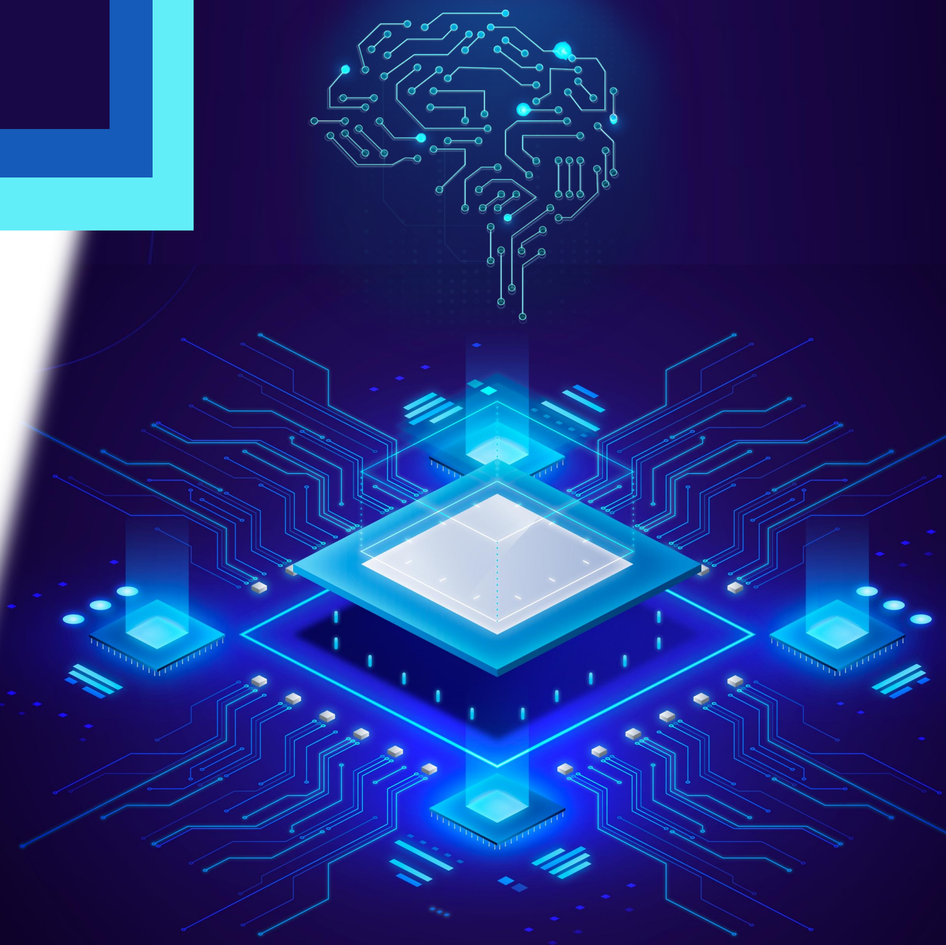
Neural architecture search for in-memory computing-based deep learning accelerators

Olga Krestinskaya
King Abdullah University of Science and Technology, 23955, Saudi Arabia

Based on:

Olga Krestinskaya, Mohammed E. Fouda, Hadjer Benmeziane, Kaoutar El Maghraoui, Abu Sebastian, Wei D. Lu, Mario Lanza, Hai Li, Fadi Kurdahi, Suhaib A. Fahmy, Ahmed Eltawil, and Khaled N. Salama.

Nature Reviews Electrical Engineering (2024): 1-17.



MiM Webinar, November 18, 2024

Agenda

Introduction, motivation and software-hardware co-design

Motivation to improve hardware efficiency and challenges in selecting optimum design

Hardware-aware Neural Architecture Search (HW-NAS)

HW-NAS methods, algorithms, hardware cost estimation methods, HW-NAS frameworks for IMC

Future directions, open challenges and final thoughts

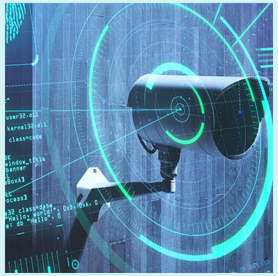
Roadmap of HW-NAS for IMC, open issues, HW-NAS and other optimization techniques, summary

Rapid AI development and Increasing Neural Network Complexity

"AI everywhere"

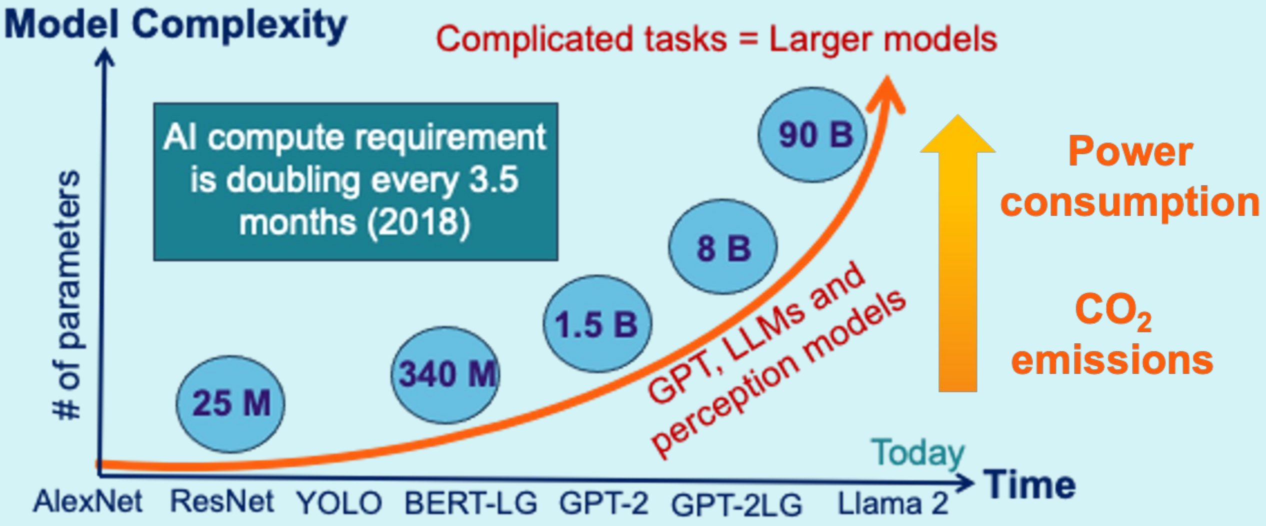

Vision and language processing


Self-driving vehicles

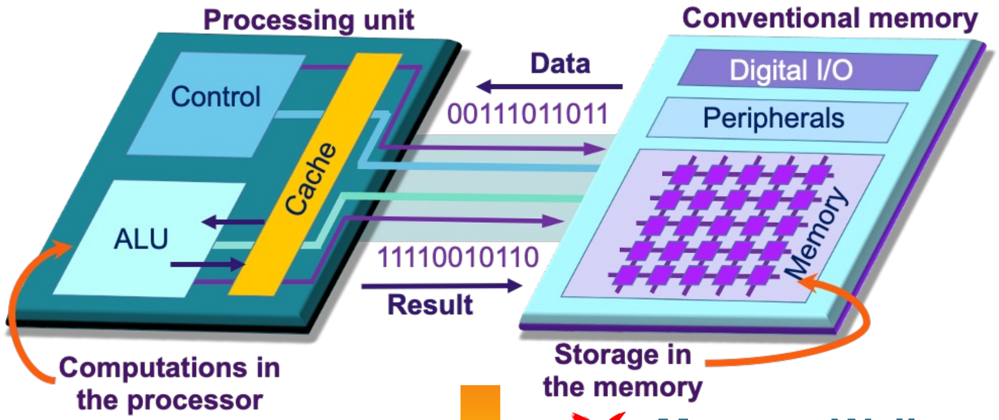

Security applications


Smart cities

Increasing complexity of neural networks

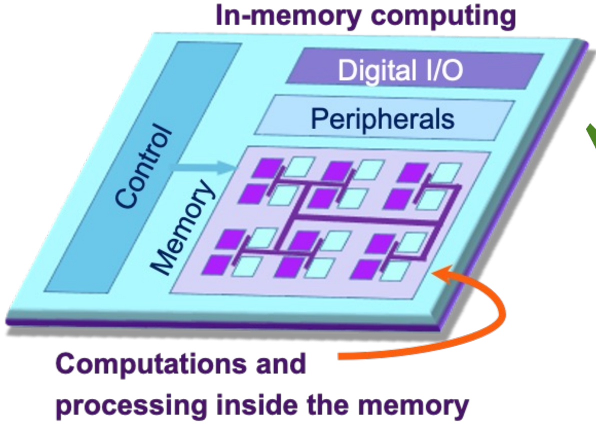


Traditional von Neumann Architecture:



- ✗ Data transfer Issues
- ✗ Off-chip Memory Bottleneck
- ✗ Memory Wall
- ✗ Constrained Power Efficiency

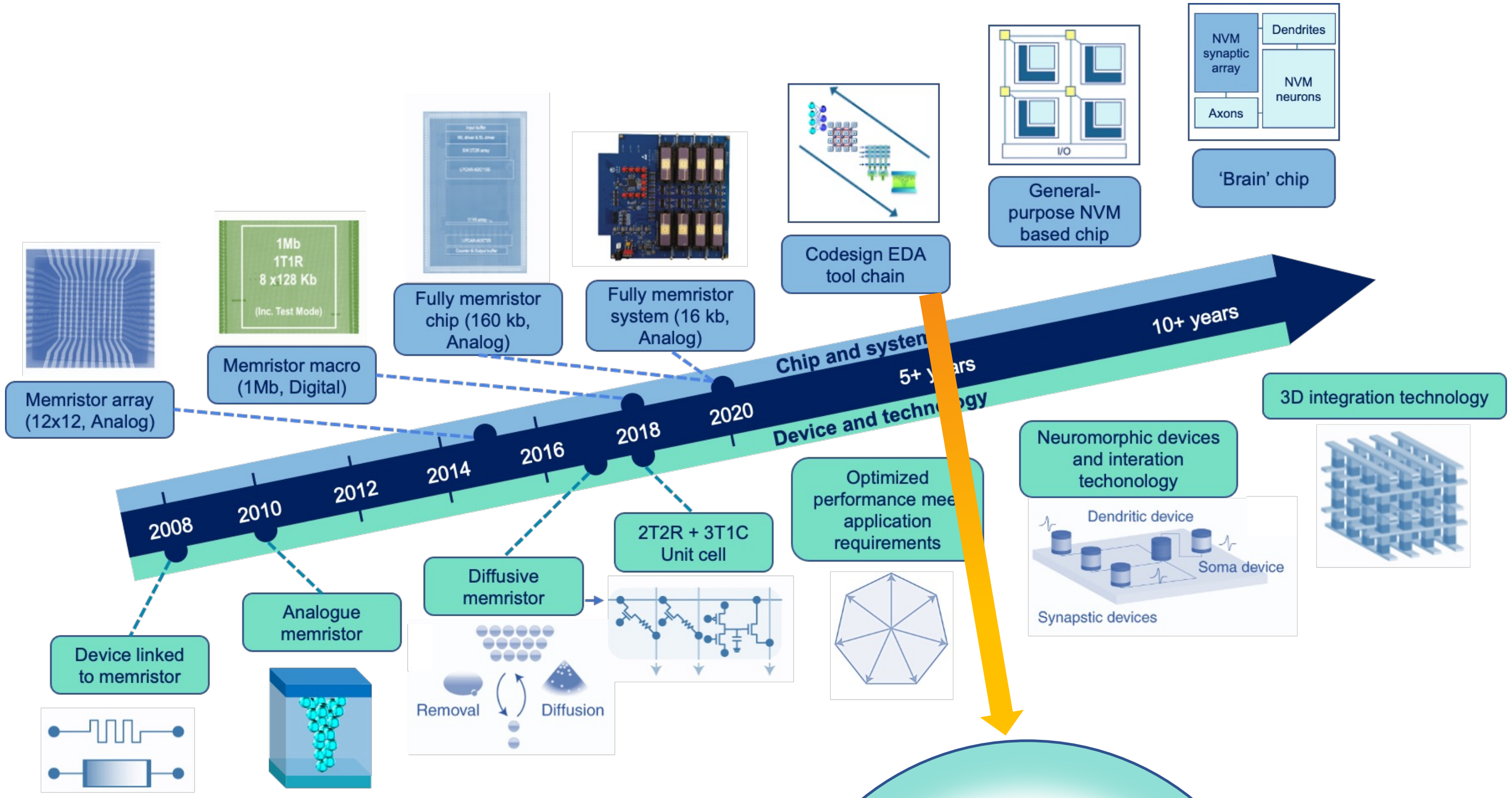
In-memory computing accelerator:

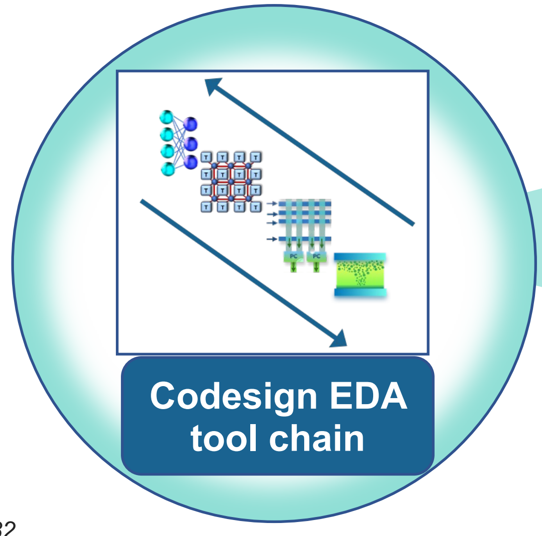


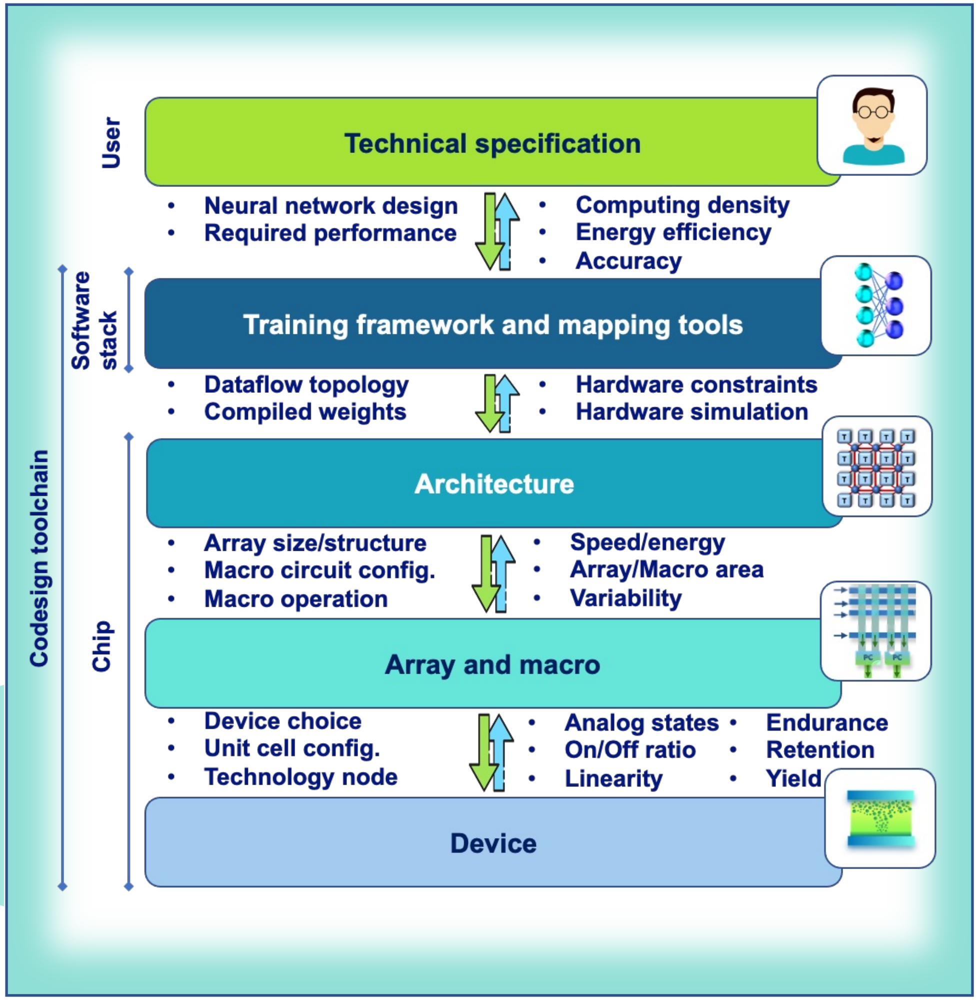
✓ Reduced Data Transmission between Memory and Processor

Hao, K. (2019). The computing power needed to train AI is now rising seven times faster than ever before. *MIT Technology Review*.

In-memory Computing Roadmap and Software-Hardware Co-design

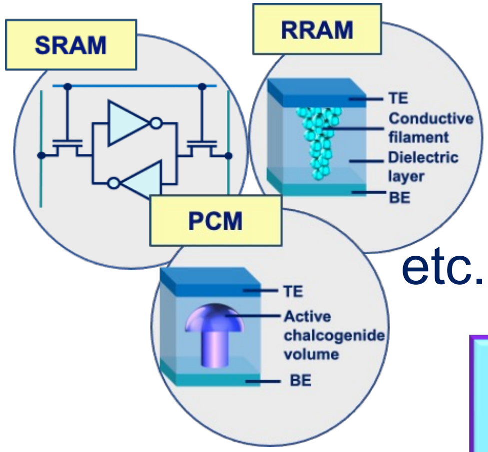






Adapted from *Zhang, W., et al. (2020). Nature electronics, 3(7), 371-382.*

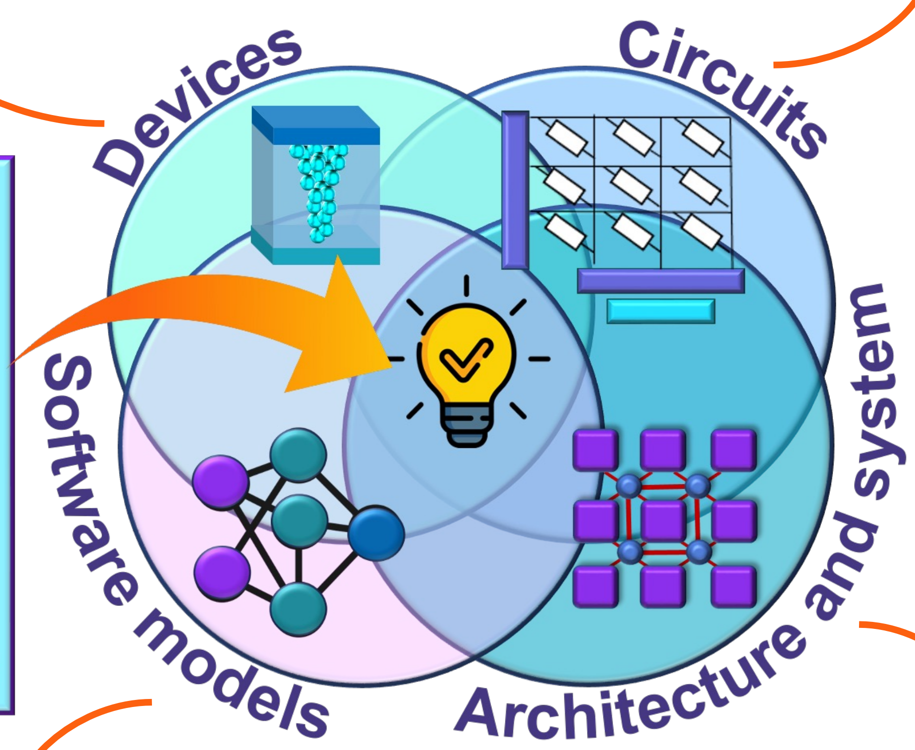
Software-Hardware Co-design for IMC Accelerators

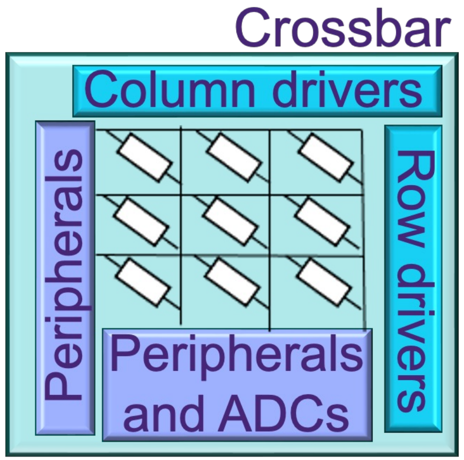


etc.

e.g. number of bits per cell

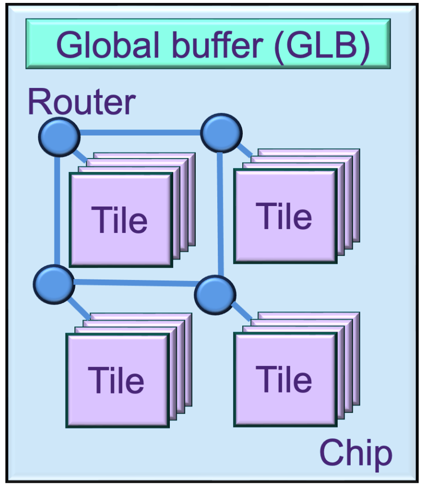
Optimum software-hardware co-design solution for AI applications



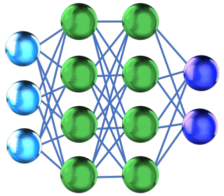


Size, ADCs per crossbar, etc.

Number of tiles, etc.



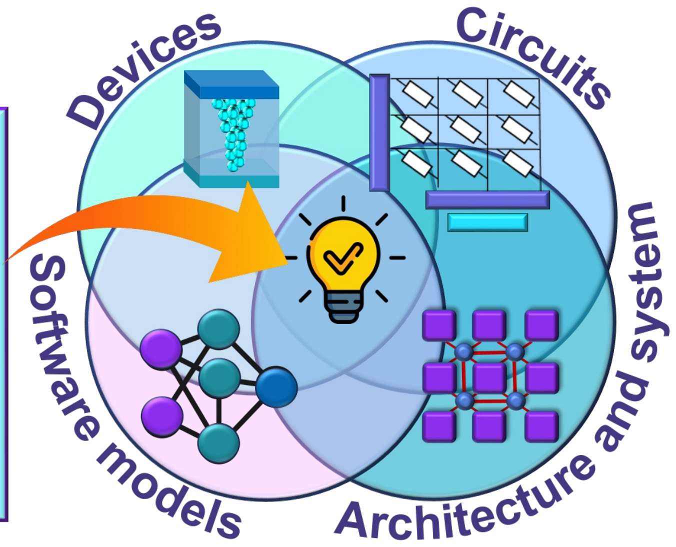
Neural network blocks, connections, layer sizes, etc.

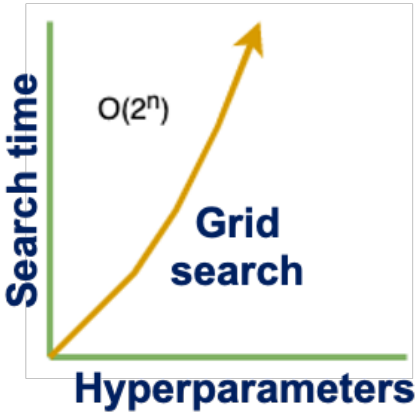


How to Select the Optimum Design?

Impossible to optimize manually

Optimum software-hardware co-design solution for AI applications





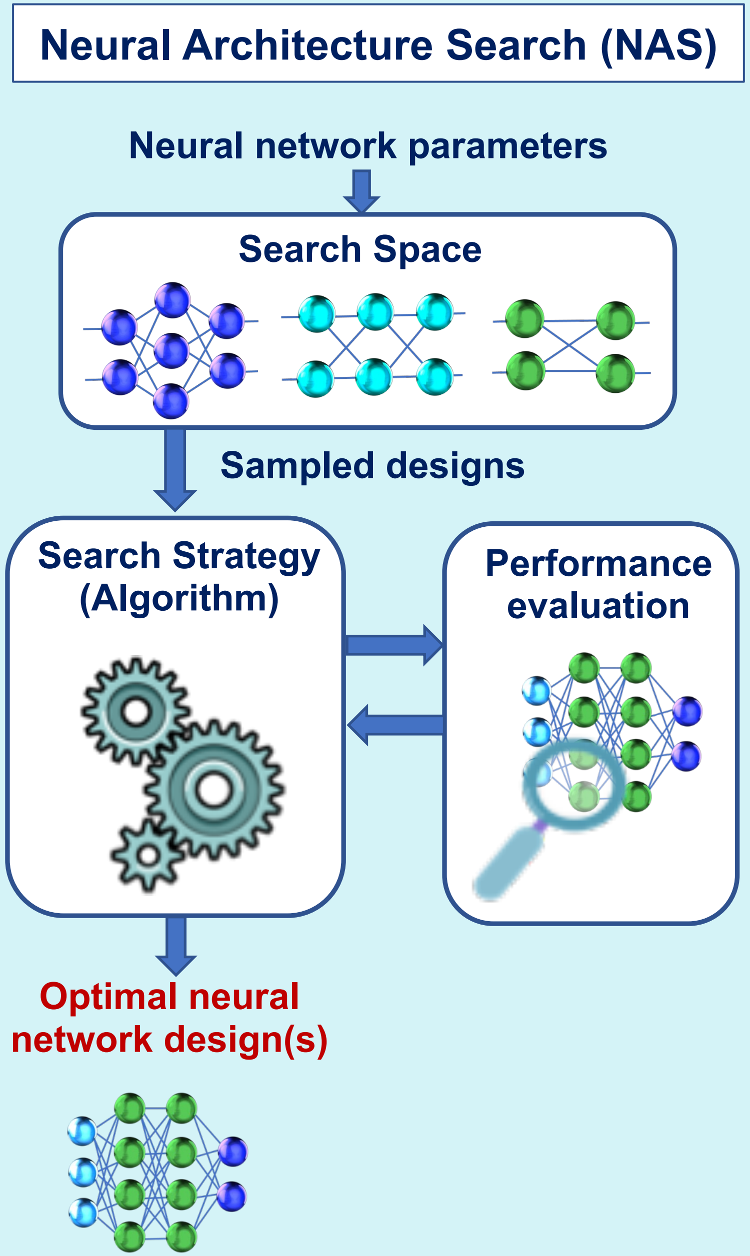
- **Too many parameters to consider**

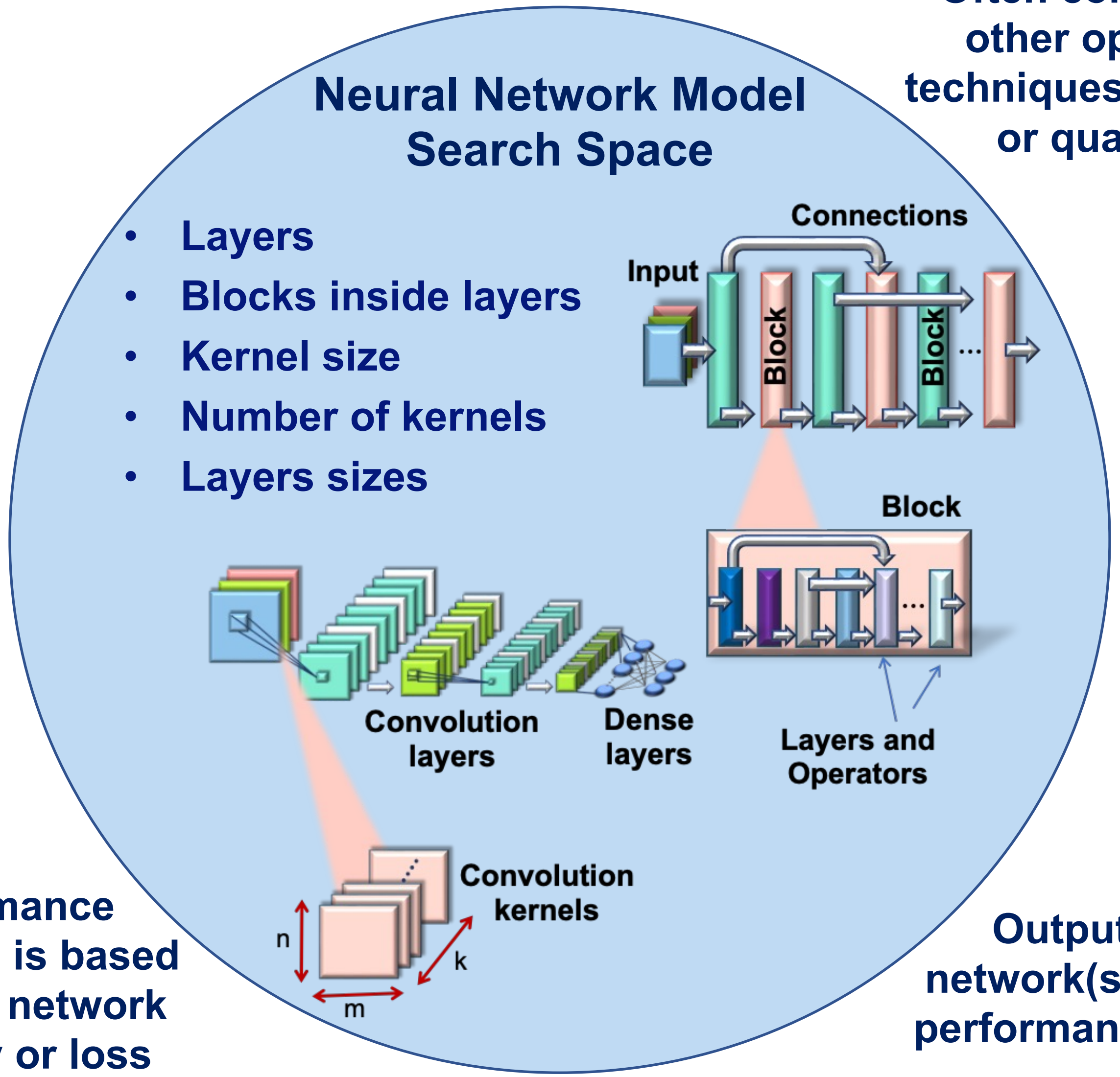
Example: 8.5 * 10^{85} possible combinations

- **Manual optimization of parameters is infeasible, based on guessing (sometimes certain rules), and require a lot of human efforts**
- **Grid search is slow and search time exponentially increases with number of hyperparameters to optimize**
- **Increasing complexity and network size**

One approach to achieving such optimization is through Hardware-aware Neural Architecture Search (HW-NAS).

Neural Architecture Search from Software Perspective





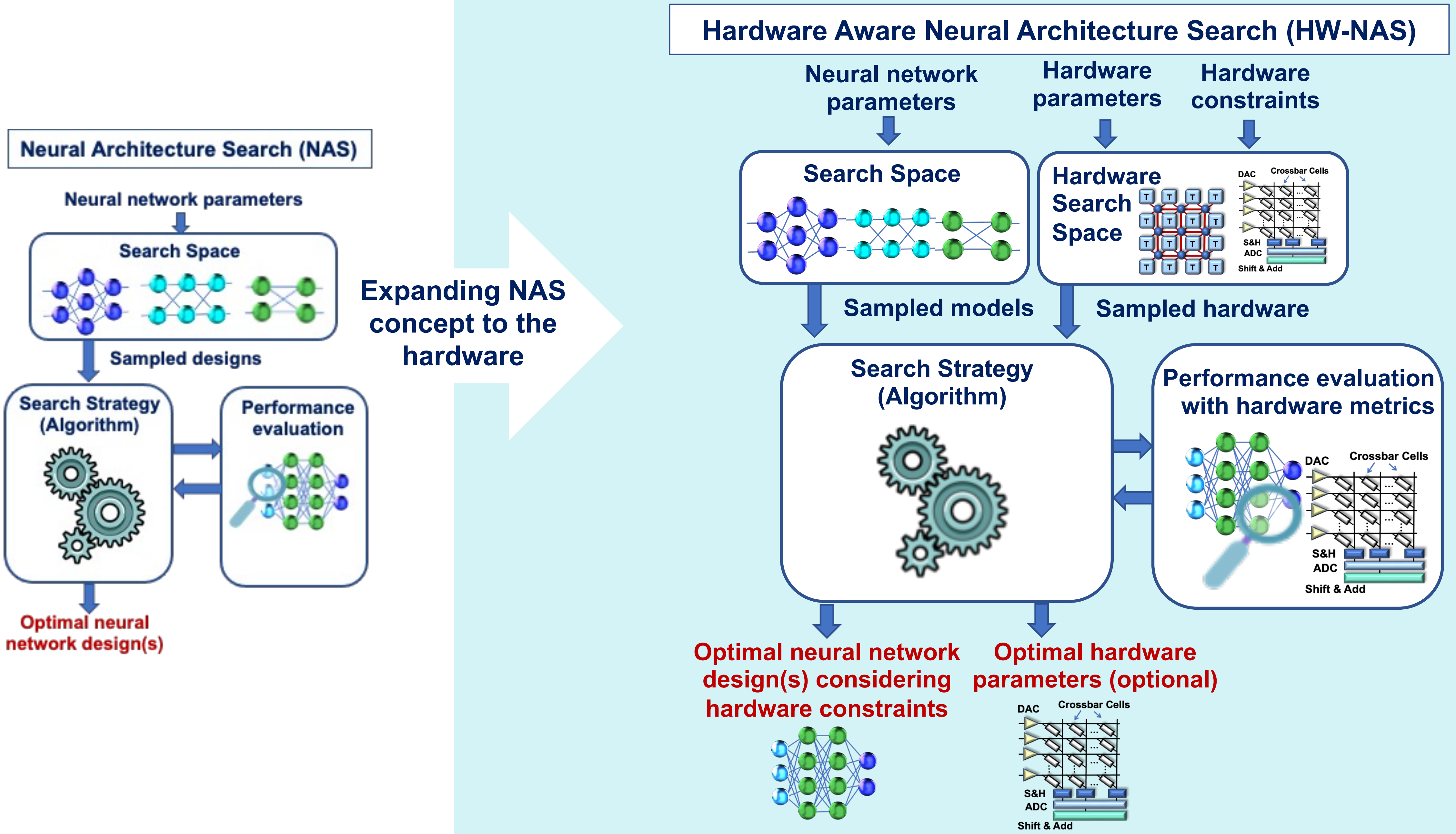
Often combined with other optimization techniques, e.g. pruning or quantization

Performance evaluation is based on neural network accuracy or loss

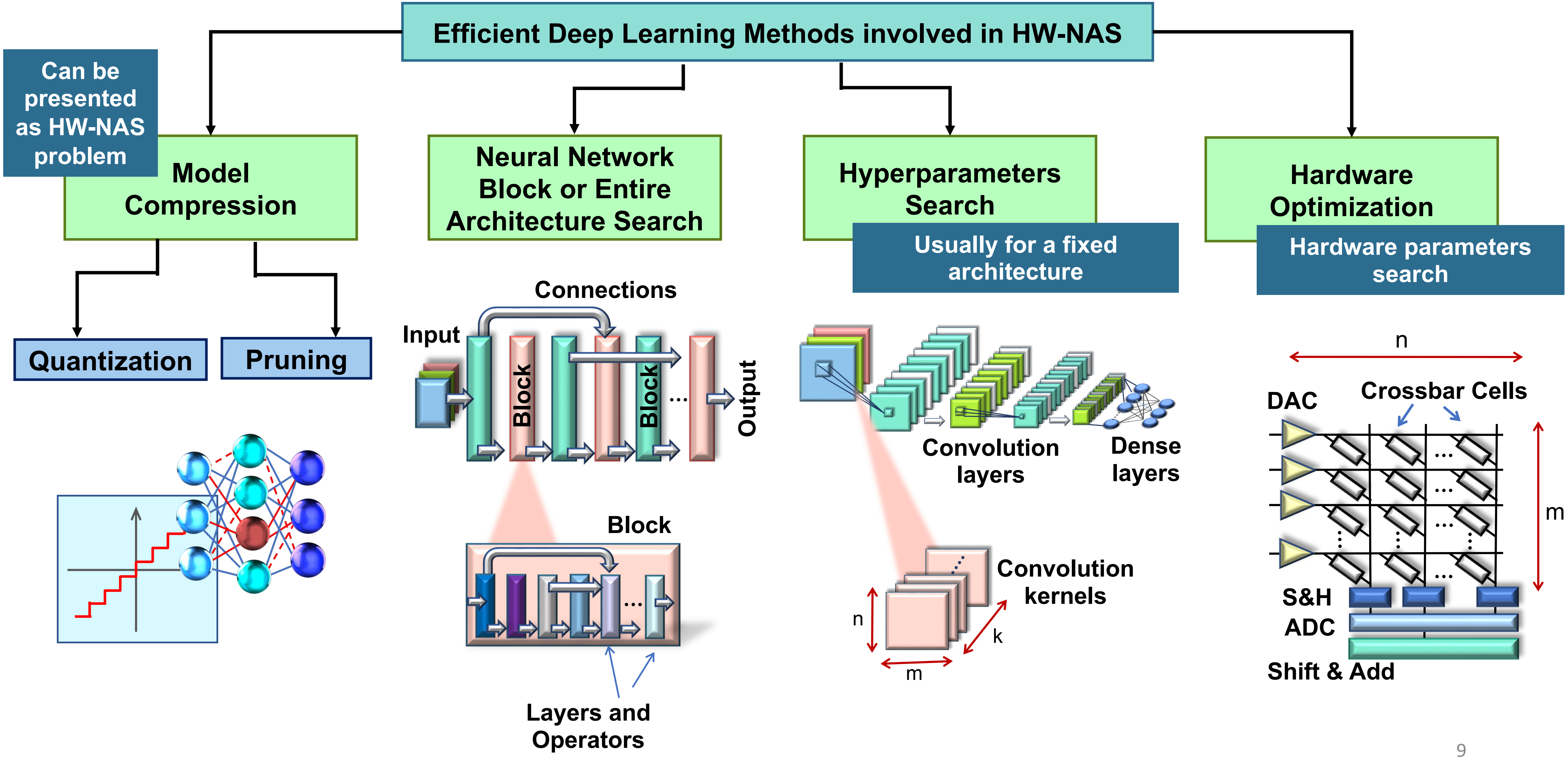
Output: neural network(s) of highest performance accuracy

No consideration of hardware efficiency or considering only high-level metrics, e.g. FLOPs

Hardware-aware Neural Architecture Search

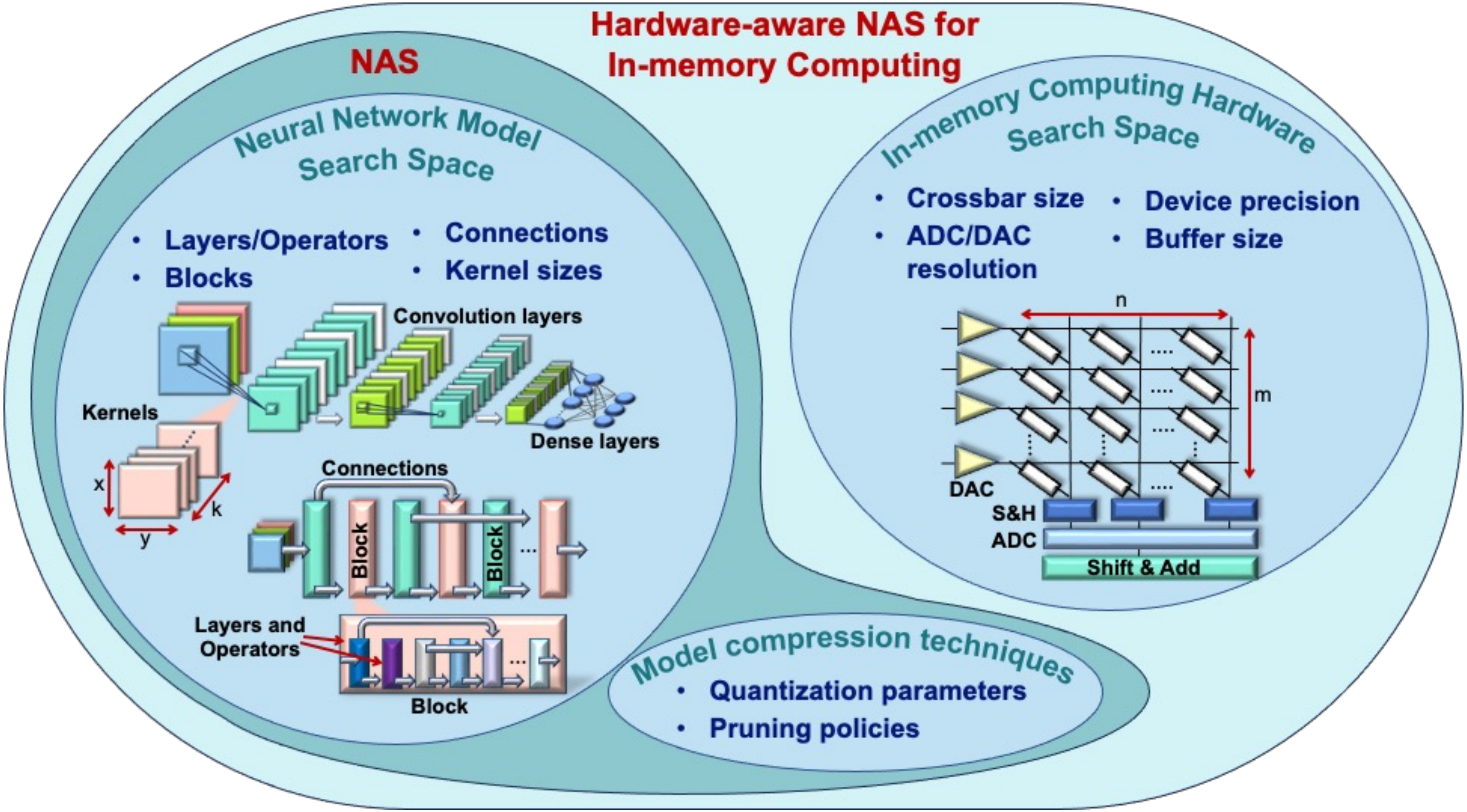


Hardware-Aware Neural Architecture Search

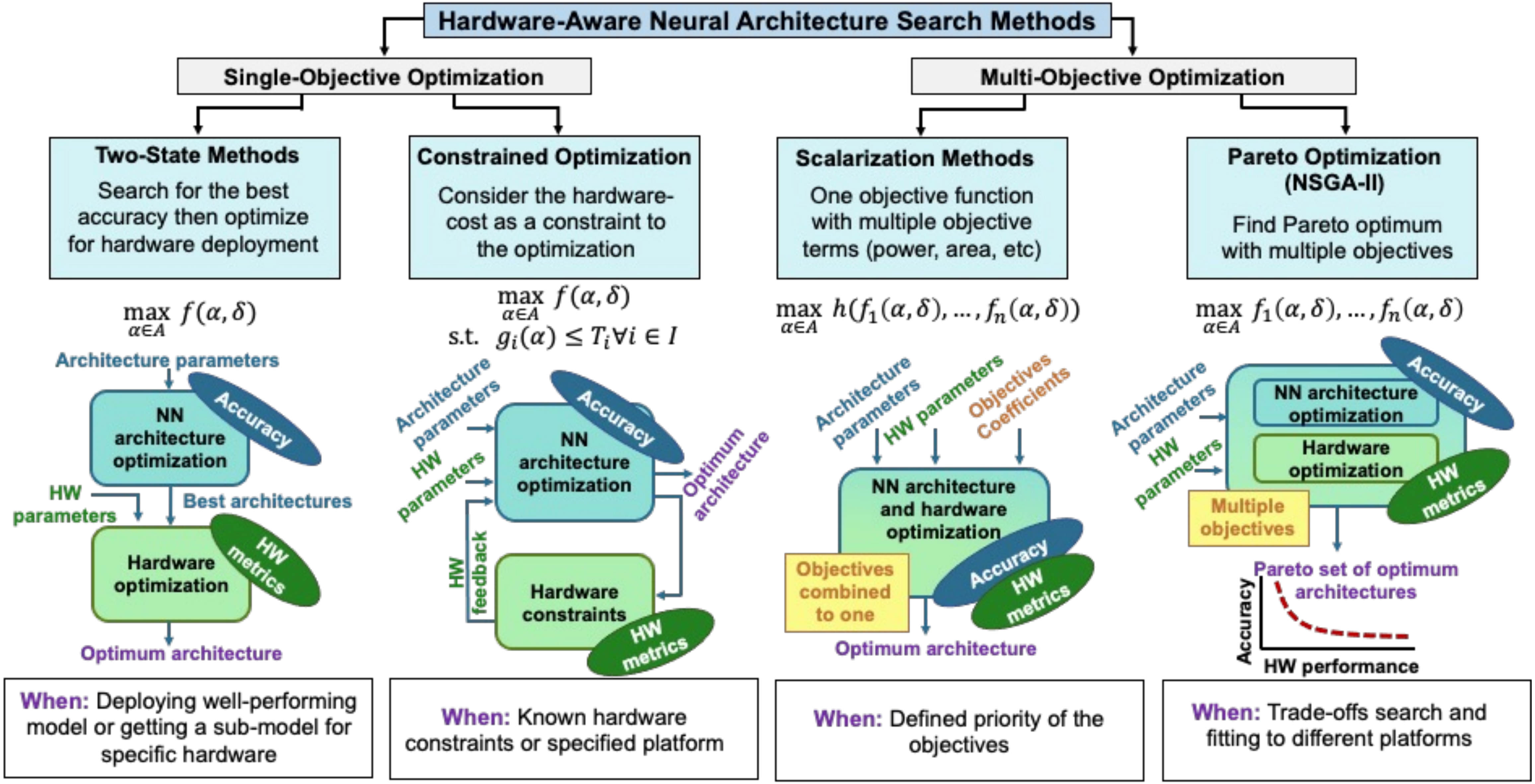


HW-NAS for In-Memory Computing: Search Space

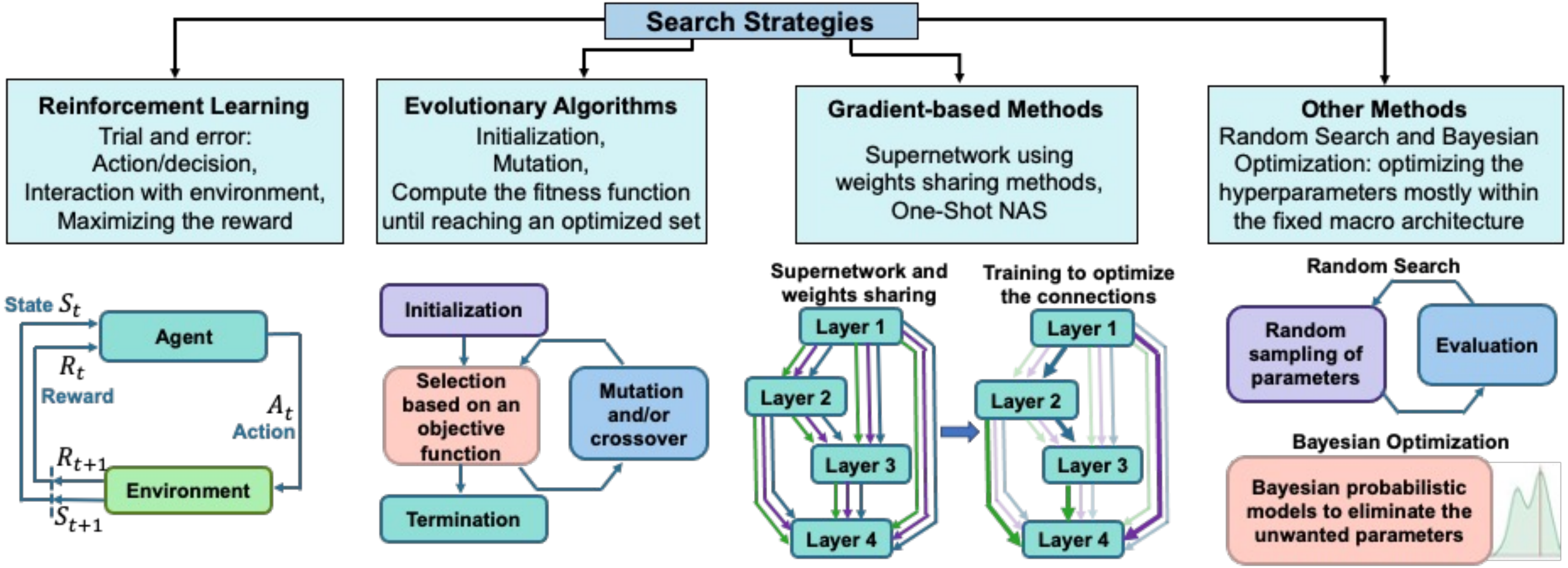
Expanding HW-NAS Search Space for IMC Architectures



HW-NAS Methods



Search algorithms



	Reinforcement Learning	Evolutionary Algorithms	Gradient-based Methods	Other Methods
Search Space:	Discrete	Discrete	Differentiable	Discrete
Speed:	Slow	Average	Fast	Slow
Computational demand:	High	Average	Low	High
Memory requirements:	Small	Small	Large	Small
Scalability[*1]:	Not Scalable	Not Scalable[*2]	Scalable[*3]	Not Scalable[*2]

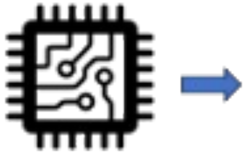
[*1]: refers to time complexity increasing with a search space, [*2]: can be used with a supernetwork search space, [*3]: search time does not increase exponentially with a search space size

Hardware cost estimation methods

Hardware Cost Estimation Methods

Real-time Measurements

Executed on a target hardware



Lookup Table Models

LUTs constructed beforehand on a target hardware, and used in a search

	Power	Latency	Area
Xbar	10uW	100ns	$1um^2$
⋮	⋮	⋮	⋮
ADC	10mW	1us	$1mm^2$

Analytical Estimation

Computing a rough estimate using the related equations

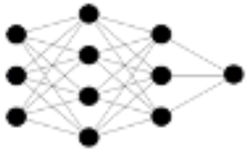
Power $P = P_1 + P_2 + \cdots + P_n$

⋮

Latency $T = T_1 + T_2 + \cdots + T_n$

Prediction Models

Using ML models for hardware cost prediction



Supports differentiable NAS

	Real-time Measurements	Lookup Table Models	Analytical Estimation	Prediction Models
Accuracy:	Highest	Average	Average	High
Speed:	Slow	Average	Fast	Fast
Model Scalability*1 :	High	Moderate	Moderate/High*2	Moderate/High*2
Pre-processing:	No	One-time measurements	Initial estimations	Model pre-training
Transferability*3 :	High	Moderate*4	Moderate/Low*5	Moderate*4
Search Scalability *6 :	High	Low	High	Moderate

*1: across neural network models, *2: depends on how similar is a new model, *3: across different hardware platforms, *4: requires regeneration, *5: depends on the hardware similarity, *6: with increasing search space (number of hyperparameters in a search)

State-of-the-art HW-NAS Frameworks for IMC

	Quantization (search)	Pruning	HW-NAS	Architecture Search Space	Hardware Search Space	Hardware cost	Algorithm	Hardware non-idealities
AnalogNAS (2023)	✗	✗	✓	# of blocks, channels, branches, kernel size	✗	AIHWKit	EA	Variations, Cond. drift
NAS4RRAM (2021)	✗	✗	✓	Layers, channels (residual blocks)	✗	RRAM simulator	EA	Variations
FLASH (2021)	✗	✗	✓	# of skip connections, cells, layers, channels	✗	NeuroSim, BookSim	SHGO	✗
NAX (2021)	✗	✗	✓	Kernel size	Crossbar size	GENIEx	DS	Wire/sourse/sink resistances
Gibbon (2022)	✓	✗	✓	# of blocks, channels, groups, kernel size, bit-width	Crossbar size, ADC/ DAC/device precision	MNSIM	EA	Variations
NACIM (2020)	✓	✗	✓	Architecture hyperparameters, bit-width (int./frac.)	Tile/buffer size, bandwidth	NeuroSim	RL	Variations
UAE (2021)	✓	✗	✓	# of channels, filter size, bit-width (int./frac.)	✗	Analytical	RL	Variations, program. errors
CMQ (2022)	✓	✗	✗	Quantization threshold, bit-width	✗	MINT	DS	Variations
Mixed-precision quantization (2021)	✓	✗	✗	Weight/inputs bit-width (int./frac.)	ADC precision	PUMAsim	RL	✗
EGQ (2021)	✓	✗	✗	Weight/activation bit-width	✗	NeuroSim	GA	✗
RaQU (2021)	✓	✗	✗	Weight/kernel bit-width	✗	Analytical	RL	✗
ASBP (2021)	✗	✓	✗	Bits of weights	✗	Analytical	RL	✗
Auto-prune (2021)	✗	✓	✗	Weights (pruned unimportaint columns)	✗	MNSIM	RL	✗

The latest frameworks (*not included in the paper but worth mentioning*):

XPert (2023)	Two-step co-optimziation of software and hardware parameters, including channel depth, ADC precision, input precision, etc.
CoMN (2024)	Design space exploration for a large IMC hardware search space, including circuits and architecture parameters
Joint Hardware-Workload Co-optimization (2024)	Design space exploration for a large IMC hardware search space to optimize IMC hardware for different workloads simultaneously

- Limited architecture search space (particular network type)
- Limited applications
- Limited hardware search space (only few hardware parameters)
- Consideration of hardware non-idealities and mitigation techniques
- Lack of a unified framework

Moitra, A., Bhattacharjee, et al. (2023, July). XPert: Peripheral Circuit & Neural Architecture Co-search for Area and Energy-efficient Xbar-based Computing. In *2023 60th ACM/IEEE Design Automation Conference (DAC)* (pp. 1-6). IEEE.
Han, L., Pan, et al. (2024). CoMN: Algorithm-Hardware Co-Design Platform for Non-Volatile Memory Based Convolutional Neural Network Accelerators. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems.*
Krestinskaya, O., Fouda, M. E., Eltawil, A., & Salama, K. N. (2024). Towards Efficient IMC Accelerator Design Through Joint Hardware-Workload Co-optimization. *arXiv preprint arXiv:2410.16759.*

Taxonomy of HW-NAS frameworks for IMC applications

Optimizing model for a fixed constrained hardware

- AnalogNAS
- NAS4RRAM
- GA for IMC AI hardware
- Pareto-based NAS
- EGQ
- UAE
- CMQ
- RaQU
- FLASH
- ASBP
- Auto-prune

Neural network model parameters optimization

Searching for the most optimum neural network parameters considering IMC hardware feedback

HW-NAS for optimum model and architecture search

- NAX
- Gibbon
- NACIM
- Mixed-precision quantization
- XPert

Co-optimizing software and hardware parameters

Optimizing hardware for a fixed model (design space exploration)

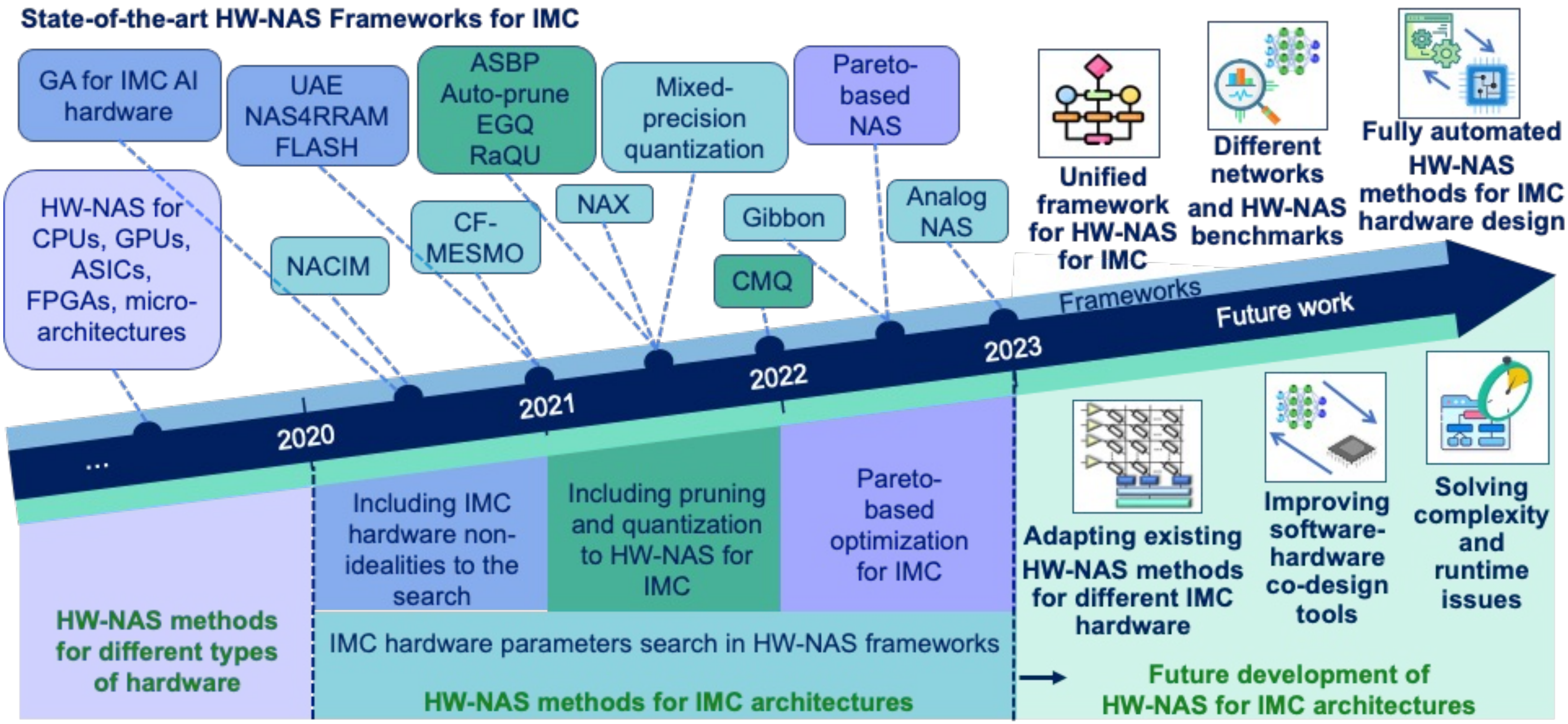
- CF-MESMO for NAS
- CoMN
- Joint workload-hardware co-optimization

Hardware parameters optimization

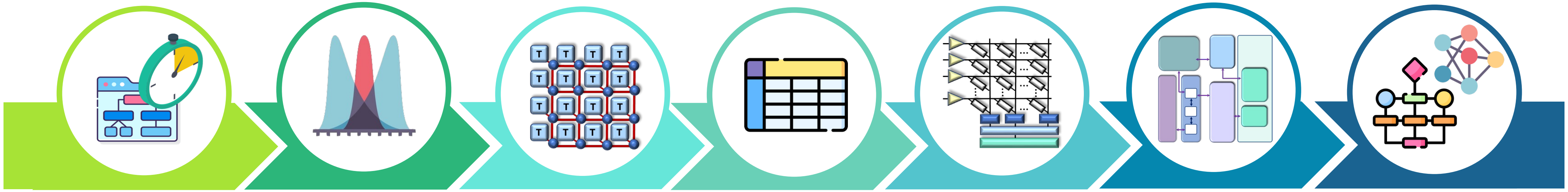
Optimizing only hardware parameters for a fixed neural network model

CF-MESMO for NAS: Yang, X. *et al.* in *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*. 1-9 (IEEE).

HW-NAS for IMC Applications Roadmap



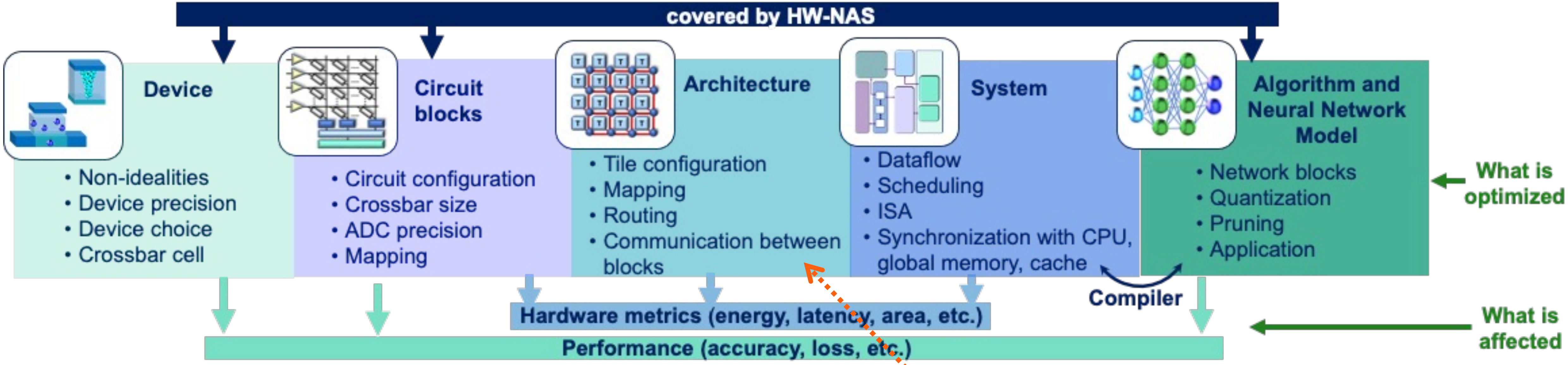
Open Problems and Way Forward



Solving complexity and runtime issues (search space size vs search time)	Including hardware non-idealities mitigation techniques to NAS framework	Expanding hardware and model search spaces, adding more applications	Creating HW-NAS benchmarks	Adapting existing HW-NAS methods for different IMC hardware	Considering system-level challenges (dataflow optimization, scheduling, etc.)	Unified framework incorporating neural network model and hardware search

Open Problems and Way Forward

Neural network model and IMC hardware optimization covered by HW-NAS:

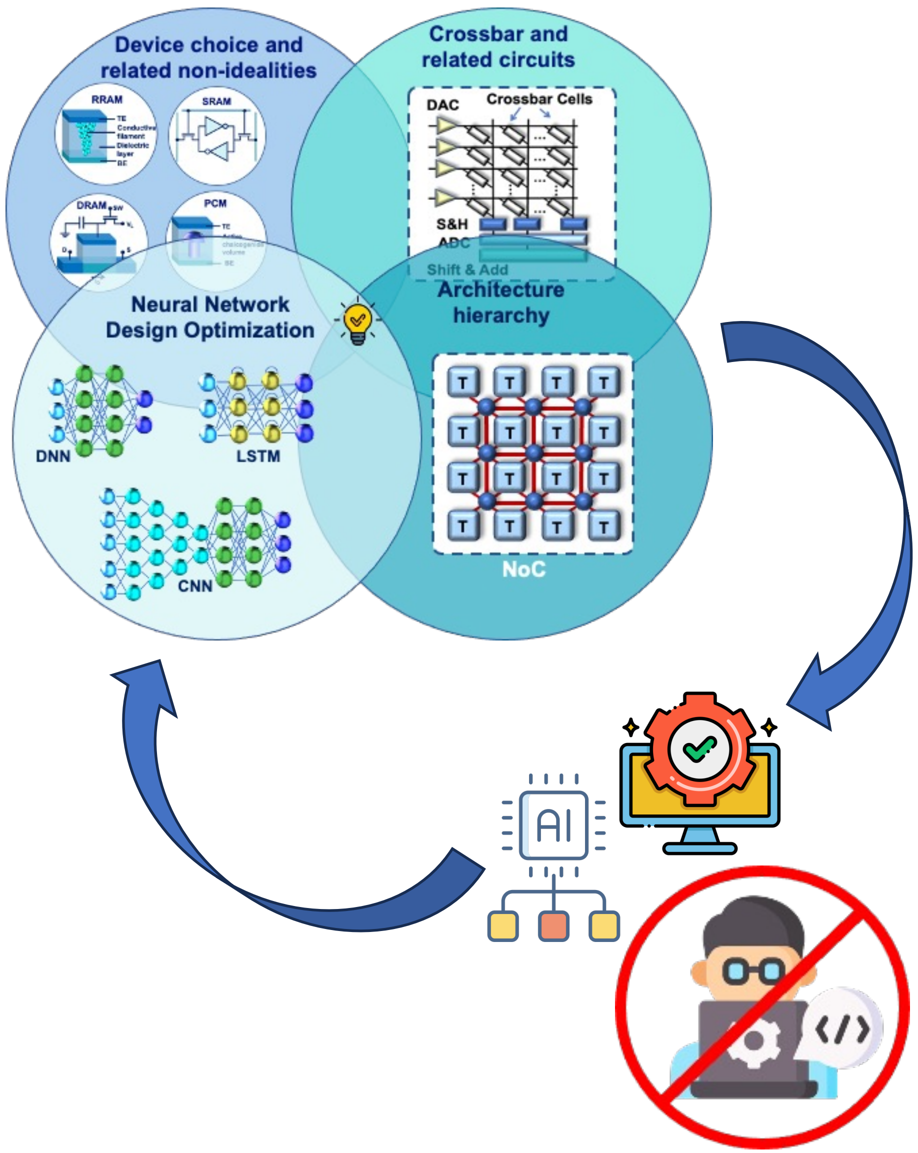


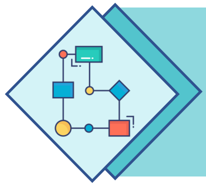
Some architecture-level optimizations can also be considered in HW-NAS.

Combining HW-NAS with other optimization techniques for end-to-end IMC hardware optimization tool?

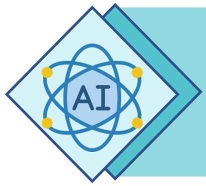
One more step forward

Optimum AI hardware design





Self-adapting Design Algorithms



AI-driven Design Tools

- Fully-automated NAS methods capable of constructing new deep learning operations and algorithms suitable for IMC with minimal human design efforts.
- Example from software: AutoML- Zero – automatically searching for the complete machine learning algorithms (model, optimization procedure, etc with minimum restriction on the form or math operations).
- Reducing human intervention in the design.
- No pre-defined blocks.
- Adaptable to different tasks and constraints.
- IMC awareness – open challenge.

Utilization of AI capabilities to improve and automate both algorithm and hardware design.

Thank you!

ok@ieee.org