tinyML® Research Symposium

Enabling Ultra-low Power Machine Learning at the Edge

April 22, 2024

www.tinyML.org

Long talk at TinyML Research Symposium, April 22, 2024

CiMNet: Towards Joint Optimization for DNN Architecture and Configuration for Compute-In-Memory Hardware

Souvik Kundu[*], Anthony Sarah[*], Vinay Joshi[#], Om J Omer[#], Sree Subramoney[#]

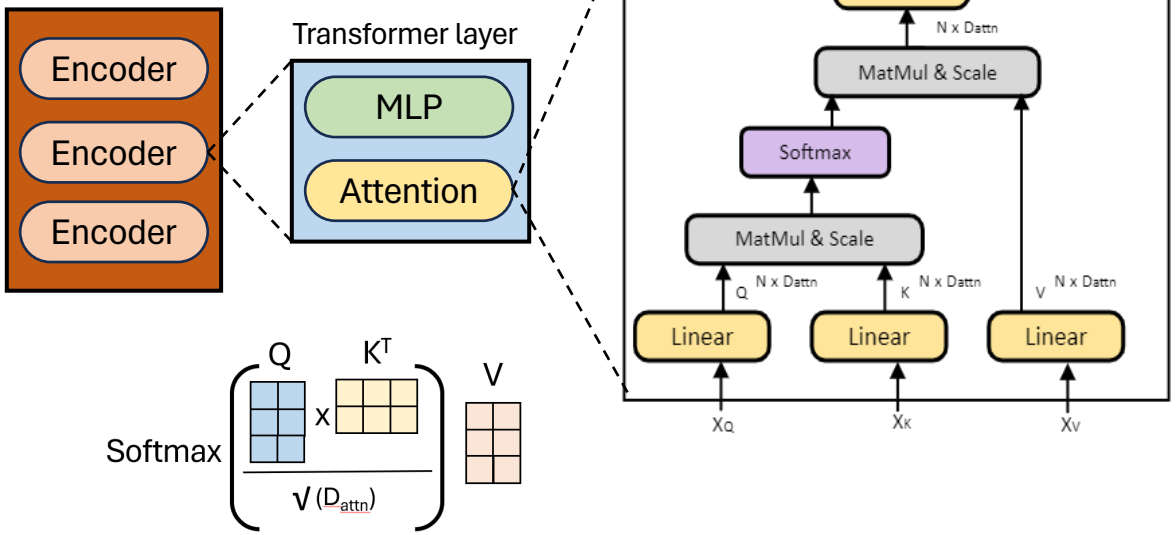
[*]Intel Labs, San Diego, USA

[#]Intel Labs, Bangalore, India

Presenter link

Paper link

The Rise of Large Foundation Models



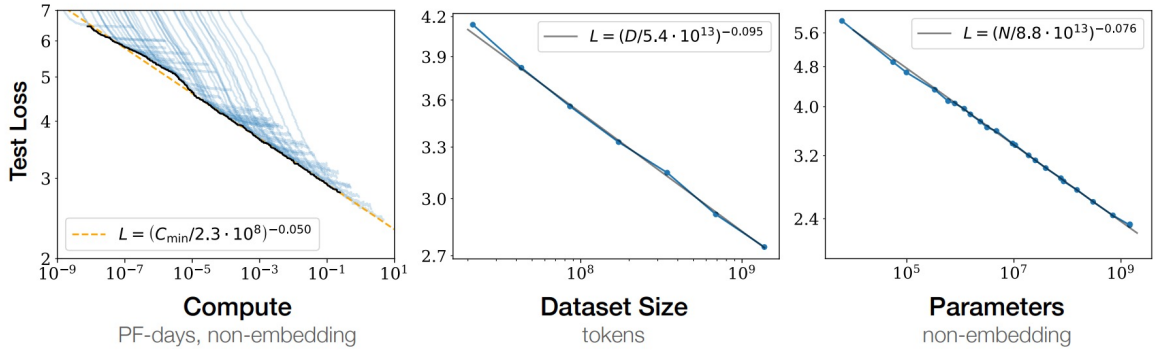
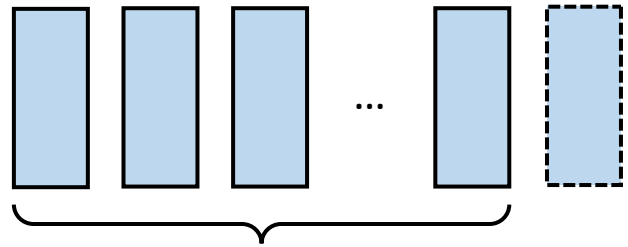


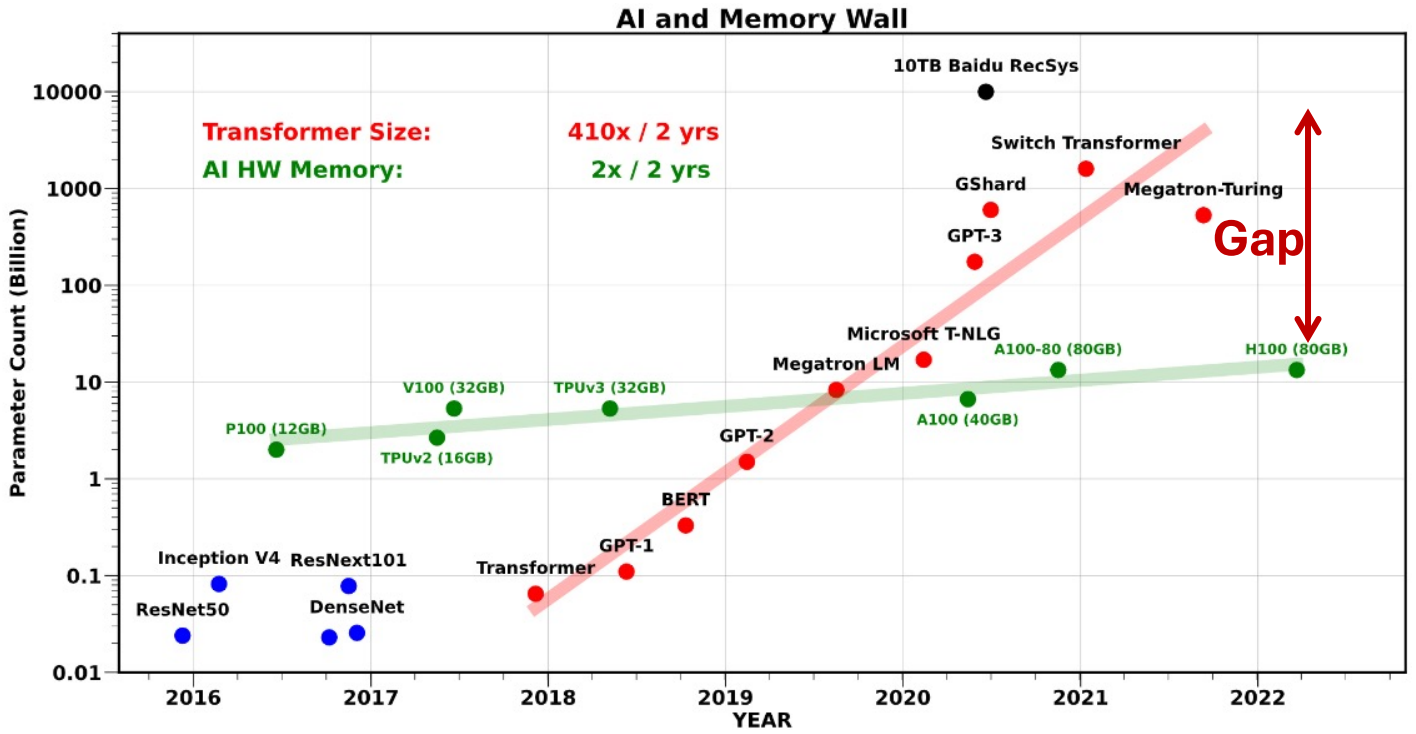
Photo Courtesy: Scaling for Natural Language Models



Increasing number of stacked up layers .. has increasing generalization ability

Recent advances in foundation model design and their promising scaling laws have triggered the rise of large DNN models

The AI Era: and Memory Wall Problem



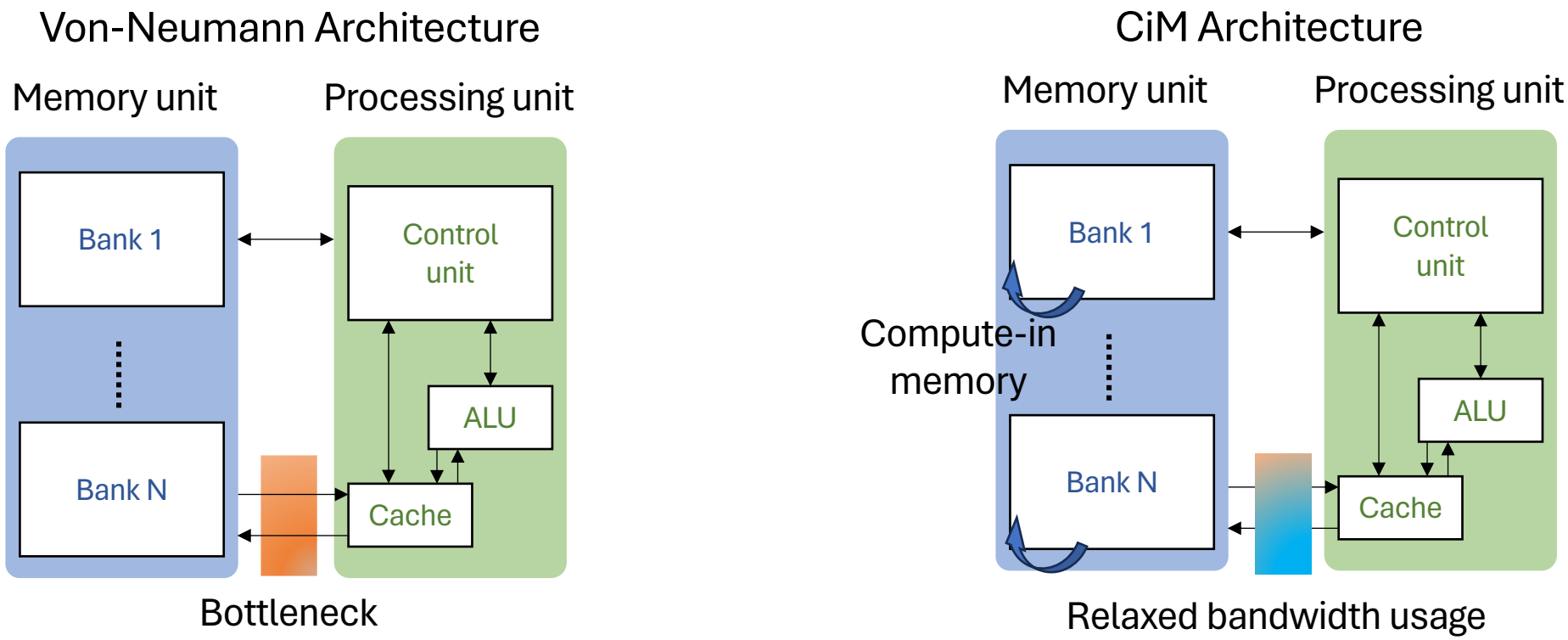
The memory bandwidth limitation has intensified the memory wall problem impacting latency, throughput, and energy

Rethinking hardware!

Photo Courtesy: https://medium.com/riselab/ai-and-memory-wall-2cb4265cb0b8

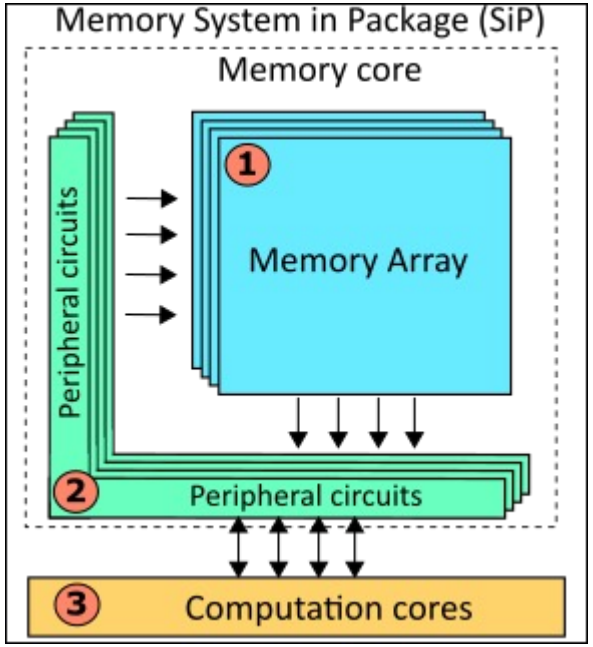
With the rise of large model use cases and with the slow down of Moore's law, the memory wall problem has intensified

Compute In Memory (CiM): A Potential Alternative



Compute-in memory alleviates bandwidth and on-chip interconnect limitations faced by Von Neumann architecture.

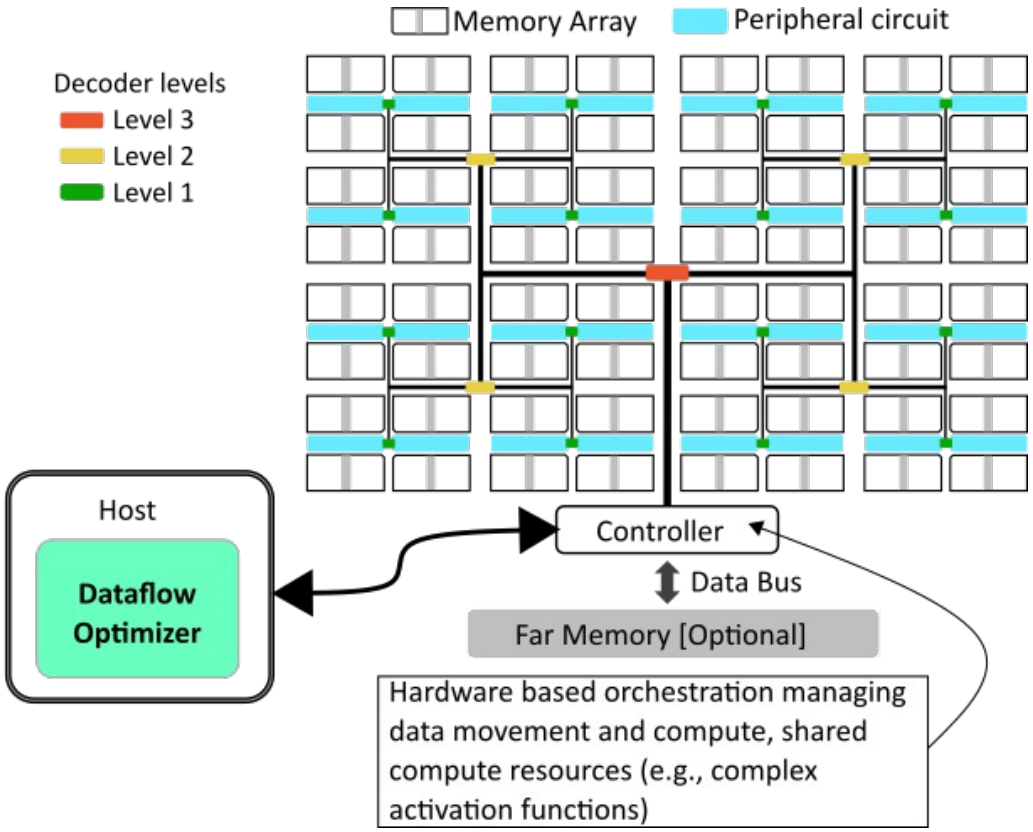
Compute Placement Opportunities



1. Memory devices enable in-situ computation of the data stored in the memory.
   Pro: Opportunity for O(1) MAC computation.
   Con: Requires complex changes to memory array and high noise sensitivity.
2. Peripheral circuit is enabled with a compute unit to perform MAC operation.
   Pro: Compatible with existing memory arrays and corresponding operations.
   Con: Small modifications are needed to peripheral circuits.
3. Compute cores enabled on the same die as that of memory.
   Pro: Memory core and its operations are unaltered.
   Con: Inability to achieve high parallelism as that of position (2).

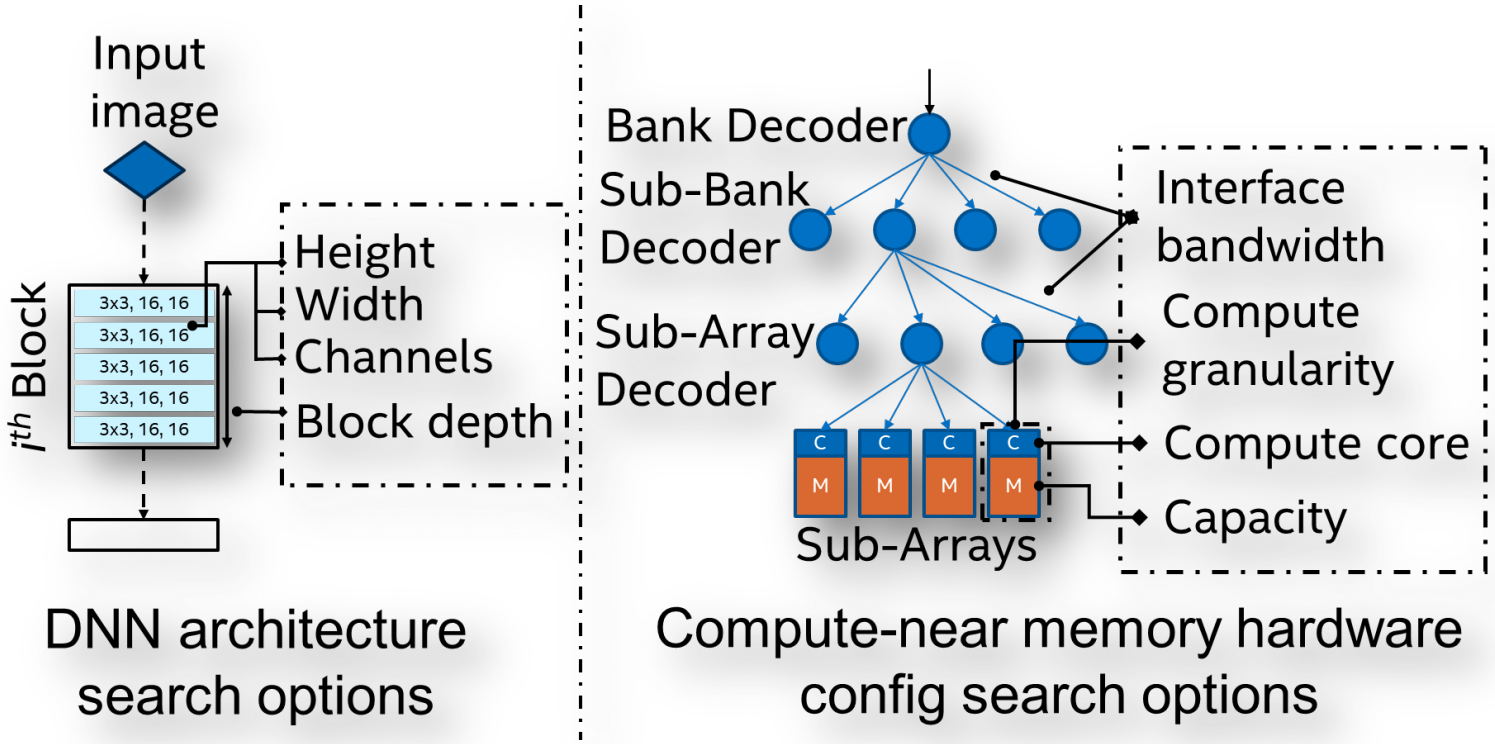
Compute-in memory with MAC compute performed in peripheral circuits offers more benefits at a small cost.

CiM Implementation



1. Memory arrays communicate via multiple levels of memory decoders.
2. Controller orchestrates data movement to (and from) host, compute, and shared compute resources.
3. Host performs upfront offline dataflow optimization for optimal execution of a DNN layer.

Affinity of Hardware Configuration: To Model



Hardware config params have a high affinity subnet arch thus impacting execution performance.

Affinity of Hardware Configuration: To Model

- Low arithmetic intensity workload requires high bandwidth for better performance.
  - Simply offering high bandwidth for execution leads to expensive implementation.
  - Also, one must allocate optimal number of cores to achieve better compute utilization.
- Arbitrary choice of hardware config **does not** guarantee optimal execution.

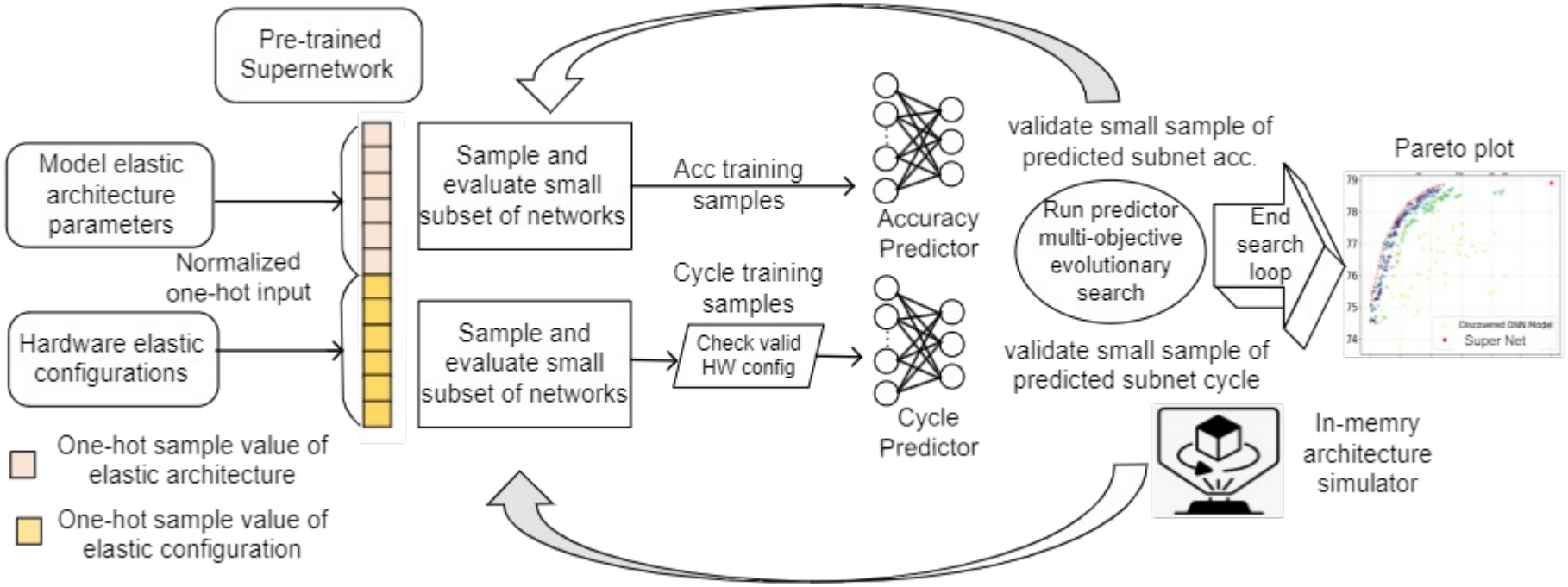
Due to the high affinity between architecture and config params, joint optimization is inevitable.

Hardware Design Space Optimization

Parameter	Description
$DRAM_{bw}$	Bandwidth offered by DRAM.
$L2_{bw}$	Bandwidth offered by level 2 decoder in the hierarchy.
$L1_{bw}$	Bandwidth offered by level 1 decoder in the hierarchy.
$L1_{num\_child}$	Number of level 1 decoders in the hierarchy.
MA_{bw}	Bandwidth offered by memory array.
MA_{num_child}	Capacity of the memory array.
MA_{mem_size}	Number of memory array nodes in our CiM architecture per L1 decoder.
$MA_{comp\_per\_core}$	Throughput of a compute core associated with memory array.

Compute-in memory architecture offers a large space of parameters that must be jointly optimized with subnet architecture.

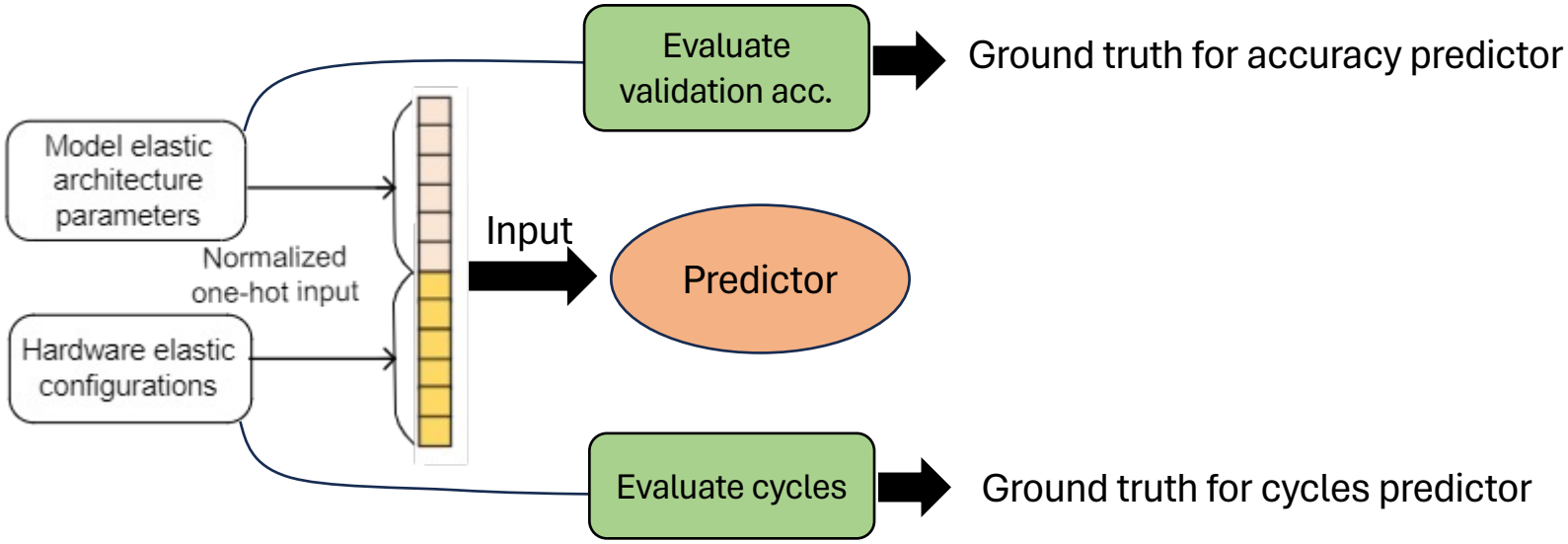
CiMNet: Framework Overview



(a) Illustration of the proposed joint architecture-hardware configuration search framework. (b) Illustration of different possible positions for the placement of a dedicated and specialized compute unit to perform MAC operations.

CiMNet: Preparing the Predictor Input/ Output

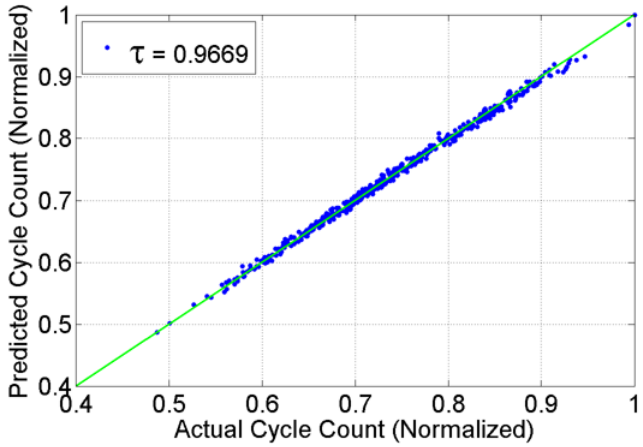
- We select a set of model elastic parameters and hardware elastic configurations
- We then one hot-encode each set and concatenate them to create the input set for the predictor training
- We use simple predictor like support vector regressor (SVR) as the predictor



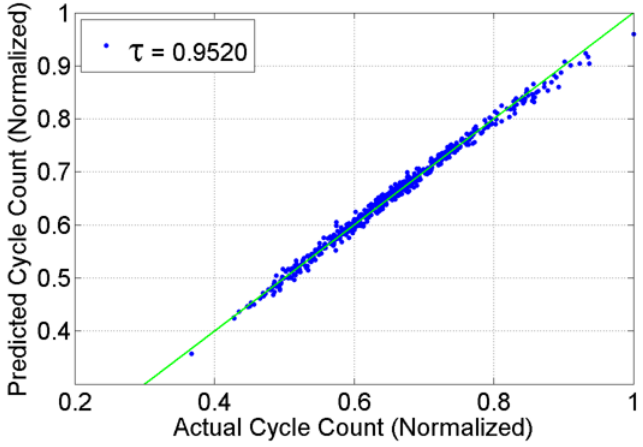
CiMNet: Predictor Training

- We use one-hot encoded vectors of the model architecture elastic parameters and hardware elastic configurations to train accuracy and cycle count predictors.
- We compute true accuracy and true cycle counts via simulation for a small set of sub-networks and hardware configurations.
- The true accuracies and cycle counts are used to train ridge and support vector regressors to predict accuracy and cycle count in the combined search space.

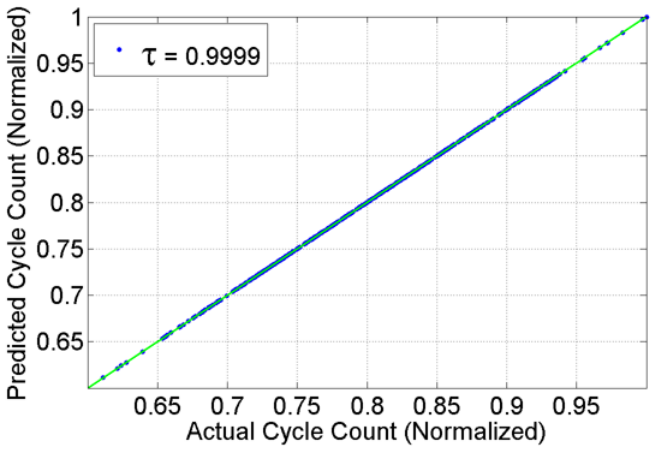
MobileNetV3



ResNet-50

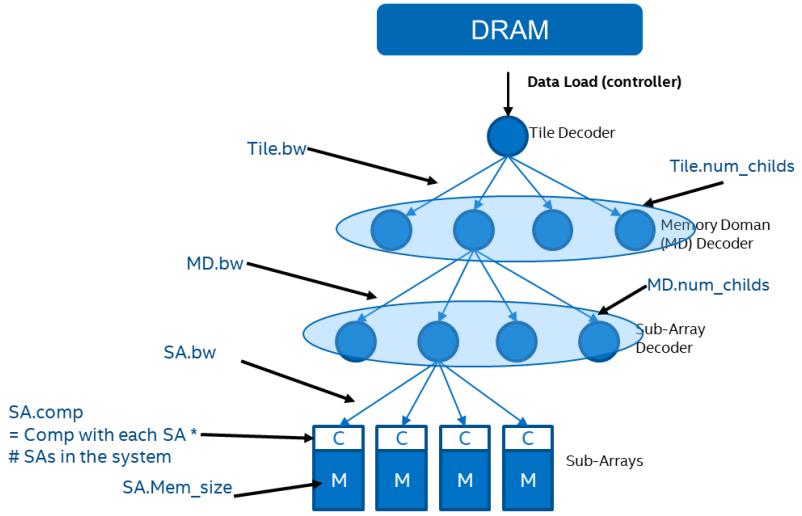


ViT-B

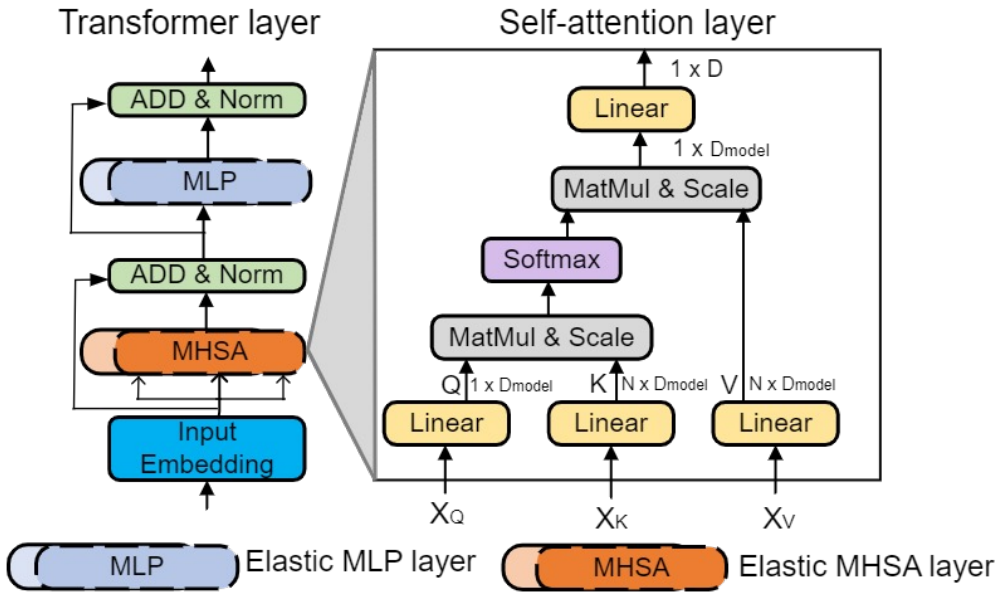


Correlation and Kendall τ coefficient between actual and predicted values after training the predictor with 1000 examples. Green lines show the ideal correlation.

CiMNet: Configuration and Architecture Elasticity

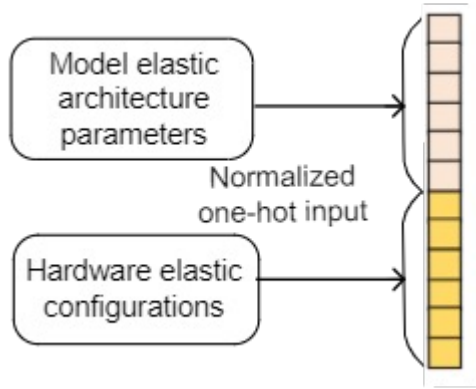


Hierarchical hardware configuration



Model architecture and elasticity

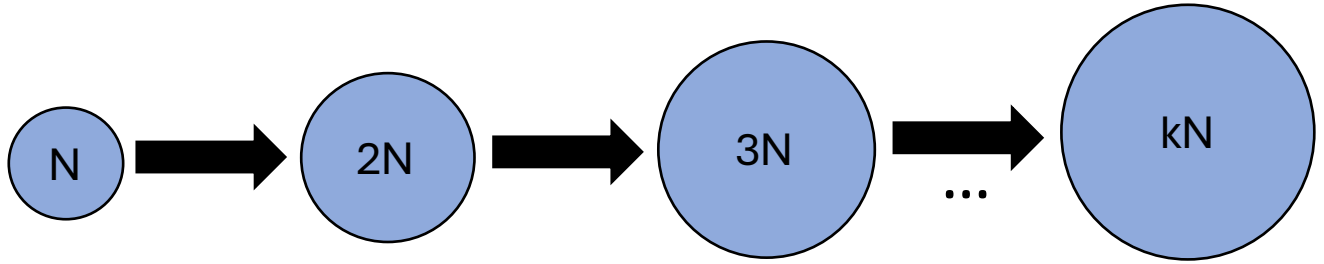
- We select eight elastic hardware configurations
- We select different model elastic parameters for CNNs and transformers. For CNNs we use kernel size, channel width and layer depth, for transformers we use MHSA head numbers, model intermediate dim, number of layers as elastic variables.



CiMNet: Iterative Optimization via NSGAII

- We use NSGAII as the multi-objective optimizer
- During the optimization we leverage the predictor to expedite the accuracy and cycle evaluations of a large corpus of samples
- At the end of each optimization iteration, we take the top N predicted samples, to evaluate their corresponding true accuracy and cycles

- Note: We use a cycle accurate simulator to evaluate the CiM hardware cycles



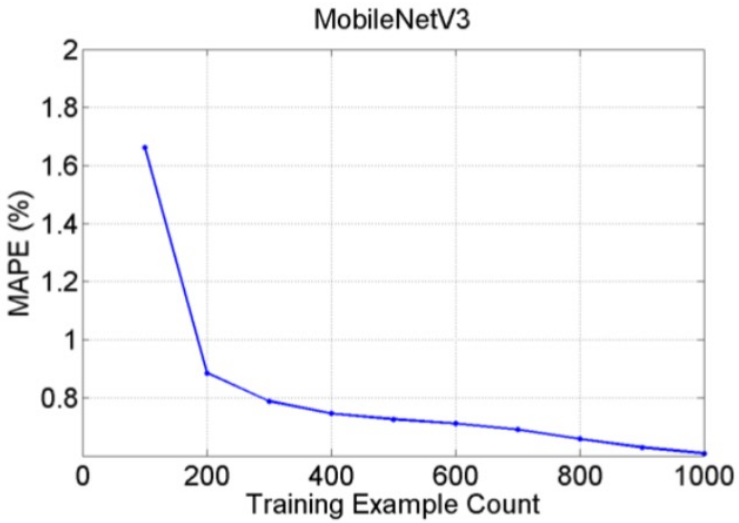
Gradually increasing sample size of the predictor over k optimization iterations

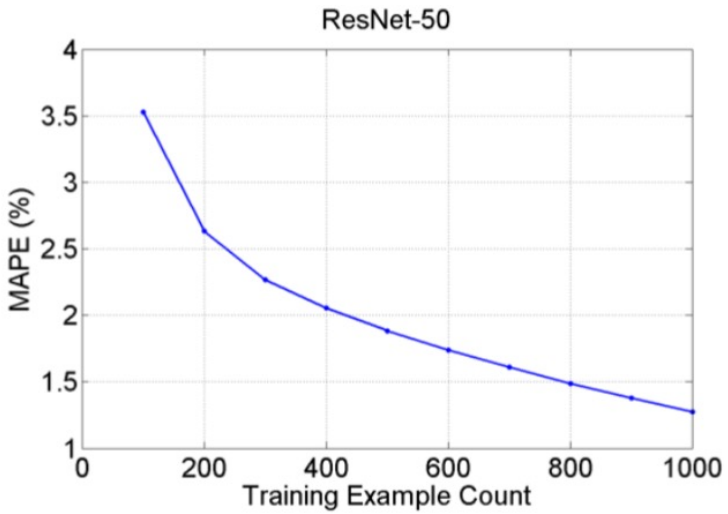
CiMNet: Evaluating the Predictor

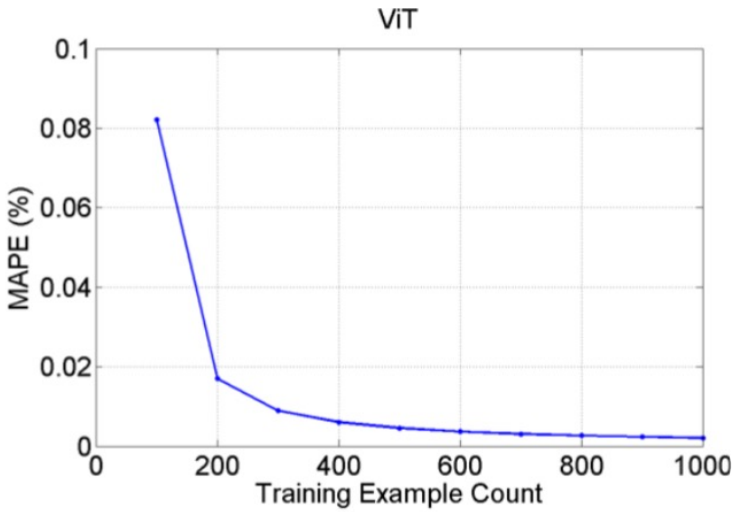
Network	Elastic Arch. Static Config.	Static Arch. Elastic Config.	Elastic Arch. Elastic Config.
MobileNetV3	0.97	0.97	0.83
ResNet-50	0.95	0.90	0.81
ViT-B	1.00	0.90	0.78

Kendall rank correlation coefficients of different search spaces for sub-networks derived from MobileNet v3, ResNet50, and ViT-B models show that the predictor has high correlation coefficient across different search configurations

CiMNet: Visualizing the Predictor MAPE



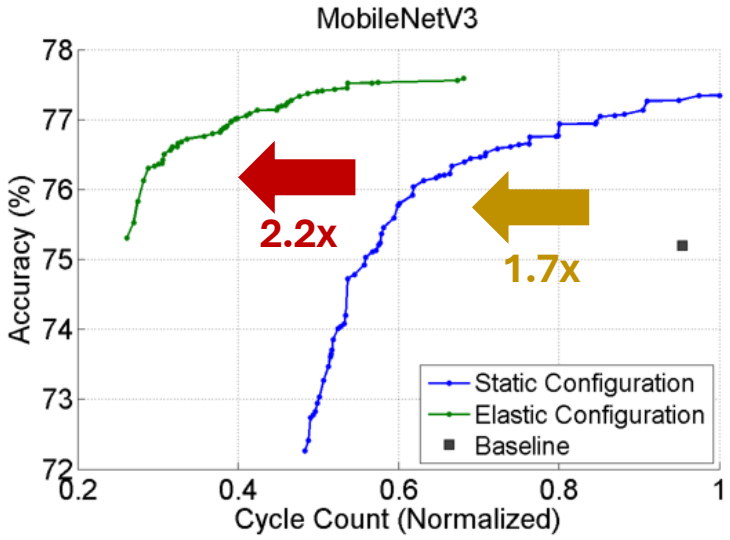


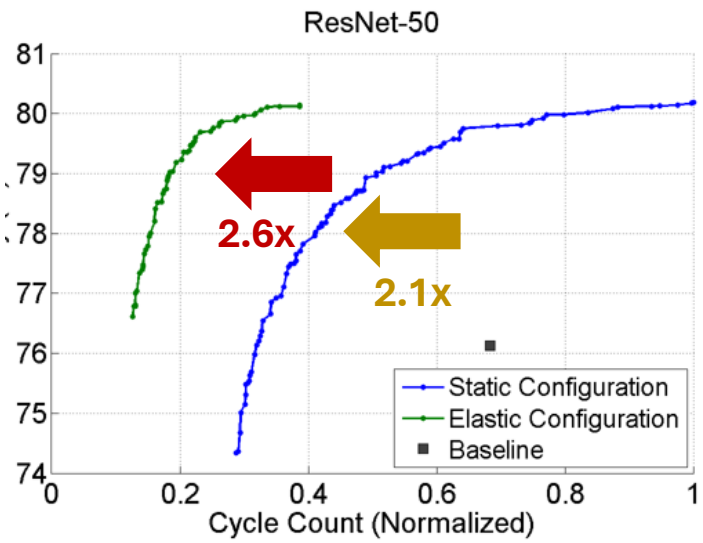


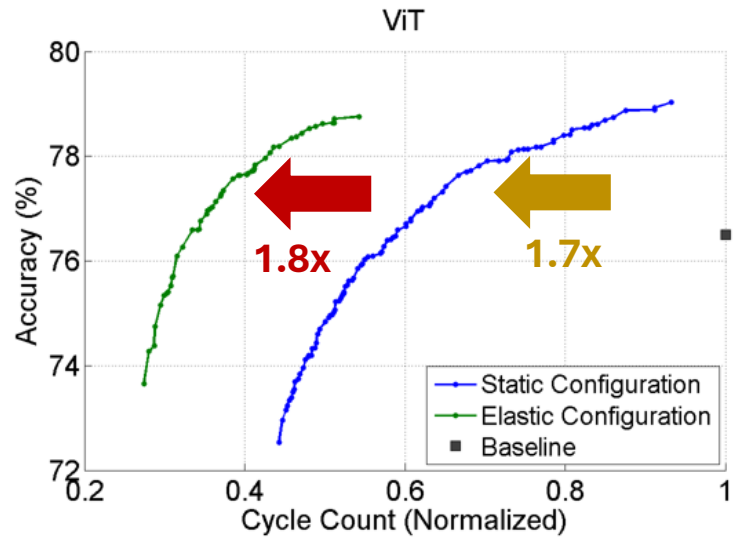
A well performing predictor can mitigate the need for any pre-calculated set of hardware metrices like cycles, latency etc.

MAPE of predictors shows the gradual improvement with the increasing training samples, demonstrating the efficacy of a predictor in predicting accuracy/cycles for joint optimization framework.

CiMNet: Pareto Analysis







With both model and hardware configuration as elastic demonstrates a better pareto optimal curve post search, this clearly highlights the benefits of proposed joint optimization

CiMNet: Analyzing the Chosen Configurations

Chosen configuration analysis for an example model: MobileNet v3

Parameter	0.125 ×	0.25 ×	0.5 ×	1.0 ×
$DRAM_{BW}$	C_s	C_{max}		C_{min}, C_{med}
$L1_{BW}$		C_s, C_{max}		C_{min}, C_{med}
$L1_{num\_child}$			C_s	$C_{min}, C_{med}, C_{max}$
MA_{BW}		C_s	C_{med}, C_{max}	C_{min}
$MA_{comp\_per\_core}$		$C_{min}, C_{med}, C_{max}$	C_s	
MA_{mem_size}			C_s, C_{med}, C_{max}	C_{min}
MA_{num_child}			$C_s, C_{min}, C_{med}, C_{max}$	
$L2_{BW}$		C_s		$C_{min}, C_{med}, C_{max}$

Hardware configuration that consumes most DRAM and L1 bandwidth results in most cycle efficient performance for MobileNet v3. This reflects the importance of these configuration variables over others.

Summary and Future Works

Summary:

- Jointly optimizing hardware configuration and subnetwork architectures can improve cycle vs accuracy pareto significantly, evaluated over different model types.

- Predictor based approach works on a dual input settings, where part of the input is subnet one-hot encoding and remaining is hardware config one hot encoded input.

Future works:

- Extending CiM based architecture in the application space for foundation model inference

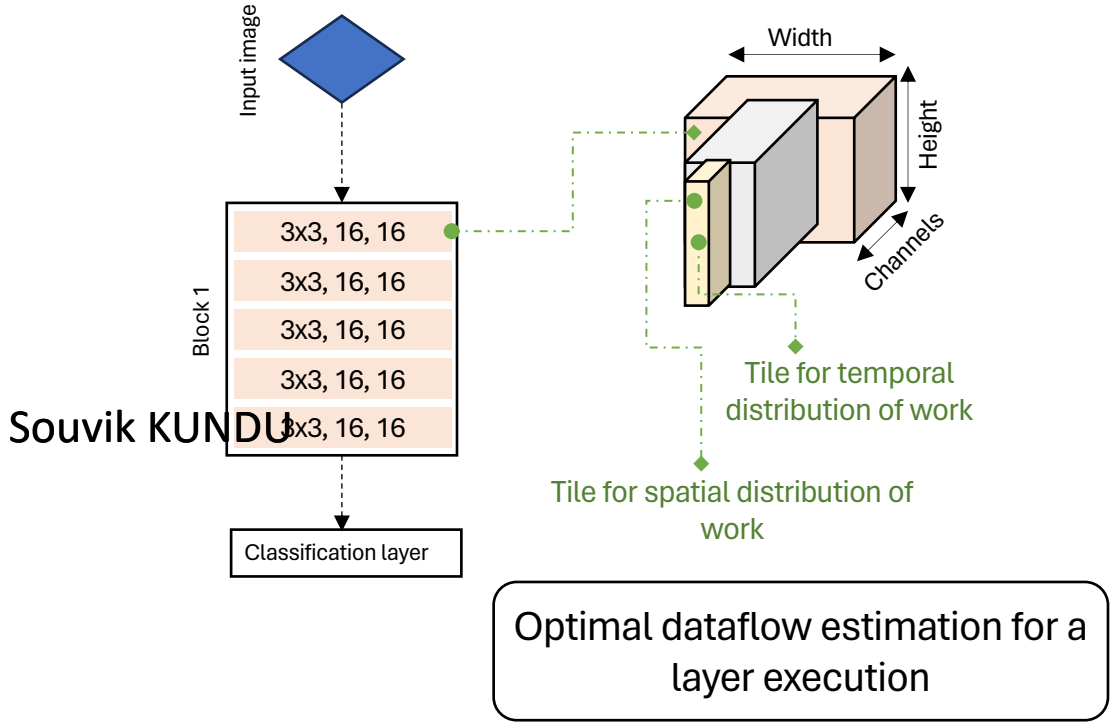
- Joint optimization across heterogenous architectures including CiM, Von-Neumann architectures.

Thank You!

Presenter link

Paper link

Supplementary on Optimal DNN Mapping on CiM



Optimal DNN layer execution requires carefully optimizing spatial and temporal distribution of work (dataflow).

Copyright Notice

www.tinyml.org