Mixed Precision Neural Architecture Search for Energy Efficient Deep Learning

Chengyue Gong*[1], **Zixuan Jiang***[2], Dilin Wang[1],
Yibo Lin[2], Qiang Liu[1], and David Z. Pan[2]

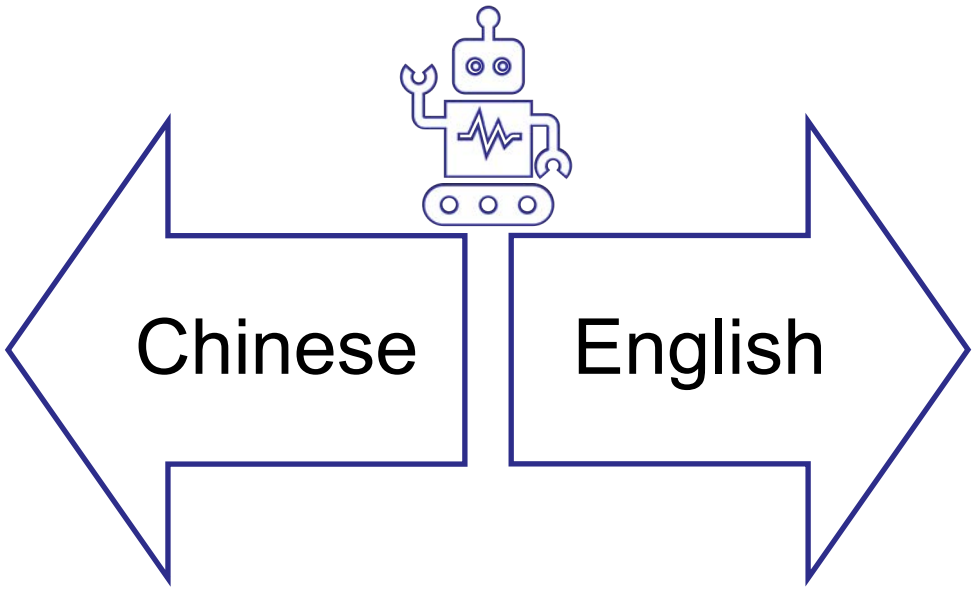
[1]CS Department, [2]ECE Department
The University of Texas at Austin

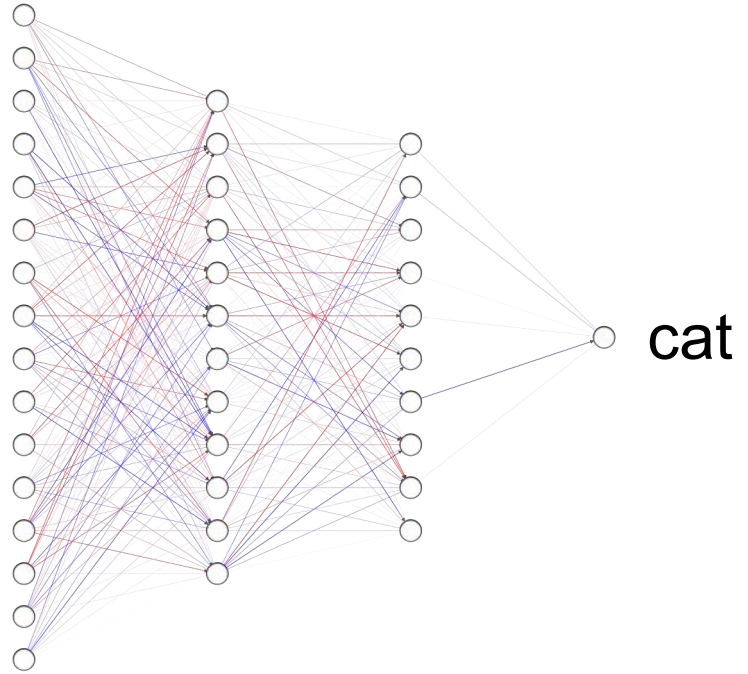
* indicates equal contributions

Contents

- **Introduction**
- Our algorithm
- Experimental results
- Conclusion

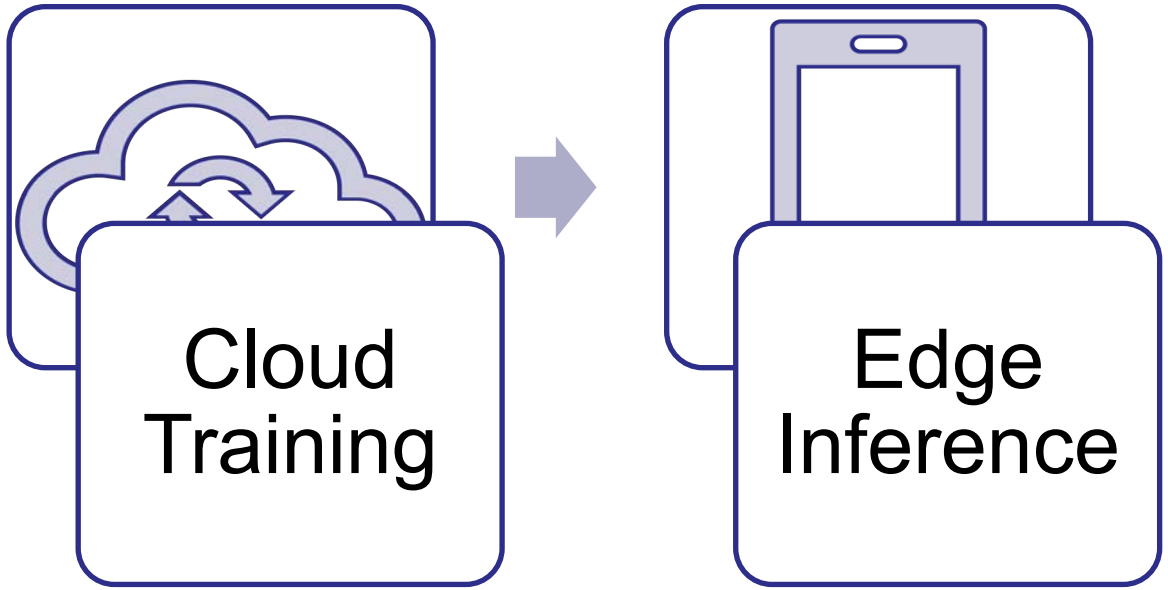
Success of Machine Learning

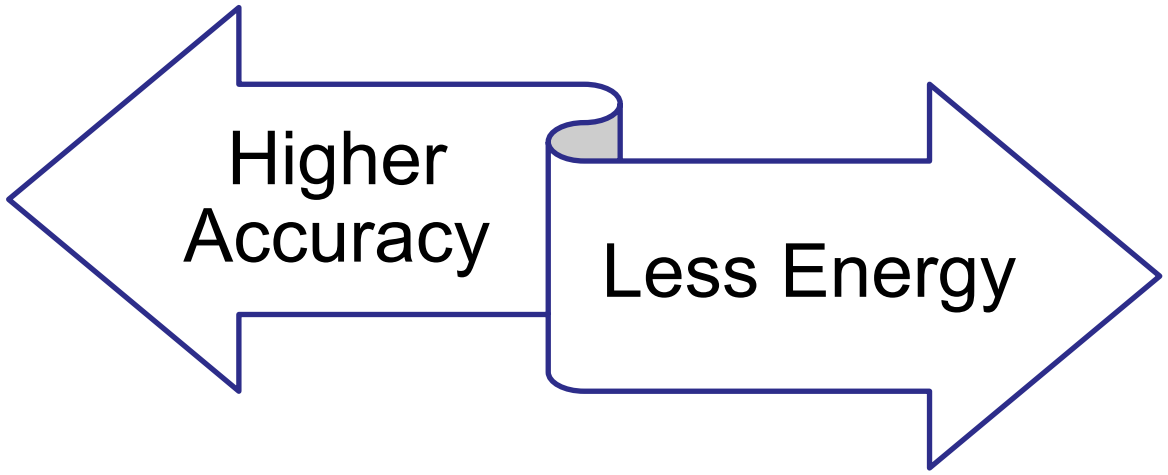




Energy Efficient Computation

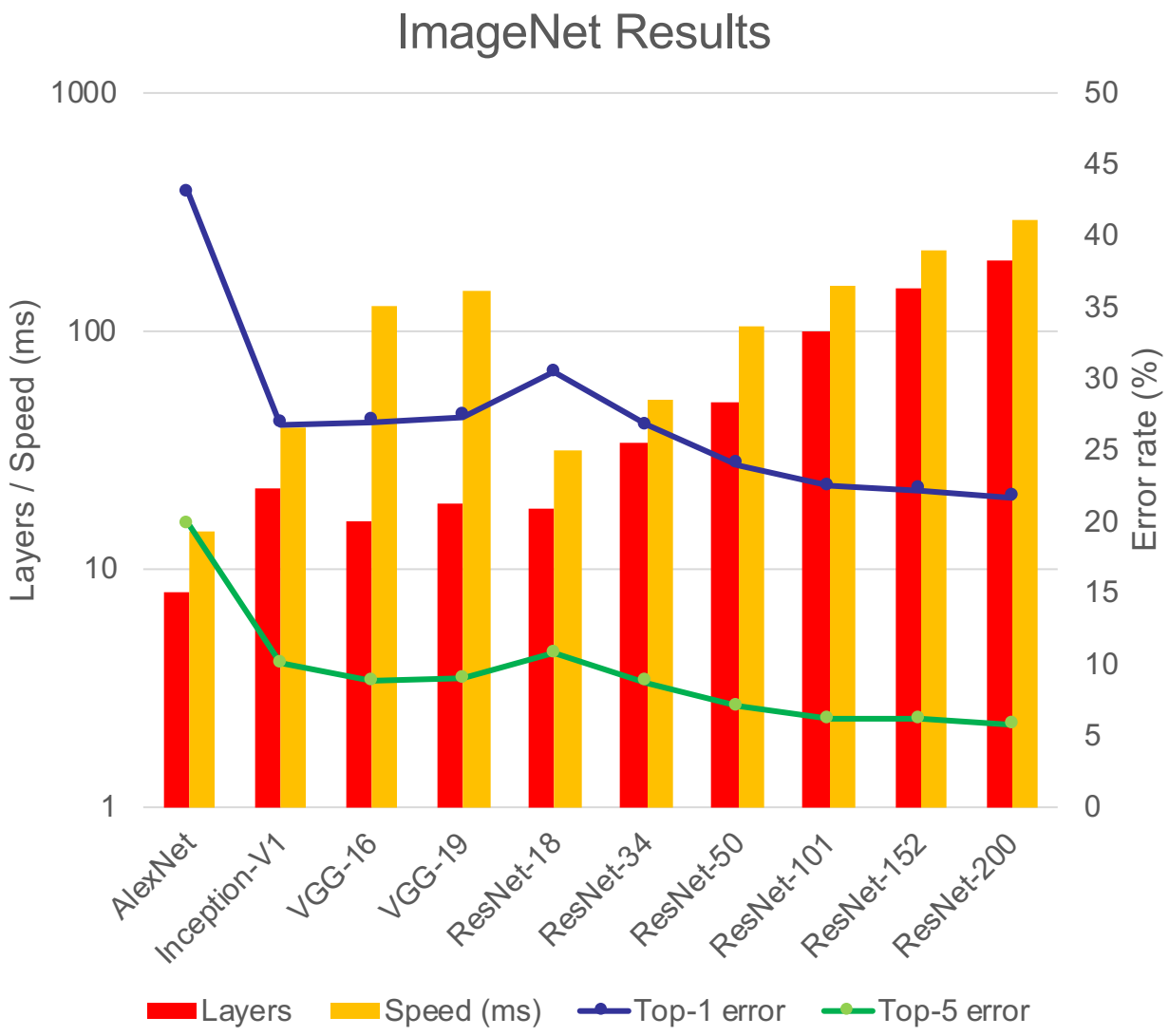
- Energy computation, latency, security, etc. are critical metrics of edge inference.
- Tradeoff between accuracy and complexity of models.
- Efficient computation
  - Neural architecture
  - Quantization



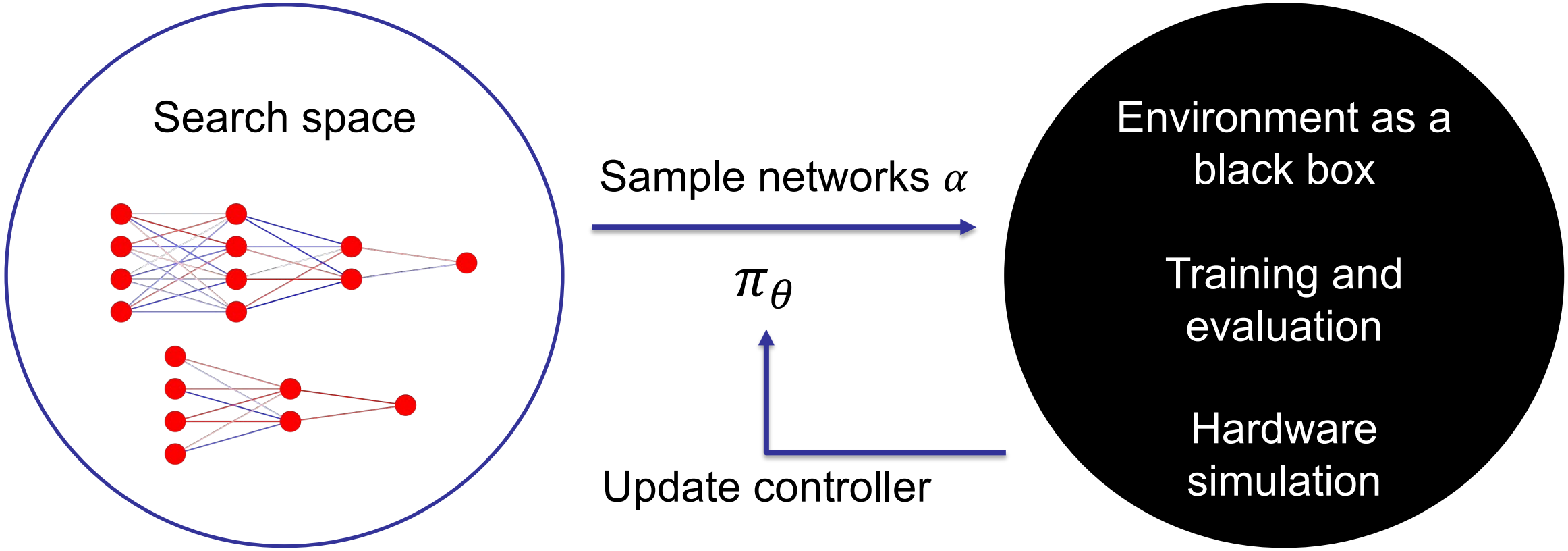


Neural Architecture Design

- Mechanism of neural networks is not well interpreted.
- Designing neural architecture is challenging.
- Can we advance AI/ML using artificial intelligence instead of human intelligence?



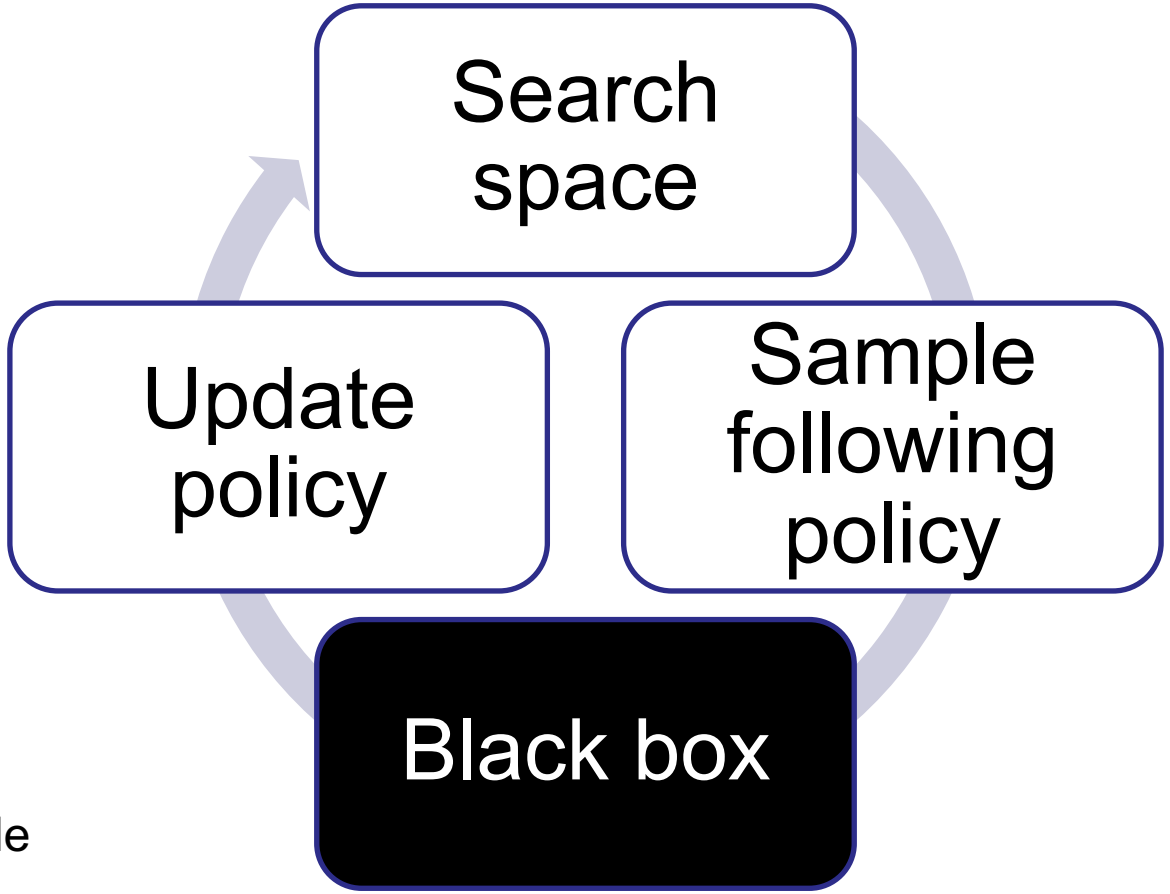
Neural Architecture Search



Neural Architecture Search

- Black box optimization
  - Find the optimal network configuration to maximize the performance
  - Huge search space
- Available methods
  - Reinforcement learning
  - Evolutionary algorithm
  - Differentiable architecture search

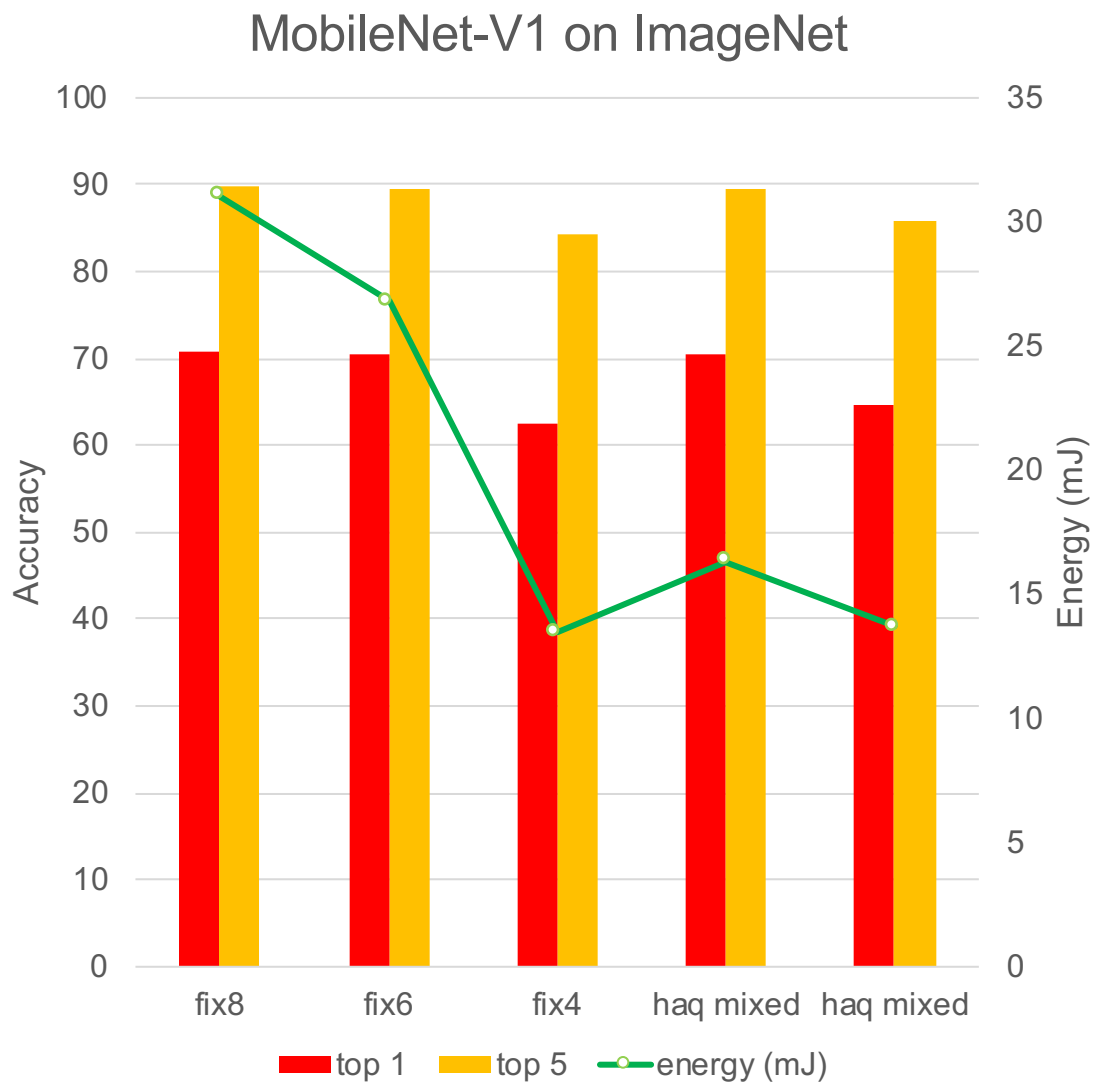
H. Liu, K. Simonyan, and Y. Yang, “Darts: Differentiable architecture search,” ICLR 2019.



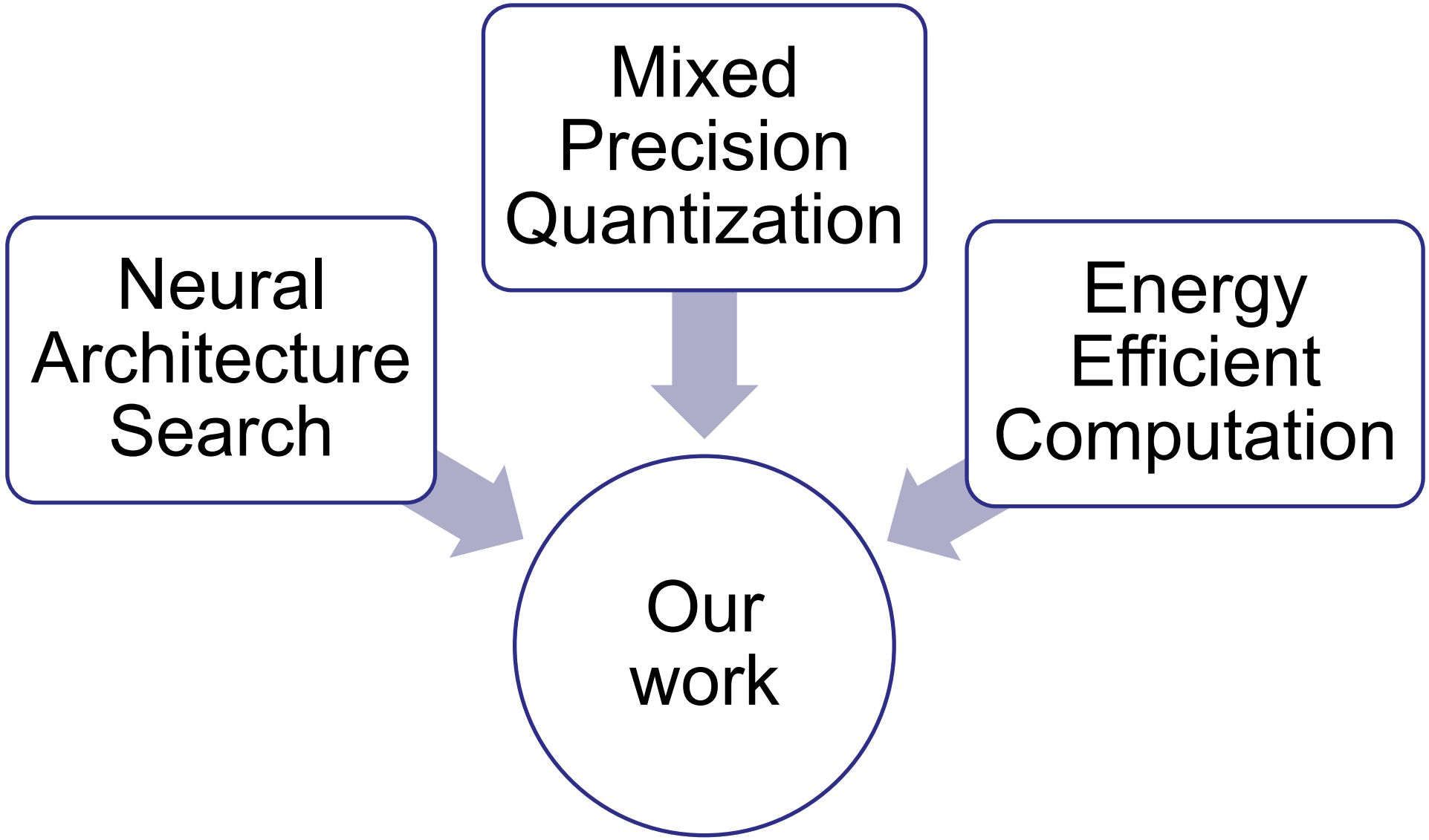
Quantization

- Weights, activations can be quantized due to the inherent redundancy in representations.
- Mixed precision for different layers
  - HAQ

K. Wang, Z. Liu, Y. Lin, J. Lin, and S. Han, “HAQ: hardware-aware automated quantization with mixed precision,” CVPR, 2019.



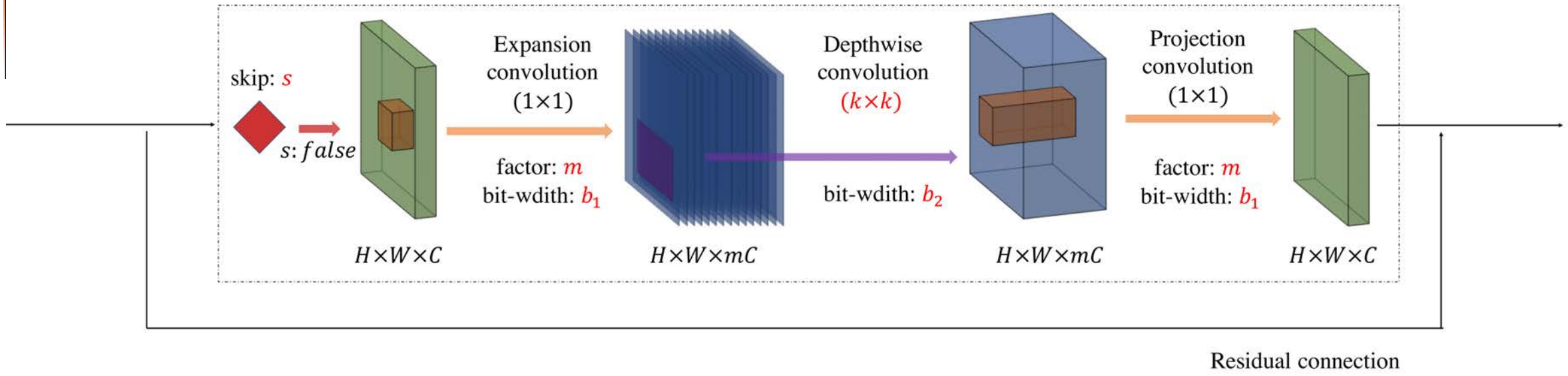
Our Work


Mixed Precision Quantization
Neural Architecture Search
Energy Efficient Computation
Our work

Contents

- Introduction
- **Our algorithm**
- Experimental results
- Conclusion

Search Space Basis: MobileNetV2 Block (MB)



expand ratio $m \in \{1,3,6\}$
network connectivity $s \in \{0,1\}$

kernel size $k \in \{3,5,7\}$
layer−wise bitwidths $b_1, b_2 \in \{2,4,6,8\}$

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L. C. MobileNetV2: Inverted residuals and linear bottlenecks. CVPR 2018.

Search Space

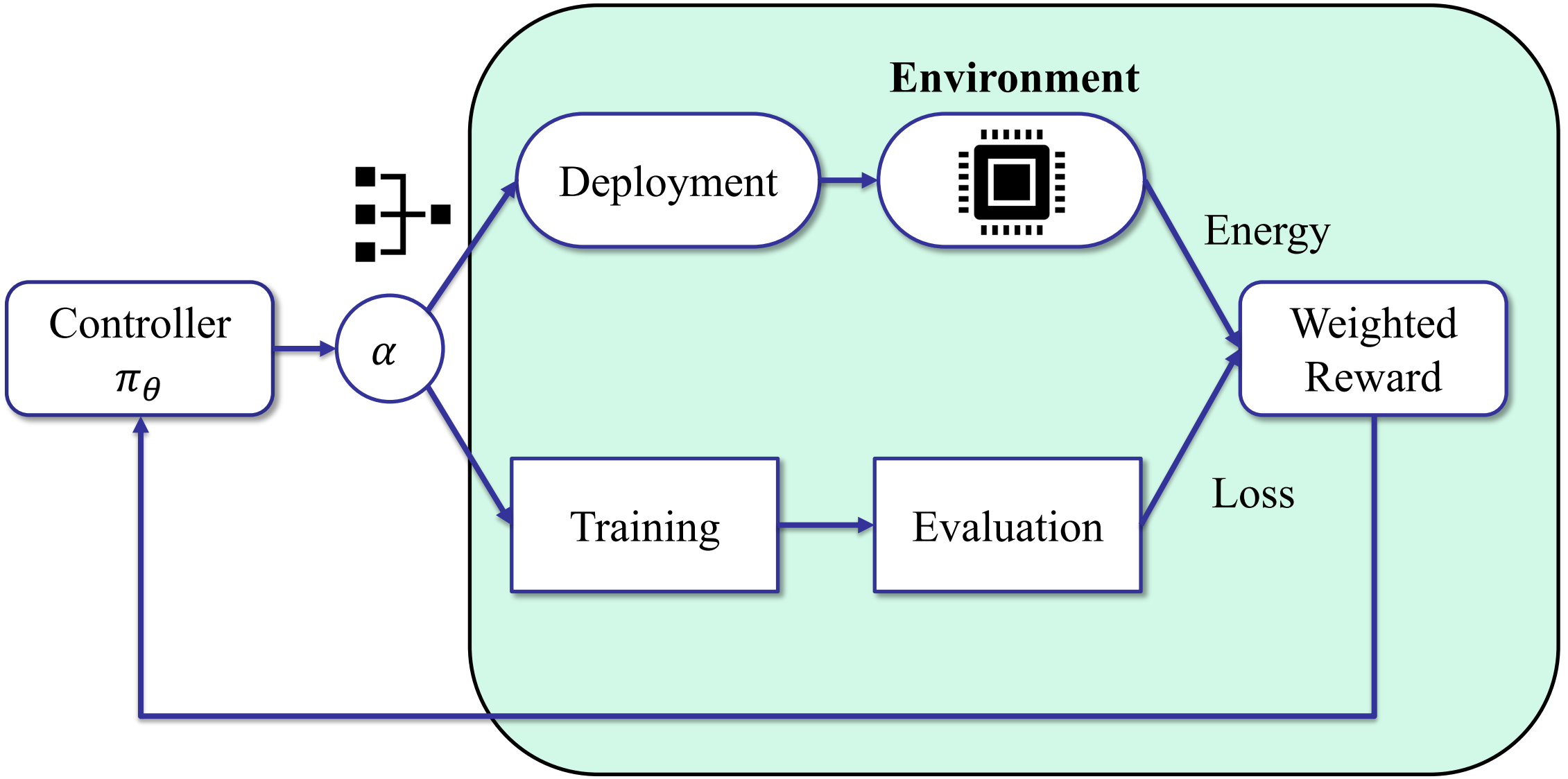
Input Shape	Block Type	Bitwidth	#Channels	Stride	#Blocks
224 × 224 × 3	Conv 3 × 3	8	32	2	1
112 × 112 × 32	Search Space block(s, m, k, b_1, b_2)		16	2	1
56 × 56 × 16			24	1	2
56 × 56 × 24	Neural architecture s, m, k		32	2	4
28 × 28 × 32			64	2	4
14 × 14 × 64			128	1	4
14 × 14 × 128	Quantization b_1, b_2		160	2	5
7 × 7 × 160			256	1	2
7 × 7 × 256	Conv 1 × 1	8	1280	1	1
7 × 7 × 1280	Pooling and FC	8	-	1	1

settings

$$\mathbf{288^{22}}$$

$$\approx \mathbf{1.28 \times 10^{54}}$$

Our Framework



Problem Formulation

- Discover neural architectures that **minimize the task-related loss** while satisfying the **energy constraint**.

$$\min_{\theta} \mathbb{E}_{\alpha \sim \pi_\theta}[L(f_{w^*}(\alpha); D_{validate})]$$ Expectation of loss

$$s.t.\ w^* = \operatorname{argmin} L(f_w(\alpha); D_{train})$$ Training of NN

$$\mathbb{E}_{\alpha \sim \pi_\theta} J(\alpha) < c$$ Energy constraint

π_θ is the policy with parameter θ

α represents a neural network with weights w

Problem Formulation

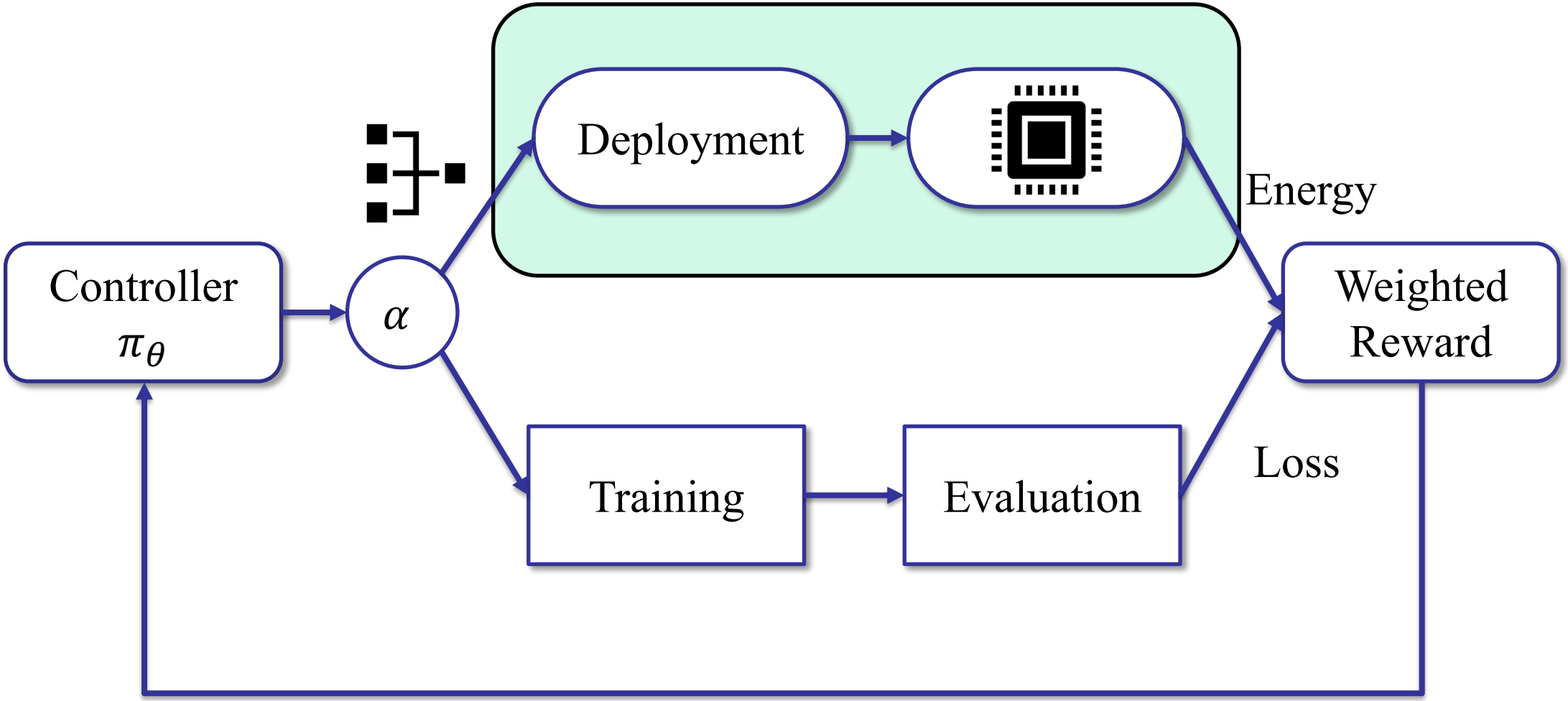
- Discover neural architectures that minimize the task-related loss while satisfying the energy constraint.

$$\min_{\theta} \mathbb{E}_{\alpha \sim \pi_\theta}[L(f_{w^*}(\alpha); D_{validate})]$$
$$s.t.\ w^* = \operatorname{argmin} L(f_w(\alpha); D_{train})$$
$$\mathbb{E}_{\alpha \sim \pi_\theta} J(\alpha) < c$$

- Relaxation

$$\min_{\theta} \mathbb{E}_{\alpha \sim \pi_\theta}[L(f_{w^*}(\alpha); D_{validate})] + \lambda \max\left(\mathbb{E}_{\alpha \sim \pi_\theta} J(\alpha) - c, 0\right)$$
$$s.t.\ w^* = \operatorname{argmin} L(f_w(\alpha); D_{train})$$

Hardware Environment



REINFORCE algorithm

- Policy gradient theorem

For any differentiable policy π_θ, for any policy objective functions J, the policy gradient is

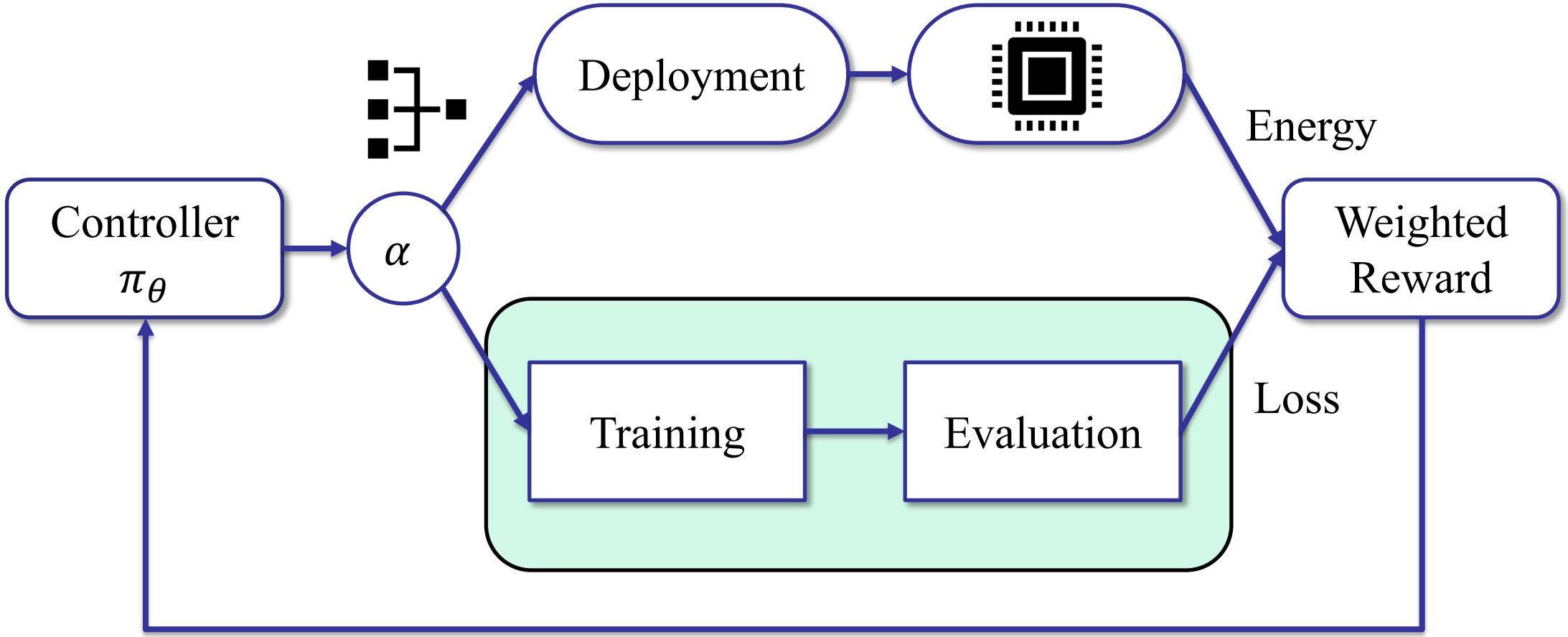
$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta \, Q^{\pi_\theta}]$$

- Non-differentiable energy measures

$$\nabla_\theta \mathbb{E}_{\alpha \sim \pi_\theta} J(\alpha) = \mathbb{E}_{\alpha \sim \pi_\theta}[J(\alpha) \nabla_\theta \log \pi_\theta]$$

$$\approx \frac{1}{k} \sum_{i=1}^{k} J(\alpha_i) \nabla_\theta \log \pi_\theta(\alpha_i)$$

Software Environment



Non-Differentiability

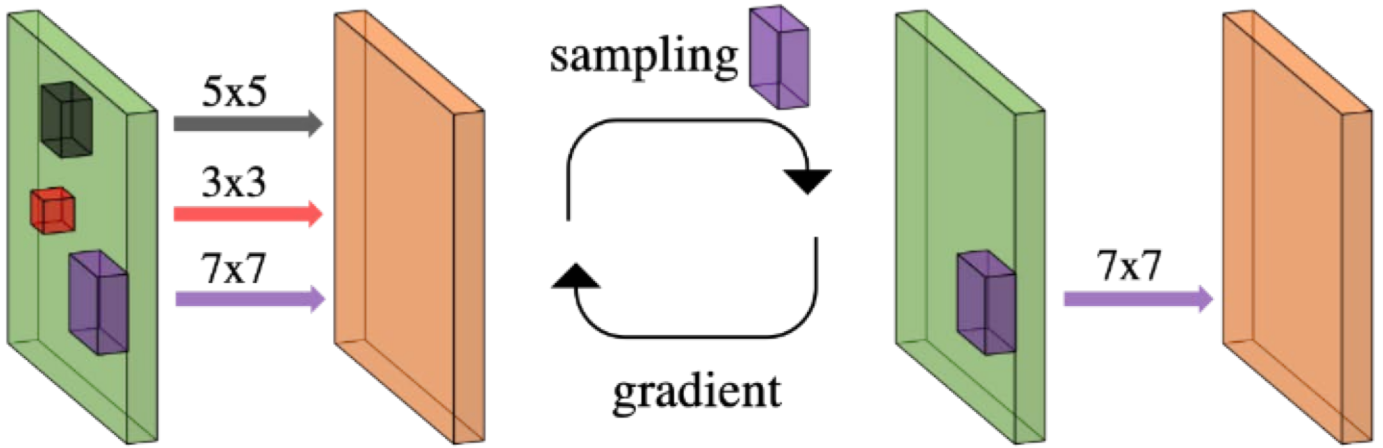
- Relax the discrete mask variable d to be a continuous random variable computed by the **Gumbel Softmax** function

$$d_i = \frac{\exp\left((g_i + \log v_i)/\tau\right)}{\sum \exp\left((g_i + \log v_i)/\tau\right)}$$

where v_i is the logit, $g_i \sim$Gumbel(0, 1), τ is the temperature.

- One-hot [0, 1, 0]

  Continuous [.3, .5, .2]



E. Jang, S. Gu, and B. Poole, "Categorical reparameterization with Gumbel-Softmax," ICLR, 2017.

Bilevel Optimization

- Whenever the policy parameter θ changes, the weights of network $w(\alpha)$ needs to be retrained.
- Motivated by differentiable architecture search (**DARTS**), we propose the following algorithm.

1. Sample minibatch of network configurations α from the controller
2. Update network models $w(\alpha)$ by minimizing the training loss
3. Update the controller parameters θ

H. Liu, K. Simonyan, and Y. Yang, "DARTS: Differentiable architecture search," ICLR 2019.

Contents

- Introduction
- Our algorithm
- **Experimental results**
- Conclusion

Experimental Settings

- Hardware simulator of Bit Fusion [1, 2]

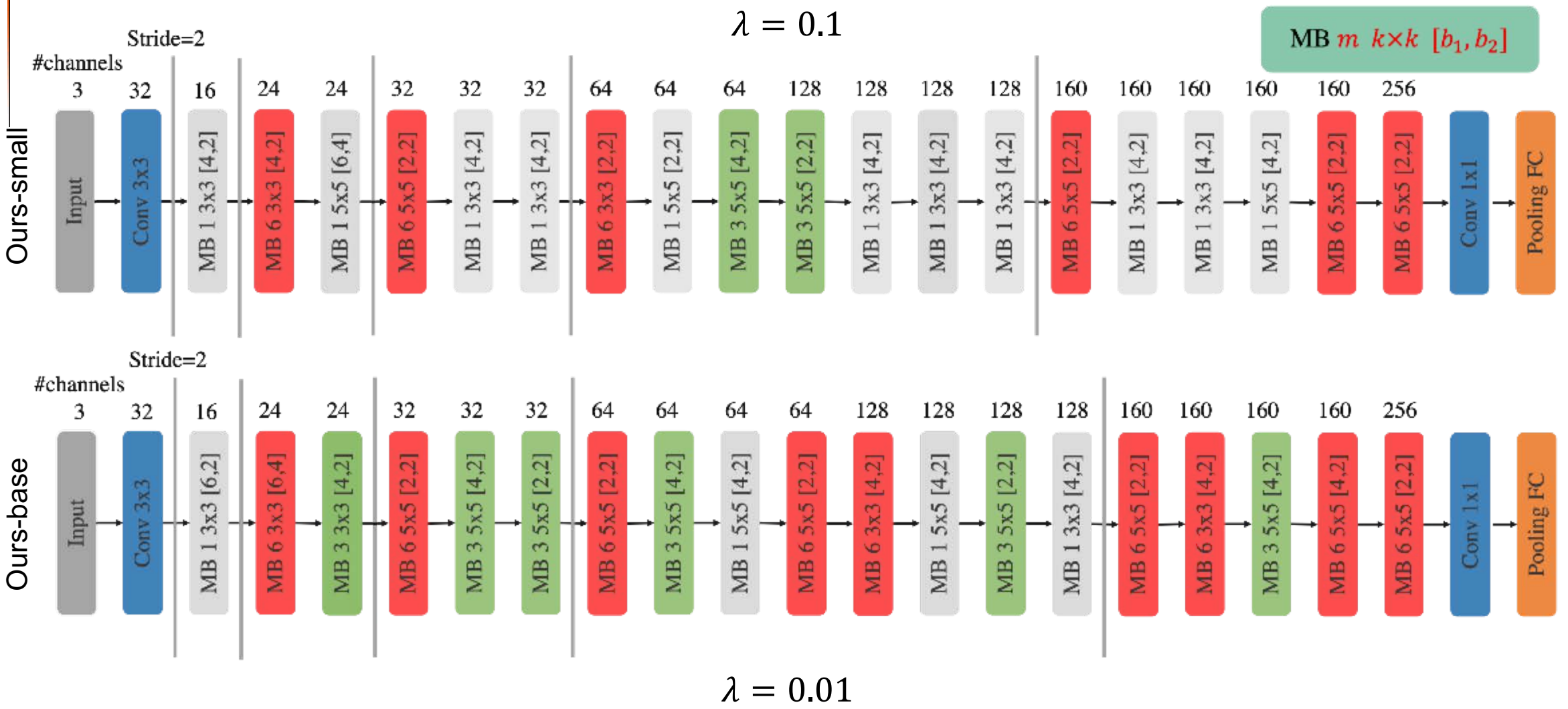
- First, search architectures and mixed precision for each layer on a proxy task, tiny ImageNet
  - Trained for a fixed 60 epochs
  - 5 days on 1 NVIDIA Tesla P100

- Next, train the discovered architectures on CIFAR-100 and ImageNet from scratch.

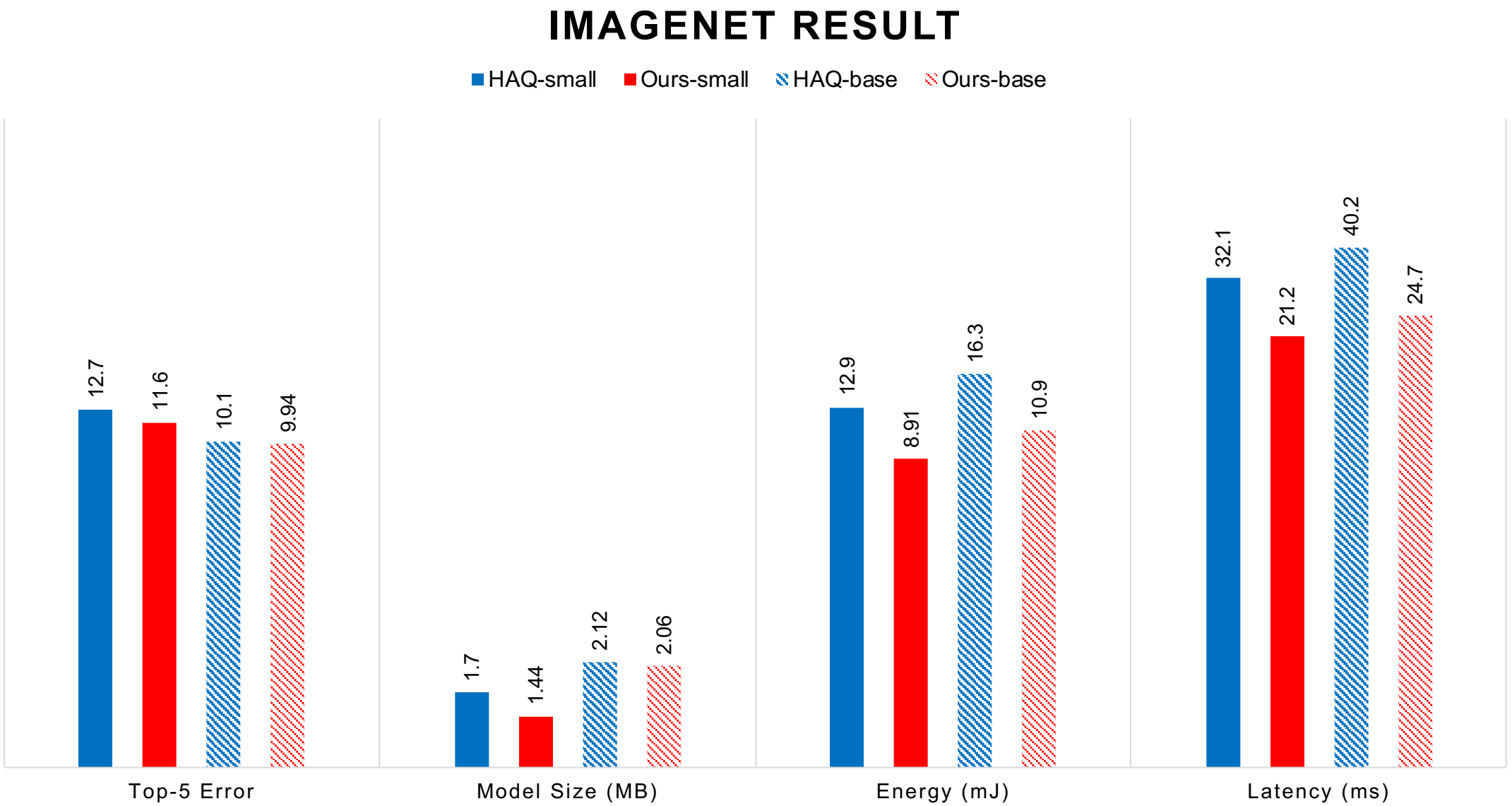
[1] H. Sharma, J. Park, N. Suda, L. Lai, B. Chau, V. Chandra, and H. Esmaeilzadeh, “Bit fusion: Bit-level dynamically composable architecture for accelerating deep neural network,” in Proc. ISCA, June 2018, pp. 764–775.

[2] https://github.com/hsharma35/bitfusion

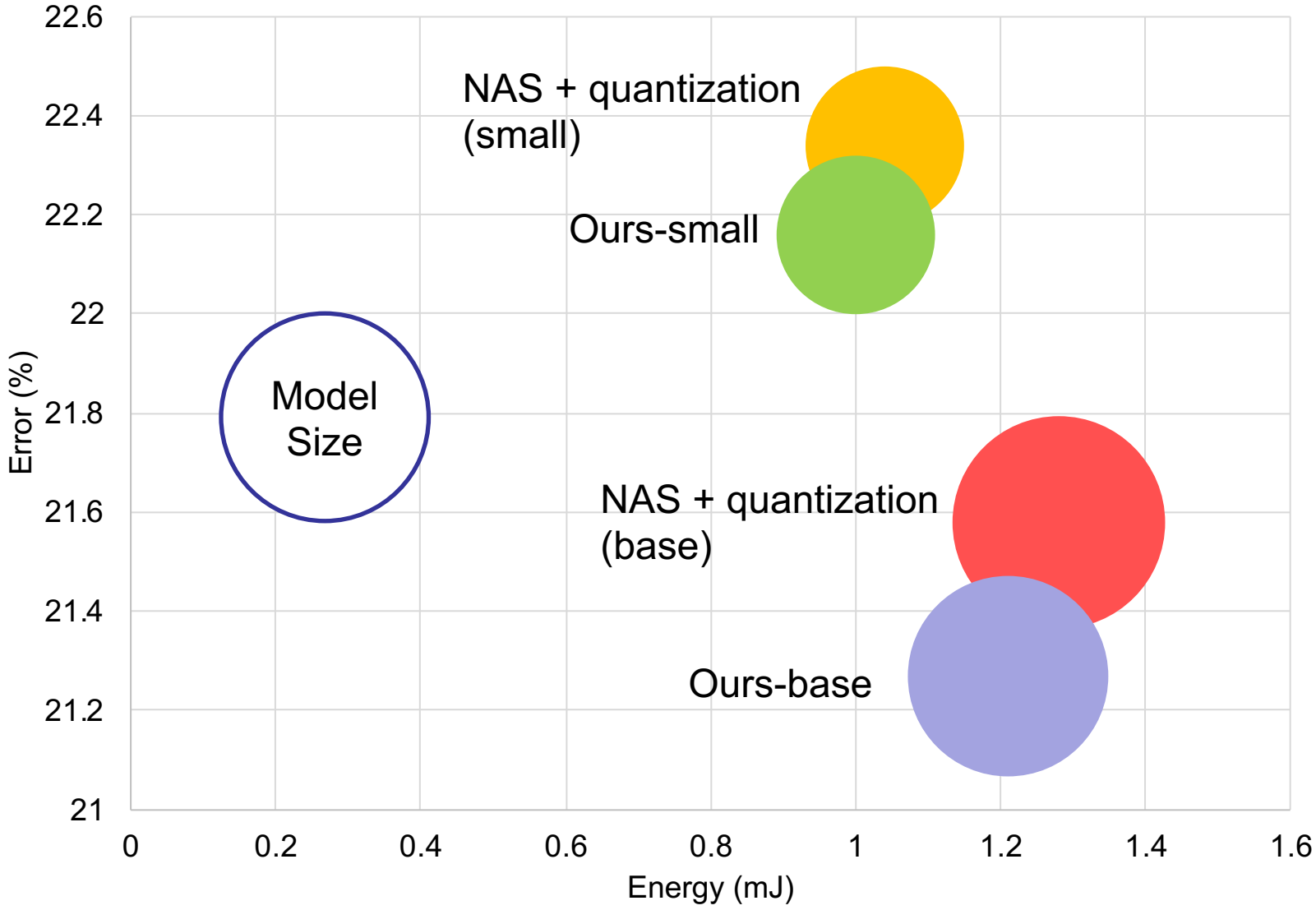
Searched Results



Results on ImageNet

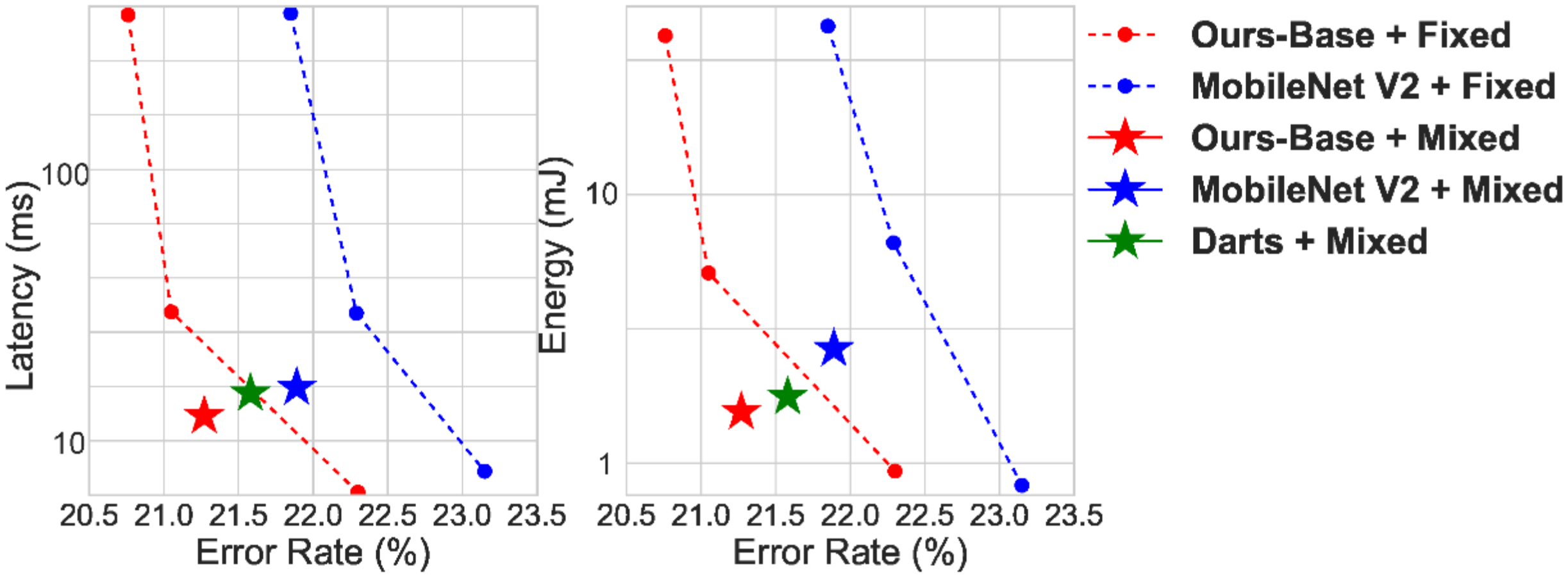


Joint NAS and Mixed Precision Quantization



Adaptive Mixed Precision Quantization

- Pareto front for error rate, latency, and energy



Contents

- Introduction
- Our algorithm
- Experimental results
- **Conclusion**

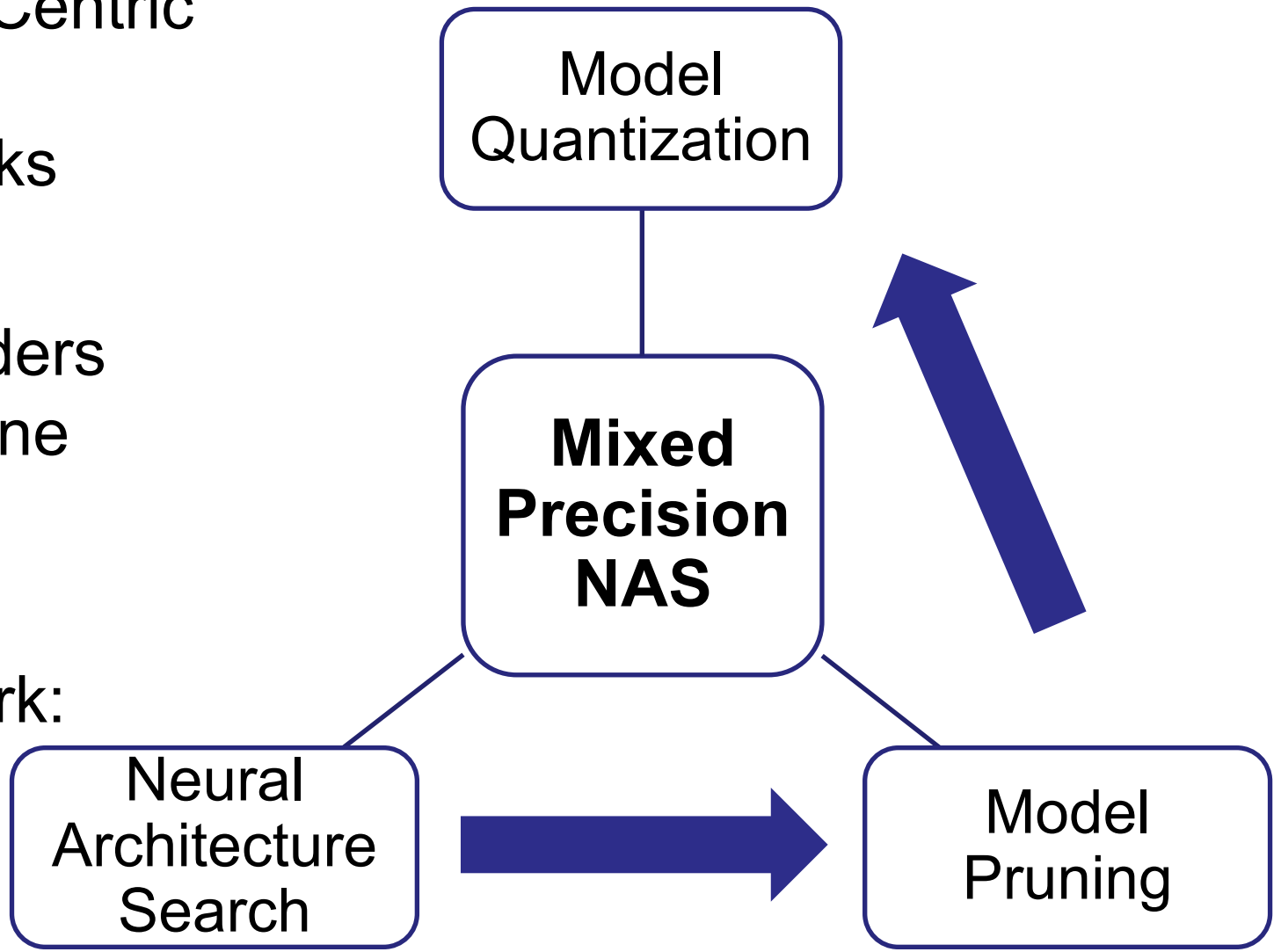
Conclusion

- We propose a new methodology to perform **joint optimization** of NAS and mixed precision quantization in the extended search space.
- Hardware performance is involved in the objective function.
- Our methodology facilitates the end-to-end design automation flow of neural network design and deployment, especially the edge inference.

Thank you!

Backup Framework

- Pipelines of Hardware-Centric Design Automation for Efficient Neural Networks
- Limited research considers each stage of the pipeline collaboratively
- Our proposed framework: Mixed Precision NAS



Backup result