Efficient Neural Architecture Search via Parameter Sharing

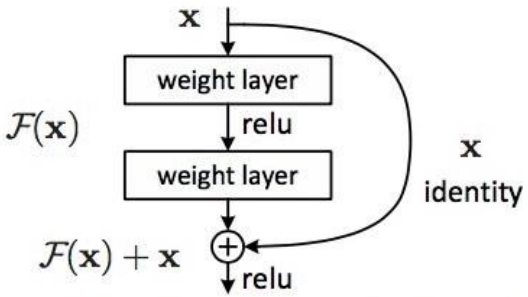
ICML 2018

Authors: Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, Jeff Dean

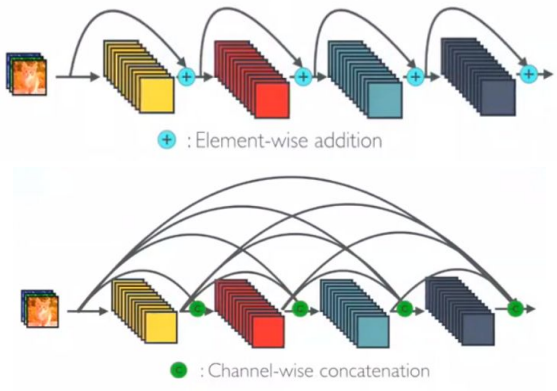
Presented By:
Bhavya Goyal, Nils Palumbo, Abrar Majeedi

Why Network Architecture Design?

- Model architecture improvements (Impressive Gains)
  - ResNet, DenseNet and more
  - ResNeXt, Wide ResNet and more



ResNet



DenseNet

Neural Architecture Design

- Extremely time and compute intensive
- Requires expertise and domain knowledge

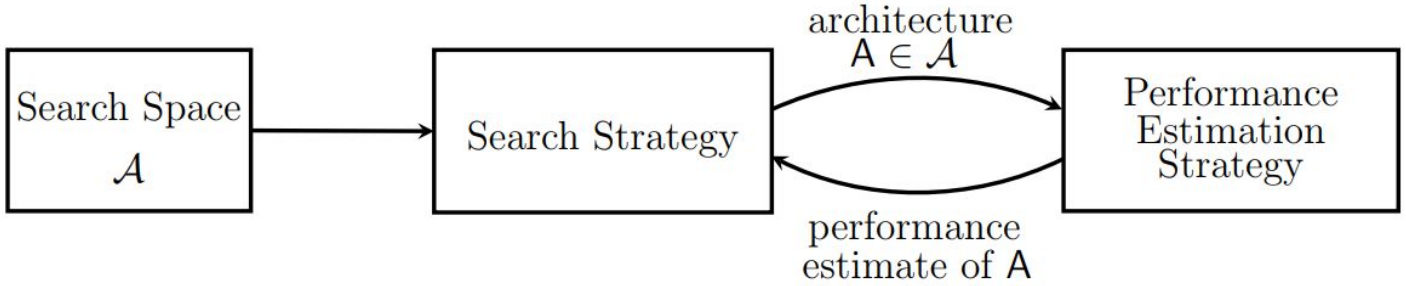
Motivation

Is it the best we can do?

Can we automate this? Can neural networks design neural networks?

Neural Architecture Search

- Automate the design of artificial neural networks

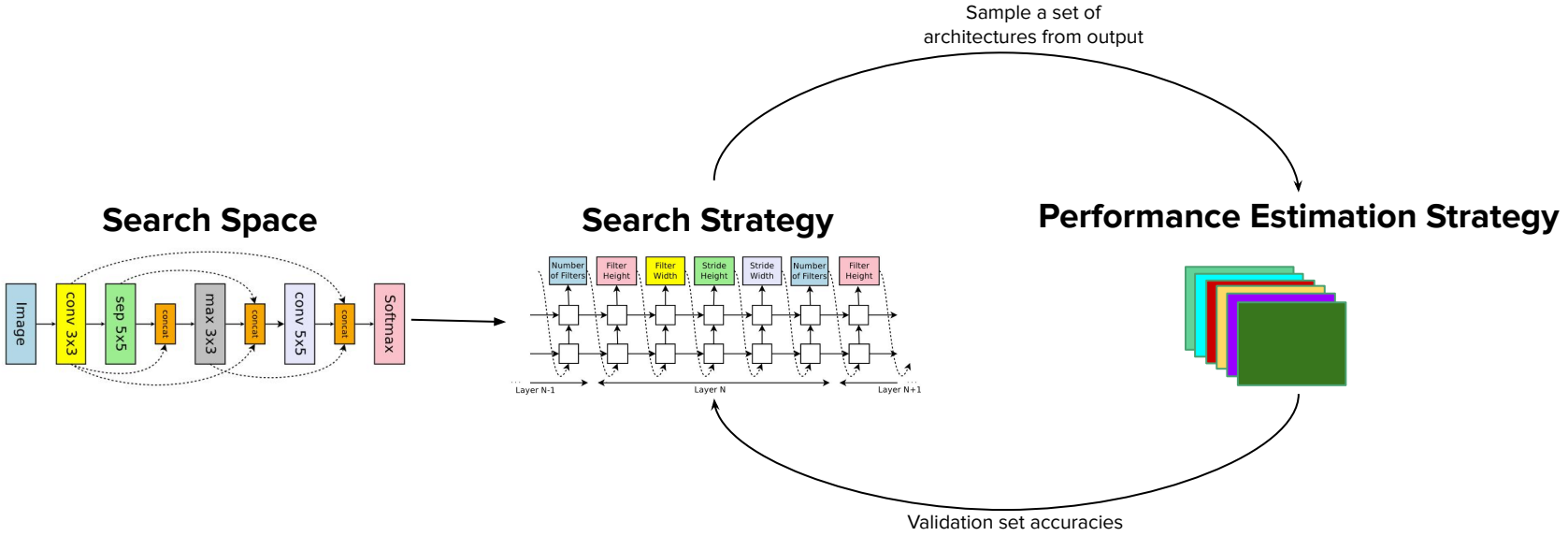


Search Space: which architectures can be represented.

Search Strategy: how to explore the search space

Performance Estimation Strategy: process of estimating the performance on unseen data

Neural Architecture Search

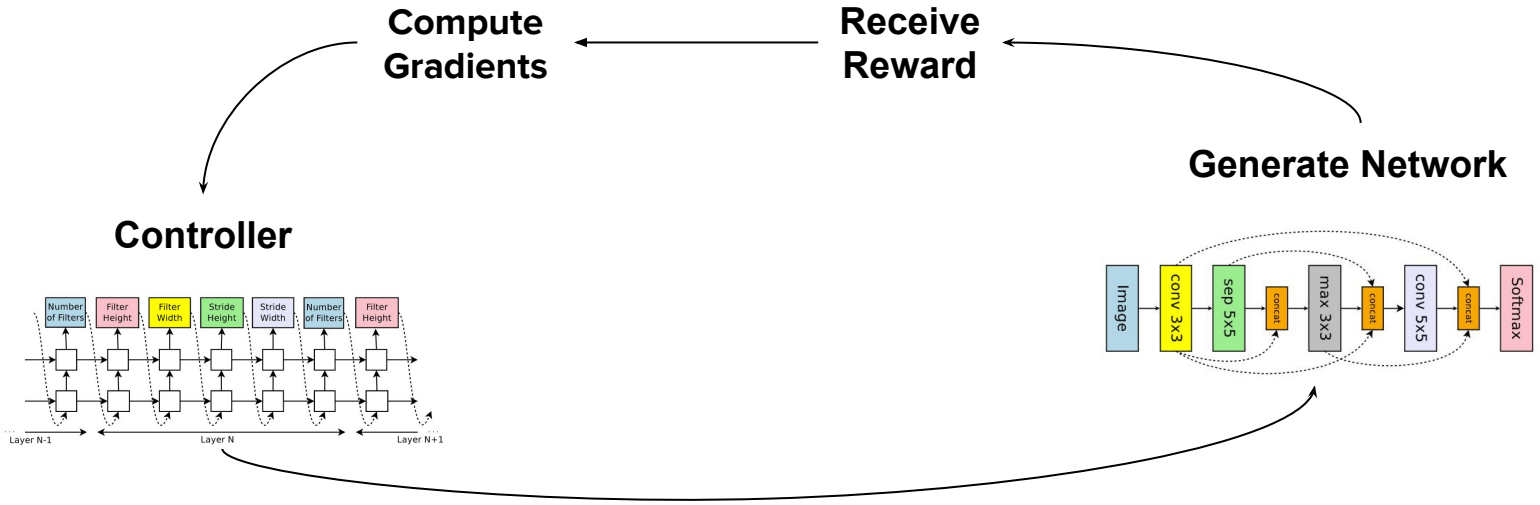


Zoph & Le, 17; Pham et al., 18

Reinforcement Learning in NAS

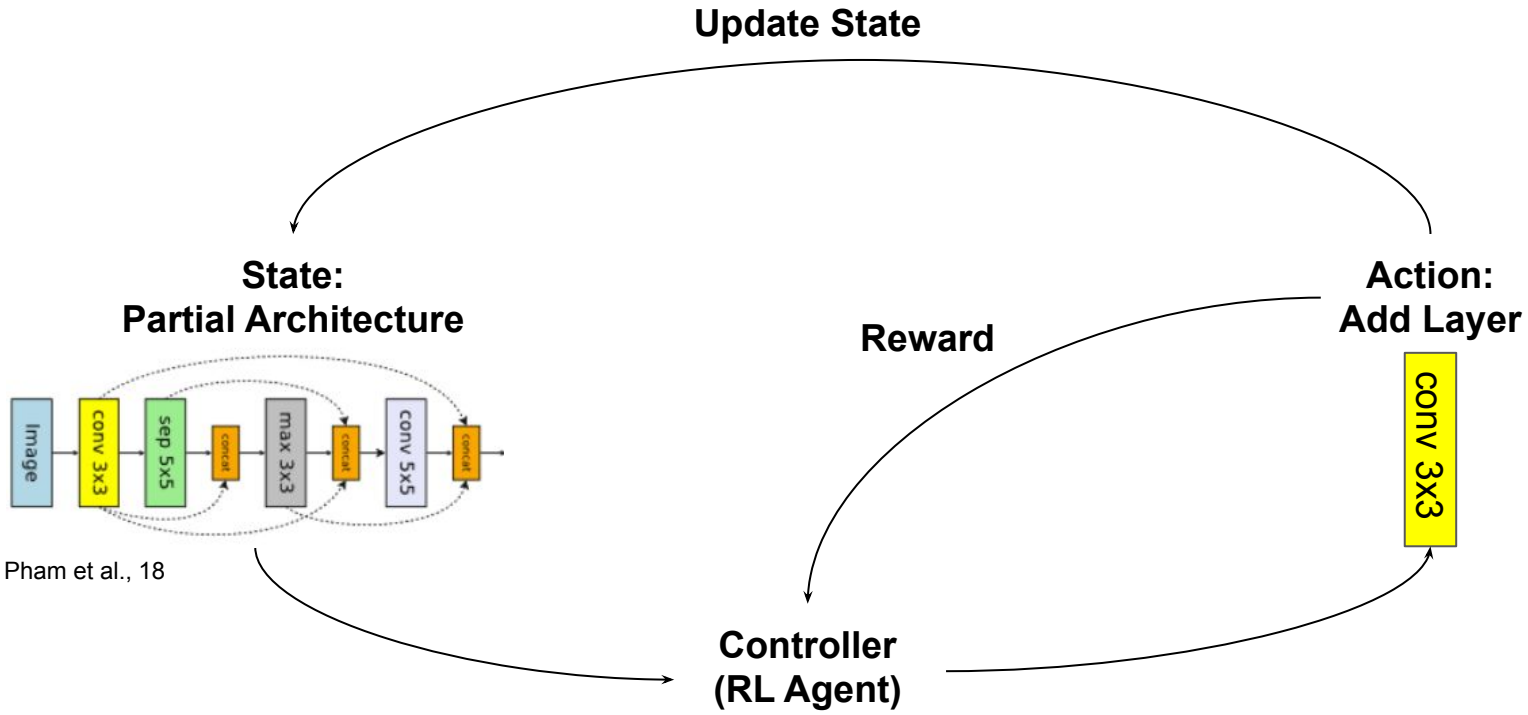
Problem: Validation set accuracy is not a differentiable function of the controller parameters

Solution: Optimize with reinforcement learning (via policy gradients)



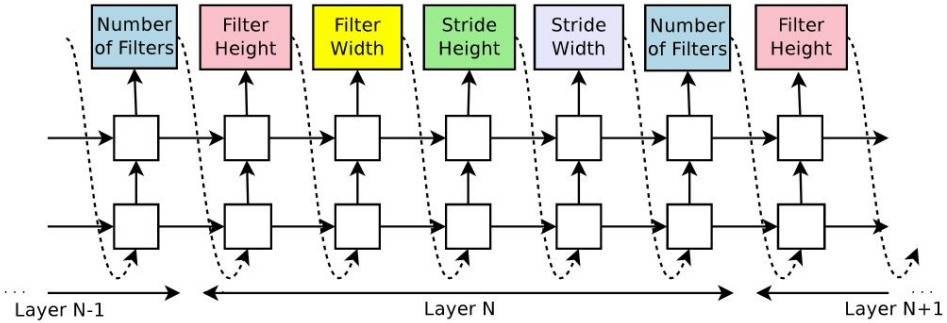
Zoph & Le, 17; Pham et al., 18

Reinforcement Learning in NAS



Zoph & Le; 17

NAS for CNNs


Number of Filters
Filter Height
Filter Width
Stride Height
Stride Width
Number of Filters
Filter Height
Layer N-1
Layer N
Layer N+1

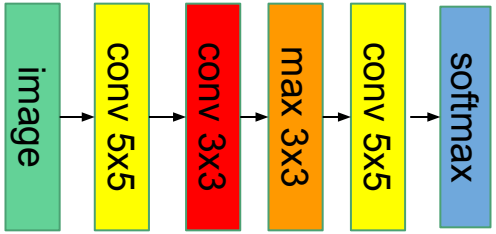
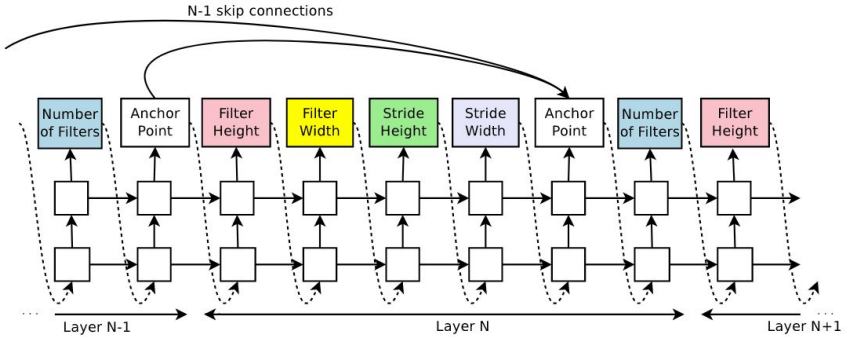
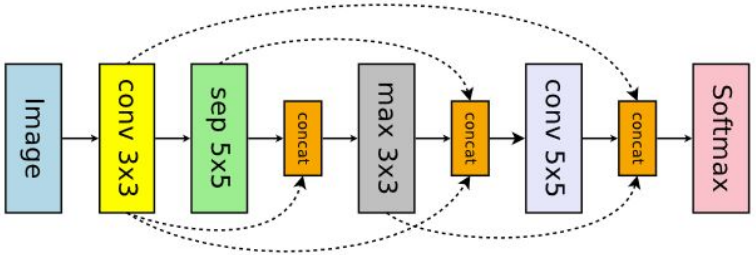


image
conv 5x5
conv 3x3
max 3x3
conv 5x5
softmax

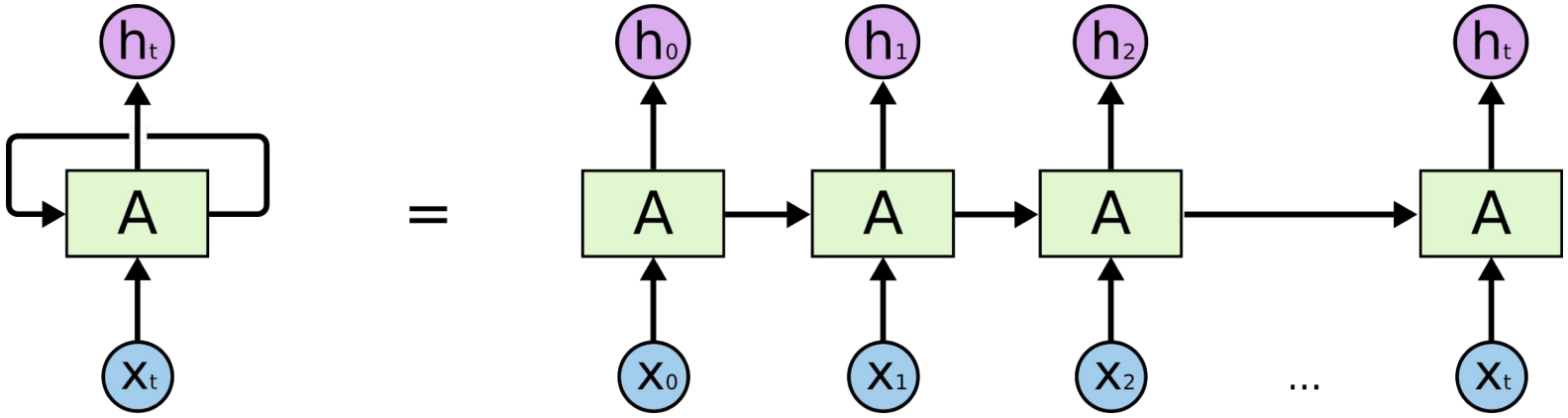
Shortcut Connections with Attention





Zoph & Le, 17; Pham et al., 18

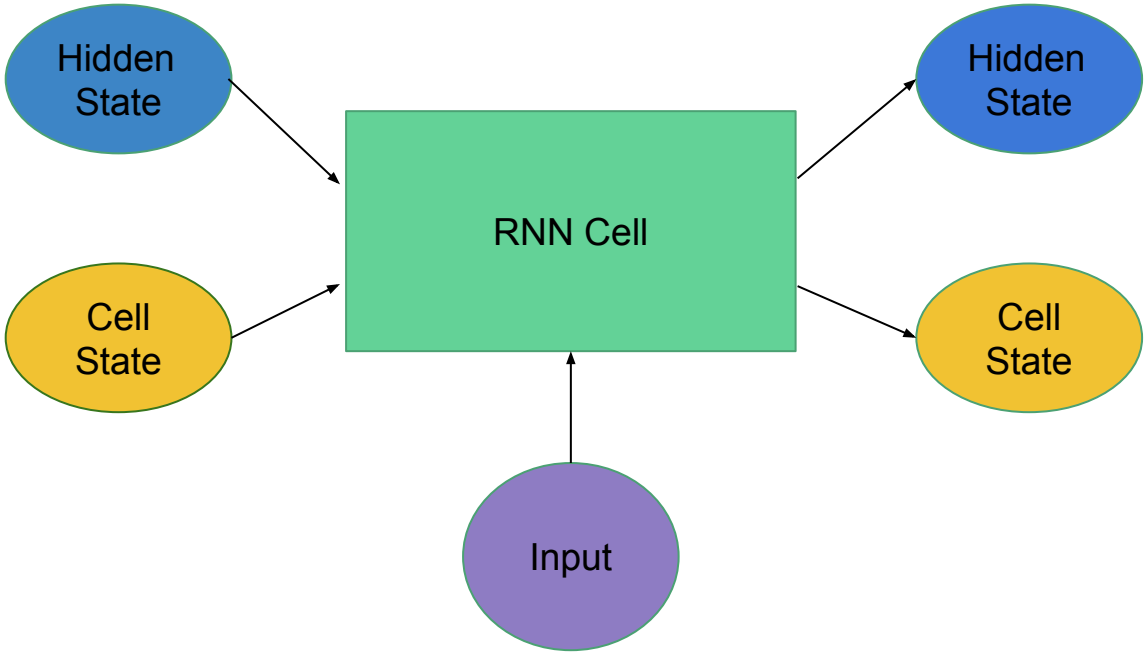
RNNs



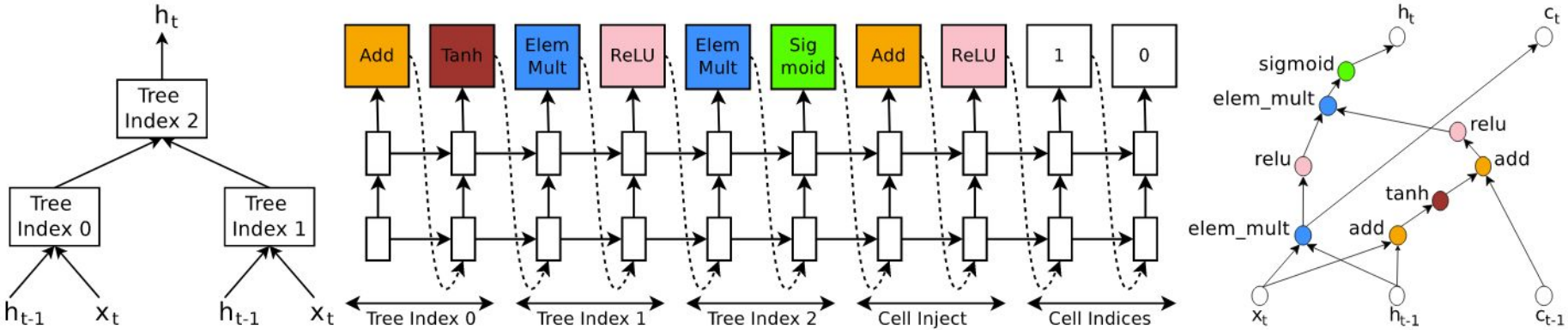
RNN

Unrolled RNN

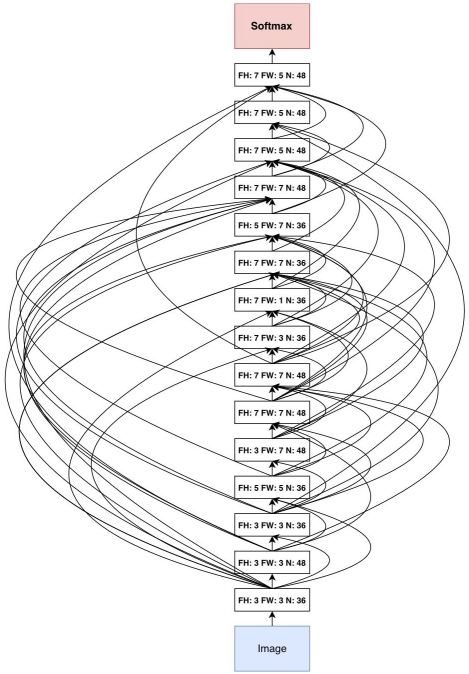
RNN Cells


Hidden State
Cell State
RNN Cell
Input
Hidden State
Cell State

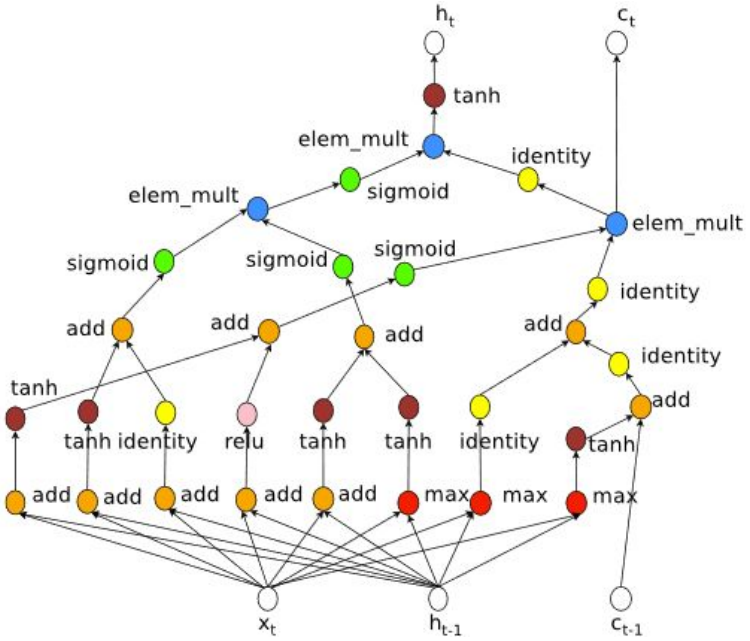
NAS for RNN Cells



Architectures Found



CNN for CIFAR-10
(Image Classification)



RNN Cell for Penn Treebank
(Language Modeling)

Zoph & Le; 17

Results of NAS

- CNN (CIFAR-10)
  Image Classification
  - Comparable performance with fewer layers

Architecture	Layers	Parameters	Test Error (%)
DenseNet-BC	190	25.6M	**3.46**
NAS	39	37.4M	3.65

- RNN (Penn Treebank)
  Language Modeling
  - 5.8% lower test perplexity with twice the performance of the above
  - Zilly et al. required executing its cell 10x per timestep

Architecture	Parameters	Test Perplexity (lower is better)
Previous SOTA (Zilly et al, 2016)	24M	66.0
NAS	54M	**62.4**

Flaws of NAS

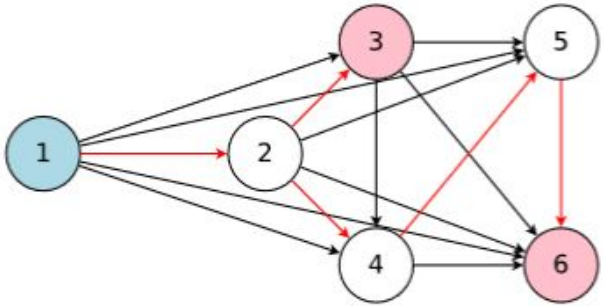
- **Wasteful**: Optimizes model parameters from scratch each time.
  - Each child model is trained to convergence, only to measure its accuracy whilst throwing away all the trained weights.
- **Computationally Intensive**: Zoph et al. used 450 GPUs for 3-4 days
  - Equivalently, 32k-43k GPU hours

ENAS

Main Idea

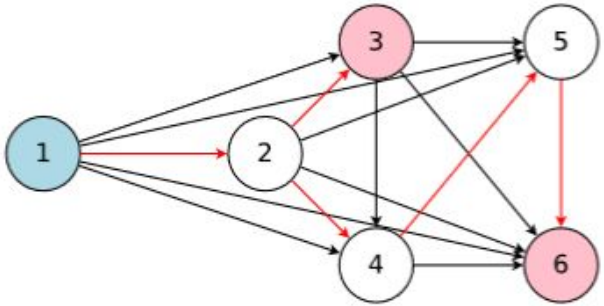
- Share parameters among models in the search space
- Alternate between optimizing model parameters on the training set and controller parameters on the validation set

Key Assumption: parameters that work well for one model architecture, would work well for others



How do we improve on NAS?

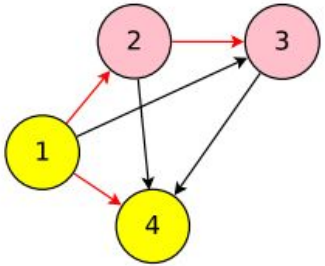
- The graphs on which NAS iterates can be viewed as sub-graphs of a larger graph:
  - Represent NAS's search space using a single **directed acyclic graph (DAG)**.



ENAS for RNN

To create a recurrent cell, the controller RNN samples N blocks of decisions implemented as a DAG with N nodes where:

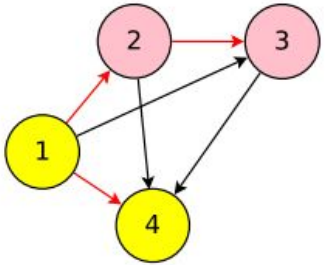
- Nodes represent local computations
- Edges represent the flow of information between the N nodes.



ENAS for RNN

To create a recurrent cell, the controller RNN samples N blocks of decisions implemented as a DAG with N nodes where:

- Nodes represent local computations
- Edges represent the flow of information between the N nodes.

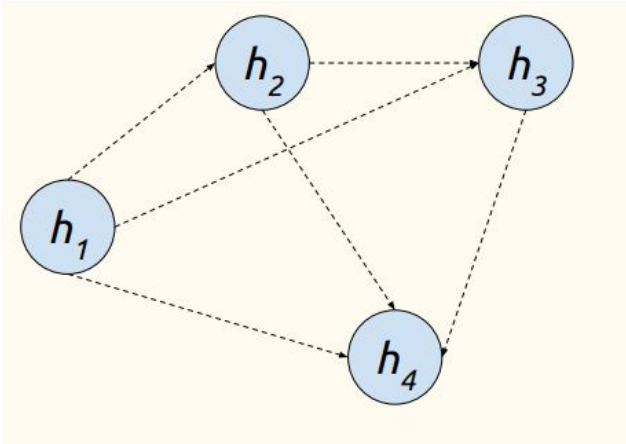


ENAS's controller is an RNN that decides:

1) which edges are activated
2) which computations are performed at each node in the DAG.

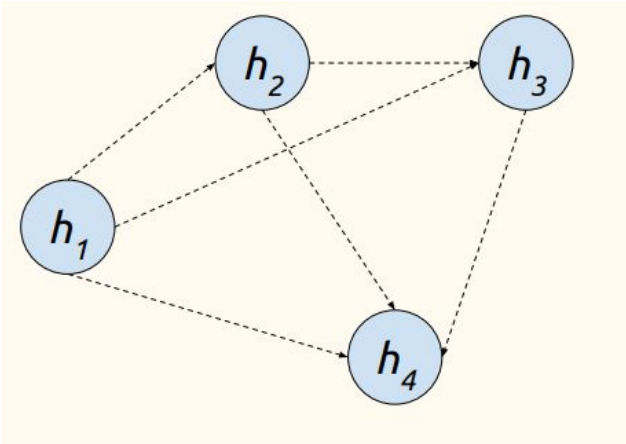
ENAS for RNN

- At step i,controller decides:
  - Which previous node $j \in \{1, \ldots, i-1\}$ to connect to node i
  - An activation function $\varphi_i \in \{\tanh, \text{ReLU}, \text{Id}, \sigma\}$
- Then,
  - $h_i = \varphi_i(W_{ij} h_j)$



ENAS for RNN

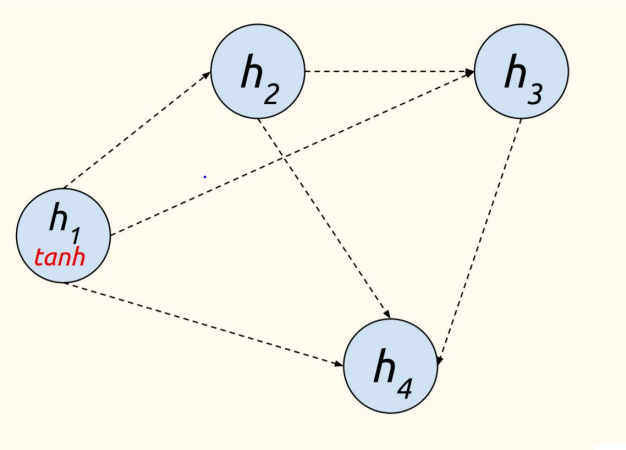
Example: N = 4, Let $x^{(t)}$ be the input signal for a recurrent cell (e.g. word embedding), and $h_{(t-1)}$ be the output from the previous time step.



ENAS for RNN

- Controller selects:
  - Step 1: tanh

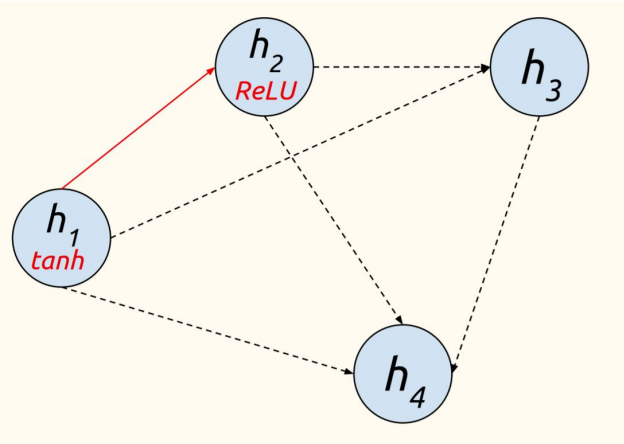
- Function computed:
  - $h_1 = \tanh(W_x x^{(t)} + W_h h_{t-1})$



ENAS for RNN

- Controller selects:
  - Step 1: tanh
  - Step 2: 1, ReLU

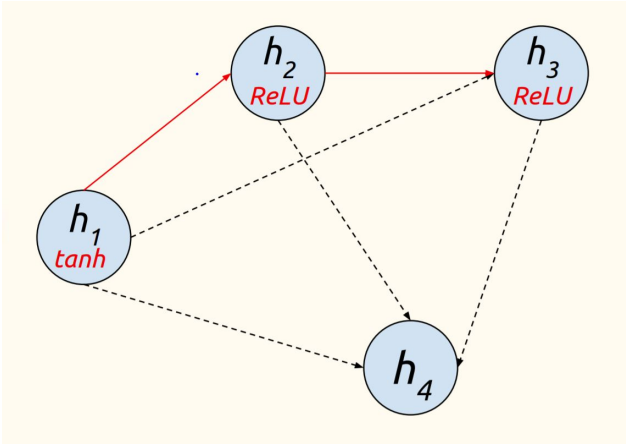
- Function computed:
  - $h_1 = \tanh(W_x x^{(t)} + W_h h_{t-1})$
  - $h_2 = \text{ReLU}(W_{12} h_1)$



ENAS for RNN

- Controller selects:
  - Step 1: tanh
  - Step 2: 1, ReLU
  - Step 3: 2, ReLU

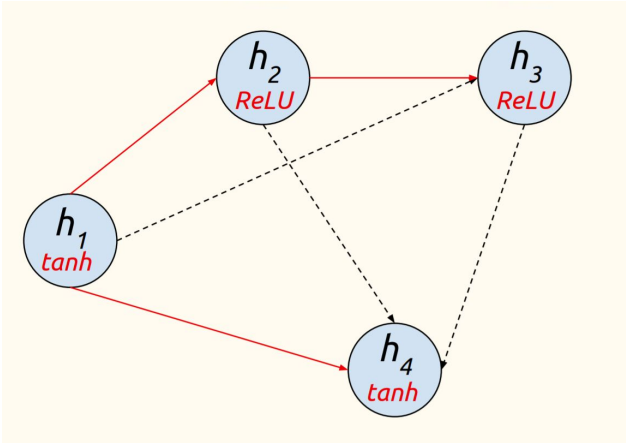
- Function computed:
  - $h_1 = \tanh(W_x x^{(t)} + W_h h_{t-1})$
  - $h_2 = \text{ReLU}(W_{12} h_1)$
  - $h_3 = \text{ReLU}(W_{23} h_2)$



ENAS for RNN

- Controller selects:
  - Step 1: tanh
  - Step 2: 1, ReLU
  - Step 3: 2, ReLU
  - Step 4: 1, tanh

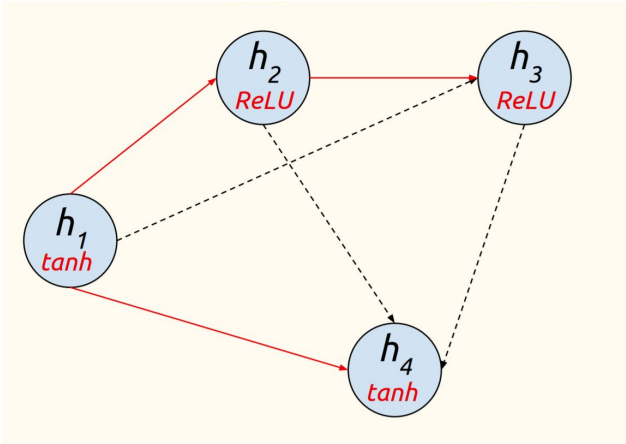
- Function computed:
  - $h_1 = \tanh(W_x x^{(t)} + W_h h_{t-1})$
  - $h_2 = \text{ReLU}(W_{12} h_1)$
  - $h_3 = \text{ReLU}(W_{23} h_2)$
  - $h_4 = \tanh(W_{14} h_1)$



ENAS for RNN

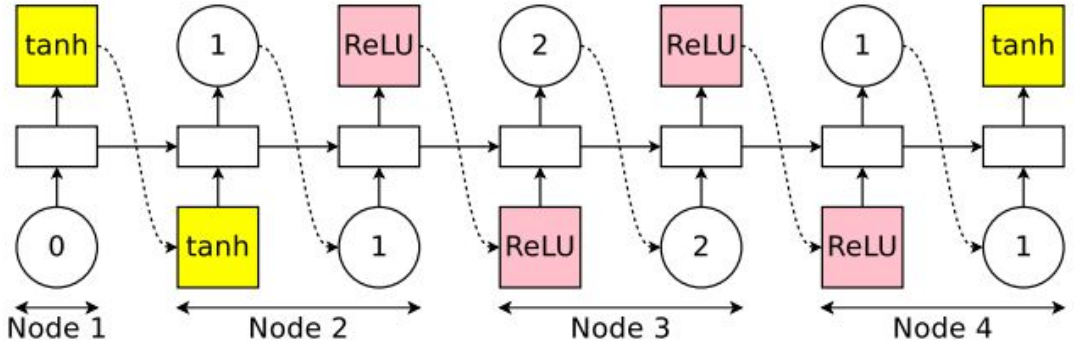
- Controller selects:
  - Step 1: tanh
  - Step 2: 1, ReLU
  - Step 3: 2, ReLU
  - Step 4: 1, tanh

- Function computed:
  - $h_1 = \tanh(W_x x^{(t)} + W_h h_{t-1})$
  - $h_2 = \text{ReLU}(W_{12} h_1)$
  - $h_3 = \text{ReLU}(W_{23} h_2)$
  - $h_4 = \tanh(W_{14} h_1)$
  - $h^{(t)} = ½ (h_3 + h_4)$

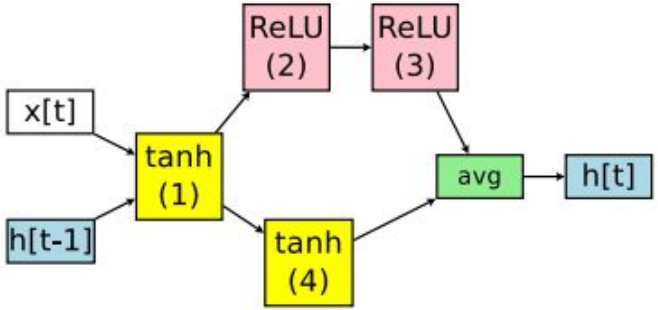


ENAS controller for RNN

RNN Controller

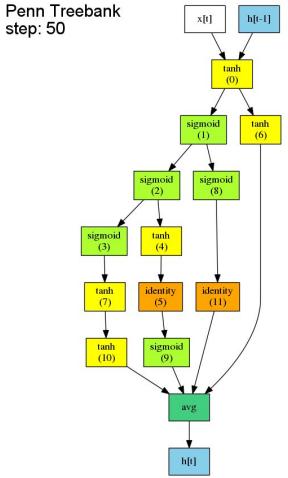


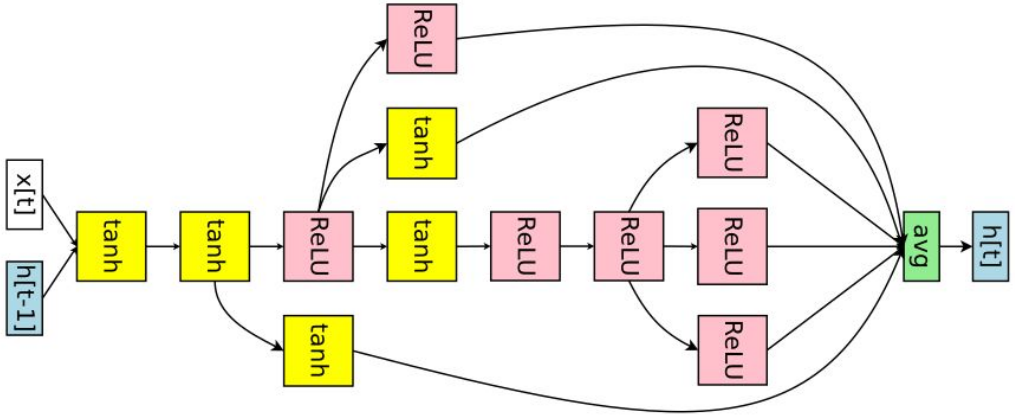
Child model Arch.



ENAS discovered RNN cell

- Search space size:
  - $4^N \times N!$





The RNN cell ENAS discovered for Penn Treebank.

*language modeling on Penn Treebank

Results for ENAS RNN

Brief results of ENAS for recurrent cell on Penn Treebank

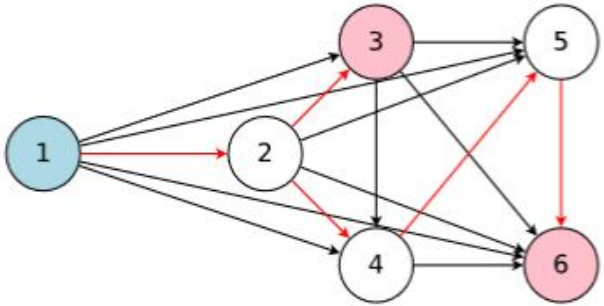
Architecture	Perplexity* (lower is better)
Mixture of Softmaxes (current State of the Art)	56.0
NAS (Zoph & Le, 2017)	62.4
ENAS	**55.8**

The search process of ENAS, in terms of GPU hours, is more than 1000x faster.

* Language modeling on Penn Treebank

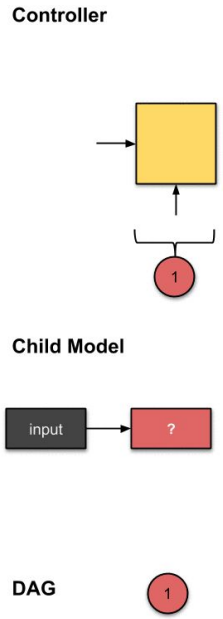
ENAS for CNN

- At step i,
  - Which previous node j $\in$ {1, ..., i-1} to connect to node i
  - Which computation operation to use: {conv $_{3x3}$, conv $_{5x5}$, sepconv $_{3x3}$, sepconv $_{5x5}$, maxpool $_{3x3}$, avgpool $_{3x3}$}

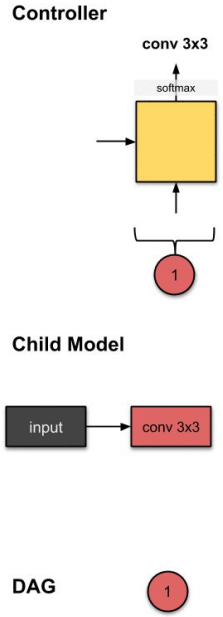


Designing Conv Network

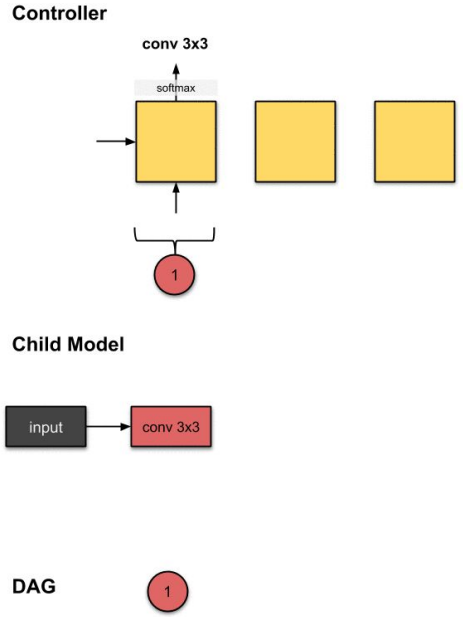
Designing Conv Network



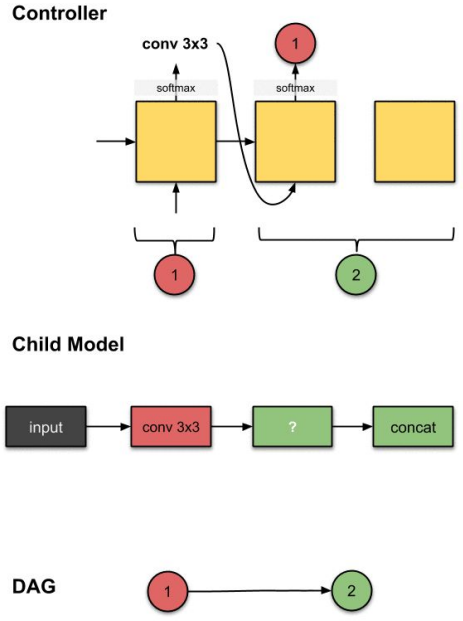
Designing Conv Network



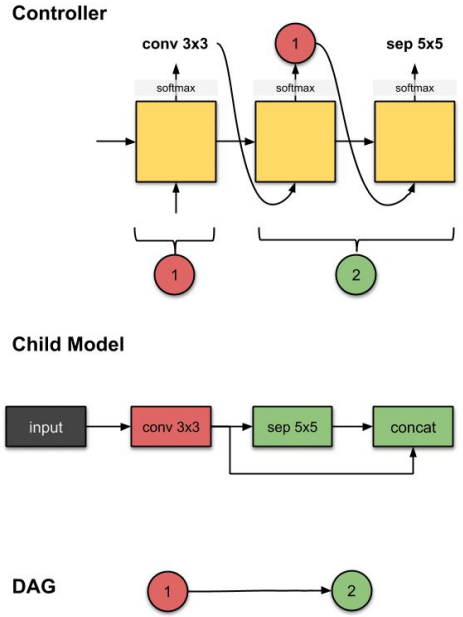
Designing Conv Network



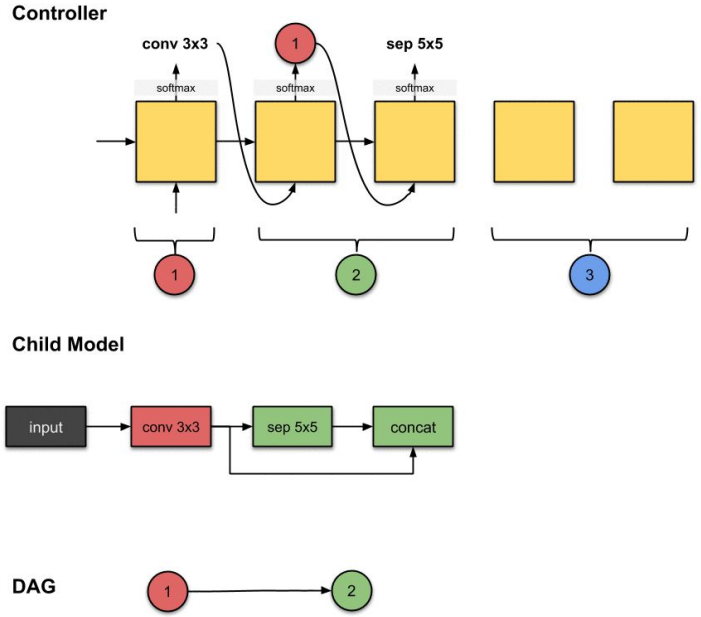
Designing Conv Network



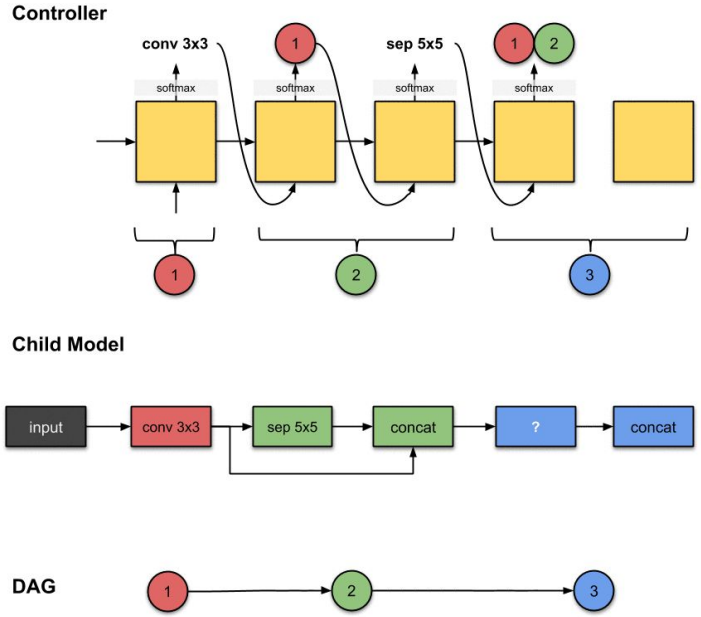
Designing Conv Network



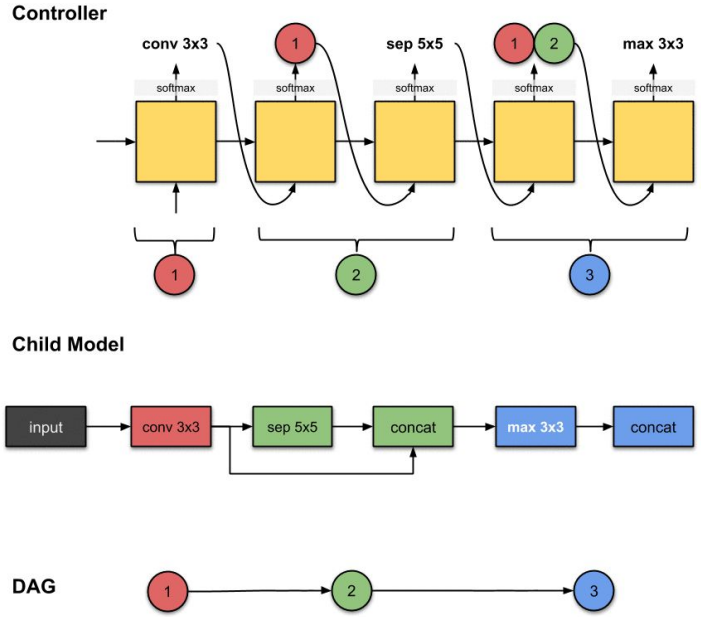
Designing Conv Network



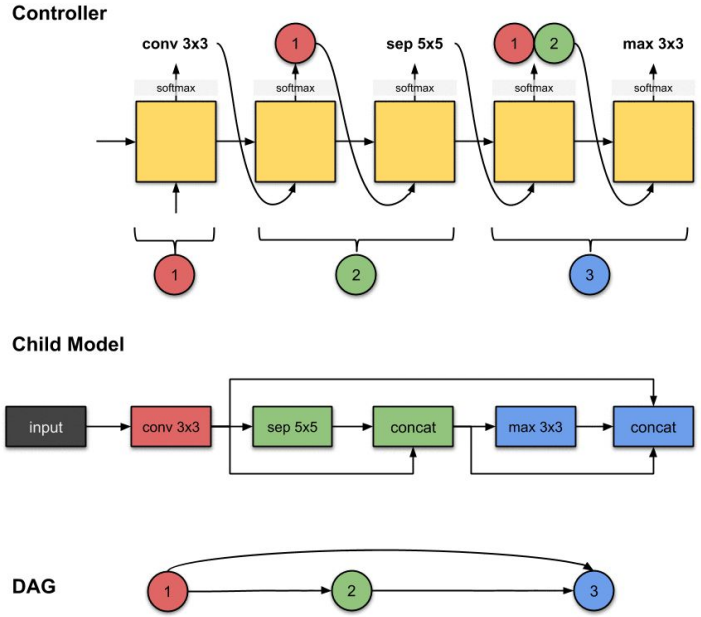
Designing Conv Network



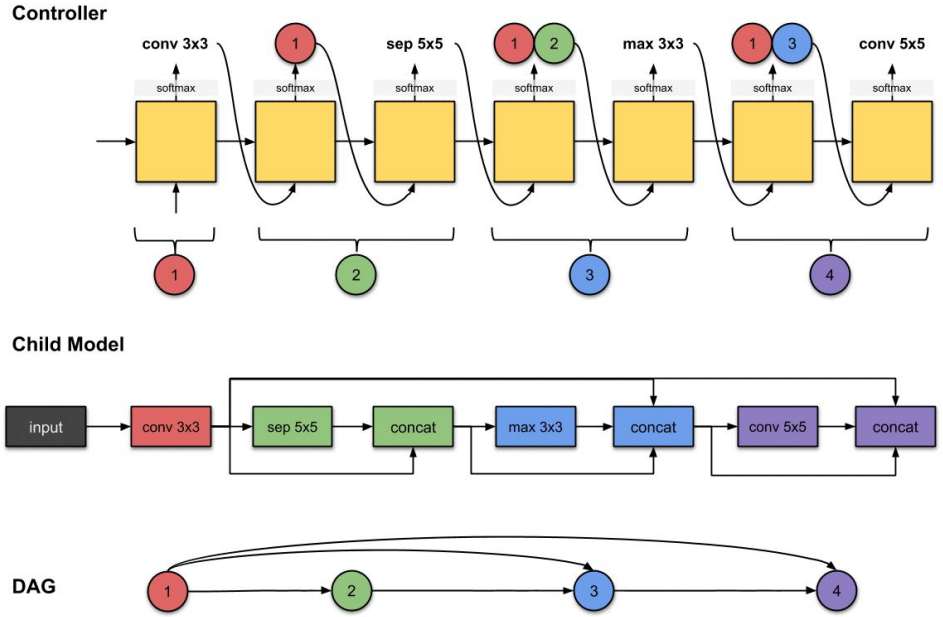
Designing Conv Network



Designing Conv Network



Designing Conv Network



Designing Conv Network

- Search space is huge - with 12 layers, 1.6×10^{29} possible networks
  - For L layers
  - #configures: $6^L \times 2^{L(L-1)/2}$

ENAS for CNN

Macro Search: Designing entire convolutional networks

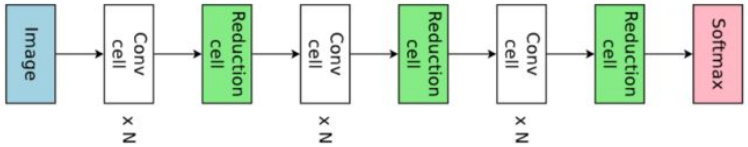
- NAS by Zoph and Le, FractalNet and SMASH

Micro Search: Designing convolutional building blocks (or modules)

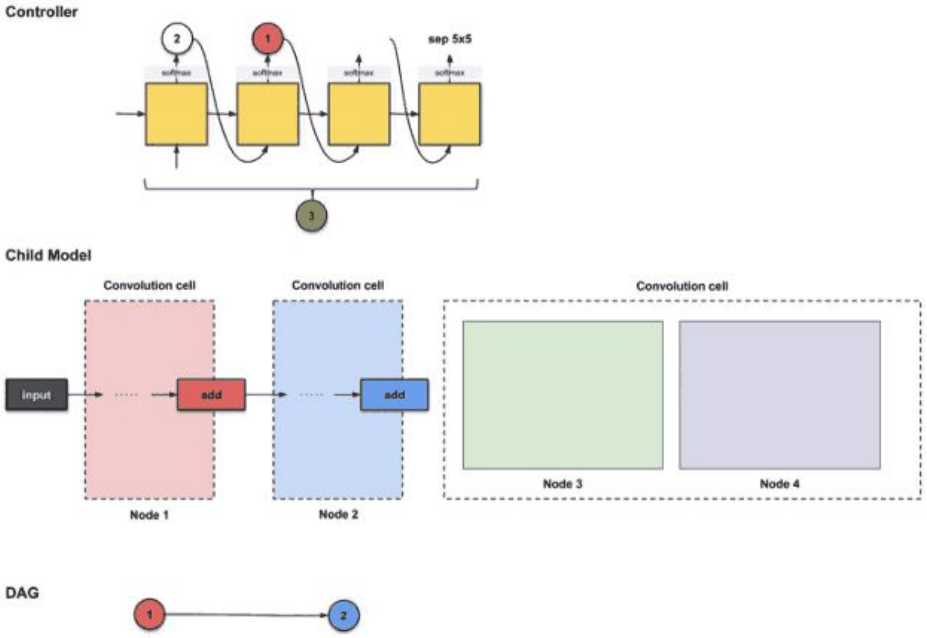
- Hierarchical NAS, Progressive NAS and NASNet

Micro Search

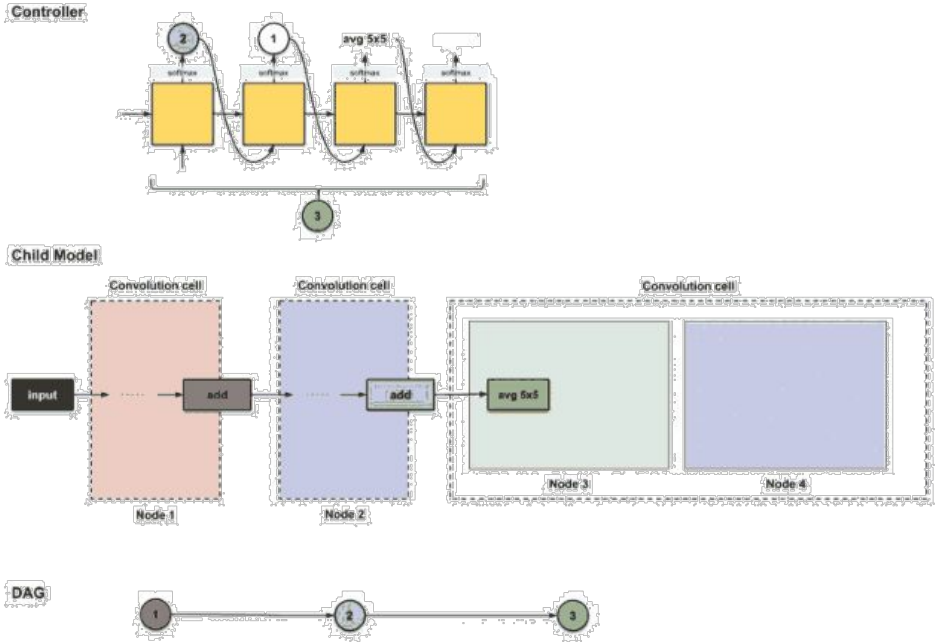
- A child model consists of several **blocks**.
- Each block consists of *N* convolutional **cells** and 1 reduction cell.
- Each convolutional/ reduction cell comprises of *B* **nodes**.



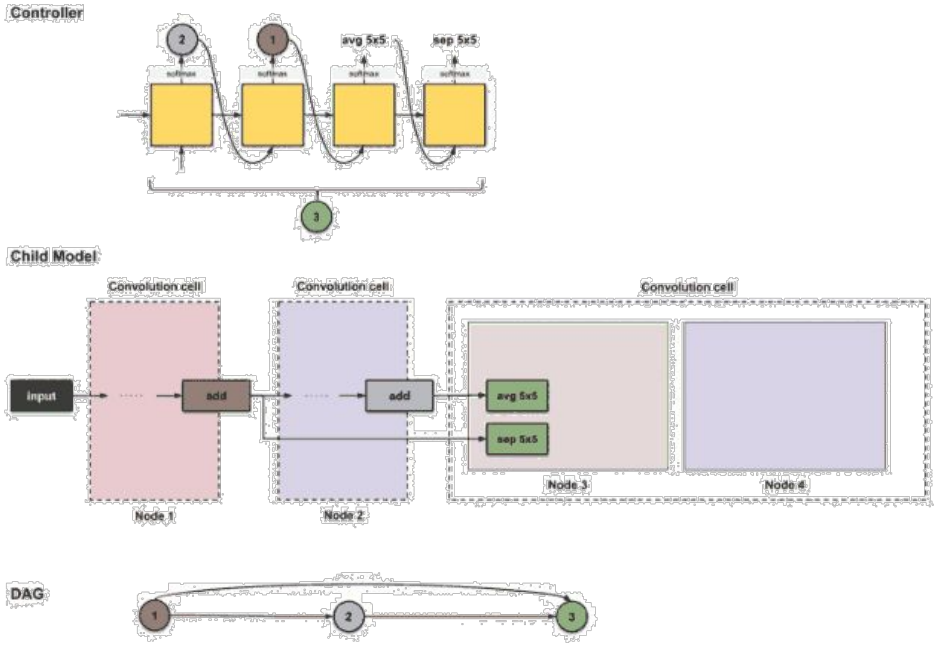
Designing Conv Blocks



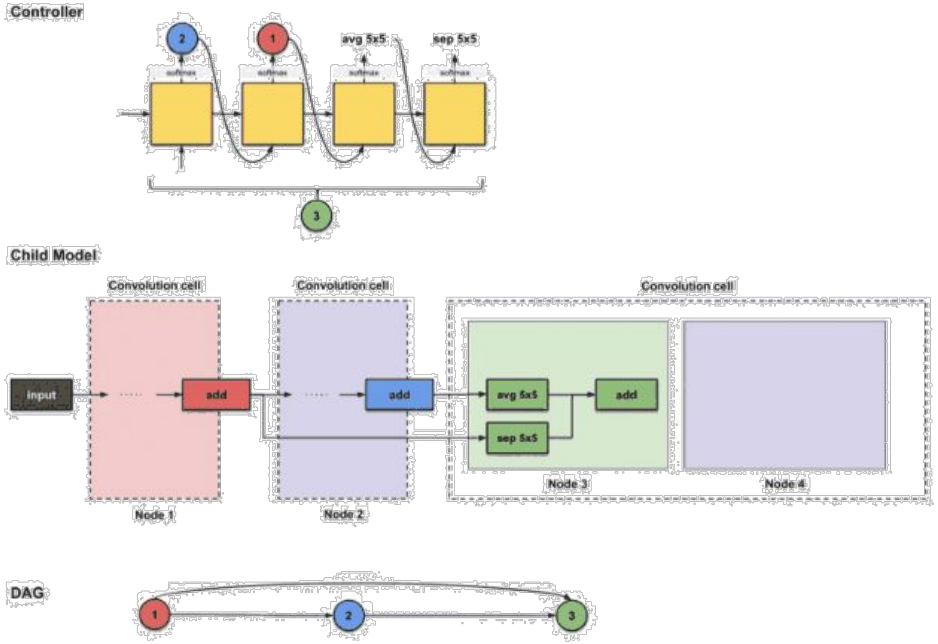
Designing Conv Blocks



Designing Conv Blocks



Designing Conv Blocks

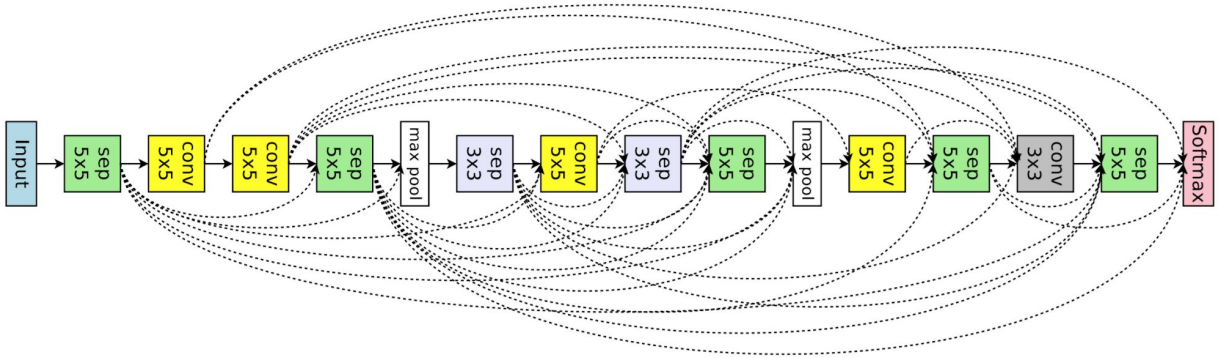


Designing Conv Blocks

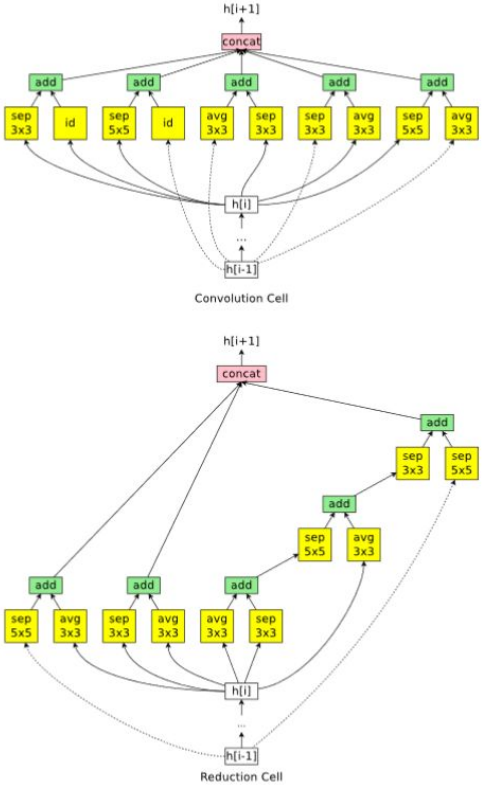
- Search space - with 7 nodes, 1.3×10^{11} configurations
  - B nodes with two nodes from previous nodes connected
  - #configurations for a cell: $(5 \times (B-2)!)^2$
  - #configurations for a convolution/reduction cell: $(5 \times (B-2)!)^4$

ENAS discovered networks

- ~7 hours to find an architecture



Macro search



Micro search

Results

- Comparable performance to NAS
- Reducing #GPU-hours by more than 50,000x compared to NAS.

Method	GPUs	Times (days)	Params (million)	Error (%)
DenseNet-BC (Huang et al., 2016)	–	–	25.6	3.46
DenseNet + Shake-Shake (Gastaldi, 2016)	–	–	26.2	2.86
DenseNet + CutOut (DeVries & Taylor, 2017)	–	–	26.2	**2.56**
Budgeted Super Nets (Veniat & Denoyer, 2017)	–	–	–	9.21
ConvFabrics (Saxena & Verbeek, 2016)	–	–	21.2	7.43
Macro NAS + Q-Learning (Baker et al., 2017a)	10	8-10	11.2	6.92
Net Transformation (Cai et al., 2018)	5	2	19.7	5.70
FractalNet (Larsson et al., 2017)	–	–	38.6	4.60
SMASH (Brock et al., 2018)	1	1.5	16.0	4.03
NAS (Zoph & Le, 2017)	800	21-28	7.1	4.47
NAS + more filters (Zoph & Le, 2017)	800	21-28	37.4	**3.65**
ENAS + macro search space	1	0.32	21.3	4.23
ENAS + macro search space + more channels	1	0.32	38.0	**3.87**
Hierarchical NAS (Liu et al., 2018)	200	1.5	61.3	3.63
Micro NAS + Q-Learning (Zhong et al., 2018)	32	3	–	3.60
Progressive NAS (Liu et al., 2017)	100	1.5	3.2	3.63
NASNet-A (Zoph et al., 2018)	450	3-4	3.3	3.41
NASNet-A + CutOut (Zoph et al., 2018)	450	3-4	3.3	**2.65**
ENAS + micro search space	1	0.45	4.6	3.54
ENAS + micro search space + CutOut	1	0.45	4.6	**2.89**

*Image classification results on CIFAR-10

Importance of ENAS (Ablation Study)

Comparing to Guided Random Search

Uniformly sample a

- Recurrent cell
- Entire convolutional network
- Pair of convolutional and reduction cells

And train using same settings as ENAS

	Test Perplexity
ENAS	**55.8**
Random Guided Search	81.2

Results on Penn Treebank

	Test Error (%)
ENAS	**4.23**
Random Guided Search	5.86

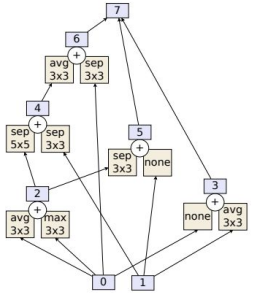
Classification Results on CIFAR-10

Limitations of ENAS/NAS

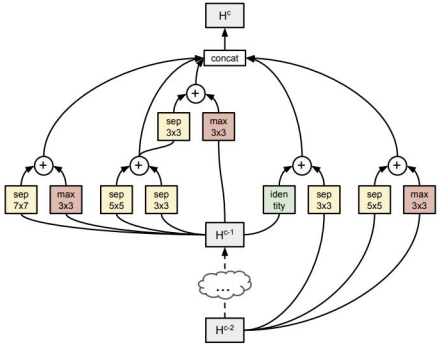
- Searching on larger datasets like ImageNet, expect different architectures
- Other modules like attention module etc. can also be included
- NAS can only organise basic building blocks of model architecture, can not come up with novel design
- Search space design is still important
- Decreases the interpretability of the model architecture.

Related Work

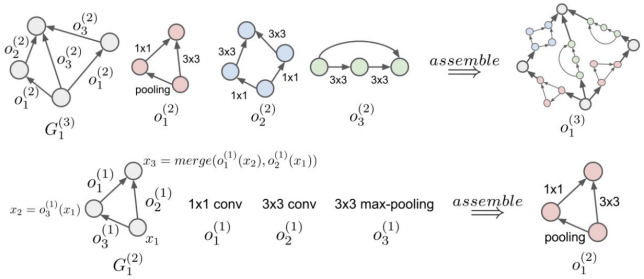
- Regularized Evolution for Image Classifier Architecture Search



- Progressive NAS



- Hierarchical Representations for Efficient Architecture Search
- SMASH: One-Shot Model Architecture Search through HyperNetworks



Conclusion

- NAS demonstrates that neural networks can design architectures comparable to or better than the best human-designed solutions
- By speeding up NAS by more than 1000x, ENAS paves the way for practical automated design of neural networks

Neural networks design neural networks (AI gives birth to AI)

Thank you!

Any Questions?

References

- *Neural Architecture Search with Reinforcement Learning*, Barret Zoph, Quoc V. Le
- *Efficient Neural Architecture Search via Parameter Sharing*, Hieu Pham et al.
- *Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning*, Ronald J. Williams
- *Regularized Evolution for Image Classifier Architecture Search*, Esteban Real et al.
- *Progressive Neural Architecture Search*, Chenxi Liu et al.
- *Hierarchical Representations for Efficient Architecture Search*, Hanxiao Liu et al.
- *SMASH: One-Shot Model Architecture Search through HyperNetworks*, Andrew Brock et al.
- *Understanding LSTM Networks,* Christopher Olah
- *Recent Architecture Advances in Neural Architecture Search*, Hao Chen

Importance of ENAS

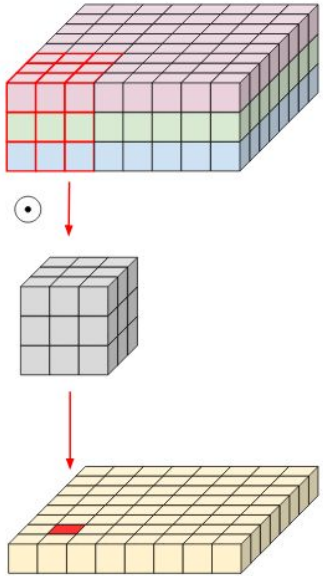
Disabling ENAS Search

- Train with shared parameters without updating controller
- Untrained random controller is similar to dropout/drop-path

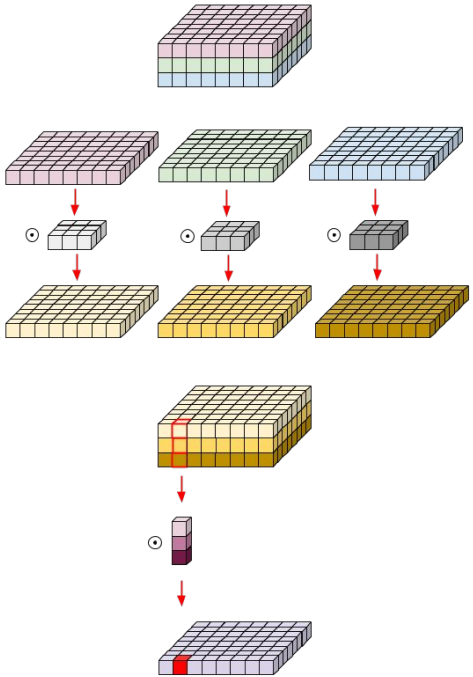
	Test Error (%)
ENAS	**4.23**
Disabling ENAS	8.92

Classification Results on CIFAR-10

Depth-wise Separable Convolution



Normal Convolution



Depth-wise separable Convolution