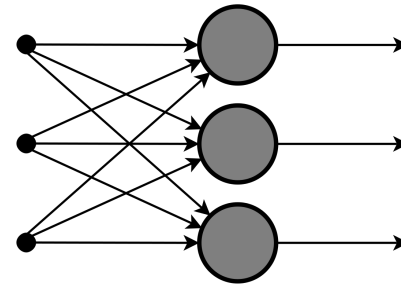
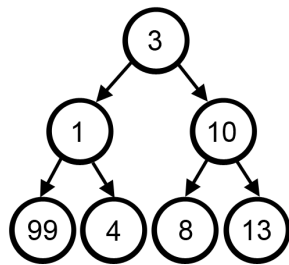


CSE 40171: Artificial Intelligence



Neural Network Model Search: Neural
Architecture Search (NAS)

Homework #4 has been released
It is due **tonight** at 11:59PM

Quiz #1 is scheduled for 10/30

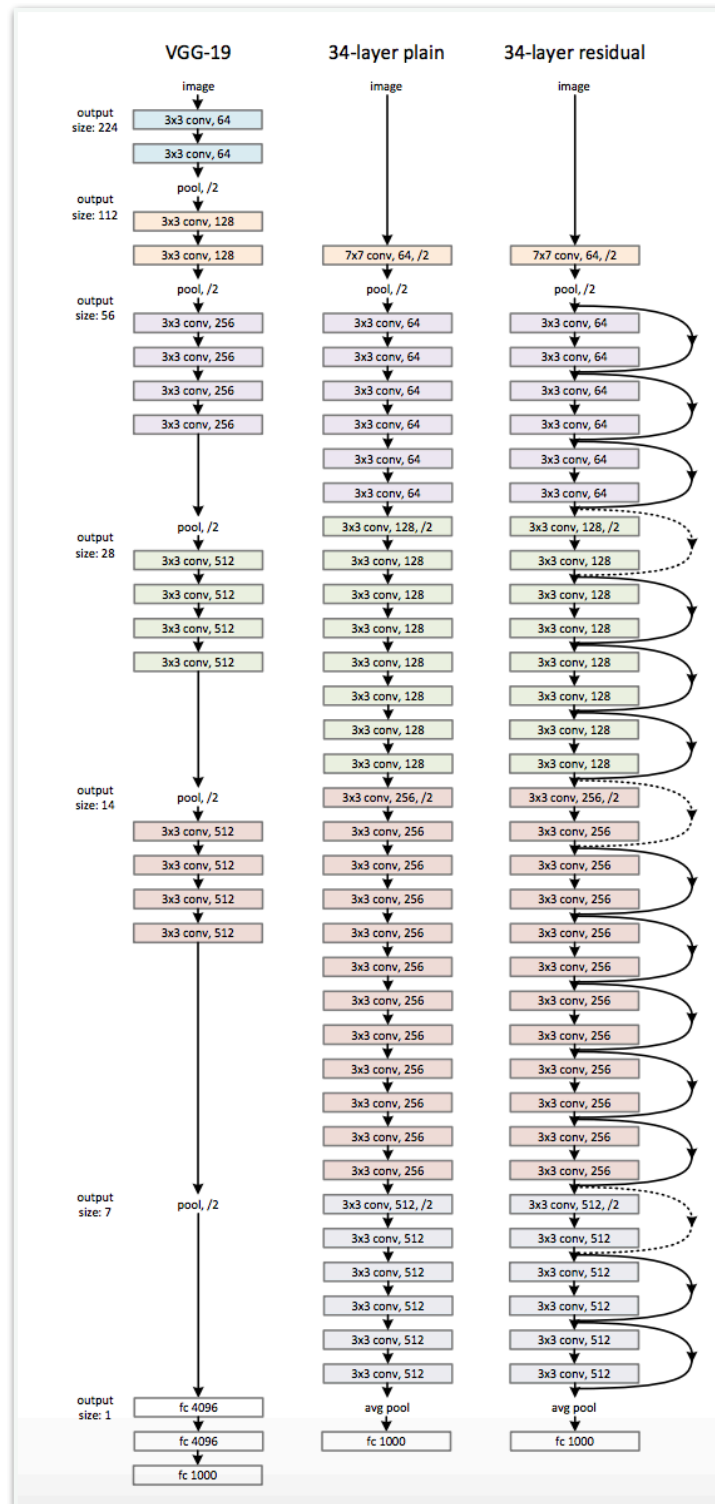
Project proposal instructions have been released. Proposals are Due 11/4.

(Let me know if you need a group)

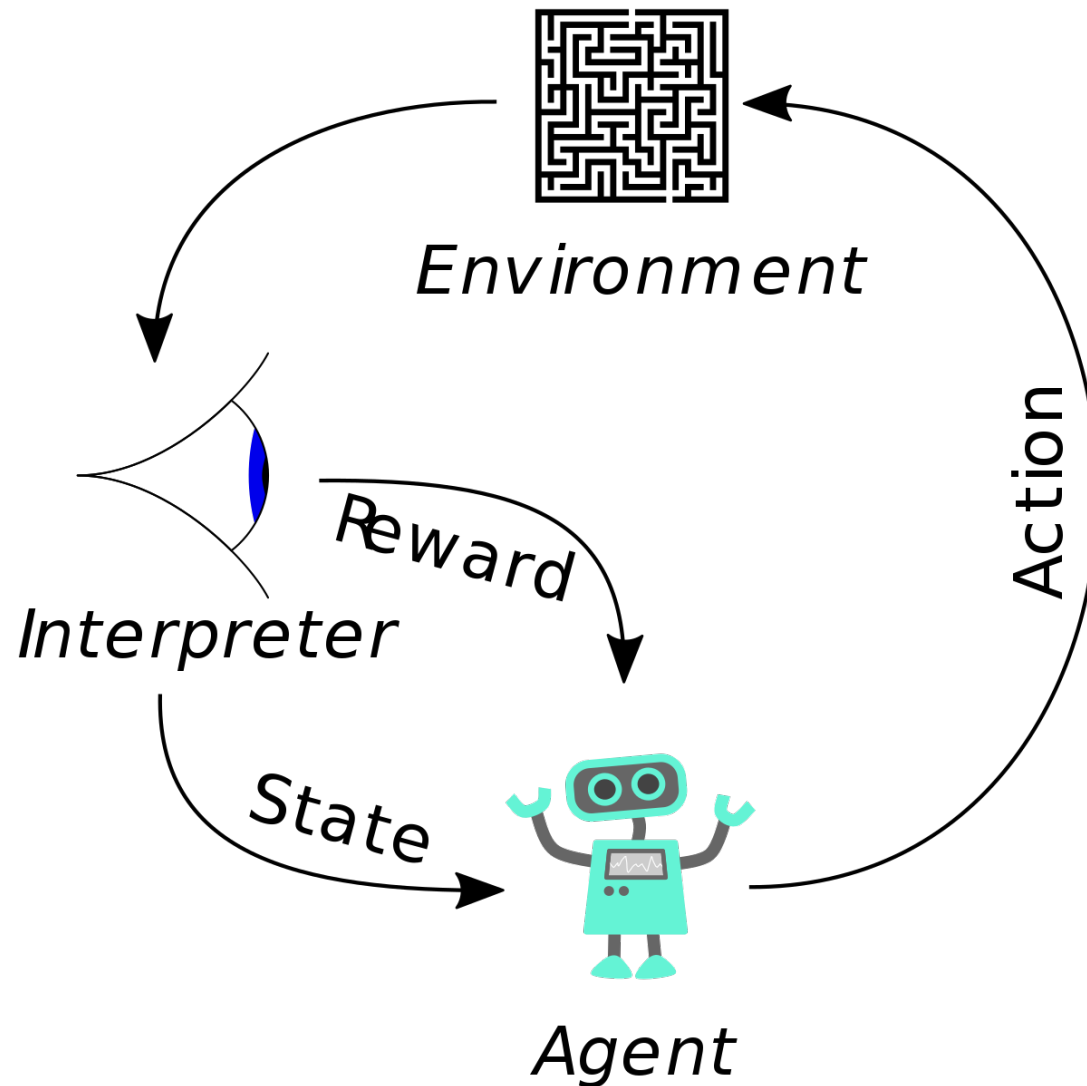
Let's take a broad view of the term
“hyperparameter”:

Definition: a hyperparameter can be *any*
free parameter of the learning system that
is not a weight

Q: What architectural components of a neural network can we automatically configure?

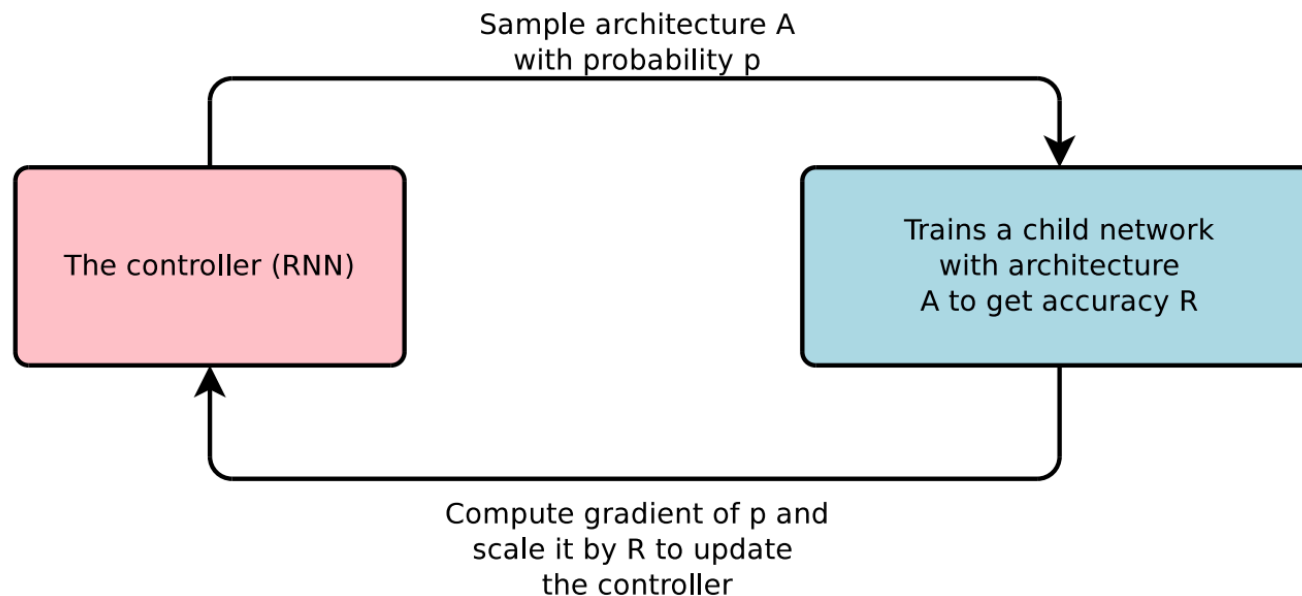


Reinforcement Learning for NAS



Gradient-based method for finding good archs.

Zoph and Le ICLR 2017



Structure and connectivity of a network can be specified by a variable length string

- ▶ It is possible to use a recurrent network to generate this string

The controller

The controller generates the architectural hyperparameters of neural networks

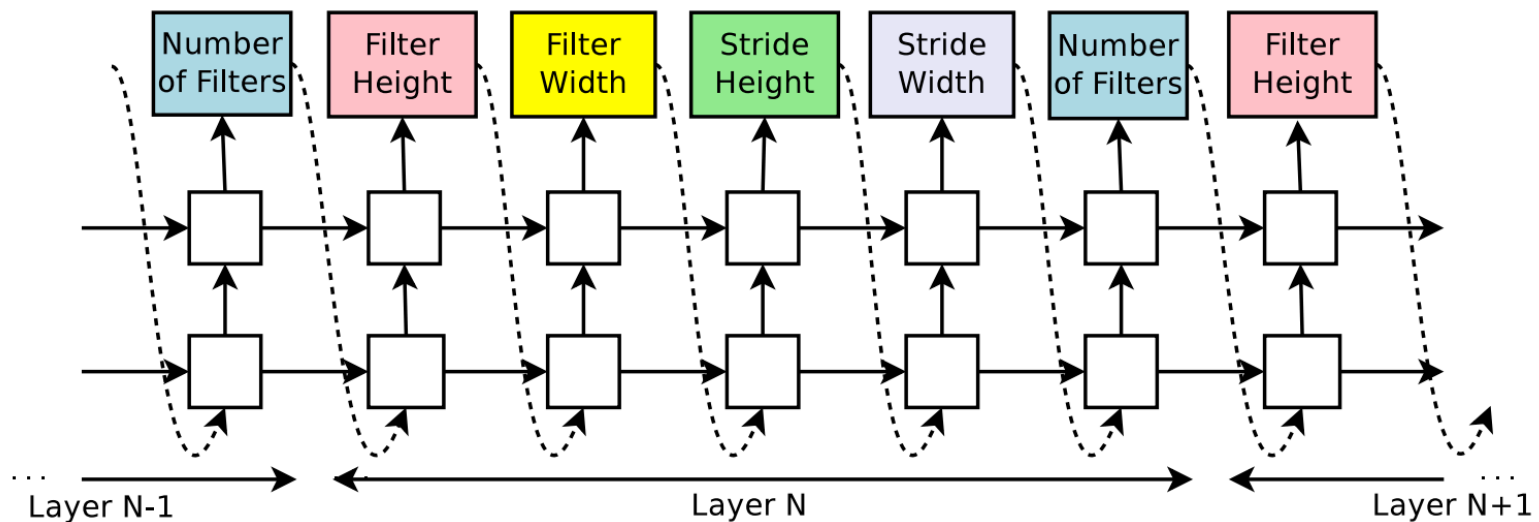


Image credit: Zoph and Le ICLR 2017

Every prediction is carried out by a softmax classifier and then fed into the next time step as input

REINFORCE

Update the classic REINFORCE rule (Williams Machine Learning 1992) for model search:

Number of Hyperparameters

Number of archs.

Hyperparameters

Actions

Baseline Function

Reward Signal

$$\frac{1}{m} \sum_{k=1}^m \sum_{t=1}^T \nabla_{\theta_c} \log P(a_t | a_{(t-1):1}; \theta_c) (R_k - b)$$

Distributed training for NAS

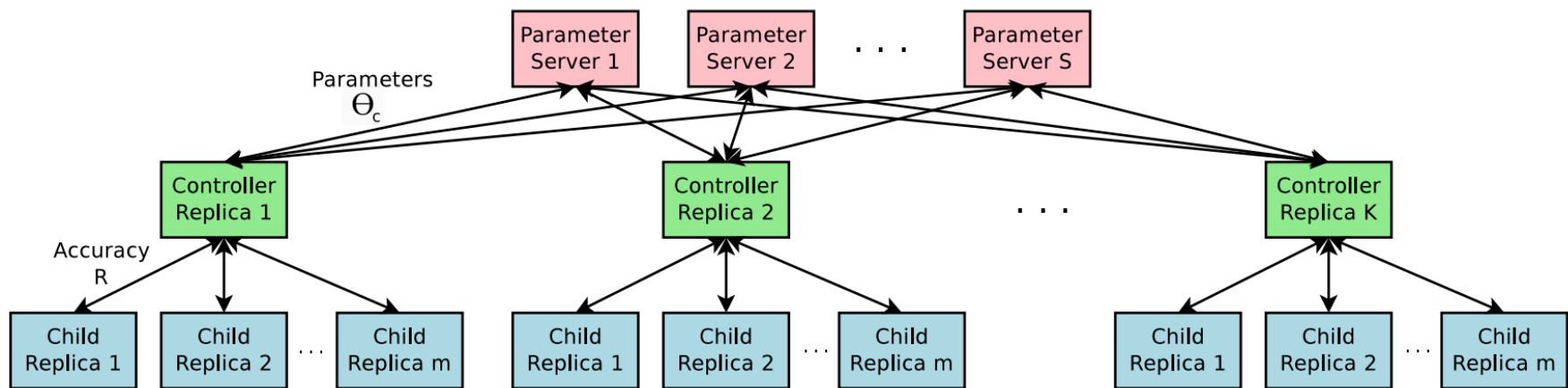


Image credit: Zoph and Le ICLR 2017

- S parameter servers to store and send parameters to K controller replicas
- Each controller replica then samples m architectures and run the multiple child models in parallel
- The accuracy of each child model is recorded to compute the gradients with respect to θ_c , which are then sent back to the parameter servers

Variation: add skip connections and other layer types

Purpose: widen the search space

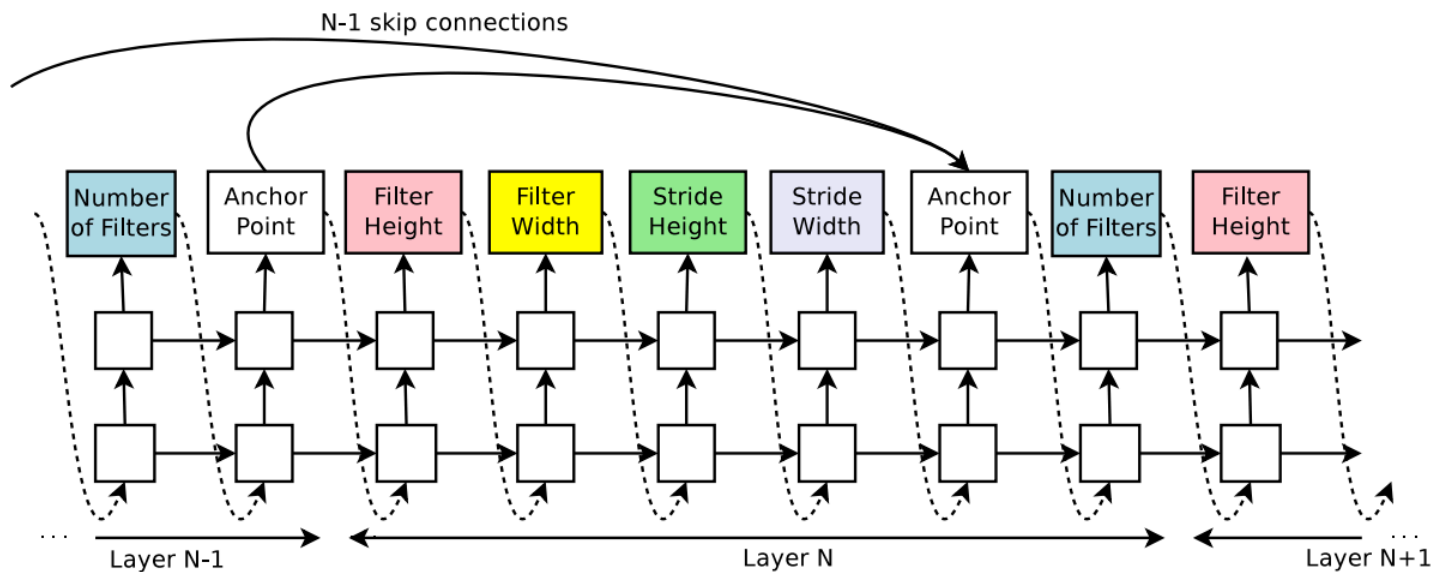


Image credit: Zoph and Le ICLR 2017

Generation of recurrent cell architectures

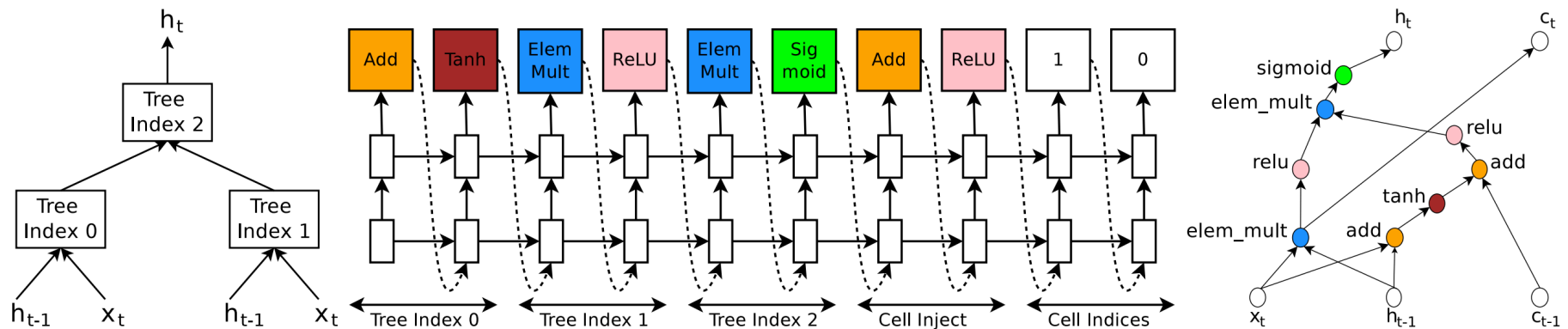


Image credit: Zoph and Le ICLR 2017

Left: tree that defines the computation steps to be predicted by controller

Center: example set of predictions made by the controller for each computation step in the tree.

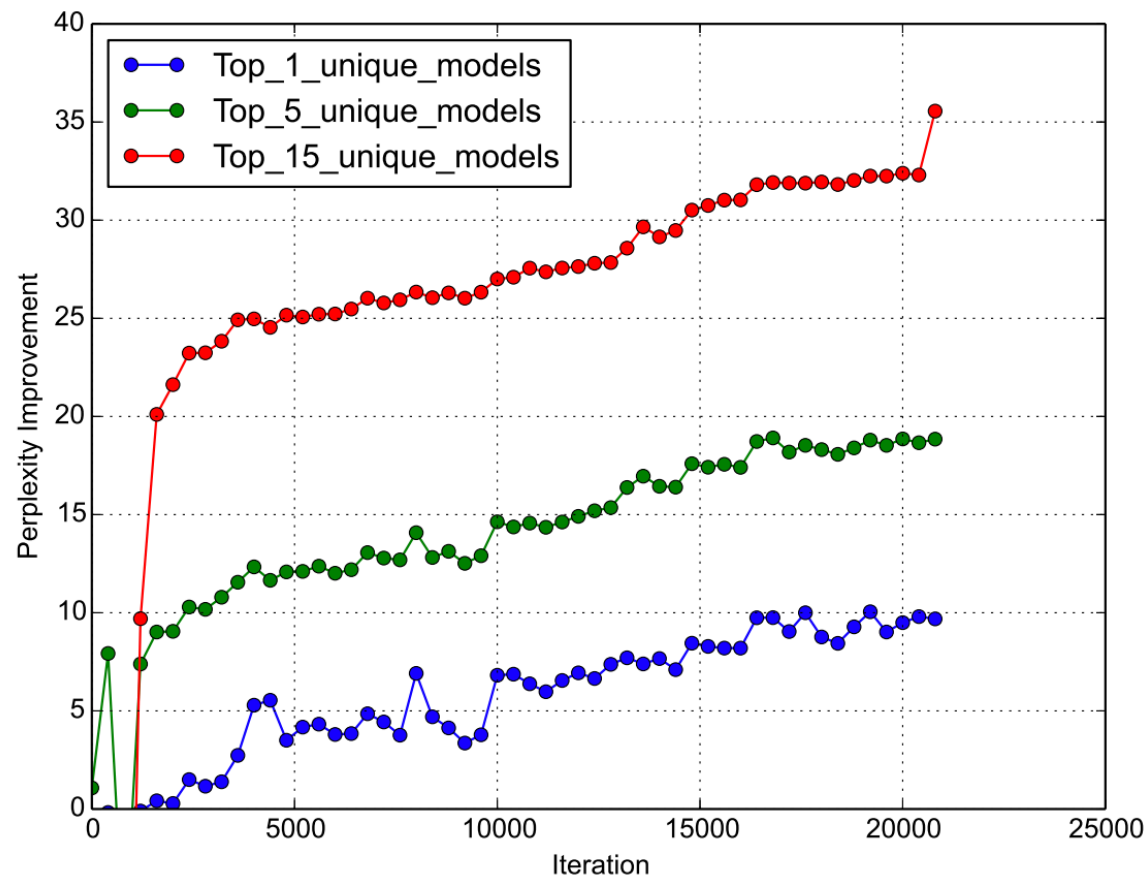
Right: the computation graph of the recurrent cell constructed from example predictions of the controller

Performance of reinforcement learning search and other state-of-the-art models on CIFAR-10

Model	Depth	Parameters	Error rate (%)
Network in Network (Lin et al., 2013)	-	-	8.81
All-CNN (Springenberg et al., 2014)	-	-	7.25
Deeply Supervised Net (Lee et al., 2015)	-	-	7.97
Highway Network (Srivastava et al., 2015)	-	-	7.72
Scalable Bayesian Optimization (Snoek et al., 2015)	-	-	6.37
FractalNet (Larsson et al., 2016)	21	38.6M	5.22
with Dropout/Drop-path	21	38.6M	4.60
ResNet (He et al., 2016a)	110	1.7M	6.61
ResNet (reported by Huang et al. (2016c))	110	1.7M	6.41
ResNet with Stochastic Depth (Huang et al., 2016c)	110	1.7M	5.23
	1202	10.2M	4.91
Wide ResNet (Zagoruyko & Komodakis, 2016)	16	11.0M	4.81
	28	36.5M	4.17
ResNet (pre-activation) (He et al., 2016b)	164	1.7M	5.46
	1001	10.2M	4.62
DenseNet ($L = 40, k = 12$) Huang et al. (2016a)	40	1.0M	5.24
DenseNet ($L = 100, k = 12$) Huang et al. (2016a)	100	7.0M	4.10
DenseNet ($L = 100, k = 24$) Huang et al. (2016a)	100	27.2M	3.74
DenseNet-BC ($L = 100, k = 40$) Huang et al. (2016b)	190	25.6M	3.46
Neural Architecture Search v1 no stride or pooling	15	4.2M	5.50
Neural Architecture Search v2 predicting strides	20	2.5M	6.01
Neural Architecture Search v3 max pooling	39	7.1M	4.47
Neural Architecture Search v3 max pooling + more filters	39	37.4M	3.65

*

Reinforcement learning search vs. random search



Difference between the average of the top k models the controller finds vs. random search every 400 models run

Evolutionary strategies for NAS

Real et al. AAI 2019

Several attempts at this, but none very good until recently

Modify the tournament selection evolutionary algorithm by introducing an age property to favor the younger genotypes

Basic tournament selection algorithm:

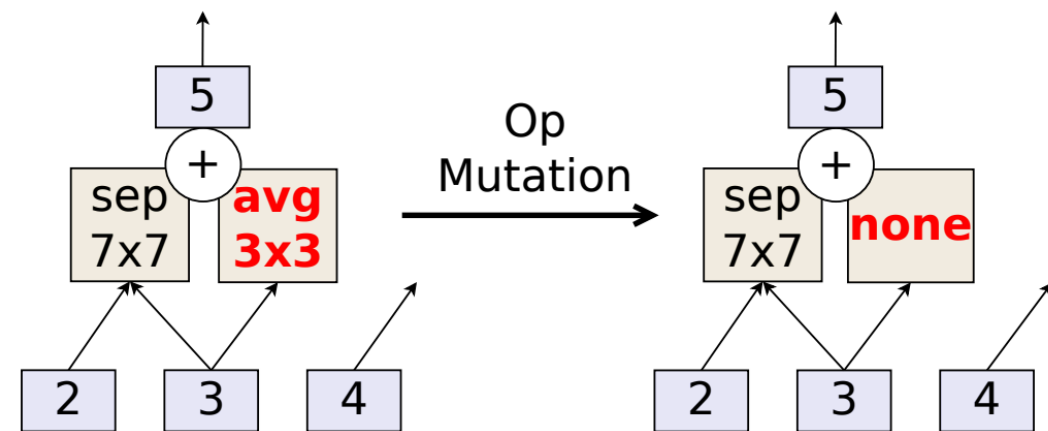
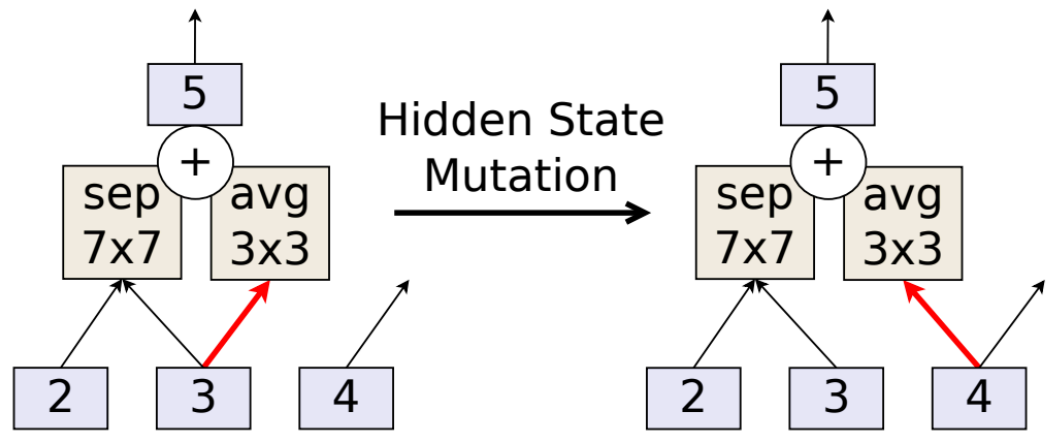
```
choose k (the tournament size) individuals from the population at random
choose the best individual from the tournament with probability p
choose the second best individual with probability p*(1-p)
choose the third best individual with probability p*((1-p)^2)
and so on
```

Aging Evolution

Algorithm 1 Aging Evolution

```
population  $\leftarrow$  empty queue  $\triangleright$  The population.  
history  $\leftarrow$   $\emptyset$   $\triangleright$  Will contain all models.  
while  $|population| < P$  do  $\triangleright$  Initialize population.  
    model.arch  $\leftarrow$  RANDOMARCHITECTURE()  
    model.accuracy  $\leftarrow$  TRAINANDEVAL(model.arch)  
    add model to right of population  
    add model to history  
end while  
while  $|history| < C$  do  $\triangleright$  Evolve for  $C$  cycles.  
    sample  $\leftarrow$   $\emptyset$   $\triangleright$  Parent candidates.  
    while  $|sample| < S$  do  
        candidate  $\leftarrow$  random element from population  
         $\triangleright$  The element stays in the population.  
        add candidate to sample  
    end while  
    parent  $\leftarrow$  highest-accuracy model in sample  
    child.arch  $\leftarrow$  MUTATE(parent.arch)  
    child.accuracy  $\leftarrow$  TRAINANDEVAL(child.arch)  
    add child to right of population  
    add child to history  
    remove dead from left of population  $\triangleright$  Oldest.  
    discard dead  
end while  
return highest-accuracy model in history
```

Two mutation types



Does adding aging to tournament selection have any effect?

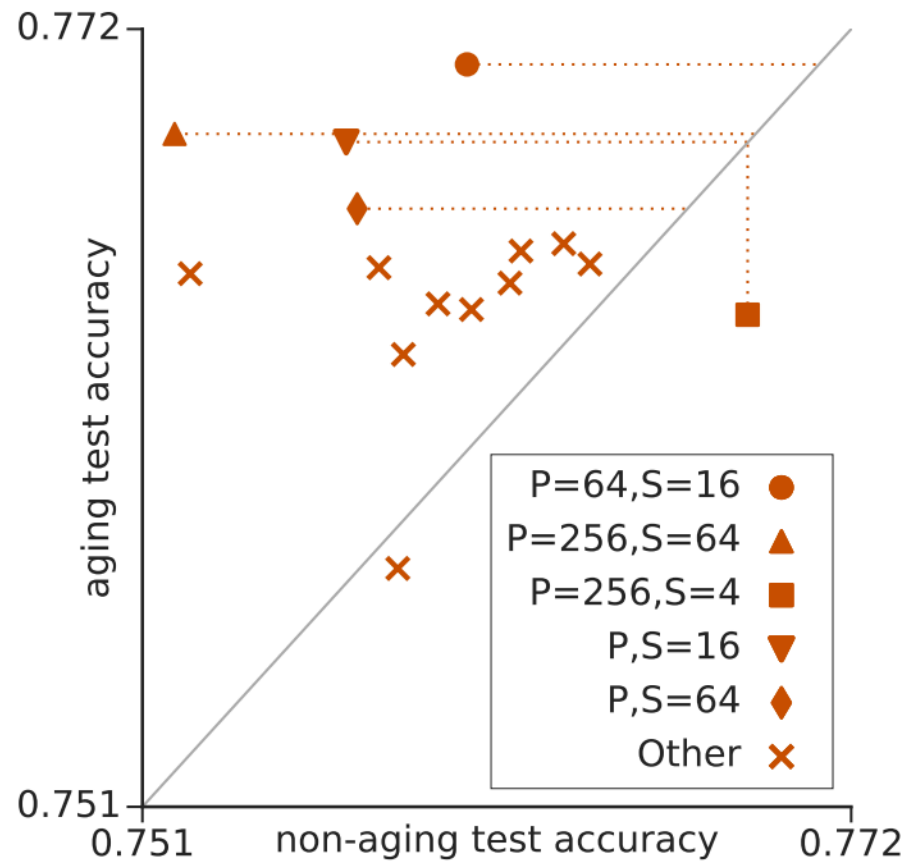


Image credit: Real et al. AAAI 2019

Results for CIFAR-10

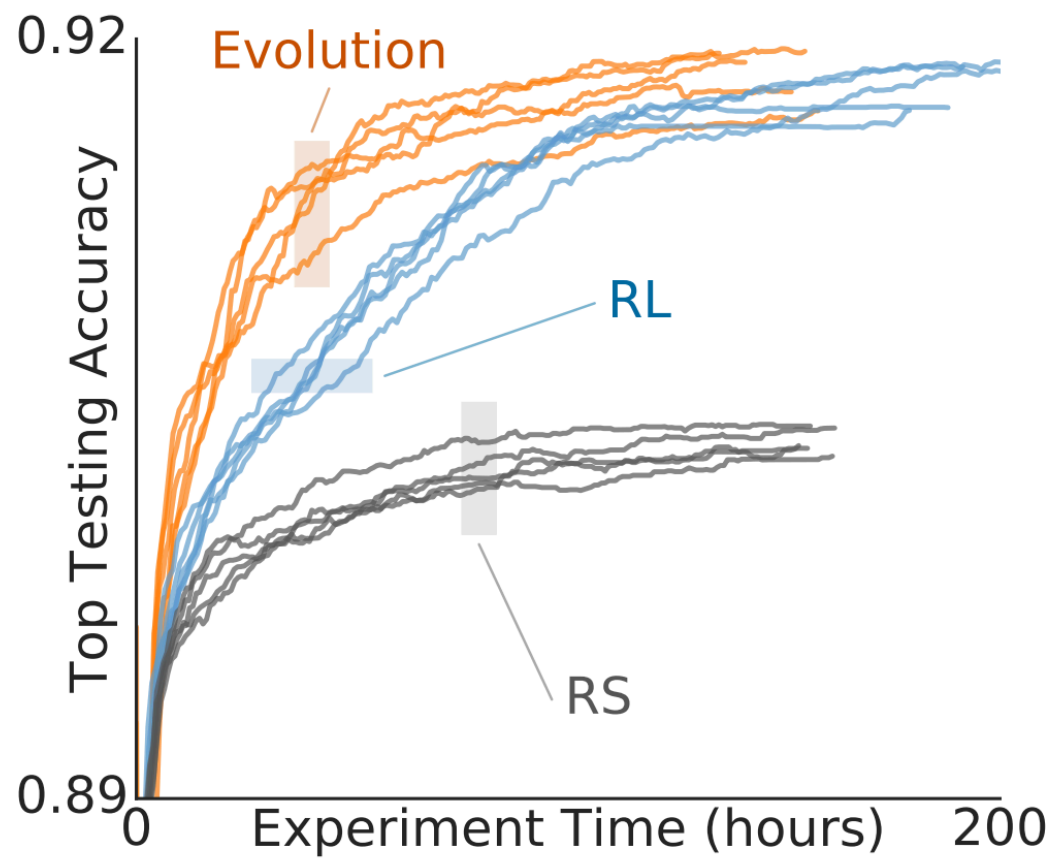
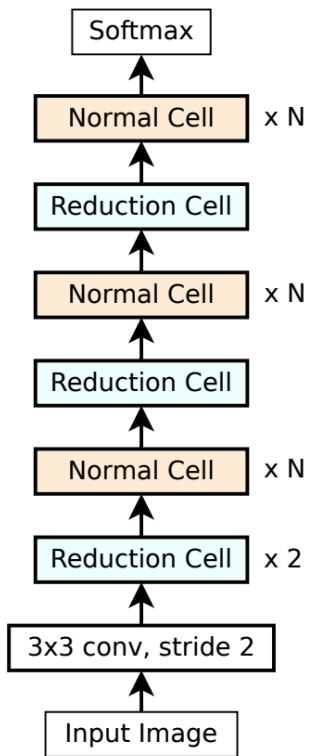
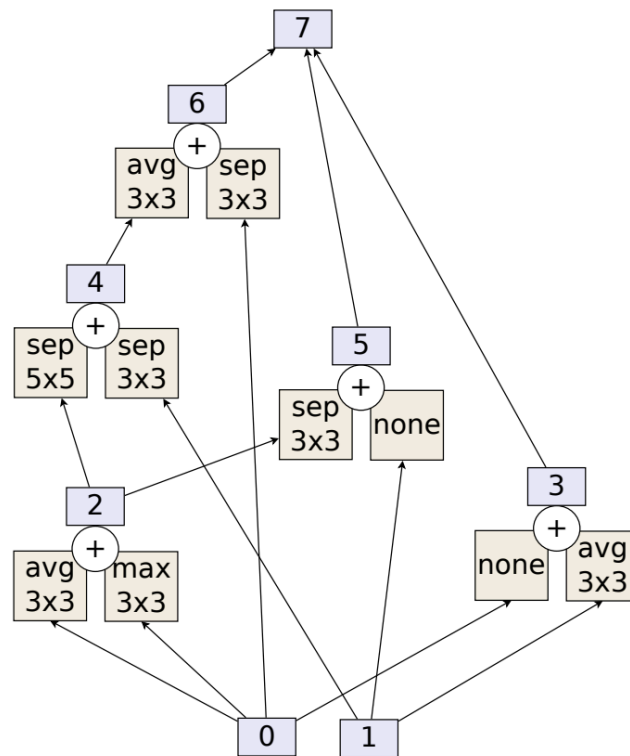


Image credit: Real et al. AAAI 2019

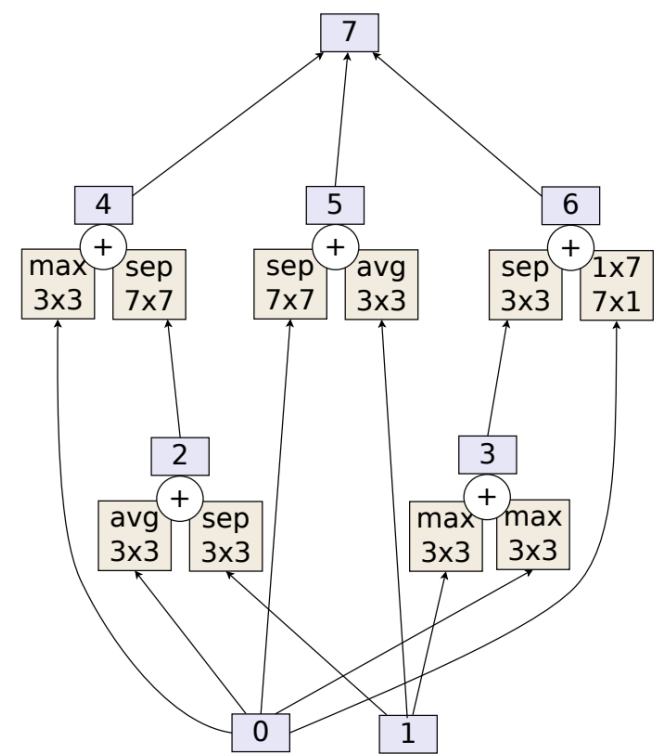
AmoebaNet-A Evolved Architecture



Overall Model



AmoebaNet-A Normal Cell



Reduction Cell

Results on ImageNet



Model	# Parameters	# Multiply-Adds	Top-1 / Top-5 Accuracy (%)
Incep-ResNet V2	55.8M	13.2B	80.4 / 95.3
ResNeXt-101	83.6M	31.5B	80.9 / 95.6
PolyNet	92.0M	34.7B	81.3 / 95.8
Dual-Path-Net-131	79.5M	32.0B	81.5 / 95.8
GeNet-2 *	156M	–	72.1 / 90.4
Block-QNN-B *	–	–	75.7 / 92.6
Hierarchical *	64M	–	79.7 / 94.8
NASNet-A	88.9M	23.8B	82.7 / 96.2
PNASNet-5	86.1M	25.0B	82.9 / 96.2
AmoebaNet-A (N=6, F=190) *	86.7M	23.1B	82.8 / 96.1
AmoebaNet-A (N=6, F=448) *	469M	104B	83.9 / 96.6

Hill-Climbing for NAS

Elskin et al. ICLR Workshop Track 2018

Simple iterative approach that, at each step:

- ▶ applies a set of alternative network morphisms to the current network
- ▶ trains the resulting child networks with short optimization runs of cosine annealing (Loshchilov & Hutter, 2017),
- ▶ moves to the most promising child network.

Network morphisms are mappings between different architectures

Visualization of Hill Climbing Approach

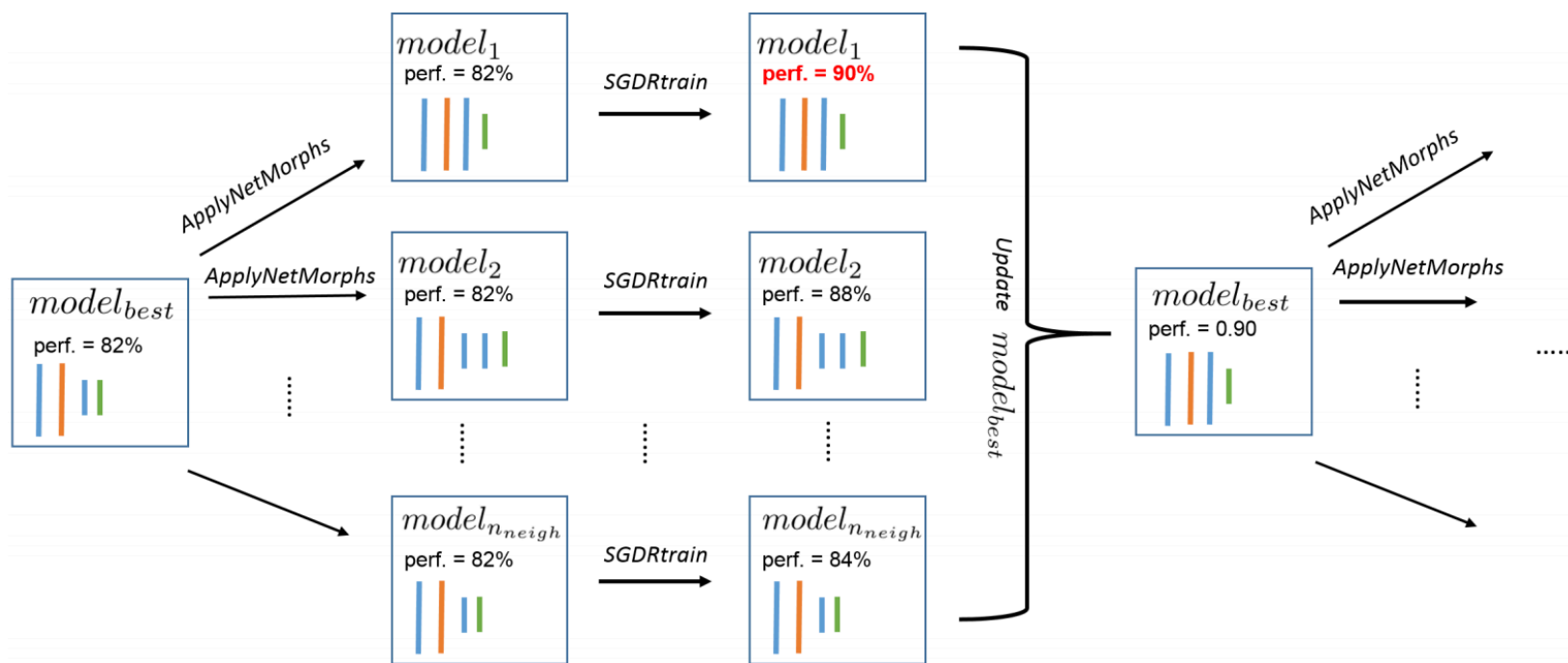


Image credit: Elskin et al. ICLR Workshop Track 2018

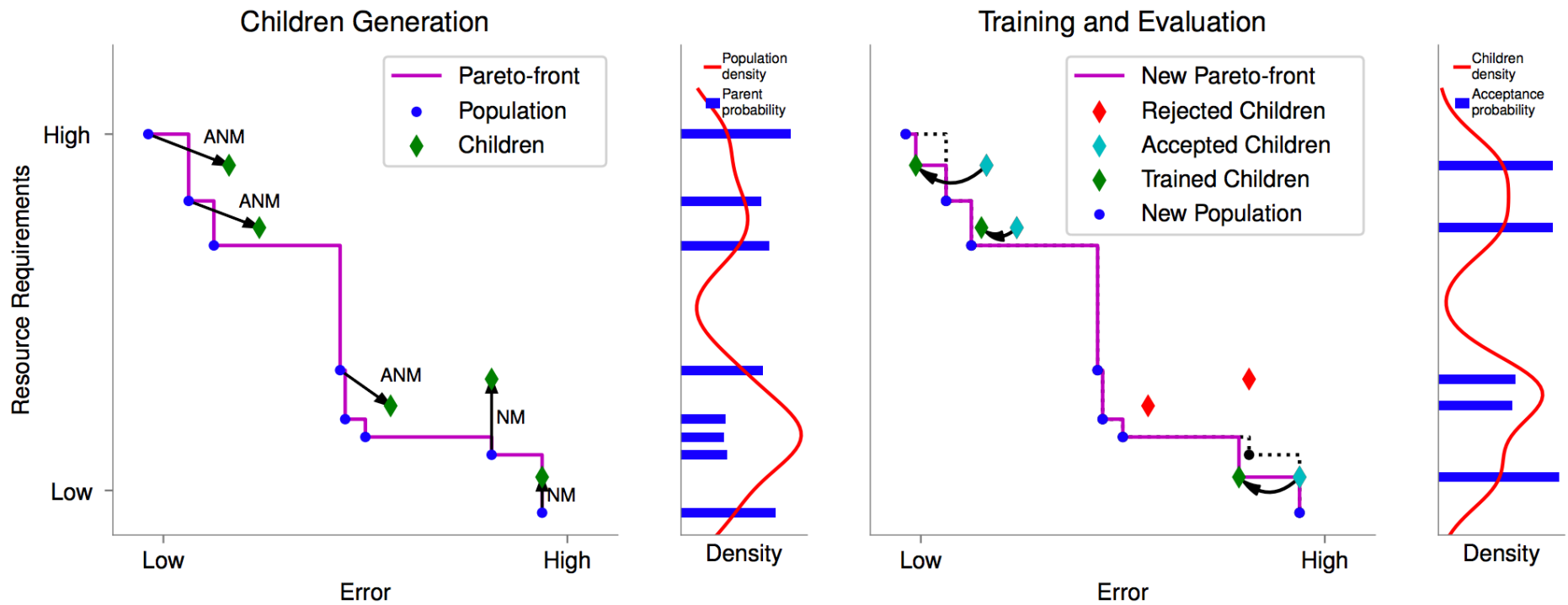
Results on CIFAR-10

model	resources spent	# params (mil.)	error (%)
* Shake-Shake (Gastaldi, 2017)	2 days, 2 GPUs	26	2.9
WRN 28-10 (Loshchilov & Hutter, 2017)	1 day, 1 GPU	36.5	3.86
Baker et al. (2016)	8-10 days, 10 GPUs	11	6.9
Cai et al. (2017)	3 days, 5 GPUs	19.7	5.7
Zoph & Le (2017)	800 GPUs, ? days	37.5	3.65
Real et al. (2017)	250 GPUs, 10 days	5.4	5.4
Saxena & Verbeek (2016)	?	21	7.4
Brock et al. (2017)	3 days, 1 GPU	16.0	4.0
Ours (random networks, $n_{steps} = 5, n_{neigh} = 1$)	4.5 hours	4.4	6.5
Ours ($n_{steps} = 5, n_{neigh} = 8, 10$ runs)	0.5 days, 1 GPU	5.7	5.7
Ours ($n_{steps} = 8, n_{neigh} = 8, 4$ runs)	1 day, 1 GPU	19.7	5.2
Ours (snapshot ensemble, 4 runs)	2 days, 1 GPU	57.8	4.7
Ours (ensemble across runs)	1 day, 4 GPUs	88	4.4

Table credit: Elsken et al. ICLR Workshop Track 2018

Multi-Objective Search for NAS

LEMONADE: Elskin et al. ICLR 2019



Algorithm 1 LEMONADE

- 1: input: $\mathcal{P}_0, f, n_{gen}, n_{pc}, n_{ac}$
- 2: $\mathcal{P} \leftarrow \mathcal{P}_0$
- 3: **for** $i \leftarrow 1, \dots, n_{gen}$ **do**
- 4: $p_{KDE} \leftarrow KDE(\{f_{cheap}(N) | N \in \mathcal{P}\})$
- 5: Compute parent distribution $p_{\mathcal{P}}$ (Eq. 1)
- 6: $\mathbf{N}_{pc}^c \leftarrow GenerateChildren(\mathcal{P}, p_{\mathcal{P}}, n_{pc})$
- 7: Compute children distribution p_{child} (Eq. 2)
- 8: $\mathbf{N}_{ac}^c \leftarrow AcceptSubSet(\mathbf{N}_{pc}^c, p_{child}, n_{ac})$
- 9: Evaluate f_{exp} for $N^c \in \mathbf{N}_{ac}^c$
- 10: $\mathcal{P} \leftarrow ParetoFront(\mathcal{P} \cup \mathbf{N}_{ac}^c, f)$
- 11: **end for**
- 12: **return** \mathcal{P}

LEMONADE Comparison Experiments

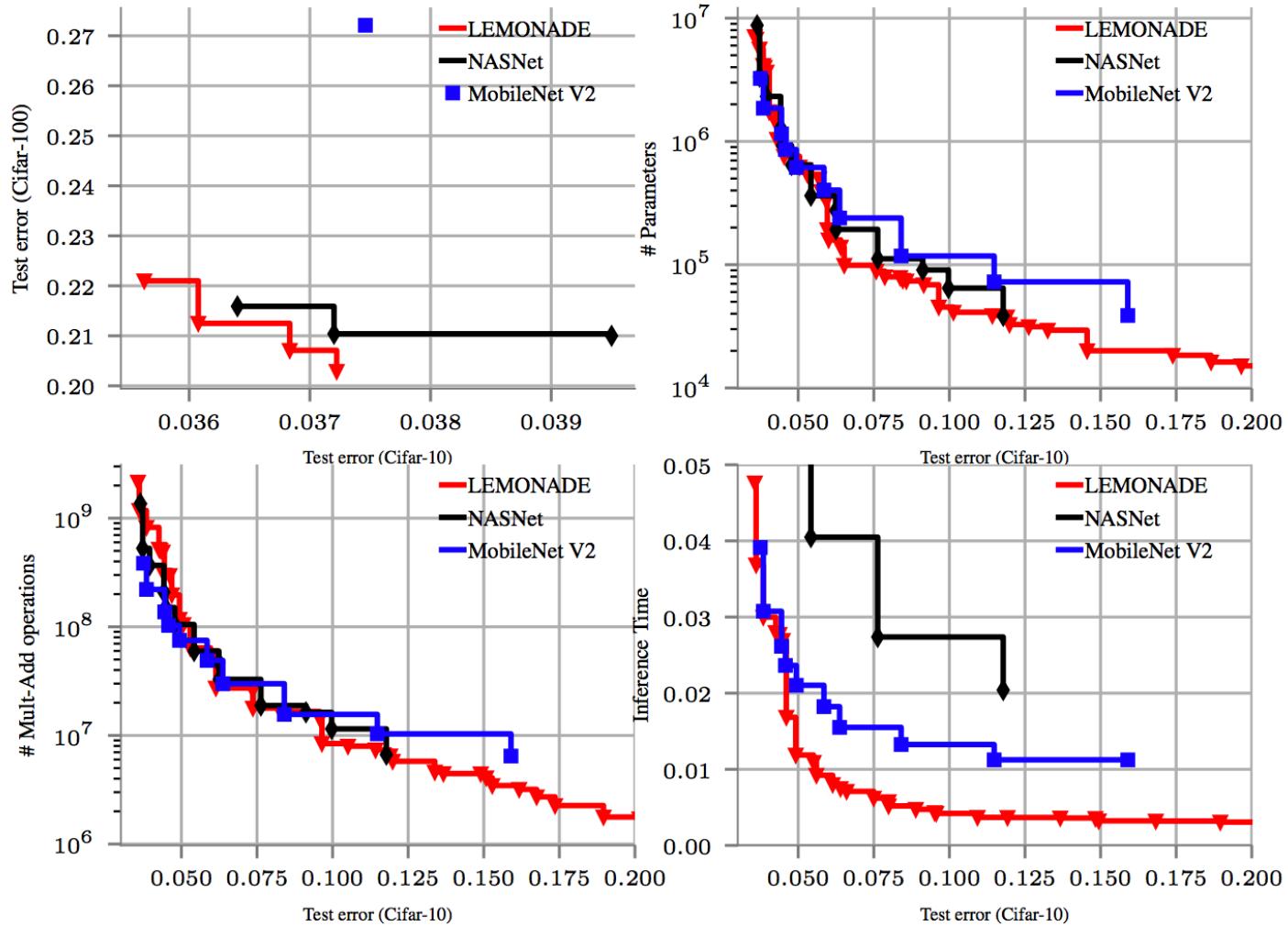
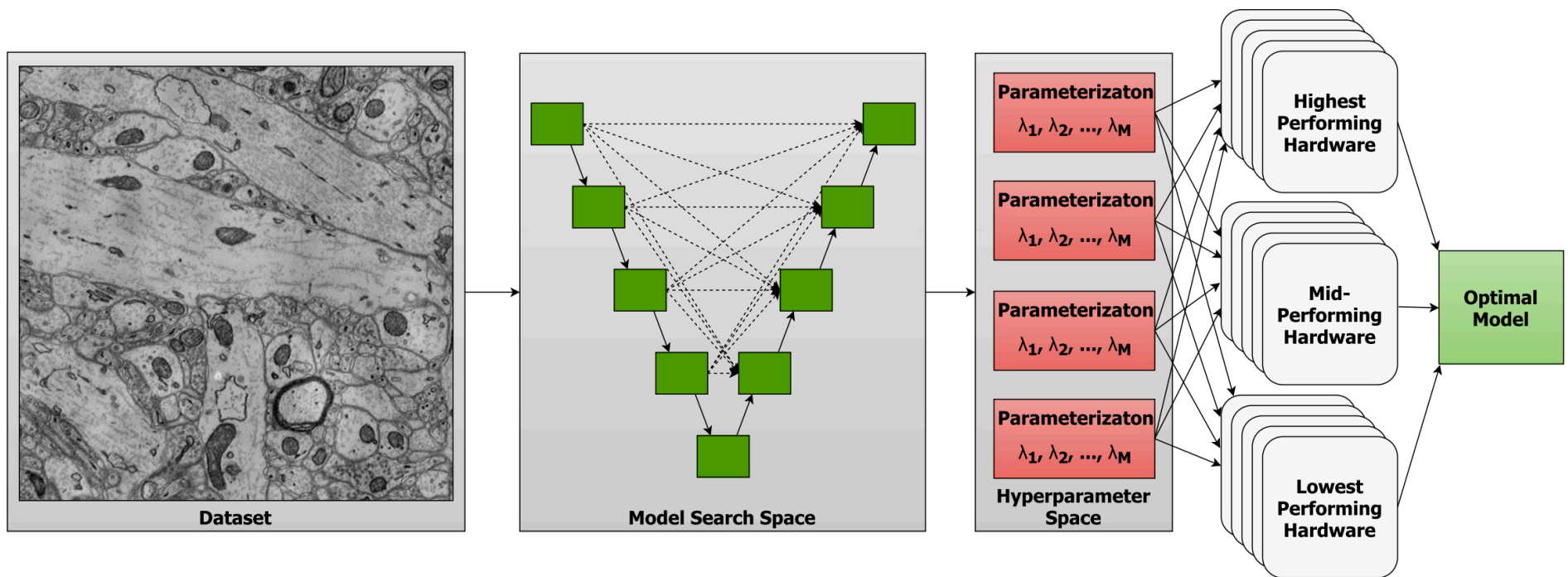


Image credit: Elskin et al. ICLR 2019

Systems Perspective: Scalable Hardware-Aware Distributed Hyperparameter Optimization

SHADHO: Kinnison et al. WACV 2018



Heuristics for hyperparameter optimization

Complexity:

Determined by the aggregate size of a model's viable hyperparameter domains

[a, b] is the closed interval containing 99% of the probability distribution governing s

$$C(s_i) = \begin{cases} 2 + \|b - a\| & \text{if } s \text{ is continuous} \\ 2 - \frac{1}{|s_i|} & \text{if } s \text{ is discrete} \end{cases}$$

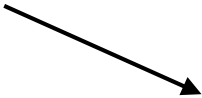
Hyperparameter Domain

Heuristics for hyperparameter optimization

Priority:

- ▶ accounts for model performance across different parametrizations
- ▶ estimate the fitness of a model to learn from the data as a function of variation in performance

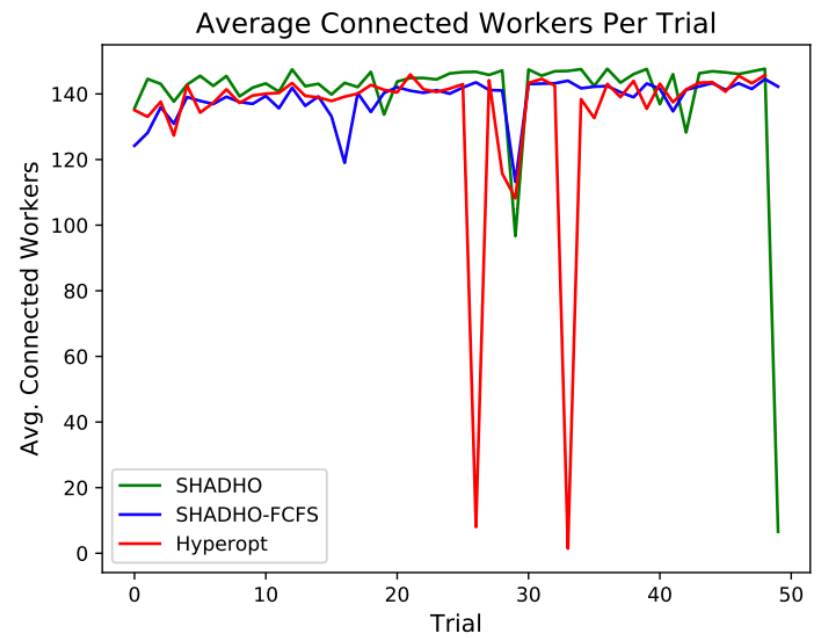
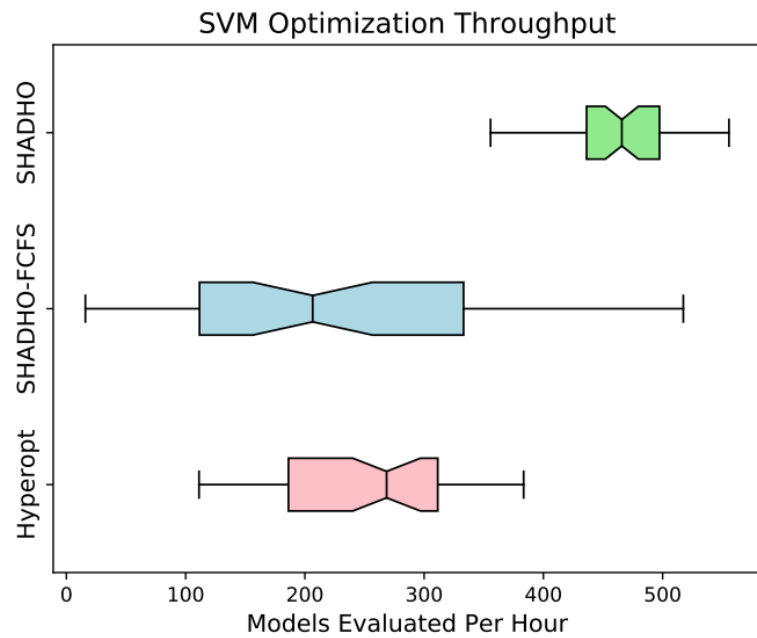
Estimate of the intrinsic fitness of a model for the learning task


$$P(L) = \min(L)^{-1} - \max(L)^{-1}$$



Length Scales

SHADHO vs. Hyperopt

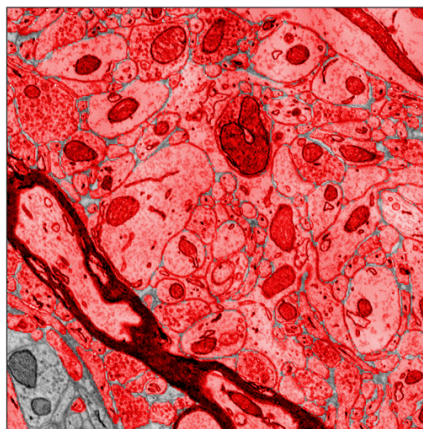


SHADHO for NAS

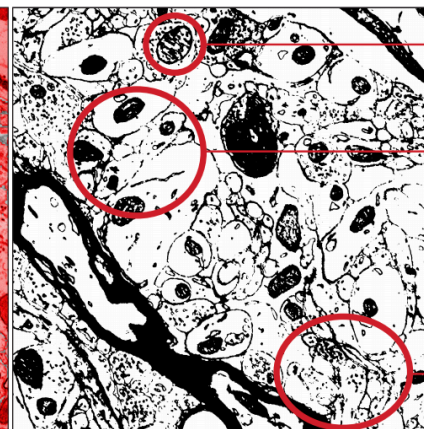
U-Net Parameters

Parameter	Values
Min. Kernels	16, 32, 64, 128
Kernel Size	1, 3, 5, 7, 9
Activations	sigmoid, tanh, relu, elu, PReLU, LeakyReLU, ThresholdedReLU
Initializers	zeros, ones, glorot_normal, he_normal
Regularizers	l1, l2, l1_l2
Dropout Rate	uniform distribution over $[0, 1]$
Learning Rate	uniform distribution over $[10^{-4}, 1]$

EM Cell Segmentation Dataset



Original U-Net Predictions



Optimized U-Net Predictions

