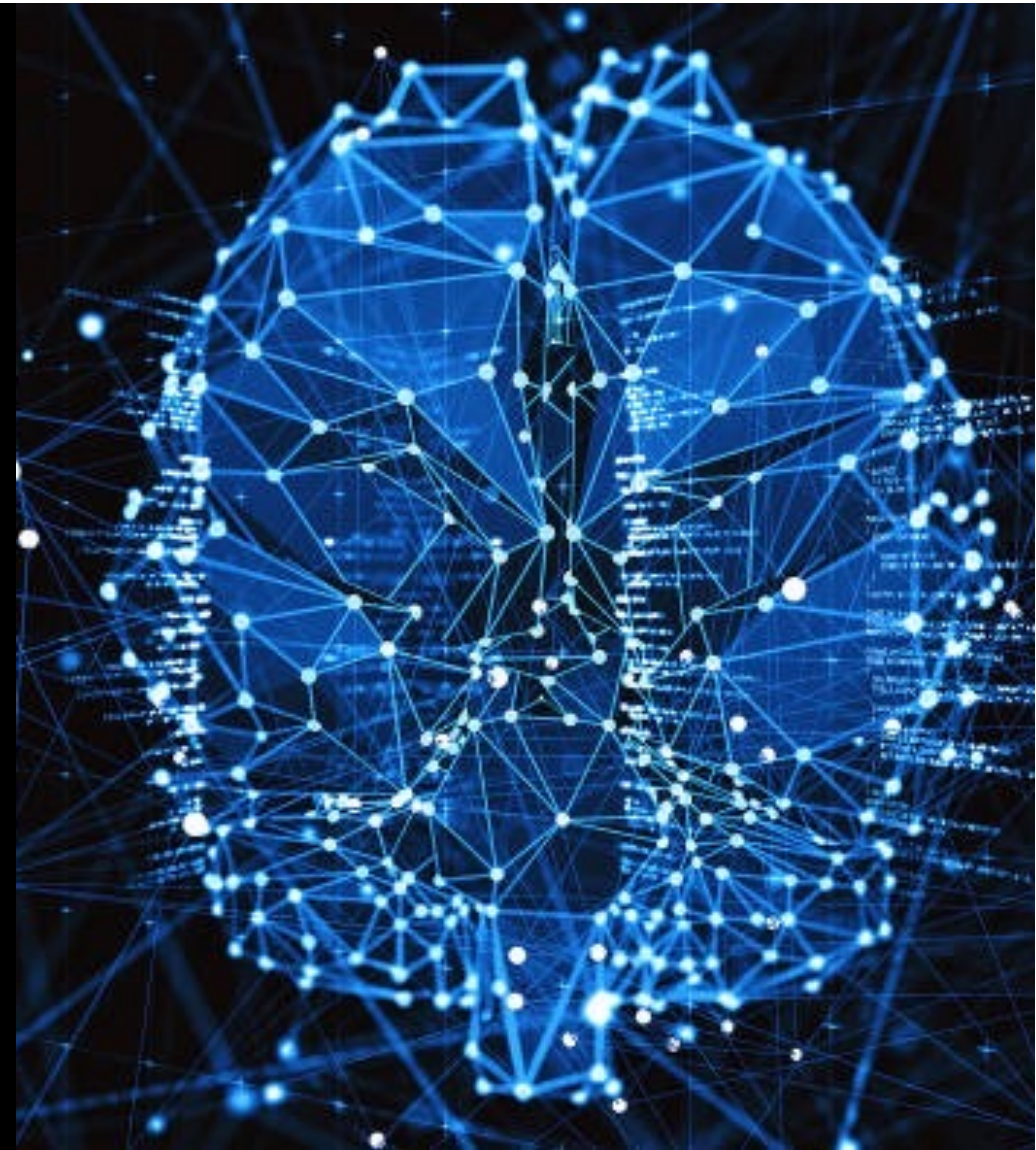


Hardware-aware Automated AI for Efficient Deep Learning across Hybrid Deployments: Current Landscape and Future Directions

Dr. Kaoutar El Maghraoui
Principal Research Scientist
AI Hardware Center Testbed Leader
IBM Research AI

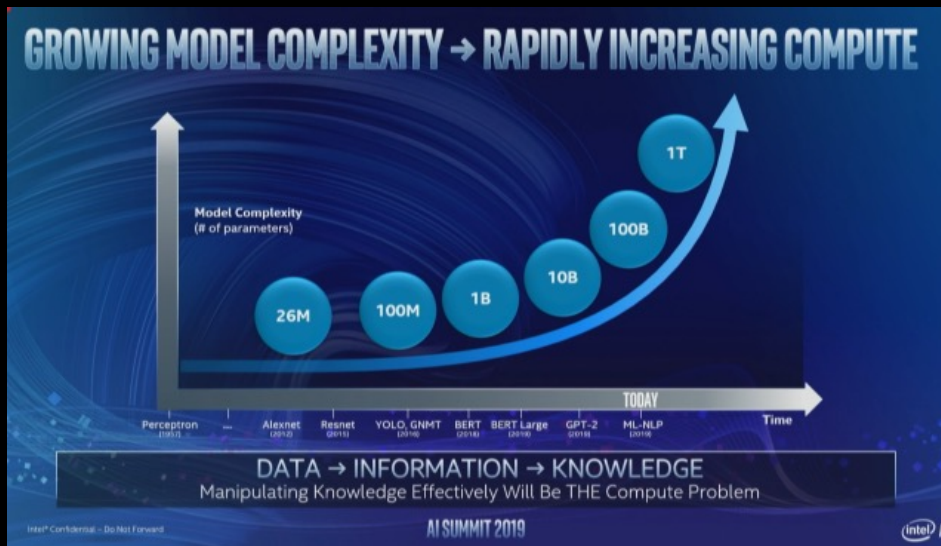
March 28, 2021
@FASTPATH 2021



The Need for Efficient Deep Learning

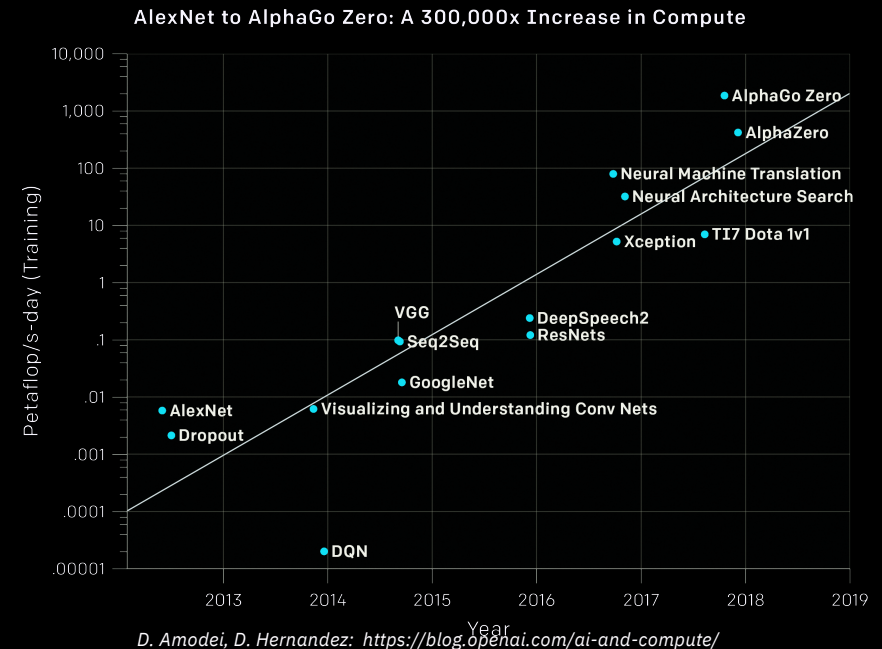
The Accelerating complexity of AI Models

The number of parameters in neural networks models is increasing on the order of 10x year on year.



Unbounded computational demands

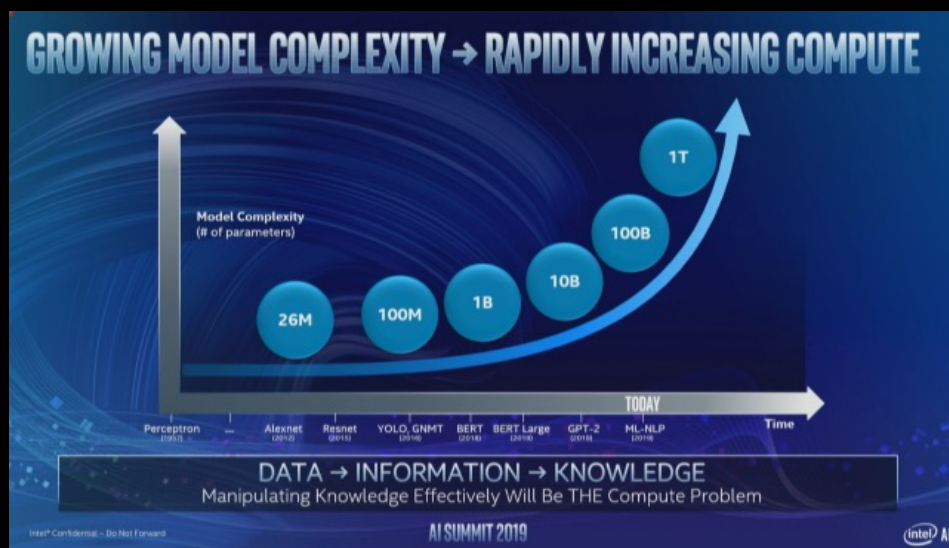
Training requirements are doubling every 3.5 months



The Need for Efficient Deep Learning

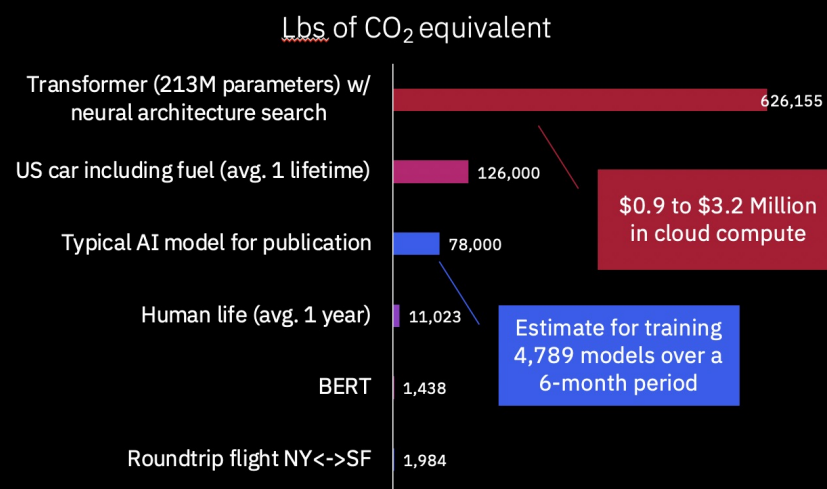
The Accelerating complexity of AI Models

The number of parameters in neural networks models is increasing on the order of 10x year on year.



Increased Carbon Footprint

Training a single model can emit as much as carbos as 5 cars in their lifetimes



Source: <https://www.technologyreview.com/2019/06/06/239031/training-a-single-ai-model-can-emit-as-much-carbon-as-five-cars-in-their-lifetimes/>

Edge Intelligence on The Rise but with Many Challenges

Resource constrained devices

Deploying models on a fleet of devices is not easy

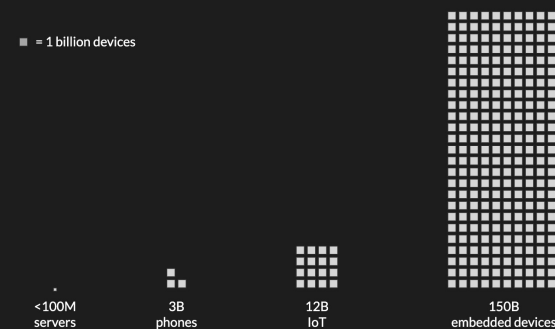
Challenges

Connectivity is not stable and guaranteed

Privacy: data cannot leave the device in many cases

Need to shift from state-of-the-art accuracy to state-of-the-art efficiency

Most intelligence will be at the edge.



Inference at the Edge



IoT

100 mW

(< few 10 GOps)

< few mm²

Single AI Core

Lower accuracy
permissible



Mobile

250 mW to <2W

(< 100's of GOps)

5-10 mm²

Few AI Cores

Accuracy
important



Automotive

20 - 50W

(10's - 100's of TOPs)

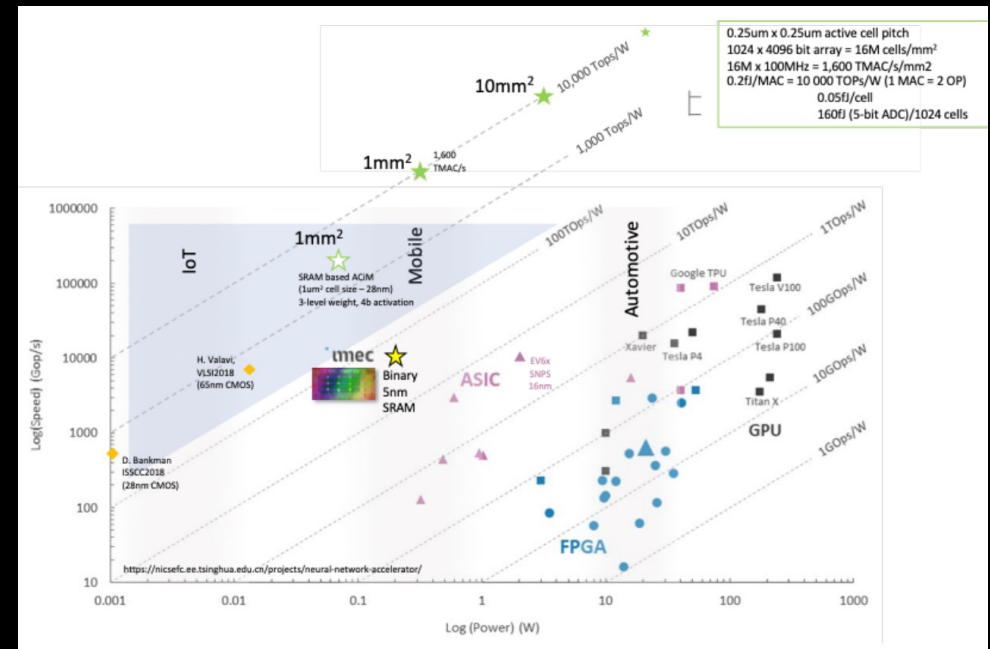
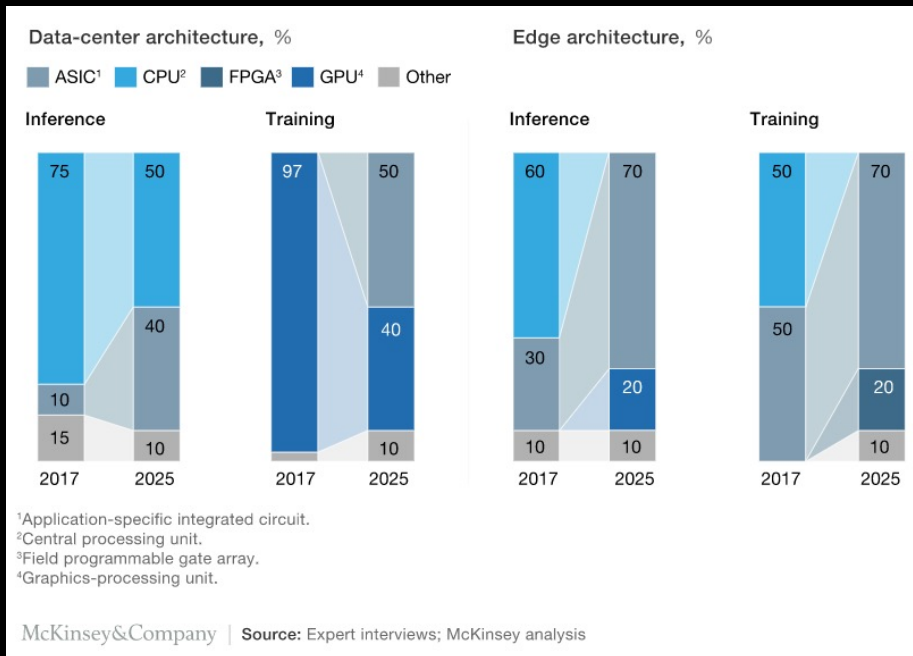
100 - 250 mm²

Multiple AI Cores+
Custom Interconnect

No loss of accuracy is
acceptable

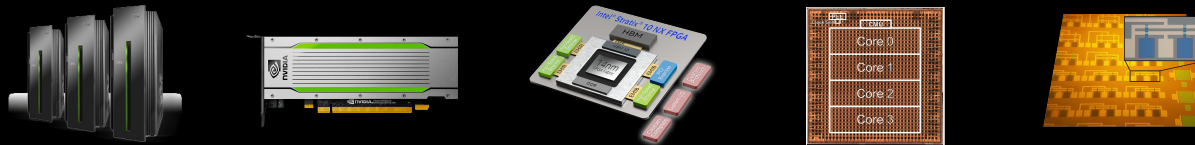
For Inference, across different domains, TOP per Watt is the key metric
Larger Model => More Memory References => More Energy

AI Hardware Trends

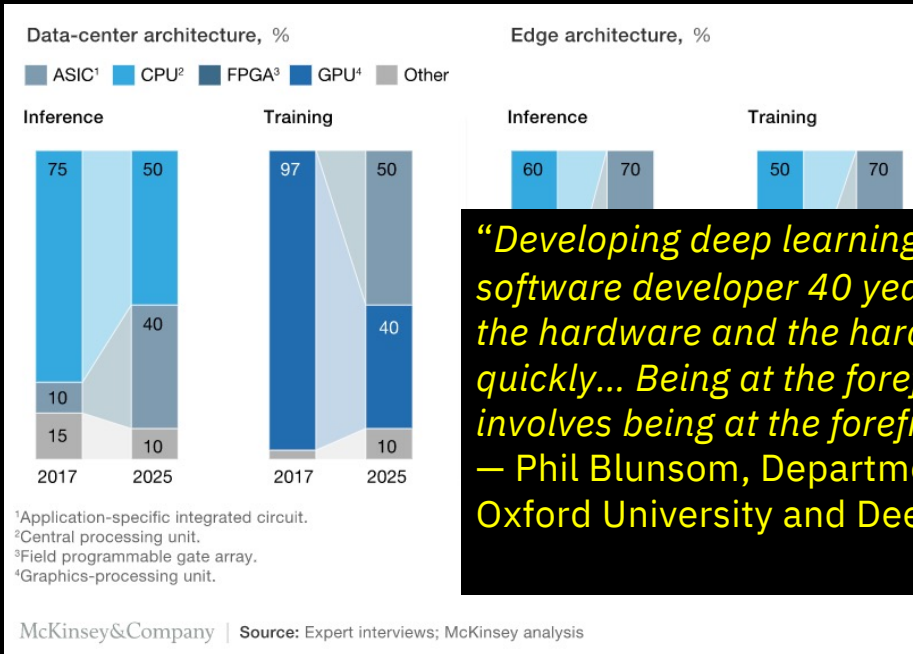


AI ASICs expect to have the biggest growth

The optimal compute architecture varies by use case



AI Hardware Trends

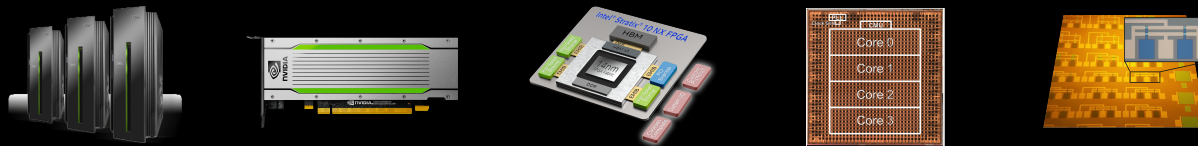


“Developing deep learning models is a bit like being a software developer 40 years ago. You have to worry about the hardware and the hardware is changing quite quickly... Being at the forefront of deep learning also involves being at the forefront of what hardware can do.”
 — Phil Blunsom, Department of Computer Science at Oxford University and DeepMind



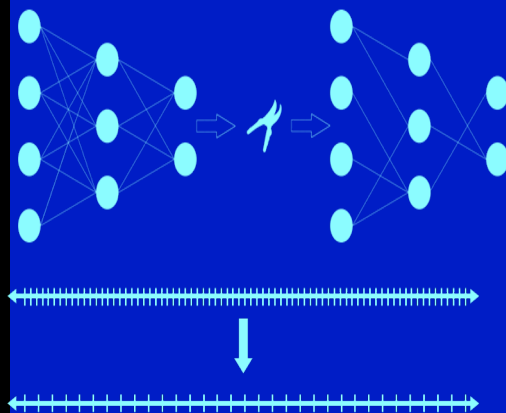
AI ASICs expect to have the biggest growth

The optimal compute architecture varies by use case



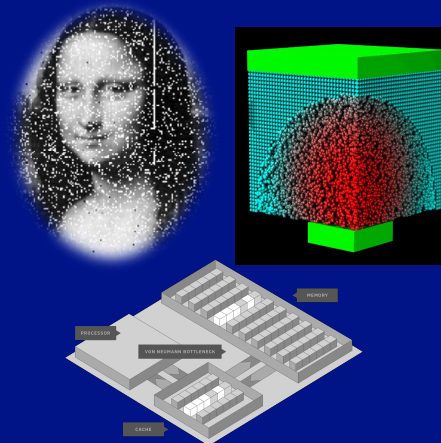
Building Efficient Neural Networks

Model Efficiency



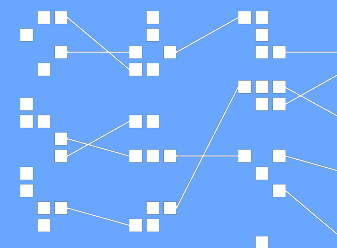
Design accurate and efficient neural networks

Hardware Efficiency



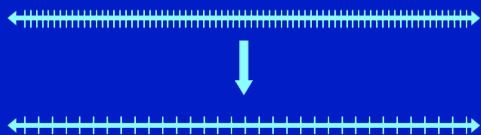
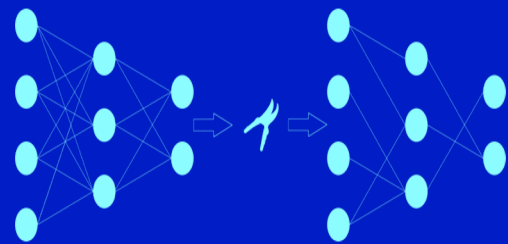
Purpose-built AI hardware

Design Efficiency



Automate the design of efficient neural networks

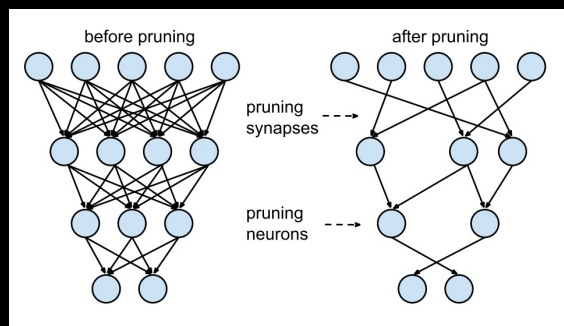
Model Efficiency



Design accurate and efficient neural networks

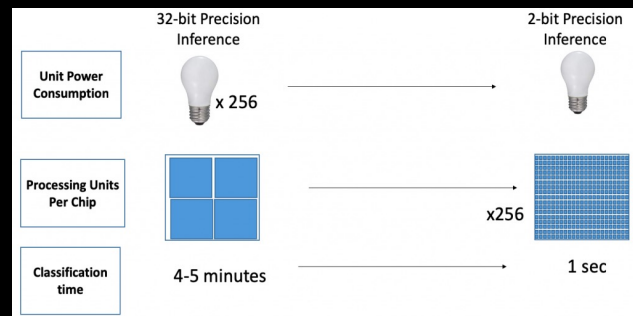
Popular Approaches

Pruning Deep Neural Networks



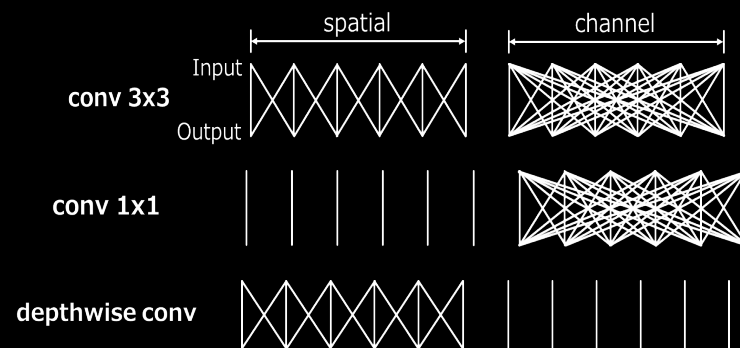
[Lecun et al. NIPS'89] [Han et al. NIPS'15]

Reduced Precision

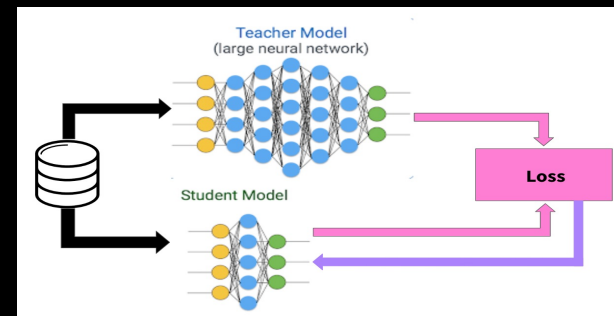


J. Choi et al, NeurIPS 2019

Compact Convolution Filters



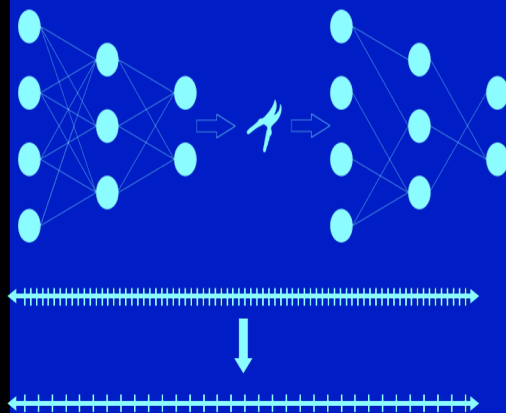
Knowledge Distillation



Hinton et al, arXiv:1503.02531

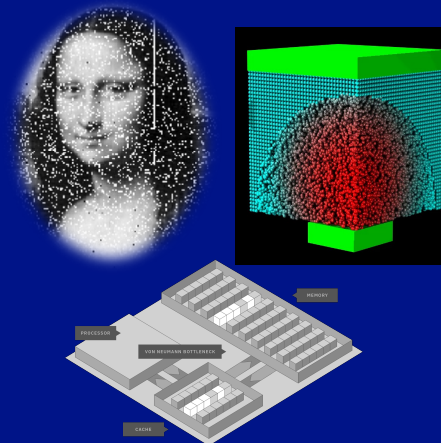
Building Efficient Neural Networks

Model Efficiency



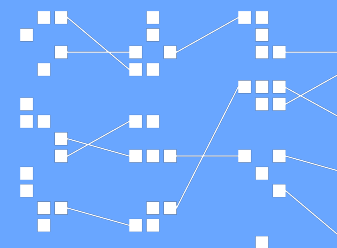
Design accurate and efficient neural networks

Hardware Efficiency



Purpose-built AI hardware

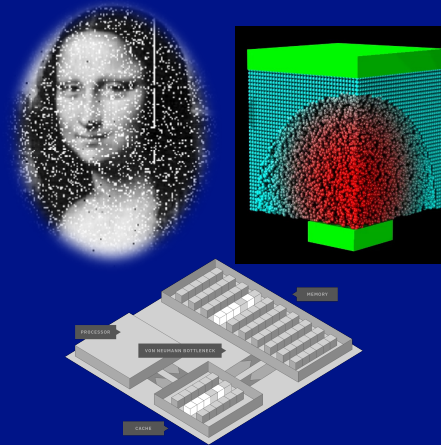
Design Efficiency



Automate the design of efficient neural networks

Building Efficient Neural Networks

Hardware Efficiency



Purpose-built AI hardware

IBM Research AI Hardware Center

**“IBM invests \$2
Billion in New
York Research
Hub for AI”**

Bloomberg

**“IBM Bets \$2B
Seeking 1000X
AI Hardware
Performance
Boost”**



An ecosystem of enterprise and
academic partners

February 7, 2019

Launch Date

\$2B

IBM Investment To Create Artificial
Intelligence Hardware Center

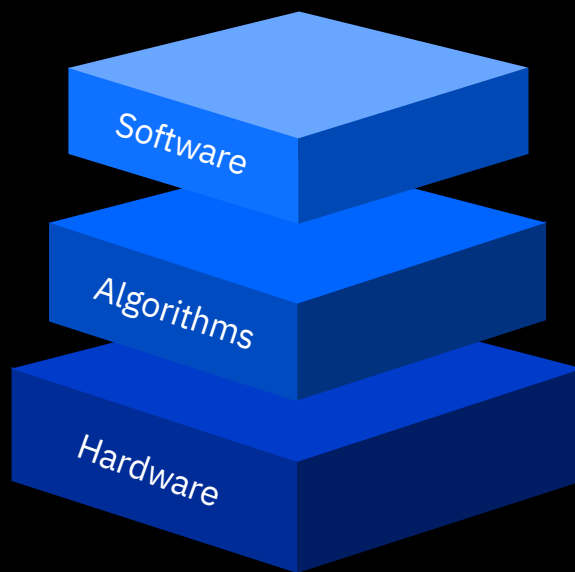
\$300M

New York State investment

16 and growing

Members of the IBM Research
AI Hardware Center

IBM AI Hardware Center Focus

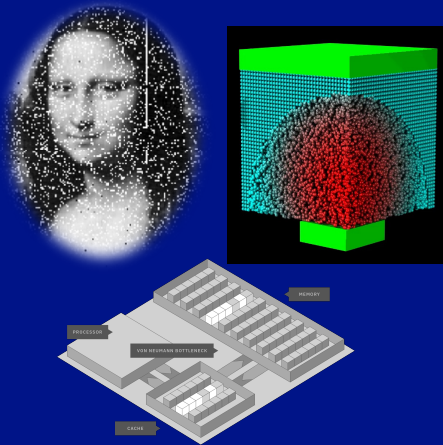


- Compiler Optimizations
- Feedback to hardware design
- Simulation/Emulation for future Hardware
- Reduced precision scaling
- Sparsity exploitation
- Quantization-aware training
- Model Compression
- Hardware-aware Auto-AI
- Digital AI Cores with reduced precision scaling and other architectural tricks
- Analog AI Cores exploiting in-memory compute advances

IBM Hybrid Cloud Infrastructure

IBM Research's Roadmap for AI Hardware

Hardware Efficiency

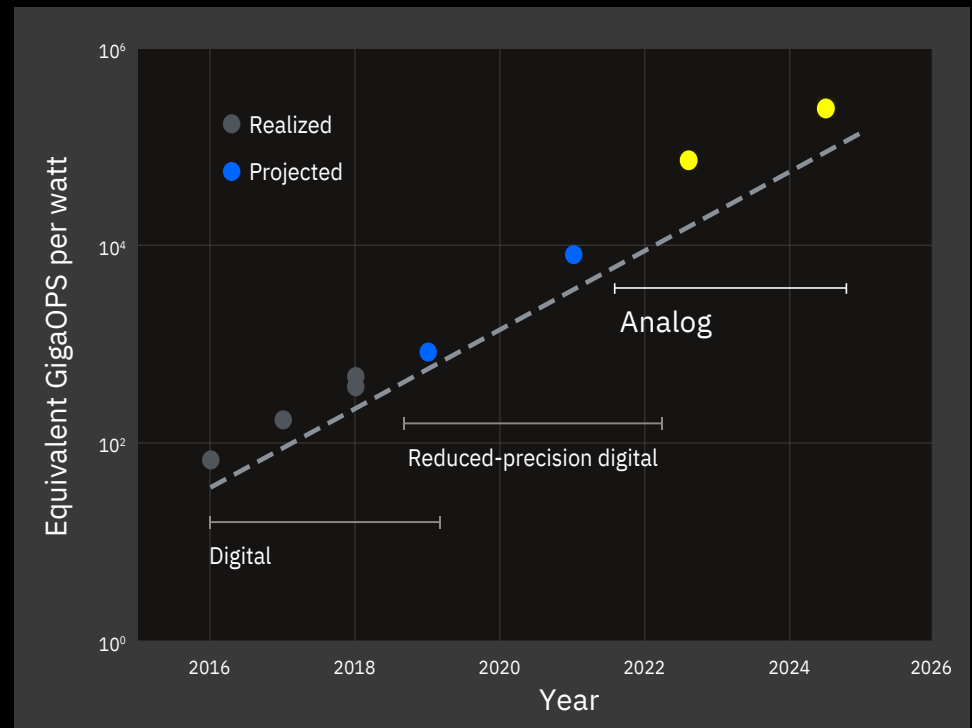


Purpose-built AI hardware

Extending performance by 2.5X / year through 2025

Approximate computing principles applied to **Digital AI Cores** with reduced precision,

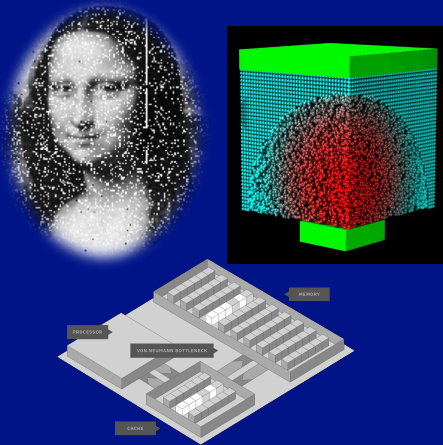
as well as **Analog AI Cores**, which could potentially offer another 100x in energy-efficiency



T. Gokmen and Y. Vlasov, *Frontiers in Neuroscience* **10**, pp. 333, 2016

IBM Research's Roadmap for AI Hardware

Hardware Efficiency

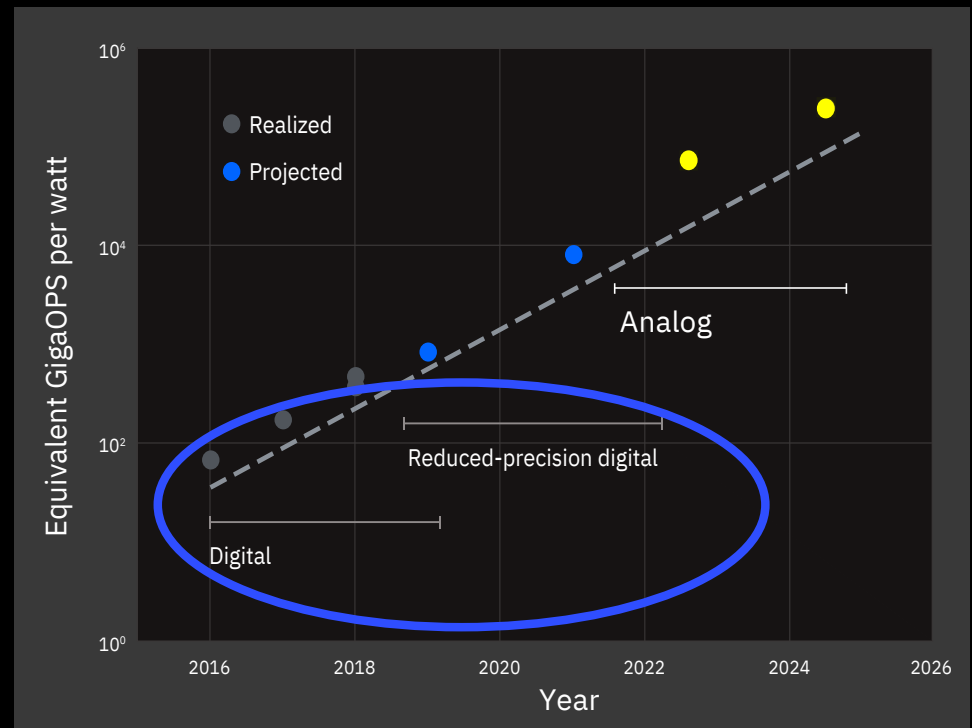


Purpose-built AI hardware

Extending performance by 2.5X / year through 2025

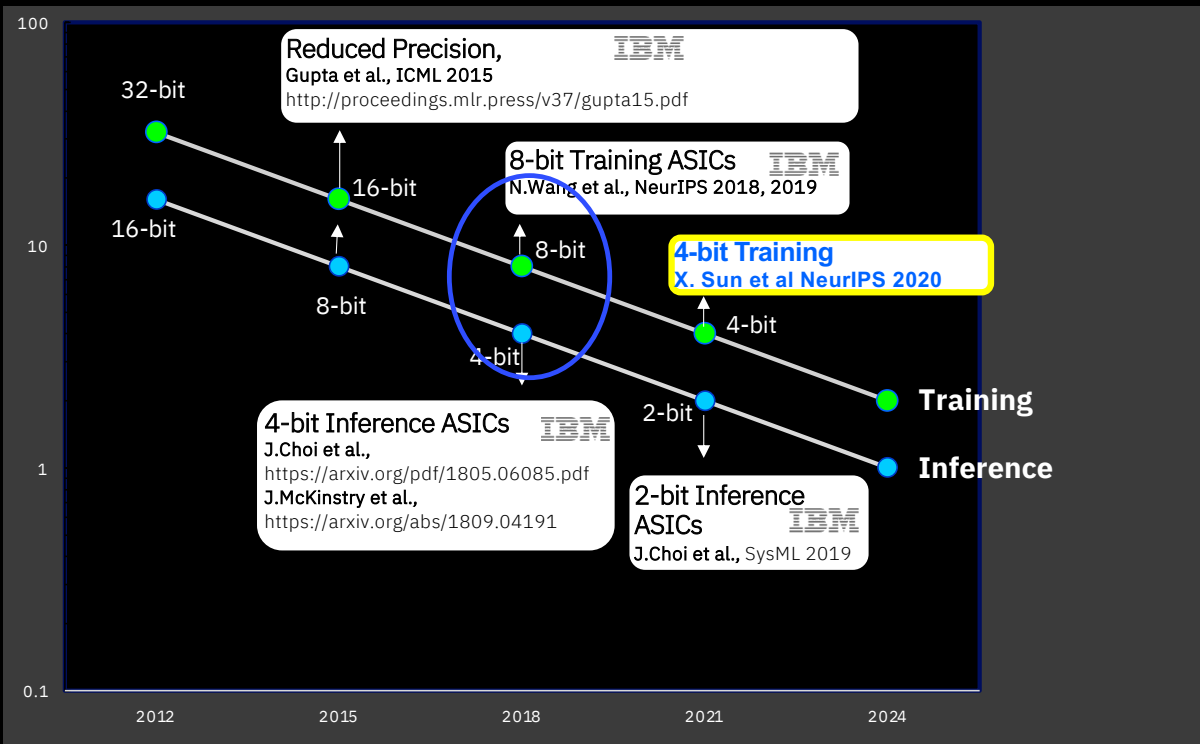
Approximate computing principles applied to **Digital AI Cores** with reduced precision,

as well as **Analog AI Cores**, which could potentially offer another 100x in energy-efficiency



T. Gokmen and Y. Vlasov, *Frontiers in Neuroscience* **10**, pp. 333, 2016

IBM AI Hardware Center Digital AI Cores: IBM Research is leading in reduced precision scaling *with iso accuracy*



- Key advancements in reduced precision arithmetic for AI driven by IBM AI Research team.
- First demonstration of 16-bit precision for Deep Learning Training (ICML 2015).
- Demonstration of world's first 8-bit training (NeurIPS 2018, NeurIPS 2019), and **world's first 4-bit training (NeurIPS 2020)**.
- Demonstration of highly accurate 2-bit and 4-bit Inference (SysML 2019)
- ISSCC 2021 : **IBM introduced the AI chip leading in precision scaling (8-bit training and 4-bit inference)**

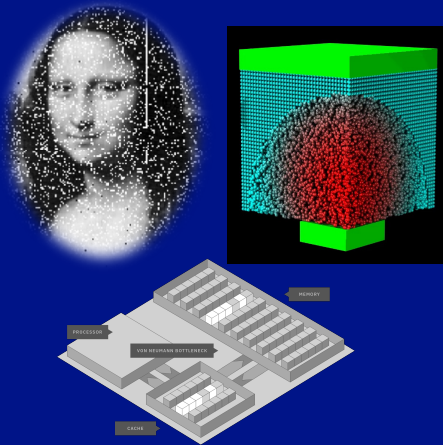
For reference - Industry standard for training:

- GPU default: 32 bit
- GPU accelerated: 16 bit (V100 & A100)
- TPU: 16 bit (Bfloat)

<https://www.ibm.com/blogs/research/2021/02/ai-chip-precision-scaling/>

IBM Research's Roadmap for AI Hardware

Hardware Efficiency

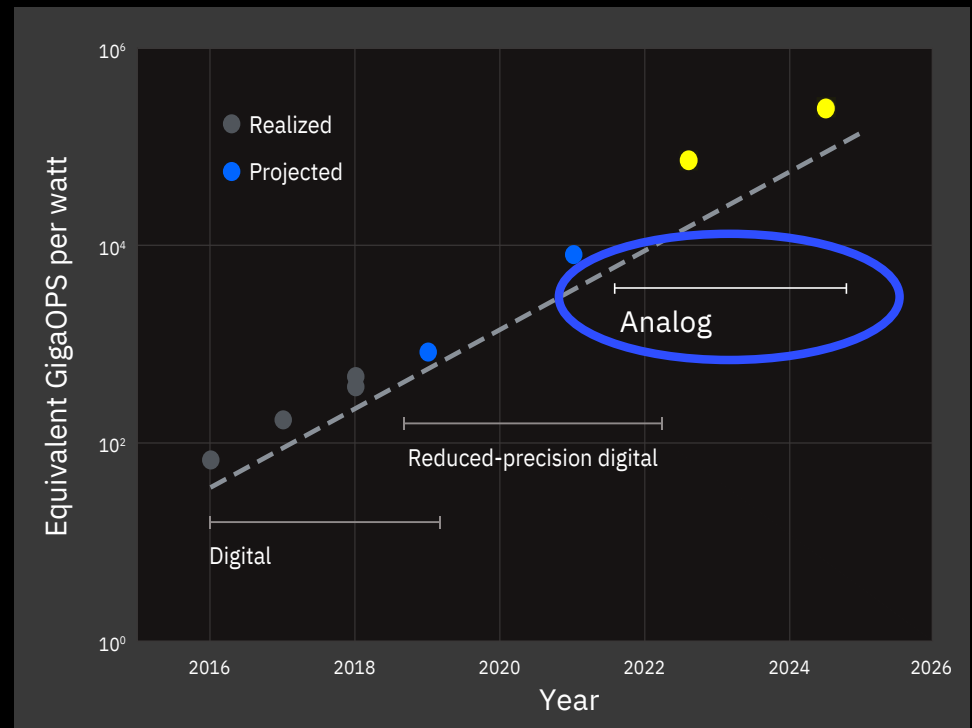


Purpose-built AI hardware

Extending performance by 2.5X / year through 2025

Approximate computing principles applied to **Digital AI Cores** with reduced precision,

as well as **Analog AI Cores**, which could potentially offer another 100x in energy-efficiency

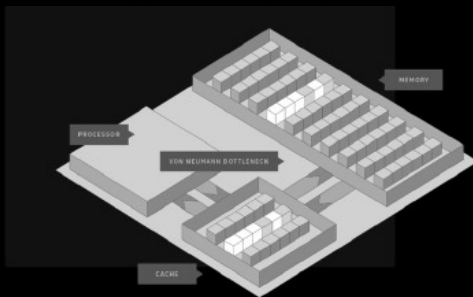


T. Gokmen and Y. Vlasov, *Frontiers in Neuroscience* **10**, pp. 333, 2016

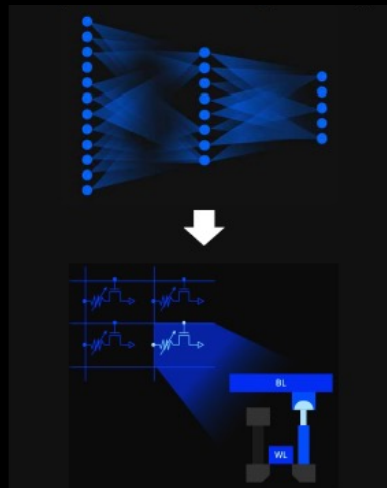
Analog NVM for in-memory compute

Eliminate the Von-Neumann bottleneck

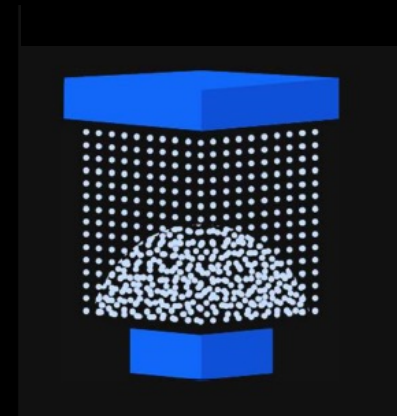
Perform computation directly in memory



Map DNNs to analog cross-point arrays

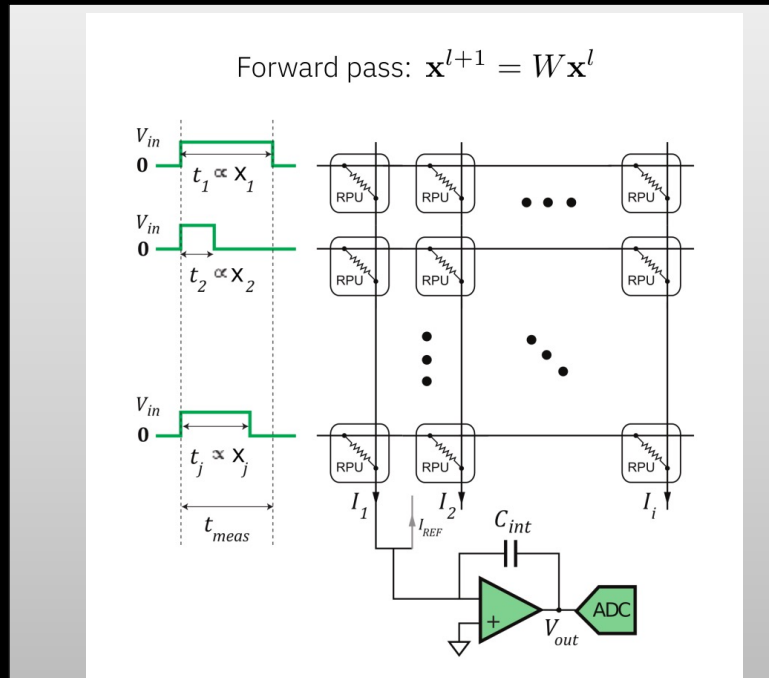


NVM materials in array crosspoints to store weights

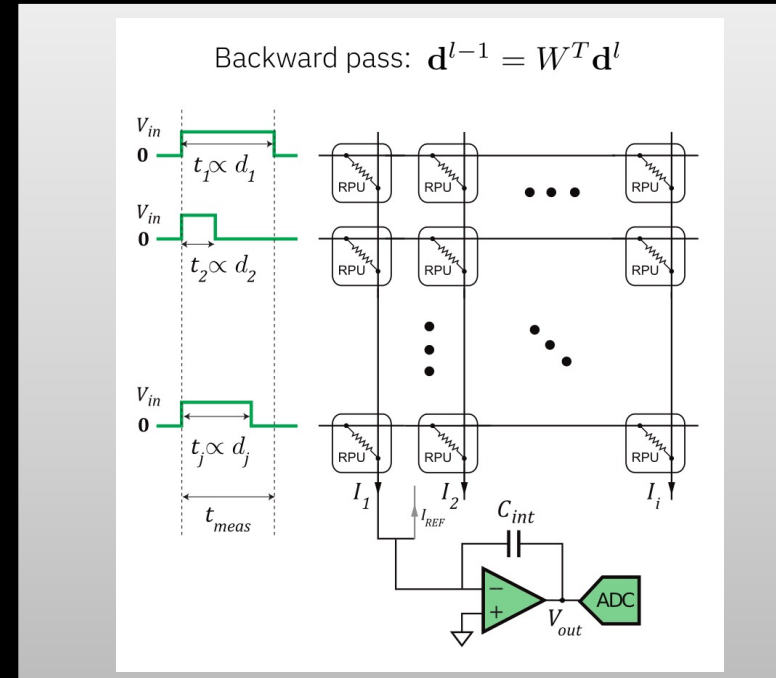


Analog AI

Fully-connected layer with resistive crossbar array



Addition: Kirchhoff's law
 Multiplication: Ohm's



(backward only needed for training)

IBM Analog Hardware Acceleration Kit

<https://analog-ai.mybluemix.net/>

IBM Analog Hardware Acceleration Kit

The Open Source python toolkit for exploring and using the capabilities of in-memory computing devices in the context of artificial intelligence.

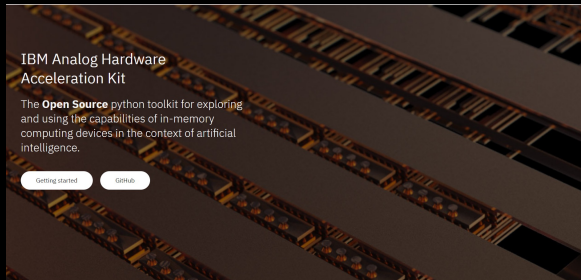
Getting started

GitHub



Analog AI Hardware Acceleration Toolkit

<https://analog-ai.mybluemix.net/>



Current Capabilities Include:

- Simulate analog MACC operation including analog backward/update pass
- Simulate a wide range of analog AI devices and crossbar configurations by using abstract functional models of material characteristics with adjustable parameters
- Abstract device (update) models
- Analog friendly learning rule
- Hardware-aware training for inference capability
- Inference capability with drift and statistical (programming) noise models

Install Analog AI Hardware Acceleration Kit

```
$ pip install aihwkit
```

Training your Analog Model

```
from torch import Tensor
from torch.nn.functional import mse_loss

# Import the aihwkit constructs.
from aihwkit.nn import AnalogLinear
from aihwkit.optim.analog_sgd import AnalogSGD

x = Tensor([[0.1, 0.2, 0.4, 0.3], [0.2, 0.1, 0.1, 0.3]])
y = Tensor([[1.0, 0.5], [0.7, 0.3]])

# Define a network using a single Analog layer.
model = AnalogLinear(4, 2)

# Use the analog-aware stochastic gradient descent optimizer.
opt = AnalogSGD(model.parameters(), lr=0.1)
opt.regroup_param_groups(model)

# Train the network.
for epoch in range(10):
    pred = model(x)
    loss = mse_loss(pred, y)
    loss.backward()

    opt.step()
    print('Loss error: {:.16f}'.format(loss))
```

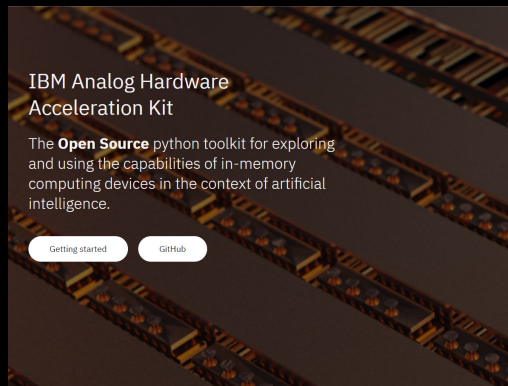
Analog Hardware Acceleration Toolkits

<https://analog-ia.mybluemix.net/>

Open Source Github Library

Released Oct 2020

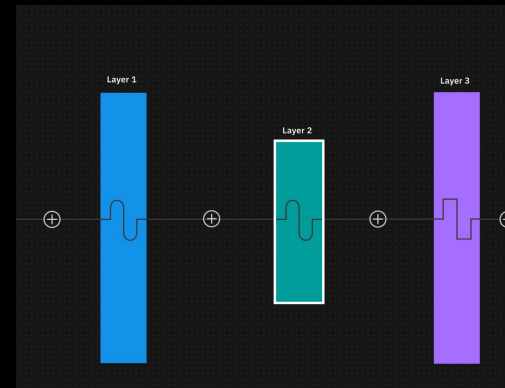
Target AI and hardware developers, ecosystem building



Cloud Experience

To be released SOON

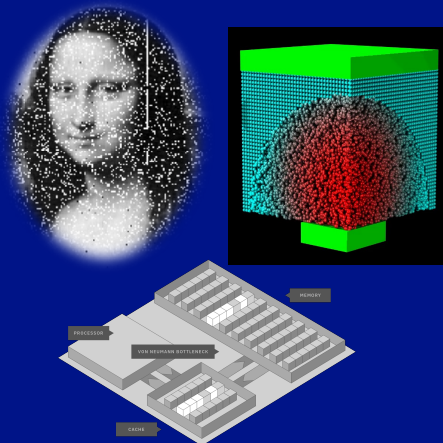
For general public: experiment with analog and neural networks



A roadmap of evolving features to grow the open-source Analog AI ecosystem

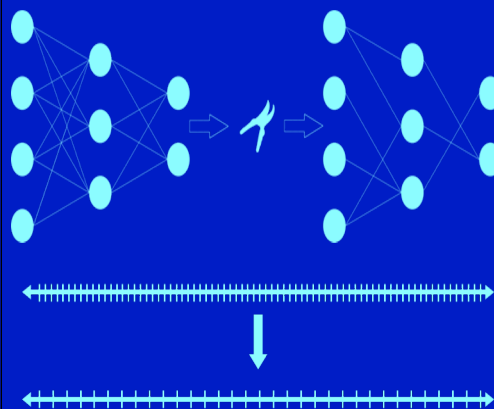
Building Efficient Neural Networks

Hardware Efficiency



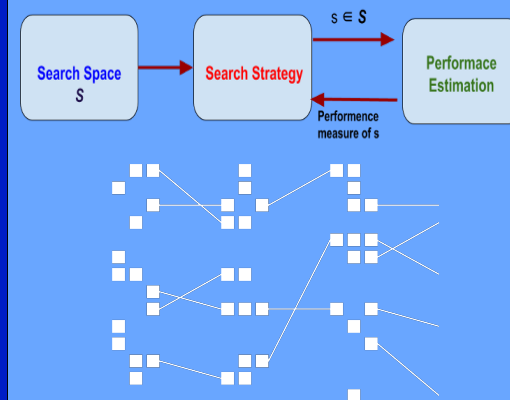
**Purpose-built AI hardware:
Reduced Precision
Digital AI Cores
and Analog**

Model Efficiency



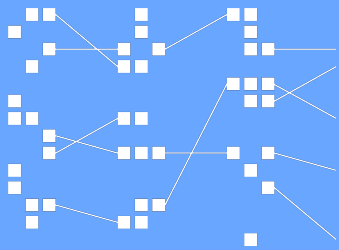
**Design accurate
and efficient
neural networks:
pruning,
quantization, etc.**

Design Efficiency



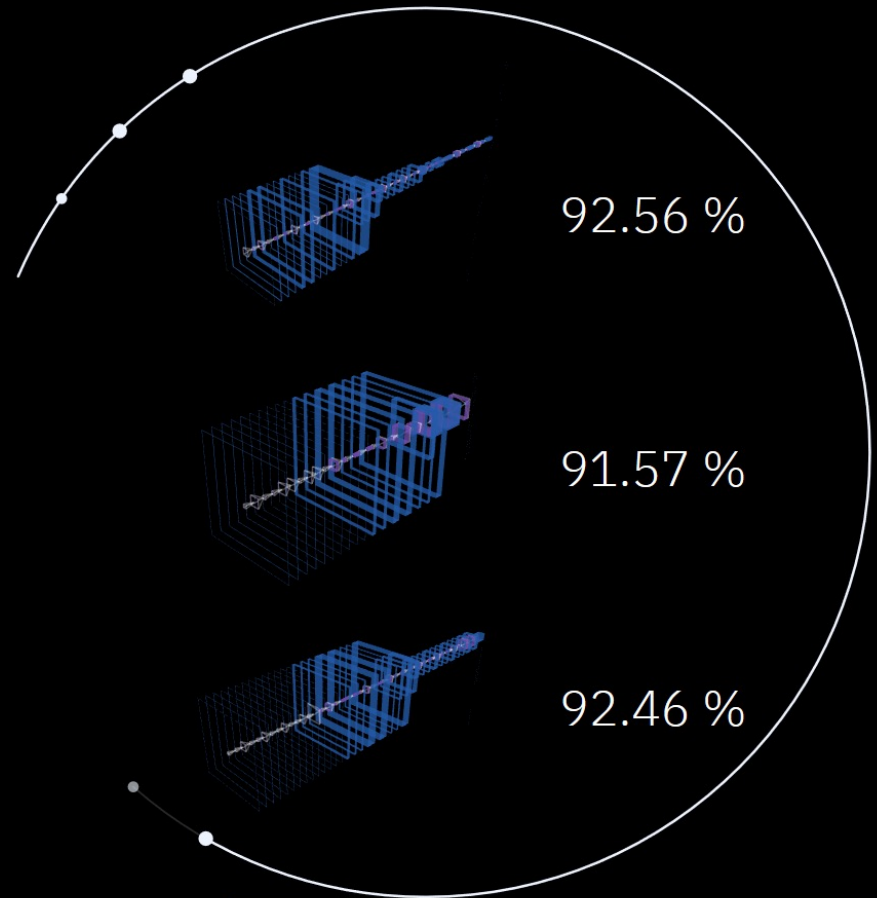
**Automate the design
of efficient neural
networks**

Design Efficiency



Automate the design of efficient neural networks

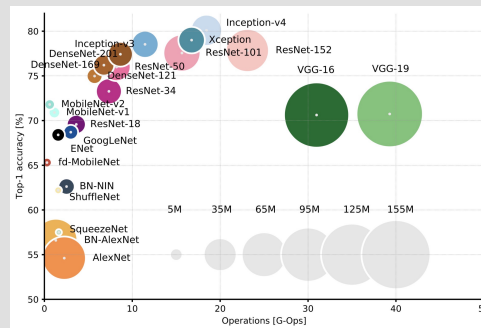
Automation for Model Synthesis for Emerging and Existing AI Hardware



Hardware-aware
Neural Architecture
Search is a key step
towards
democratizing AI

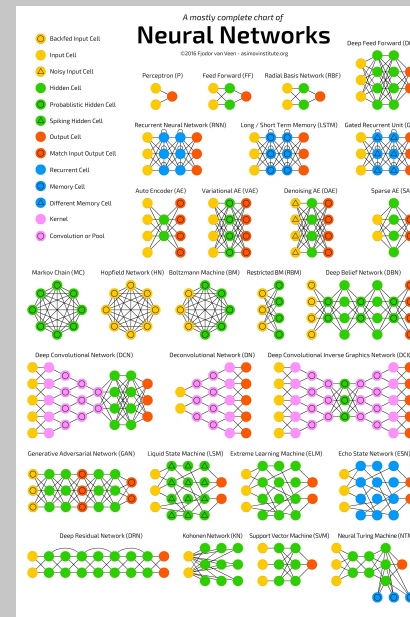
AutoML is catching
up with human
experts

Today's neural
networks are
extremely complex
and keep growing in
size.

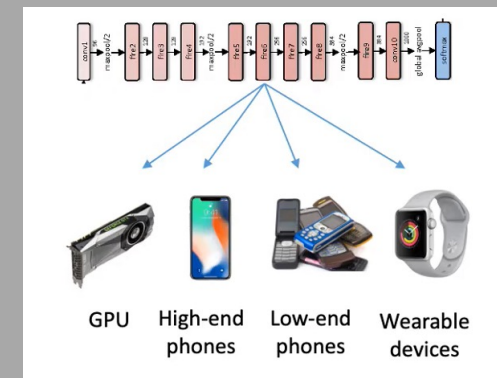


<https://culurciello.medium.com/analysis-of-deep-neural-networks-dcf398e71aae>

Hand-crafting neural
networks is an
expensive, time-
consuming and ad-
hoc process

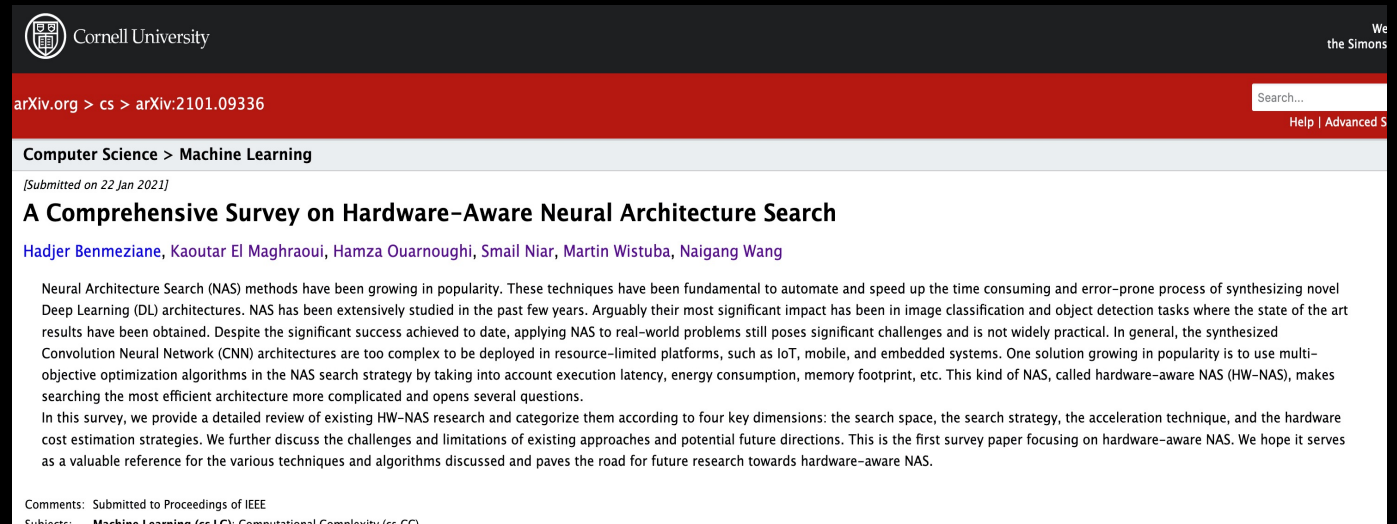


Edge Intelligence is
inevitable and
requires hardware-
efficiency built into
the design of neural
networks. A hard
task for non-
hardware experts.



Key Questions

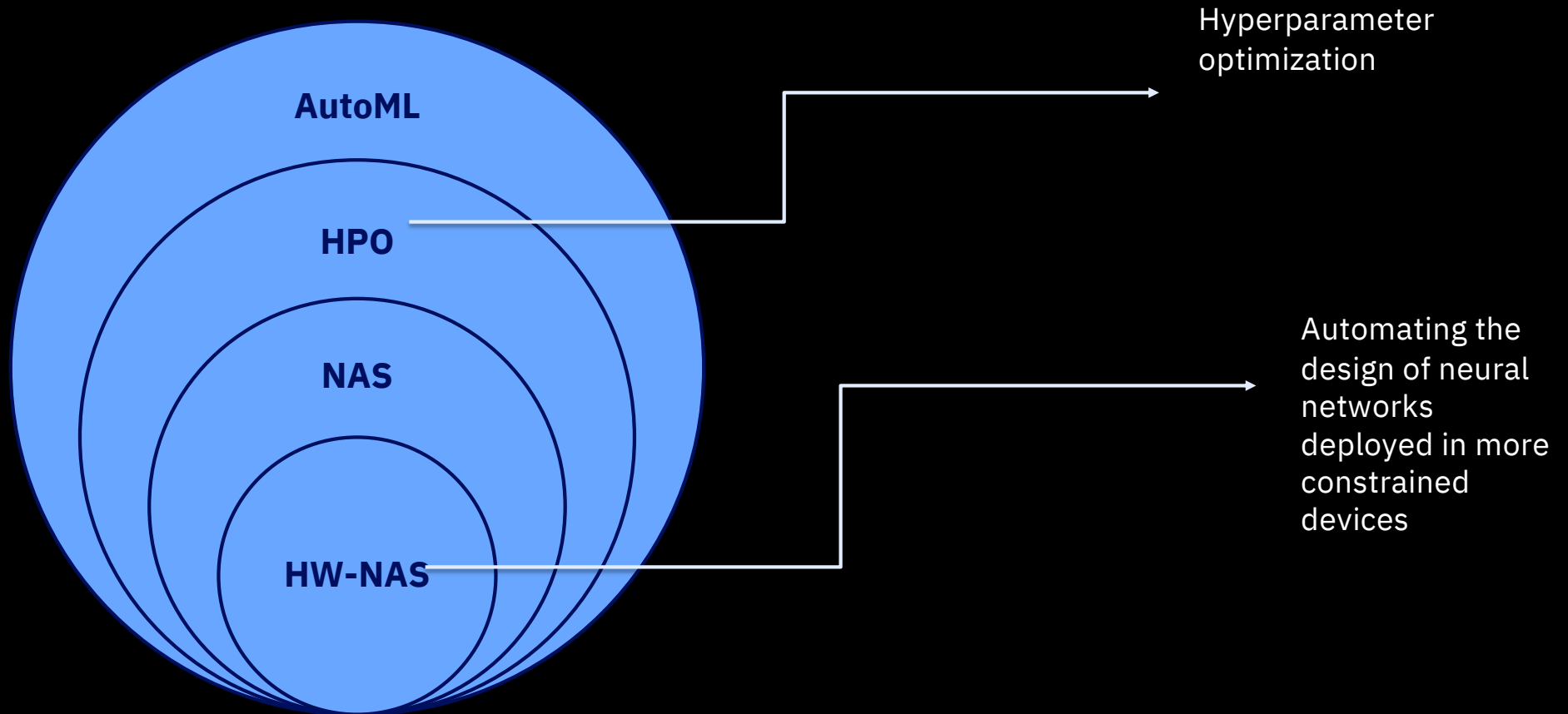
- What the key algorithmic components of HW-NAS?
- What search spaces are more friendly to hardware-aware NAS?
- What are the existing challenges and future directions?



The screenshot shows the arXiv page for the paper "A Comprehensive Survey on Hardware-Aware Neural Architecture Search". The page header includes the Cornell University logo and name. The breadcrumb trail is "arXiv.org > cs > arXiv:2101.09336". The paper title is "A Comprehensive Survey on Hardware-Aware Neural Architecture Search" by Hadjer Benmeziane, Kaoutar El Maghraoui, Hamza Ouarnoughi, Smail Niar, Martin Wistuba, and Naigang Wang. The submission date is [Submitted on 22 Jan 2021]. The abstract text is: "Neural Architecture Search (NAS) methods have been growing in popularity. These techniques have been fundamental to automate and speed up the time consuming and error-prone process of synthesizing novel Deep Learning (DL) architectures. NAS has been extensively studied in the past few years. Arguably their most significant impact has been in image classification and object detection tasks where the state of the art results have been obtained. Despite the significant success achieved to date, applying NAS to real-world problems still poses significant challenges and is not widely practical. In general, the synthesized Convolution Neural Network (CNN) architectures are too complex to be deployed in resource-limited platforms, such as IoT, mobile, and embedded systems. One solution growing in popularity is to use multi-objective optimization algorithms in the NAS search strategy by taking into account execution latency, energy consumption, memory footprint, etc. This kind of NAS, called hardware-aware NAS (HW-NAS), makes searching the most efficient architecture more complicated and opens several questions. In this survey, we provide a detailed review of existing HW-NAS research and categorize them according to four key dimensions: the search space, the search strategy, the acceleration technique, and the hardware cost estimation strategies. We further discuss the challenges and limitations of existing approaches and potential future directions. This is the first survey paper focusing on hardware-aware NAS. We hope it serves as a valuable reference for the various techniques and algorithms discussed and paves the road for future research towards hardware-aware NAS." The page also includes a "Comments" section (Submitted to Proceedings of IEEE) and "Subjects" (Machine Learning (cs.LG); Computational Complexity (cs.CC)).

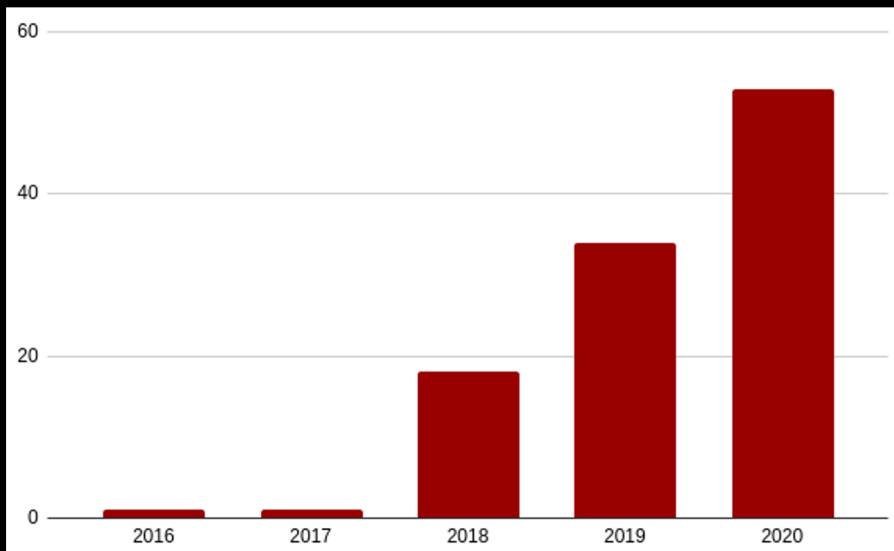
Collaboration with University Polytechnique Hauts-de-France (UPHF)

Hardware-aware NAS (HW-NAS) vs. NAS

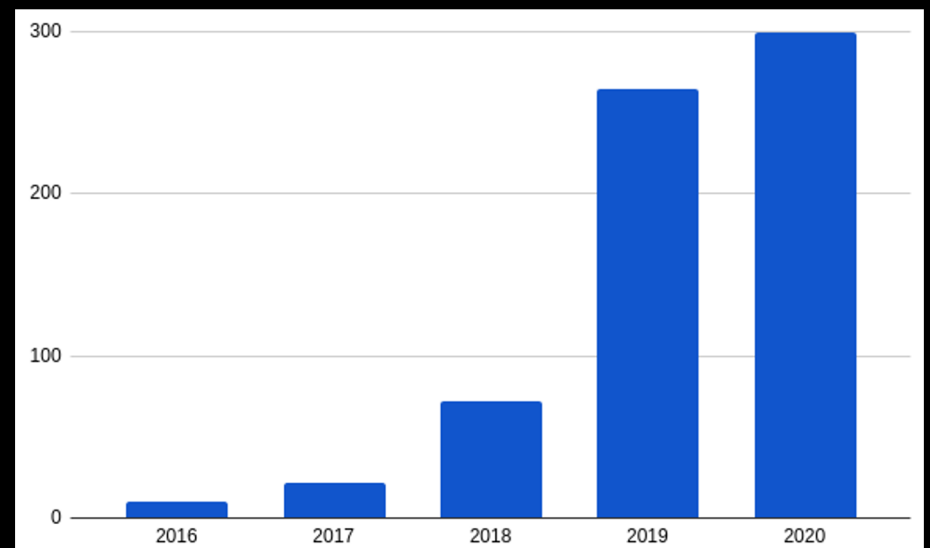


NAS ... a Trending Topic

Hardware-Aware NAS



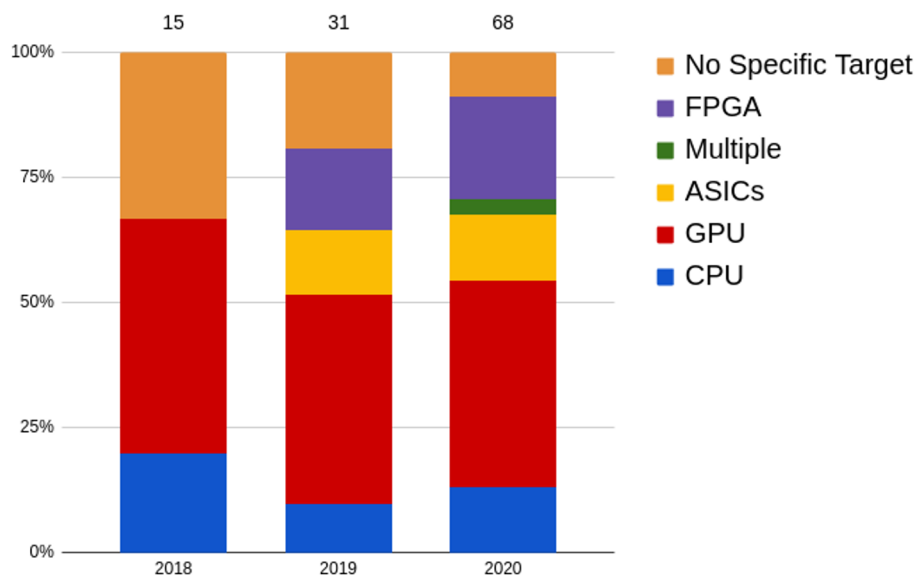
All NAS papers



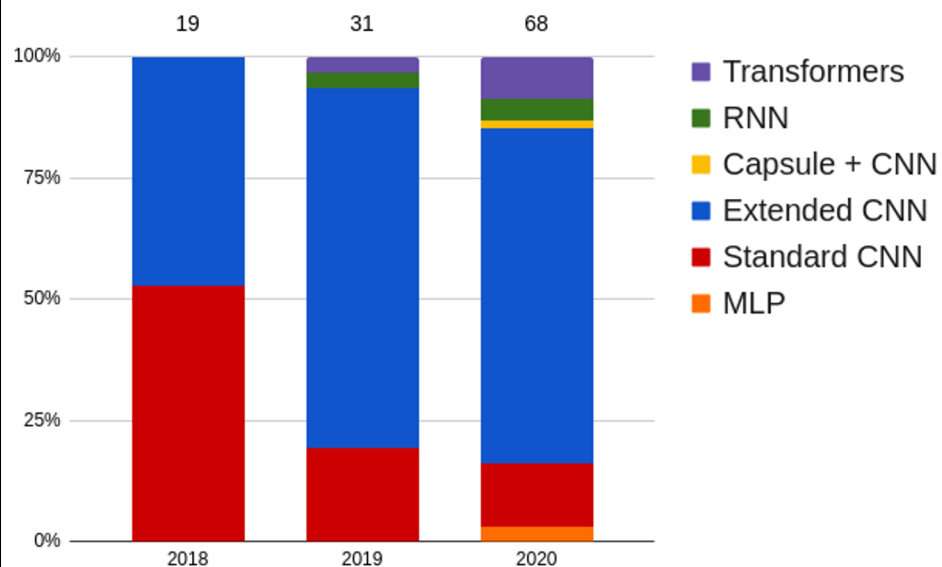
Main targeted journals and conferences: NeurIPS, CVPR, ECCV, ACM GEECO, IEEE Access
(data collected as of December 2020)

HW-NAS Trends

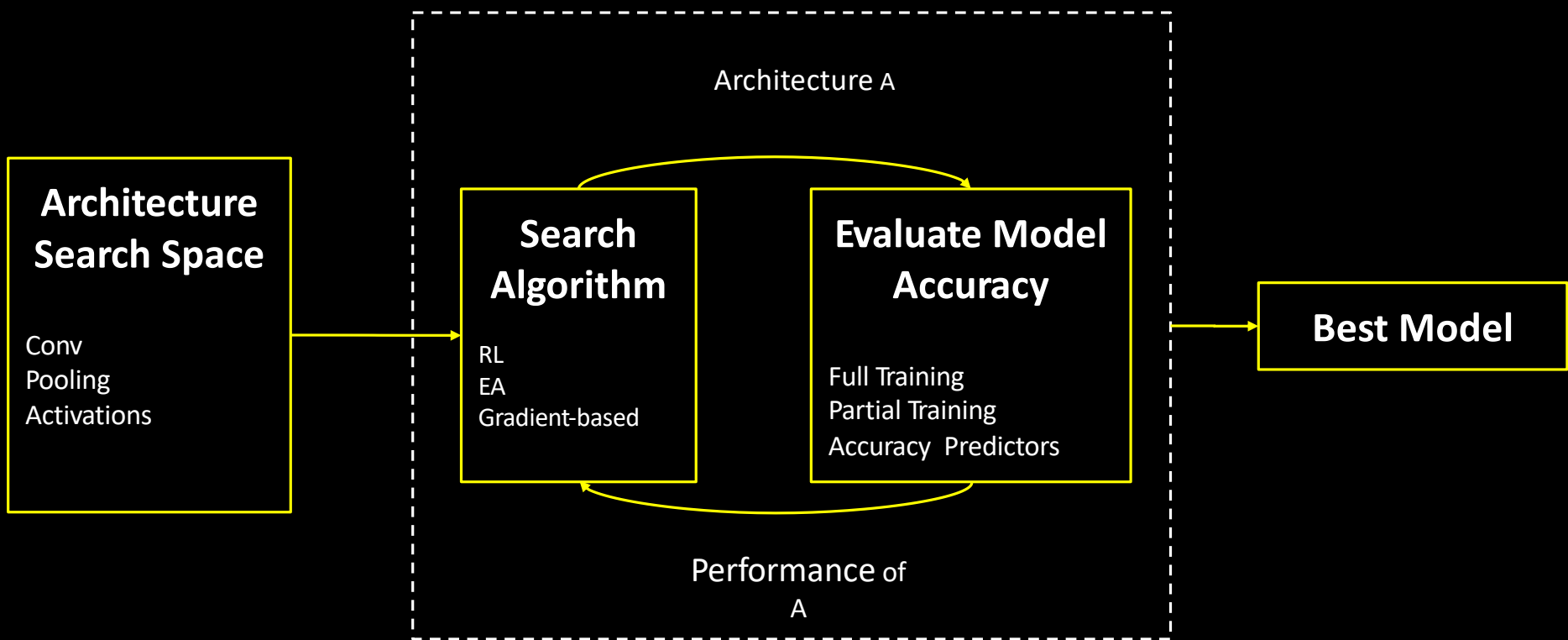
Targeted Hardware Platforms



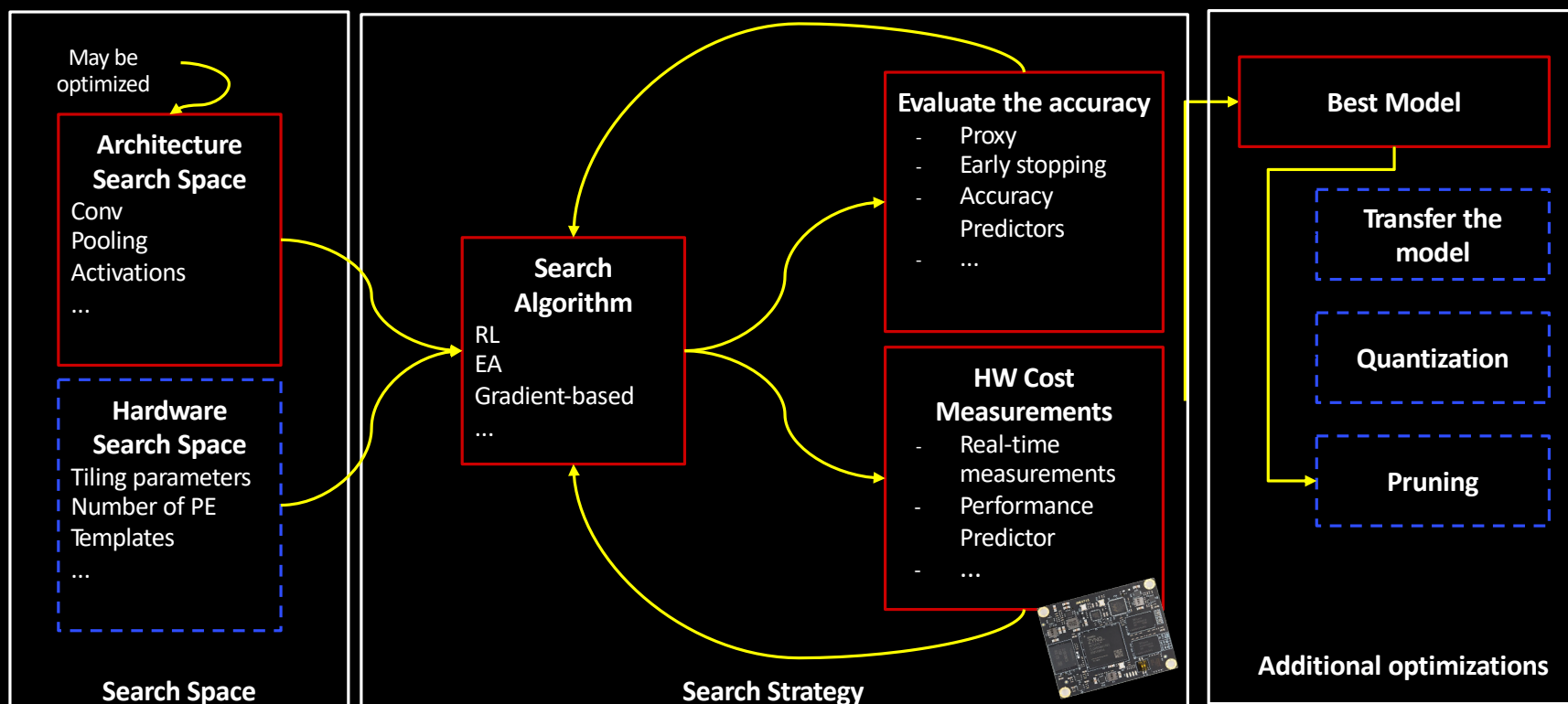
Type of Networks considered in HW-NAS



General NAS Components

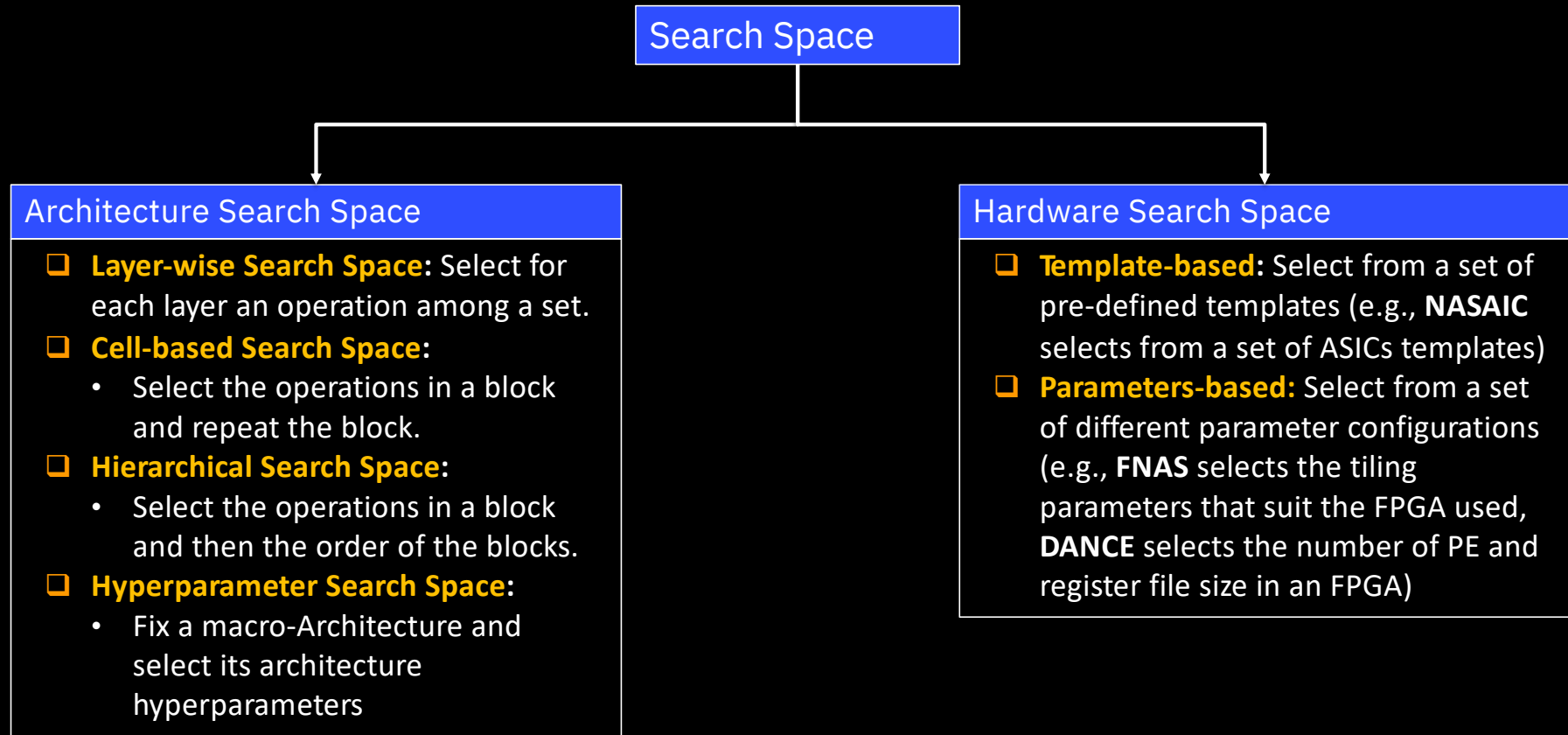


General structure of HW-NAS



--- Optional
— Found in all HW-NAS

HW-NAS Search Space



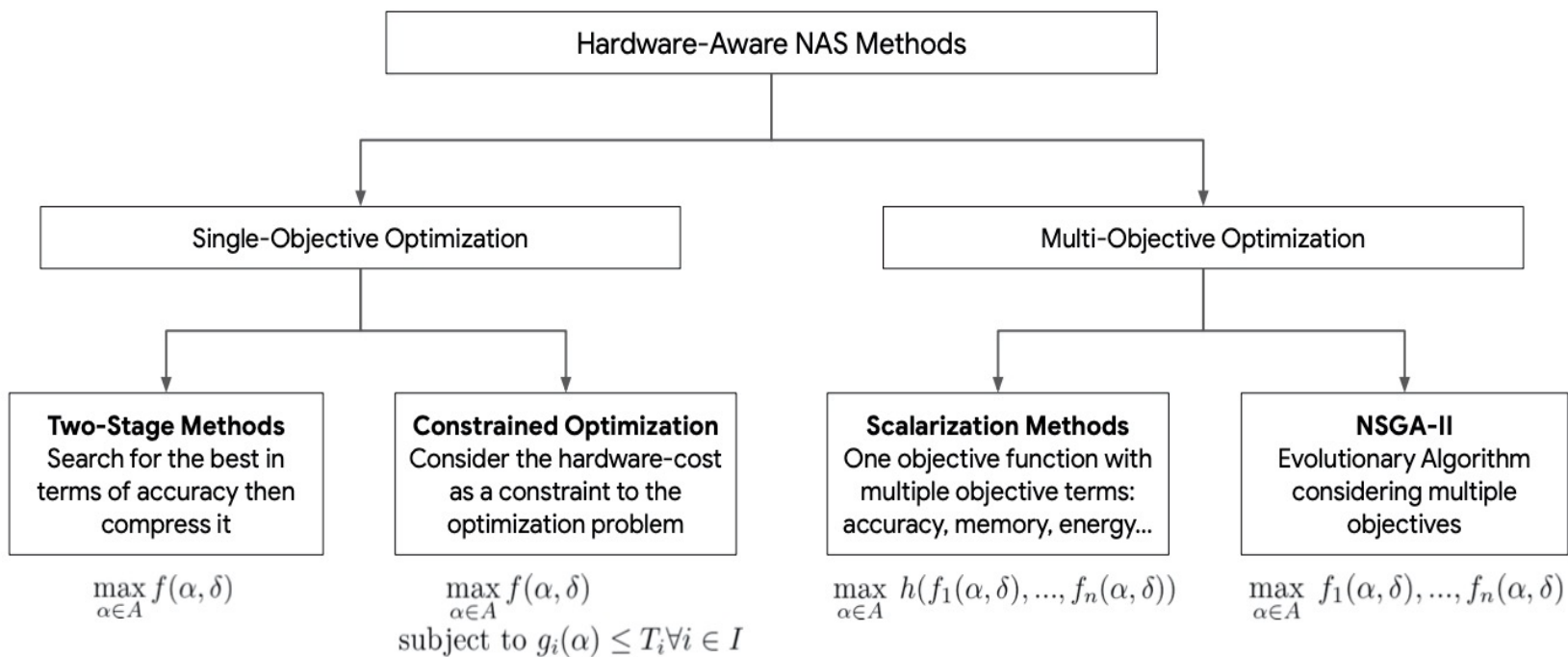
HW-NAS Search Formulation

- NAS Formulations:
(2) $\max_{\alpha \in A} f(\alpha, \delta)$ where,
 - A is the space of all feasible architectures (search space).
 - The optimization method is looking for the architecture α that maximizes the performance metric denoted by f for a given dataset δ (f could simply be the accuracy of the model)
- Constrained optimization

$$\max_{\alpha \in A} f(\alpha) \cdot [LAT(\alpha)/T]^w$$

- LAT is the latency of the model and T is the threshold. w is a learnable parameter to control the effect of the hardware constraints on the global objective function.

Single vs. Multi-Objective Optimization

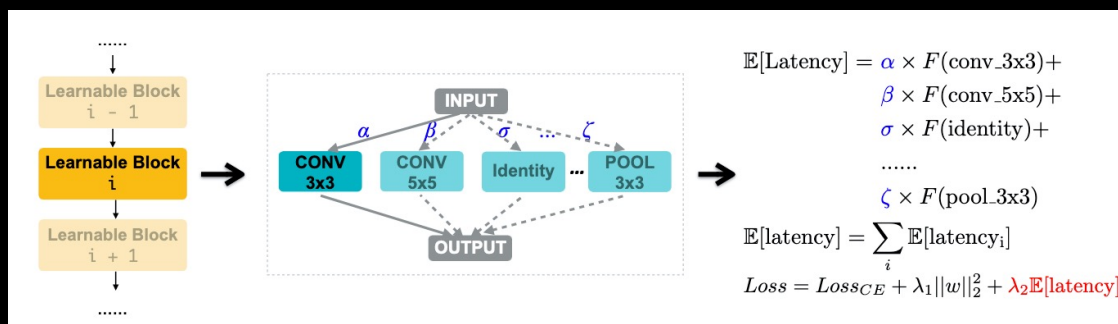


ProxylessNAS Example

- ProxylessNAS uses a loss function that comprises of the cross-entropy (CE) loss and hardware-aware constraints.

$$\mathcal{L} = \mathcal{L}_{CE} + \lambda_1 ||w||^2 + \lambda_2 E[\text{latency}]$$

- Equation above illustrates the loss calculated by the reinforcement learning agent used by ProxylessNAS.
- λ_1 and λ_2 are learnable parameters that adjust the effect of the efficiency of the overall loss.
- A policy is learned that decides whether to add, remove or keep a layer as well as whether to alter its number of filters.



Making latency differentiable by introducing latency regularization loss

Cai et al, ICLR 2019

DNAS: Differentiable Neural Architecture Search

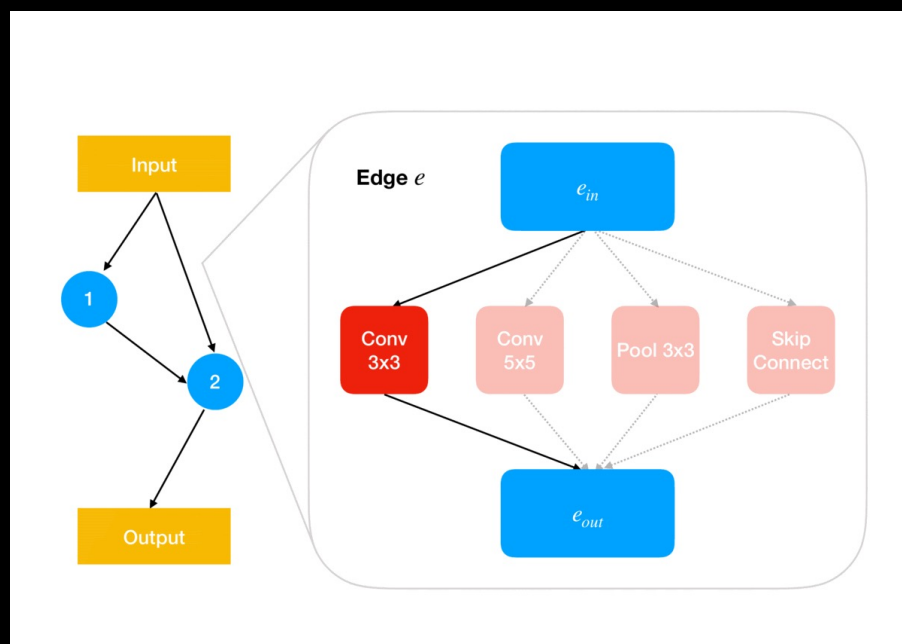


Figure 1: Animation of how DARTS and other weight-sharing methods replace the discrete assignment of one of four operations $o \in O$ to an edge e with a θ -weighted combination of their outputs. At each edge e in the network, the value at input node e_{in} is passed to each operation in $O = \{1:\text{Conv } 3 \times 3, 2:\text{Conv } 5 \times 5, 3:\text{Pool } 3 \times 3, 4:\text{Skip Connect}\}$; the value at output node e_{out} will then be the sum of the operation outputs weighted by parameters $\theta_{e,o} \in [0, 1]$ that satisfy $\sum_{o \in O} \theta_{e,o} = 1$.

Image courtesy of DeterminedAI and CMU CSD

IBM Research AI / © 2021 IBM Corporation

Most famous speed up technique used in NAS which relies on weight sharing and one-shot models

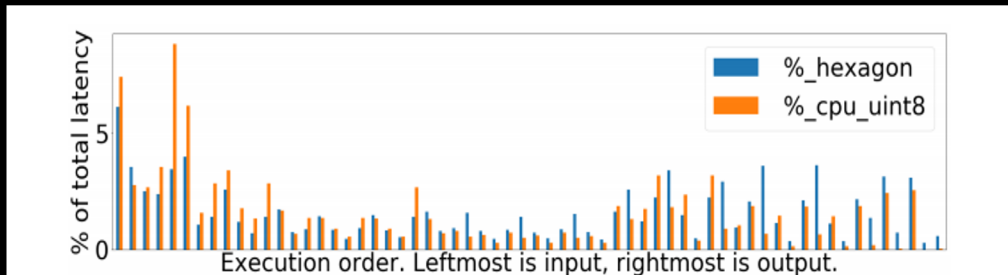
Earlier methods used reinforcement learning and required many computational resources.

- **2000 GPU** days of reinforcement learning or **3150 GPU** days of evolution.
- Not at all feasible

DNAS reduced the search time to few GPU days by relaxing the fixed set of operations into continuous parameters.

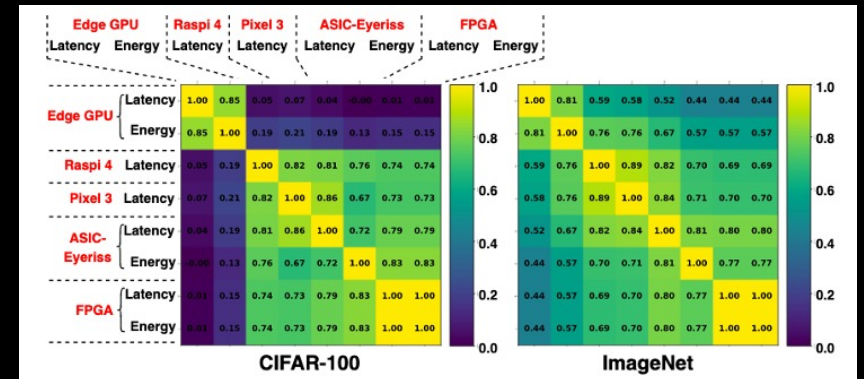
Challenges of Targeting Multiple Hardware Devices

- Different devices have different design choices
- different hardware devices can favor very different network structures under the same hardware-cost metric, and



Per layer profiling of MobileNetV3 minimalistic on Pixel4 CPU uint8 and QUALCOMM hexagon. The leftmost is the input layer while the output layer is on the right

Source: G. Chu et al, "Discovering multi-hardware mobile models via architecture search," 2020.

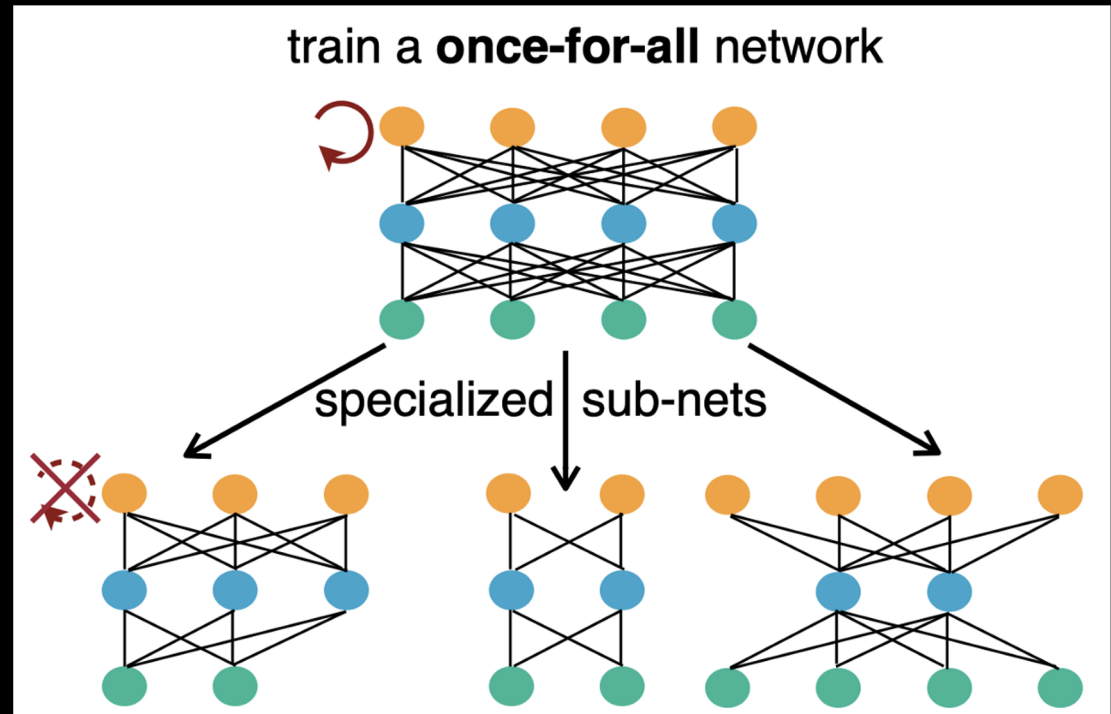


Kendall rank correlation between real/estimated hw cost between different devices considering the FBNet search space

Source: C. Li et al, «HW-NAS Bench», ICLR 2021

HW-NAS: Speedup Techniques

- Early Stopping
- Proxy Dataset
- Weight Sharing/ Super Network
- Accuracy Predictor
 - Peephole
 - PNAS
 - TAPAS



Source: H. Cai, C. Gan, T. Wang, Z. Zhang, and S. Han, "Once-for-all: Train one network and specialize it for efficient deployment," 2020.

HW Cost Evaluation Techniques

Method	How the method is achieved ?	Hardware Metric	Cost	References
Real-time measurements	The sampled model is executed on the hardware target while searching.	Latency		MNASNet[21] NetAdapt[58] [59] MCUNet[45]
		Energy		NetAdapt[58] MONAS[36] [60]
Lookup Table Models	A lookup table is created beforehand and filled with each operator latency on the targeted hardware. Once the search starts, the system will calculate the overall cost from the lookup table.	Latency		FBNet[16] HotNAS[32]
Low Fidelity Estimation	Compute a rough estimate using the processing time, the stall time, and the starting time.	Latency		FNAS[19] NASCaps[29] [61] [42]
		Energy		NASCaps[29]
		Memory footprint		NASCaps[29]
		Area		NASAIC[23]
Prediction Model	Build a ML model to predict the cost using architecture and dataset features.	Latency		proxylessNAS[20] NASAIC[23] NeuNets[62] LEMONADE[36]

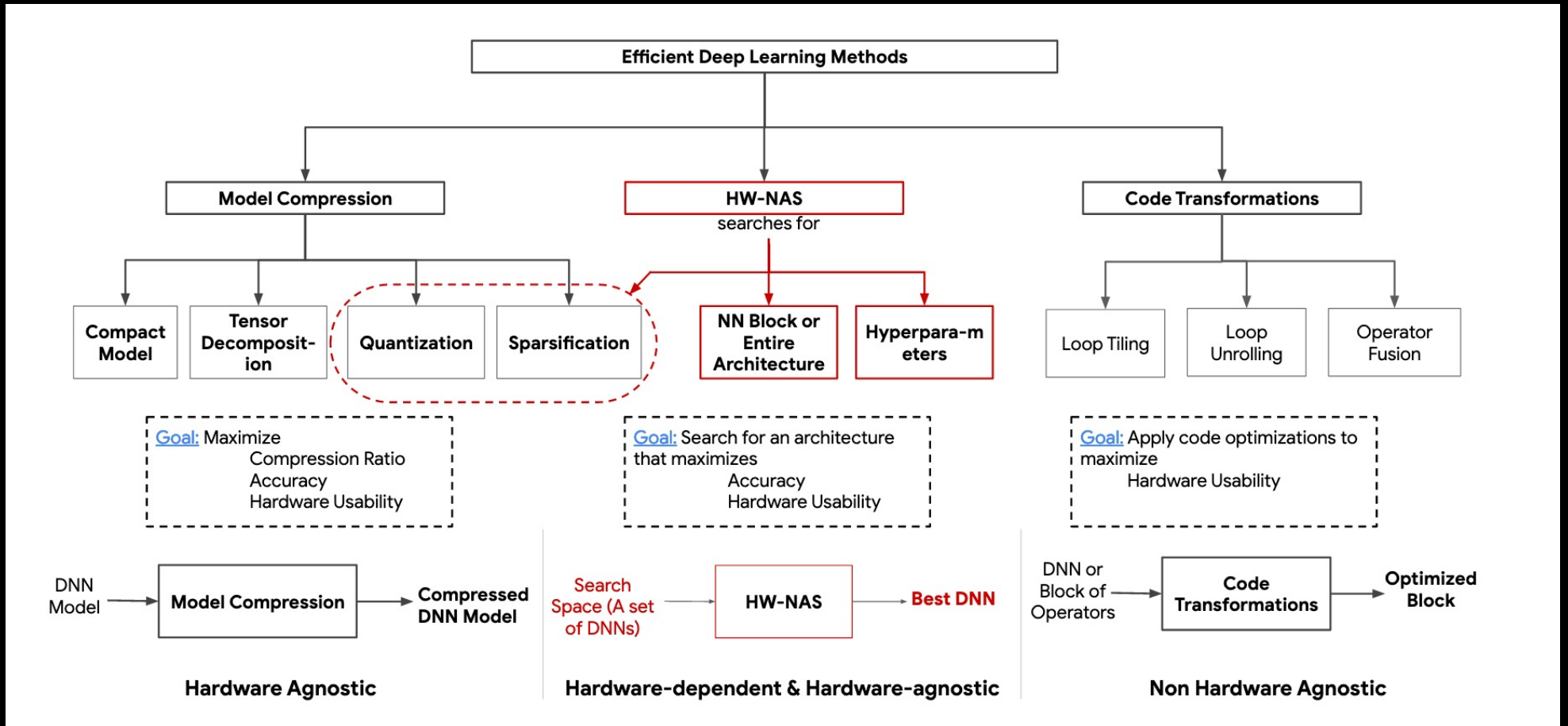
Key Challenges

- **Combinatorial explosion of the search space** if considering hw-metrics or efficient strategies like quantization and pruning.
- **High computational cost of the search strategy**
- **Adding hardware-cost increases further the search complexity**
- **Benchmarking and Reproducibility**
 - Lack of HW-NAS benchmarks. HW-NAS-Bench (first hardware-aware Benchmark, published at ICLR 2021)
- **Transferability of the AI Models**
 - Cell-based vs. layer-wise search spaces
- **Transferability of the hardware-aware NAS across multiple platforms**
 - Transfer the entire NAS process
 - Specialize the final model
- Most NAS and HW-NAS approaches are limited to computer vision applications.

HW-NAS Key Takeaways

- HW-NAS is a hard problem and still not a fully solved – still no principled approaches exist
- Largely unexplored in non-vision application areas.
 - Success of NAS to discover Efficient networks for Vision – still not as good for NLP.
- Key techniques that stand out
 - Layer-wise search spaces are more hardware friendly than cell-based search search (e.g., FBNet)
 - Using differentiable techniques and weight sharing
 - Using both gradient based methods and RL/Evolutionary methods (First is used to train the architecture weights and second is used to incorporate the hardware cost)
 - Reducing the search space
 - Using predictors for training and predictors for hardware cost measurements
 - Quantization-aware and pruning-aware predictors

Efficient Deep Learning Methods are Inevitable and still an Evolving Landscape



Thank you!

—

Dr. Kaoutar El Maghraoui
Principal Research Staff Member
IBM Research AI

@kaoutarTech

IBM Research

