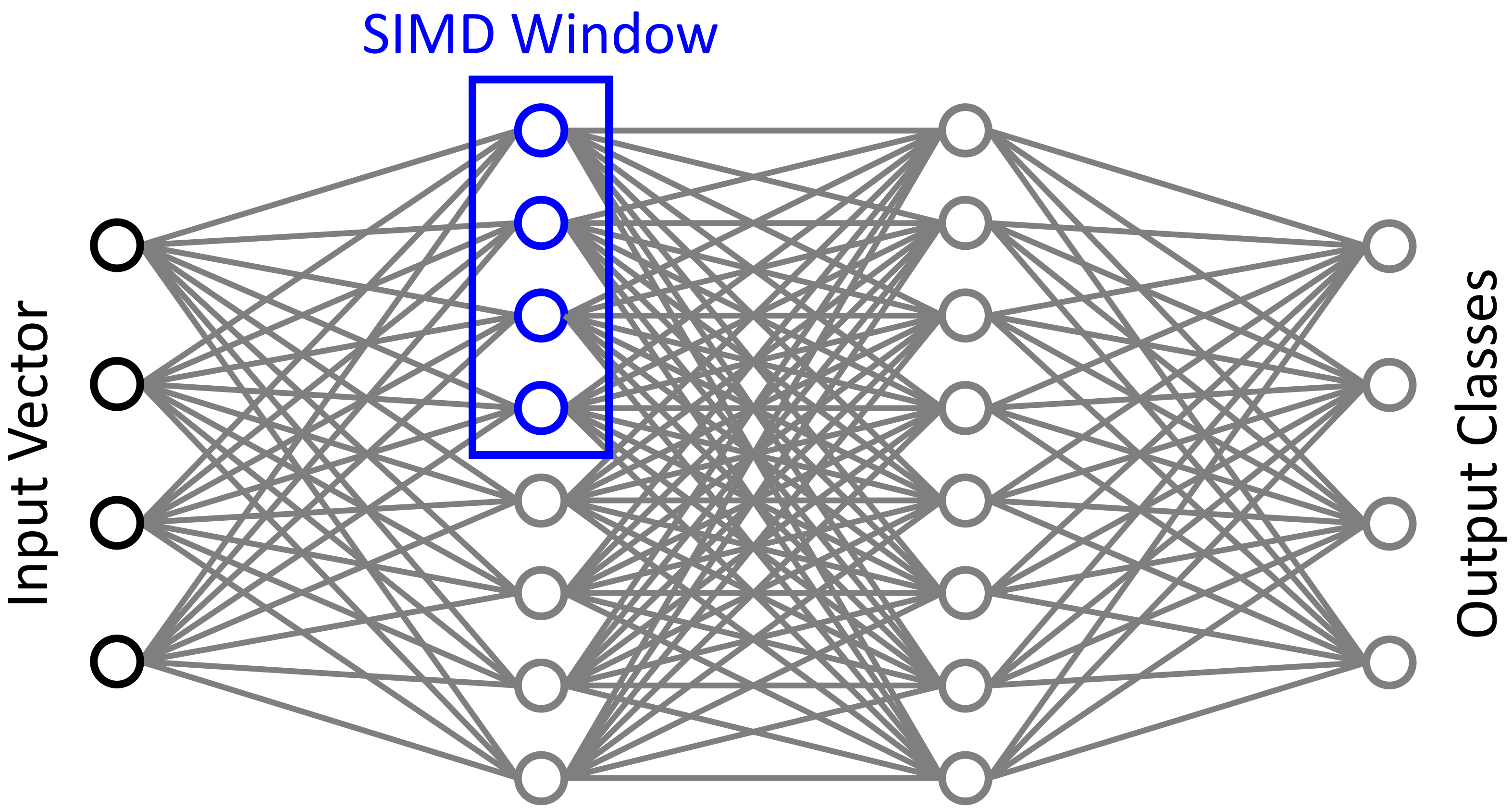
Fully-connected DNN graph



Fully-connected DNN graph

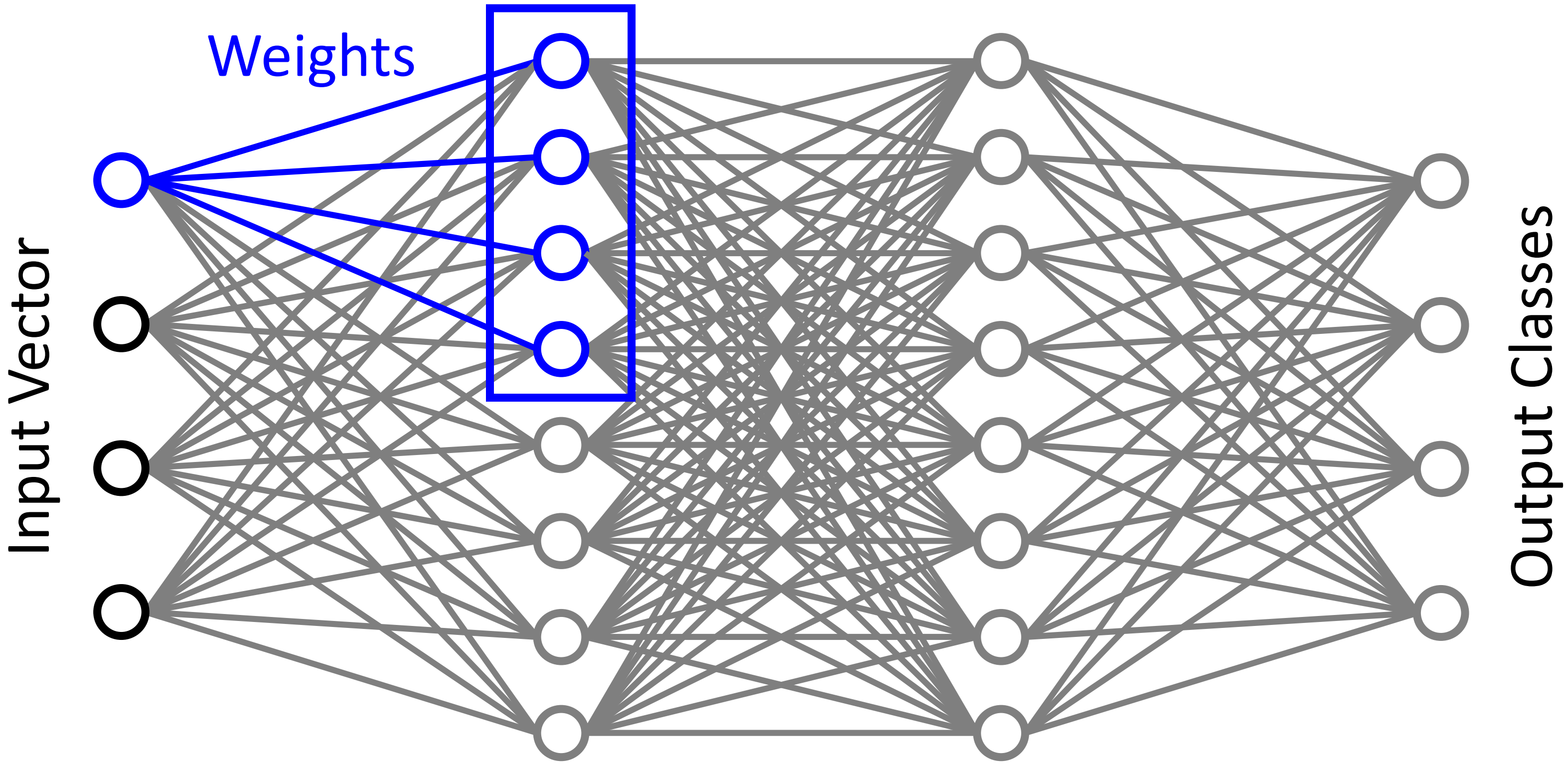


Fully-connected DNN graph



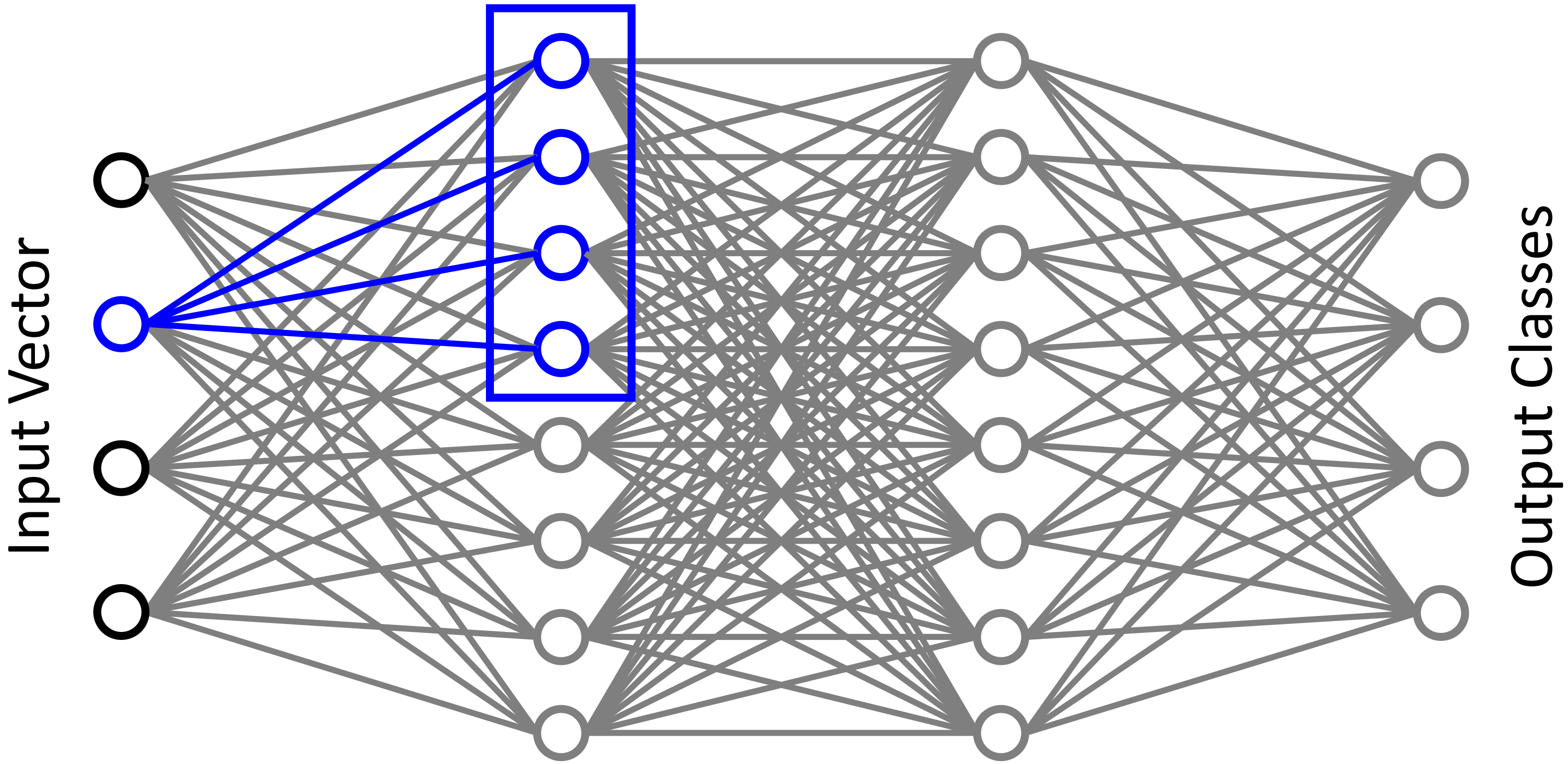
Process a group of neurons in parallel

Fully-connected DNN graph

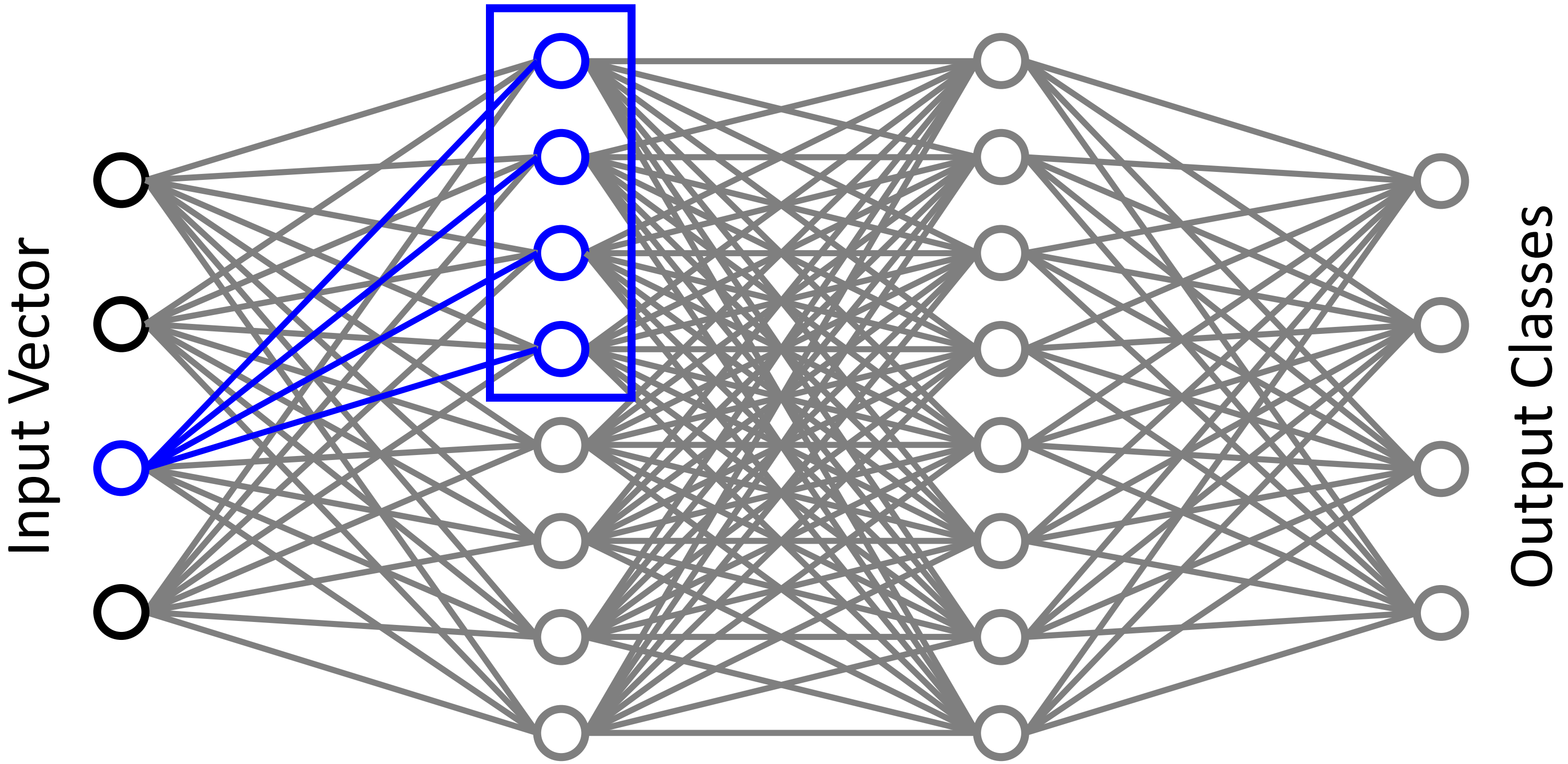


Re-use of activation data across neurons

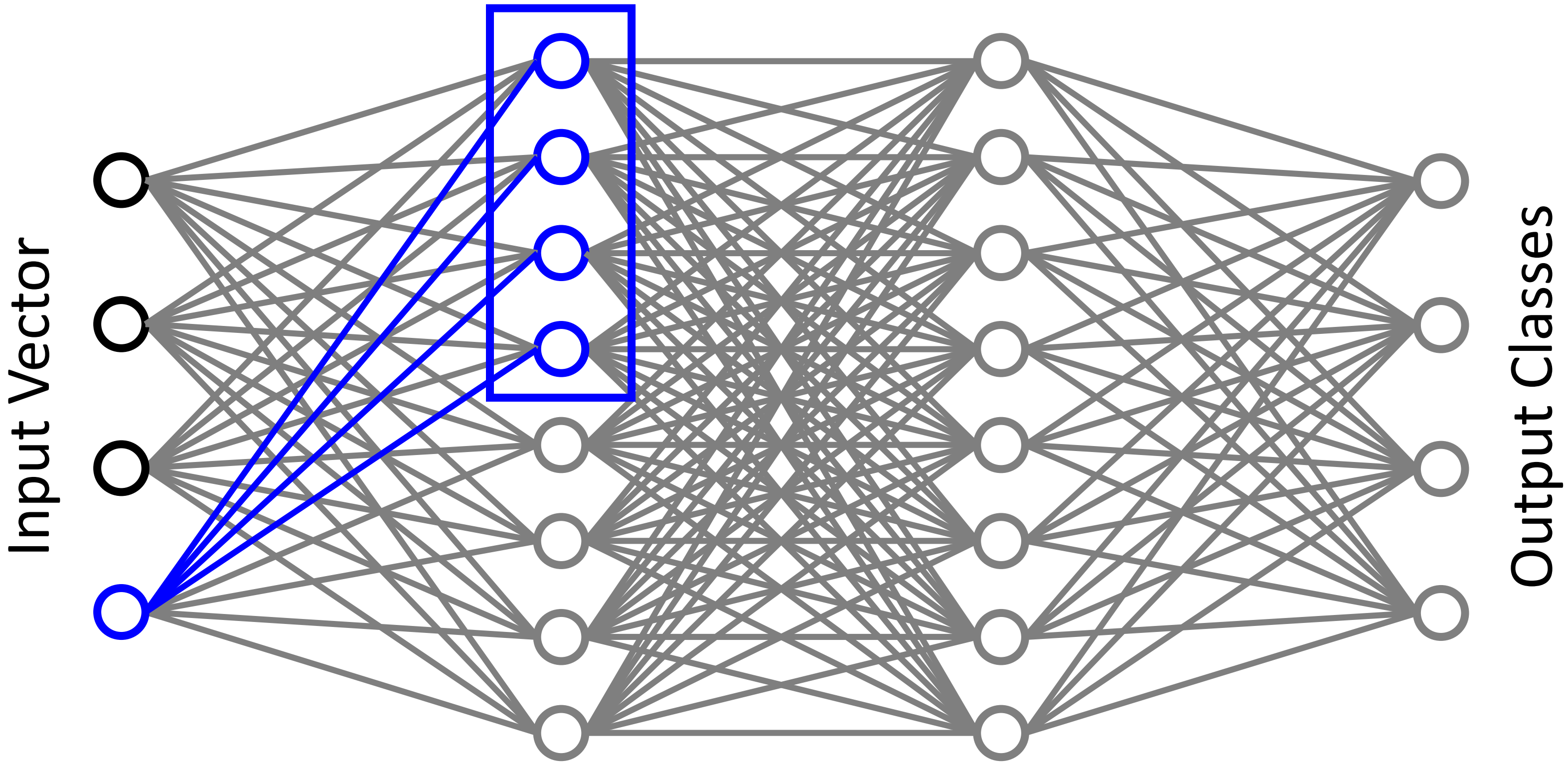
Fully-connected DNN graph



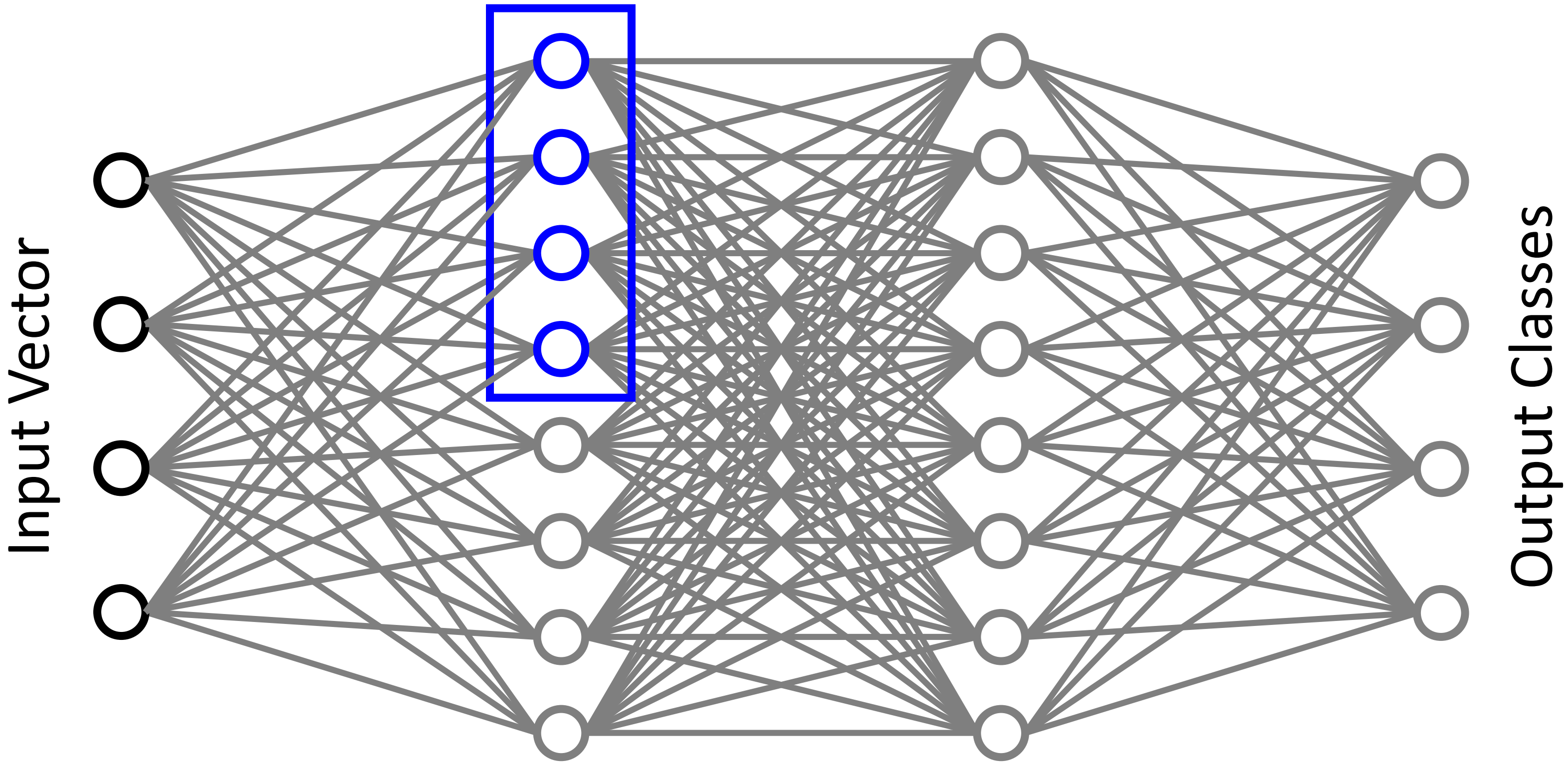
Fully-connected DNN graph



Fully-connected DNN graph

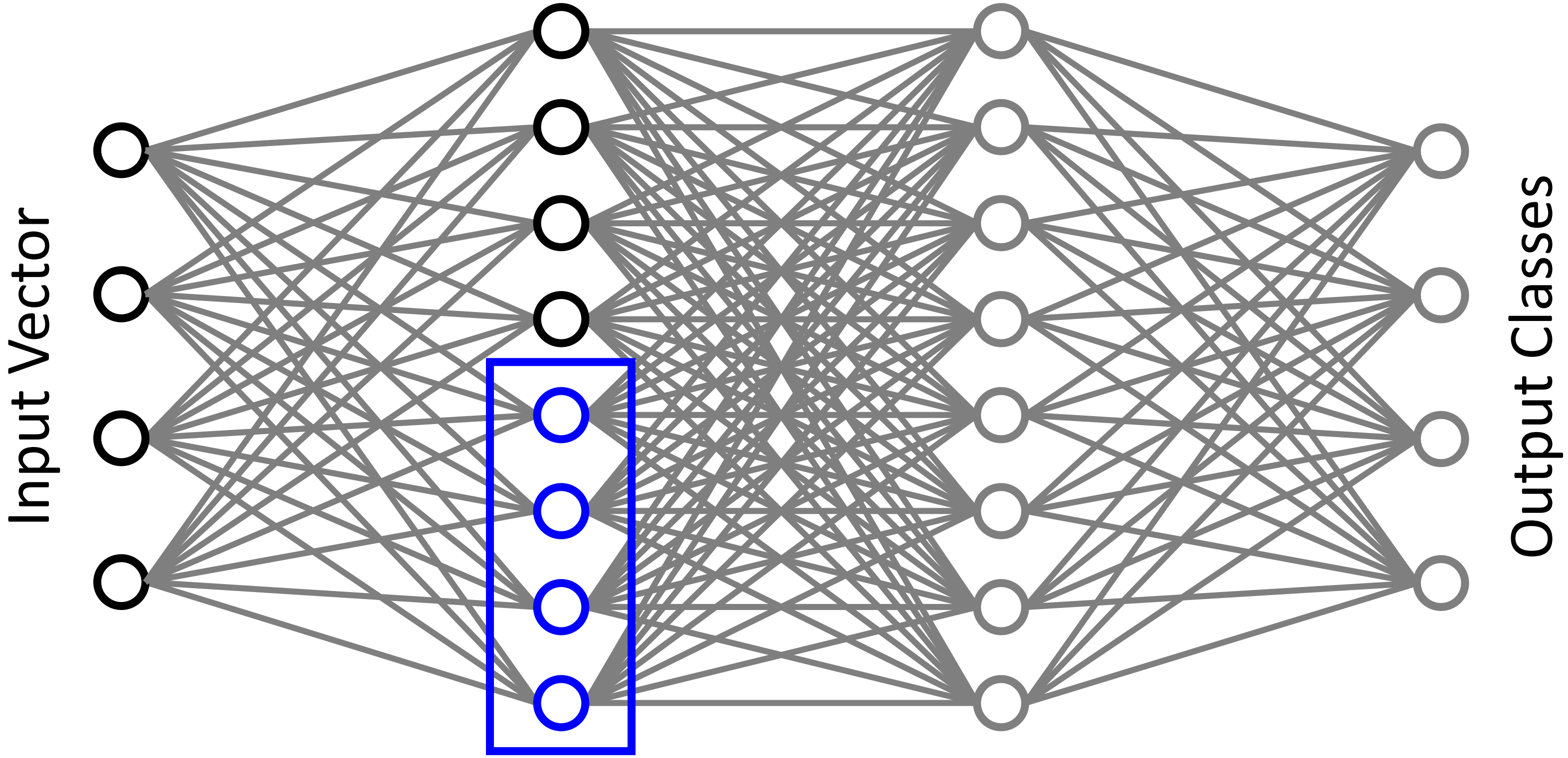


Fully-connected DNN graph

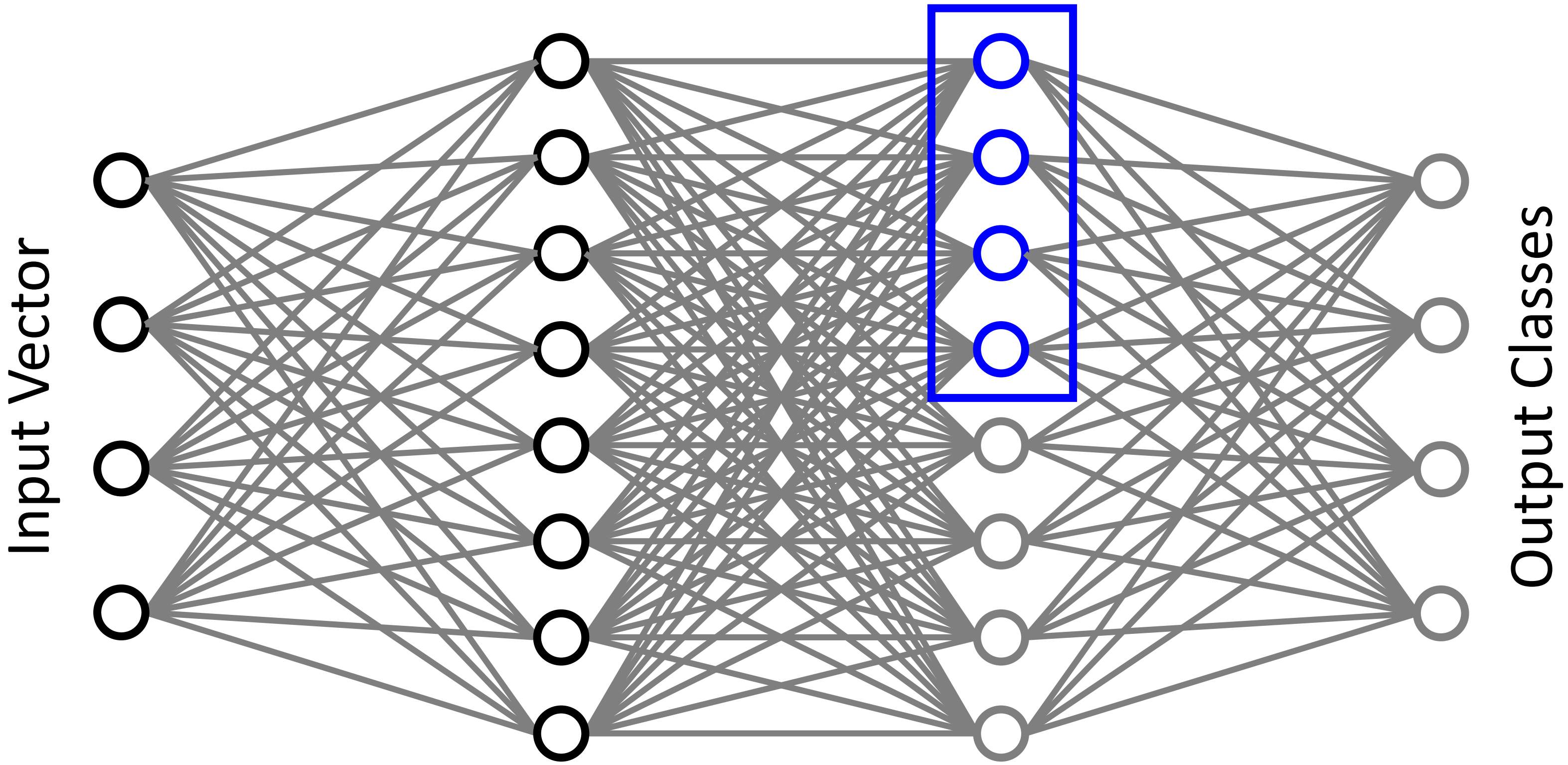


Add bias and apply activation function

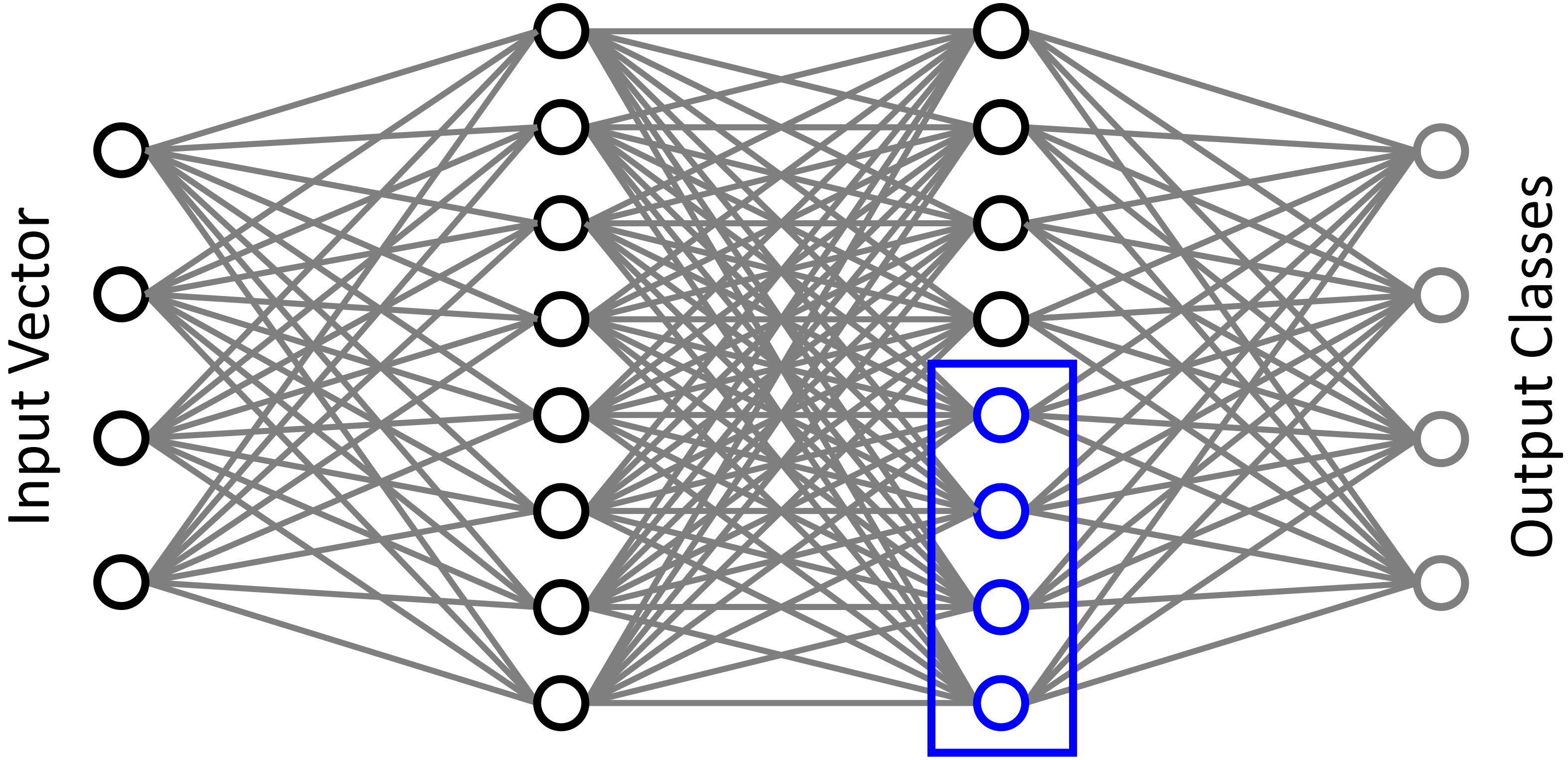
Fully-connected DNN graph



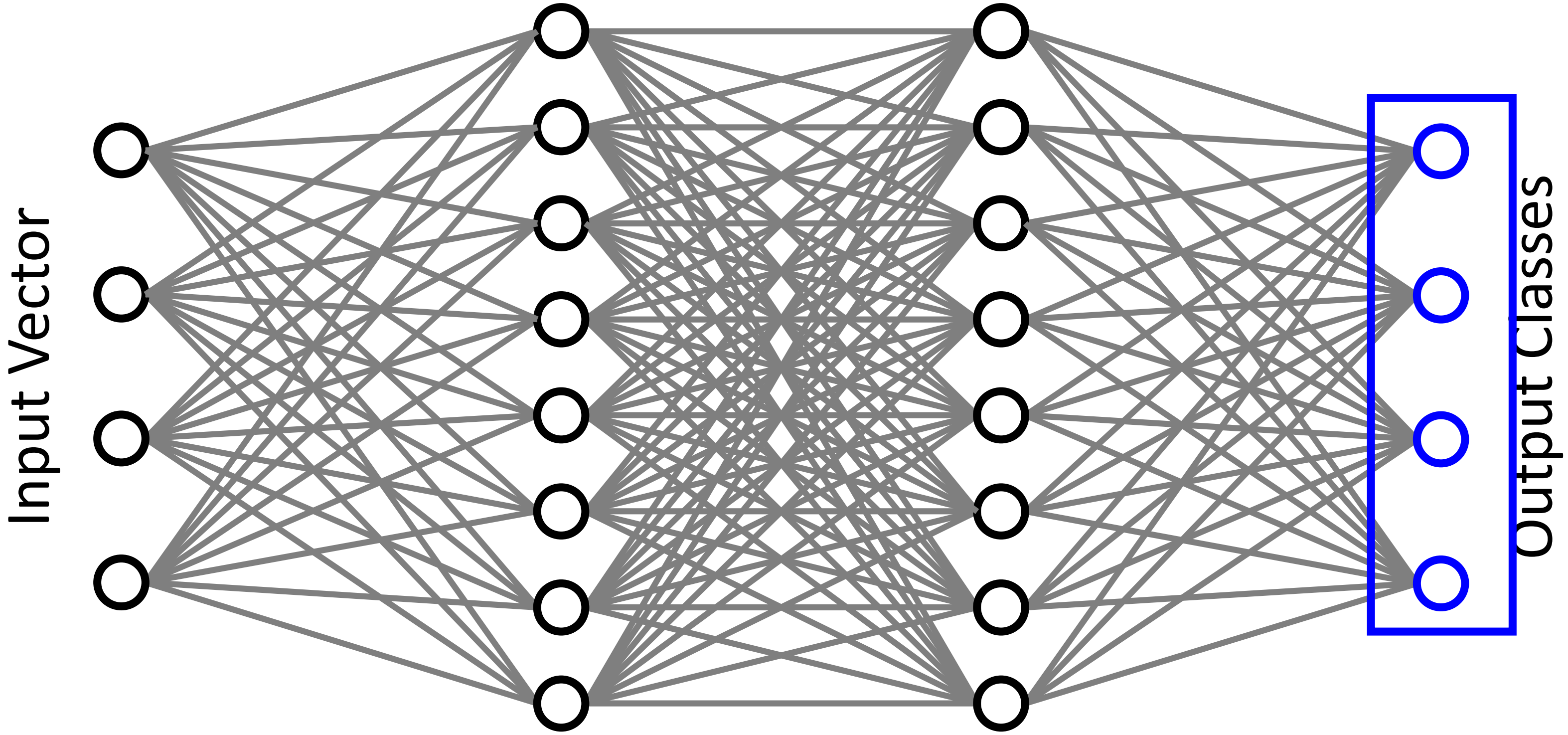
Fully-connected DNN graph



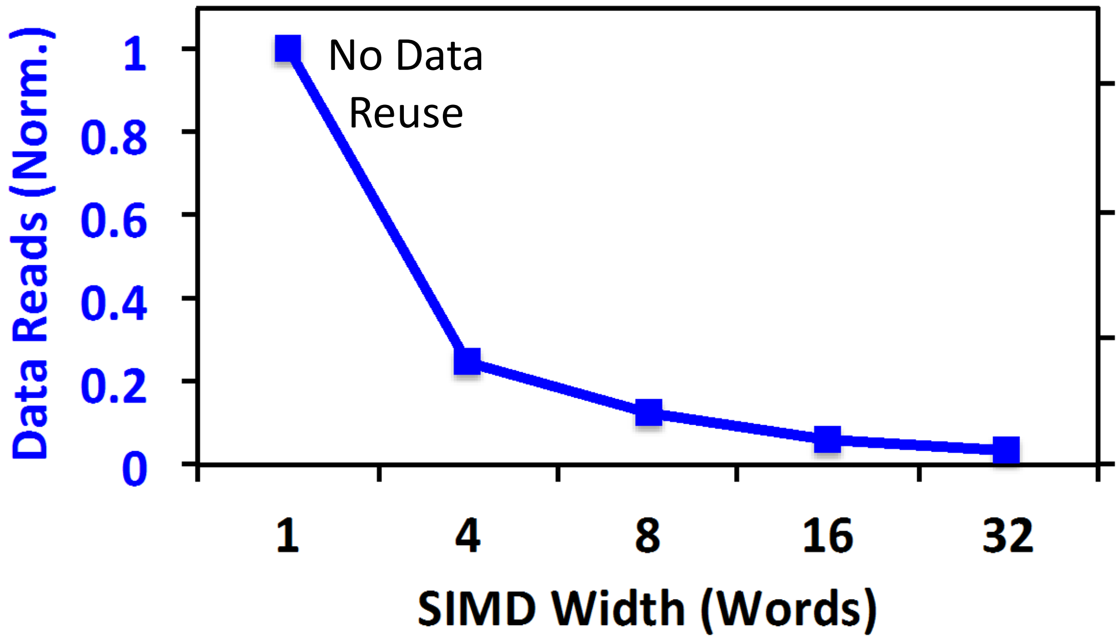
Fully-connected DNN graph



Fully-connected DNN graph

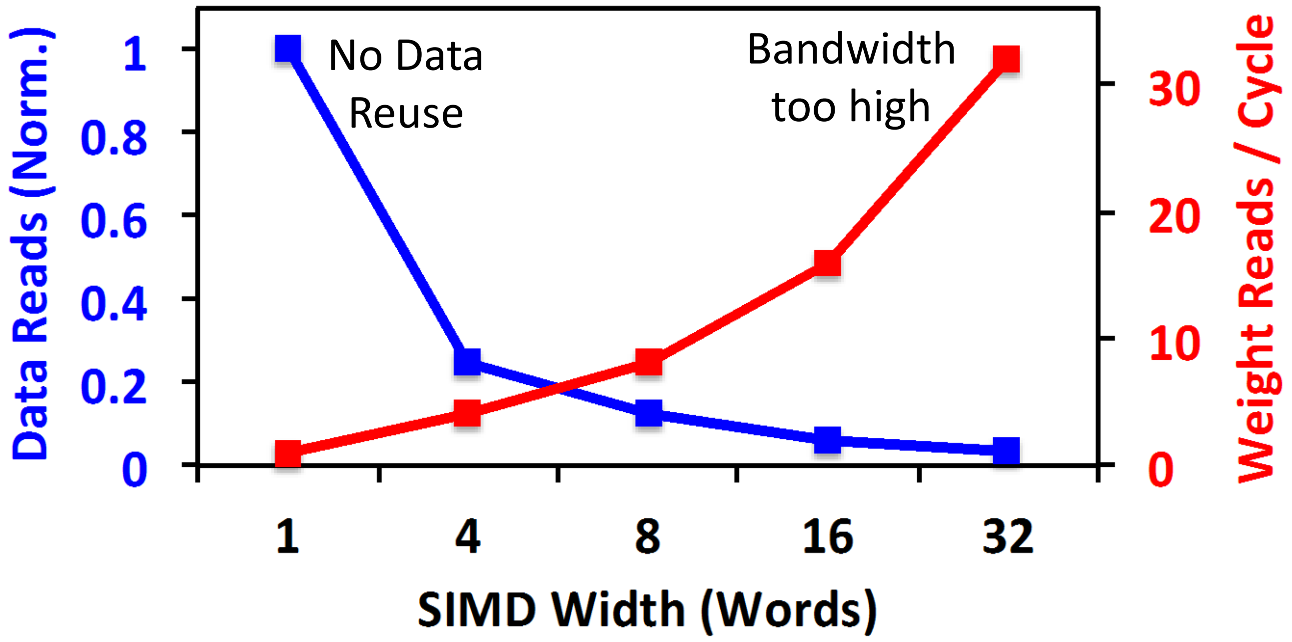


Balancing efficiency and bandwidth



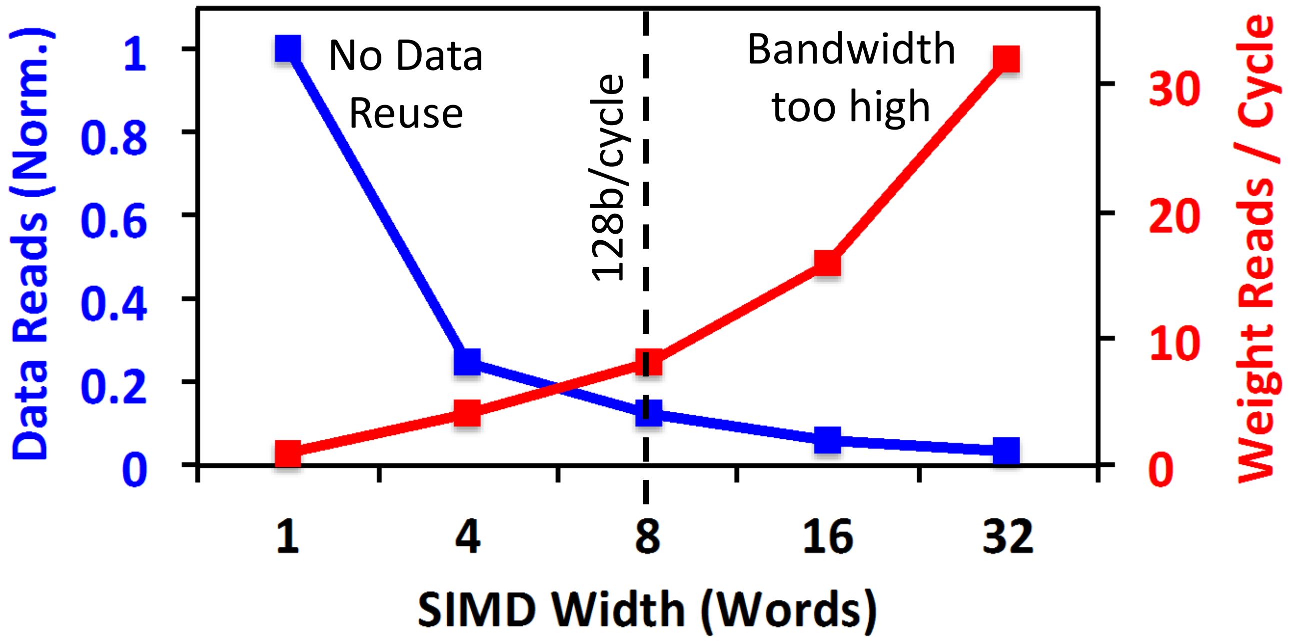
- Parallelism increases throughput and data reuse

Balancing efficiency and bandwidth



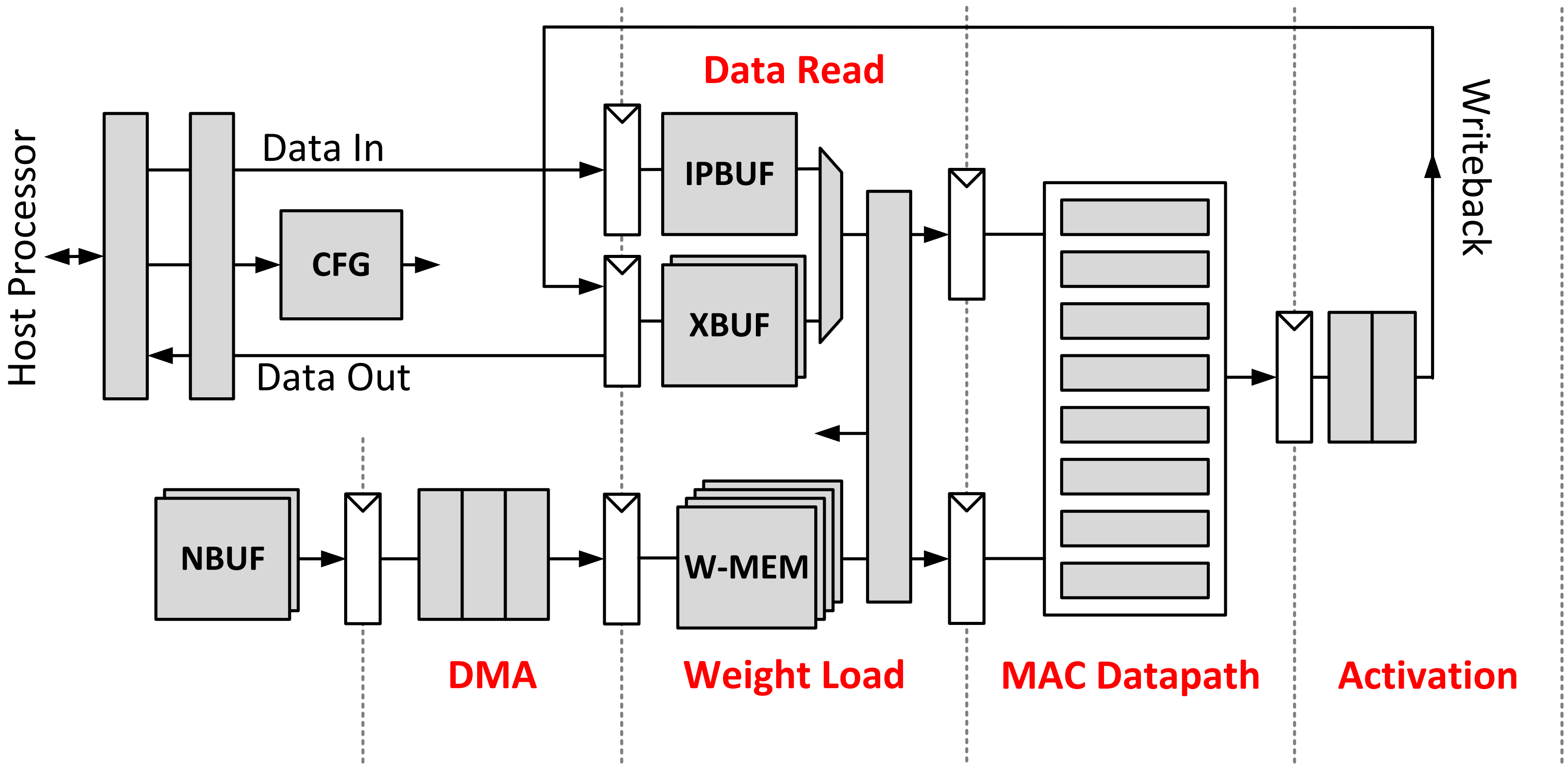
- Parallelism increases throughput and data reuse
  - But also increases Memory Bandwidth demands

Balancing efficiency and bandwidth



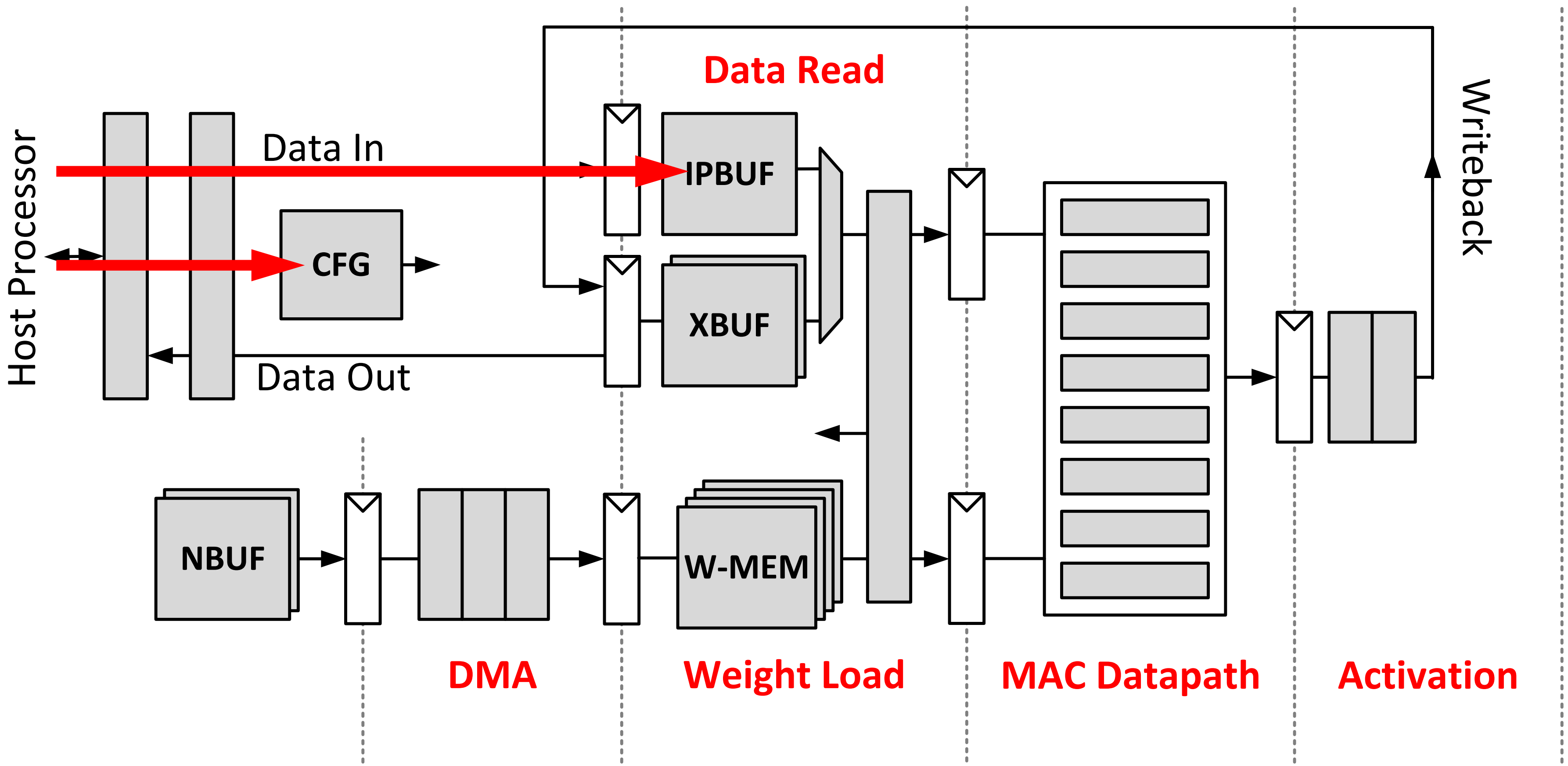
- Parallelism increases throughput and data reuse
  - But also increases Memory Bandwidth demands
- 8-Way SIMD is efficient at reasonable memory BW
  - 10x Activation reuse, with 128b AXI channel

DNN ENGINE micro-architecture



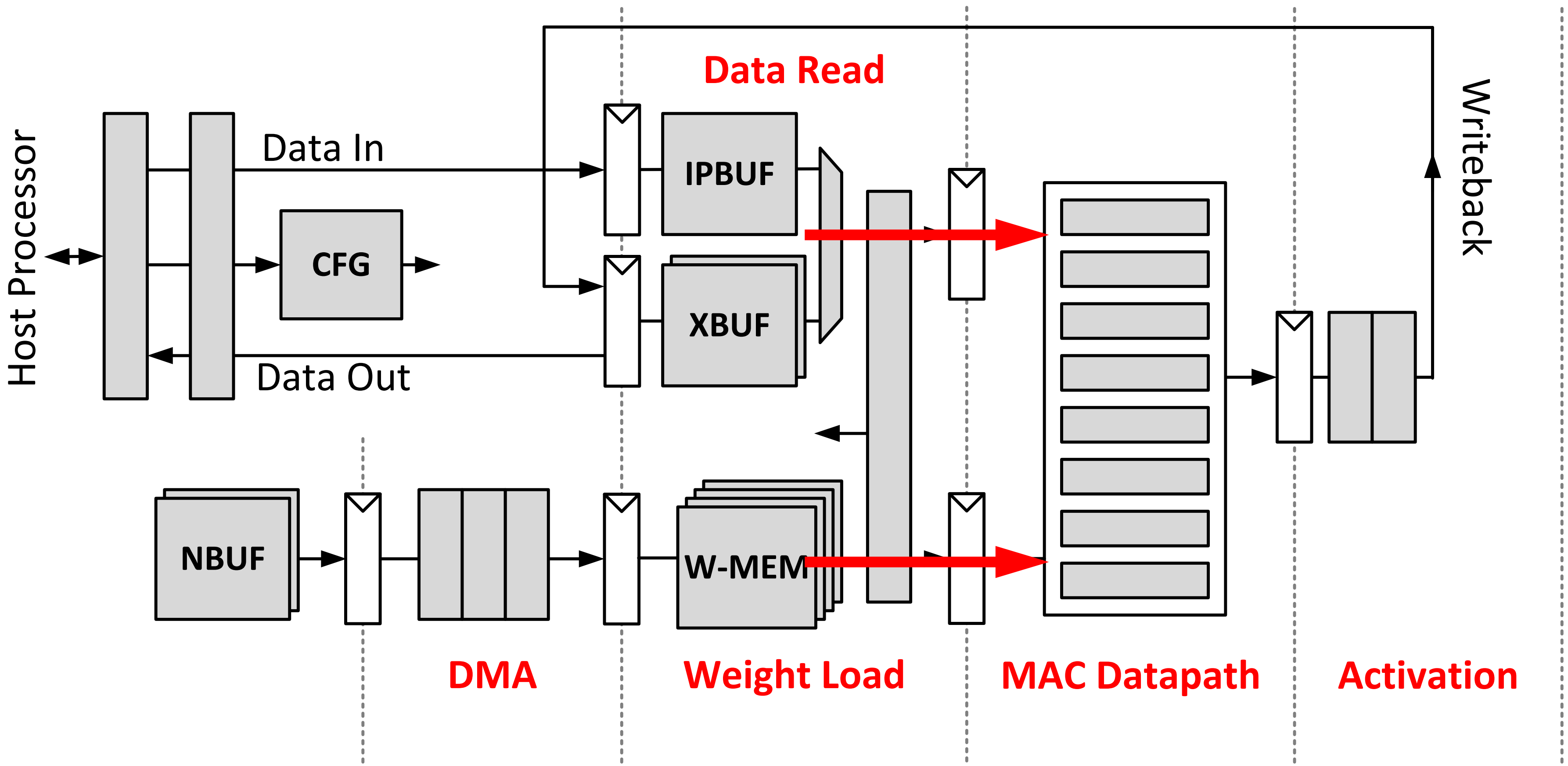
8-Way SIMD accelerator architecture

DNN ENGINE micro-architecture



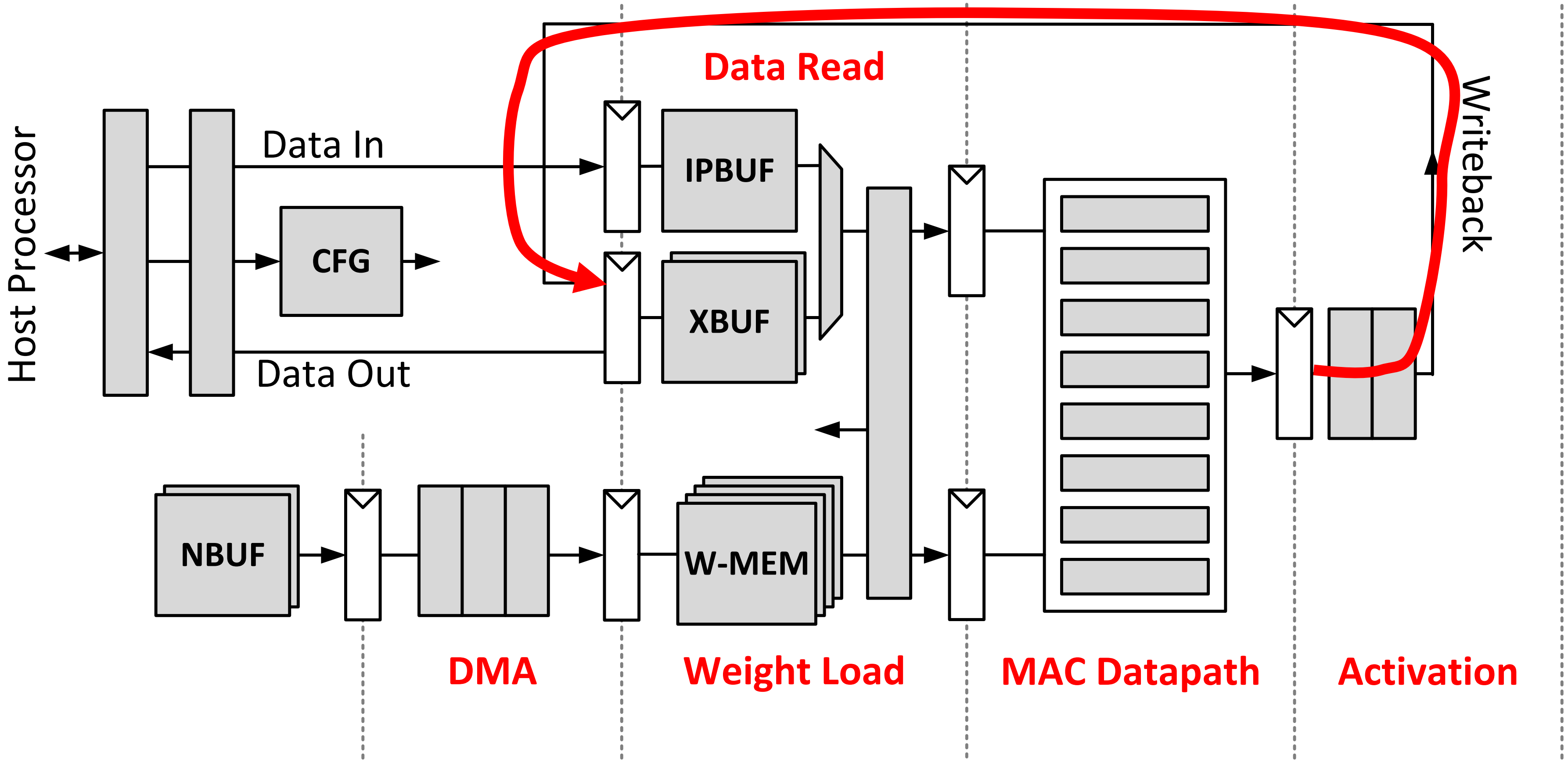
Host Processor loads configuration and input data

DNN ENGINE micro-architecture



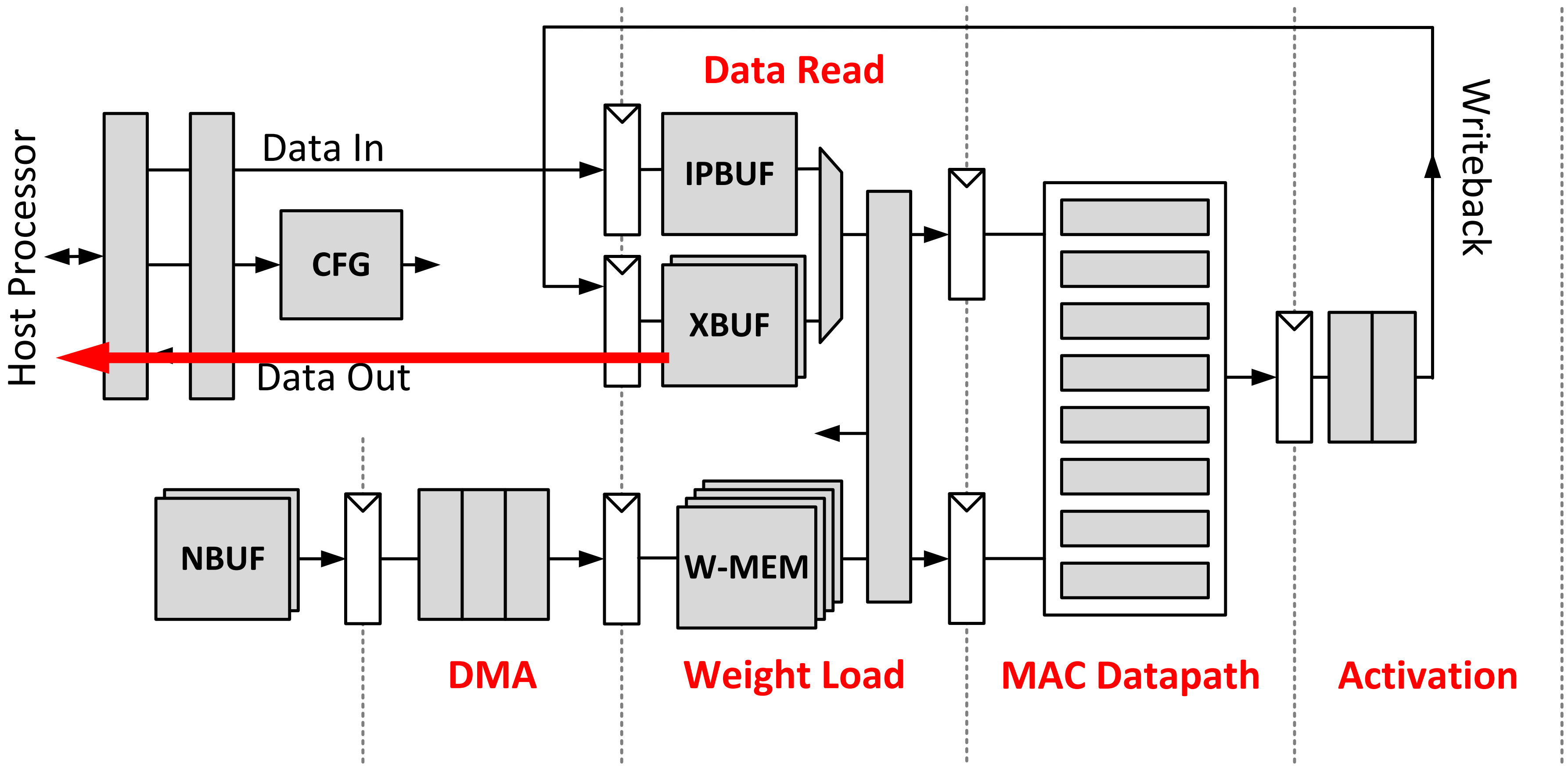
Accumulate products of Activation and Weights

DNN ENGINE micro-architecture



Add bias, apply ReLU activation and writeback

DNN ENGINE micro-architecture

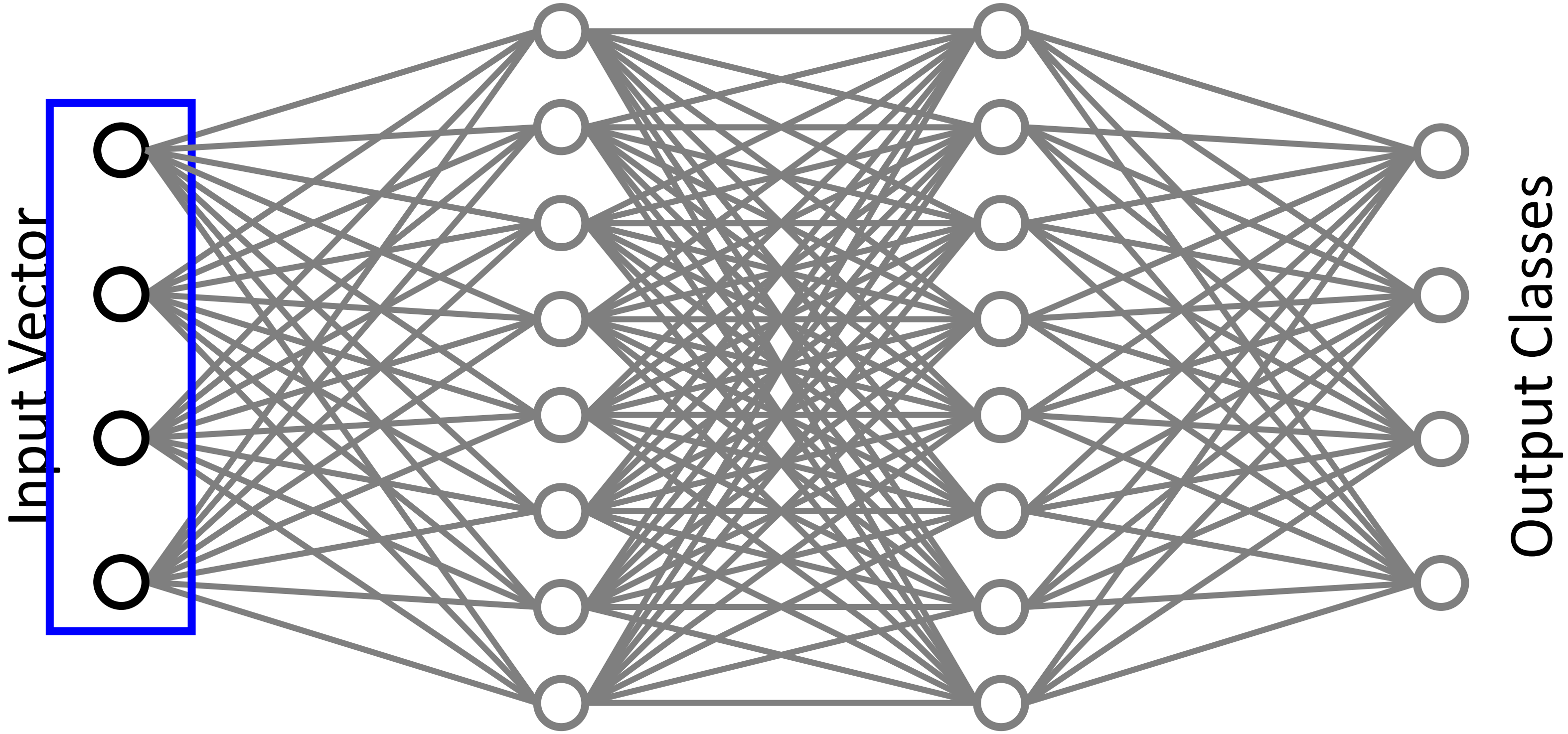


IRQ to host, which retrieves output data

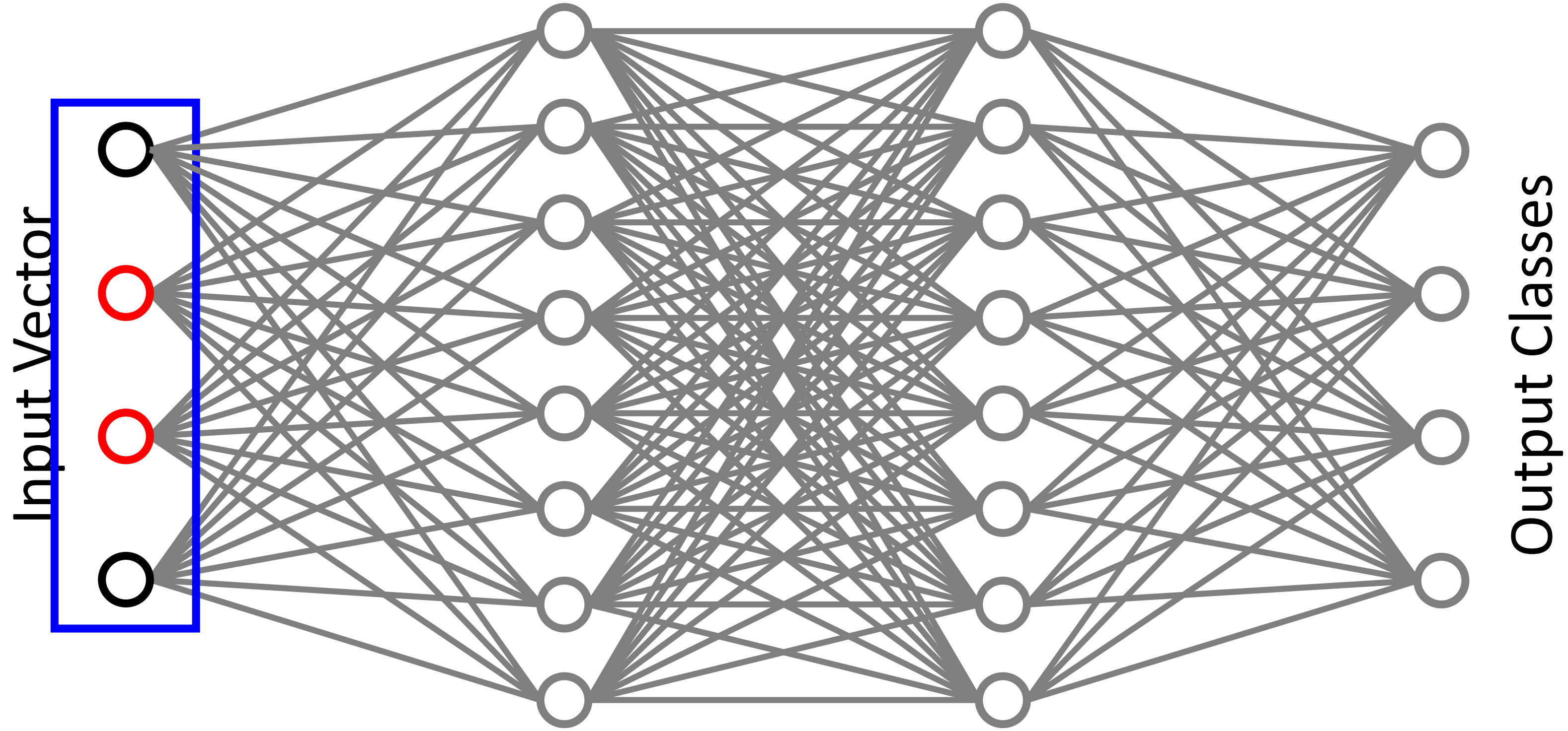
Outline

- Background and motivation
- DNN ENGINE
  - Parallelism and data reuse
  - Sparse data and small data types
  - Algorithmic resilience
- Measurement Results
- Summary

Exploiting sparse data

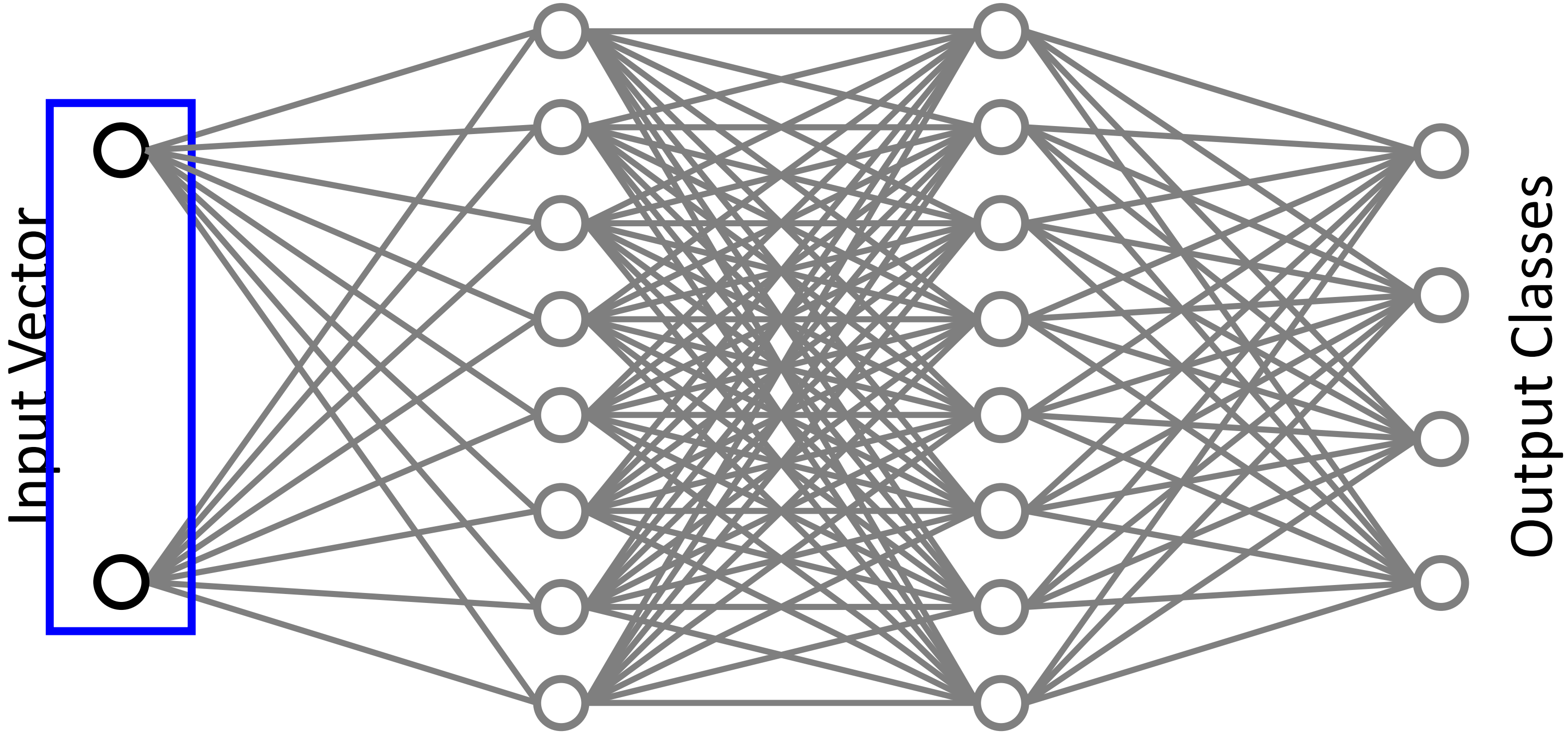


Exploiting sparse data



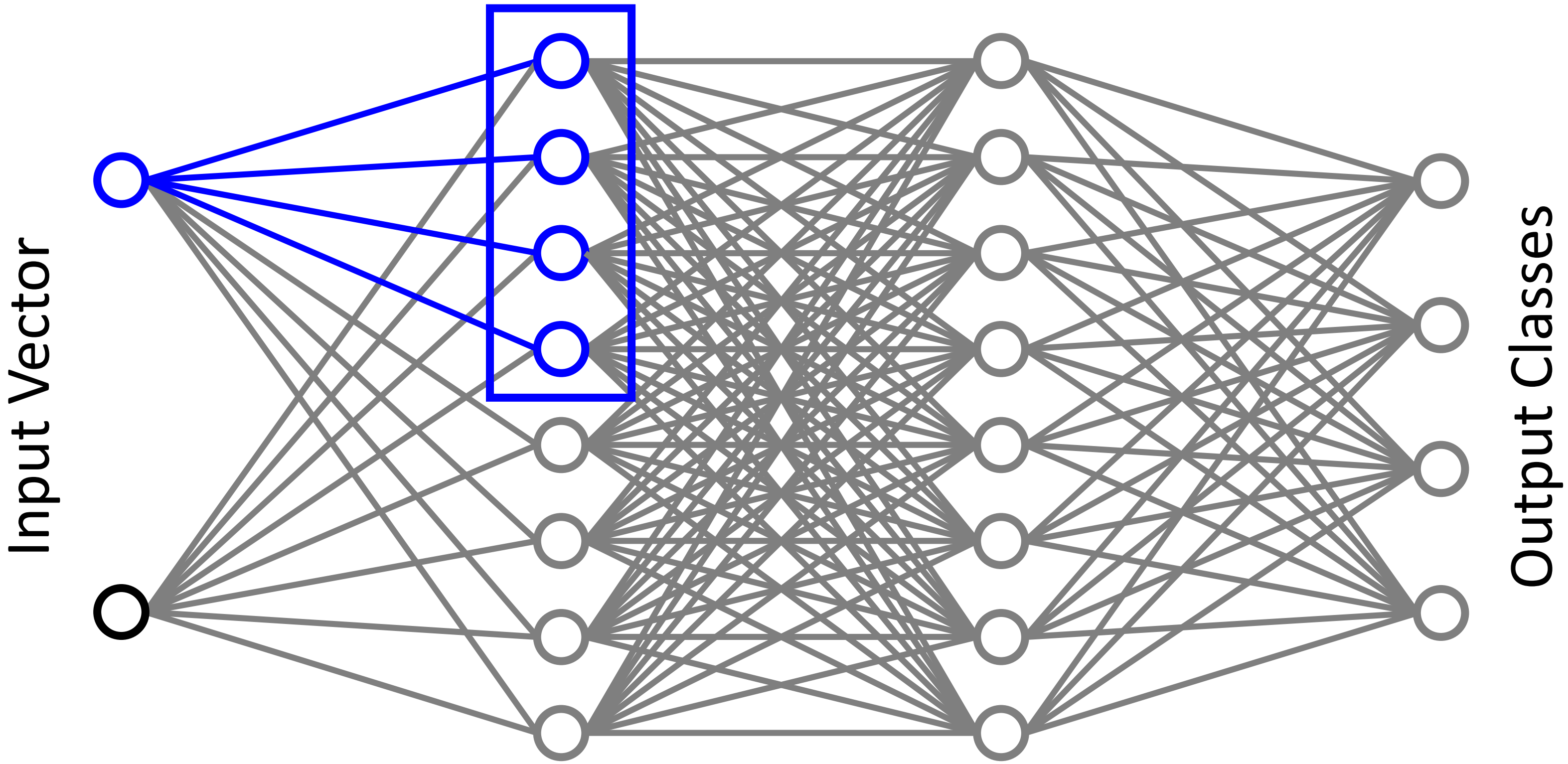
Discard small activation data

Exploiting sparse data

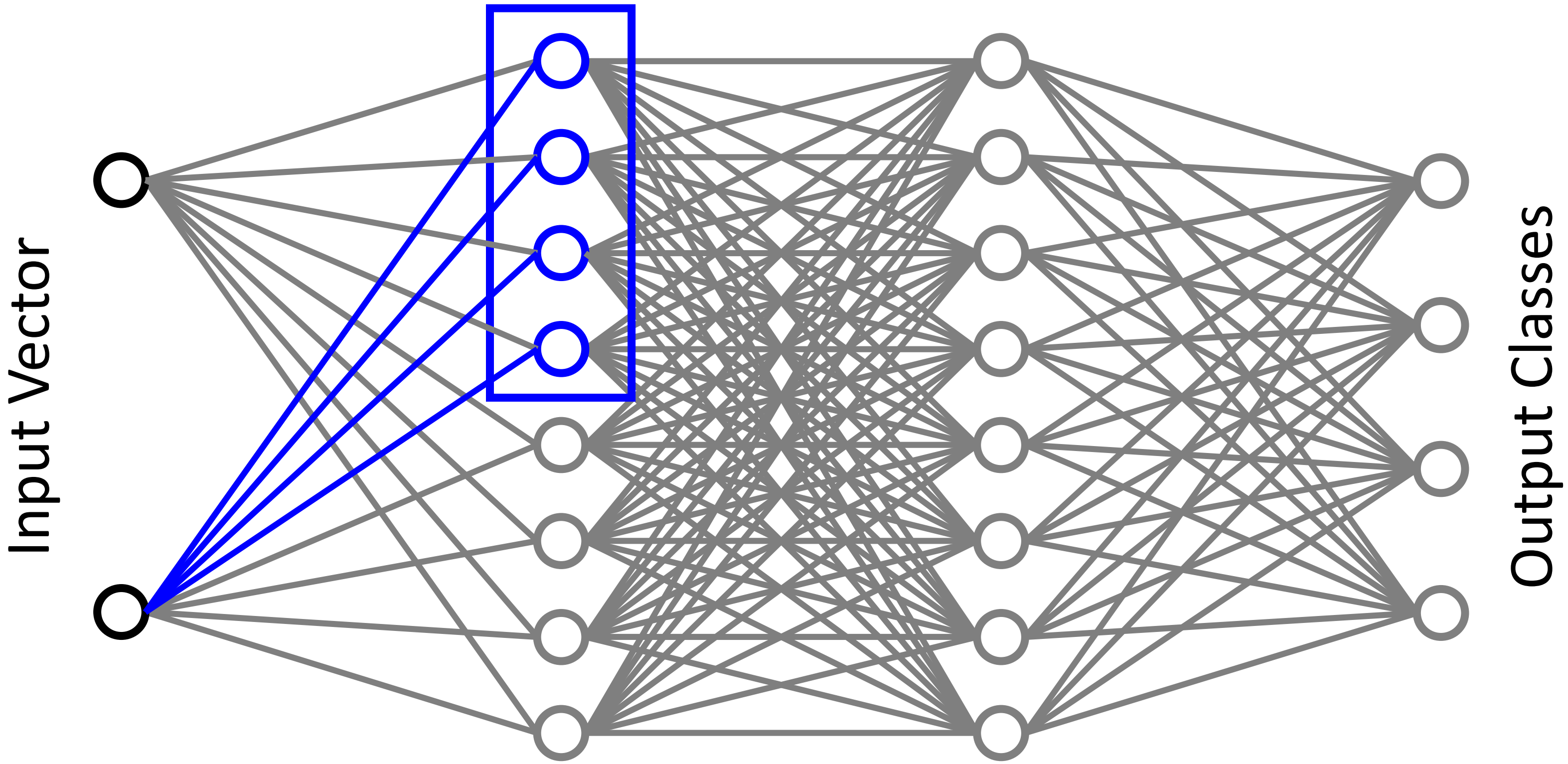


Dynamically prune graph connectivity

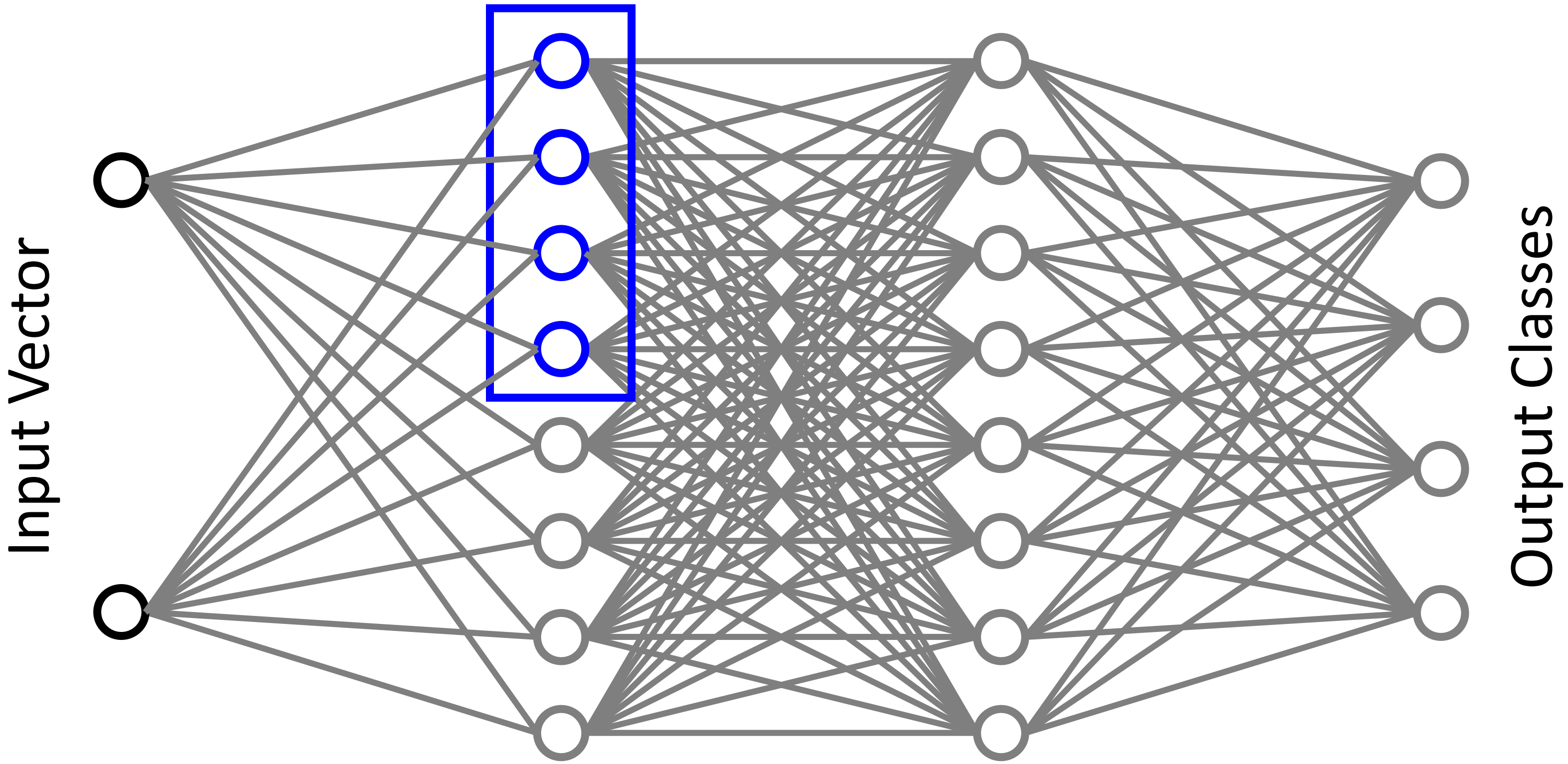
Exploiting sparse data



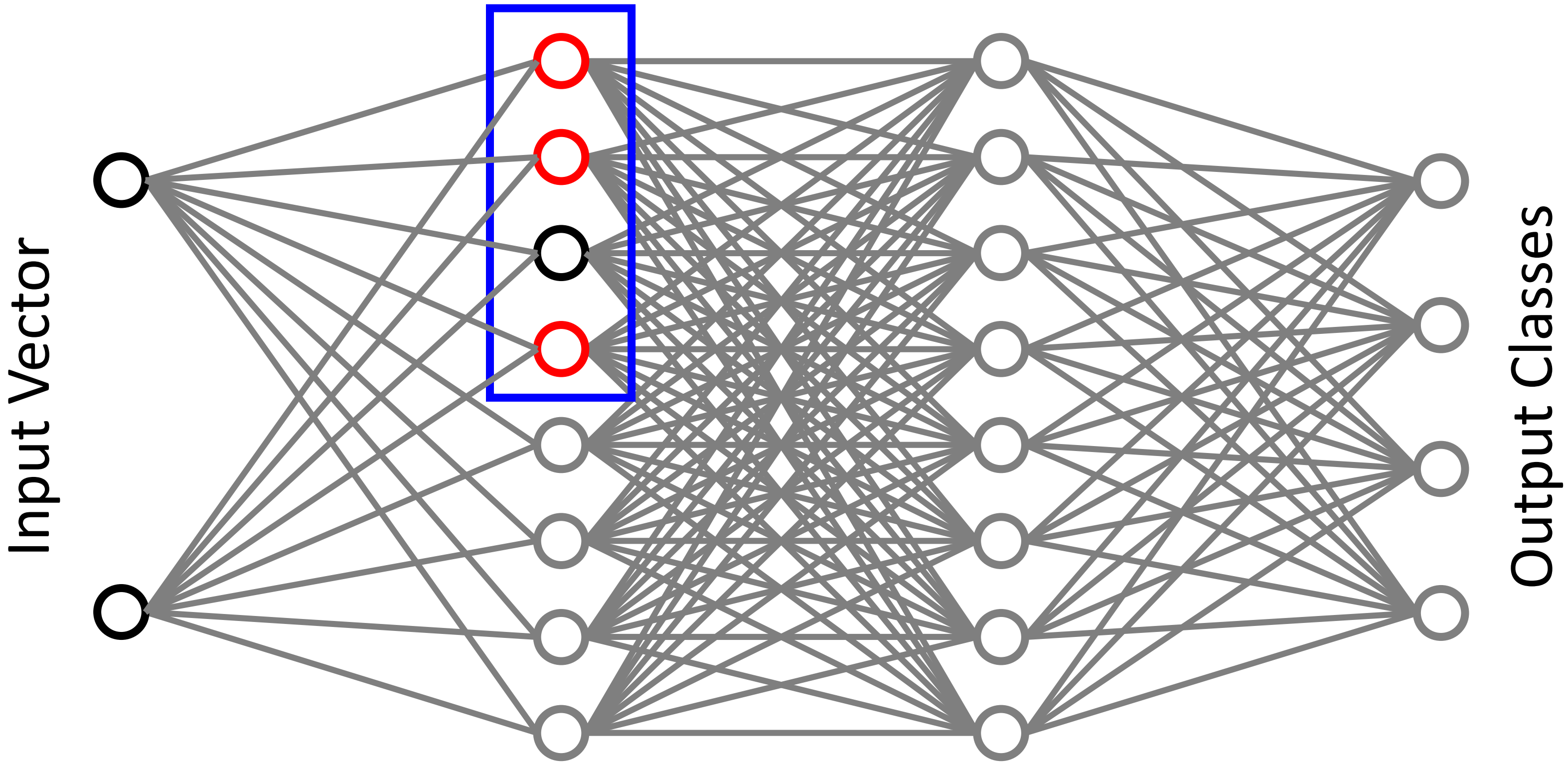
Exploiting sparse data



Exploiting sparse data

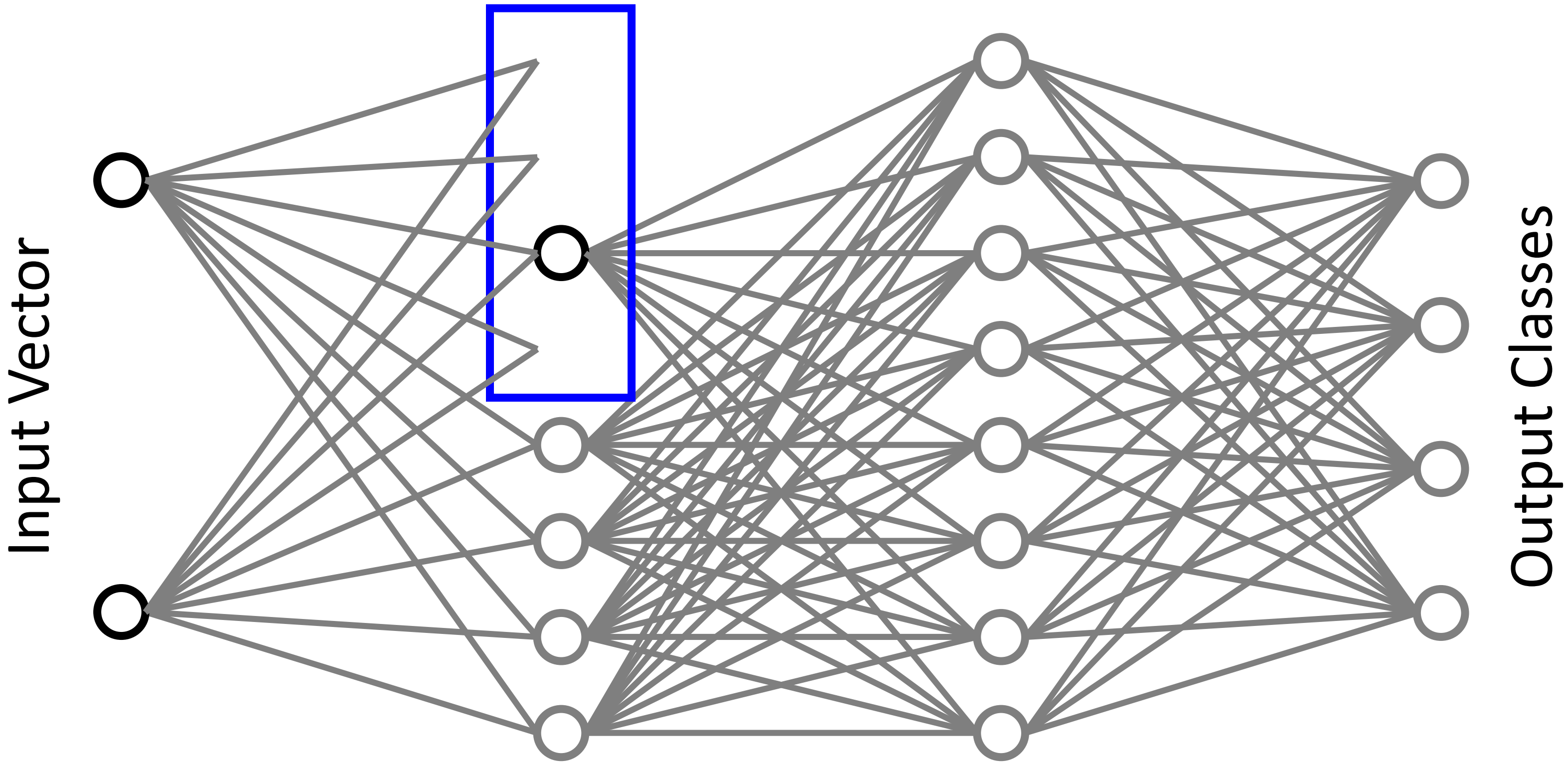


Exploiting sparse data



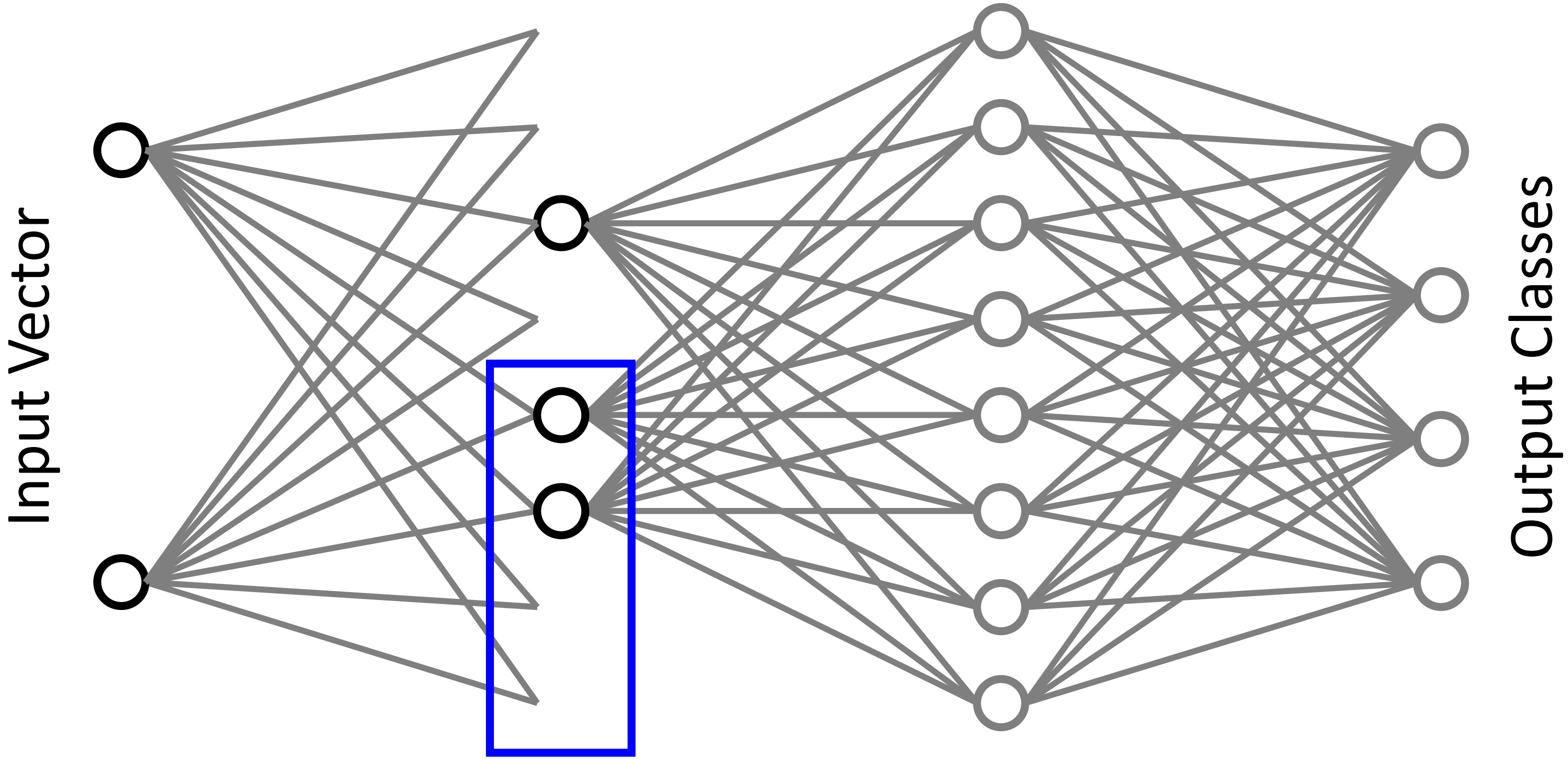
Discard small activation data

Exploiting sparse data

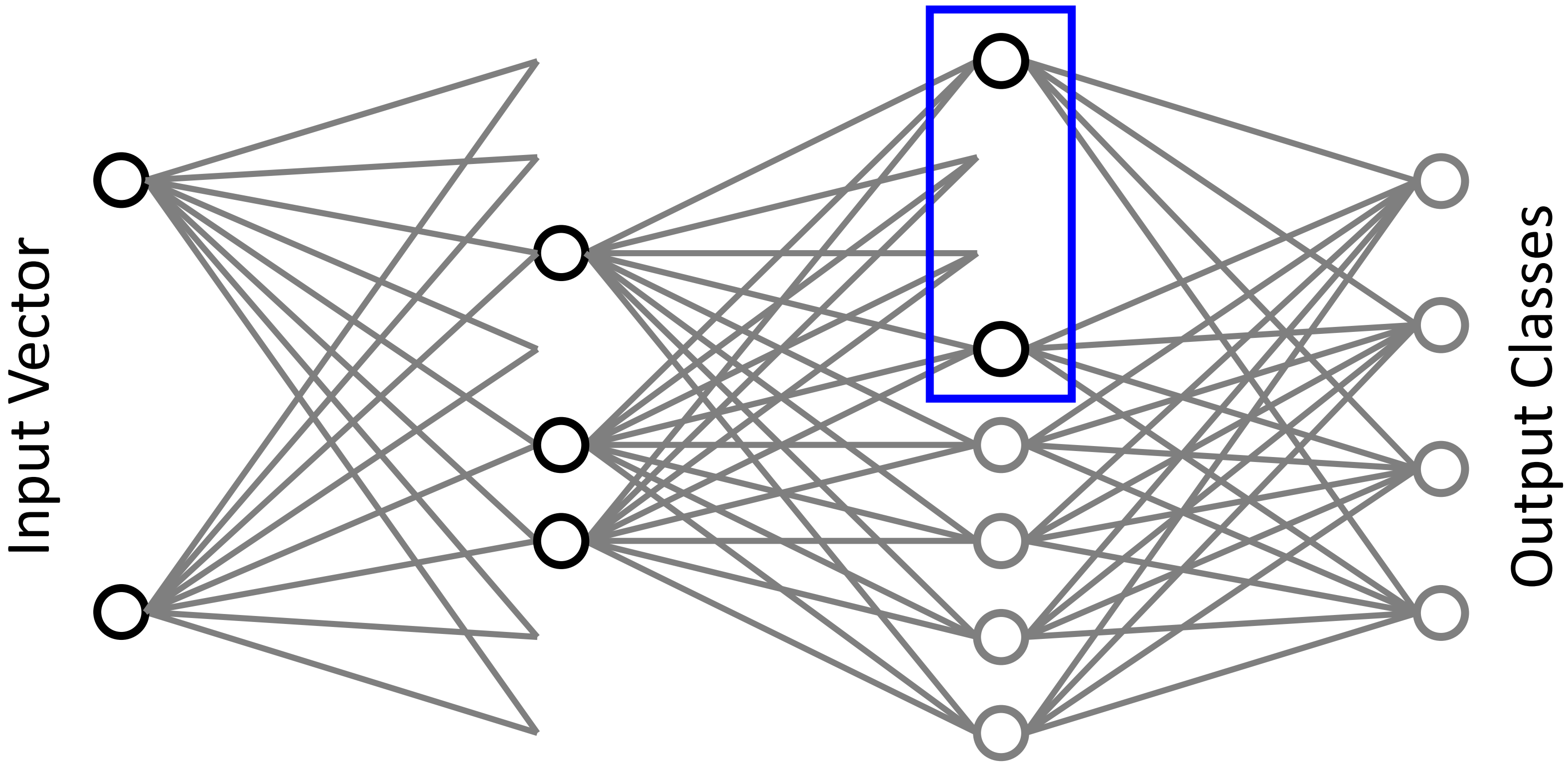


Dynamically prune graph connectivity

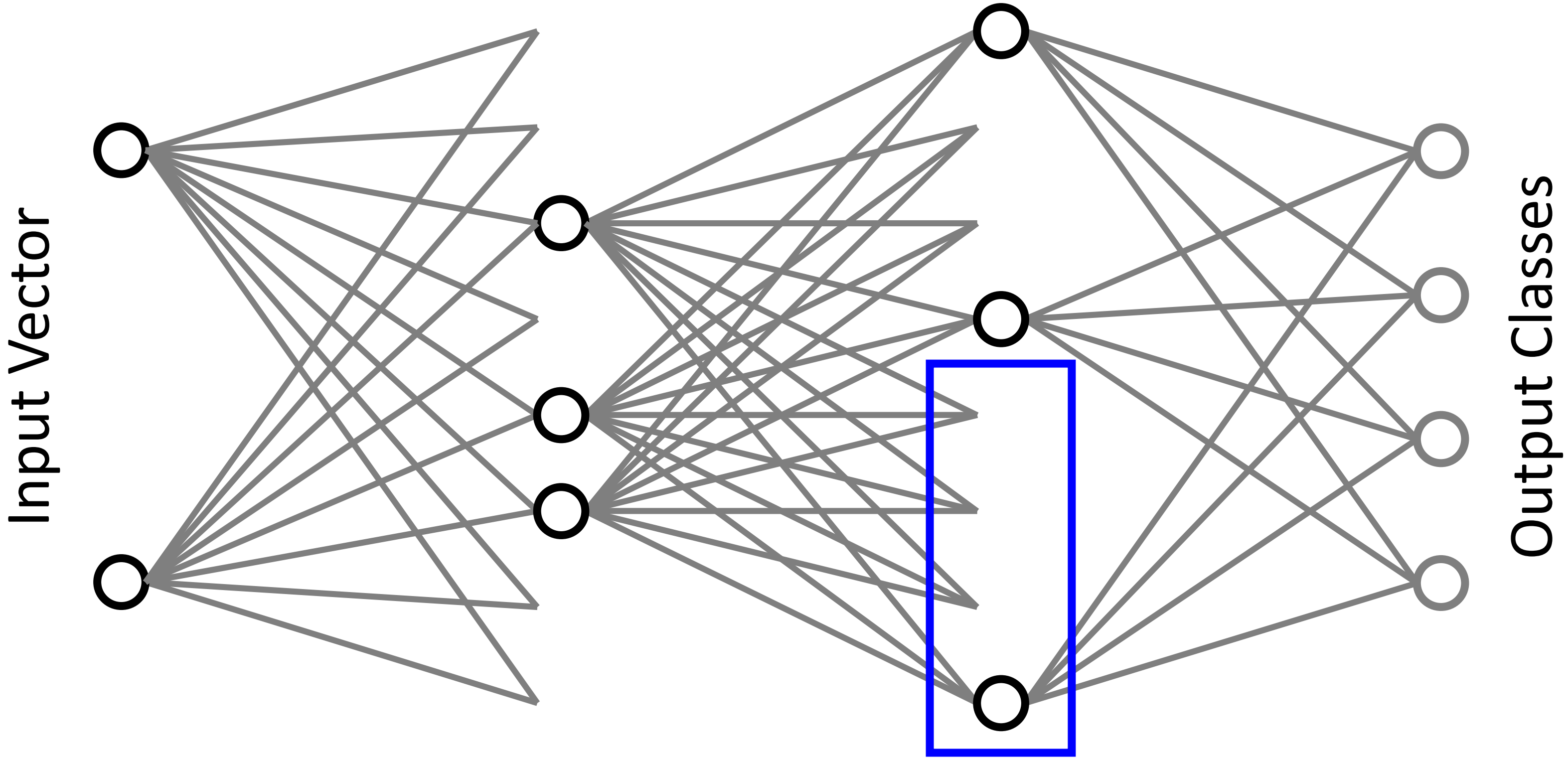
Exploiting sparse data



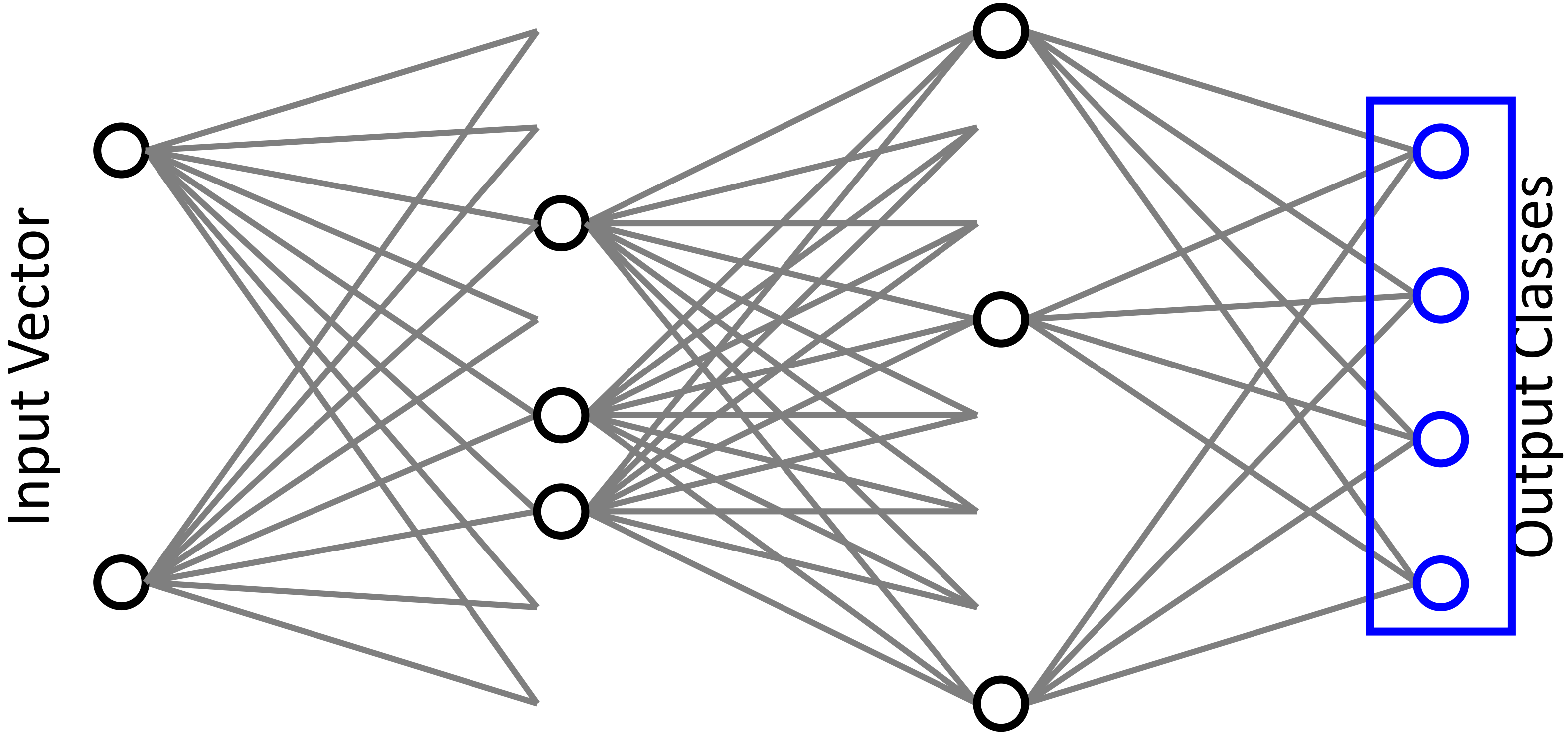
Exploiting sparse data



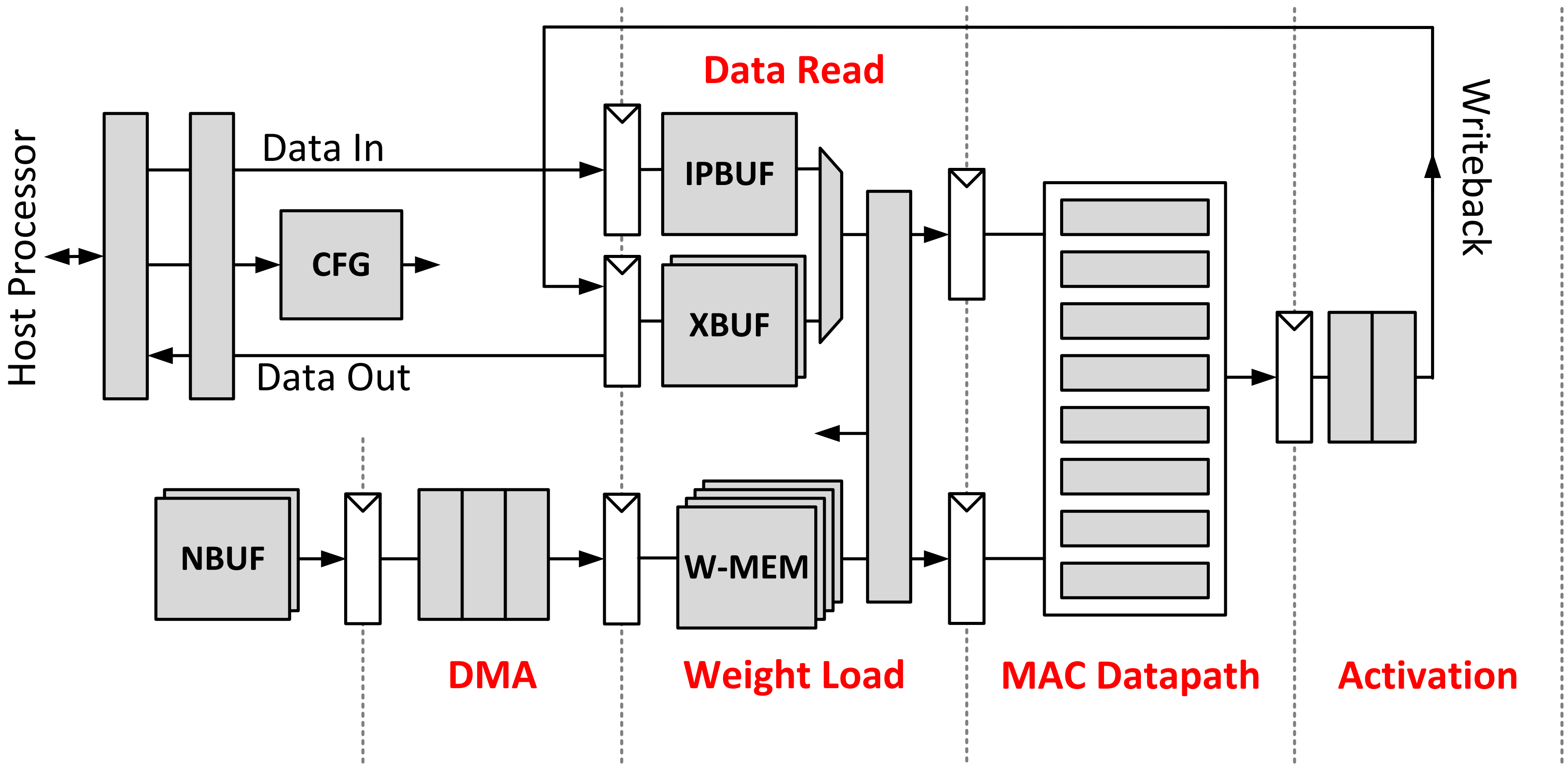
Exploiting sparse data



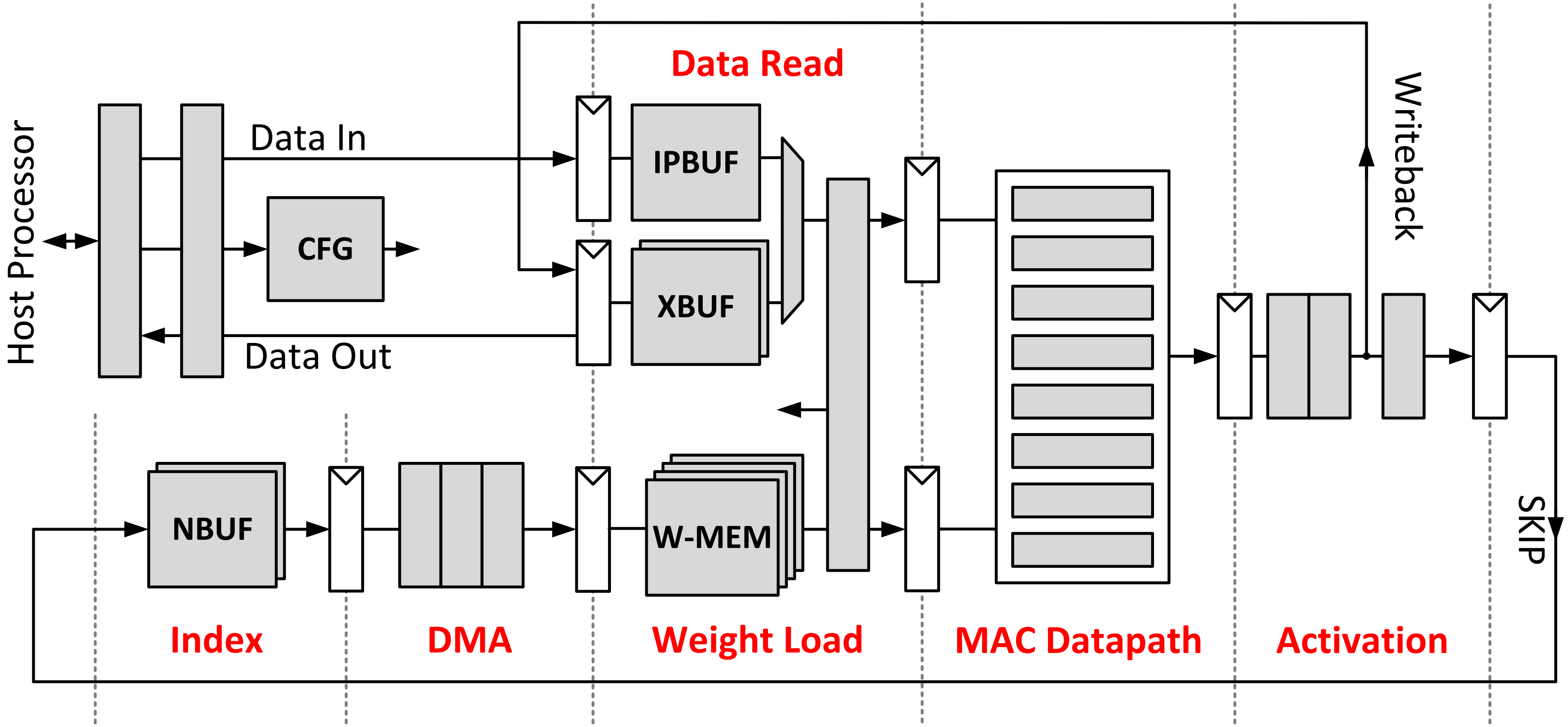
Exploiting sparse data



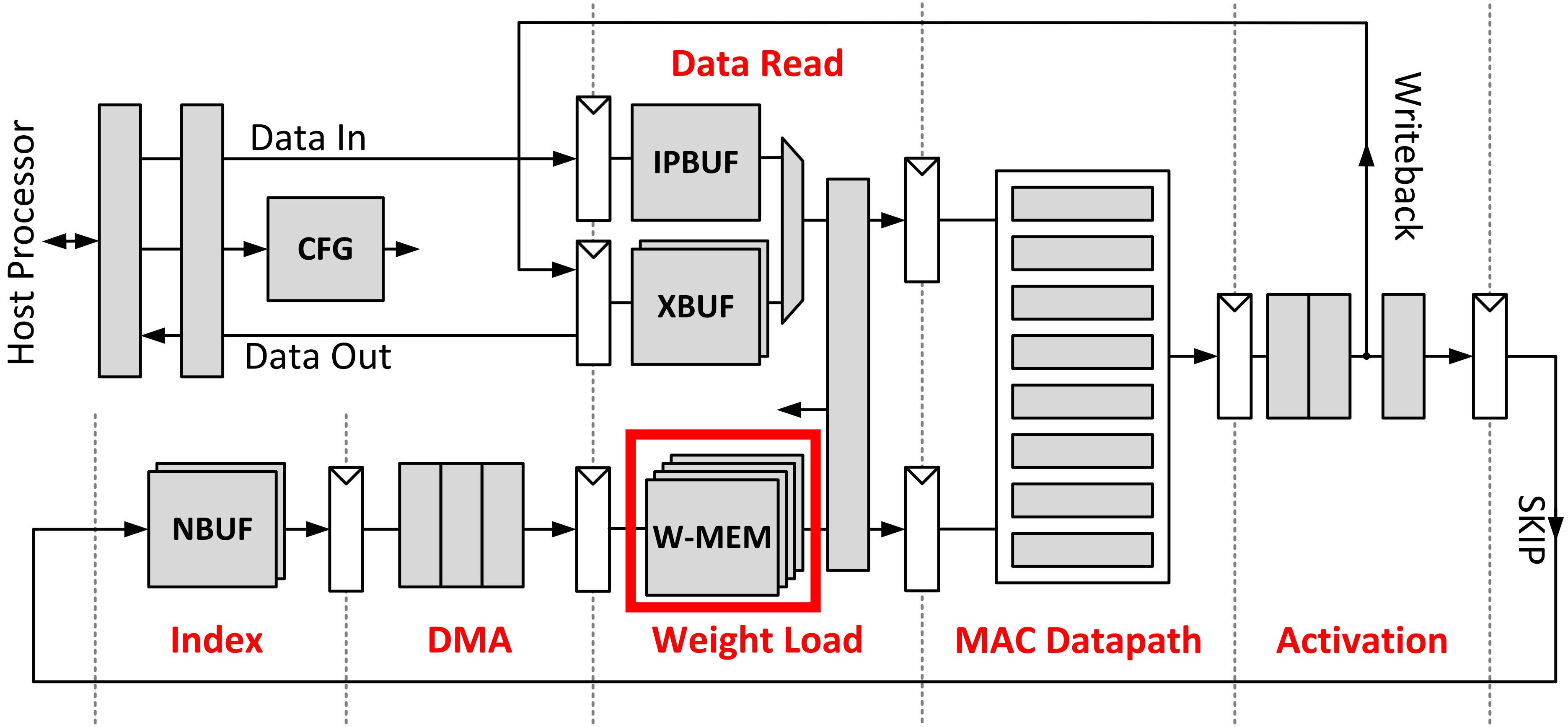
DNN ENGINE micro-architecture



DNN ENGINE micro-architecture



DNN ENGINE micro-architecture

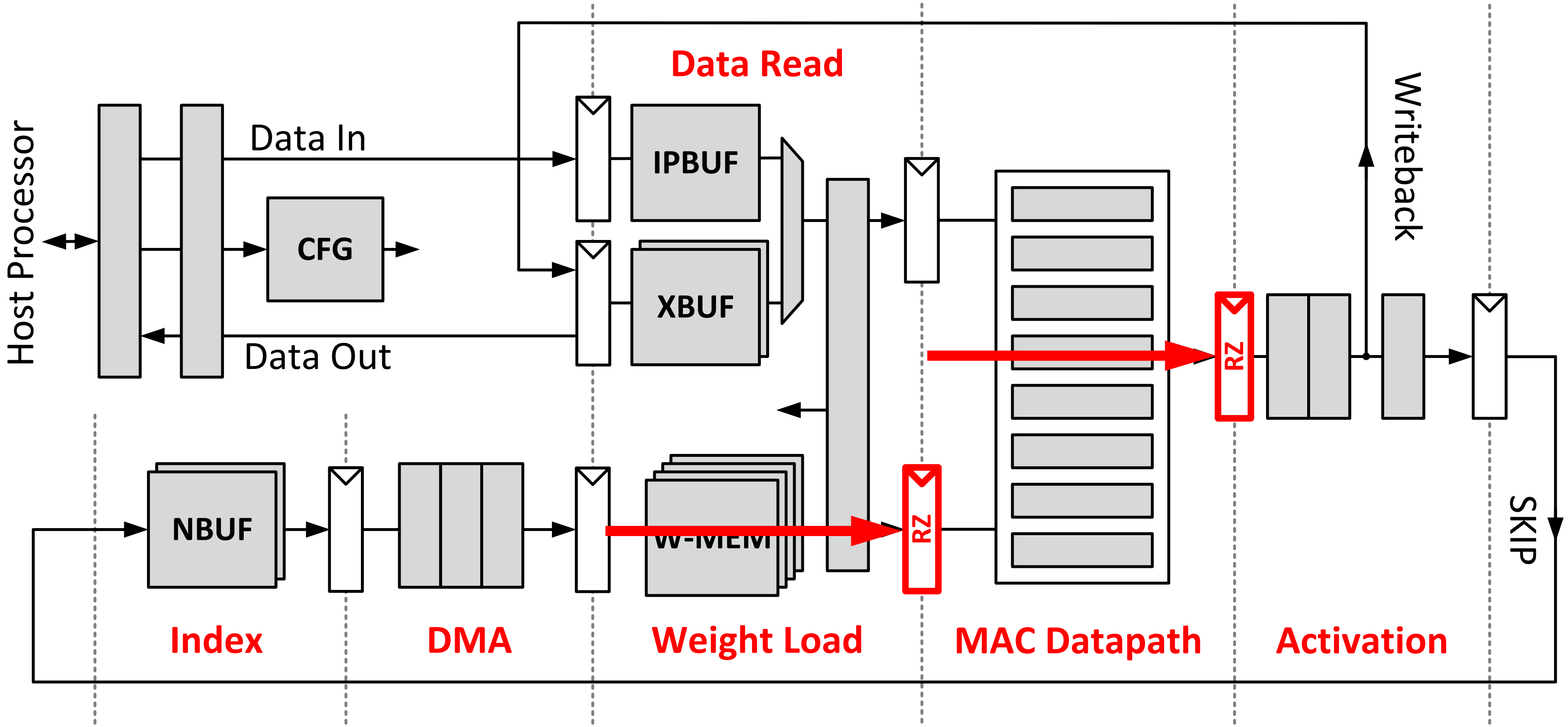


W-MEM supports 8b or 16b data types

Outline

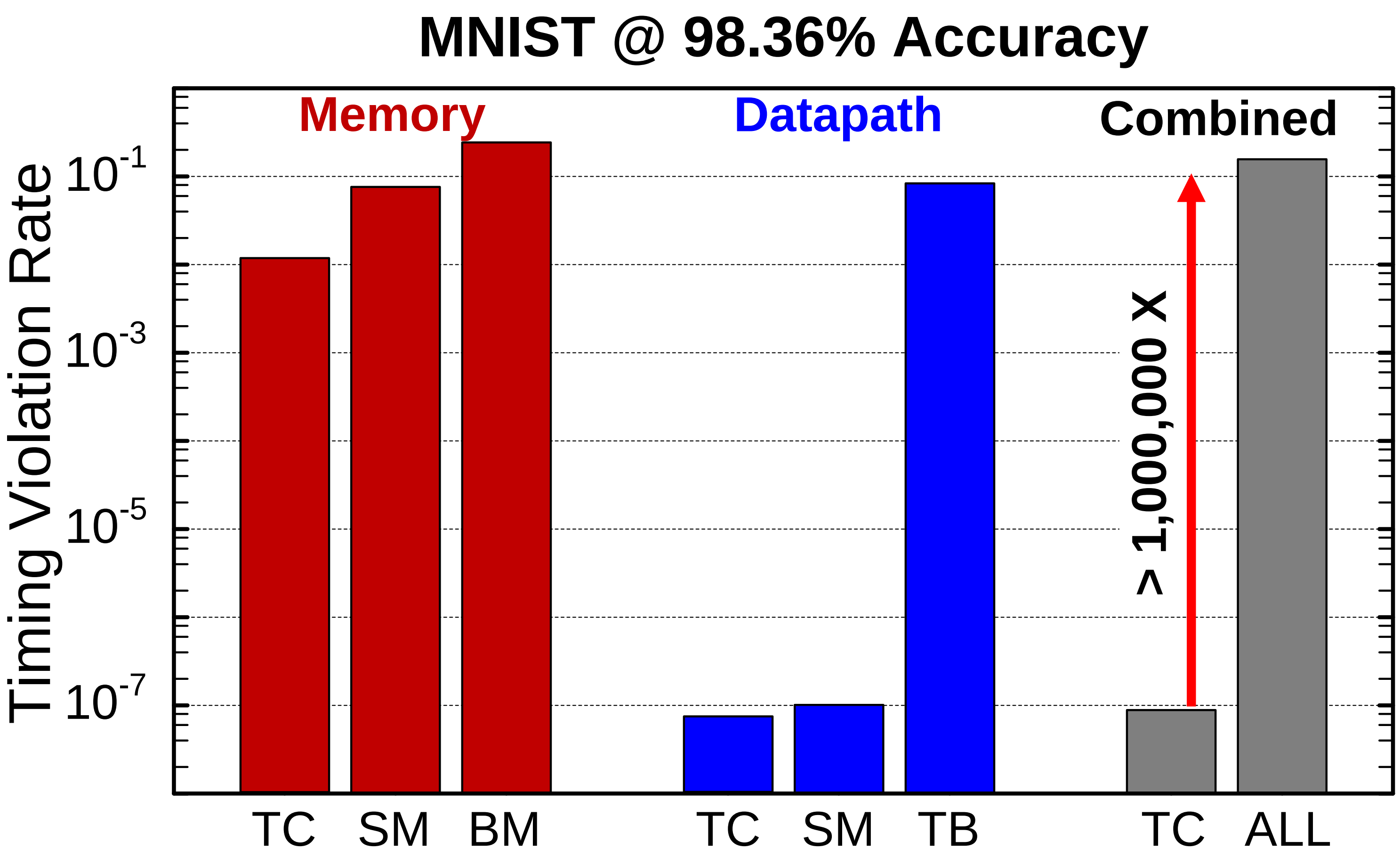
- Background and motivation
- DNN ENGINE
  - Parallelism and data reuse
  - Sparse data and small data types
  - Algorithmic resilience
- Measurement Results
- Summary

DNN ENGINE micro-architecture



[Whatmough et al., ISSCC'17]

Timing error tolerance

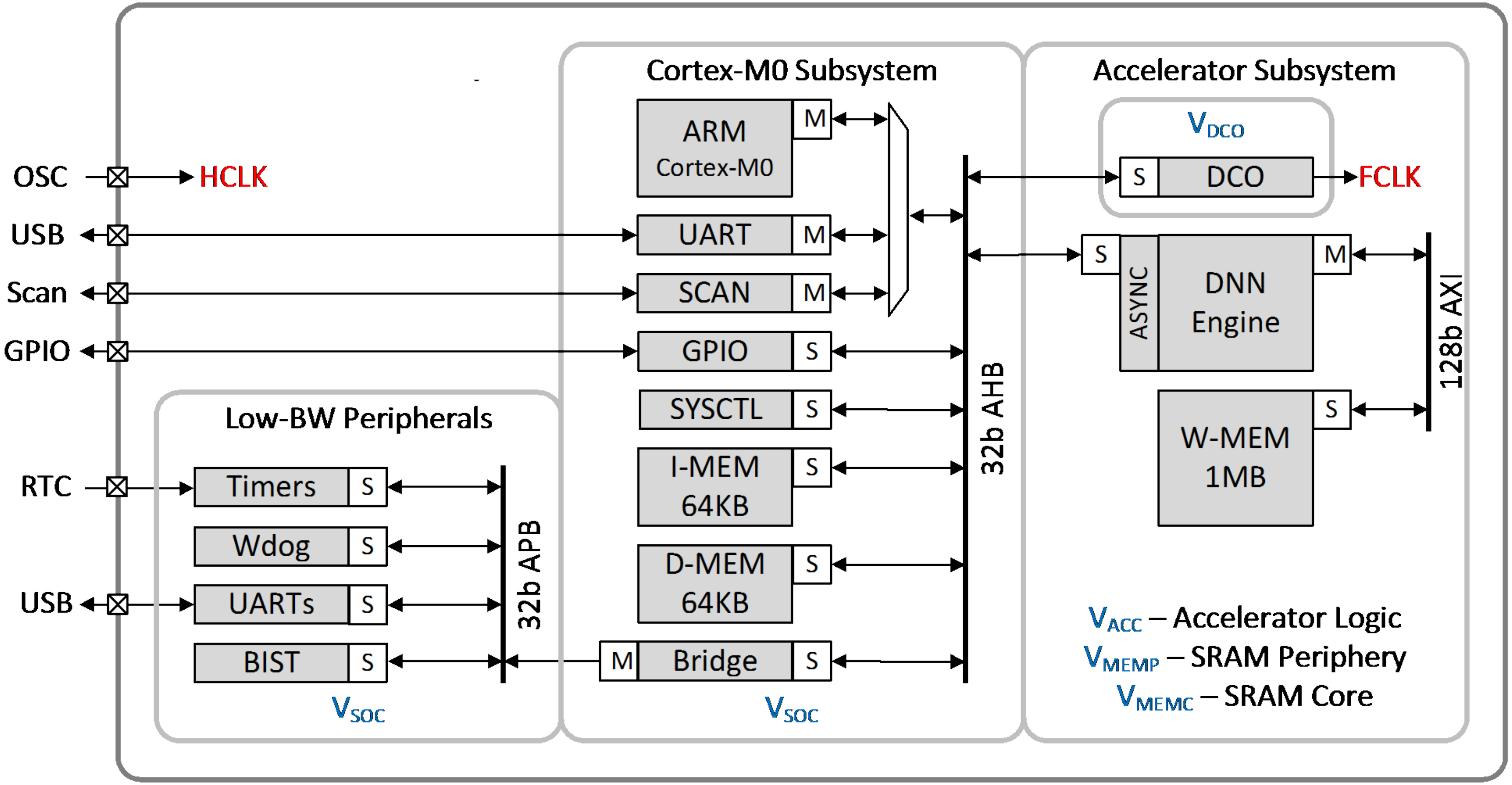


[Whatmough et al., ISSCC'17]

Outline

- Background and motivation
- DNN ENGINE
  - Parallelism and data reuse
  - Sparse data and small data types
  - Algorithmic resilience
- Measurement Results
- Summary

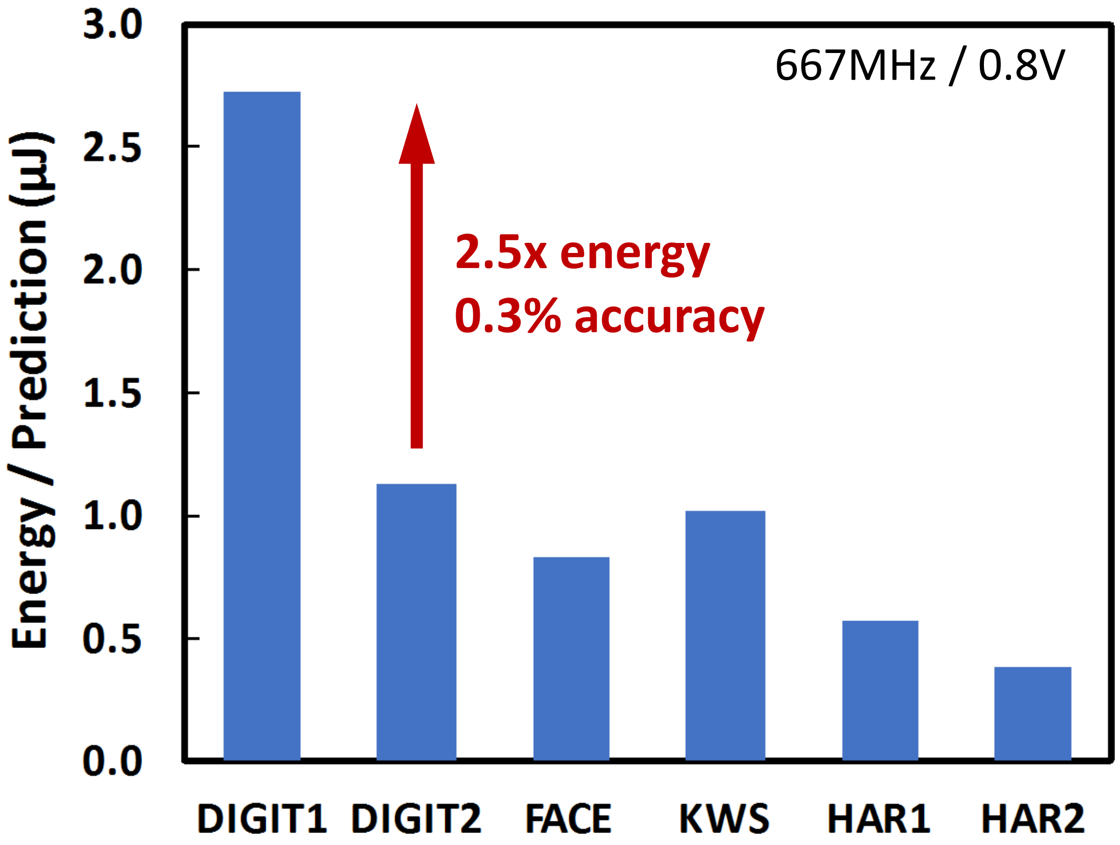
16nm SoC for always-on applications



16nm SoC for always-on applications

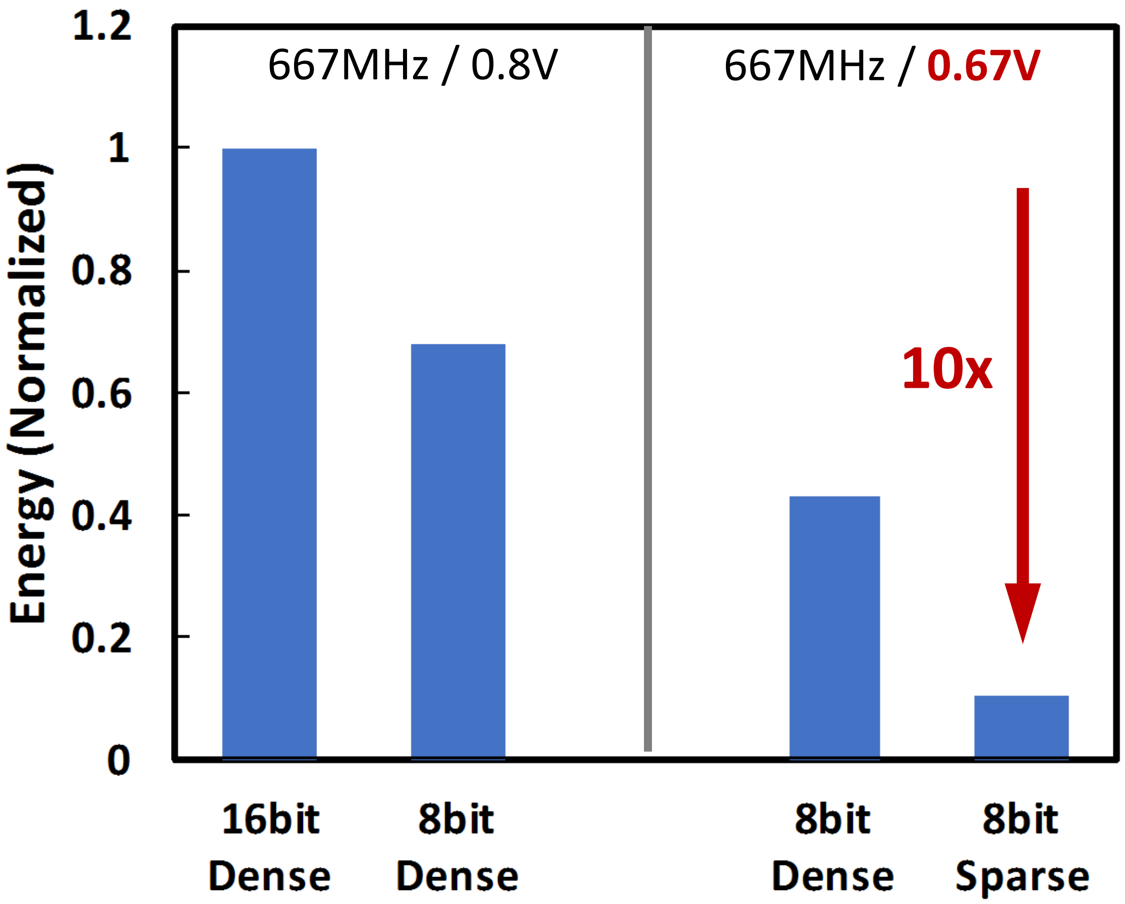
Measured energy for always-on applications

- Nominal Vdd / signoff Fmax
- Energy varies with application
  - Exponential with accuracy
- On-chip memory critical
  - A few KBs from DRAM >1uJ
  - Constrains model size



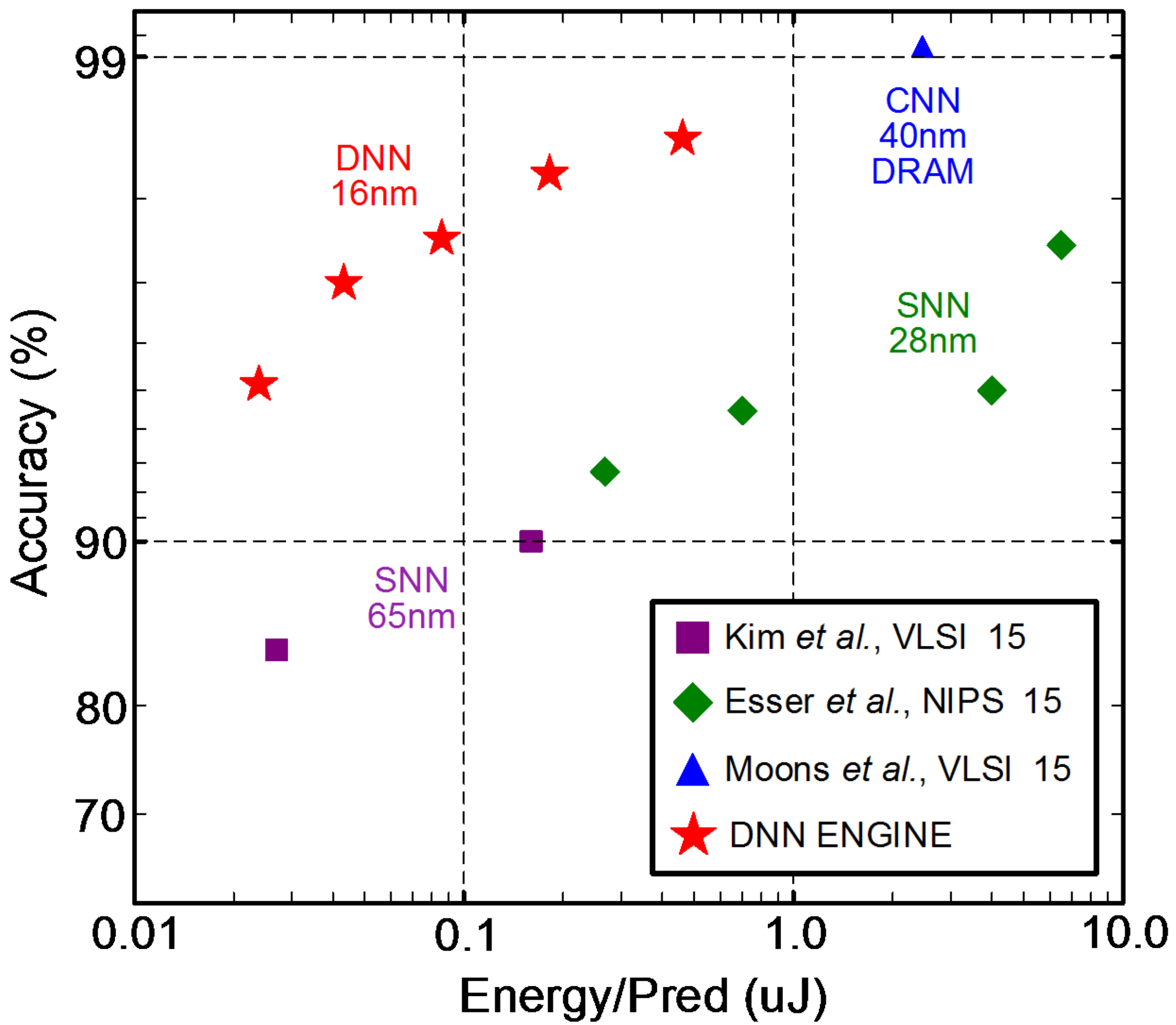
Measured energy improvement

- Joint architecture and circuit optimizations
- Typical silicon at room temp
- Measurements demonstrate
  - 10x energy reduction
  - 4x throughput increase



State of the art classifiers

- Dedicated NN accelerators
  - Different performance points
  - Different technologies
- Accuracy critical metric
  - Linear classifier 88%
  - Exponential cost
- The next 10x improvement
  - Algorithm innovations?



Summary

- Inference moving from cloud to the device: always-on sensing
- DNN ENGINE architecture optimizations
  - Parallelism and data reuse
  - Sparse data and small data types
  - Algorithmic resilience
- 16nm test chip measurements
  - Critical to store the model in on-chip memory
  - 10x energy and 4x throughput improvement
  - 1uJ per prediction for always-on applications

We are grateful for support from DARPA CRAFT and PERFECT projects