



MOFFETT AI

# Challenge and Opportunities to Accelerate ML Inference with Sparsity

Dr. Zhibin Xiao  
Co-founder and Chief-Architect  
Moffett AI

EDPS 2023, Oct 5<sup>th</sup>, 2023

# Outline

- 
- **Introduction to ML Inference**

---

  - Sparsity in ML Inference

---

  - Hardware Software Co-design for Sparsity

---

  - Case Studies: Sparsity Support in CPU, GPU and AI Chips

---

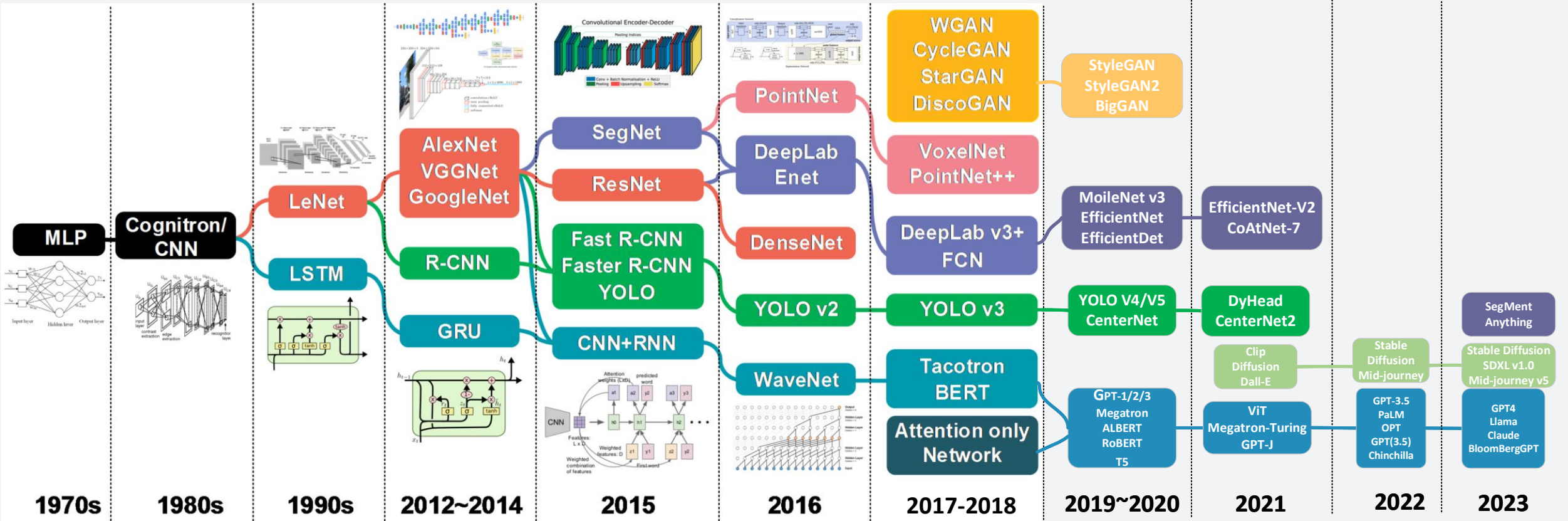
  - Summary

---

# A Brief History of AI Models

Computer Vision (CV) Models Explosion

Large Language Model (LLM) Explosion



\* Source: Partly from Hoi-Jun Yoo, ISSCC 2019

# Characteristics of Vision and Language Models

## Vision Models

- + Small models (millions of parameters)
- + Large Input Size (4k/8k images)
- + Throughput-sensitive within latency constraint
- + From convolution to transformer
- + Non-AI functions (image/video/pre-post processing)
- + Higher Parallelism and Computation-bounded

## Language Models

- + Large Models (billions of parameters)
- + Small input size (context window 128 - 32K)
- + Latency and Throughput Sensitive
- + Data-dependent computation (token by token)
- + Pre-post processing: Tokenizer, Beam Search (etc.)
- + Single-card to multiple-card inference
- + Memory or I/O bounded

**Sparsity benefits both Vision and Language Models:**

Reduce memory capacity and bandwidth requirement

Faster Computation

# Outline

- 
- Introduction to ML Inference

---

  - **Sparsity in ML Inference**

---

  - Hardware Software Co-design for Sparsity

---

  - Case Studies: Sparsity Support in CPU, GPU and AI Chips

---

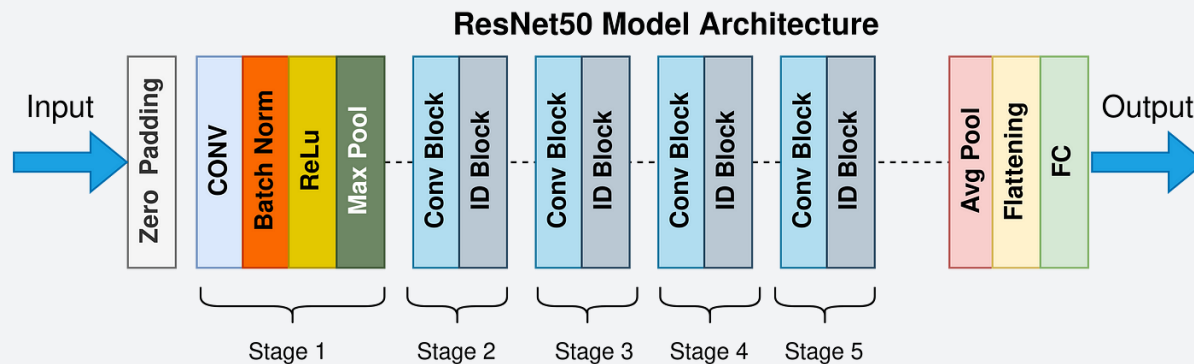
  - Summary

---

# Introduction to ML Inference

+ ML Model Operations Converges to a small subset of operators

- ONNX v1.15.0 (192 Operators)
- Key operators:
  - >90% of Number of Parameters and Computation FLOPS
  - Convolution, Matrix Multiplication, Inner Product, Element-wise Addition, Mean, Reshape, etc.



**ResNet50:** Conv, Matrix Multiplication, Pooling, ReLU

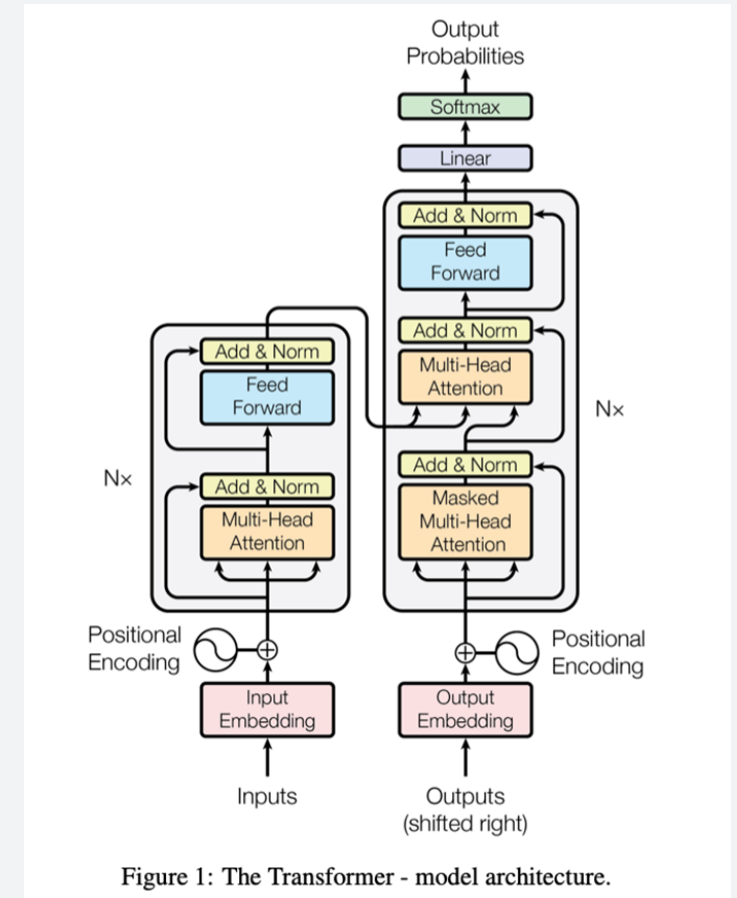
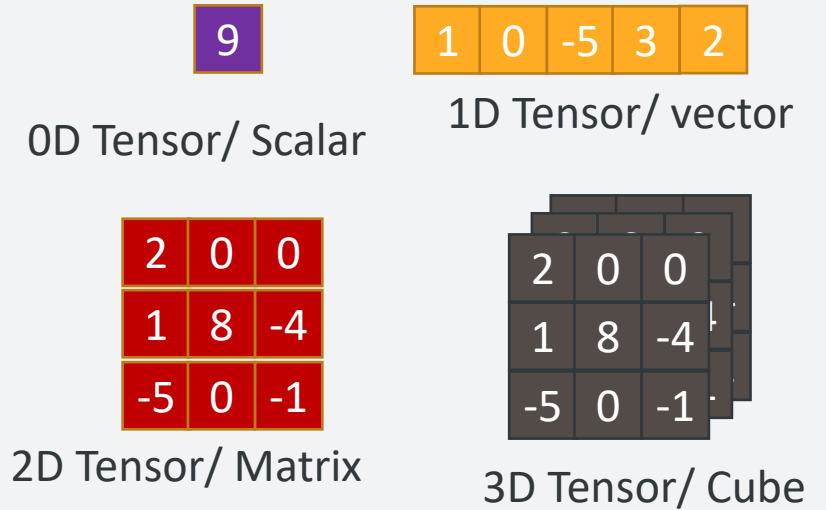


Figure 1: The Transformer - model architecture.

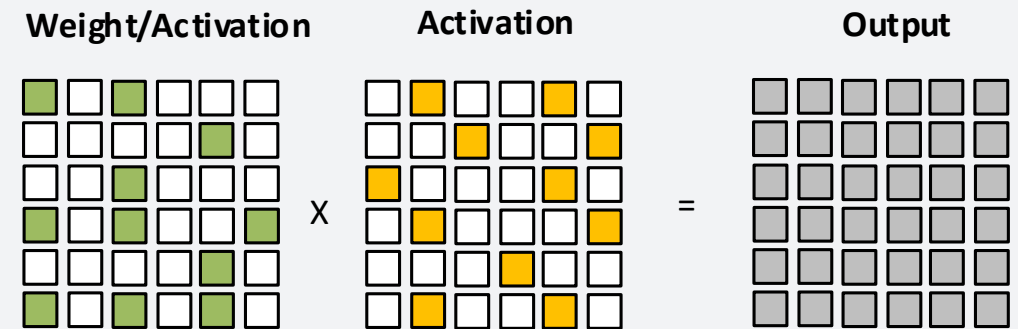
**Transformer:** Matrix Multiplication, Element-wise Operations, GELU, Softmax, Embedding Lookup, etc.

# Sparsity in ML Inference

- + The core of ML inference is **Tensor Algebra**
- + Zeros naturally exist or can be induced in Tensors
- + No need to store zero or compute zero in a tensor
  - **Huge benefits:** less storage, computation time, memory bandwidth, reduce power
  - **“Sparsity Tax”:** Extra HW cost for compression, decompression, schedule (limit the throughput and extra power/area overhead)



## Tensor Format



## Sparse Matrix Multiplication

# Sparsity is an Active Algorithm Research Area

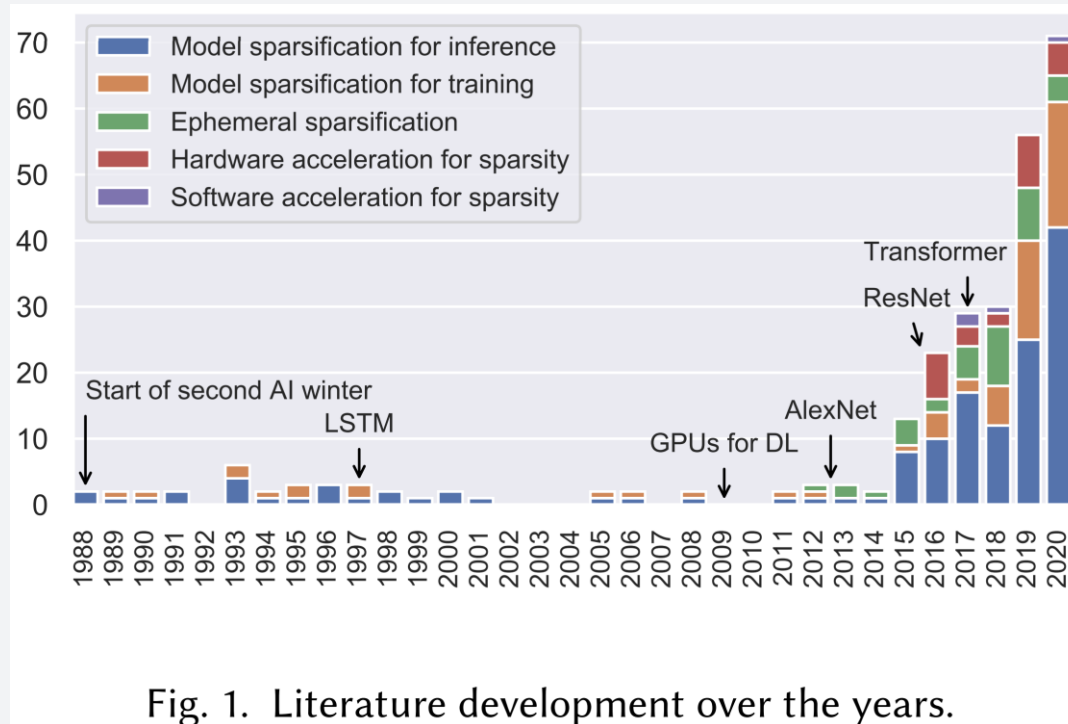
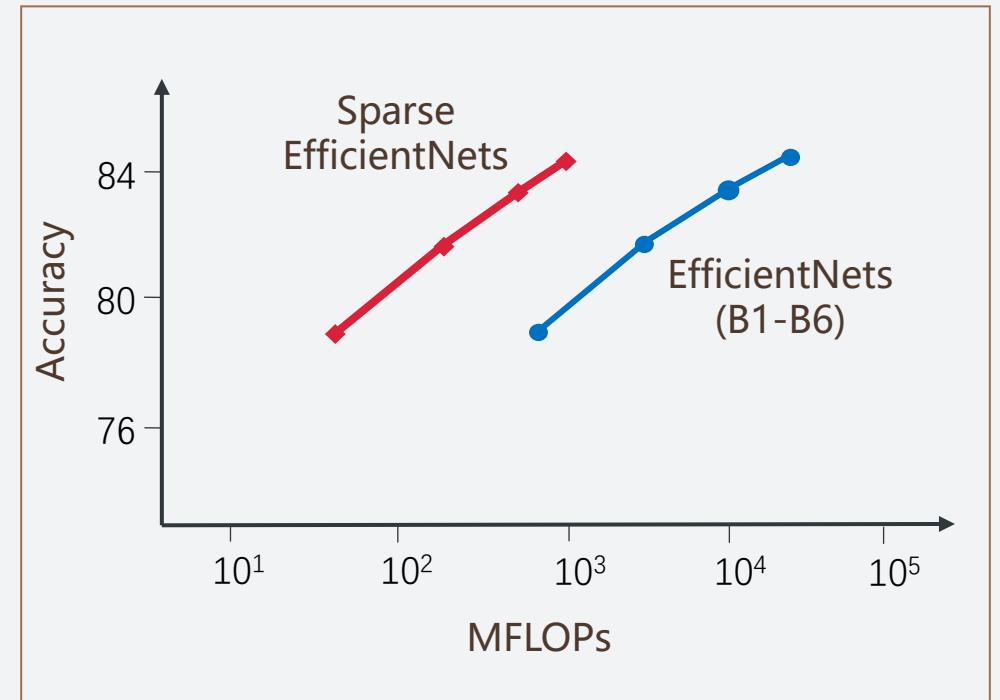


Fig. 1. Literature development over the years.



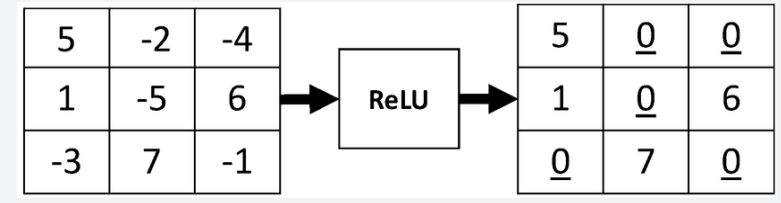
Google & Deepmind paper, "Fast Sparse ConvNets"

- The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks (MIT) – ICLR 2019 Best Paper
  - Any dense neural network contains one sparse neural network

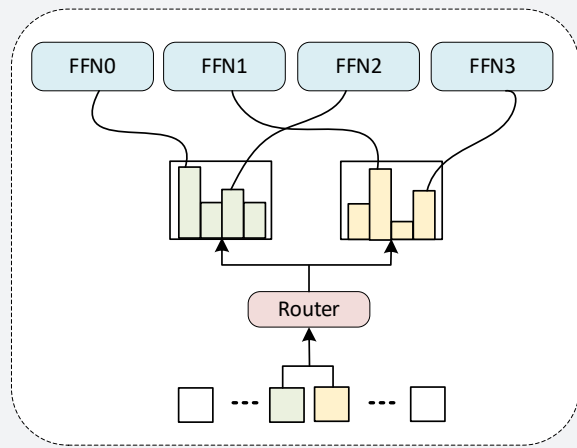


# Type of Sparsity in ML Inference

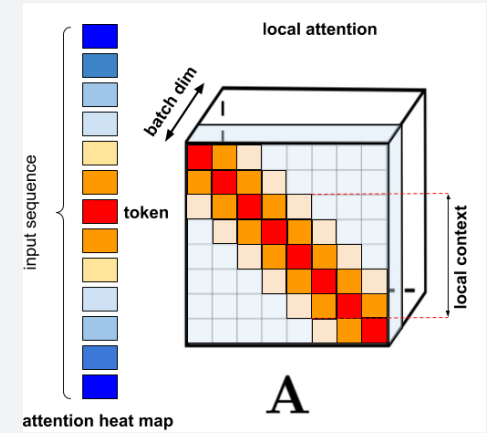
- + **Static and Dynamic Sparsity**
  - + **Static Sparsity**
    - + Static Weight Sparsity (Pruning)
  - + **Dynamic Sparsity**
    - + Activation Sparsity
    - + Conditional Sparsity
    - + Contextual/Attention Sparsity
    - + Mixture of Experts (MoE)
- + **Sparsity Granularity**
  - + **Coarse-granularity Sparsity**
  - + **Fine-grained Sparsity**
- + **Sparsity Patten**
  - + **Structured sparsity**
  - + **Unstructured sparsity**



**Activation Sparsity**



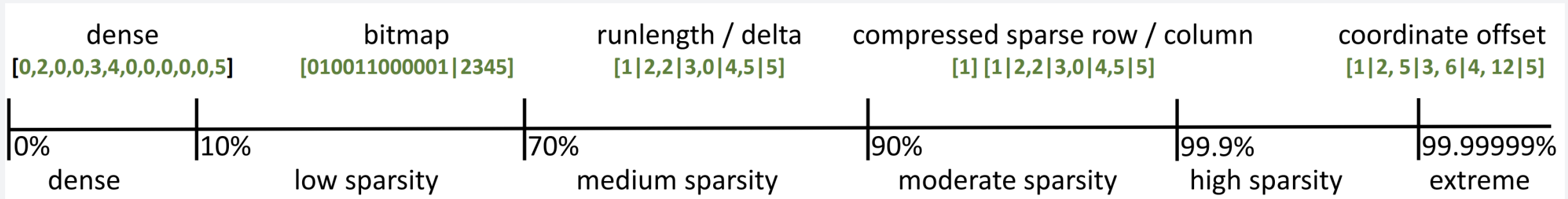
**Mixture of Experts (MoE)**



**Sparse Attention**

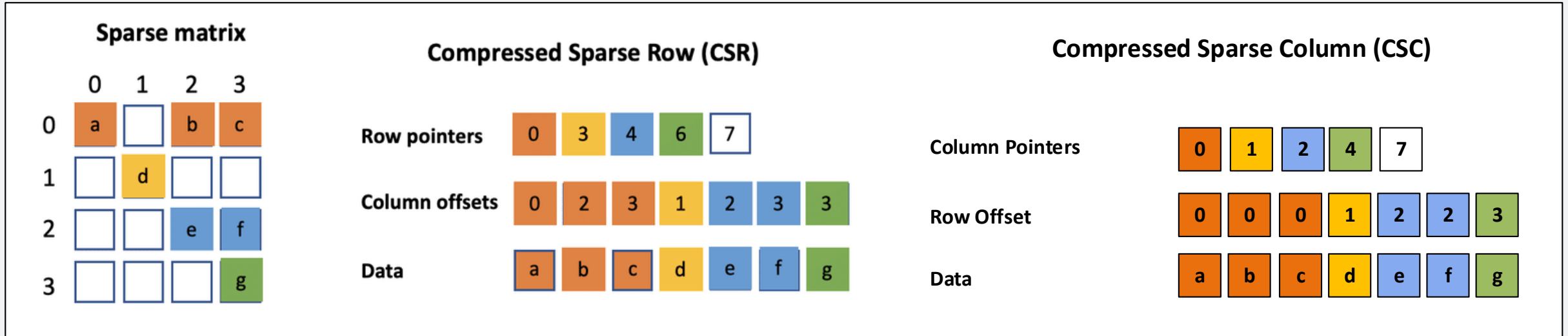
**Conditional Sparsity**

# Sparse Matrix Storage Format



- + Bitmap
- + Run-length / delta
- + Compressed Sparse Row / Column (CSR/CSC)
- + Coordinate Offset (index, value)
- + Hierarchical Hybrid Sparse Format

# Sparse Matrix Format: CSR and CSC Format



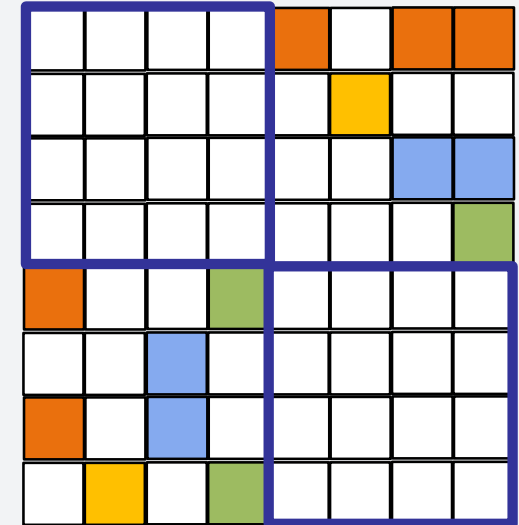
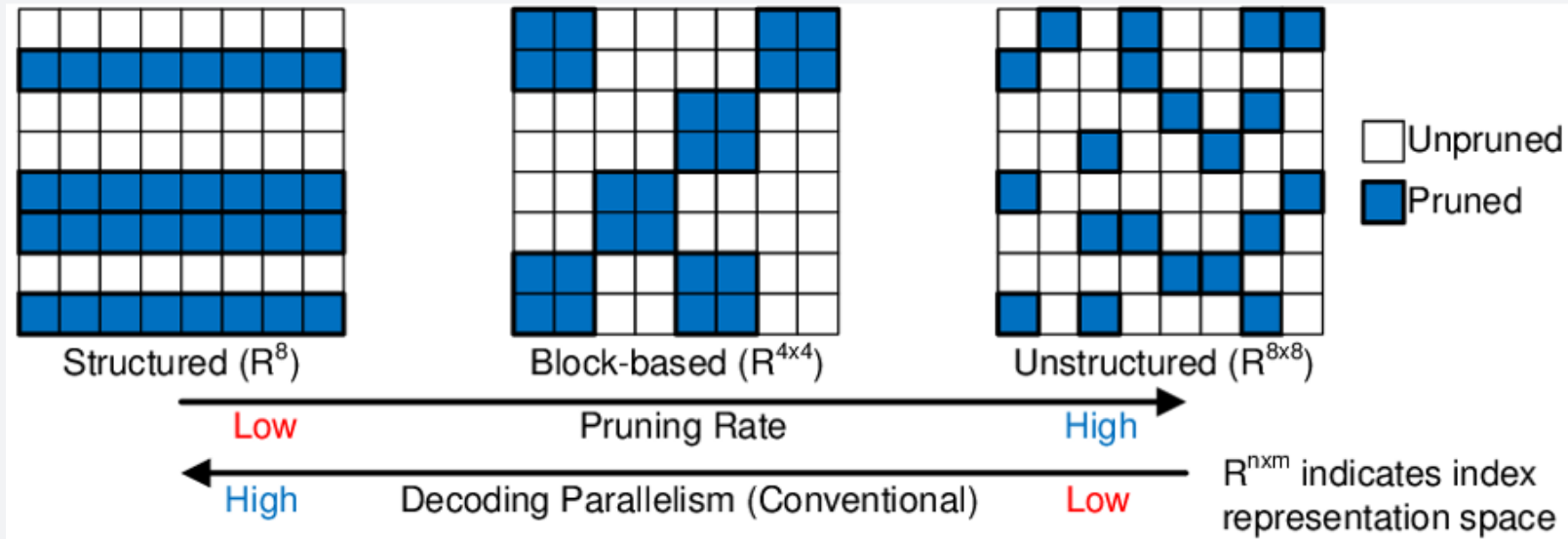
## CSR Format

- Data: an array for all non-zero values
- Column\_offsets[i]: records the actual column index of the data[i]
- Row\_pointers[i]: records the number of non-zero of all (i-1) rows

## CSC Format

- Data: an array for all non-zero values
- Row\_offsets[i]: records the actual row index of the data[i]
- Column\_pointers[i]: records the number of non-zero of all (i-1) columns

# Sparse Matrix Format: Coordinate Index and Hierarchical Hybrid Format



**Coordinate Index**  
Structured and Unstructured Sparsity

**Hierarchical Hybrid Format**  
Top-level: bit-vector format: (0, 1, 1, 0)  
Block-level: CSR/CSC/Coordinate Offset

# Outline

- 
- Introduction to ML Inference

---

  - Sparsity in ML Inference

---

  - **Hardware Software Co-design for Sparsity**

---

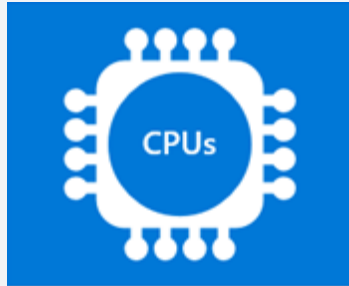
  - Case Studies: Sparsity Support in CPU, GPU and AI Chips

---

  - Summary

---

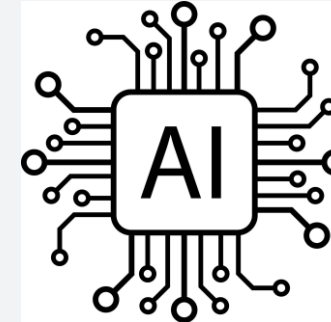
# Sparsity Support on Hardware Devices



Highly-sparse Matrix/Vector  
HPC field



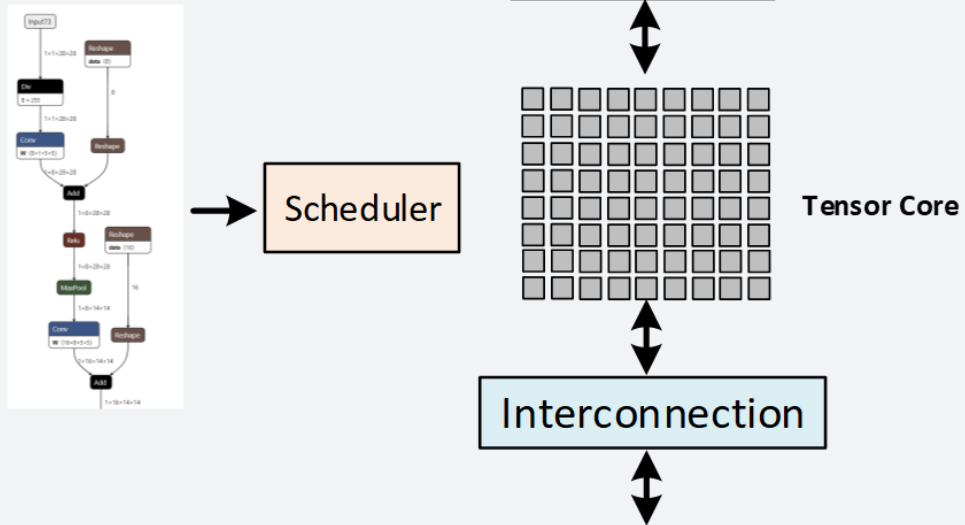
Coarse-grained sparsity  
Fine-grained 2:4 Structure Sparsity



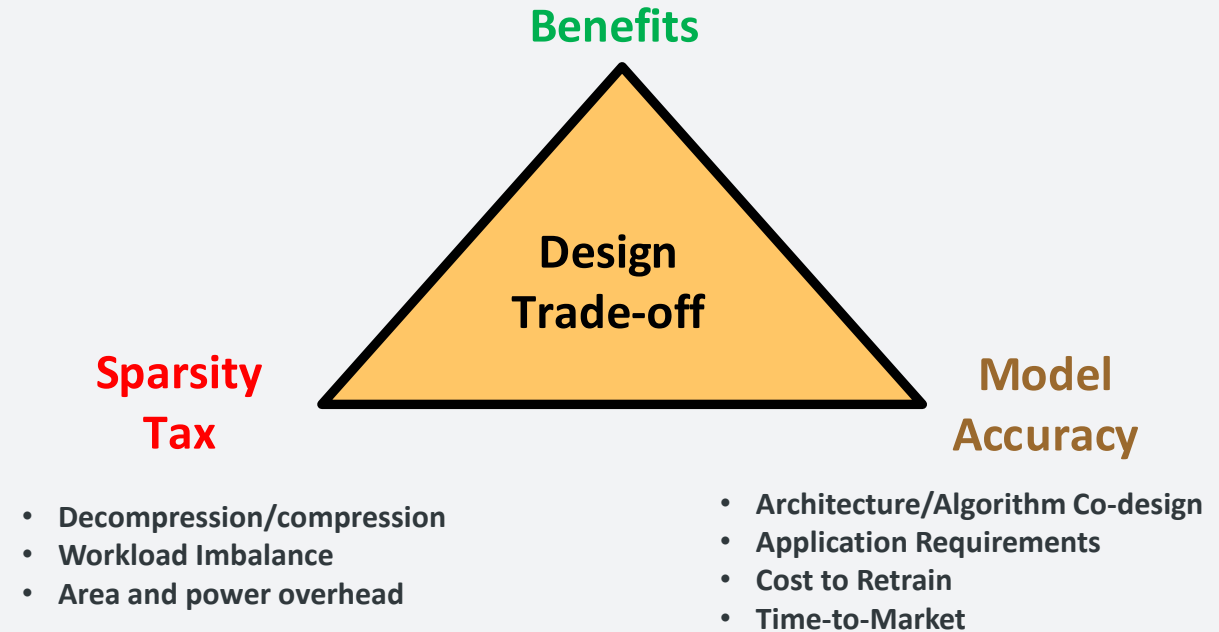
All sparsity type  
(Dynamic, Static, Structured, non-structured, fine-  
grained, coarse-grained, conditional execution)

# Challenges in Designing Sparse Accelerators

- TCO Saving
- Wall-clock speedup
- Power saving
- Area Saving (memory, die size)

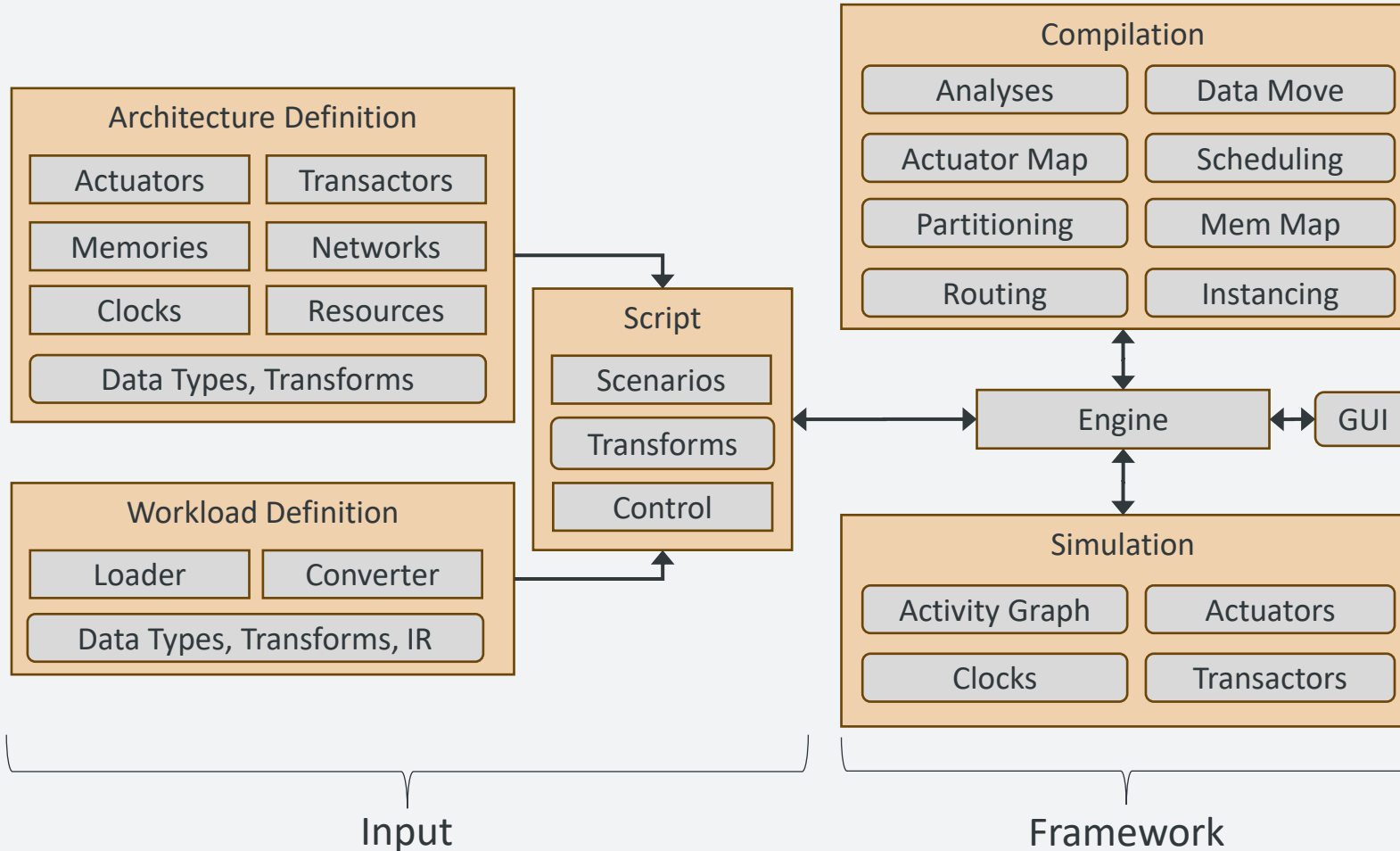


General AI Accelerator Architecture

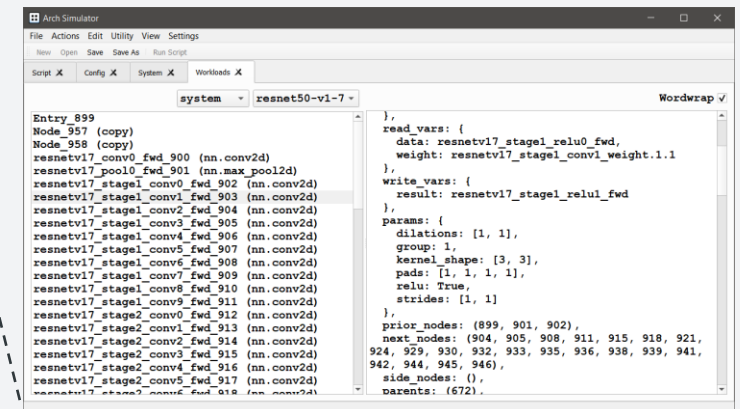


Sparse AI Accelerator Design Trade-off

# Example of End-to-End Compile-Aware Architecture Simulator



- + Rapid architectural exploration
  - Hardware Architecture Models
  - Workload Models
- + Adaptive compiler and simulator
- + Sparsity ratio and overhead in the loop
- + Results in seconds



```
Entry_899
Node_957 (copy)
Node_958 (copy)
resnetv17_conv0_fwd_900 (nn.conv2d)
resnetv17_pool0_fwd_901 (nn.max_pool2d)
resnetv17_stagel_conv0_fwd_902 (nn.conv2d)
resnetv17_stagel_conv1_fwd_903 (nn.conv2d)
resnetv17_stagel_conv2_fwd_904 (nn.conv2d)
resnetv17_stagel_conv3_fwd_905 (nn.conv2d)
resnetv17_stagel_conv4_fwd_906 (nn.conv2d)
resnetv17_stagel_conv5_fwd_907 (nn.conv2d)
resnetv17_stagel_conv6_fwd_908 (nn.conv2d)
resnetv17_stagel_conv7_fwd_909 (nn.conv2d)
resnetv17_stagel_conv8_fwd_910 (nn.conv2d)
resnetv17_stagel_conv9_fwd_911 (nn.conv2d)
resnetv17_stagel_conv0_fwd_912 (nn.conv2d)
resnetv17_stagel_conv1_fwd_913 (nn.conv2d)
resnetv17_stagel_conv2_fwd_914 (nn.conv2d)
resnetv17_stagel_conv3_fwd_915 (nn.conv2d)
resnetv17_stagel_conv4_fwd_916 (nn.conv2d)
resnetv17_stagel_conv5_fwd_917 (nn.conv2d)
resnetv17_stagel_conv6_fwd_918 (nn.conv2d)
```

```
read_vars: {
  data: resnetv17_stagel_relu0_fwd,
  weight: resnetv17_stagel_conv1_weight.1.1
},
write_vars: {
  result: resnetv17_stagel_relu1_fwd
},
params: {
  dilations: [1, 1],
  group: 1,
  kernel_shape: [3, 3],
  pads: [1, 1, 1, 1],
  relu: True,
  strides: [1, 1]
},
prior_nodes: (899, 901, 902),
next_nodes: (904, 905, 908, 911, 915, 918, 921,
924, 929, 930, 932, 933, 935, 936, 938, 939, 941,
942, 944, 945, 946),
side_nodes: (),
parents: (672),
```



# Outline

- 
- Introduction to ML Inference

---

  - Sparsity in ML Inference

---

  - Hardware Software Co-design for Sparsity

---

  - **Case Studies: Sparsity Support in CPU, GPU and AI Chips**

---

  - Summary

---

# An Overview of Mainstream AI Accelerator Architecture

## Popular AI Accelerators

- CPU (X86, RISC-V): Vector/Matrix Instruction Extension
- Nvidia Tensor Core: 4x4 GEMM
- Huawei Ascend: 16x16 GEMM + VPU
- Google TPU: Systolic Array + VPU
- Graphcore: Massively Parallel BSP Cores
- SambaNova: Dataflow RDU
- Cerebras: Wafer-scale many-core architecture
- Habana Labs/Intel Spring Hill: DSP Array + GEMM
- Cambricon/Hanguang 800/NVDLA/Tesla FSD: DSA Accelerators

## Special Technology AI Accelerators

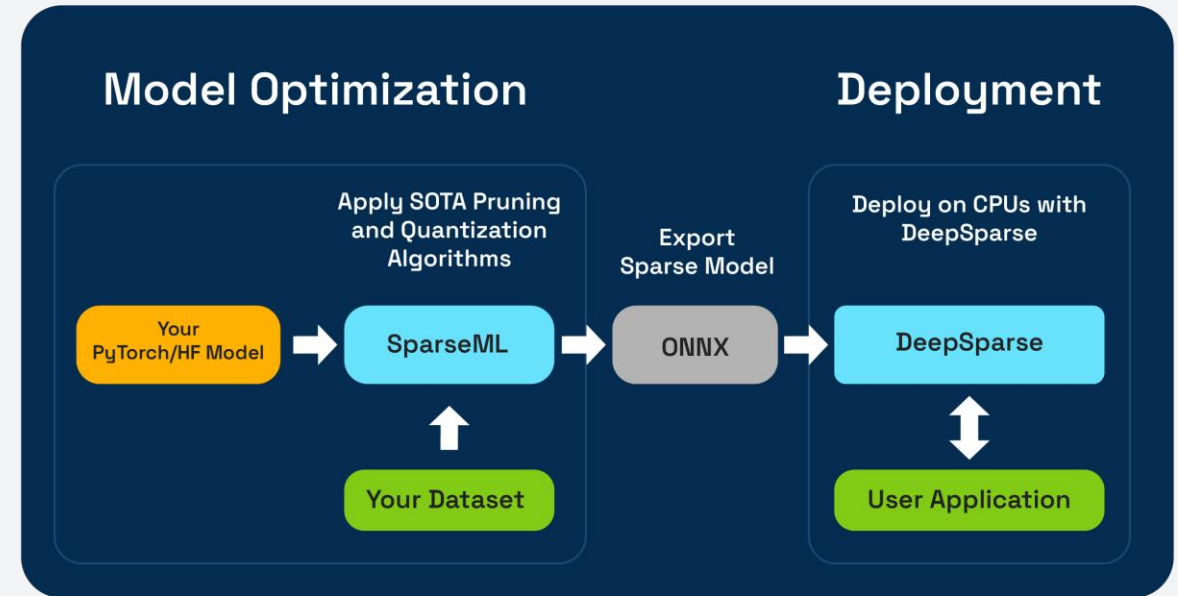
- Spiking Neural Nets and Neuromorphic Architectures
- Resistor/Memristor matrices and Analog Computing
- Optical and Spintronics Implementations

- **Key buzz words:** Systolic Array, Tensor Core, Vector Core, Many-core, DSA, Dataflow
- **Special Technology AI Accelerators:** Very efficient for specific applications, limited operator support

# Sparsity Support on CPUs

## + CPU offers thread-level parallelism and dense vector/matrix extension

- Limited by low peak MAC performance of CPUs
- Limited by SW for sparse matrix compress and decompress
  - Limit speedup for sparse matrix
  - Available Intel Sparse BLAS support

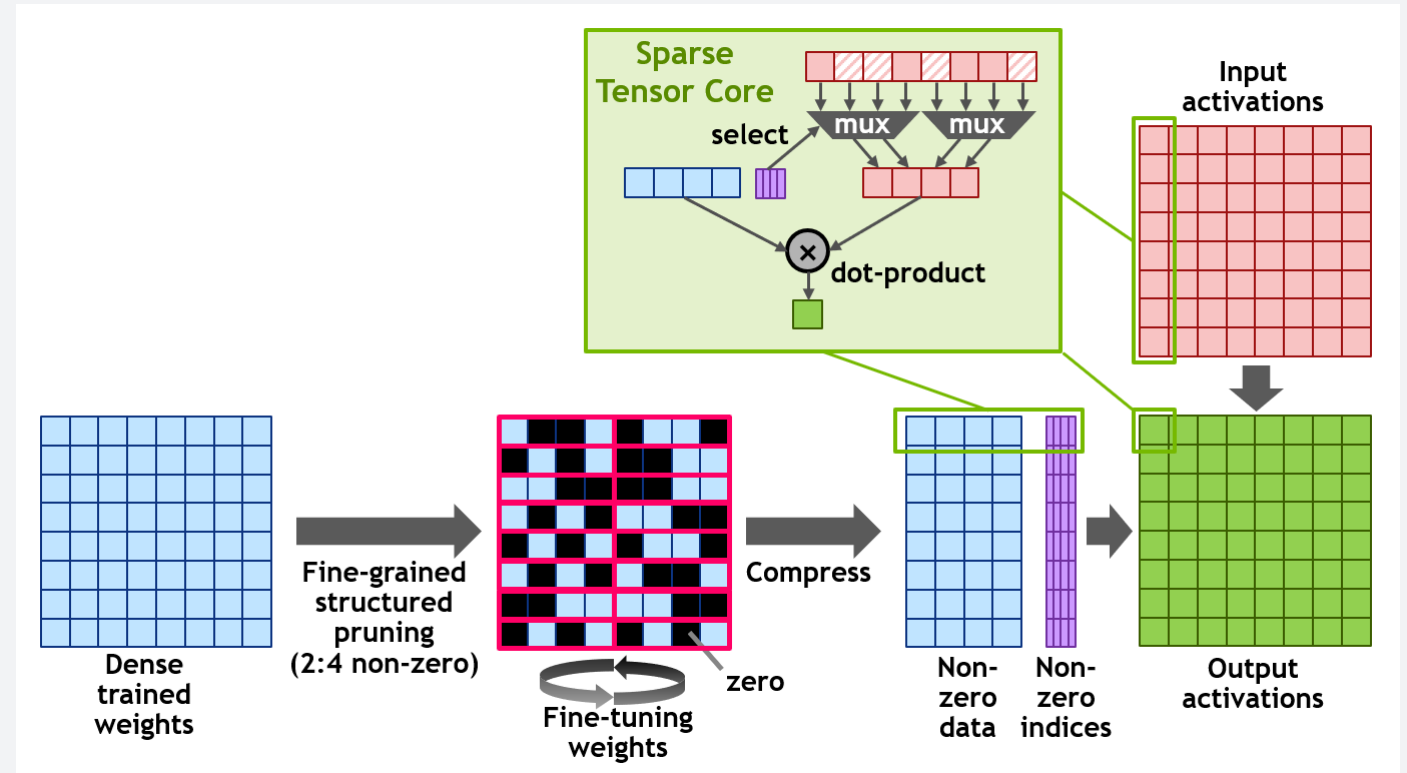


Neuralmagic's DeepSparse Inference Runtime on CPU

# Nvidia Ampere/Hopper Sparse Tensor Core

## + Fine-grained Structured 2:1 Sparsity

- Minimum change on Tensor-core Design
- Strong constraint on the sparsity distribution
- 1.44x – 1.85x speedup on Matmul/Conv Kernels
- 1.3x – 1.5x speedup for end-to-end applications (BERT/ResNetXt)



# MIT Eyeriss Project – Eyeriss v1 (2016 ISSCC)

## + One of the earliest AI Accelerator chip

- A Spatial Multi-PE architecture
- Support Weight Sparsity by reducing memory footprint and bandwidth
- Saving power by clock gating PE for zero operands
- No wall-clock speedup

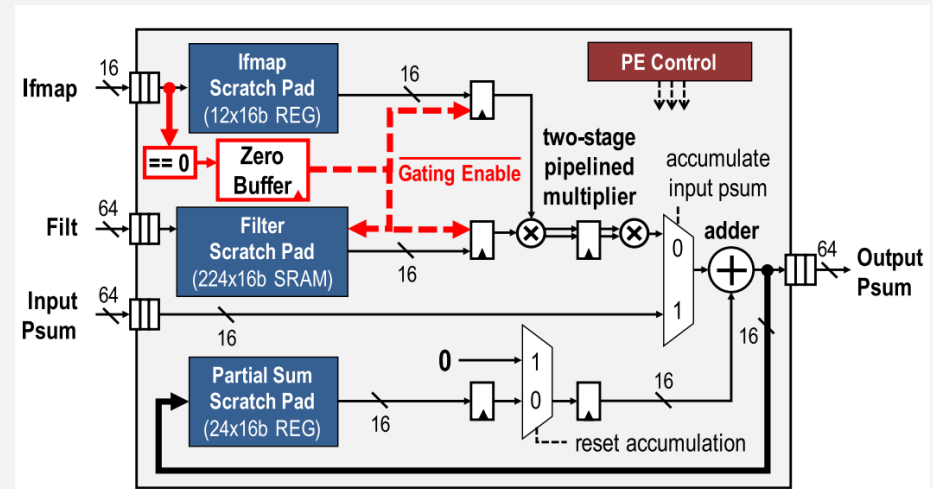
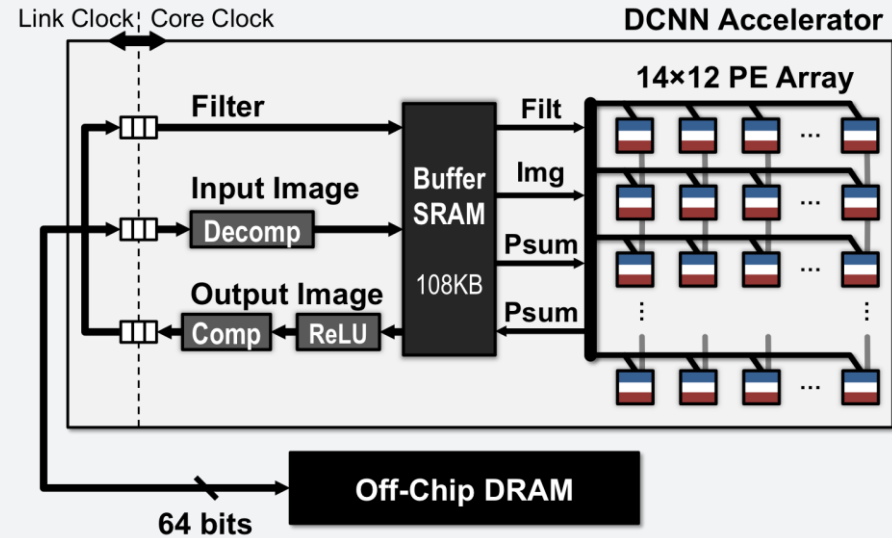


Fig. 12. PE architecture. The datapaths in red show the data gating logic to skip the processing of zero ifmap data.

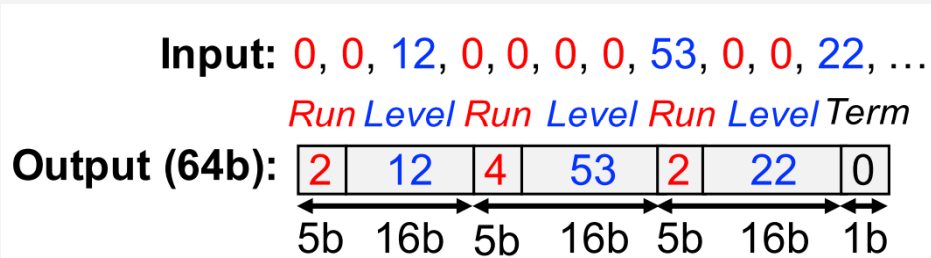


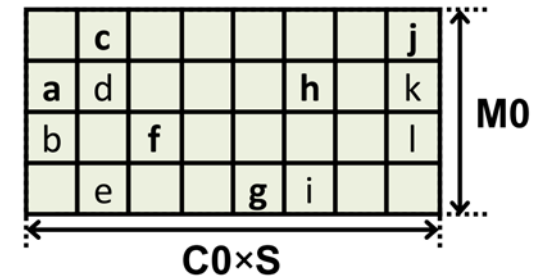
Fig. 8. Encoding of the RLC.

# MIT Eyeriss Project – Eyeriss v2 (2018)

## + Compared to Eyeriss v1

- A Scalable Architecture
- Change of matrix compressed format
- Dual-sparsity Support
- Wall-clock speedup

### Weight Matrix

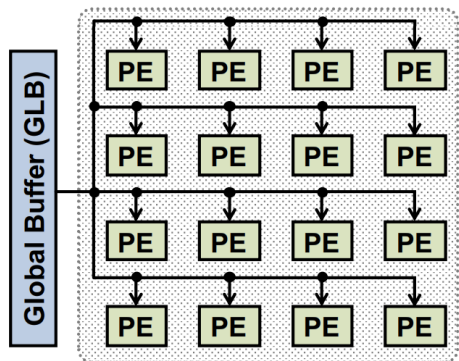


### CSC Compressed Data:

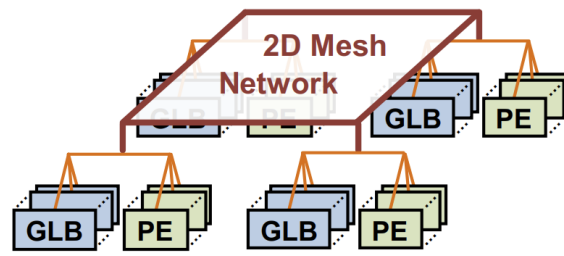
data vector: {a, b, c, d, e, f, g, h, i, j, k, l}

count vector: {1, 0, 0, 0, 1, 2, 3, 1, 1, 0, 0, 0}

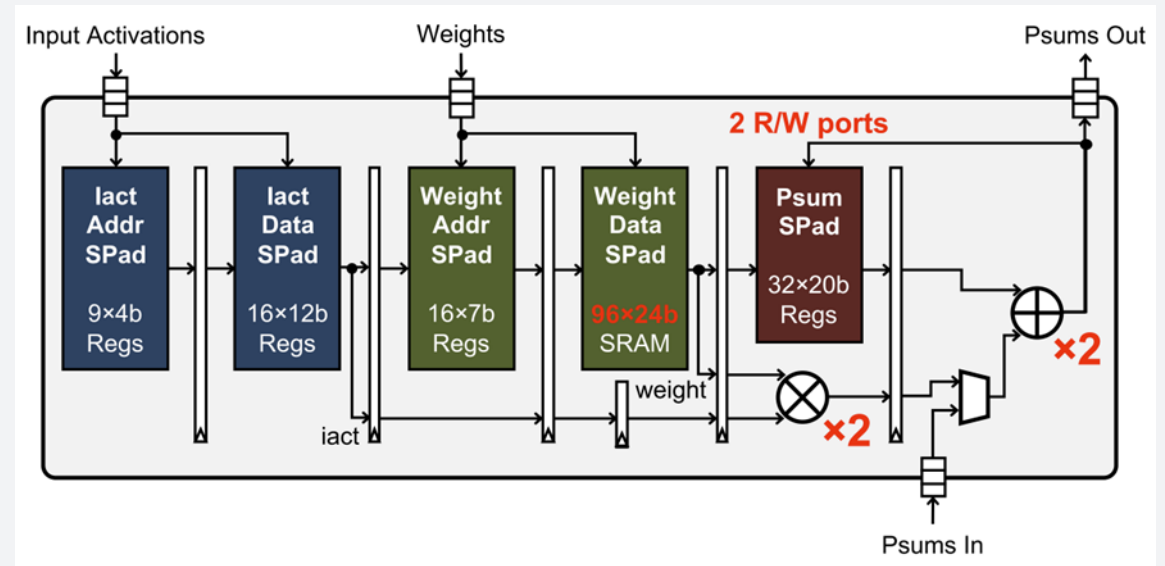
address vector: {0, 2, 5, 6, 6, 7, 9, 9, 12}



(a) Original Eyeriss



(b) Eyeriss v2



# EIE: Efficient Inference Engine on Compressed Deep Neural Network (2016)

## + One of the earliest AI Accelerator research

- A Spatial Multi-PE architecture
- Support dual sparsity by reducing memory footprint and bandwidth and save wall-clock speedup
- Weight matrices: CSC format
- Proposed an activation buffer before different PEs for workload balance
- Use activation to lookup compressed weight

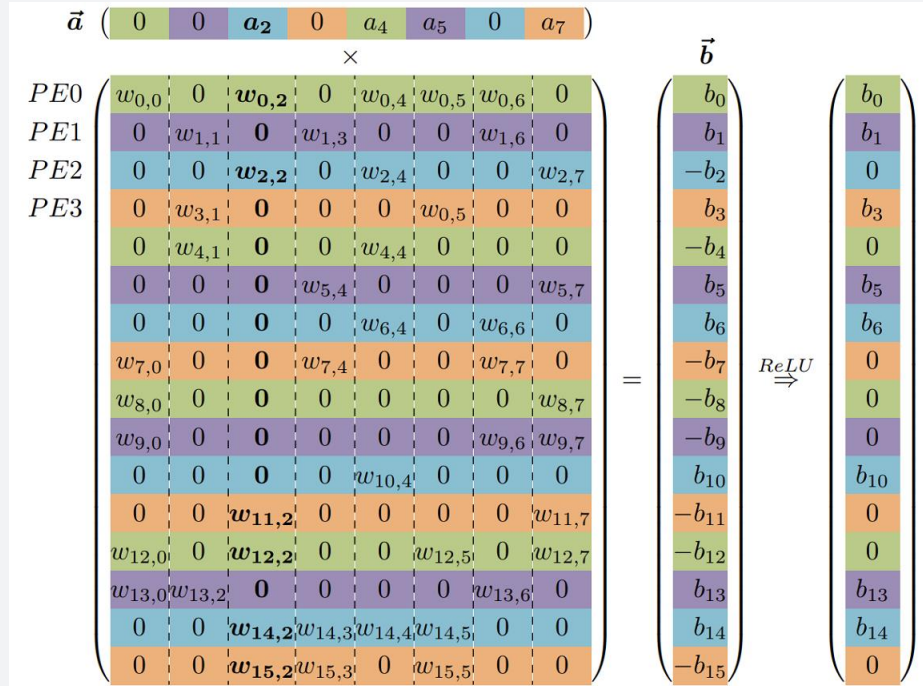


Figure 2. Matrix  $W$  and vectors  $a$  and  $b$  are interleaved over 4 PEs. Elements of the same color are stored in the same PE.

Virtual Weight	$W_{0,0}$	$W_{8,0}$	$W_{12,0}$	$W_{4,1}$	$W_{0,2}$	$W_{12,2}$	$W_{0,4}$	$W_{4,4}$	$W_{0,5}$	$W_{12,5}$	$W_{0,6}$	$W_{8,7}$	$W_{12,7}$
Relative Row Index	0	1	0	1	0	2	0	0	0	2	0	2	0
Column Pointer	0	3	4	6	6	8	10	11	13				

Figure 3. Memory layout for the relative indexed, indirect weighted and interleaved CSC format, corresponding to  $PE_0$  in Figure 2.

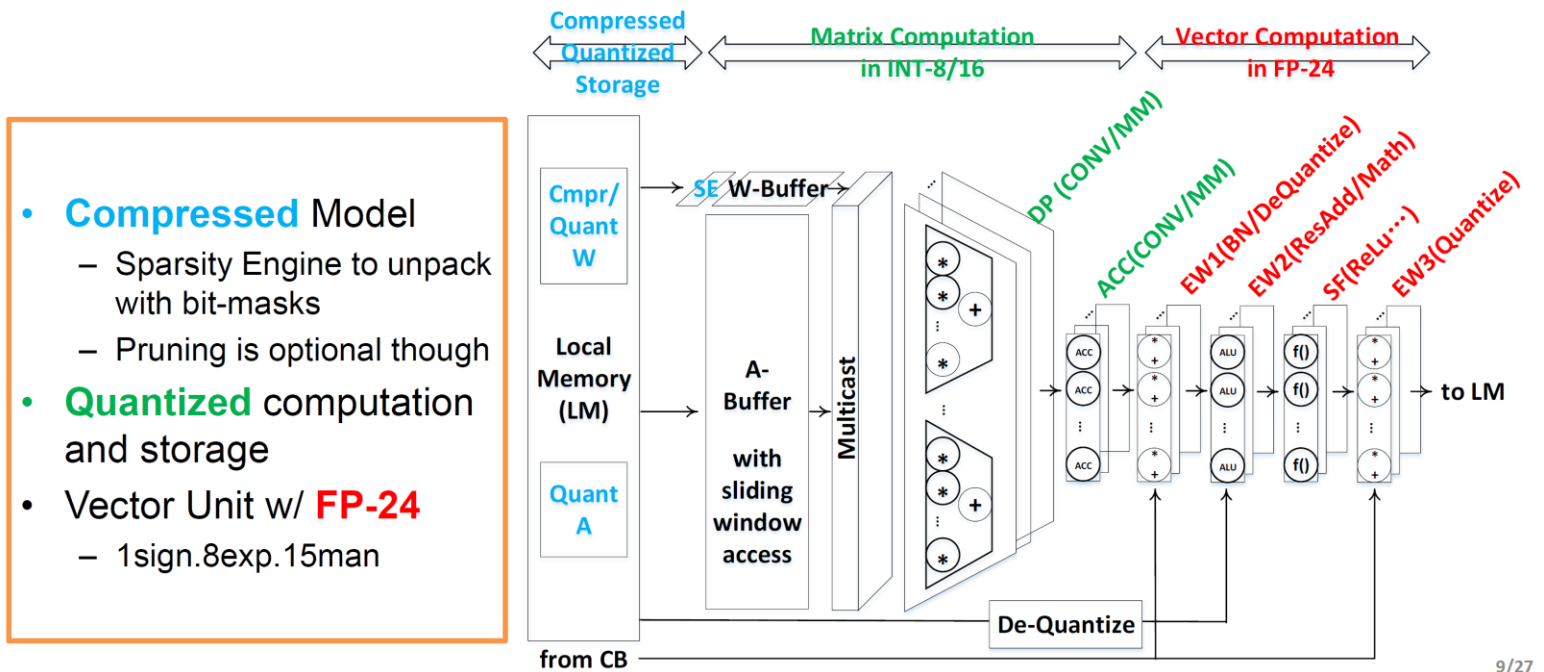


# Alibaba Hanguang-800 Sparsity Engine (2020)

## + A High-performance Commercial Data-center Inference Chip

- DSA architecture
- Support weight compression in memory to reduce memory footprint
- No external DDR and all-on-chip Memory
- Weight matrices: bit-vector representation for low to medium sparsity
- No wall-clock speedup

## Compressed and Quantized Storage/Processing



9/27

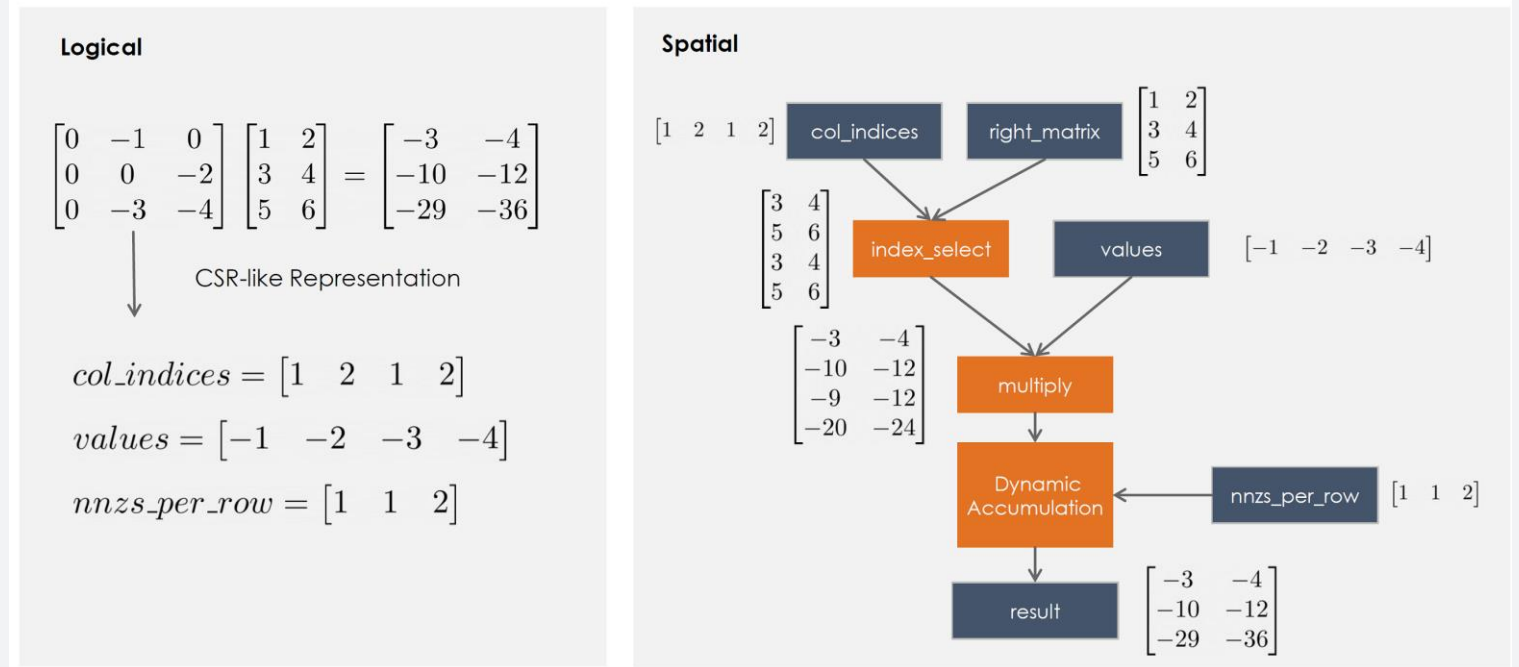


# SambaNova RDU Sparsity Support

## + A Reconfigurable Dataflow tiled Architecture (RDU series)

- Scalable design with on-chip switch connect array of RDUs and memory units
- Scale-out support
- Support CSR-like matrix compression
- Wall clock-time speedup

### Sparse Matrix Multiply on RDU



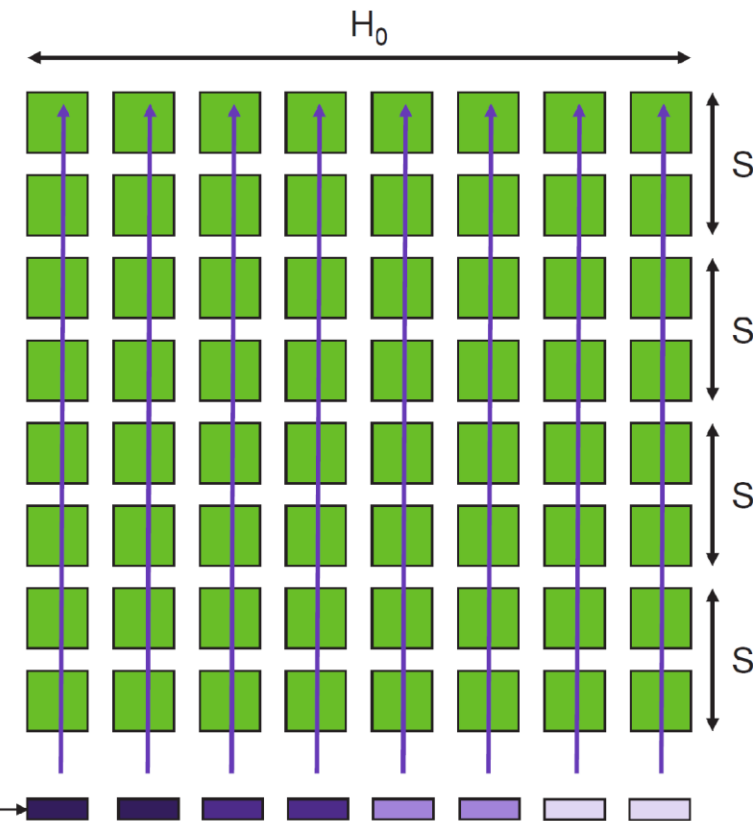
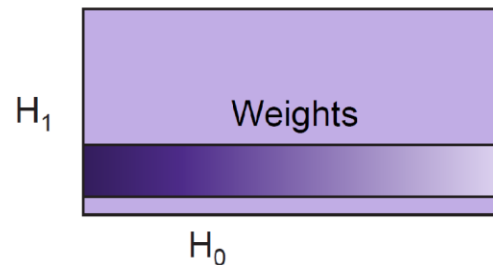
# Cerebras Sparsity Support

- + A commercial data-flow wafer-scale spatial architecture
- + Fine-granularity fully unstructured sparse MatMul
- + 10x sparse utilization vs. GPU
- + Not clear on the weight sparse storage format

## GEMM with Sparse Input

**Dataflow scheduling enables fully unstructured sparse MatMul with low overhead**

- Executed as a series of AXPY operations per row
- Row of non-zero weights broadcast over columns of cores
- Each individual weight triggers FMACs
- No compute for zero weights, not streamed in at all
- No memory used for weights, not even stored temporarily



# Moffett Deep-Sparse AI Inference Cards



- Complete system-on-chip with deep sparse processing units supporting up to **32x sparsity**
- One chip multiple PCIe products
- Complete end-to-end software toolchain (SparseOPT, SparseRT, SOLA runtime), please check <http://docs.moffettai.com>
- AI benchmark MLCommons validated performance results, please check <http://mlcommons.org>



# Summary

## + Sparsity is an active research area

- Promising direction for both Vision and LLM
- Save computation, memory bandwidth/capacity and power
- Reduce TCO

## + The memory storage format is the key

- Affected by algorithm (sparsity ratio, accuracy)
- Impact on Memory/Datapath/Scheduler Design

## + Sparse AI Accelerator needs trade off on more dimension

- Model Accuracy, Sparsity Overhead & Sparsity Benefits

## + Research and Commercial AI accelerators are embracing sparsity



MOFFETT AI

Thank you and Questions?