NeuronFlow

MEET: Affordable Temporal execution

Orlando Moreira, PhD
GrAICore Chief Architect

(credit to the whole team: Luc Waeijen, Steven Roos, Hamid Tabani, Zeqi Zhu, Arash Pourtaherian, Savvas Sioutas, Rij Jan Zwartenkot, Peter Kievits ...)

Neural network Sparsity

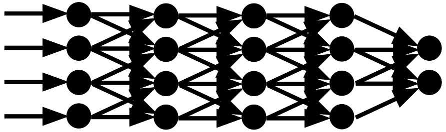
- **Structural Sparsity**
  - Pruning of needless weights
    - In typical image networks >70% with negligible loss
- **Activation Sparsity**
  - No relevant data results in 0-valued activations.
    - RELU activation function: ~50% of activations are 0-valued
  - More, if trained for activation suppression!
- **Temporal Sparsity**
  - Little change from instant to instant
    - why re-process the whole image?

Full frame

Difference consecutive frames

Introduction: Sparse Computing

Process everything



Prediction

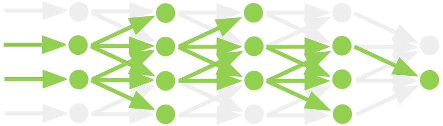
Dog

GPUs, CPUs

Redundant

Inefficient

Process only non-zeros



Prediction

Dog

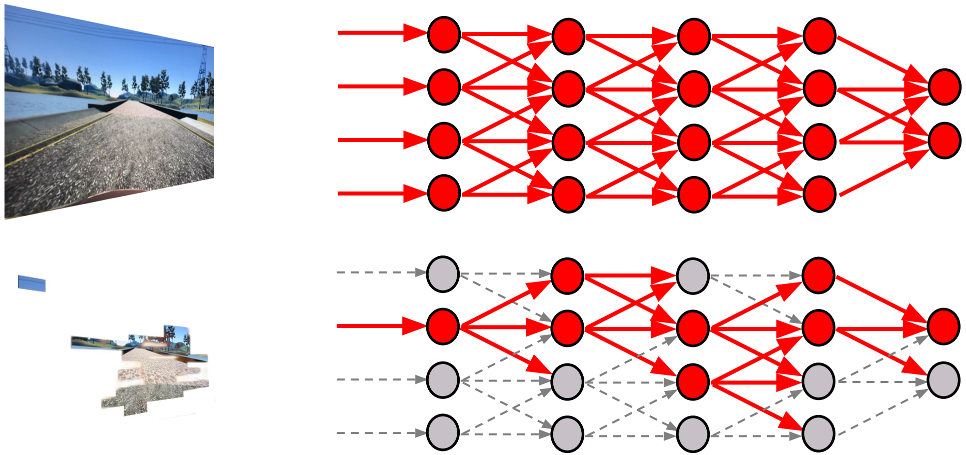
GrAI-VIP

Low power

Low latency

Fewer activations result in **less compute load**

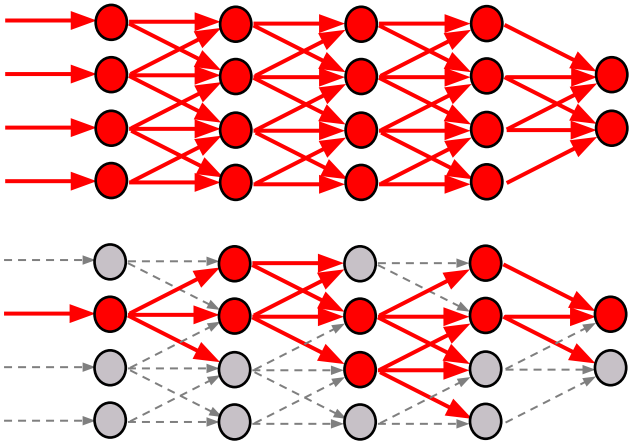
Exploiting temporal sparsity



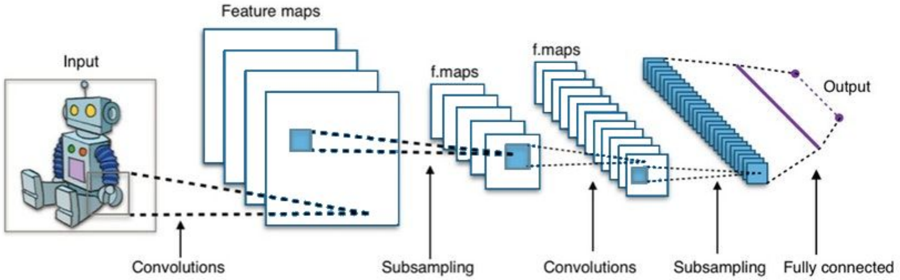
Red = active links and activated neurons

- **Event-based: only propagates changes**;
- Difference calculation
  - Computes difference
  - Integrates output
- Requires **resilient** neuron state;
  - used to cost a lot of memory (5x)
  - we reduced to 1.2x (new result)
- **Threshold:** per neuron, how much change is needed to warrant propagation.

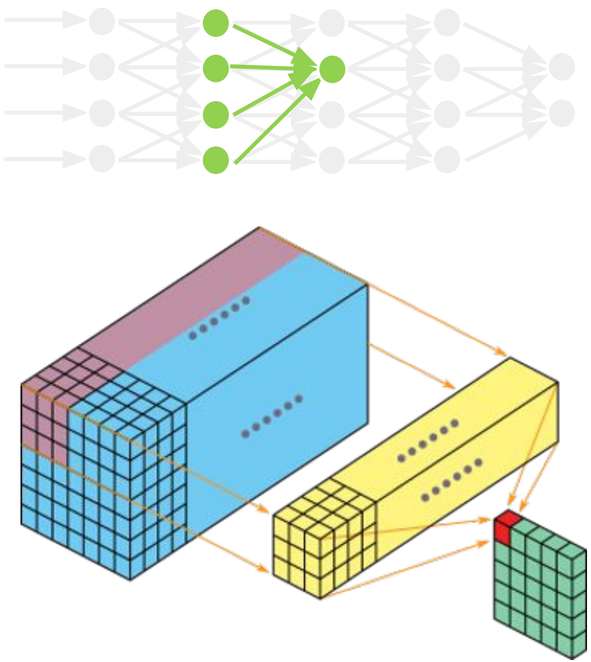
Consequences of Activation Sparsity for Computer Architecture



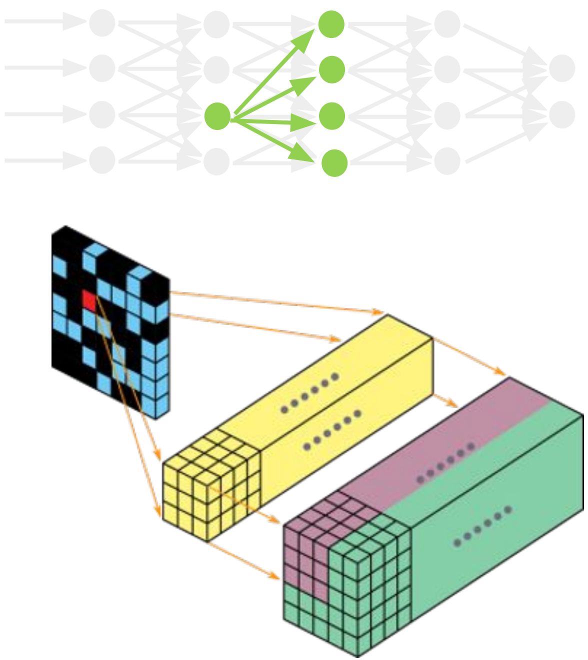
- Sparsity reduces **regularity** in compute demand:
  - Breaks sequential memory access
  - Sporadic activity, non-deterministic;
- Exploiting activation sparsity means **skipping activations:**
  - Input stationary execution (next slide)
- Temporal sparsity needs resilient neuron state
  - Storing state frame to frame
- **Training will have a major impact on performance.**



Convolution Execution Order

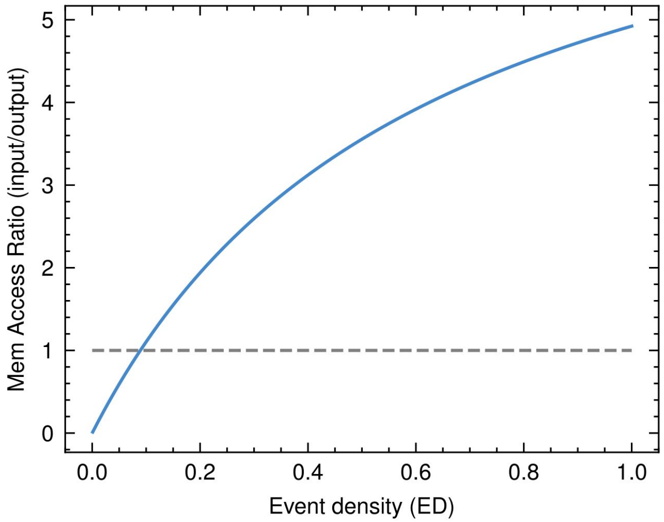


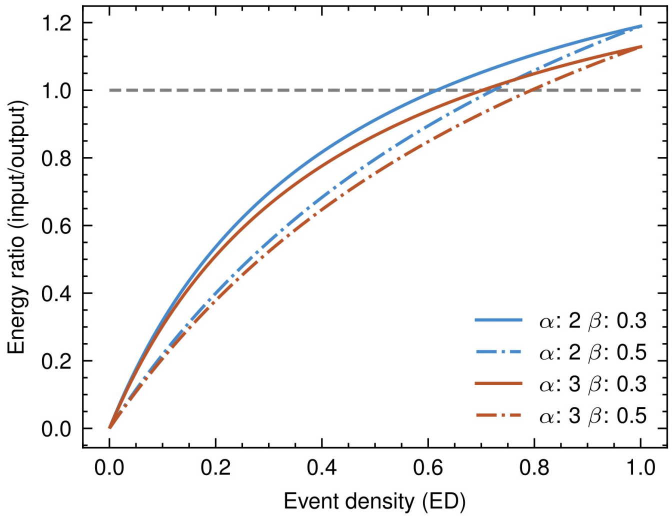
Standard
Convolution
(output centric)



Event-driven
Convolution
(input centric)

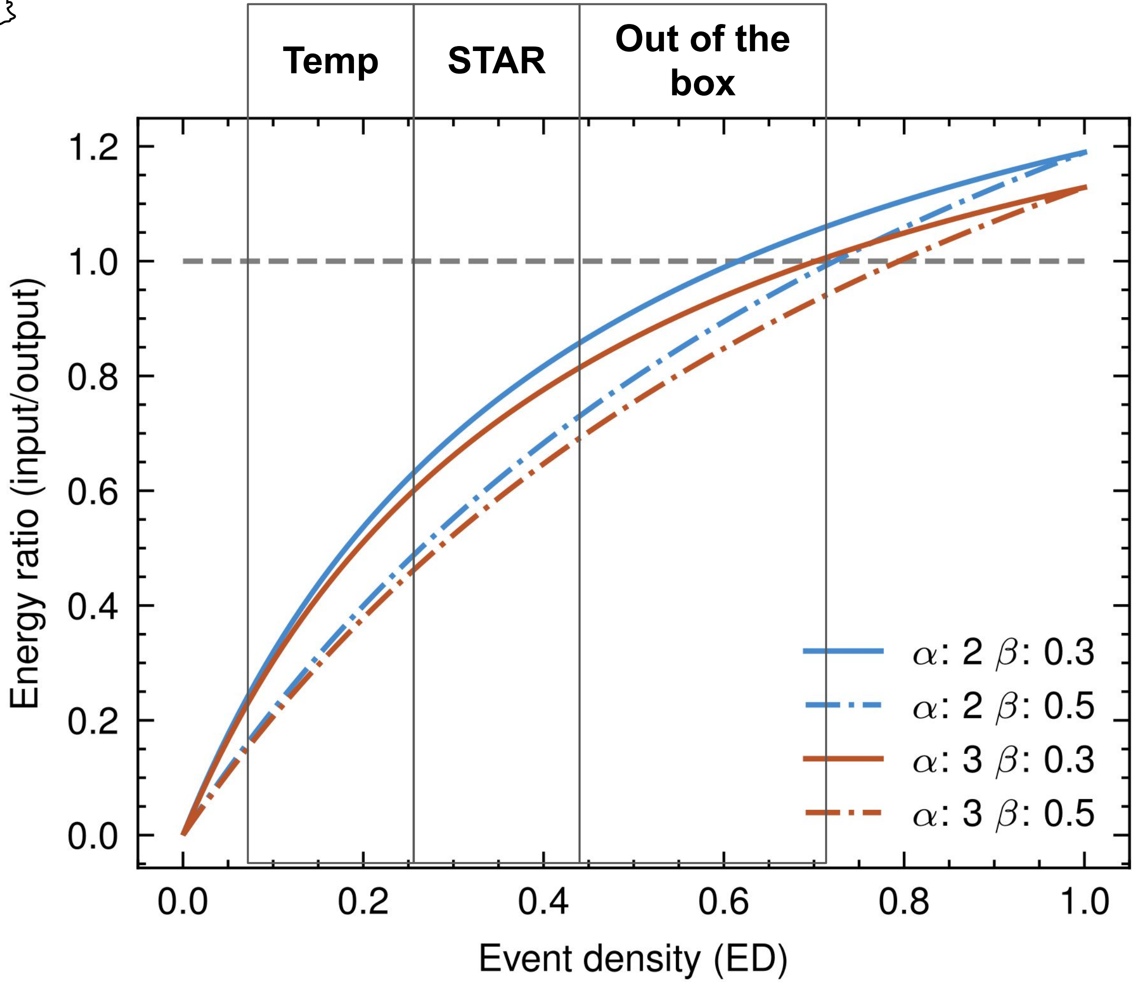
Event-driven vs standard convolution with sparsity





Alpha: logic energy/memory energy
Beta: nop logic energy/avg logic energy

Event-driven has more mem accesses, but much lower energy with sparsity!



So where are we?

- STAR: training for activation suppression helps a lot!
- Assumes FP16 datapath for input stationary; FP8 for output stationary;
- **Temporal is just too expensive: 5x more memory required!!!**
    - **Same problem as neuromorphic computing!**

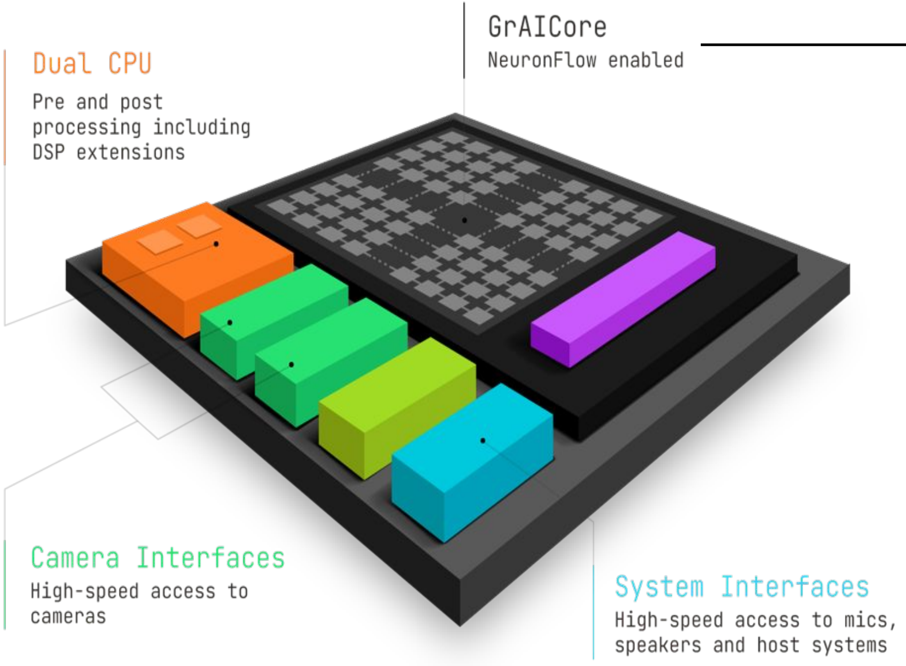
Impact of Training on Performance

- **Weight pruning** lowers number of computations and memory requirements
  - Enables larger networks
  - Small power reduction due to loading fewer weights.
- **Weight quantization** sames as Weight Pruning.

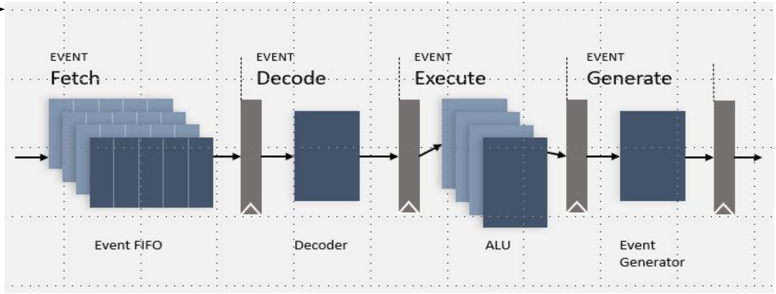
- **Activation suppression** yields large energy and performance benefits
  - Event-driven: suppressed event means skipping all computation for that event…
  - Particularly efficient with RELU activation
    - 2x event reduction out of the box
  - 2x with training to increase activation sparsity

- **Temporal sparsity**
  - More Activation Suppression,
  - Costs memory for storing inter-frame states (out of the box ~5x for mobile CNNs)
    - >2x energy and performance gains possible on top of activation sparsity
      - Depends on input data dynamicity

Brain-Inspired Neural Processor on the Edge (GrAI-VIP)



Zoomed-in view

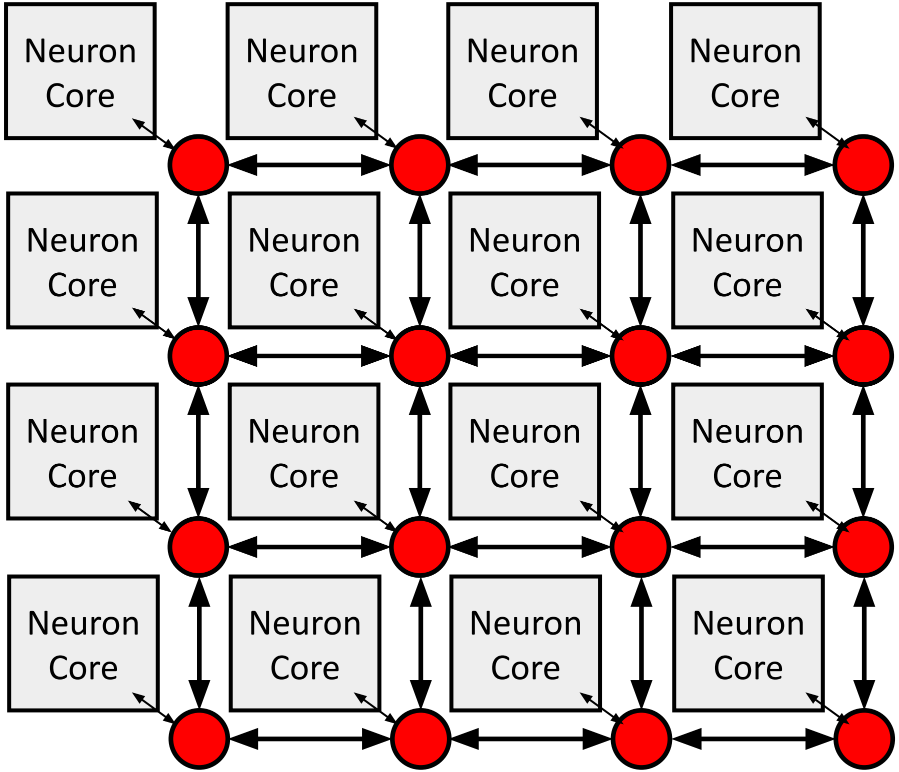


Event-Driven Neural Network Accelerator– GrAI-VIP

- 12-nm tapped-out chip
- 144 SIMD-4 cores @ 650MHz
- 12x12 grid of event-driven cores
- 256 KB on-chip memory per core
- FP16 algebra, with floating weight quantization at 8, 4, 2b

NeuronFlow Array

- Homogeneous array;
- Local memories per core:
  - Local weight and neuron state storage;
  - Near memory computation;
- Packet-switched NoC, with torus topology;
- Sparse computation, event-driven schedule:
  - Avoids bulk data movement;
  - In-place state updates;
- Dataflow: input data arrival triggers execution
- No (fast) external memory interface
- GrAICore 4.2: 12x12 cores
- GrAI 4.2:
  - FP16 2xSIMD-4 datapath



MEET

Towards MEmory-Efficient Temporal Deep Neural Networks
(Accepted at CVPR'25)

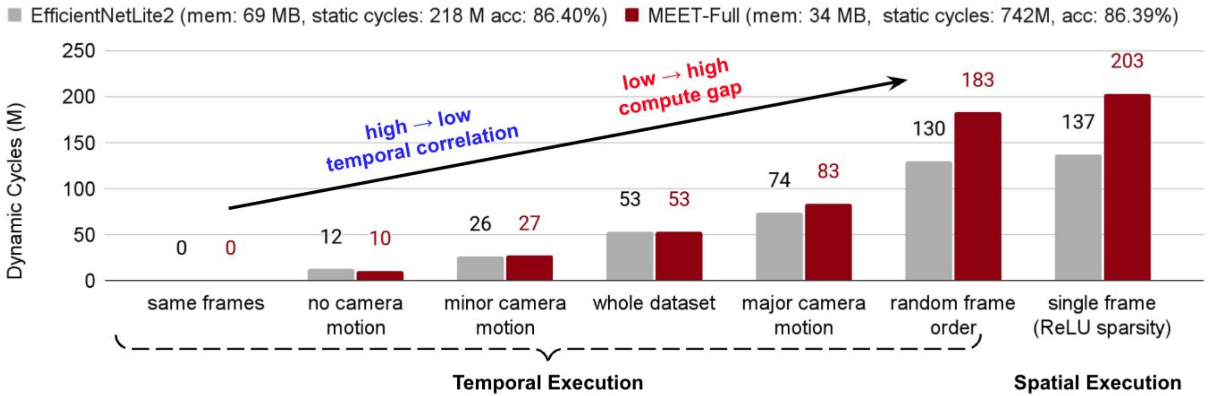
MEET:Towards **ME**mory-**E**fficient **T**emporal Neural Networks

Problem:

- Neuronflow can execute **Temporal Neural Networks (TNNs)**, to exploit temporal redundancy
- Results in **dramatic reduction** in computation: **2.5x** to **>10x;**
- **But** the **high memory cost of TNNs (5x)** remains - needs to store full feature maps.
    - This prevents TNNs from **meeting** the ***on-chip memory constraints***.

Idea:

Reduce state memory costs by increasing weight memory costs.



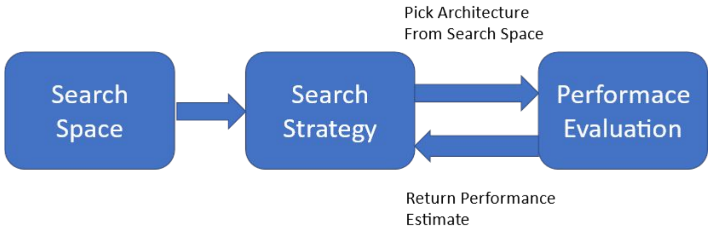
Network Architecture Search

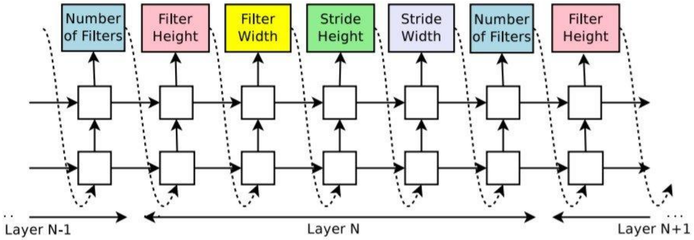
Automatically find a good network for your dataset.

- Move from network-centric to data-centric specification

NAS for event-based processors

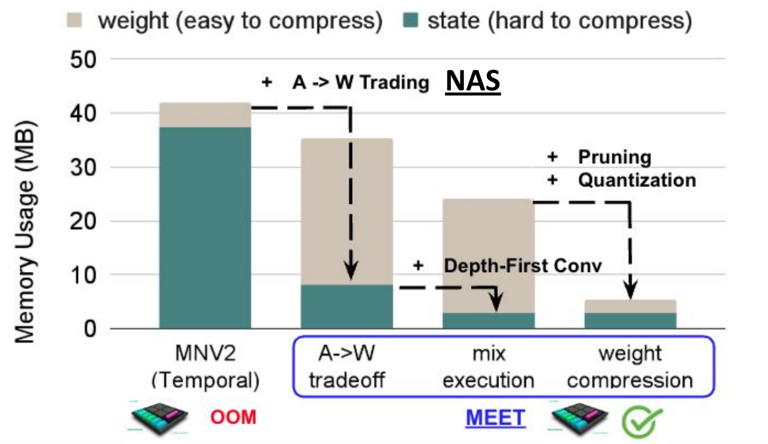
- Add processor/mapping KPIs to evaluation.
- Redesign NNs to fit our hardware.





MEET:Towards MEmory-Efficient Temporal Neural Networks

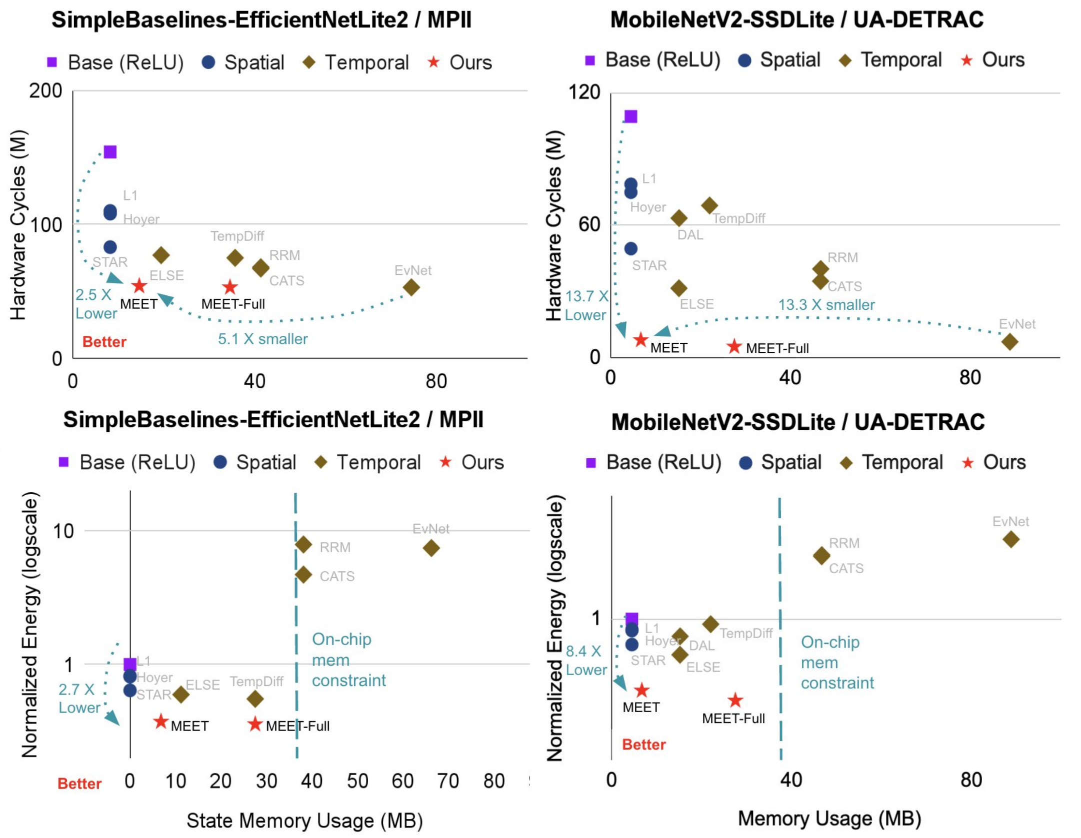
Method Overview:

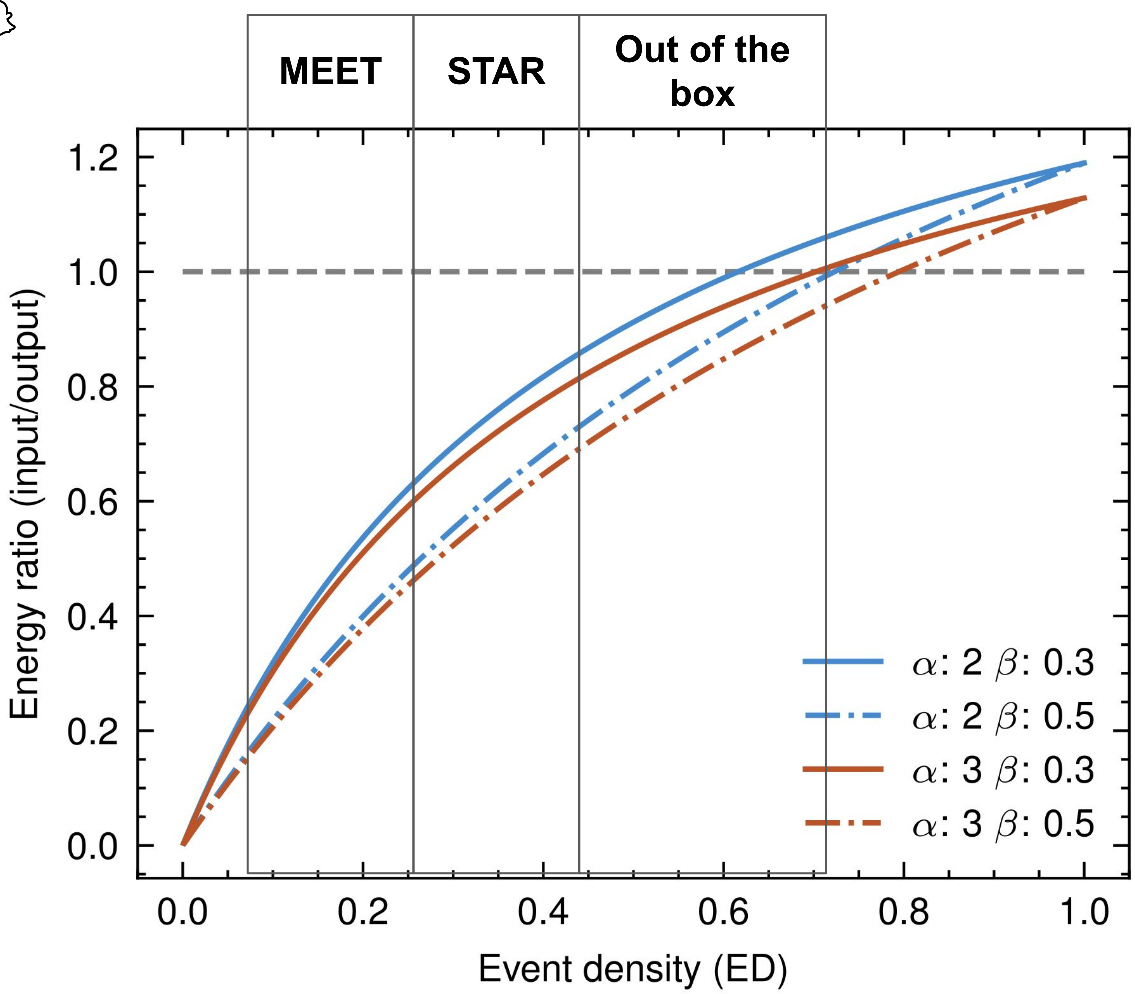


Conclusion:

- External memory cost negates energy savings from sparse compute.
- MEET significantly reduces memory cost wrt SOTA TNNs while maintaining accuracy and efficiency.

Experimental Results:





So where are we now?

- MEET: “cheap” temporal sparsity requires NAS
- Assumes FP16 datapath for input stationary; FP8 for output stationary;

Conclusions

- Event based processors exploit sparsity to reduce load
    - low latency (~1ms), very low power (<100mW @60 fps).
    - event-based convolutions efficiently exploit activation sparsity
- Optimization training is essential
    - ~ 2x performance boost for simple activation sparsity
    - > 2.5x - 5x for temporal sparsity
- Popular networks are not designed for event based processors
    - What if we design networks for event based processors?
        - MEET applies NAS to reduce penalties of temporal execution
        - From 5x Mem to 1.2x Mem!!

Thank you!

omoreira@snap.com