NASNet and AutoML

Authors: Google Brain Team

Thursday 12th March, 2020

Presented By: Oladotun Aluko

Content

Introduction

Introduction

- Developing neural network image classification models often requires significant architecture engineering
- Exploring meta-learning to predict new neural network architecures
- Searching for a good architecture on a proxy dataset and transferring architecture to a larger dataset so that the complexity of the architecture is independent of the depth of the network and the size of input images
- Automating model selection and hyperparameter optimization

Methodology

Methodology

- Using search methods to find convolutional architectures on the dataset of interest. The main search method used is the Neural Architecture Search (NAS) framework
- In NAS, a controller recurrent neural network (RNN) samples child networks with different architectures. The child networks are trained to convergence to obtain some accuracy on a held-out validation set. The resulting accuracies are used to update the controller so that the controller will generate better architectures over time

Methodology(contd)

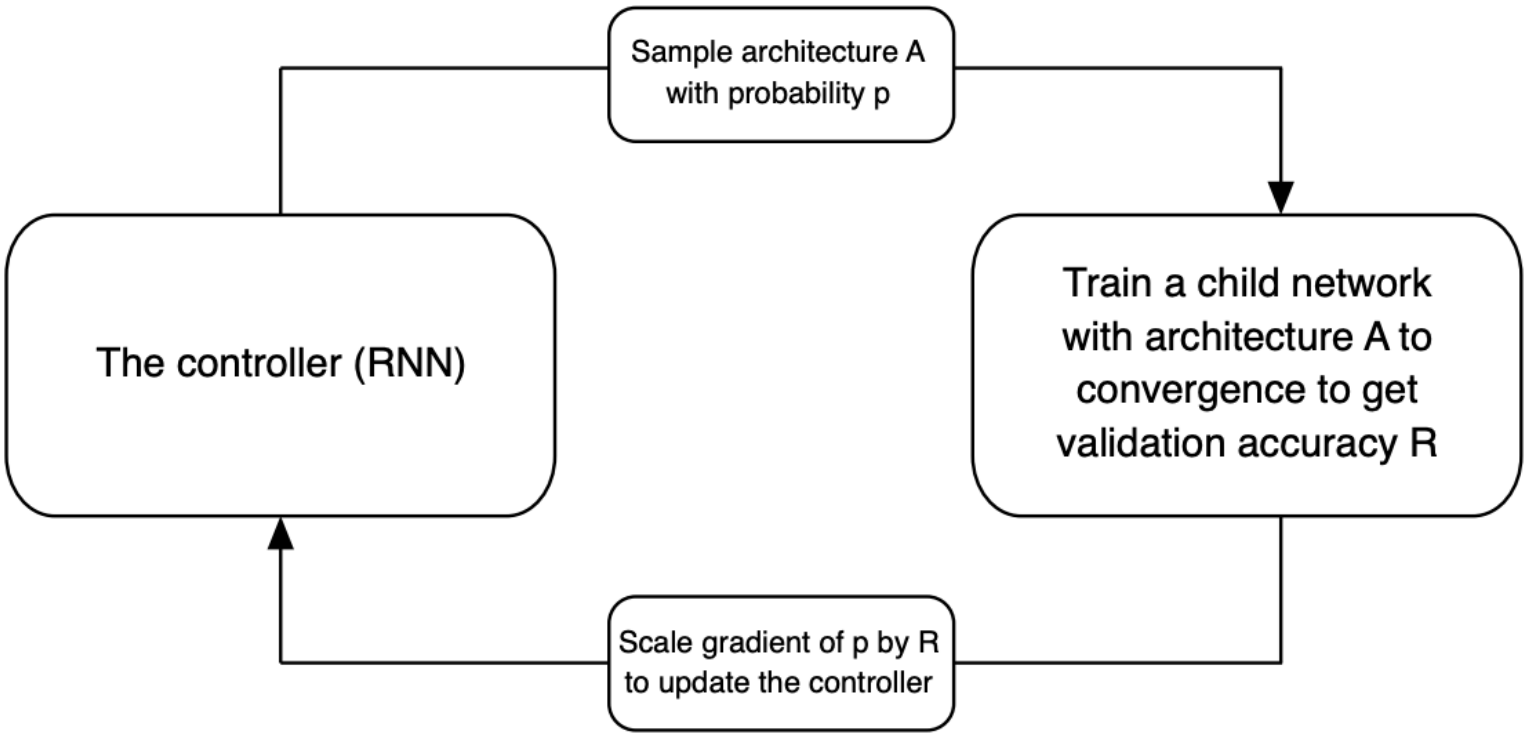


Figure 1. Overview of Neural Architecture Search [71]. A controller RNN predicts architecture A from a search space with probability p. A child network with architecture A is trained to convergence achieving accuracy R. Scale the gradients of p by R to update the RNN controller.

Methodology(contd)

- The actual work is the design of a novel search space, such that the best architecture found on the CIFAR-10 dataset would scale to larger, higher resolution image datasets across a range of computational settings. This search space is named the NASNet search space
- Architecture engineering with CNNs often identifies repeated motifs consisting of combinations of convolutional filter banks, nonlinearities and a prudent selection of connections to achieve state-of-the-art results such as the repeated modules present in the Inception and ResNet models

Methodology(contd)

- These observations suggest that it may be possible for the controller RNN to predict a generic convolutional cell expressed in terms of these motifs. This cell can then be stacked in series to handle inputs of arbitrary spatial dimensions
- The overall architectures of the convolutional nets are manually predetermined. They are composed of convolutional cells repeated many times where each convolutional cell has the same architecture, but different weights

Methodology(contd)

- The highlight is the design of two convolutional cells to serve two main functions when taking in a feature map as input:
  1. A normal cell that returns a feature map of the same dimension
  2. A reduction cell that returns a feature map where the feature map height and width is reduced by a factor of two

Methodology(contd)

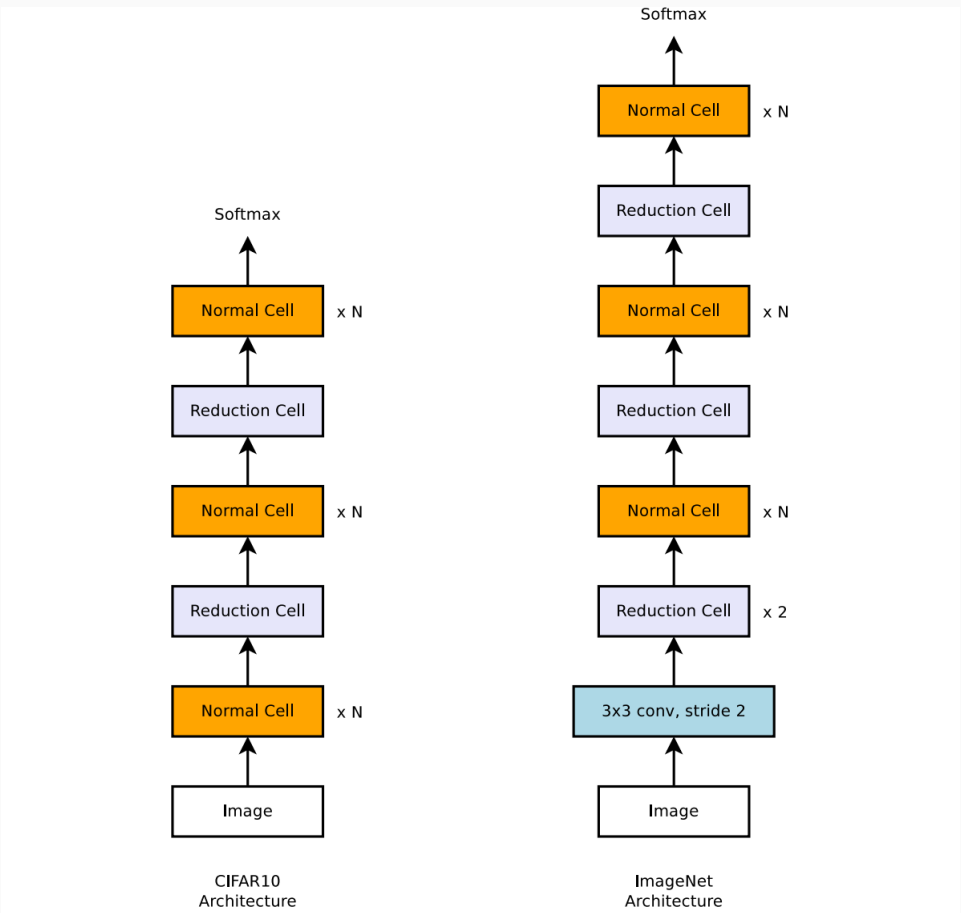


Figure 2. Scalable architectures for image classification consist of two repeated motifs termed *Normal Cell* and *Reduction Cell*. This diagram highlights the model architecture for CIFAR-10 and ImageNet. The choice for the number of times the Normal Cells that gets stacked between reduction cells, N, can vary in our experiments.

Methodology(contd)

- The structures of the cells can be searched within a search space. In the search space used, each cell receives as input two initial hidden states hi and hi−1 which are the outputs of two cells in the previous two lower layers or the input image. The controller RNN recursively predicts the rest of the structure of the convolutional cell, given these two initial hidden states

Methodology(contd)

Step 1. Select a hidden state from h_i, h_{i-1} or from the set of hidden states created in previous blocks.

Step 2. Select a second hidden state from the same options as in Step 1.

Step 3. Select an operation to apply to the hidden state selected in Step 1.

Step 4. Select an operation to apply to the hidden state selected in Step 2.

Step 5. Select a method to combine the outputs of Step 3 and 4 to create a new hidden state.

Methodology(contd)

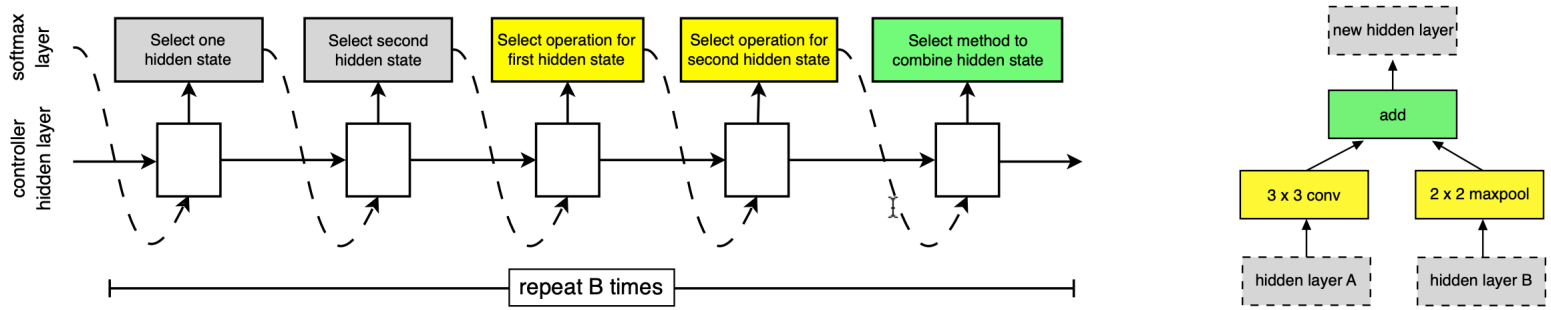
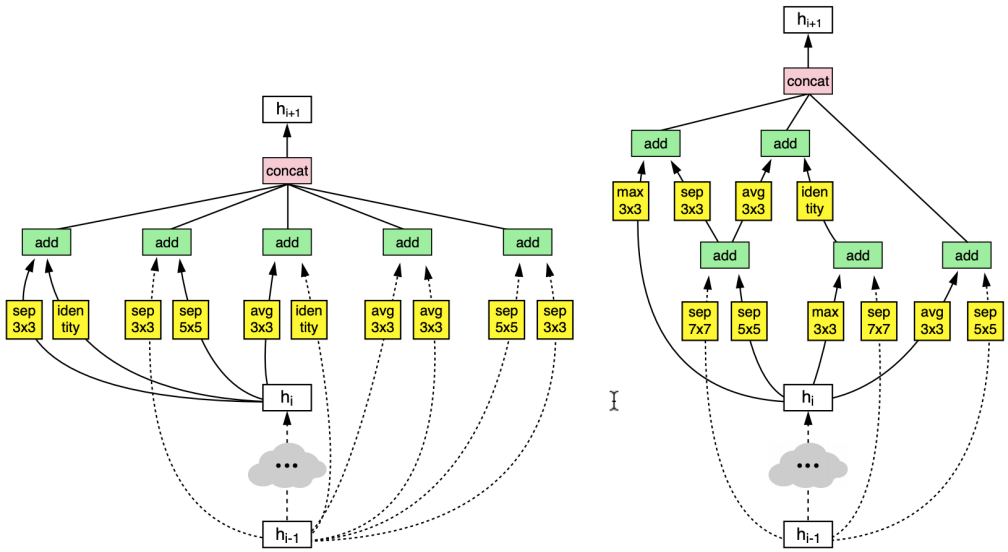


Figure 3. Controller model architecture for recursively constructing one block of a convolutional cell. Each block requires selecting 5 discrete parameters, each of which corresponds to the output of a softmax layer. Example constructed block shown on right. A convolutional cell contains B blocks, hence the controller contains $5B$ softmax layers for predicting the architecture of a convolutional cell. In our experiments, the number of blocks B is 5.

Methodology(contd)

- identity
- 1x7 then 7x1 convolution
- 3x3 average pooling
- 5x5 max pooling
- 1x1 convolution
- 3x3 depthwise-separable conv
- 7x7 depthwise-separable conv
- 1x3 then 3x1 convolution
- 3x3 dilated convolution
- 3x3 max pooling
- 7x7 max pooling
- 3x3 convolution
- 5x5 depthwise-seperable conv

Methodology(contd)



Normal Cell *Reduction Cell*

Figure 4. Architecture of the best convolutional cells (NASNet-A) with $B = 5$ blocks identified with CIFAR-10 . The input (white) is the hidden state from previous activations (or input image). The output (pink) is the result of a concatenation operation across all resulting branches. Each convolutional cell is the result of B blocks. A single block is corresponds to two primitive operations (yellow) and a combination operation (green). Note that colors correspond to operations in Figure 3.

Experiments and Results

Experiments and Results

On CIFAR-10, with N = 4 or 6

model	depth	# params	error rate (%)
DenseNet ($L = 40, k = 12$) [26]	40	1.0M	5.24
DenseNet($L = 100, k = 12$) [26]	100	7.0M	4.10
DenseNet ($L = 100, k = 24$) [26]	100	27.2M	3.74
DenseNet-BC ($L = 100, k = 40$) [26]	190	25.6M	3.46
Shake-Shake 26 2x32d [18]	26	2.9M	3.55
Shake-Shake 26 2x96d [18]	26	26.2M	2.86
Shake-Shake 26 2x96d + cutout [12]	26	26.2M	2.56
NAS v3 [71]	39	7.1M	4.47
NAS v3 [71]	39	37.4M	3.65
NASNet-A (6 @ 768)	-	3.3M	3.41
NASNet-A (6 @ 768) + cutout	-	3.3M	2.65
NASNet-A (7 @ 2304)	-	27.6M	2.97
NASNet-A (7 @ 2304) + cutout	-	27.6M	2.40
NASNet-B (4 @ 1152)	-	2.6M	3.73
NASNet-C (4 @ 640)	-	3.1M	3.59

Table 1. Performance of Neural Architecture Search and other state-of-the-art models on CIFAR-10. All results for NASNet are the mean accuracy across 5 runs.

Experiments and Results

On ImageNet

Model	image size	# parameters	Mult-Adds	Top 1 Acc. (%)	Top 5 Acc. (%)
Inception V2 [29]	224×224	11.2 M	1.94 B	74.8	92.2
NASNet-A (5 @ 1538)	**299×299**	**10.9 M**	**2.35 B**	**78.6**	**94.2**
Inception V3 [60]	299×299	23.8 M	5.72 B	78.8	94.4
Xception [9]	299×299	22.8 M	8.38 B	79.0	94.5
Inception ResNet V2 [58]	299×299	55.8 M	13.2 B	80.1	95.1
NASNet-A (7 @ 1920)	**299×299**	**22.6 M**	**4.93 B**	**80.8**	**95.3**
ResNeXt-101 (64 x 4d) [68]	320×320	83.6 M	31.5 B	80.9	95.6
PolyNet [69]	331×331	92 M	34.7 B	81.3	95.8
DPN-131 [8]	320×320	79.5 M	32.0 B	81.5	95.8
SENet [25]	**320×320**	**145.8 M**	**42.3 B**	**82.7**	**96.2**
NASNet-A (6 @ 4032)	**331×331**	**88.9 M**	**23.8 B**	**82.7**	**96.2**

Table 2. Performance of architecture search and other published state-of-the-art models on ImageNet classification. Mult-Adds indicate the number of composite multiply-accumulate operations for a single image. Note that the composite multiple-accumulate operations are calculated for the image size reported in the table. Model size for [25] calculated from open-source implementation.

Experiments and Results

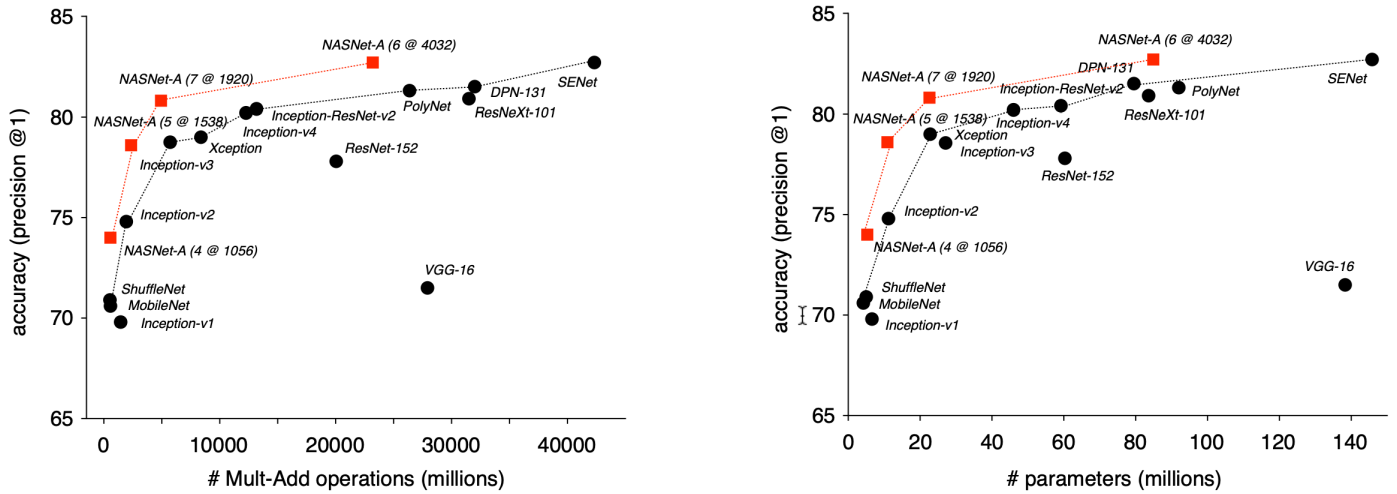


Figure 5. Accuracy versus computational demand (left) and number of parameters (right) across top performing published CNN architectures on ImageNet 2012 ILSVRC challenge prediction task. Computational demand is measured in the number of floating-point multiply-add operations to process a single image. Black circles indicate previously published results and red squares highlight our proposed models.

Conclusion

Conclusion

- Learning scalable, convolutional cells from data that transfer to multiple image classification tasks. The learned architecture is quite flexible as it may be scaled in terms of computational cost and parameters to easily address a variety of problems
- The key insight in the approach is to design a search space that decouples the complexity of an architecture from the depth of a network. This resulting search space permits identifying good architectures on a small dataset (i.e.,CIFAR-10) and transferring the learned architecture to image classifications across a range of data and computational scales
- Building an architecture on ImageNet is very computationally intensive with it taking 1800 GPU days (the equivalent of almost 5 years for 1 GPU) to learn the architecture (the team at Google used 500 GPUs for 4 days

QUESTIONS?

I can SEARCH for answers ;-)