IBM IEEE CAS/EDS
AI Compute Symposium 2021
October 13-14

A Survey on the Optimization of Neural Network Accelerators for Micro-AI On-Device Inference

Arnab Neelim Mazumder
Graduate Student
University of Maryland, Baltimore County

Motivation

AI has evolved in the last few decades from being able to do small object-oriented tasks to executing large compute-intensive tasks in a matter of minutes.

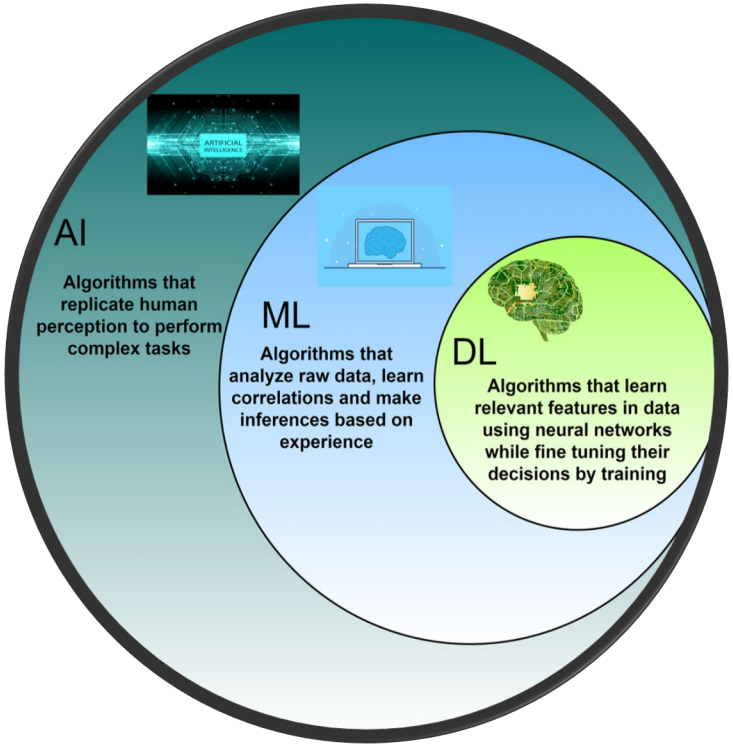
Modern-day DNN applications require more direct acceleration of DNN computation.

DNN accelerators provide a specialized hardware platform to account for the computation of the complex networks.

Specialized hardware accelerators provide efficient processing but incurs a huge overhead of device utilization and energy consumption.

One solution is to make DNN frameworks scale down to the micro-AI level

Tiny frameworks get accelerated to compensate for the stringent device constraints of latency, power consumption, and memory.



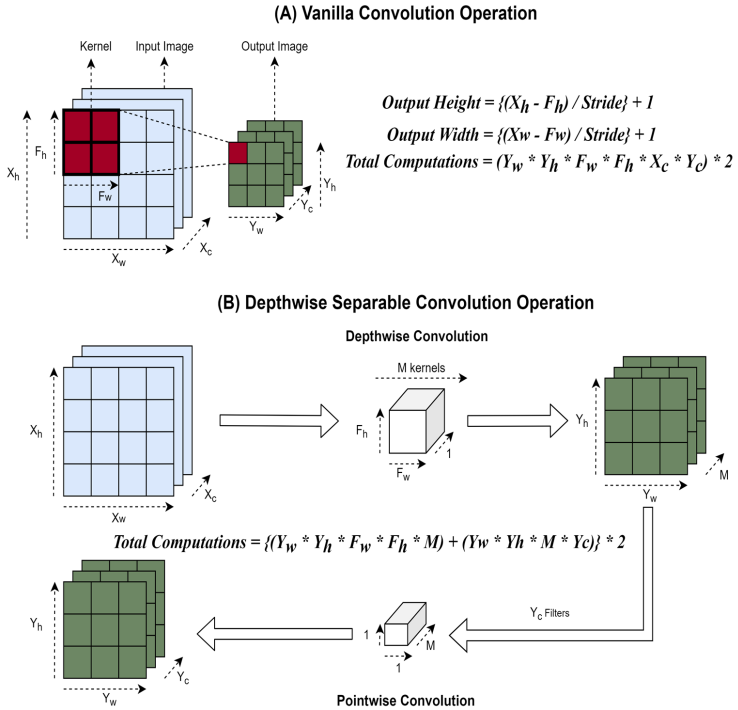
Background

CNNs in Practice

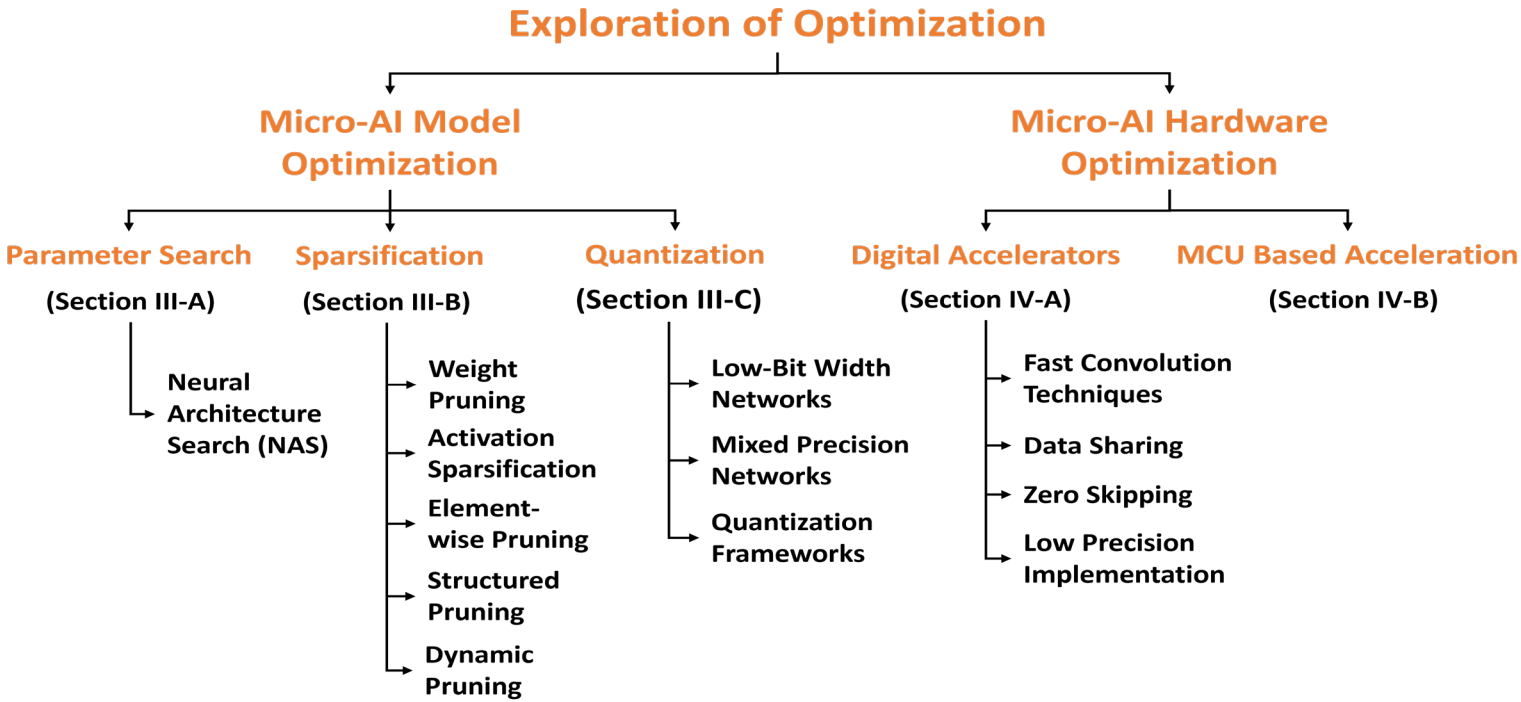
- CNNs are able to isolate spatial and temporal relevancies in an image with the help of filters.
- A kernel of a predefined shape iterates through the whole image with a predefined stride set by the user.
- Convolutional layers coupled with pooling layers make up the backbone of traditional CNN frameworks.

Why Inference Engines?

- Training requires back propagation
- Needs intermediate feature map values to stored for every layer
- Becomes extremely compute-intensive
- Requires bulky off-the-chip memories
- Non-uniform precision of data and weights is difficult to handle



Optimization Exploration

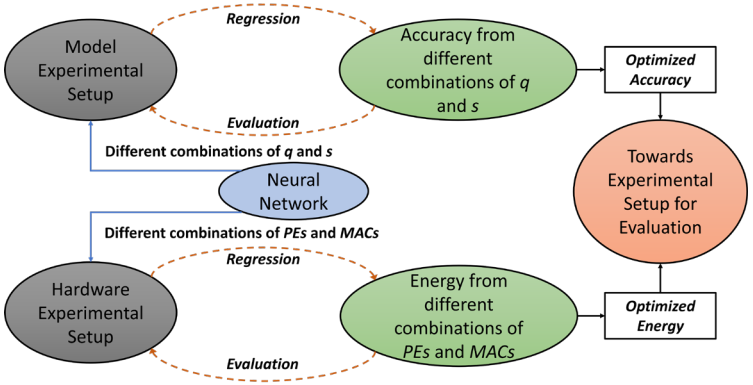


Micro-AI Model Design – Parameter Search

- In DNNs there are several variables related to each layer along with different combinations of optimizer and learning rates known as hyperparameter.
- Traditional methods of hyperparameter search include brute force algorithms such as grid search, random search and genetic algorithms.
- There have been recent developments in the domain of automated frameworks and tools with regards to neural architecture search (NAS) that allow such broad search approaches to be streamlined conveniently and promptly.

Neural Architecture Search (NAS)

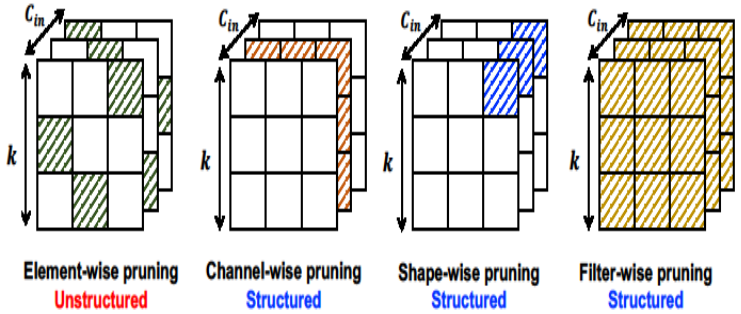
- RL based co-design NAS
  - Used as a profiling technique to map hardware power and latency performance.
- Regression based optimization NAS
  - The regression-based approach determines the optimal unknown variables (q, s or r) for a given network so that it can be streamlined for energy-efficient implementation.



Regression-based NAS

Micro-AI Model Design – Sparsification

- The hardware benefits~(e.g., energy, latency reduction) obtained from the sparse neural network could be different with various sparsification analogies.
- Weight Pruning
  - A binary mask forces certain some "unimportant" weights to be zero. The accuracy degradation caused by the pruning often requires further training to recover.
- Activation Pruning
  - Storing the intermediate activation is memory intensive. Exploitation of the activation sparsity could induce more energy and latency reduction compared to static weight pruning.
- Element-wise Pruning
  - Considers each weight as a single element and forces the unimportant parameters to zero.
- Structured Pruning
  - Eliminates the unimportant weights in a group-wise manner to allow the sparse model to have better hardware compatibility.
- Dynamic Pruning
  - Since the importance of the channels are varying for different input features, the essence of dynamic pruning is activating the computation for the salient features while ignoring the trivial characteristics.



Different Pruning Methods

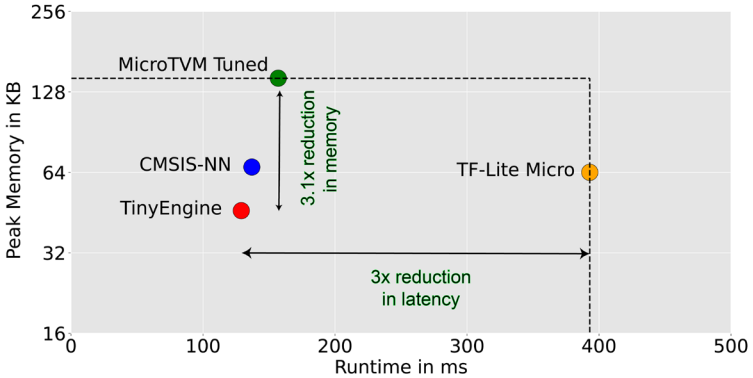
Micro-AI Model Design – Quantization

- Moving from floating-point representations to low-precision fixed integer values represented in four bits or less holds the potential to reduce the memory footprint and latency by a factor of 16x
- in fact, reductions of 4x to 8x are often realized in practice in these applications.
- Mixed Precision Networks
  - Different bit precision would be allocated to different layers based on their importance.
  - More important layers would be assigned higher precision, and comparatively less important layers would be in lower precision.
  - However, the search space for selecting this bit setting grows exponentially with the number of layers, making it challenging to implement.

- Low Precision Networks
  - Corresponds to BNNs and TNNs
  - Binarization, where the quantized values are constrained to a 1-bit representation, thereby drastically reducing the memory requirement by 32×.
  - Naive binarization and ternarization methods generally result in severe accuracy degradation.
  - Quantization Error Minimization, Improved Loss function, and Improved Training Method can act as solutions

Micro-AI Hardware Design – MCU Based Designs

- Microcontrollers are severely limited by their storage capacity to facilitate deep learning frameworks in their design. In general, microcontrollers have memory ranging from a few KB to a maximum of 1 MB.
- Popular MCU based designs include ShuffleNet, Hello Edge, Sparse, and MCUNet.
- ShuffleNet [1] reduces workloads to 40MFLOPs compared AlexNet implementation.
- Sparse [2] uses a NAS-based approach to find superior CNN architectures by considering the memory constraints of microcontrollers.
- Hello Edge [3] focuses on keyword spotting with the google speech commands dataset using depthwise separable convolutions for memory-limited microcontrollers.
- MCUNet [4] proposes a system-algorithm co-design framework to optimize the TinyNAS architecture to reduce memory efficiently and deploy ImageNet scale deep learning on microcontrollers.

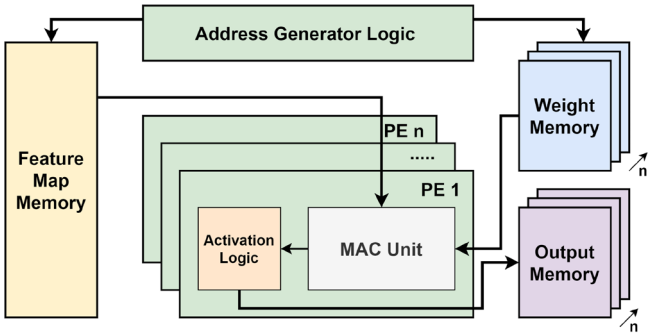


Efficacy of TinyEngine (MCUNet) compared to other frameworks

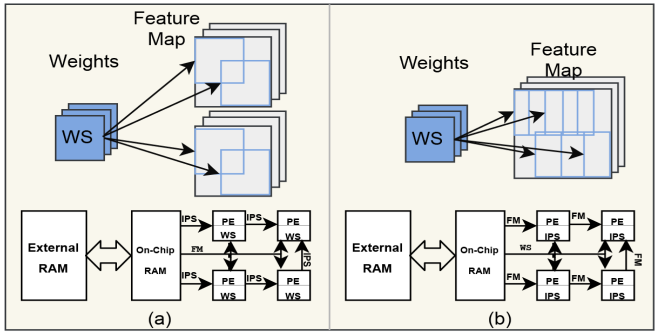
[1] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 6848–6856.
[2] I. Fedorov, R.P. Adams, M. Mattina, and P.N. Whatmough, "Sparse: Sparse architecture search for cnns on resource constrained microcontrollers, "arXiv preprint arXiv:1905.12107, 2019
[3] Y. Zhang, N. Suda, L. Lai, and V. Chandra, "Hello edge: Keyword spotting on microcontrollers, "arXiv preprint arXiv:1711.07128, 2017.
[4] J. Lin, W.M. Chen, Y. Lin, J. Cohn, C. Gan, and S. Han, "Mcunet:Tiny deep learning on iot devices, "arXiv preprint arXiv:2007.10319,2020

Micro-AI Hardware Design – Digital Accelerators

- Digital accelerators are also known as systolic architectures because the arrangement of the processing engines, mimics the rhythmical data flow system.
- **Fast Convolution**
- Alters the convolution operation to minimize the computational complexity through transform algorithms.
  - Fast Fourier Transform (FFT)
  - Winograd minimal filtering
- **Zero Skipping**
  - Mimics sparse architectures by exploiting inherent zeros in weights and activations.
- **Data Sharing**
  - Dataflow techniques increases the reuse of data read from memories lower in the hierarchy for the spatial architecture.
  - Different data sharing approaches include weight stationary, output stationary, no local reuse, row stationary and loop tiling schemes.



Systolic Array of Digital Accelerators



Exploration of Data Sharing Techniques

Conclusion

The main takeaways for a reader from this article will be the following

- Understanding of different search spaces to pinpoint the best micro-AI model configuration.
- Ability to interpret different quantization and sparsification techniques.
- And finally, the realization of the micro-AI models on resource-constrained hardware and different design considerations associated with it.