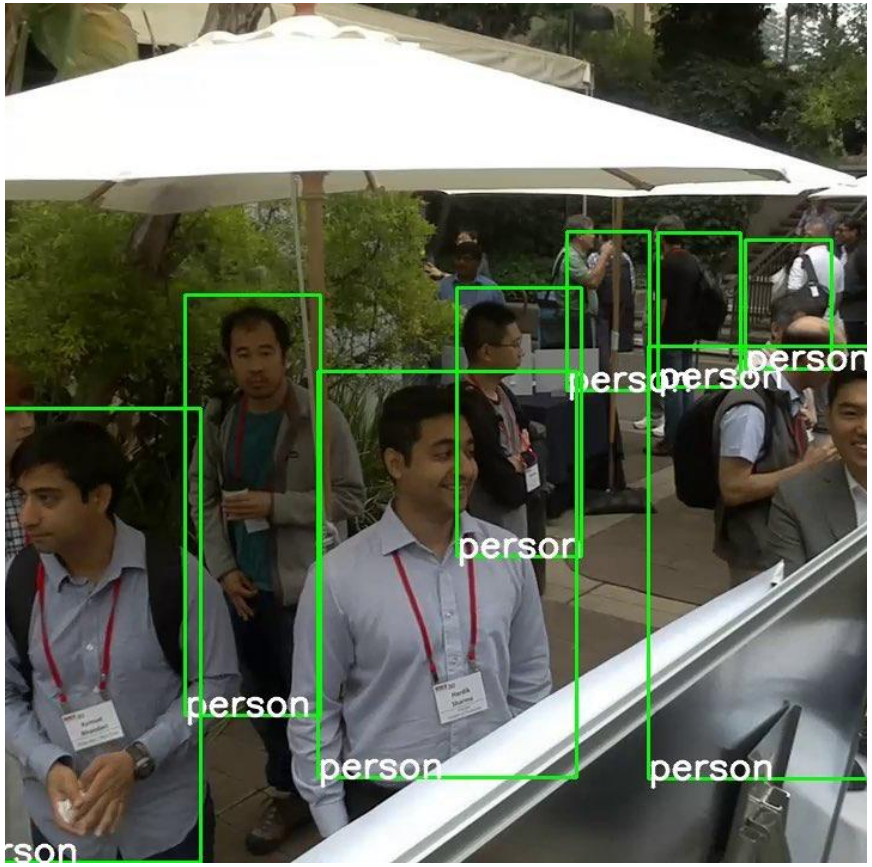
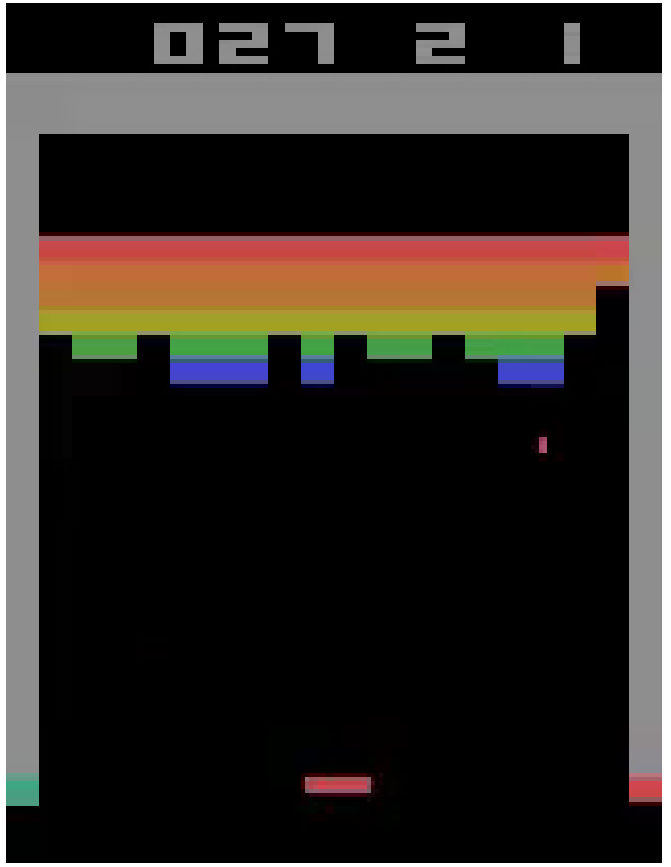
Faster Execution on CPUs and GPUs with *no Change in Hardware*

Reinforcement Learning and Adaptive Sampling for Optimized DNN Compilation

ICML 2019 Workshop on RL for Real Life

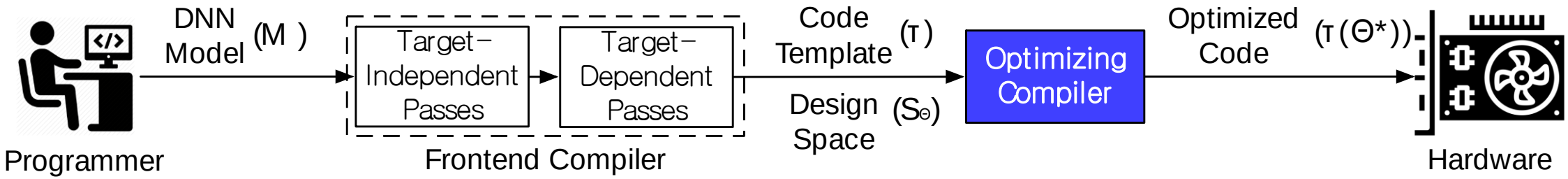
Neural network compilation





Deep learning compiler's objective is to
Speed Up deep learning models' inference, detection…

Neural network compilation



Code Template

```
@pragma unroll=N
for y.outer in range(0, height, height / tile_y):
    for x.outer in range(0, width, width / tile_x):
        for y.inner in range(0, tile_y, stride):
            for x.inner in range(0, tile_x, stride):
                output[y][x] = 0
                for ky in range(0, ry):
                    for kx in range(0, rx):
```

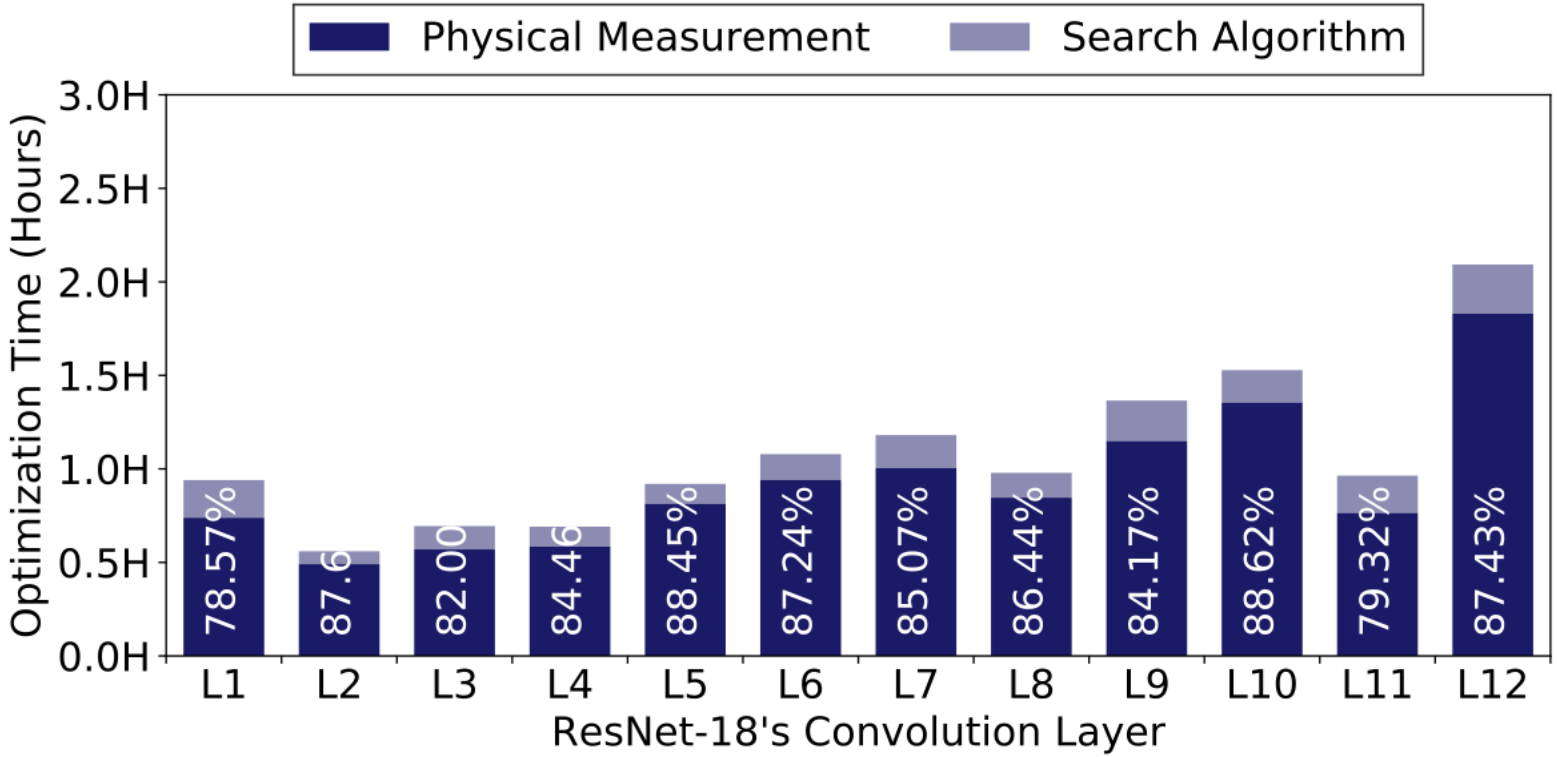
Design Space

Dimension / Knob	Description
tile_f, tile_y, tile_x	Tiling and binding for # of filters, height, and width of feature map
tile_rc, tile_ry, tile_rx	Tiling and binding for reduction axis such as channels, height, and width of filters
	p be eration
	ase

Optimizing Compiler's Objective is to

Find Optimal Configuration

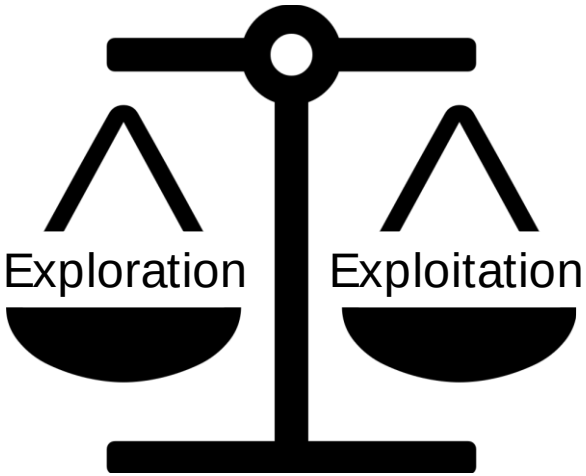
Prohibitive optimization time limits further innovation and diversity in neural networks



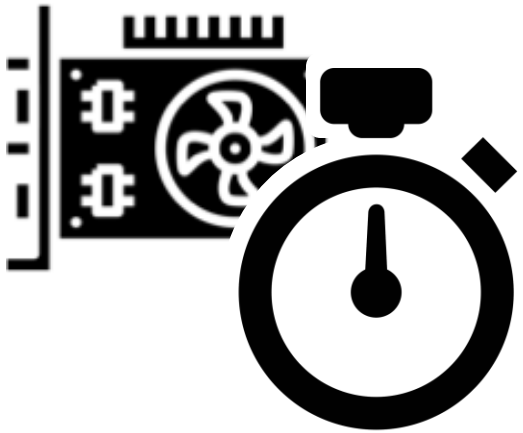
Takes Over **10 Hours** to Optimize **ResNet-18**

Challenges

1 How to improve efficacy of the search?



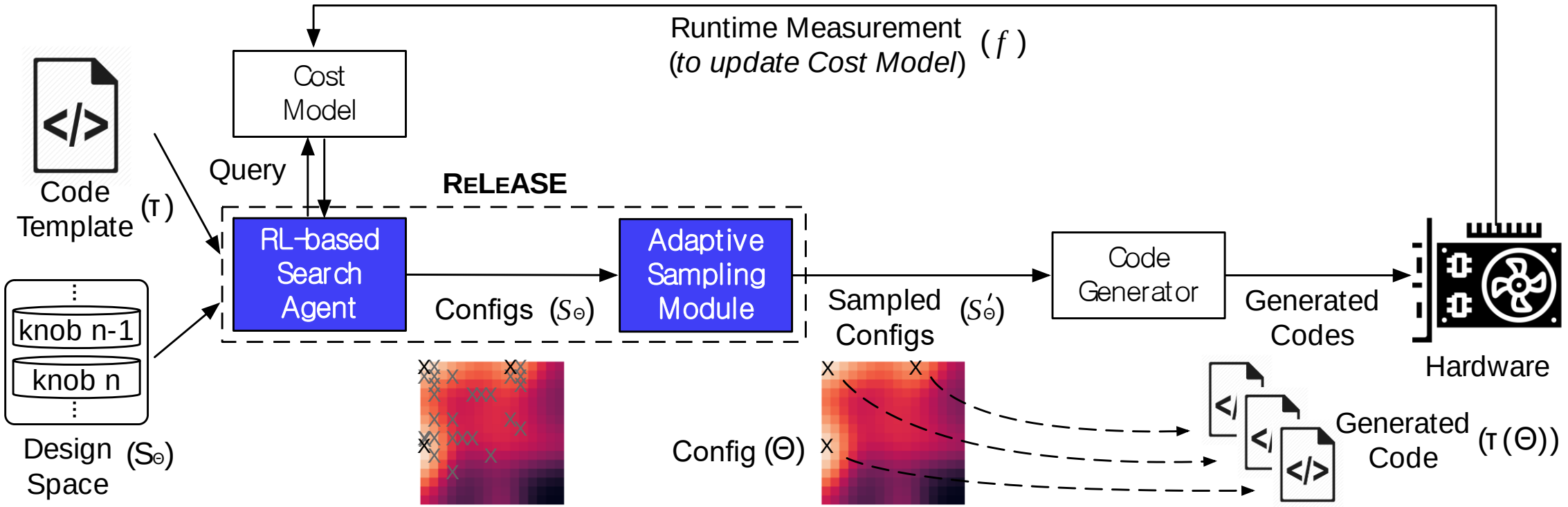
2 How to reduce costly hardware measurements?



We need to overcome two interrelated challenges

RELEASE

Reinforcement Learning Compiler with Adaptive Sampling for Efficiency

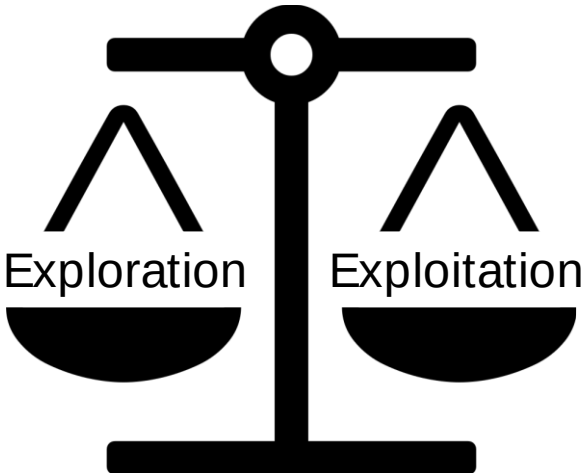


Reinforcement Learning improves **Search Efficacy**

Adaptive Sampling improves **Sample Efficiency**

Challenges

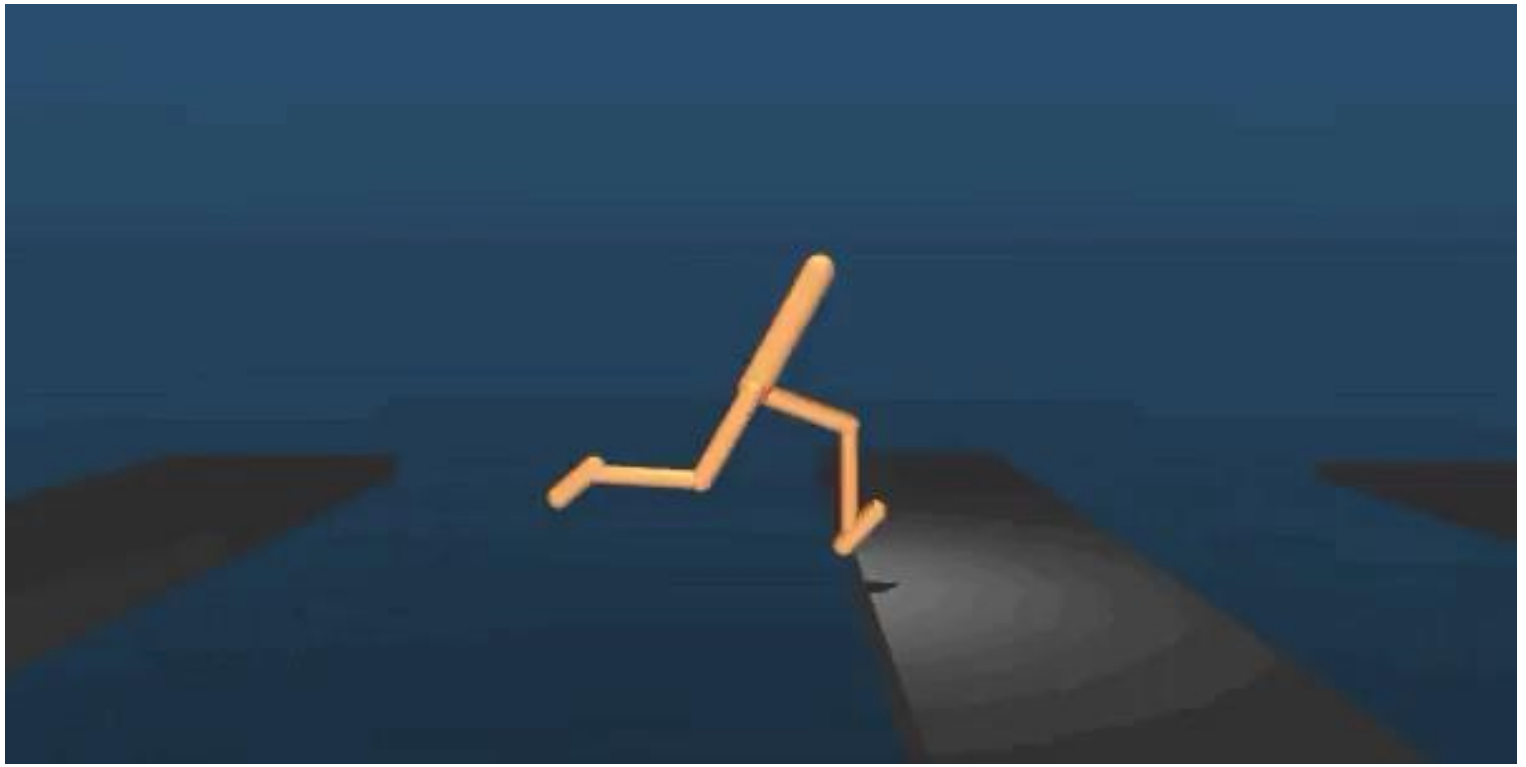
1 How to improve efficacy of the search?



2 How to reduce costly hardware measurements?

We need to overcome two interrelated challenges

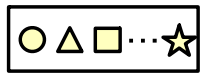
RL has been successful with autonomous decision making in complex settings



Apply **Reinforcement Learning** to **Search Process**

RL-based Search Agent

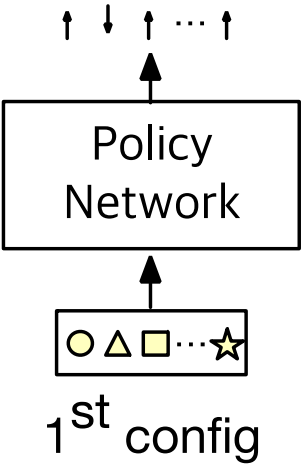
1. Start off with initial set of points (128)
   For each point…



1st config

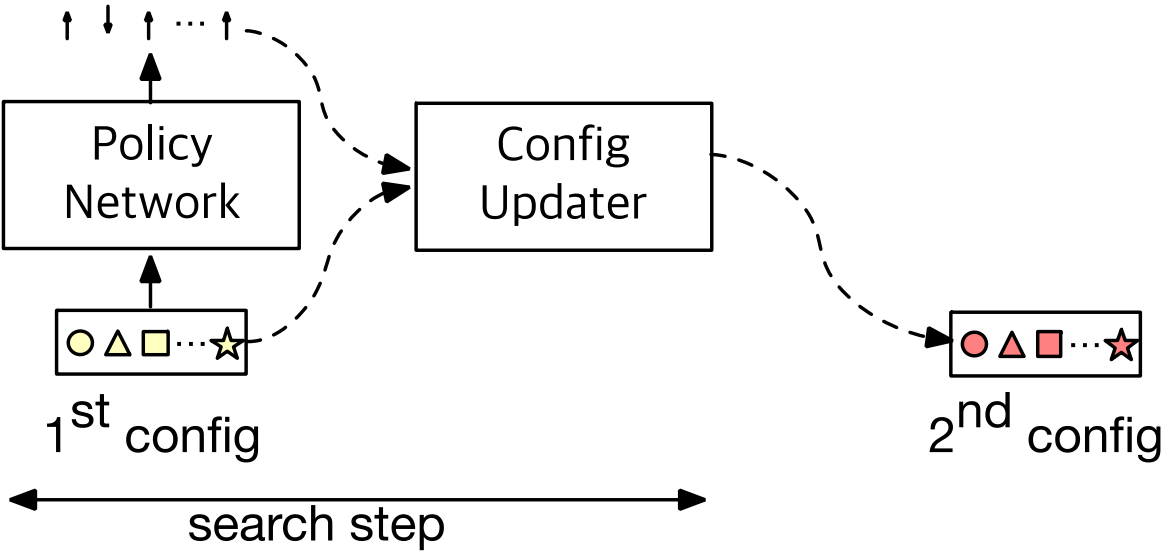
RL-based Search Agent

2. Pass it through policy network for action (increment ↑,
stay -,
decrement ↓)



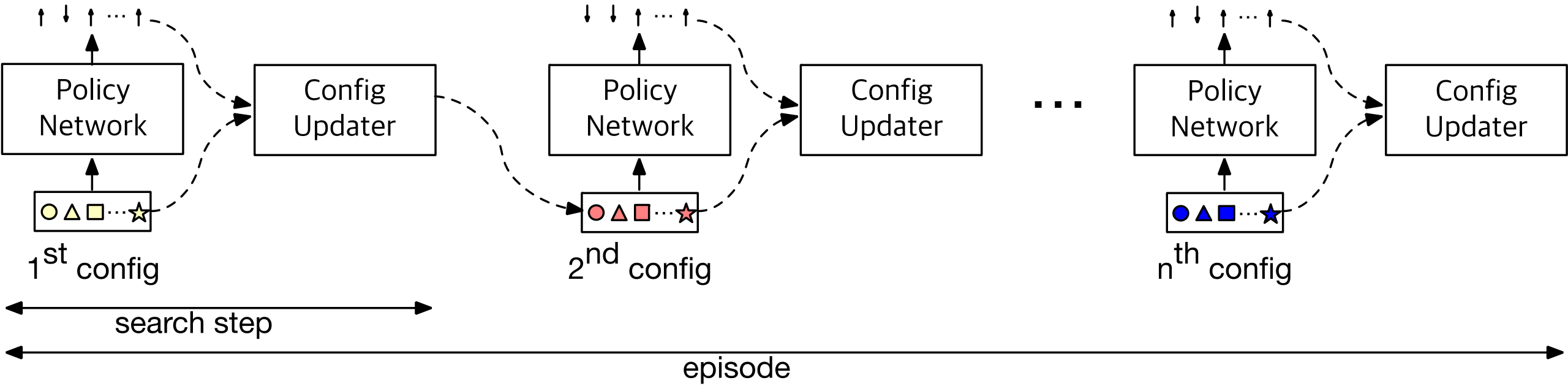
RL-based Search Agent

3. Use configuration updater to acquire new configuration



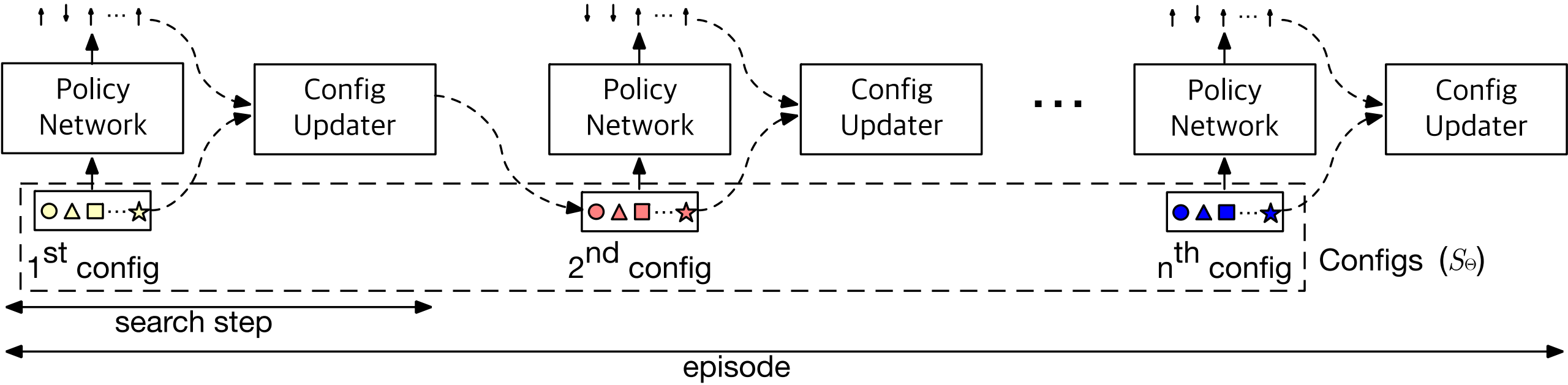
RL-based Search Agent

4. Repeat the search step 500 times = episode



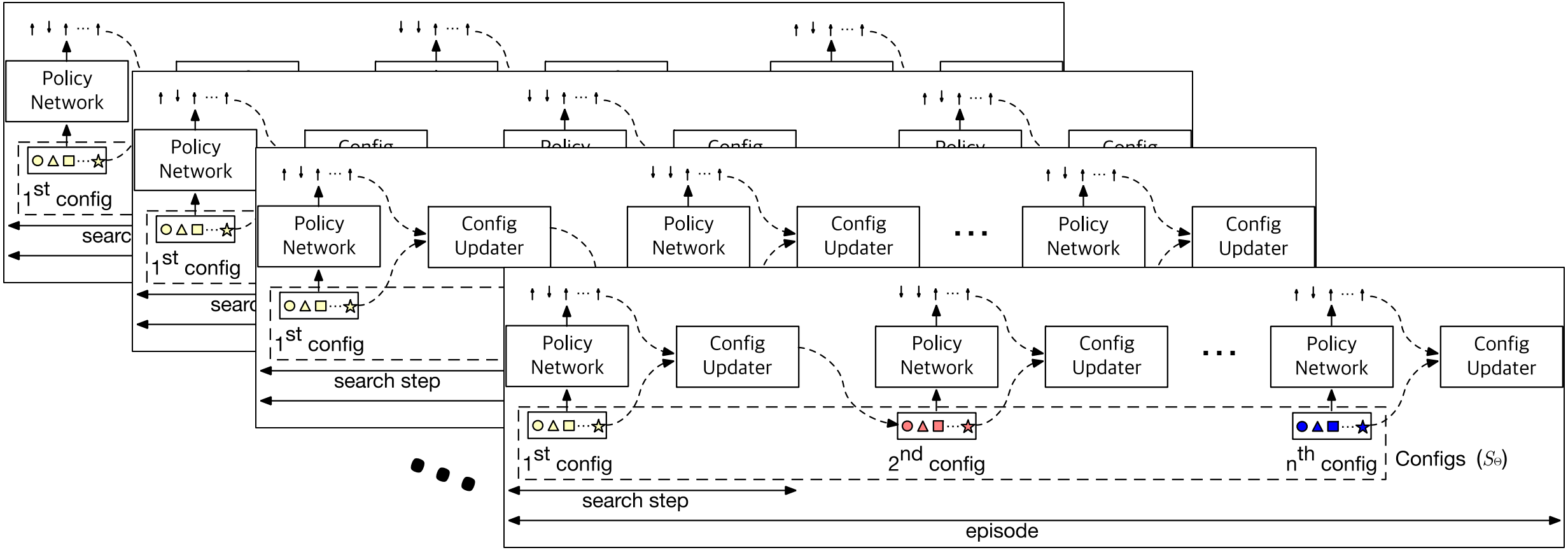
RL-based Search Agent

5. Update agent using reward (estimated hardware measurement)

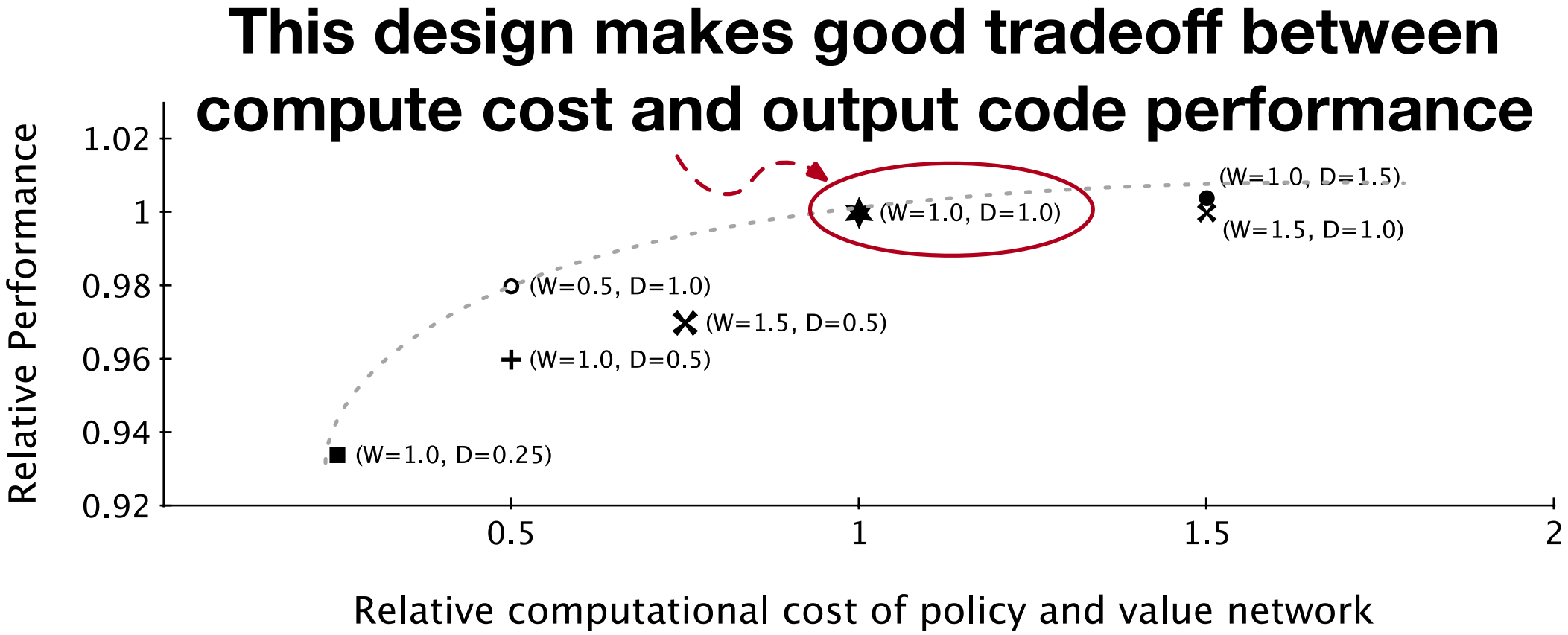


RL-based Search Agent

6. Repeat this episode for all 128 points



Wouldn’t RL be too slow?

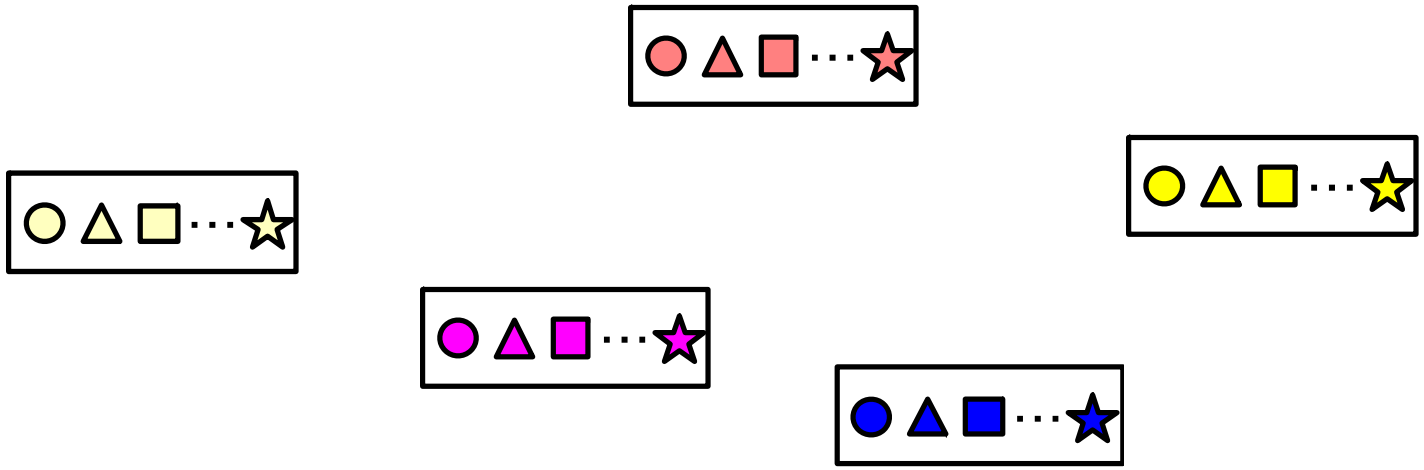


We conduct **design exploration** for **optimal design** in terms of **performance/computation tradeoff**

We have 128 x 500 = 64,000 samples

64,000 are too many to be measured on real hardware

1s x 64,000 = 18 hours

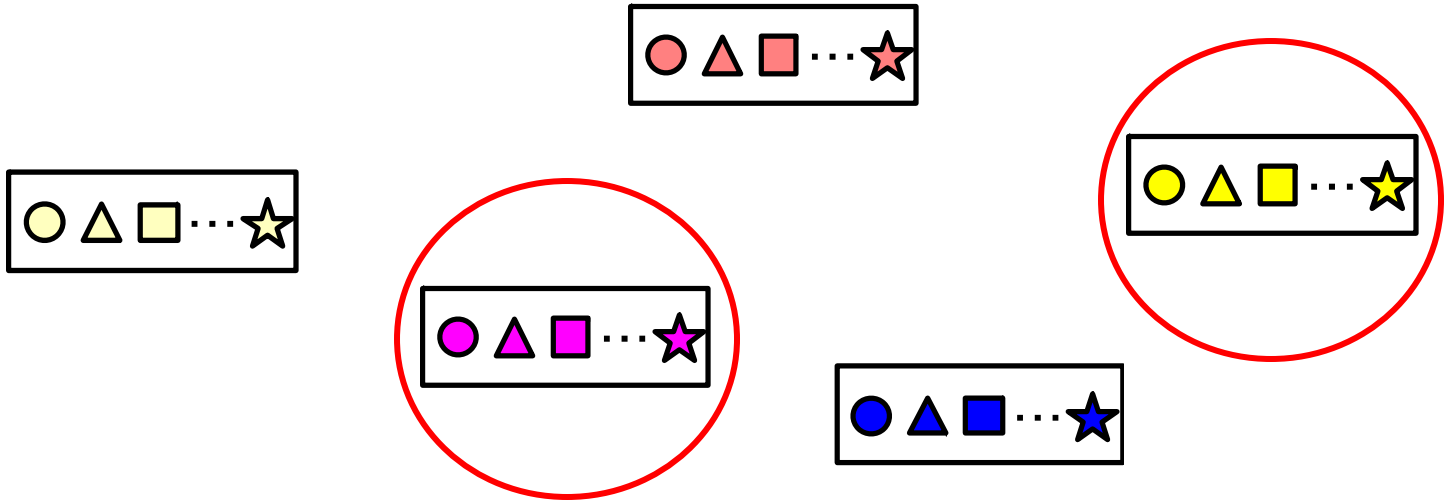


We have 128 x 500 = 64,000 samples

64,000 are too many to be measured on real hardware

1s x 64,000 = 18 hours

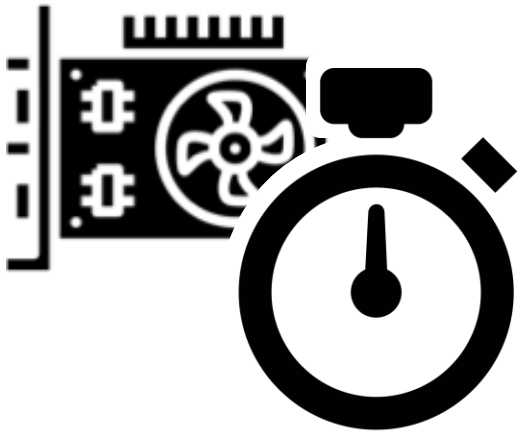
How to choose meaningful samples to measure on real hardware?



Challenges

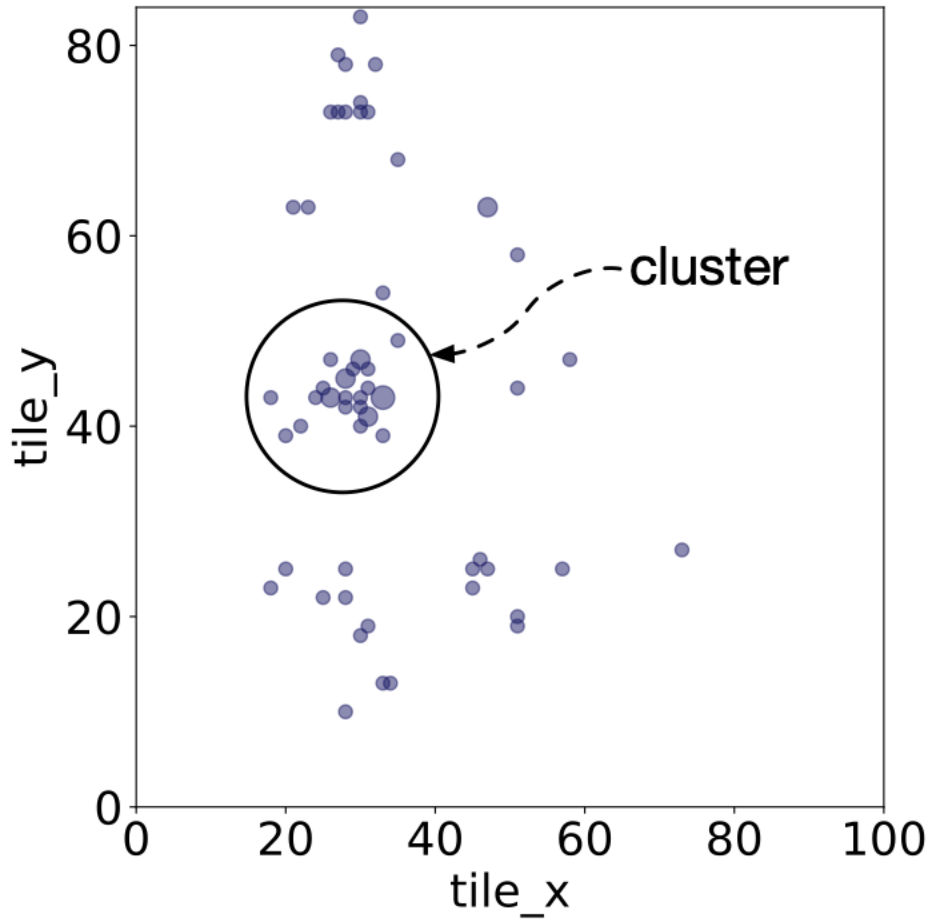
1 How to improve efficacy of the search?

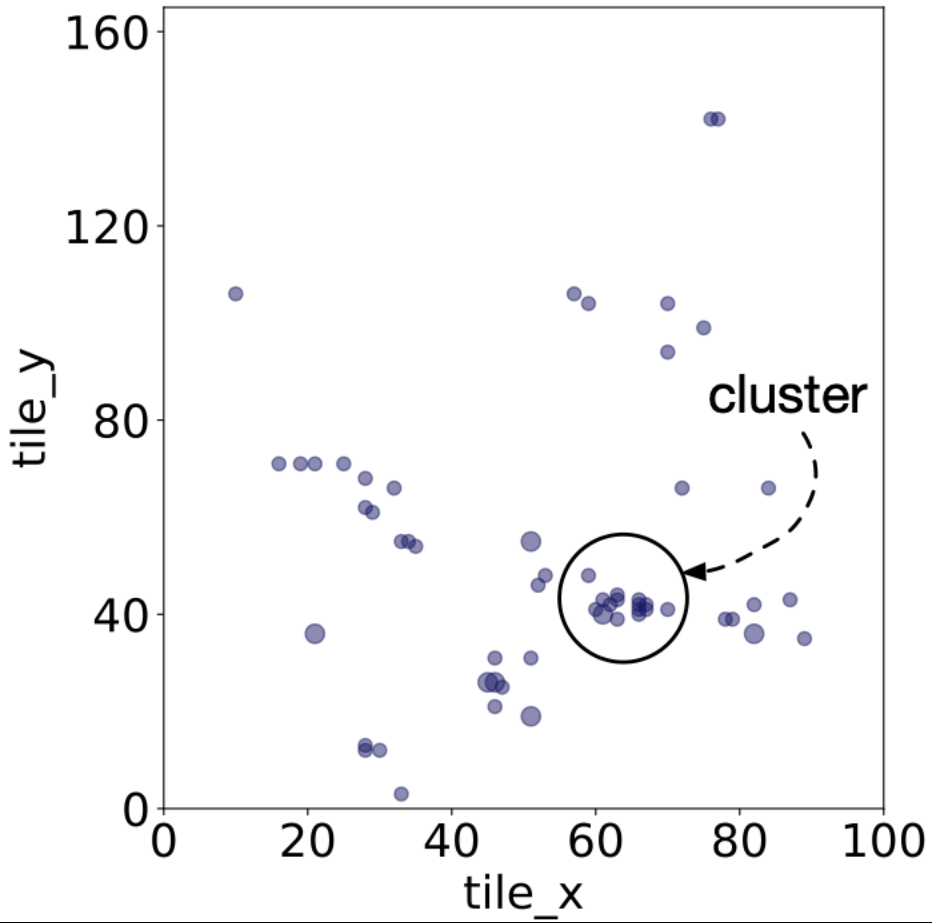
2 How to reduce costly hardware measurements?



We need to overcome two interrelated challenges

Analyzing the configuration sample distribution

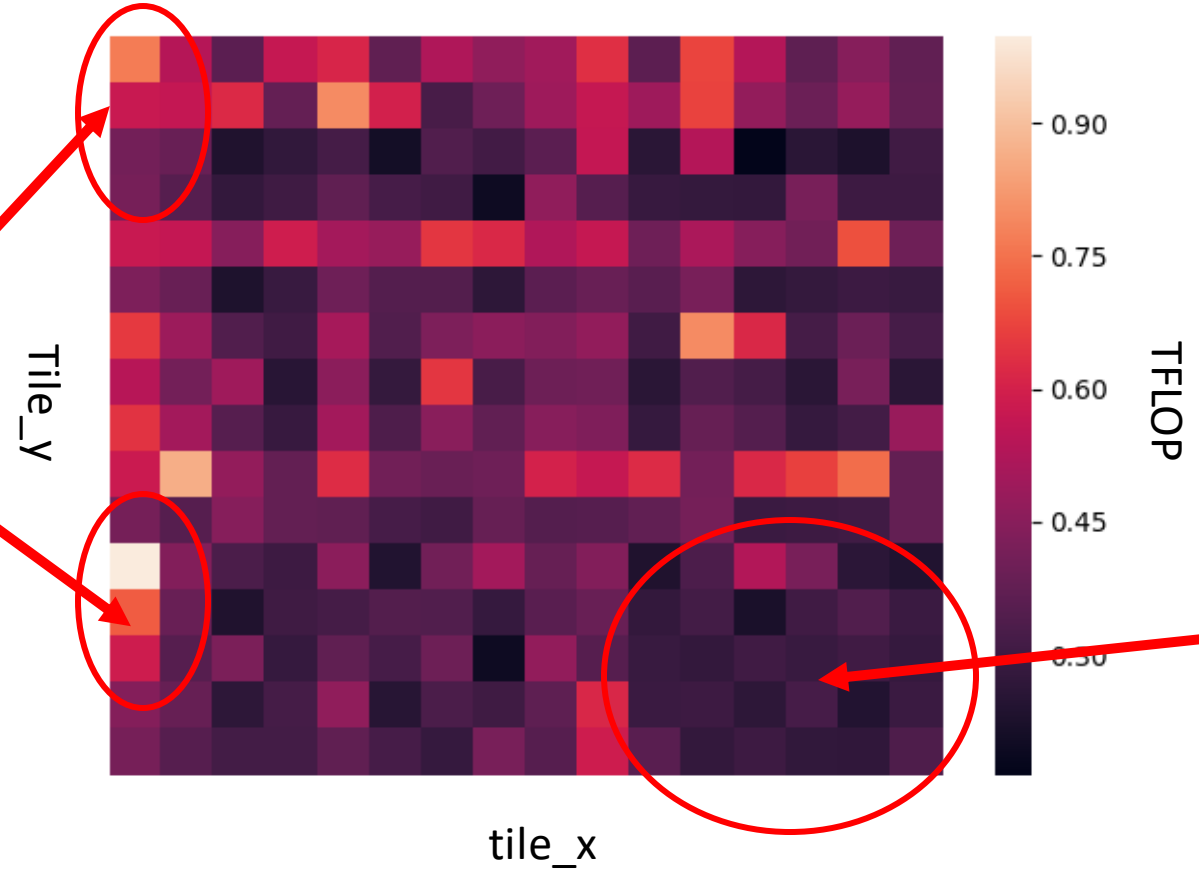




Reduce cluster to point that are each **representative of the cluster**

Wouldn’t that compromise performance?

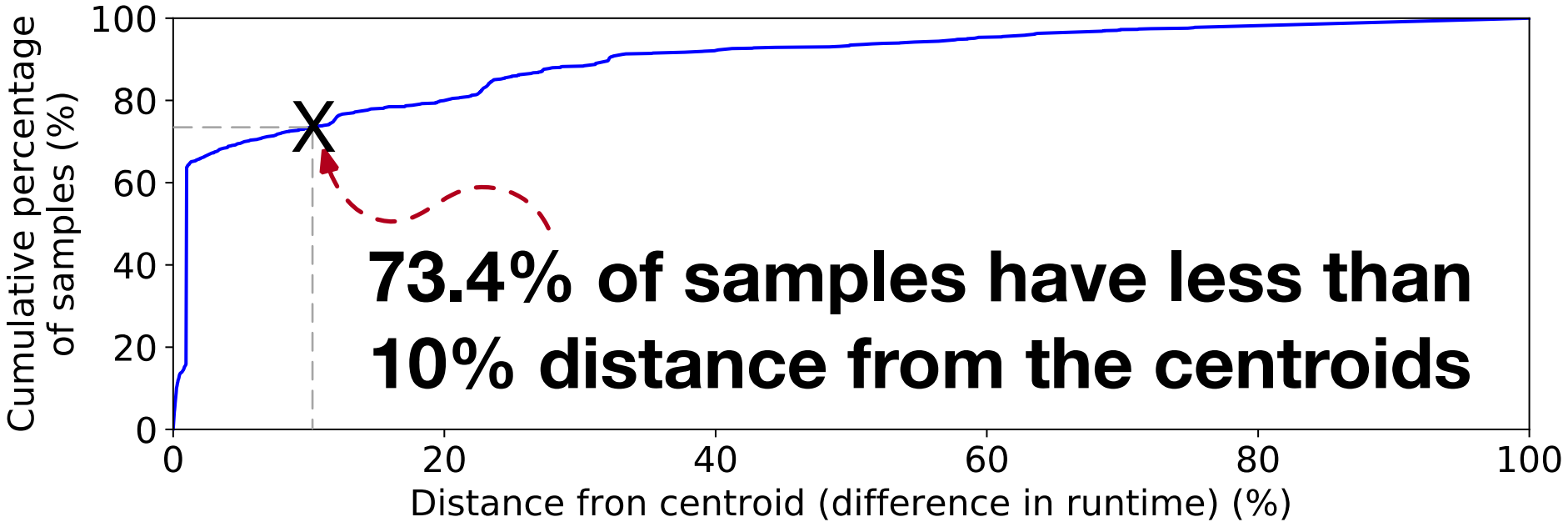
Relatively good region



Relatively bad region

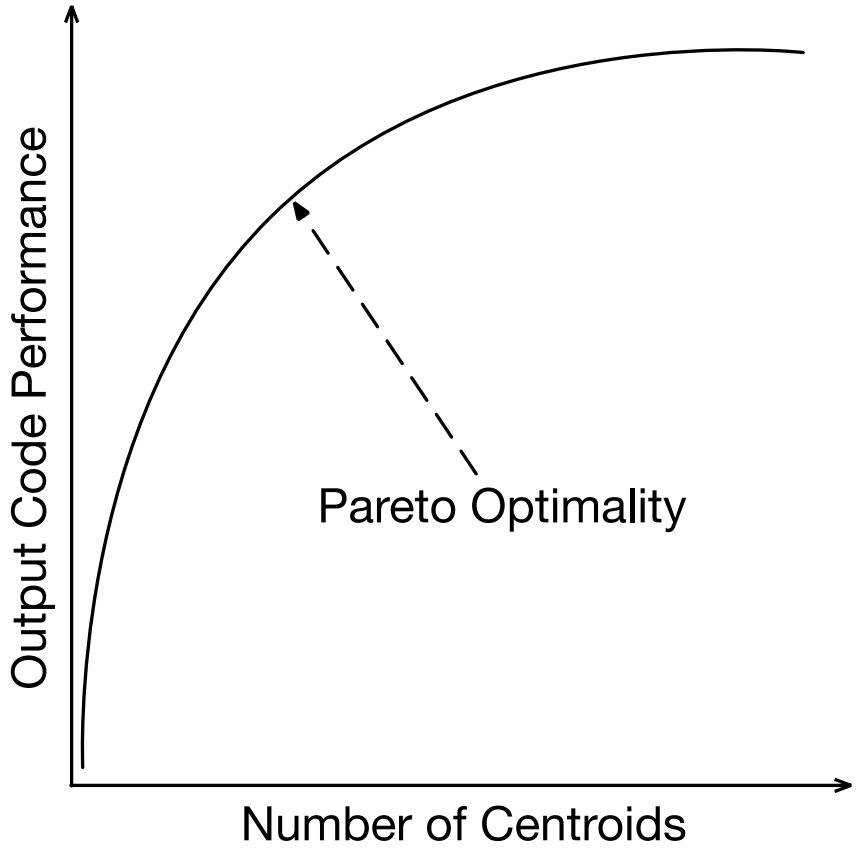
Although the **search space** is **unstructured**
changes along the axis are **gradual**

Wouldn’t that compromise performance?



More than 73% of the samples have **less than 10% distance from the centroids** (Empirical Evidence)

How many clusters?



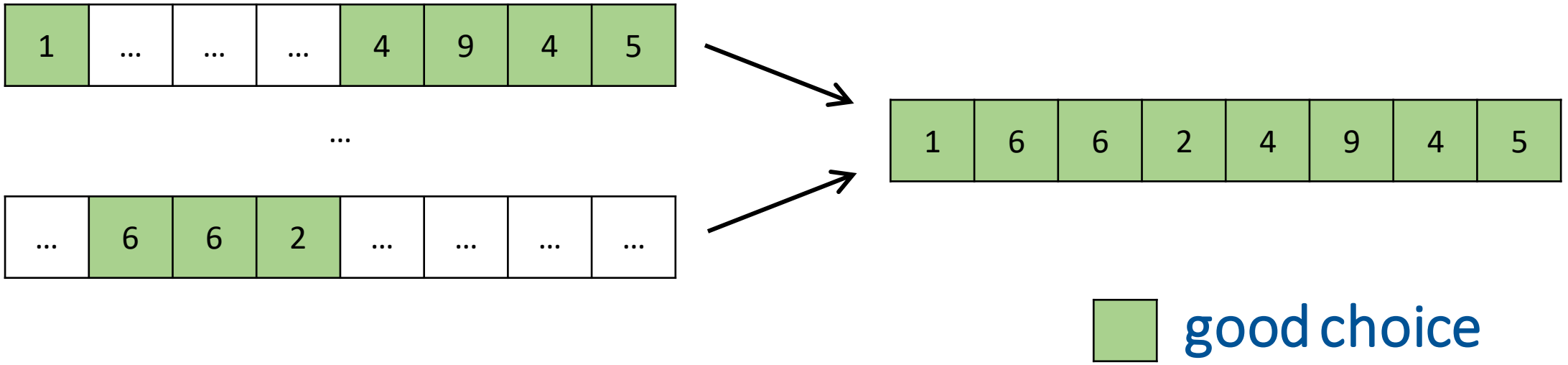
Less centroids = Faster optimization
More centroids = Better output code perf.

Iterate through different number of centroids and find knee of the loss curve

knee = optimal number of clusters

Trick to improve sampling

Frequent choices for each dimension = Potentially most suitable choice



Generate synthetic samples from frequent choices
(c.f. Crossover in Genetic Algorithm)

Evaluation

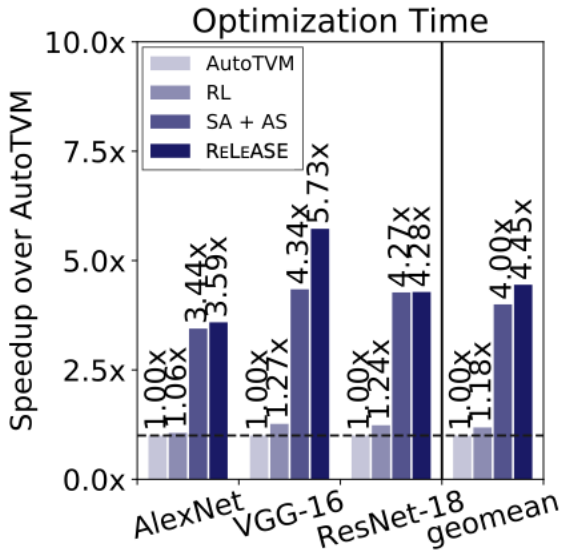
Benchmarks for RELEASE

Name	Description	Layer Type	Dataset	No. of Tasks
AlexNet	Image Classification	Convolution	ImageNet	5
VGG-16	Image Classification	Convolution		9
ResNet-18	Image Classification	Convolution		12

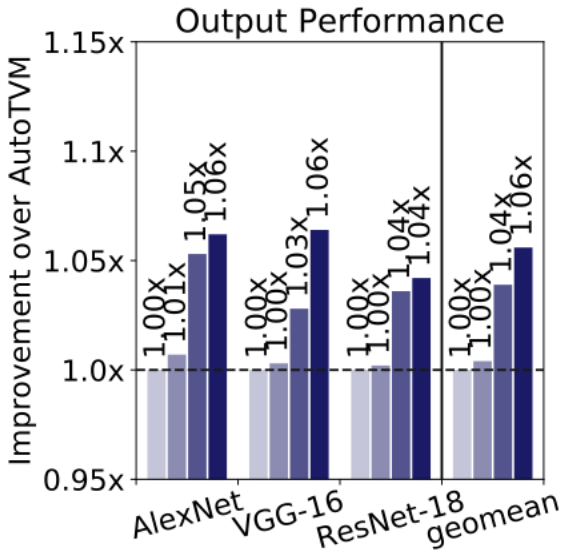
Hardware

Name	Description
Titan Xp	Server-class GPU

ReLeASE



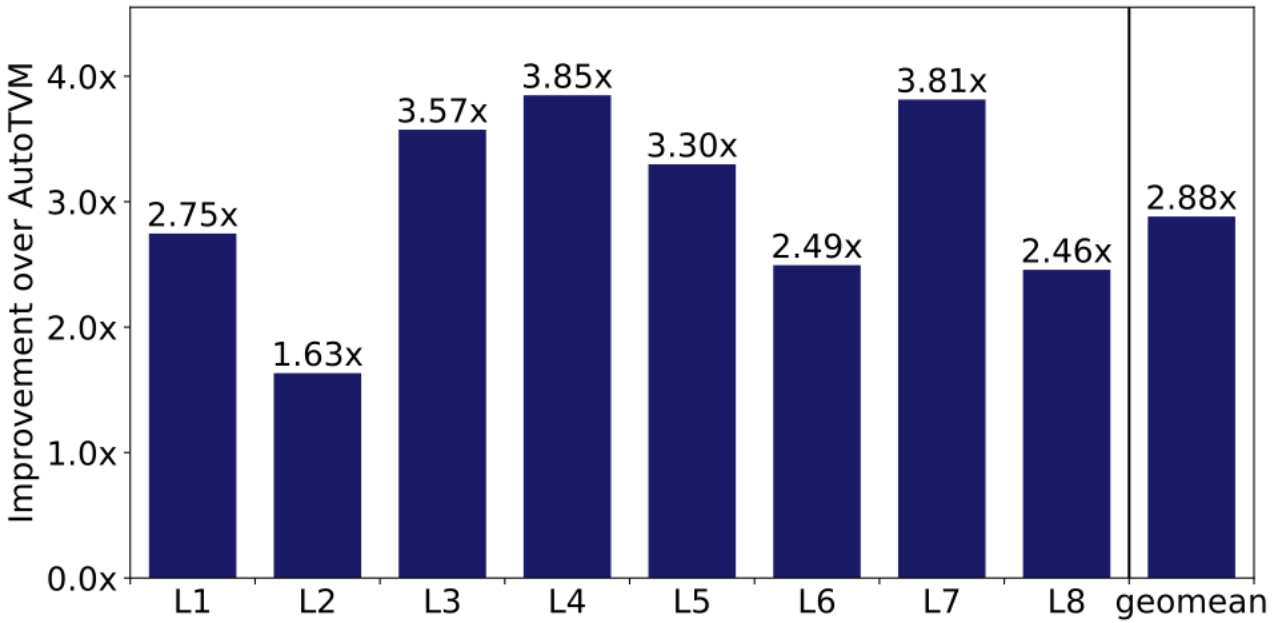
Network	AutoTVM [1]	RL	SA + AS	ReLeASE
AlexNet [2]	4.31 Hours	4.06 Hours	1.25 Hours	**1.20 Hours**
VGG-16 [4]	11.2 Hours	8.82 Hours	2.57 Hours	**1.95 Hours**
ResNet [5]	9.13 Hours	7.39 Hours	2.14 Hours	**2.13 Hours**



Network	AutoTVM [1]	RL	SA + AS	ReLeASE
AlexNet [2]	1.0277 ms	1.0207 ms	0.9762 ms	**0.9673 ms**
VGG-16 [4]	3.9829 ms	3.9710 ms	3.8733 ms	**3.8458 ms**
ResNet [5]	1.0258 ms	0.9897 ms	0.9897 ms	**0.9831 ms**

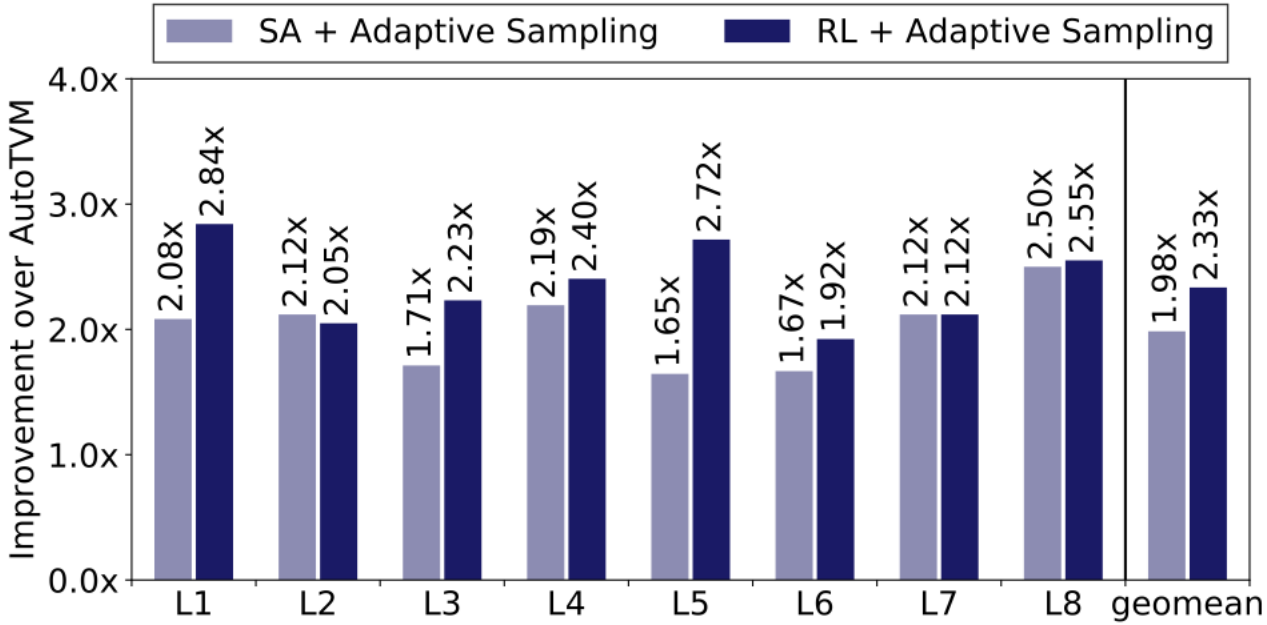
ReLeASE shows **4.00x improvement** in **optimization time**
with **5.6% improvement** in **output performance** than AutoTVM

RL-based Search Agent



Reinforcement Learning based search shows **2.88x improvement** in **search steps** over **Simulated Annealing** in AutoTVM

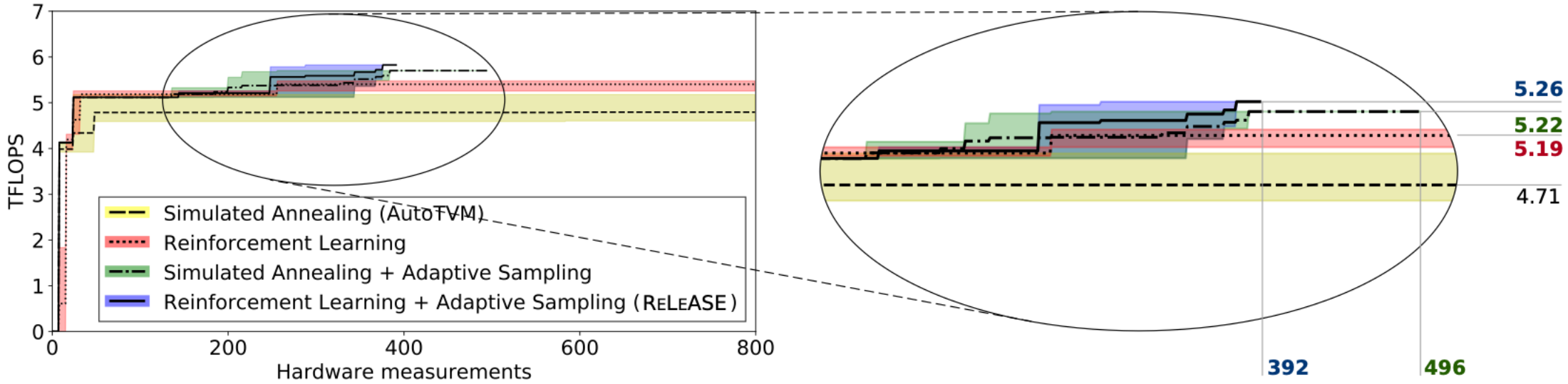
Adaptive Sampling Module



Adaptive Sampling shows **2.33x improvement** for RL-based search in **sample efficiency** over **Naïve sampling**

Adaptive Sampling even shows **1.98x improvement** for **Simulated Annealing** in AutoTVM

Putting it all together



RELEASE achieves **better performance**
with **less hardware measurements** than AutoTVM

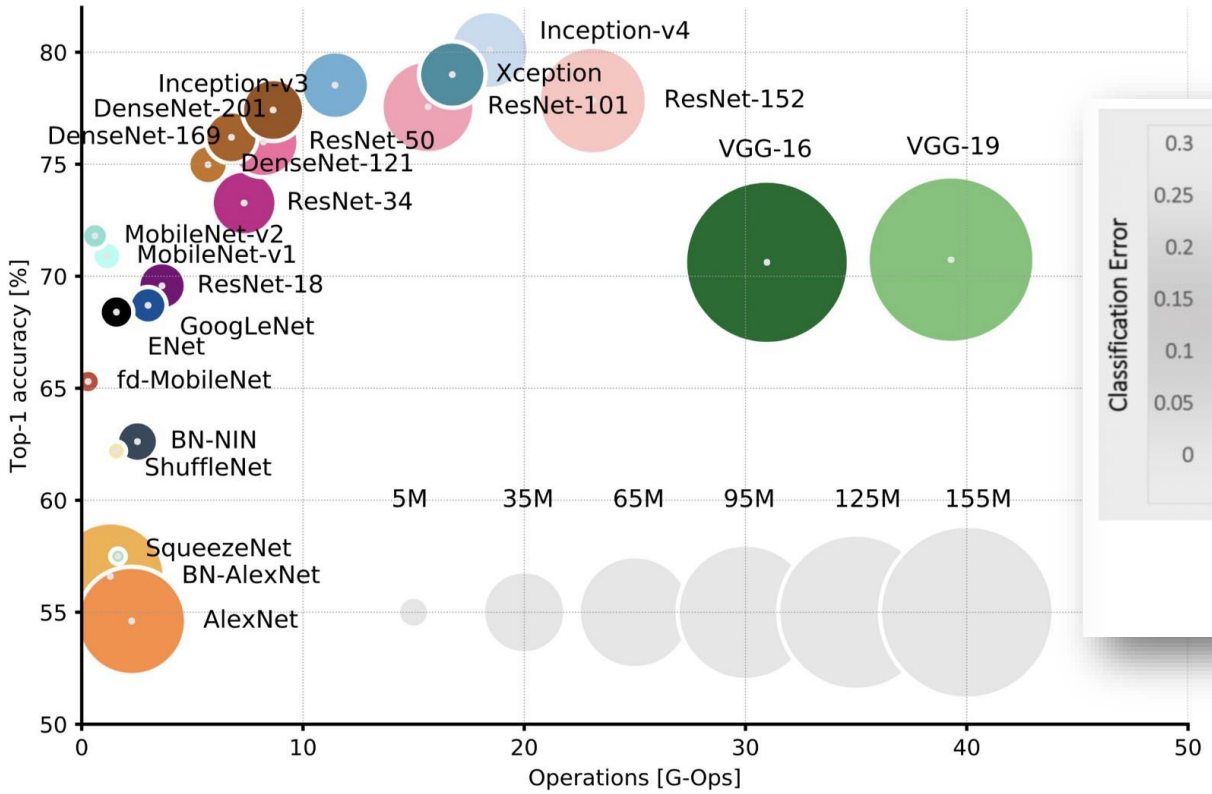
We formulate optimizing compilation of DNNs as a **Reinforcement Learning** problem in contrast to simulated annealing taking **fewer steps to converge to better or same quality solution**.

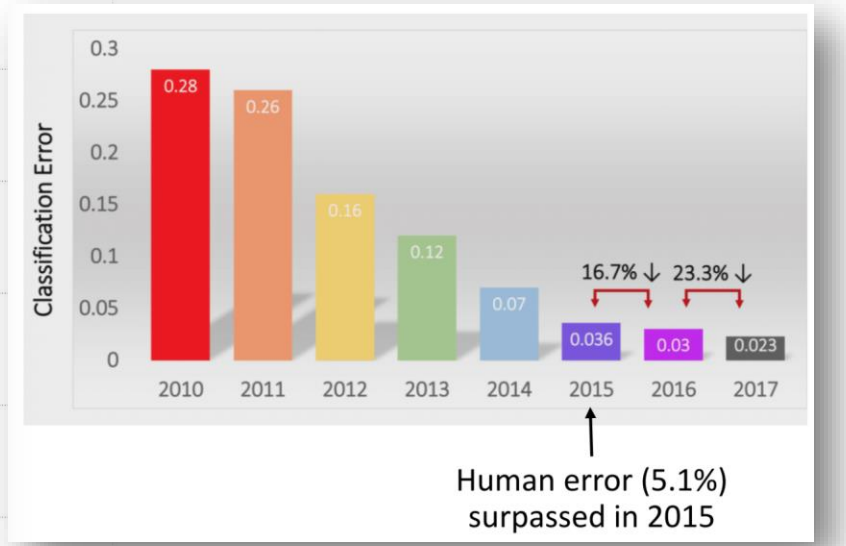
We devise an **Adaptive Sampling** algorithm that leverages clustering to **focus measurements to representative samples**, reducing number of measurements while keeping their relevance to search high.

ReLeQ

An Automatic Reinforcement Learning Approach for Deep Quantization of Neural Networks

Background

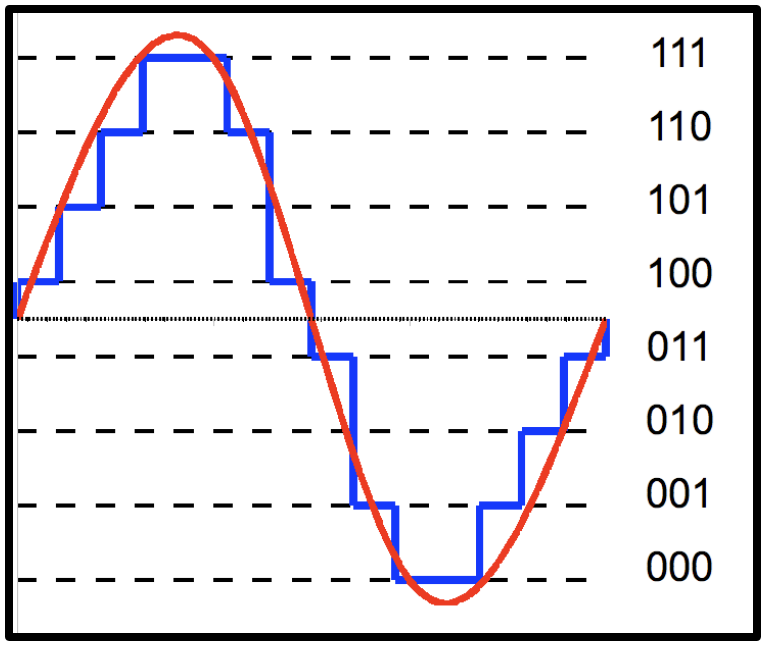




Deep Neural Networks become increasingly complex with higher computation and memory cost

*Alfredo Canziani, An Analysis of Deep Neural Network Models for Practical Applications, ArXiv, 2016

One solution: Quantization



- **Round-to-nearest**

$$= \begin{cases} \lfloor x \rfloor & \text{if } \lfloor x \rfloor \le x \le \lfloor x \rfloor + \frac{\epsilon}{2} \\ \lfloor x \rfloor + \epsilon & \text{if } \lfloor x \rfloor + \frac{\epsilon}{2} < x \le \lfloor x \rfloor + \epsilon \end{cases}$$

- **Stochastic rounding**

$$= \begin{cases} \lfloor x \rfloor & \text{w.p. } 1 - \frac{x - \lfloor x \rfloor}{\epsilon} \\ \lfloor x \rfloor + \epsilon & \text{w.p. } \frac{x - \lfloor x \rfloor}{\epsilon} \end{cases}$$

Largest quantization level closer to x

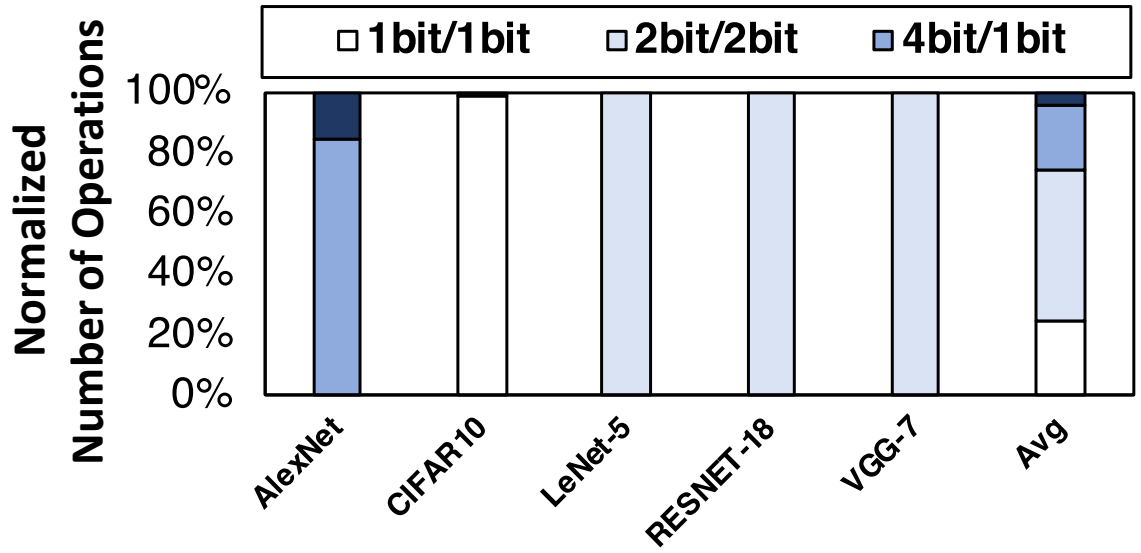
Quantization step

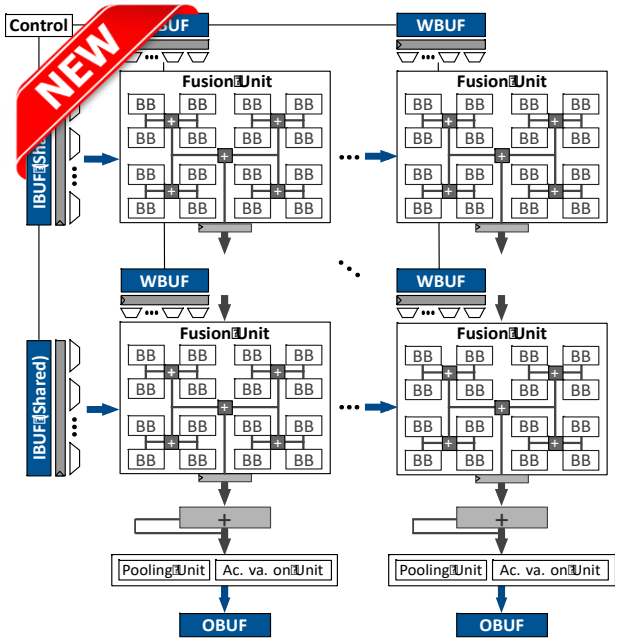
S. Gupta, et. al., Deep Learning with Limited Numerical Precision

Quantization reduces the bitwidth of the operations which leads to a reduction in both the **storage** and **compute** requirements of the network

Bitwidth Flexibility is Necessary for Accuracy

Hardware perspective





BitFusion [ISCA 2018]

Bitwidth varies **across DNN models**,
and even **within layers of the same DNN**

Motivation (1)

Quantization Levels

Example

Vgg-16 exposes a hyper-parameter space of:
$8^{16} > 2.8 \times 10^{14}$
For 16 layers and 8 possible quantization levels.

✗ **Laborious**

The design space for finding the optimal quantization level for each layer of DNNs is **prohibitively large**

Motivation (2)

Quantization Levels ☑ **Automatic Quantization**

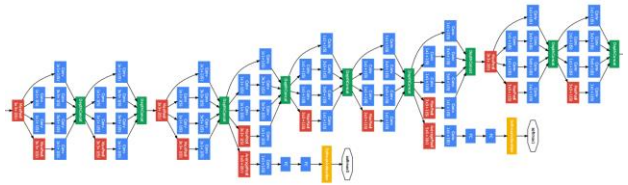
Reinforcement Learning MAY efficiently explore this prohibitively large design space of quantization levels

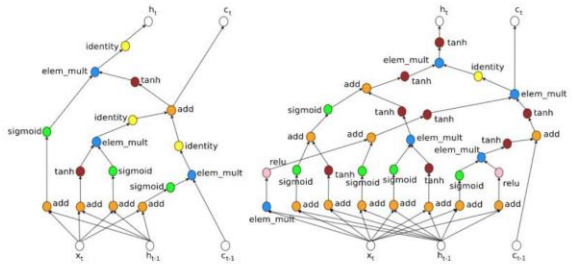
Motivation (3)

Two major challenges for deep Neural Network

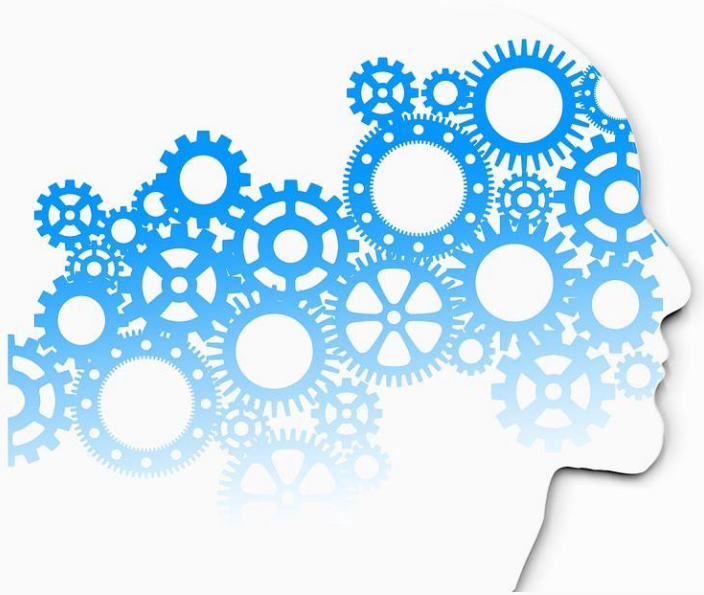
Compute/Memory Intensity

wider, **deeper** with increased **complexity**





Hyper-parameter Optimization

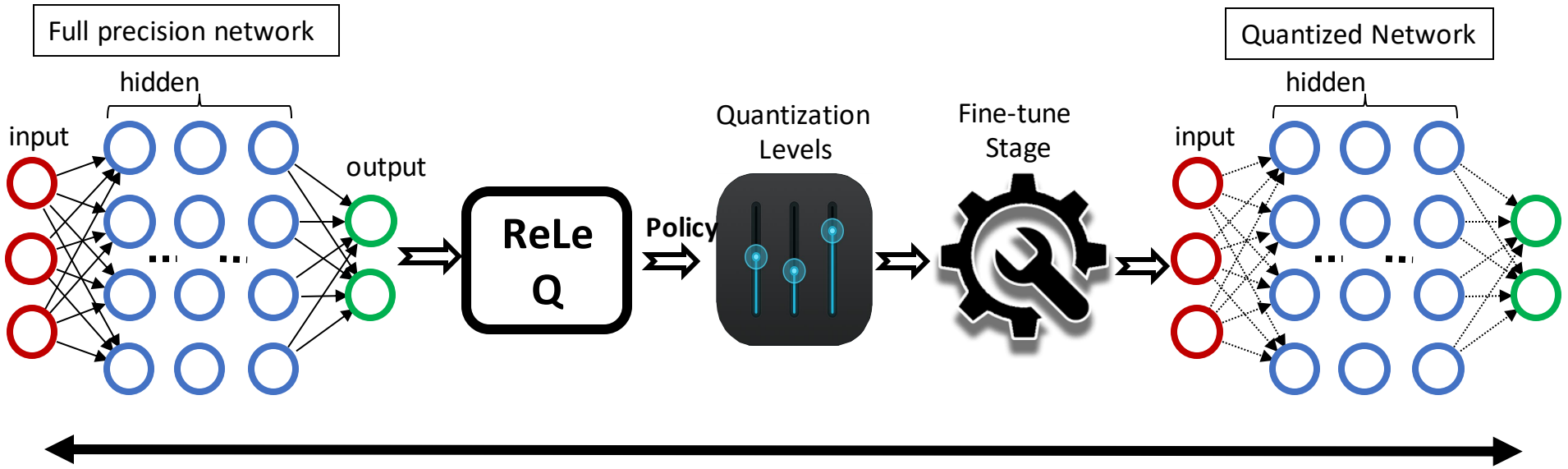


ReLeQ = Quantization + Automated Machine Learning

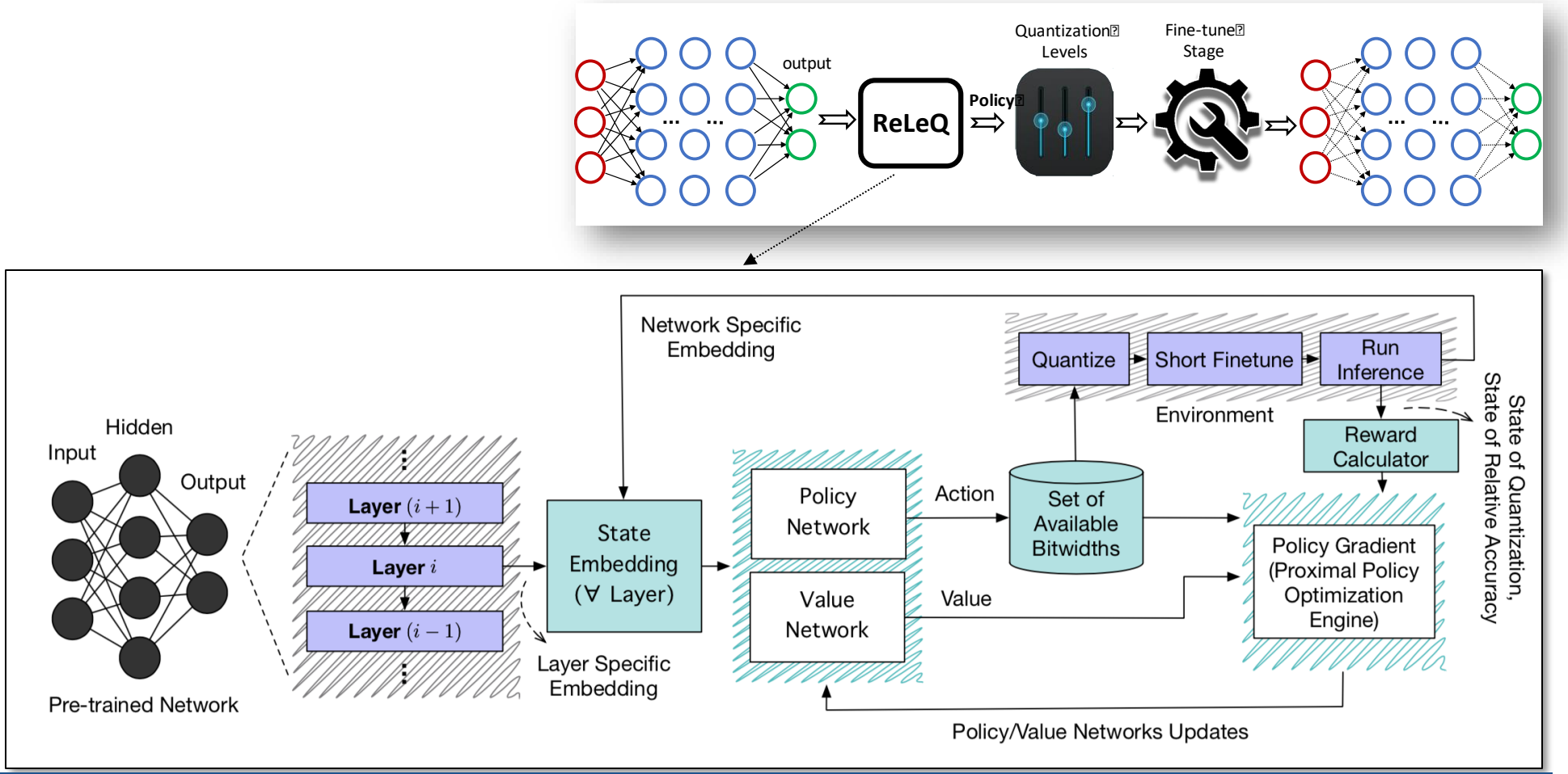
ReLeQ + BitFusion
= End-to-end Quantization Utility for Deep Neural Networks

What is ReLeQ?

An Automatic Reinforcement Learning Approach for Deep Quantization of Neural Networks

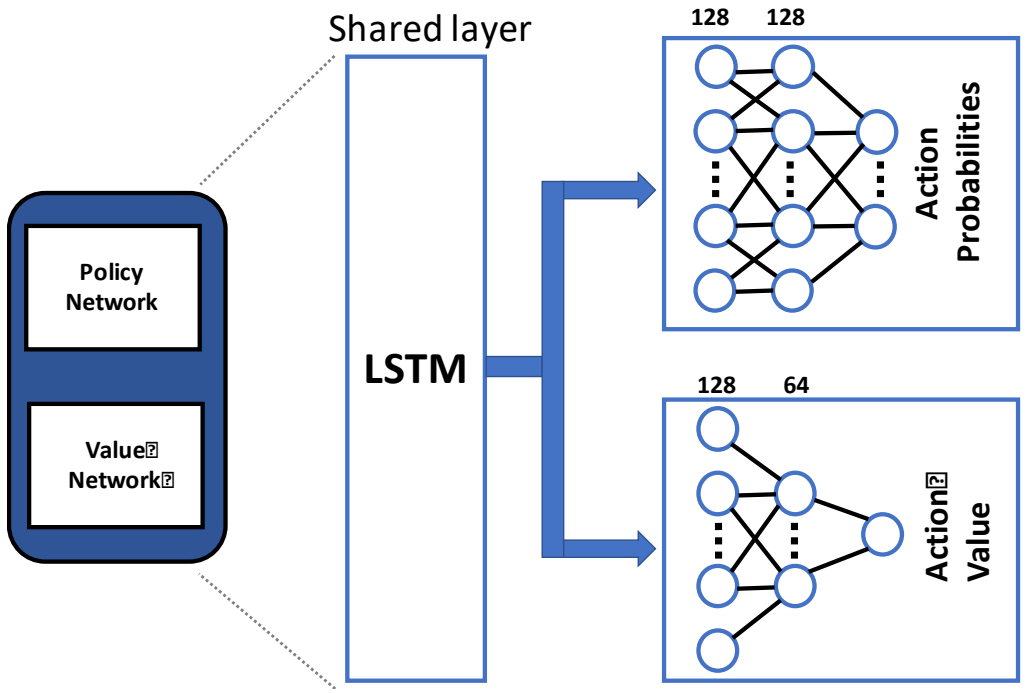


ReLeQ Framework



Overview of ReLeQ, which starts from a pre-trained network and delivers its corresponding deeply quantized network

ReLeQ Policy and Value Networks



Policy Network

Value Network

Table 1. PPO Hyperparameters used in ReLeQ.

Hyperparameter	Value
Adam Step Size	1×10^{-4}
Discount Factor	0.9
GAE Parameter	0.99
Number of Epochs	3
Clipping Parameter	0.3

- ✓ LSTM-based network is used to consider the **effect of quantization of network layers collectively** on the final classification accuracy
- ✓ LSTM-based network also **improves ReLeQ convergence**

Problem Formulation

$$Quantization_{state} = \frac{\sum_{l=0}^{L}[(n_l^w \times \frac{E_{MemoryAccess}}{E_{MAcc}} + n_l^{MAcc}) \times n_l^{bits}]}{\sum_{l=0}^{L}[n_l^w \times \frac{E_{MemoryAccess}}{E_{MAcc}} + n_l^{MAcc}] \times n_{max}^{bits}}$$

$$Accuracy_{state} = \frac{Acc_{curr}}{Acc_{fp}}$$

Parameters	Description
$\mathbf{l}$	Layer index
$\mathbf{L}$	Number of layers
$\mathbf{n_l^w}$	#weights of layer l
$\mathbf{n_l^{bits}}$	#bits of layer l
$\mathbf{n_{max}^{bits}}$	Max #bits
$\mathbf{n_l^{MAcc}}$	#MAcc operations
$\mathbf{E_{MemoryAccess}}$	Memory Access Energy
$\mathbf{E_{MAcc}}$	MAcc Energy
$\mathbf{Acc_{curr}}$	Current Accuracy
$\mathbf{Acc_{fp}}$	Full Precision Accuracy

The objective of ReLeQ is **minimizing the quantization state** while **maximizing the accuracy state**

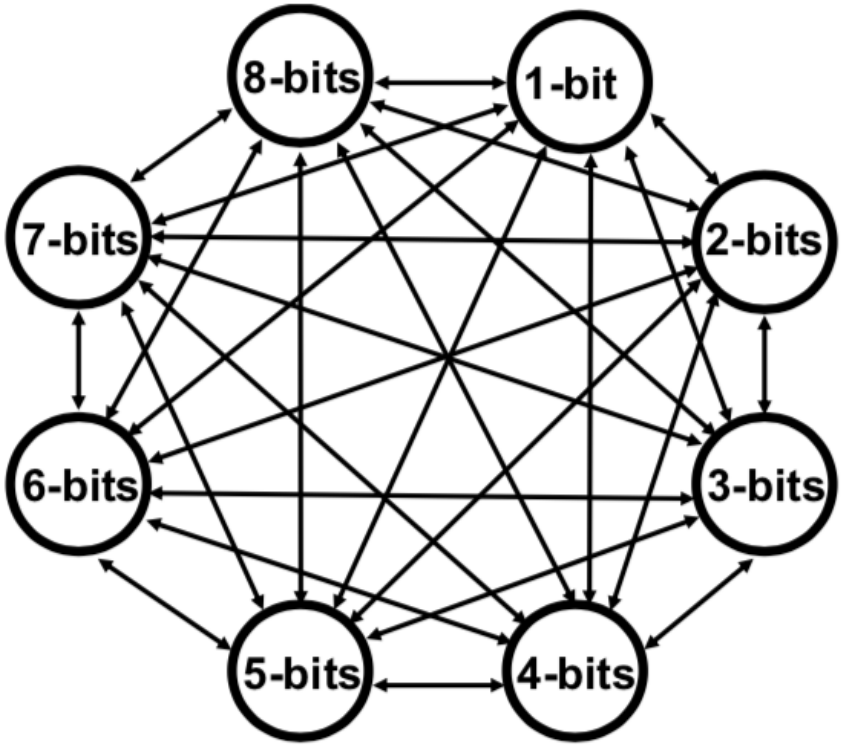
State Space

	Layer Specific	Network Specific
Static	Layer Index	N/A
	Layer Size	
	Initial Weight Statistics (Standard Deviation)	
Dynamic	Quantization Level (Number of Bits)	Quantization State
		Accuracy State

- ✓ The size and dimension of the network layers are the most important embedding for convergence
- ✓ The initial weights standard deviations are the second most important parameter for convergence

Action Space

Flexible action space

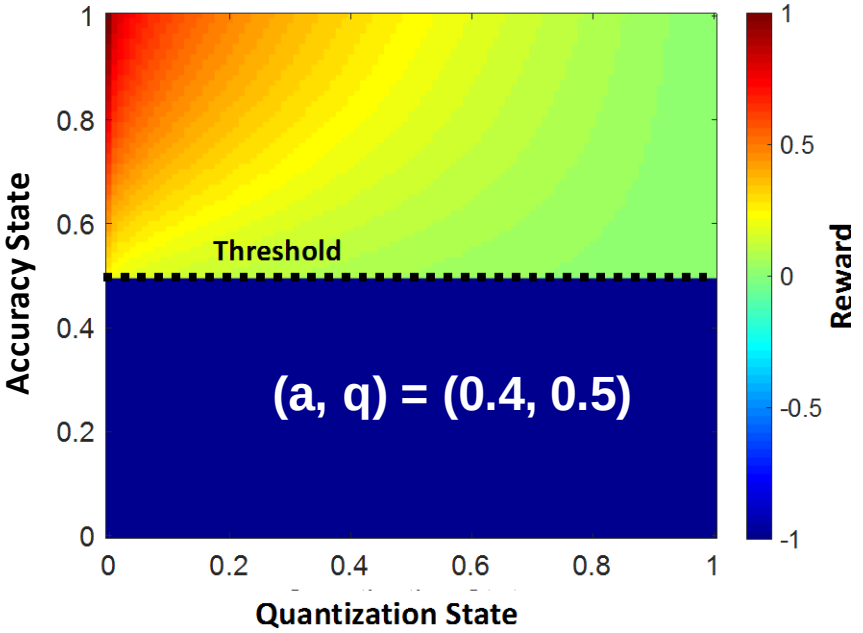


The proposed action space grants the agent the **flexibility of moving between different bitwidths** per each step which promotes better exploration

ReLeQ Reward Shaping

Q_{st}: Quantization State
Acc_{st}: Accruacy State

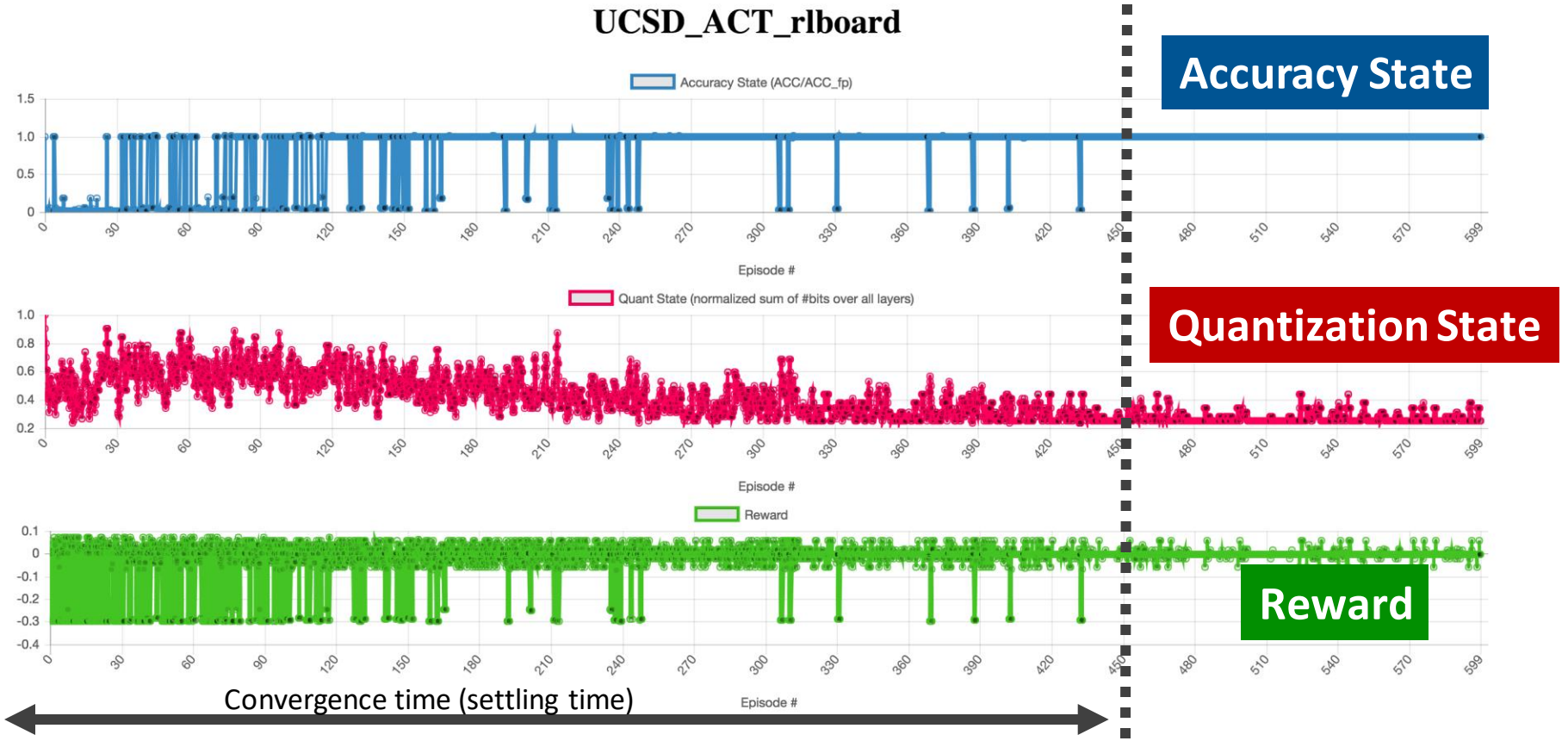
$Reward = 1 - (Q_{st})\hat{}a$
if $(Acc_{st} < th)$ ***then***
    $Reward = -1$ ⟶ 
else
    $Acc_{discount} = \max(Acc_{st}, th)^{(q/\max(Acc_{st},th))}$
    $Reward = Reward \times Acc_{discount}$
end if



*a, and q control the tradeoff between the accuracy quantization states

ReLeQ Reward is a function of:
(1) **Accuracy state** and (2) **Quantization state**

ReLeQ Development: Ideas ..

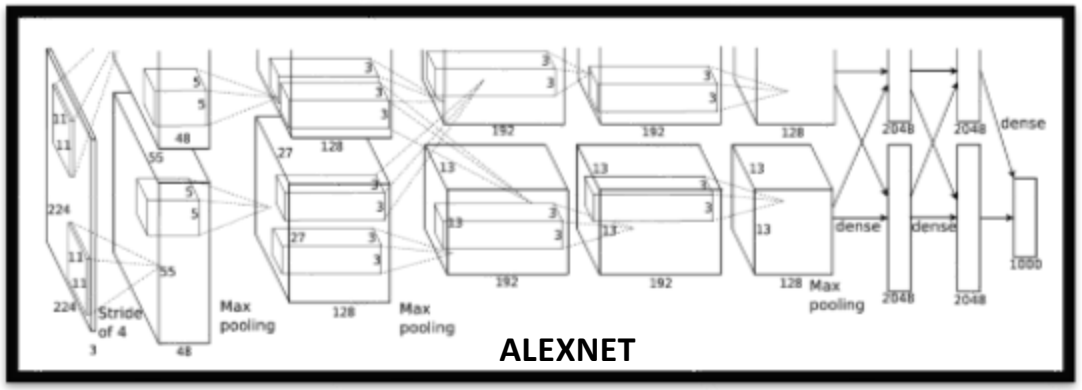


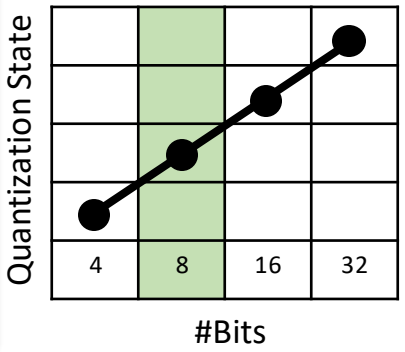
http://kkachi.ucsd.edu:5000/rlboard

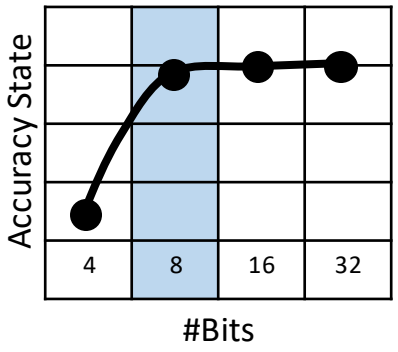
(1) RL Board (Observability)

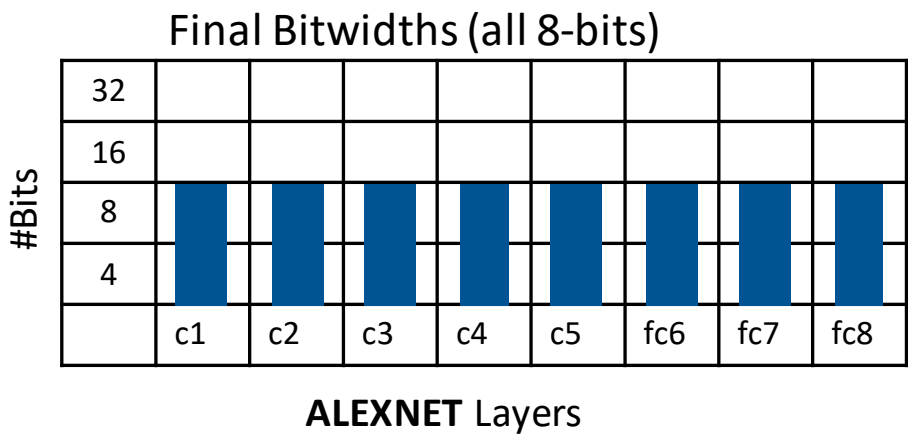
ReLeQ Development: Ideas ..

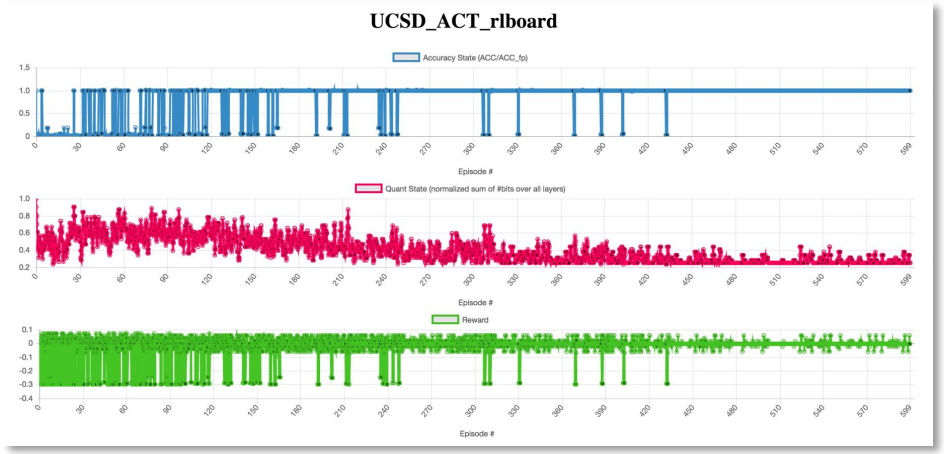
"Never calculate unless you already know the answer" by John Archibald Wheeler





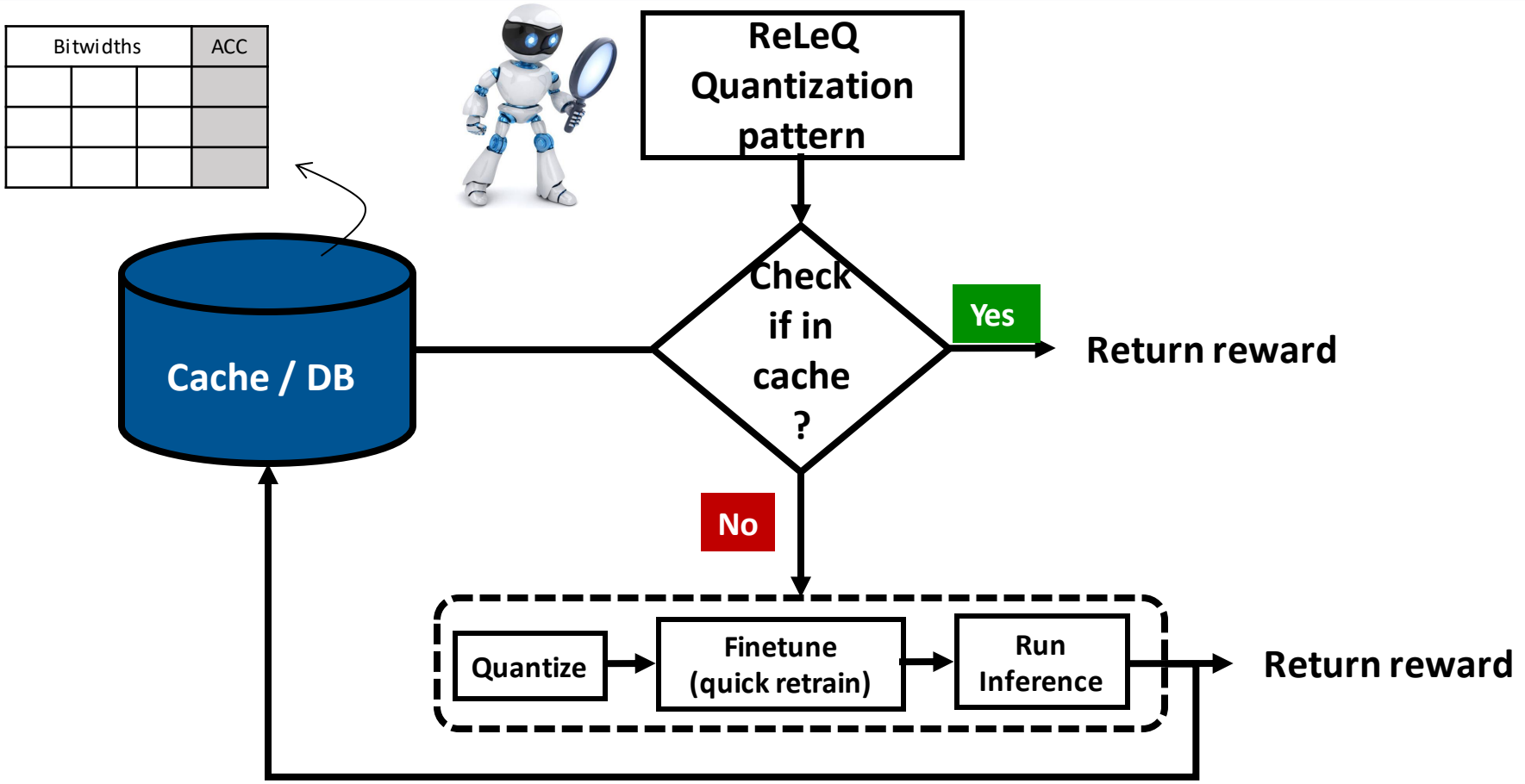






(2) Sanity Check

ReLeQ Development: Ideas ..

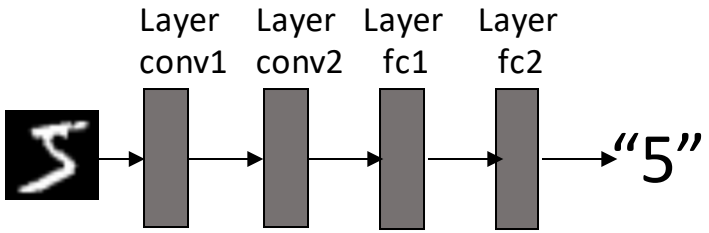


(3) Caching

Experimental Evaluation

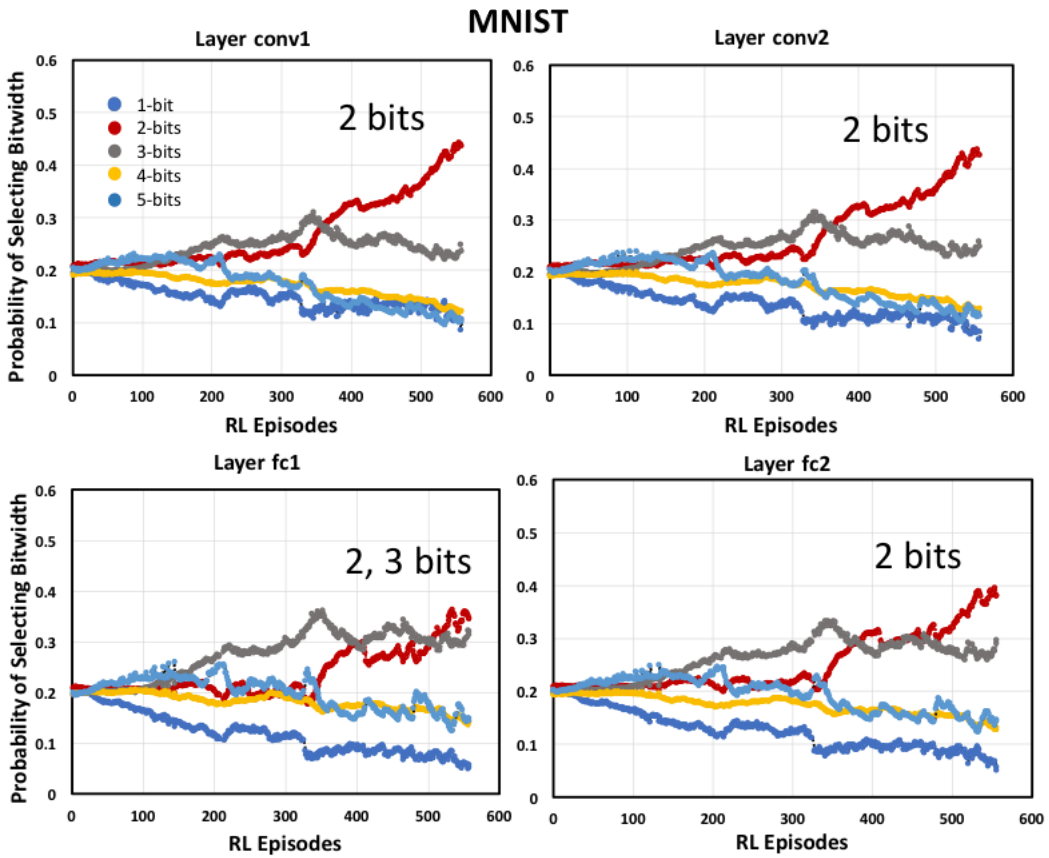
Evaluation: Quantization Policy

LeNet (MNIST Dataset)



ReLeQ learnt policy for LeNet layers quantization

Final suggested patterns:
(2,2,2,2)
(2,2,3,2)

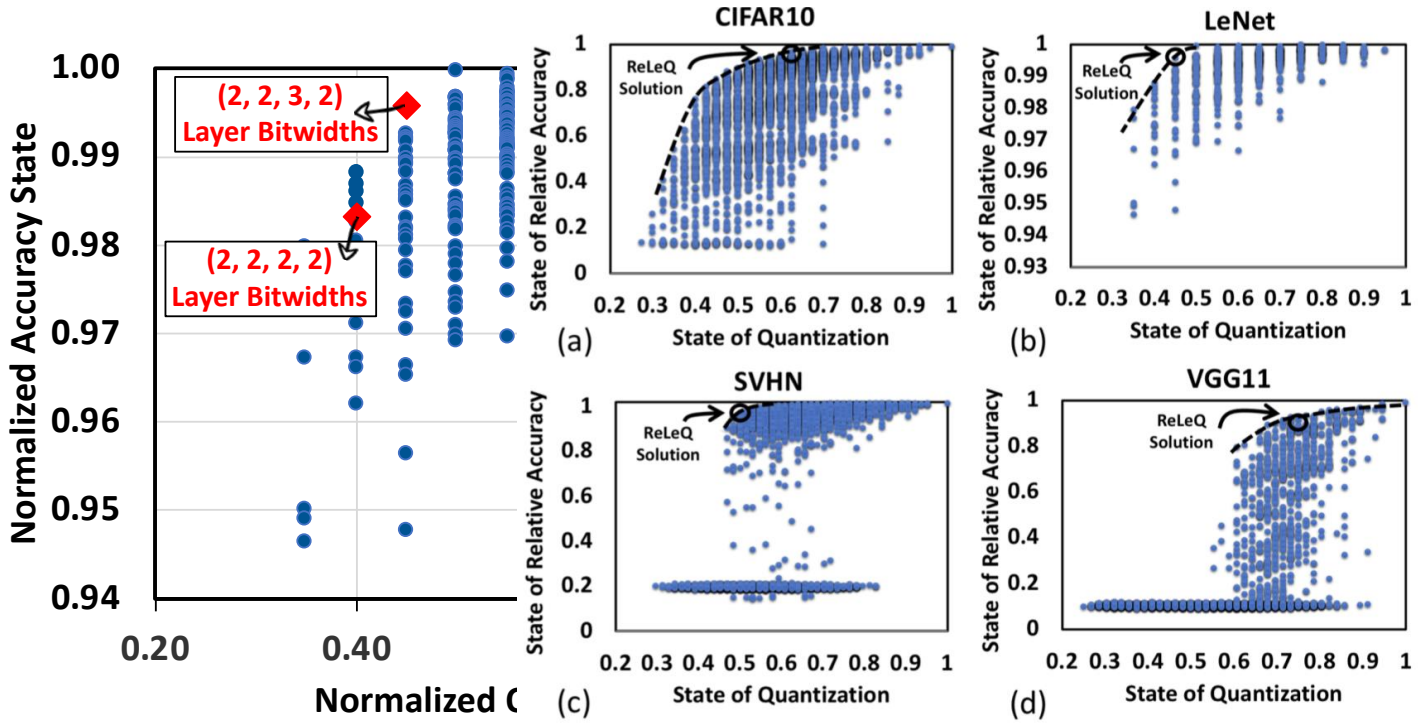


- ✓ **Bitwidth probability profiles are not uniform across DNN layers**
- ✓ **RL agent is able to understand the sensitivity of the network accuracy to the bitwidth of each layer**

ReLeQ vs. Exhaustive Search

New

Validation: Pareto Analysis



ReLeQ agent finds the Pareto optimal bitwidth assignment, striking a balance between accuracy and computation complexity

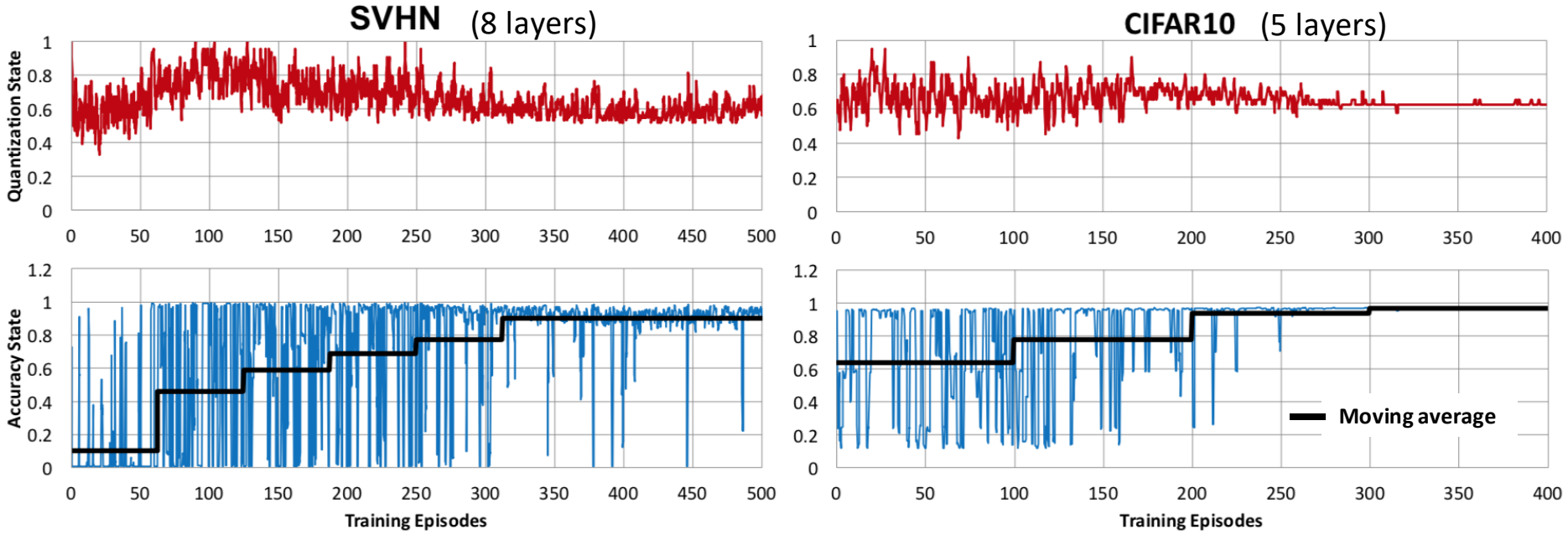
ReLeQ Evaluation Summary

Neural Network	Dataset	Number of Layers	Quantization Bitwidths	Avg. BW	Accuracy Loss (%)
AlexNet	ImageNet	8	{8,4,4,4,4,4,4,8}	5.00	0.0
CIFAR10-Net	CIFAR10	5	{5,5,5,5,5}	5.00	0.30
LeNet-5	MNIST	4	{2,2,3,2}	2.25	0.29
MobileNet	ImageNet	30	{8,5,6,6,4,4,7,8,4,6,8,5,5,8,6, 7,7,7,6,8, 6,8,8,6,7,5,5,7,8,8}	6.43	0.26
SVHN-Net	SVHN	8	{4,4,4,4,4,4,4,4}	4.00	0.0
Vgg-16	CIFAR10	16	{8,8,8,6,8,6,8,6,8,6,8,6,8,6,8,8}	7.25	0.1
ResNet-20	CIFAR10	20	{8,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,8}	4.40	0.0

< 0.3% Accuracy Loss

Evaluation

Learning and Convergence Analysis (1)

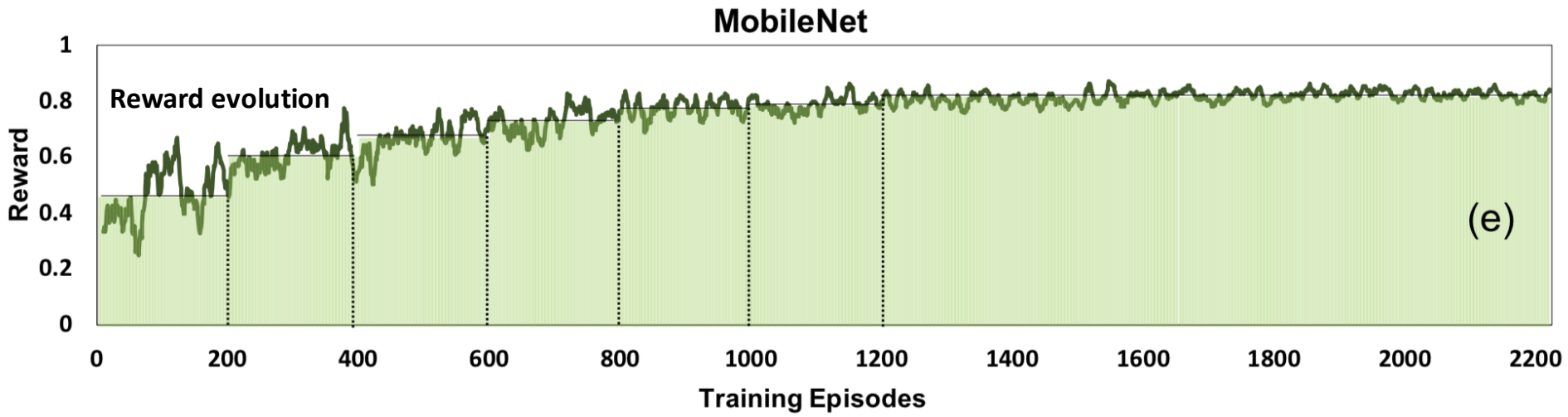


- ✓ Consistent Improvement in Solution Quality
- ✓ Convergence in around 300 episodes → efficiently explore huge search

Evaluation

New

Learning and Convergence Analysis (2)



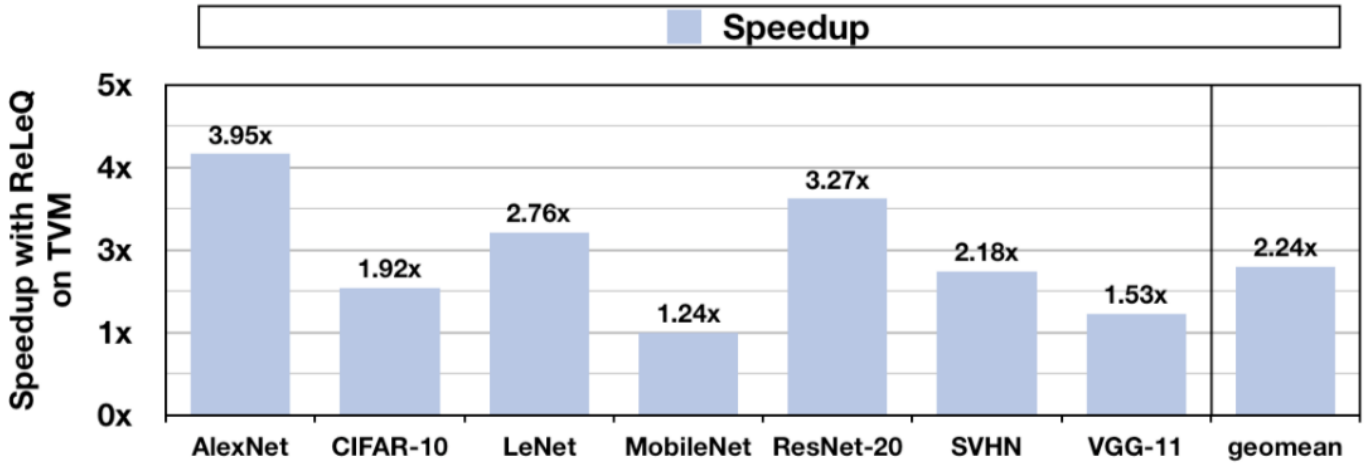
- ✓ Consistent Improvement in Solution Quality
- ✓ Convergence in around 300 episodes → efficiently explore huge search

Evaluation

New

Execution Time and Energy Benefits of ReLeQ with Conventional Hardware

Speedup with ReLeQ for conventional hardware using TVM over the baseline run using 8 bits



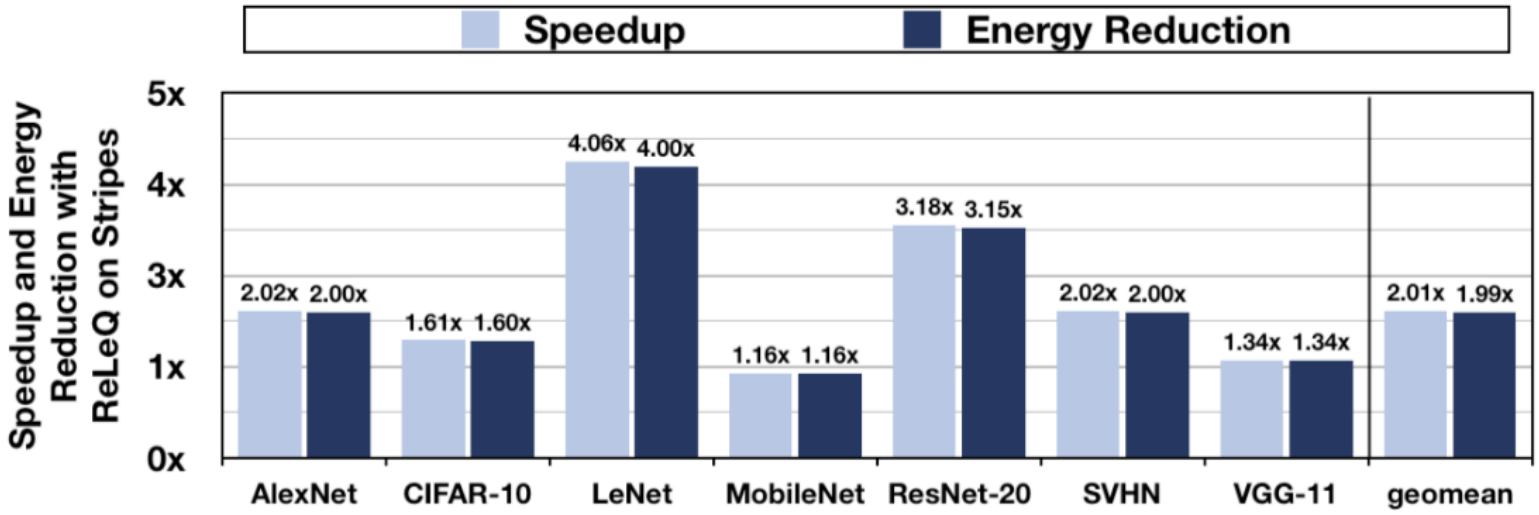
✓ **ReLeQ's solution offers, on average, 2.2× speedup over the baseline as the result of merely quantizing the weights that reduces the amount of computation and data transfer during inference**

Evaluation

New

Execution Time and Energy Benefits of ReLeQ with Custom Hardware Accelerator

Energy reduction and speedup with ReLeQ for **Stripes*** over the baseline execution when the accelerator is running 8-bit DNNs



*P. Judd, J., et al, "Stripes: Bit-serial deep neural network computing," MICRO 2016

✓ **ReLeQ's solutions yield, on average, 2.0× speedup and an additional 2.0× energy reduction**

ReLeQ provides a framework for **automatic** deep quantization of neural networks using reinforcement learning

Our results across multiple neural networks show that **ReLeQ** is effective in deep quantization of neural networks

< 0.3%
Accuracy Loss

~ 2.01x
Speedup

~ 1.99x
Energy Reduction

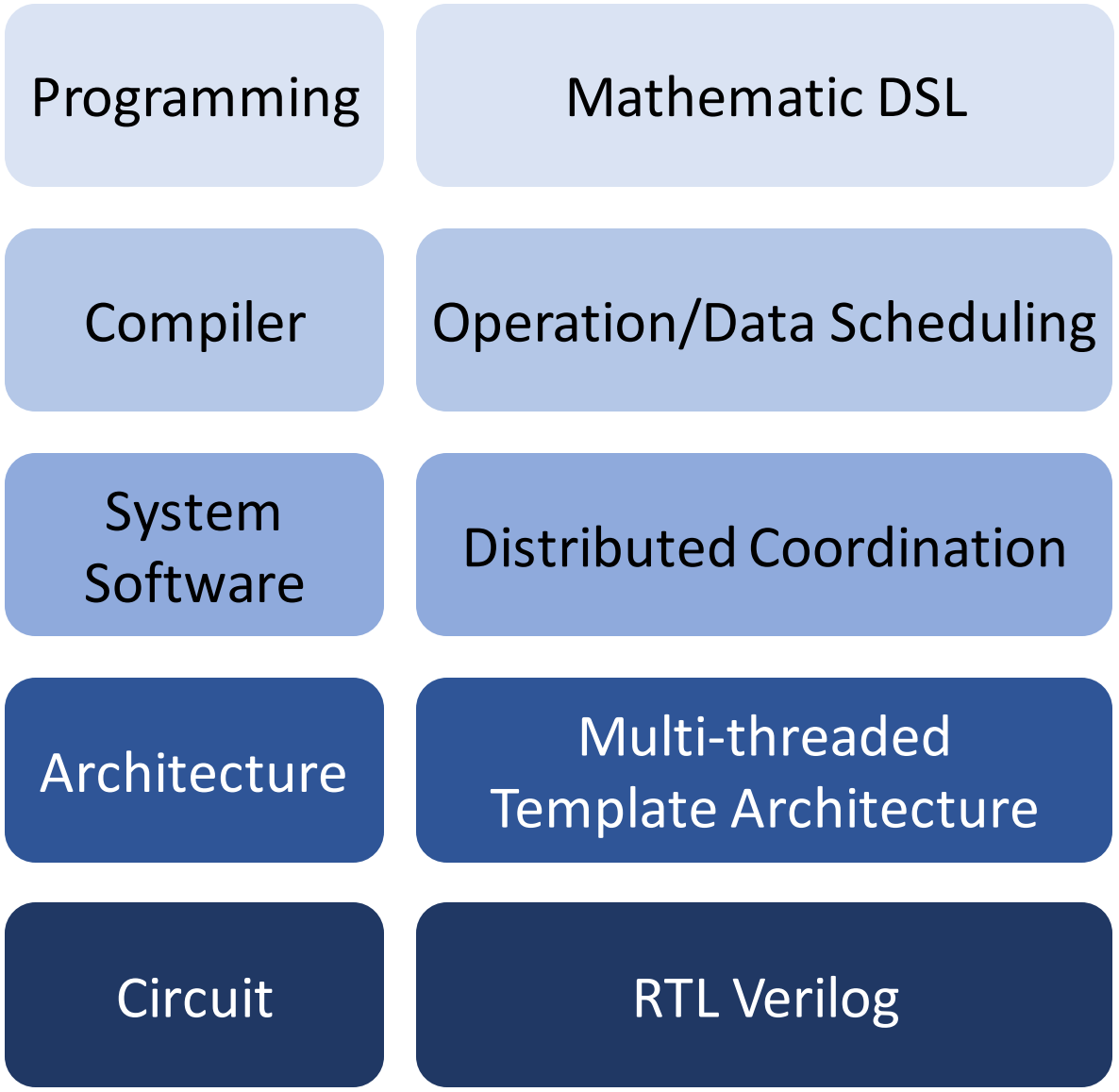
These encouraging results marks ReLeQ as the **initial step** towards automating the deep quantization of neural networks

ReLeQ: A Reinforcement Learning Approach for Deep Quantization of Neural Networks, ML for Systems Workshop, NeurIPS 2018

https://arxiv.org/pdf/1811.01704.pdf

- **NIH R01:**
  - **NeuroWeaver: A framework for developing translatable intelligent neural interface systems for precision neuromodulation therapies**
  - With Babak Mahmoudi and Joseph Manns (Emory University)
- **DARPA RTML**
  - **VeriGOOD-ML: Open-source Verilog Generator for RTML**
  - With Sachin Sapatnekar (University of Minnesota), Andrew Kahng (UCSD), Jie Gu (Nothwestern University)

Domain-Specific Stacks for Neuroscience and Machine Learning



The fifth day of creation …

By Maestro Mahmoud Farshchian