Efficient DNN Design with Neural Architecture Search

Dr Hadjer Benmeziane

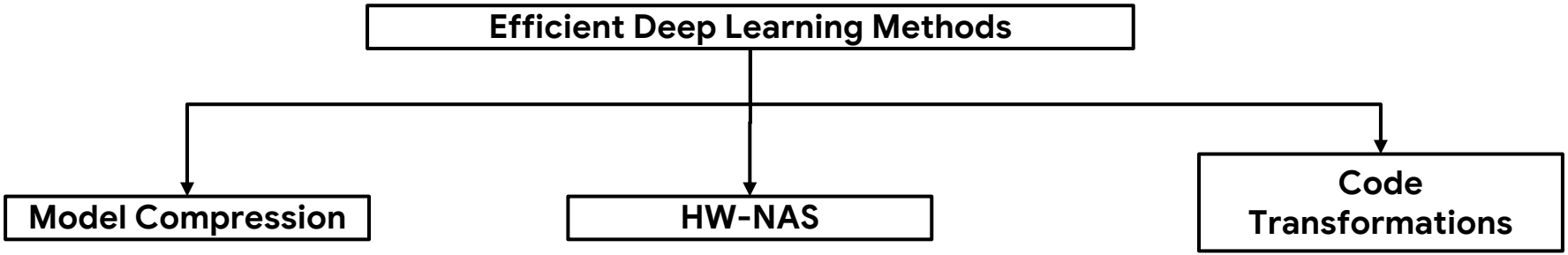
Lecture Plan

- Review deep learning optimizations
- Introduce AutoML & Neural Architecture Search
- Understand Hardware-Aware Neural Architecture Search
  - Search Space
  - Search Strategy
  - Expensive Objective Evaluation
- Case Study: Apply HW-NAS for Analog In-memory Computing

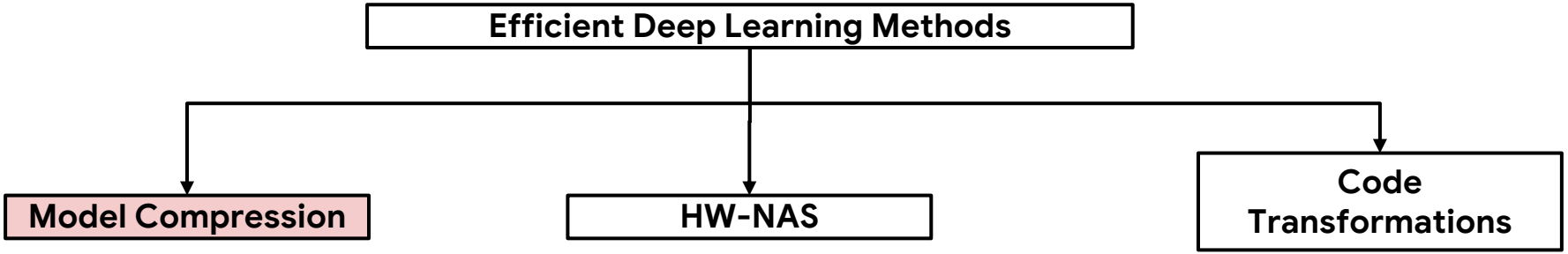
LAB:

Use AnalogAI-NAS package to run and search for an efficient DNN for Analog In-memory Computing.

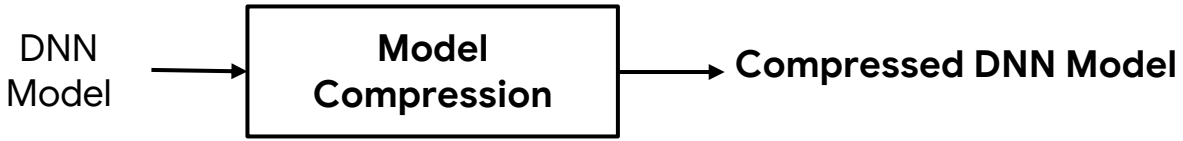
Deep Learning Optimizations



Deep Learning Optimizations

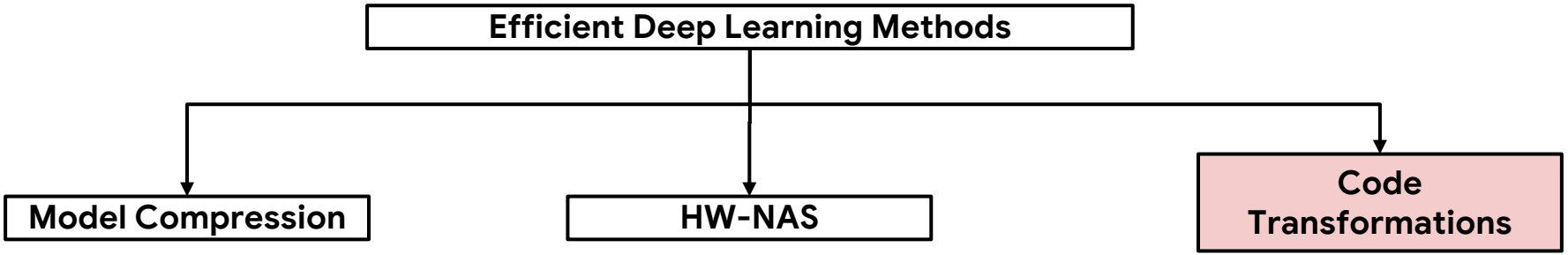


Goal: Maximize
- Compression Ratio
- Accuracy
- Hardware Usability



Hardware Agnostic

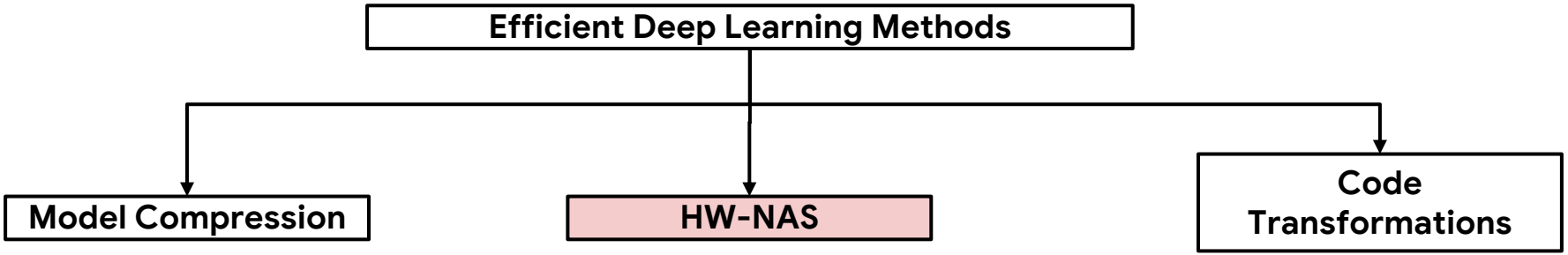
Deep Learning Optimizations



Goal: Apply code optimizations to maximize Hardware Usability

Non Hardware Agnostic

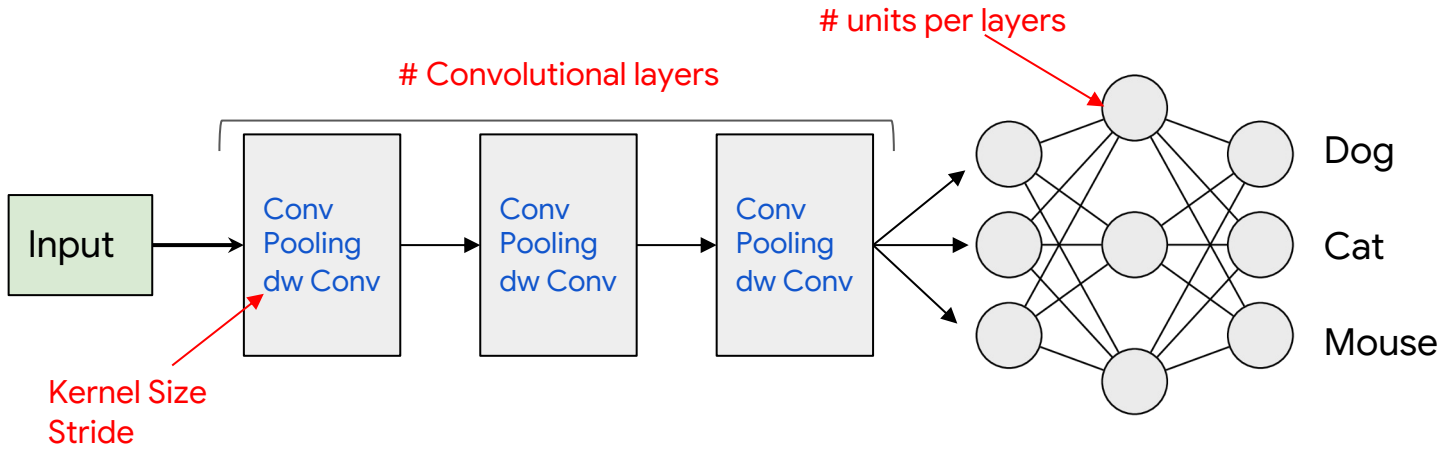
Deep Learning Optimizations



Goal: Search for an architecture that maximizes
Accuracy
Hardware Usability

Hardware-dependent & Hardware-agnostic

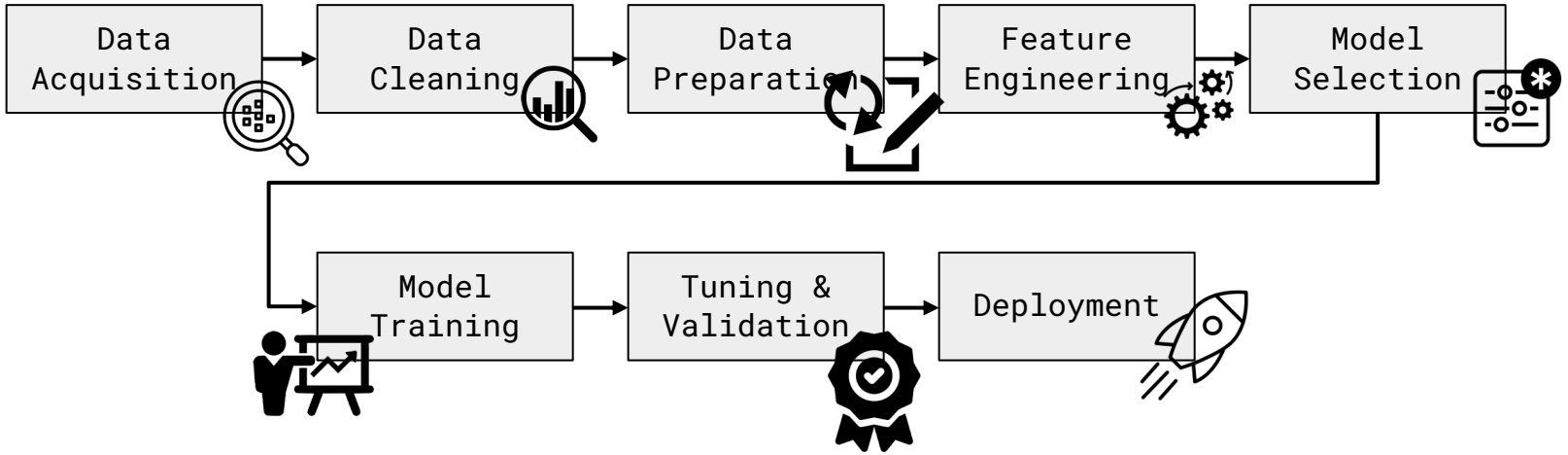
Building the best DNN!



- Architecture
  - Operation per layer
  - Hyperparameters
- Training Hyperparameters
  - Learning Rate
  - Momentum
  - Batch Size

Accuracy & Hardware efficiency are very sensitive to the architecture and hyperparameters used in the model.

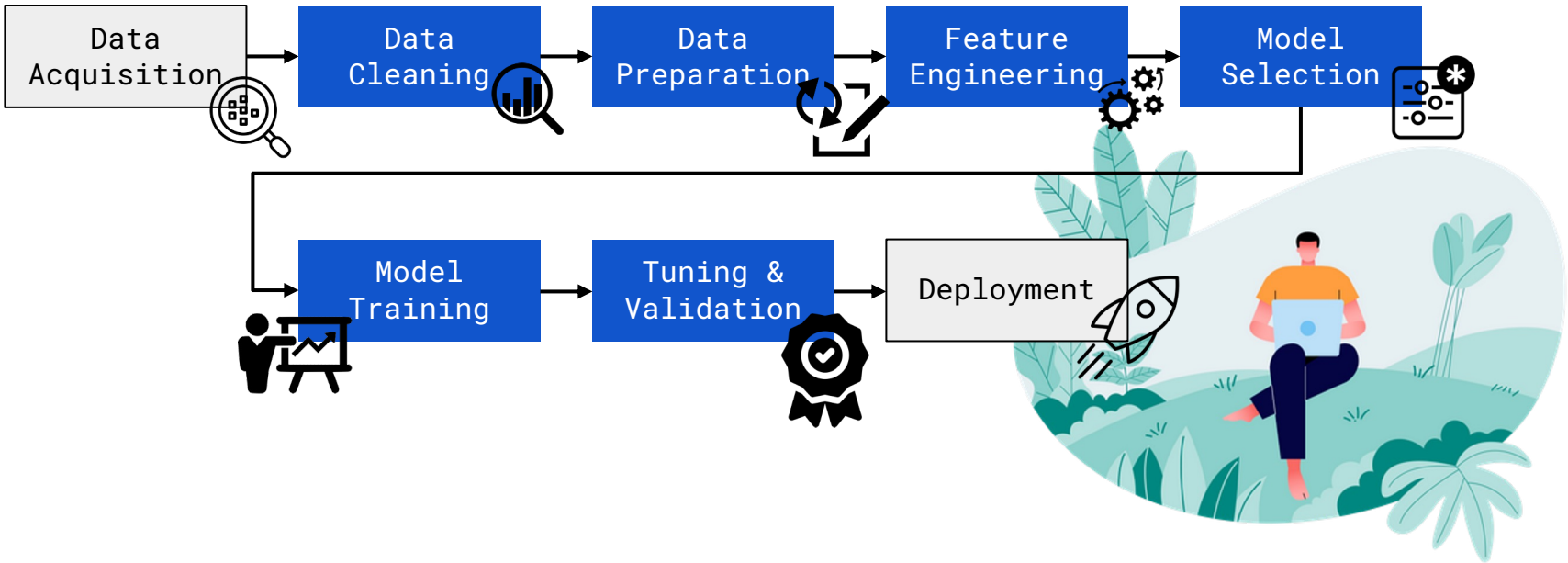
Recall: Machine Learning Pipeline



Traditional Machine Learning Workflow

AutoML

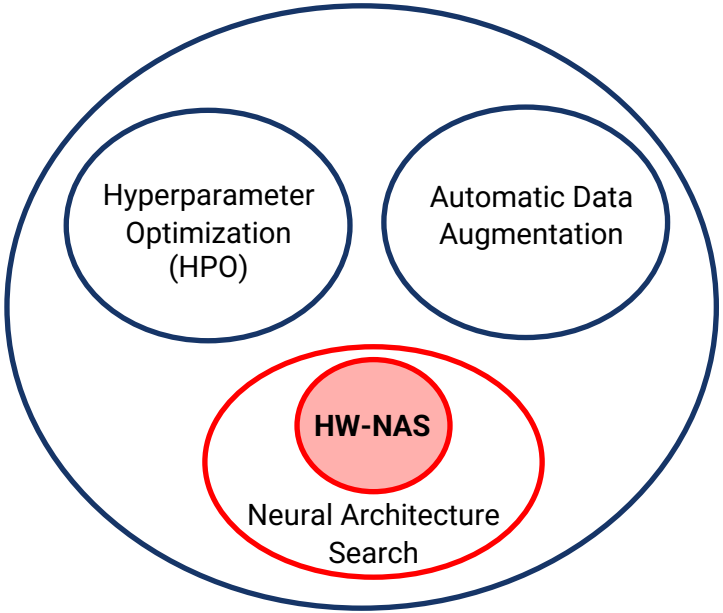
AutoML can allow non-ML experts to build and use machine learning systems for handling targeted tasks without the need of domain knowledge.



AutoML Benefits

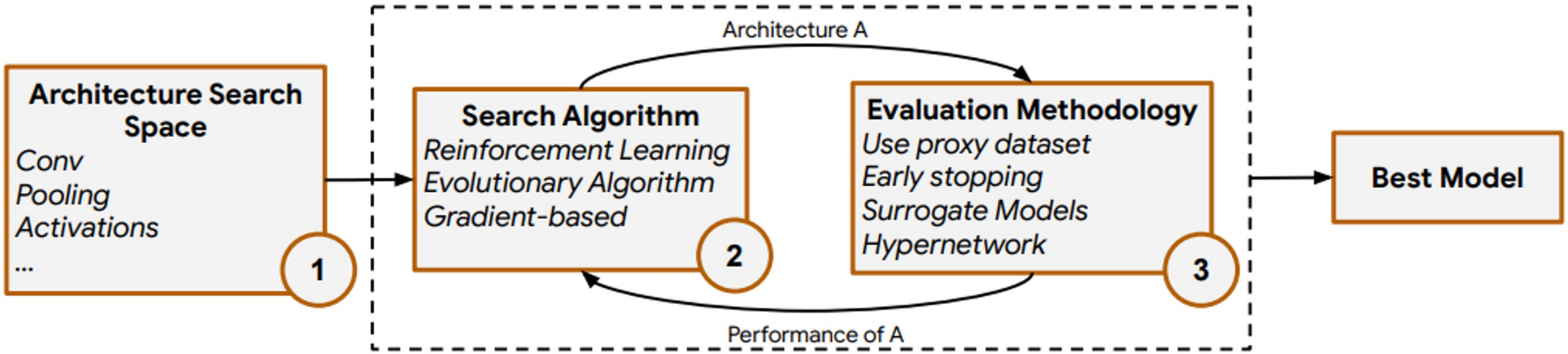
- Allow non-expert to build ML systems
- Reduce the cost of hyperparameters tuning
- Allow ML practitioners to improve their model's accuracy
- Allow ML experts to find new architectures (MobileNet V3)

  Allow ML experts to find efficient architectures for different hardware platforms



Neural Architecture Search

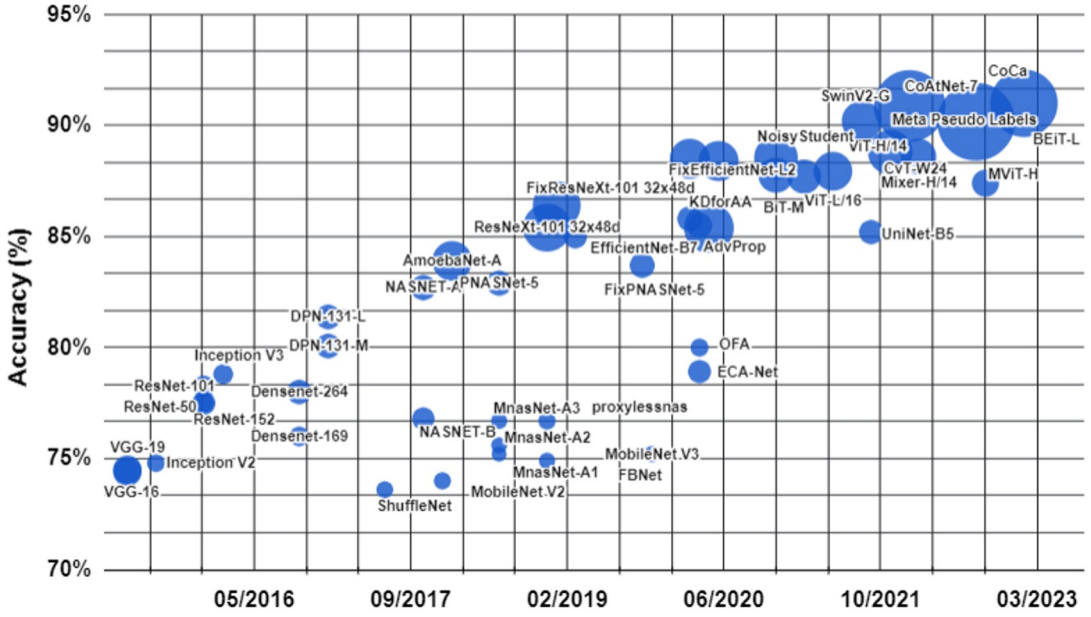
Neural Architecture Search (NAS) is a technique to automatically design deep neural networks.



Overview of Conventional NAS Components.

Context: DNNs are larger !

- Current deep learning models are extremely complex and large.
- Edge & Tiny devices struggle to achieve high performance.
- Cloud and powerful devices consumes a lot of energy, memory and are expensive.



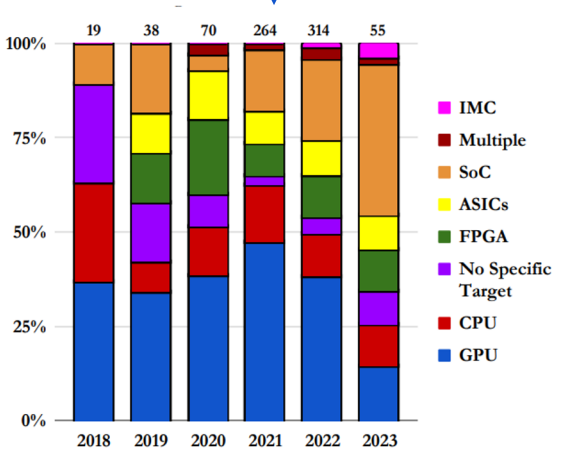
Accuracy of various CNN models on ImageNet for Image Classification task with the number of parameters[1, 2].

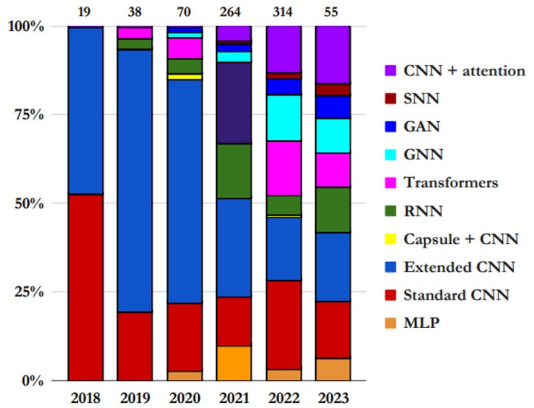
[1] Benmeziane, Hadjer, et al. "A comprehensive survey on hardware-aware neural architecture search." arXiv preprint arXiv:2101.09336 (2021).
[2] Benmeziane, Hadjer, et al. "Hardware-Aware Neural Architecture Search: Survey and Taxonomy." IJCAI. 2021.

Hardware-Aware Neural Architecture Search

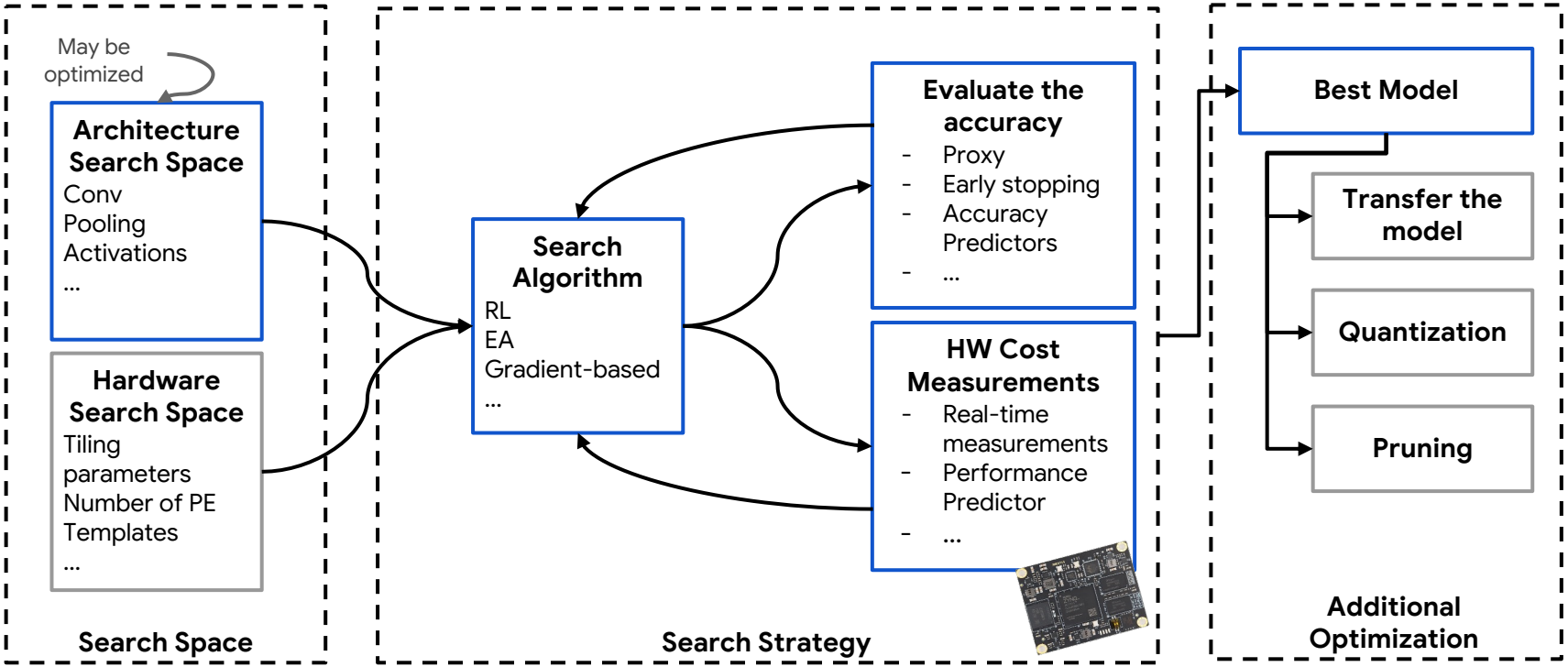
Goal: Find the most efficient architecture for a target hardware for a specific task.

- **Task-specific Performance:**
  - Accuracy
  - Intersection over Union
  - Average Precision
  - ...
- **Hardware Efficiency:**
  - Fast (Latency)
  - Energy-efficient
  - Deployable in tiny memories (memory size)

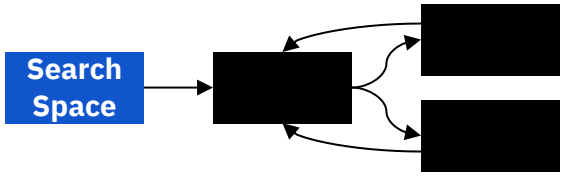




Hardware-Aware Neural Architecture Search



HW-NAS: Architecture Search Space



1- Layer-wise Architecture Space

Predefined List

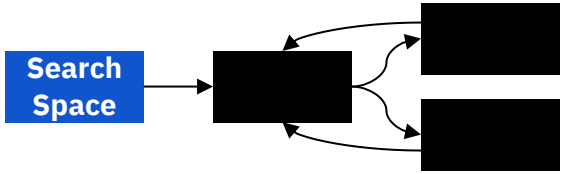
Standard Conv
Depthwise Conv
Dense
Relu
...

Select Operators

Input → **Standard Convolution** → **ReLu** → ... → **Dense** → **Outputs**

Select a depth from a pre-defined range

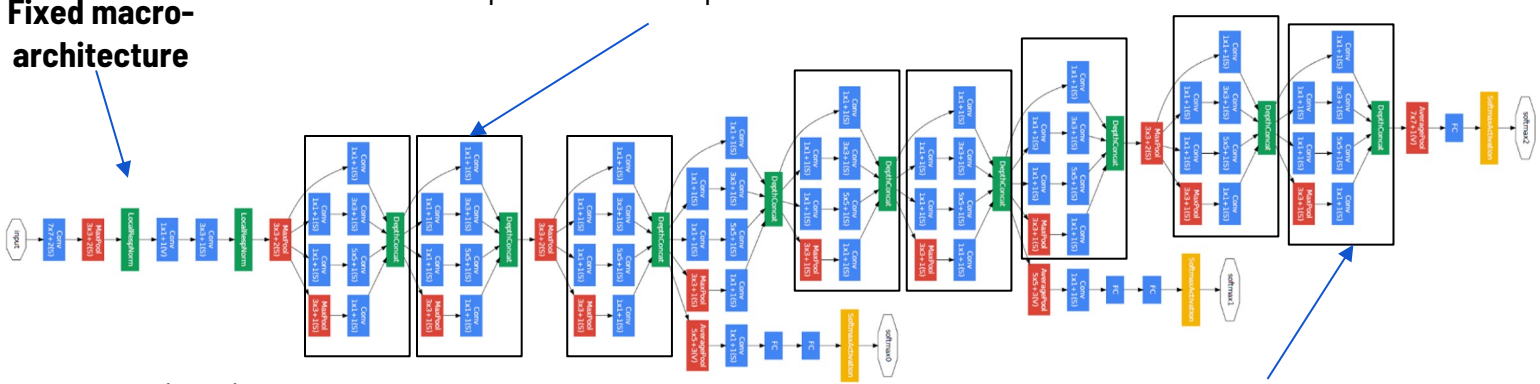
HW-NAS: Architecture Search Space



2- Cell-based Architecture Space

Fixed macro-architecture

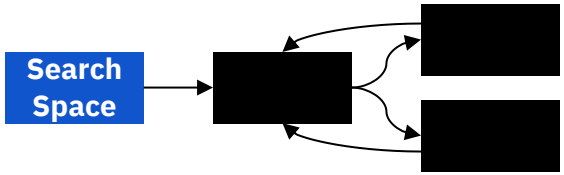
Select Operators from a pre-defined list within the cell



GoogleNet (2014), with InceptionV2 cells

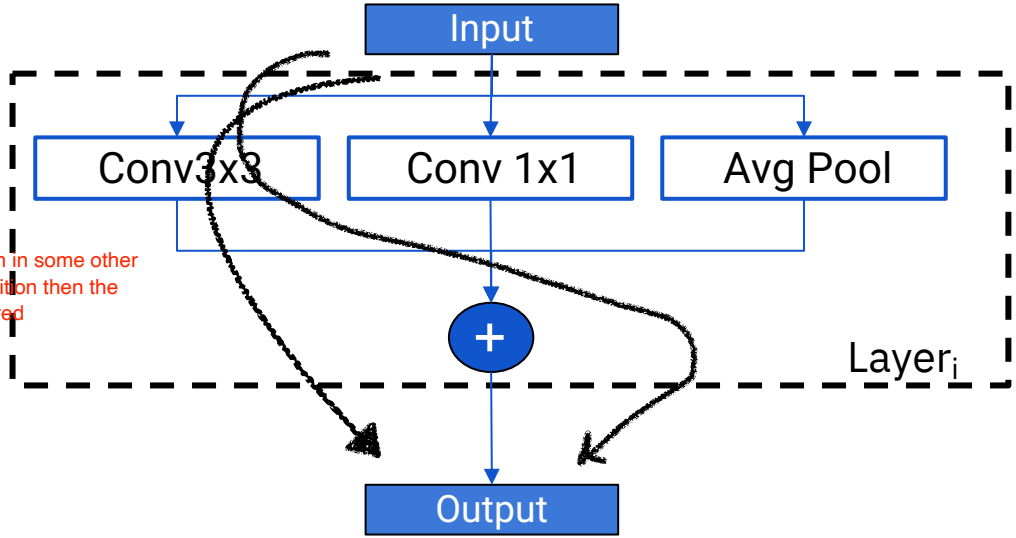
Select the number of cells

HW-NAS: Architecture Search Space



3- Supernetwork

- Each layer contains many blocks at the same time.
- This definition allows the use of weight sharing estimation strategy.
- Examples of Supernetworks include: DARTS[3], ProxylessNAS[4].



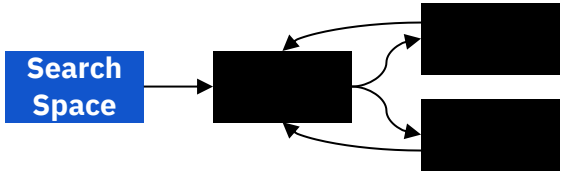
if a conv 3x3 block is used again in some other configuration at the same position then the weights will be shared

first we train the super network and then we search for the best architecture

this weight sharing feature allows us to have a very large search space in case of super network

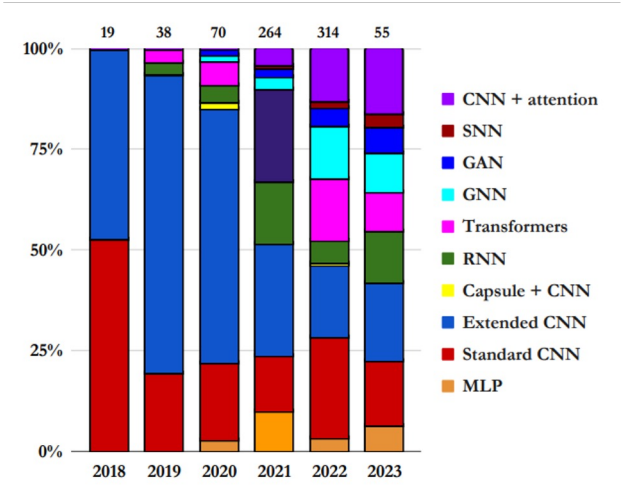
[3] Liu, Hanxiao, Karen Simonyan, and Yiming Yang. "DARTS: Differentiable Architecture Search." ICLR. 2018.
[4] Cai, Han, Ligeng Zhu, and Song Han. "ProxylessNAS: Direct Neural Architecture Search on Target Task and Hardware." ICLR. 2018.

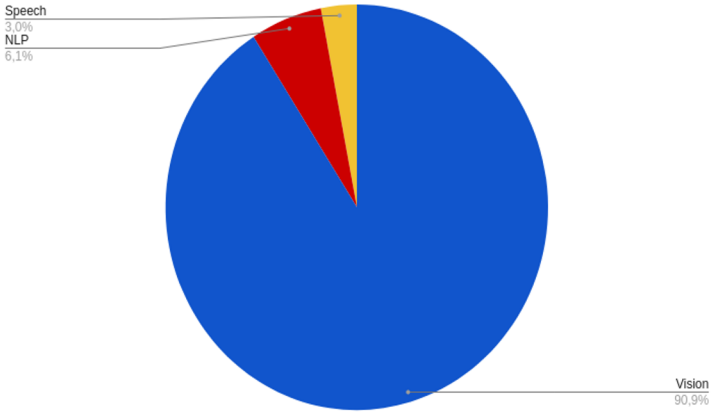
HW-NAS: Architecture Search Space



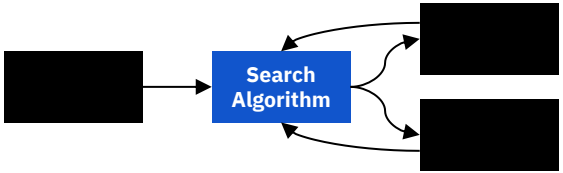
Challenges in designing a search space:

a. Unavoidable Human Bias
b. Increasing Search Space size especially when adding hardware parameters
c. Focused on Vision Domain and Image Classification Task





HW-NAS: Search Strategy



Hardware-Aware NAS Methods

Single-Objective Optimization

Multi-Objective Optimization

Two-Stage Methods
Search for the best in terms of accuracy then compress it

$$\max_{\alpha \in A} f(\alpha, \delta)$$

Constrained Optimization
Consider the hardware-cost as a constraint to the optimization problem

$$\max_{\alpha \in A} f(\alpha, \delta)$$
$$\text{subject to } g_i(\alpha) \leq T_i \forall i \in I$$

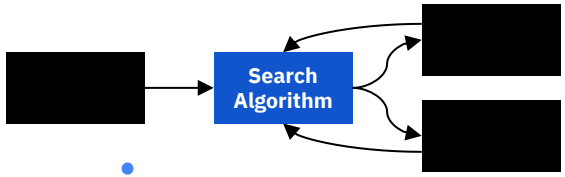
Scalarization Methods
One objective function with multiple objective terms: accuracy, memory, energy...

$$\max_{\alpha \in A} h(f_1(\alpha, \delta), ..., f_n(\alpha, \delta))$$

Pareto-aware Methods
Algorithm considering multiple objectives simultaneously

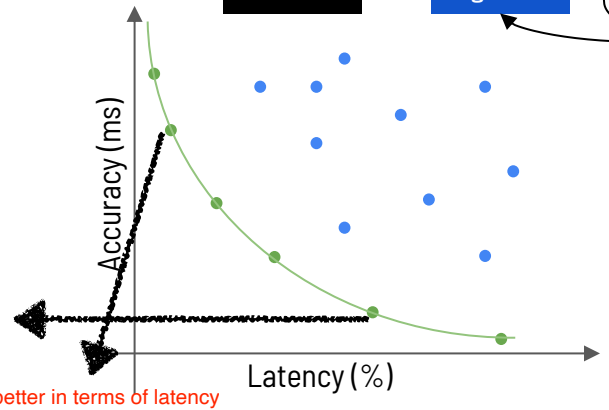
$$\max_{\alpha \in A} f_1(\alpha, \delta), ..., f_n(\alpha, \delta)$$

HW-NAS: Search Strategy



Dominance & Pareto front

The Pareto front is a set of solutions in a multi-objective optimization problem where no other solutions in the search space are superior to them when considering all objectives.



To find the Pareto front, we use the dominance:

A solution 'A' is said to dominate solution 'B' if 'A' is better than or equal to 'B' in all objectives and strictly better in at least one objective.

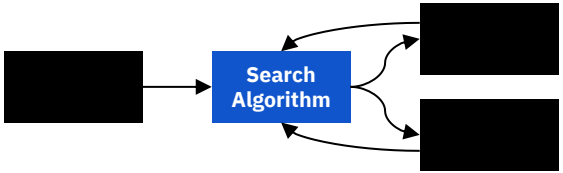
$$A \prec B \iff \begin{matrix} \forall i, & f_i(A) \leq f_i(B) \\ \exists j, & f_j(A) < f_j(B) \end{matrix}$$

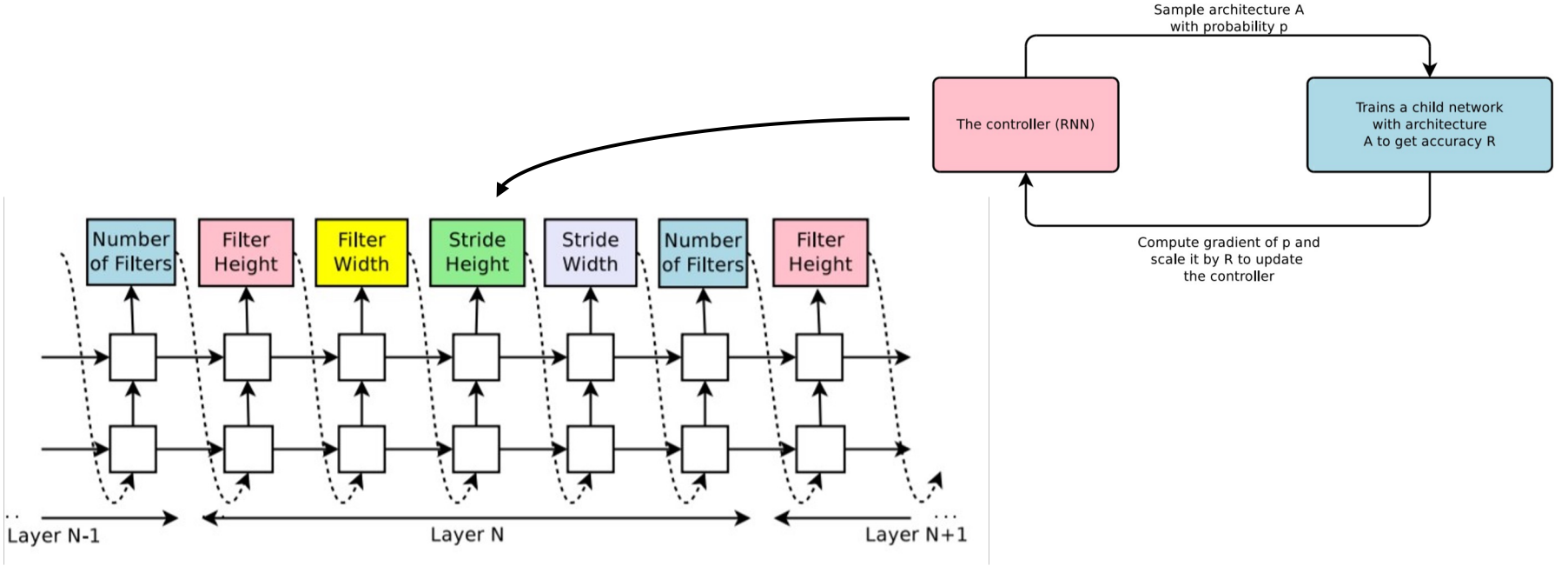
- *Non-dominated solutions:* there is no other solution that can increase the accuracy without increasing the latency.
- *Dominated solutions:* there is a better solution in terms of accuracy-latency trade-off.
- *Pareto Front:* set of all non-dominated solutions

For two solutions on pareto front, no one dominates another one. thats why we cant say which one is better

HW-NAS: Search Strategy

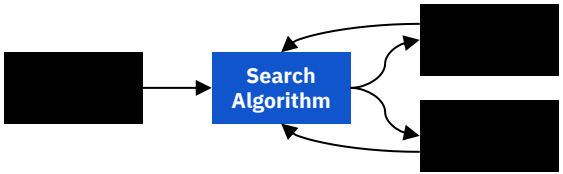
Reinforcement Learning (as a search strategy)



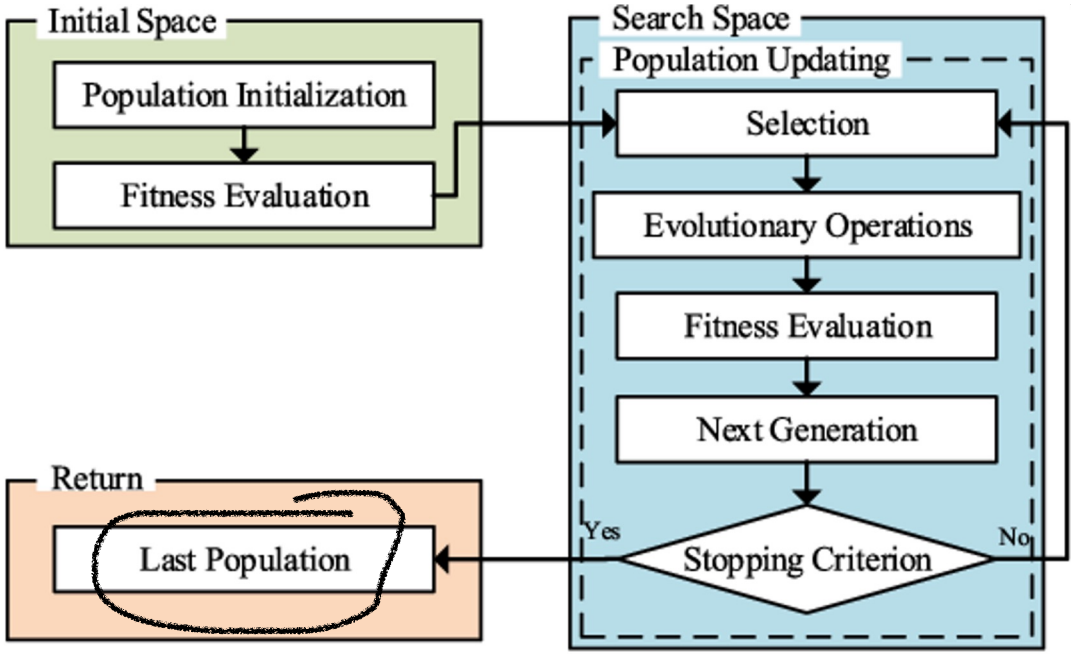


Zoph, Barret, and Quoc Le. "Neural Architecture Search with Reinforcement Learning." *ICLR.* 2016.

HW-NAS: Search Strategy

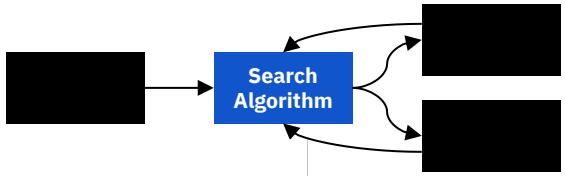


Evolutionary Algorithm



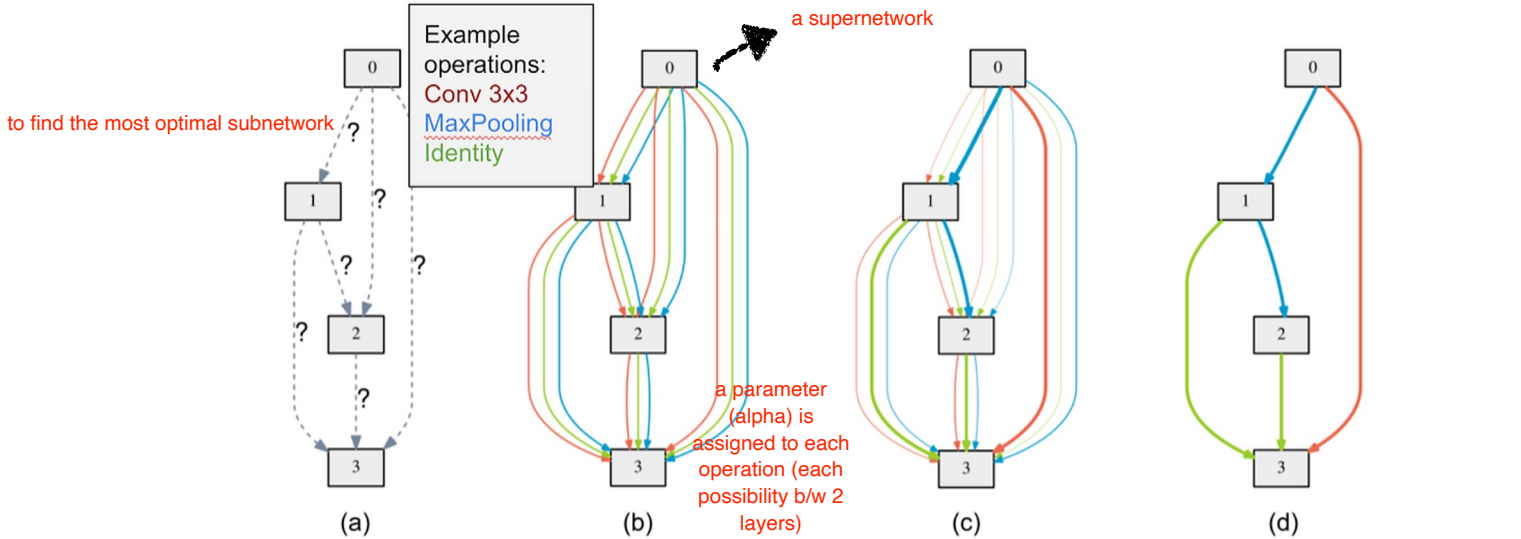
Liu, Yuqiao, et al. "A survey on evolutionary neural architecture search." *IEEE transactions on neural networks and learning systems* 34.2 (2021): 550-570.

HW-NAS: Search Strategy



Gradient-based Algorithms

this works with the super network type search space



Goal: Find the optimal cell, by placing proper operations (e.g. conv, pooling) at edges

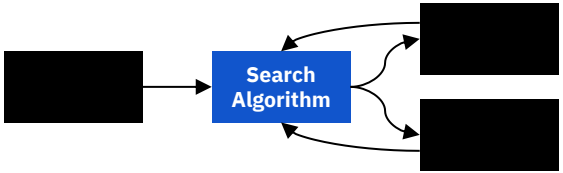
Superpose: each edge is the sum over the outputs of multiple operations, weighted by continuous "architecture parameters" α

Search: Optimize the architecture weights α, using gradient descent on validation loss

Discretize: select the operation with the highest architecture weight, to be the final architecture

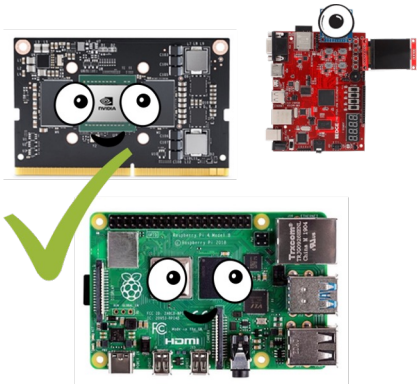
Liu, Hanxiao, Karen Simonyan, and Yiming Yang. "DARTS: Differentiable Architecture Search." *ICLR.* 2018.

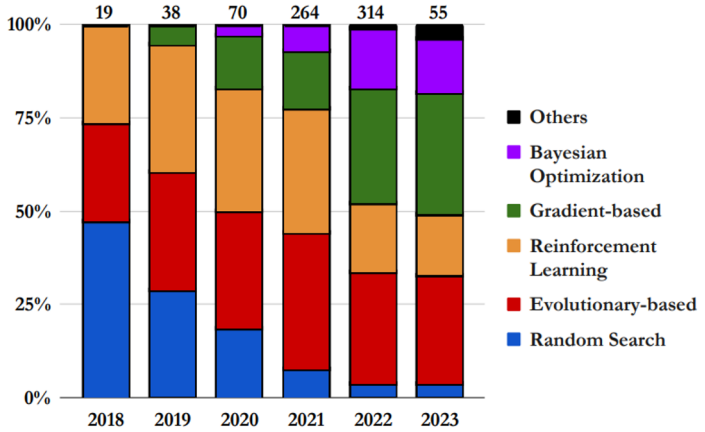
HW-NAS: Search Strategy



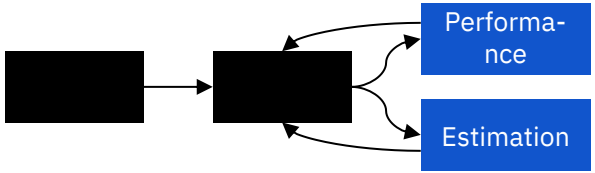
Challenges in search exploration:

a. Computational Cost and time of the search
b. With the increasing search space size, it has become important to carefully choose and optimize the search strategy.
c. Multi-objective search strategies allow the search to find optimal architectures in terms of trade-off.
d. Hardware-Aware Search Space VS Hardware-Aware Search Strategy





HW-NAS: Evaluation Component



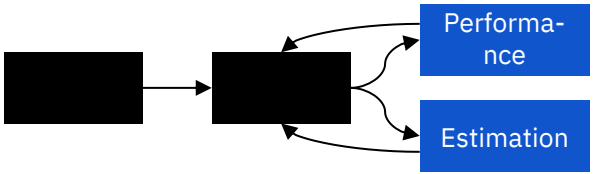
- The main bottleneck component of HW-NAS is: **the accuracy and HW efficiency estimation component.**
- Training 1 architecture on ImageNet requires ~2hrs
- Running 1 architecture on GPU to get the inference latency requires ~10min

The evaluation methods are the main time-consuming components

Model	Search Cost (GPU Hours)
NASNet-A[5]	48,000
AmoebaNet-A[6]	75,600
MNASNet[7]	40,000

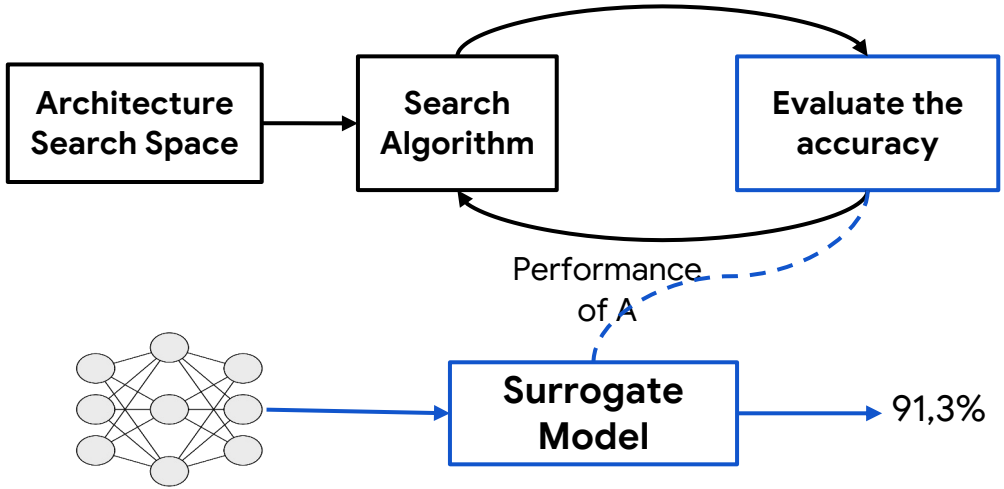
[5] Pham, Hieu, et al. "Efficient neural architecture search via parameters sharing." International conference on machine learning. PMLR, 2018.
[6] Real, Esteban, et al. "Regularized evolution for image classifier architecture search". AAAI. 2019.
[7] Tan, Mingxing, et al. "Mnasnet: Platform-aware neural architecture search for mobile.". CVPR. 2019.

HW-NAS: Evaluation Component



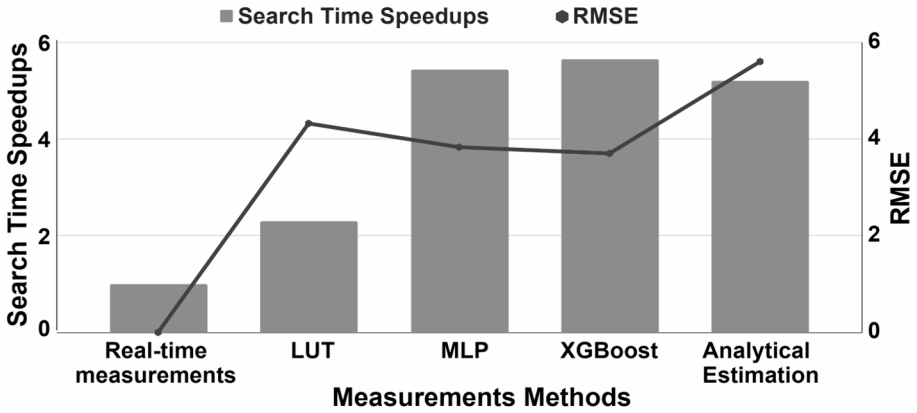
1. Full Training — impractical - will take too much time
2. Partial Training (Early Stopping)
3. Proxy Datasets — example - using a smaller dataset for NAS
4. Learning Curve Extrapolation
5. Surrogate Models

A model that tries to predict the accuracy of new architectures based on the results found so far



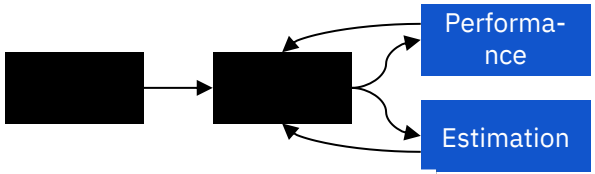
[8] White, Colin, et al. "How powerful are performance predictors in neural architecture search?." Advances in Neural Information Processing Systems 34 (2021).

HW-NAS: Evaluation Component



Comparison of different measurements methods for latency on NAS-Bench-201

- Real-time measurements add another layer of complexity to the NAS algorithms which are already expensive.
- Predictive models are better latency estimators than LUT.

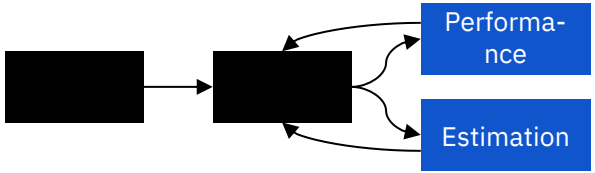


Method	Hardware Cost Metric	References
Real-time measurements	Latency	MNASNet NetAdapt
	Energy	NetAdapt MONAS
Lookup Table Models	Latency	FBNet HotNAS
Low Fidelity Estimation	Latency	FNAS NASCaps
	Energy	NASCaps
	Memory footprint	NASCaps
	Area	NASAIC
Prediction Model	Latency	proxylessNAS NASAIC NeuNets LEMONADE

[8] White, Colin, et al. "How powerful are performance predictors in neural architecture search?." Advances in Neural Information Processing Systems 34 (2021).

HW-NAS: Evaluation Component

Challenges in building an estimator:

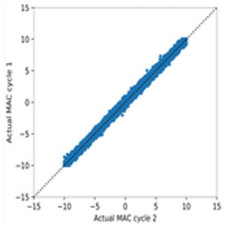


- **Scalability**
- **Generalization**
- **Model Complexity**
- **Evaluation Stability**

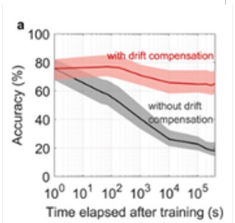
Case study
Analog-Aware
Neural Architecture Search

Case study: Analog-Aware Neural Architecture Search

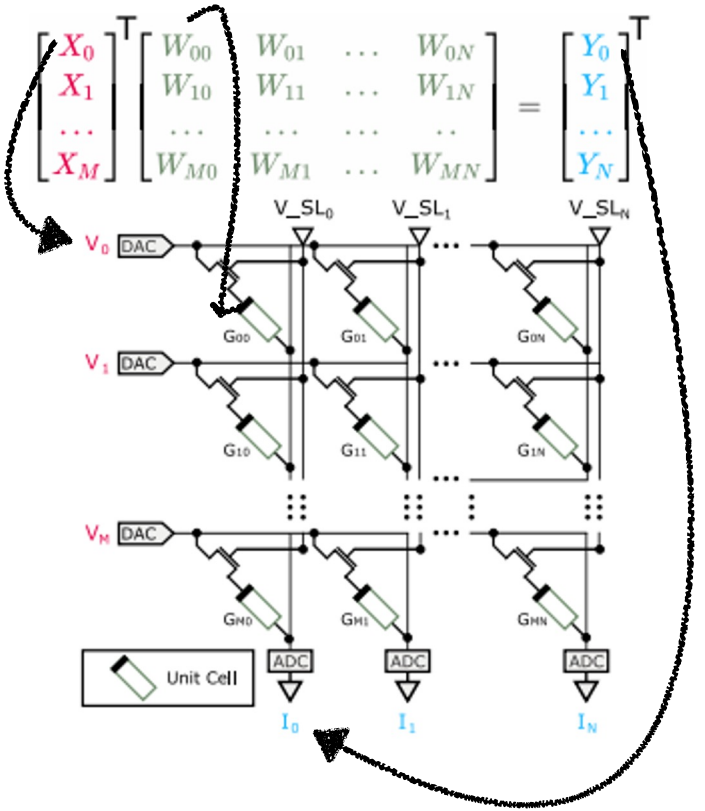
- Analog in-memory computing significantly reduces the energy consumption by performing computations directly within memory cells.
- Eliminates the need for data movement between memory and processing units, a major energy consumer in digital computing
- **BUT!** these devices require architectures that are **robust** and **resilient** to analog compute characteristics.



Noise



Drift

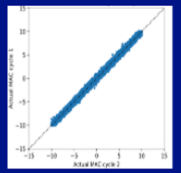


Why Analog NAS

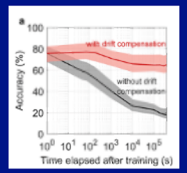
Model Efficiency

Today's manually or NAS generated neural networks are not suitable for Analog IMC accelerators.

Analog Compute Characteristics



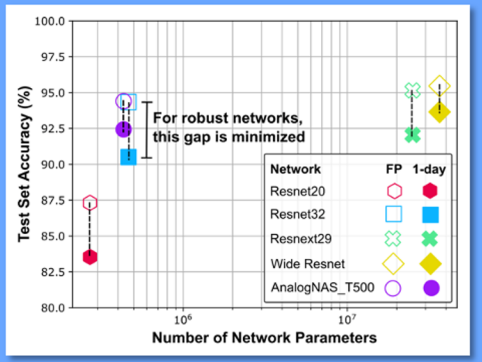
Noise



Drift

Need to design architectures that are robust and resilient to Analog Compute characteristics.

Analog NAS



Automate the design of efficient neural networks that can run efficiently on IMC hardware.

AnalogNAS Overview

A subset of HW-NAS which automatically searches for the best DL architecture for a given task that run efficiently on a target hardware platform

ResNet-like search space

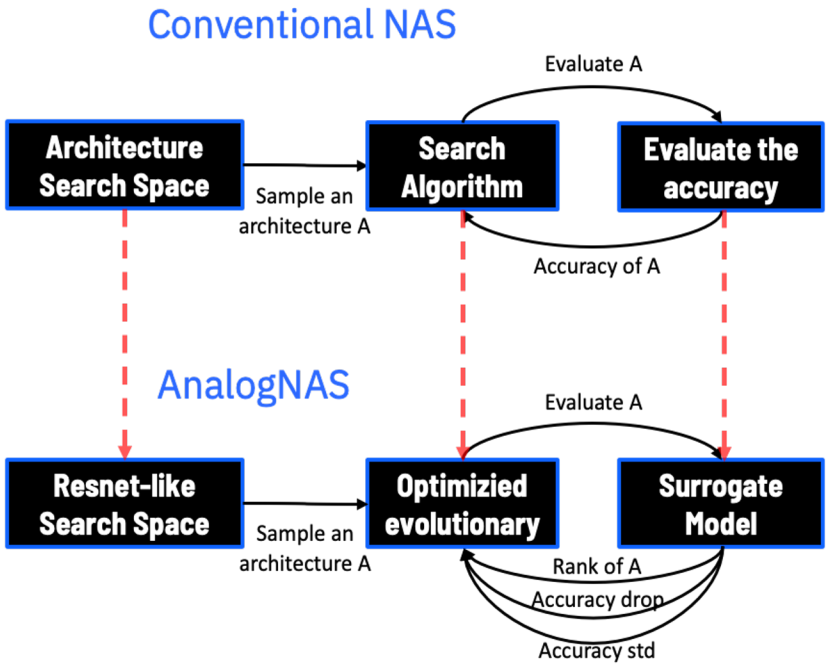
- Large and flexible search space
- Diverse architectures including ResNext, Wide Resnet and standard Resnets.

Surrogate Model

- Trained offline.
- Uses Analog hardware-aware training using aihwkit to construct the dataset.
- Given a sampled architecture, it predicts:
  - 1-day accuracy
  - 1-day accuracy standard deviation
  - Accuracy Variation for 1-month (AVM)

Evolutionary Search Strategy

- Using the surrogate model, the search strategy looks for accurate and robust architecture for a given task.



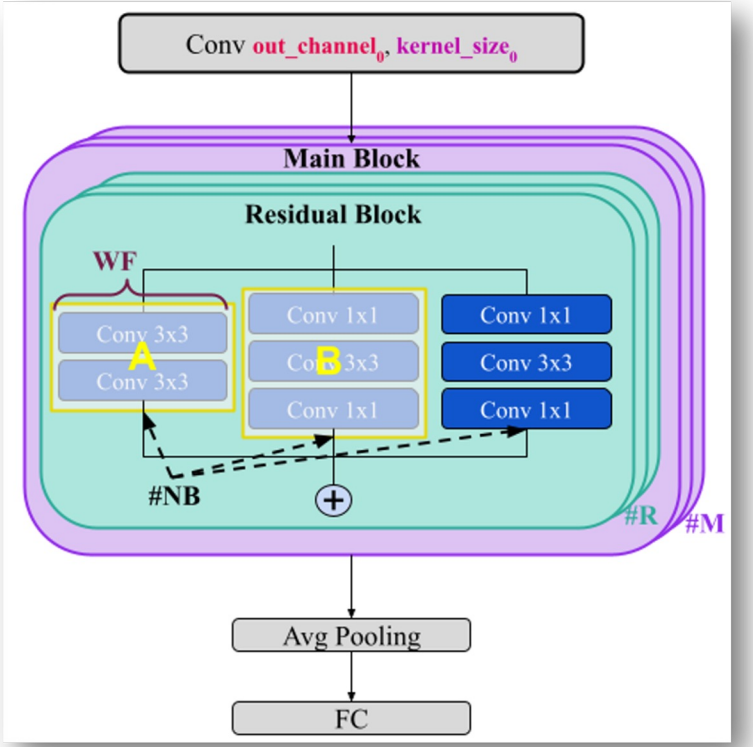
Resnet-like Search Space

Why Resnet-like Search Space ?

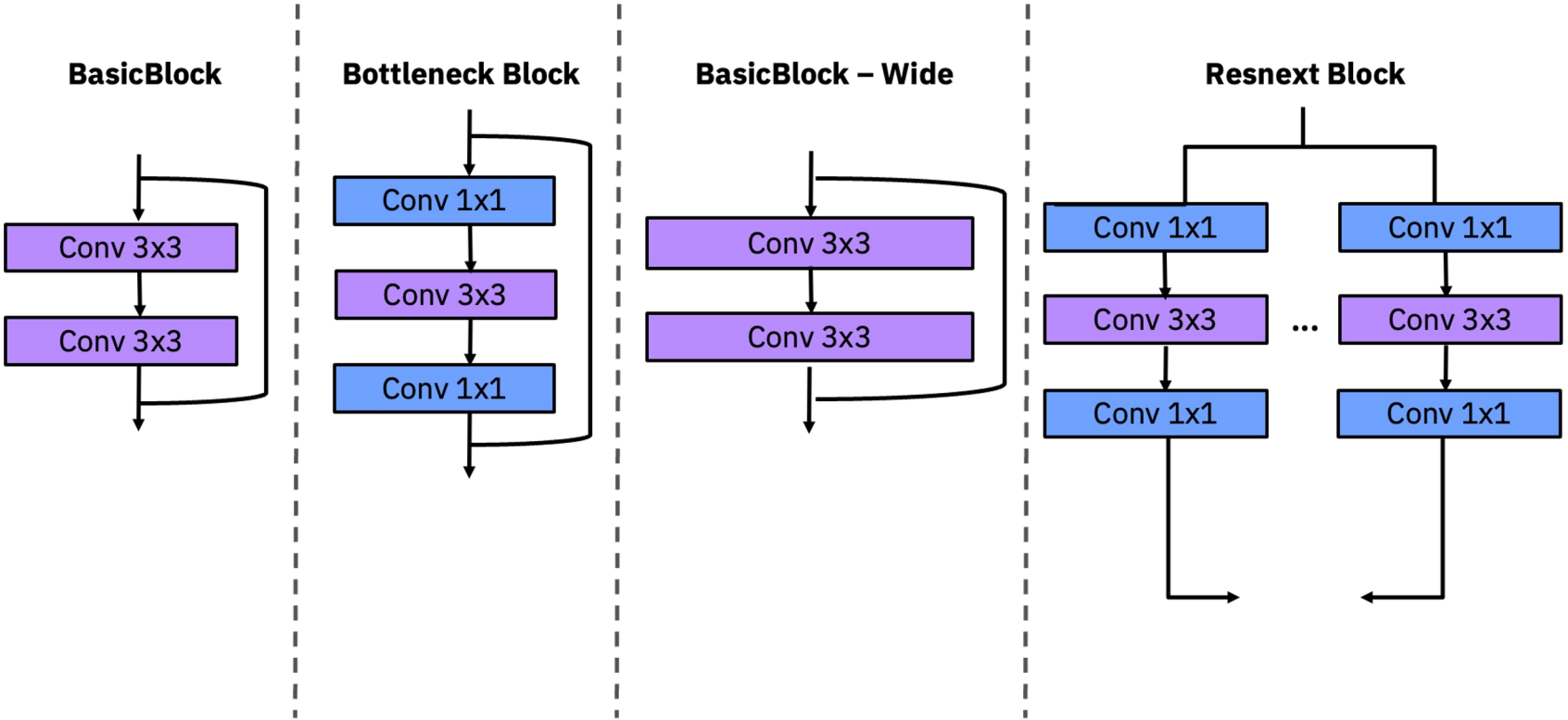
- ✓ *State of the art ConvNets comprises Resnets and its variants and EfficientNet.*
- ✓ *Ability to tackle the three targeted tasks: Image Classification, Keyword Spotting and Visual Wake Words.*
- ✓ *Amenable to Analog Hardware.*

Is this search space diverse enough ?

- ✓ Includes Standard Resnets (18,20,32,34,44,...)
- ✓ Includes Wide-Resnets
- ✓ Includes Resnexts
- ✓ Size of the search space: 73B architectures !!!



Case study: Analog-Aware Neural Architecture Search



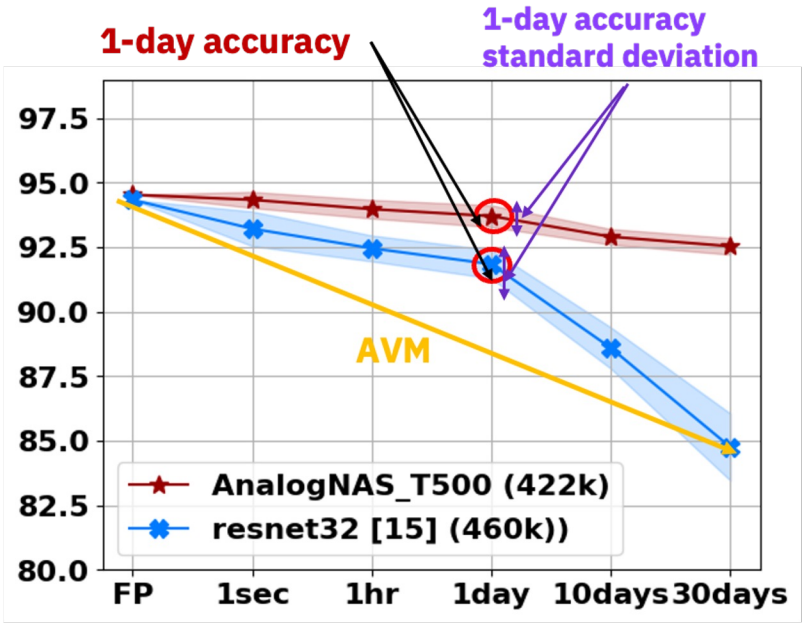
Objectives

What metrics are important to evaluate an Analog Model ?

1. **The 1-day accuracy** measures the performance of an architecture on a given dataset.

2. **The Accuracy Variation over one Month (AVM):** computes the difference between the 1-month and 1-sec accuracy. This objective is essential to measure the robustness over a fixed time duration.

3. **The 1-day accuracy standard deviation** measures the variation of the architecture's performance across experiments. A lower standard deviation indicates that the architecture produces consistent results on hardware deployments



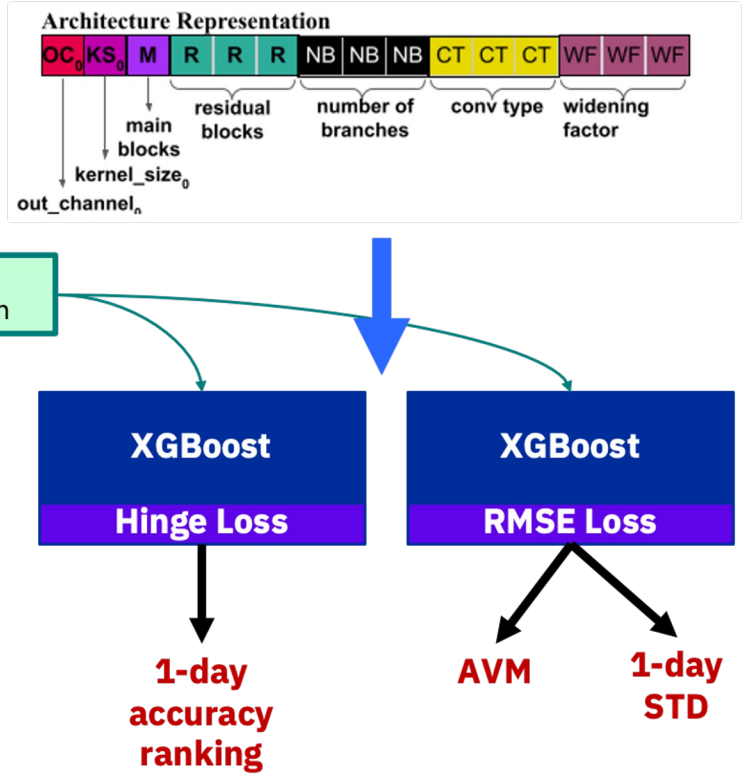
Surrogate Model

Input features:
- Architectural features
- RPU Configuration

Dataset:
- The surrogate model is task-specific.
- Overall, 1230 architectures were extensively trained to create the dataset.
- 80% train, 20% valid
- ~300 more for testing

Training loss
Hinge loss with m=0.1,
varied it [0.05, 0.1, 0.15, 0.2]

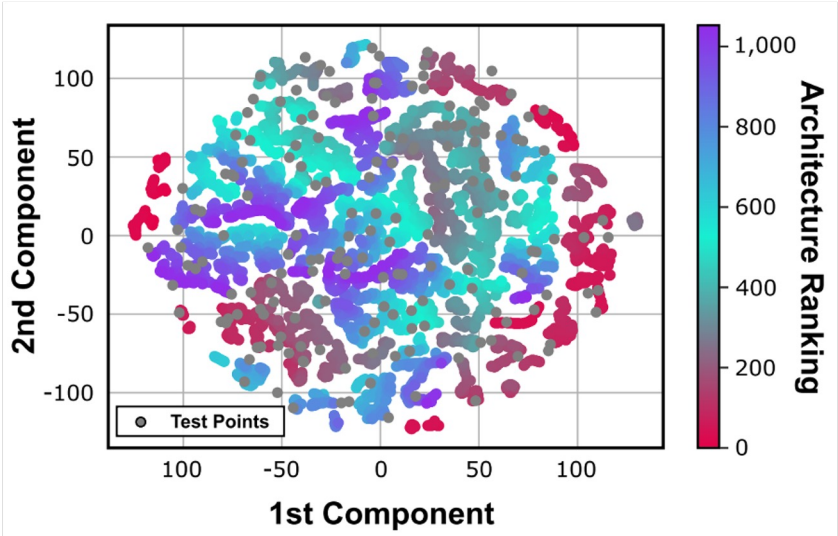
$$L(\{a_j, y_j\}_{j=1,\cdots,N}) = \sum_{j=1}^{N} \sum_{i, y_i > y_j} \max[0, m - (\mathrm{P}(a_i) - \mathrm{P}(a_j))]$$



Surrogate Model

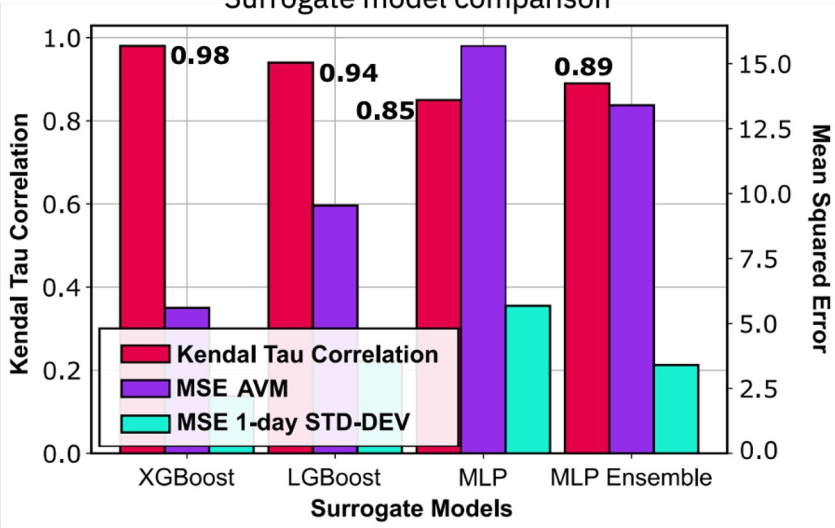
Surrogate model dataset is general and represents a large portion of the explorable search space

T-sne visualization of the sampled architectures for CIFAR-10



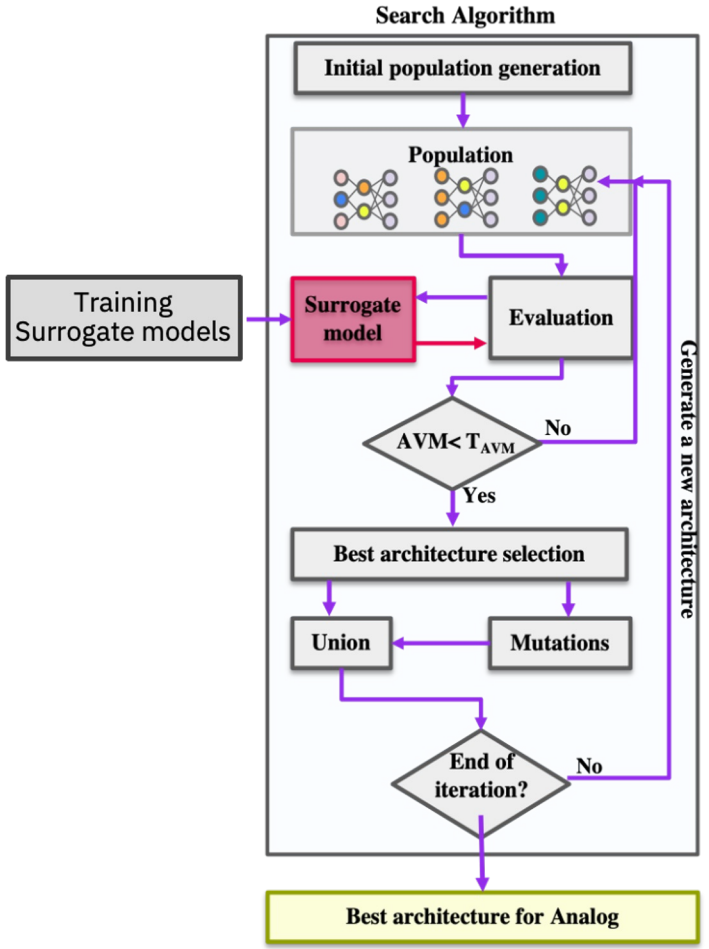
XGBoost enables a more accurate ranking and prediction of the different objectives

Surrogate model comparison



AnalogNAS Search Algorithm

- ✓ *The **evolutionary algorithm** involves two steps:*
  - ➢ *Selection*
    - ➢ *chose the best-performing individuals to move on to the next generation.*
  - ➢ *Mutation*
    - ➢ *Randomly perturbed the architecture to explore new regions of the search space.*



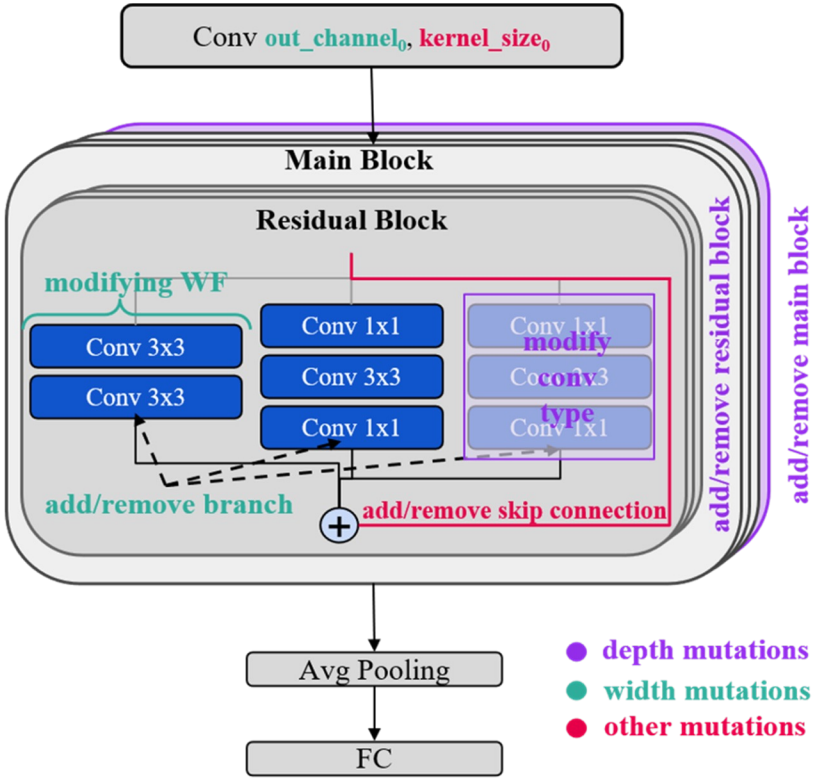
AnalogNAS Search Algorithm

Evolutionary Search Mutations

Depth Mutations:
- Add or remove a Main block
- Add or remove a residual block
- Modify the convolution type (A/B)

Width Mutations:
- Add or remove a branch
- Modify the initial output channel
- Modify the widening factor

Other Mutations:
- Add or remove a residual connection
- Modify initial kernel size



Search Algorithm

Problem formulation

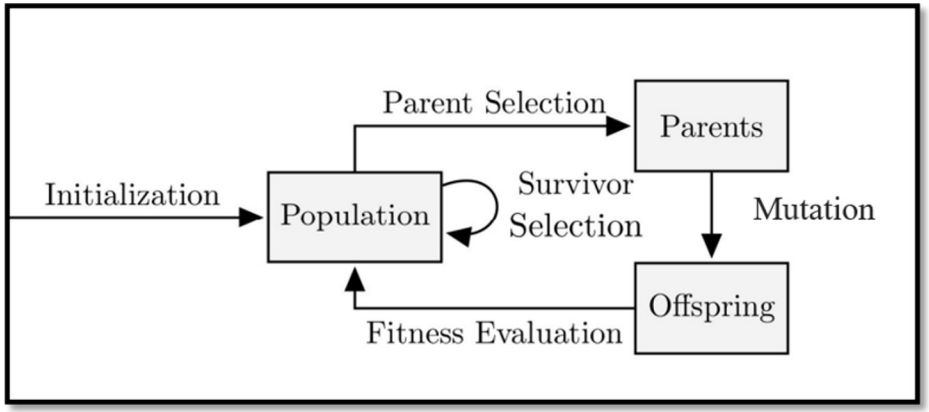
$$\max_{\alpha \in S} \frac{ACC(\alpha)}{\sigma(\alpha)}$$

s.t $\varphi(\alpha) < T_p$

$AVM(\alpha) < T_{AVM}$

- $ACC(\alpha)$ → 1-day accuracy
- $\sigma(\alpha)$ → 1-day accuracy STD
- $\varphi(\alpha) < T_p$ → Number of parameters threshold
- $AVM(\alpha) < T_{AVM}$ → AVM threshold

Hyperparameter	Value
Number of Iterations	200
Time limit	1hour
Population size	100
Mutation Probability	[0.8,0.8,0.6]
Selection	80
Nb_param_threshold	User defined

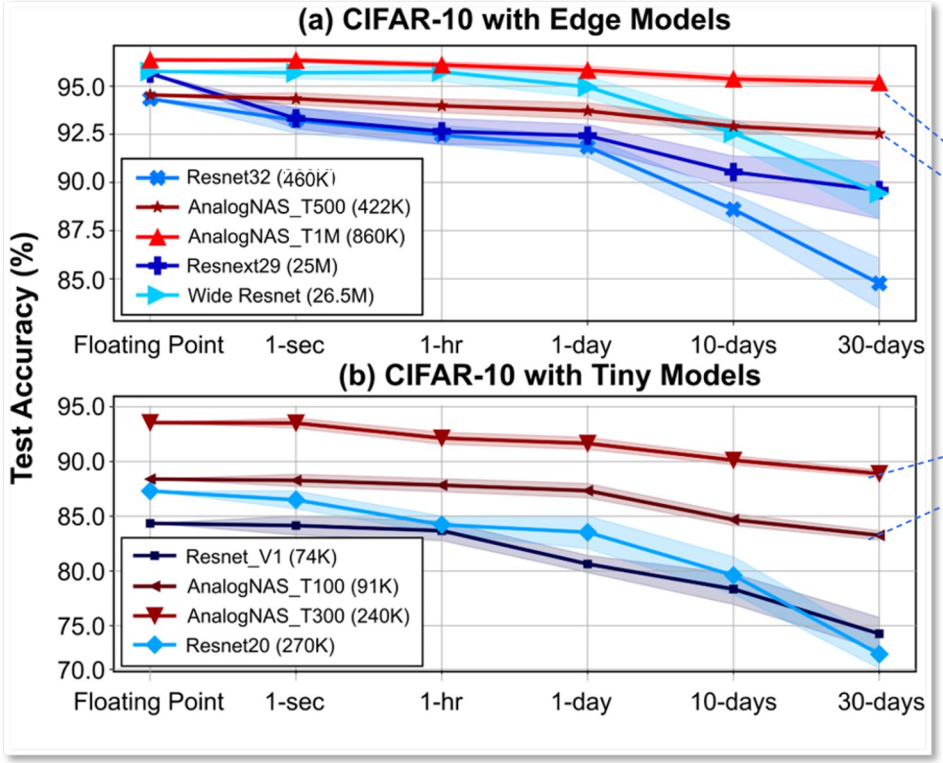


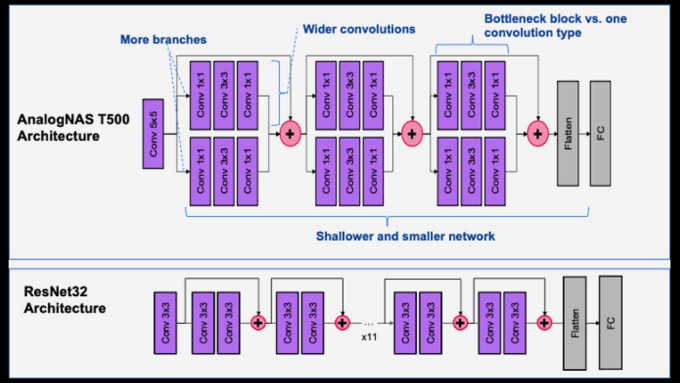
Typical Policies for Parent Selection and Offspring Retention

Tournament, Fitness-proportionate selection, Youngest

AnalogNAS – Results

In an average of 20min, our search strategy is able to find SOTA results.





AnalogNAS automatically generates neural network architectures that exhibit better resilience to drift.