Google

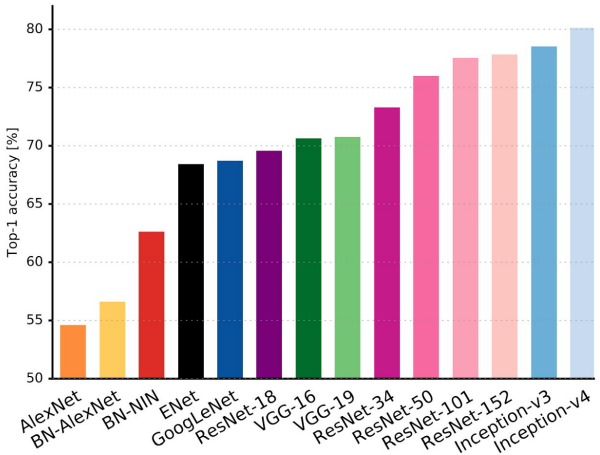
Neural Architecture Search

Quoc Le

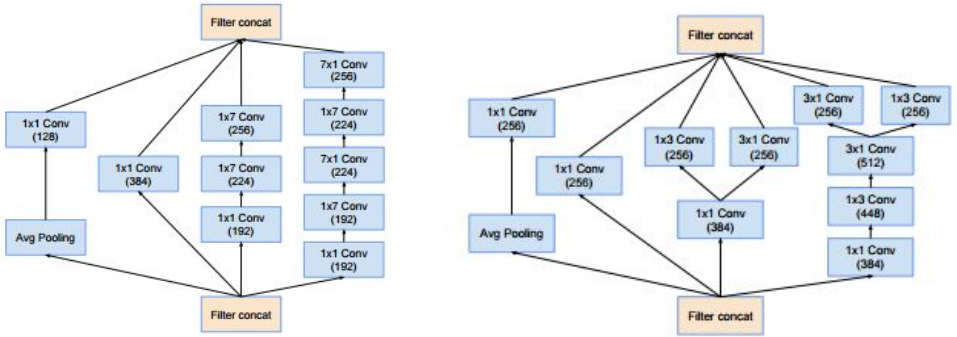
Thanks: Barret Zoph, Vijay Vasudevan, Irwan Bello, Jon Shlens and others
Google Brain team

Importance of architectures for Vision

- Designing neural network architectures is hard
- Can we try and learn good architectures automatically?



Canziani et al, 2017



Two layers from the famous Inception V4 computer vision model.

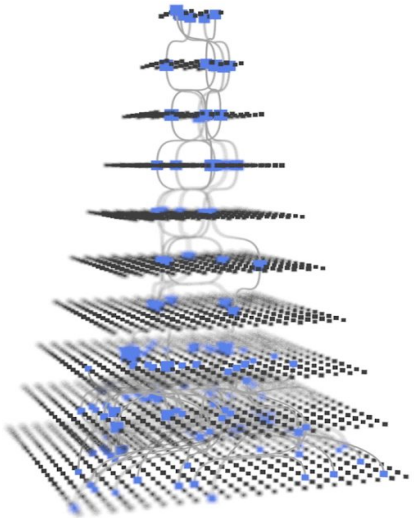
Szegedy et al, 2017

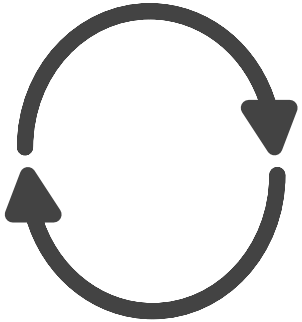
Google

Neural Architecture Search

- Key idea is that we can specify the structure and connectivity of a neural network by using a configuration string
  - [“Filter Width: 5”, “Filter Height: 3”, “Num Filters: 24”]
- Our idea is to use a RNN (“Controller”) to generate this string that specifies a neural network architecture
- Train this architecture (“Child Network”) to see how well it performs on a validation set
- Use reinforcement learning to update the parameters of the Controller model based on the accuracy of the child model

Controller: proposes ML models

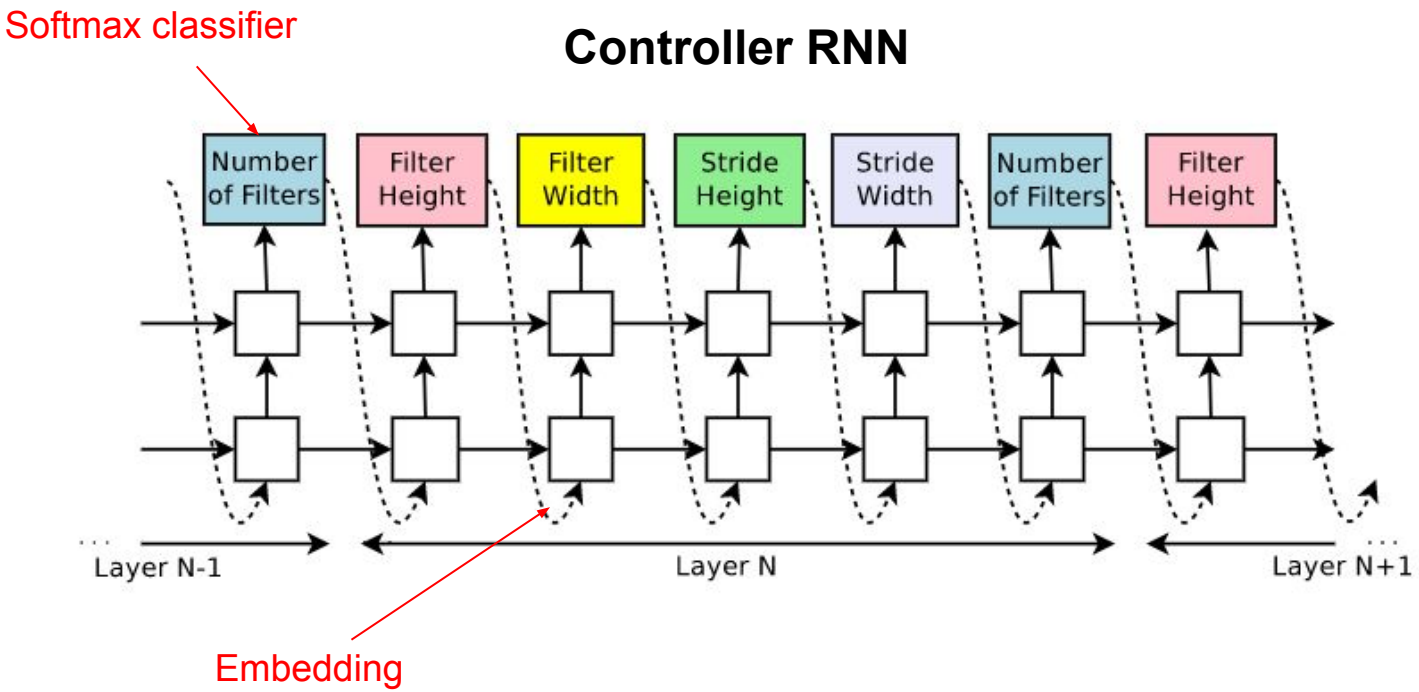




Iterate to find the most accurate model

Train & evaluate models

Google


Softmax classifier
Controller RNN
Number of Filters
Filter Height
Filter Width
Stride Height
Stride Width
Number of Filters
Filter Height
Layer N-1
Layer N
Layer N+1
Embedding

Training with REINFORCE

Parameters of Controller RNN

Accuracy of architecture on held-out dataset

$$J(\theta_c) = E_{P(a_{1:T};\theta_c)}[R]$$

Architecture predicted by the controller RNN viewed as a sequence of actions

$$\nabla_{\theta_c} J(\theta_c) = \sum_{t=1}^{T} E_{P(a_{1:T};\theta_c)}\big[\nabla_{\theta_c} \log P(a_t | a_{(t-1):1}; \theta_c) R\big]$$

Number of models in minibatch

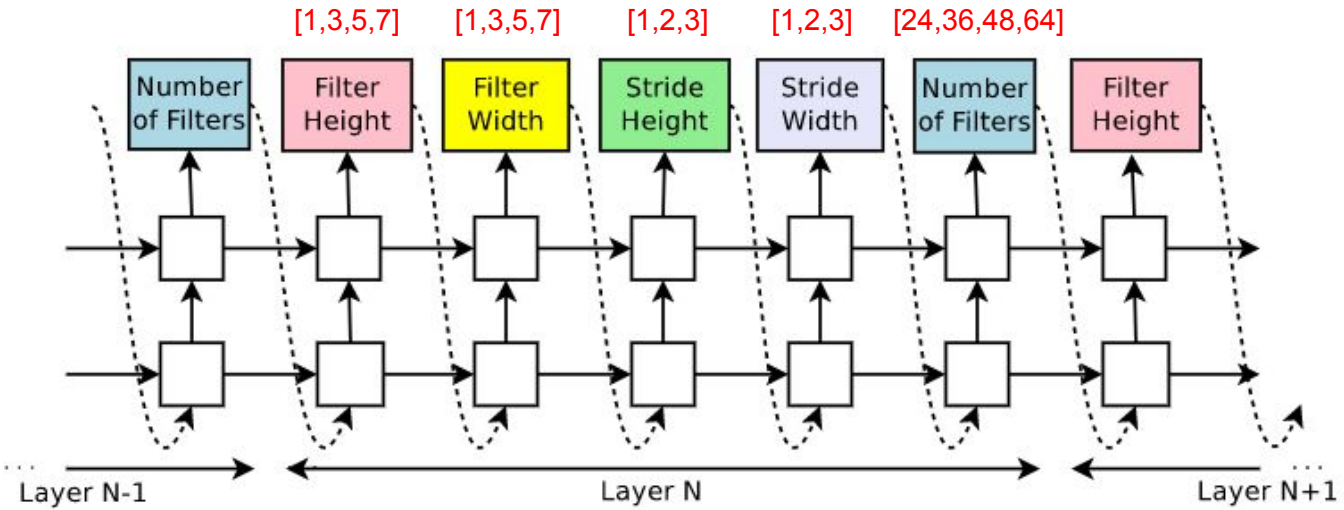
$$\frac{1}{m} \sum_{k=1}^{m} \sum_{t=1}^{T} \nabla_{\theta_c} \log P(a_t | a_{(t-1):1}; \theta_c) R_k$$

Google

CIFAR-10 Experiments

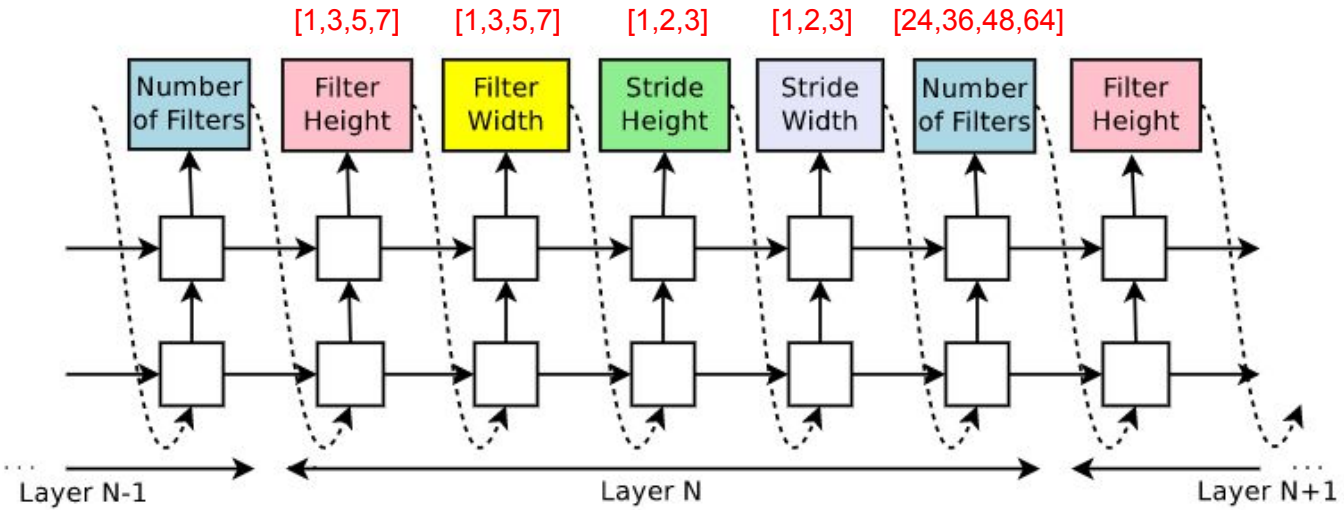
- We apply Neural Architecture Search to predicting convolutional networks on CIFAR-10
- Predict the following for a fixed number of layers (15, 20, 13):
  - Filter width/height
  - Stride width/height
  - Number of filters

Neural Architecture Search for CIFAR-10



Google

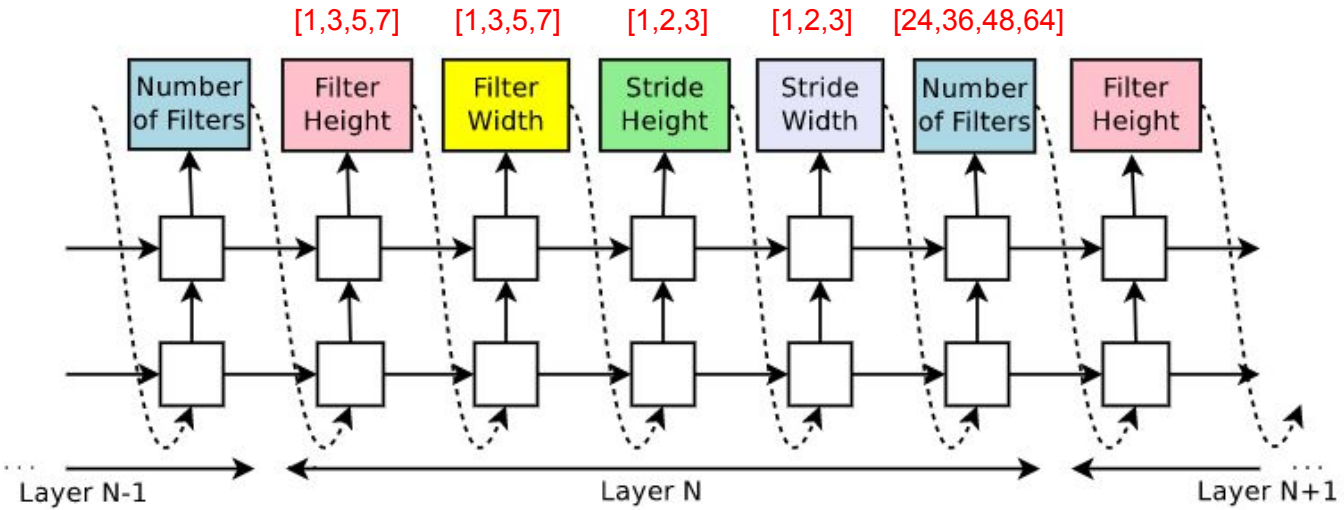
Neural Architecture Search for CIFAR-10



3 ; 7 ; 1 ; 2 ; 36

Google

Neural Architecture Search for CIFAR-10



3 ; 7 ; 1 ; 2 ; 36

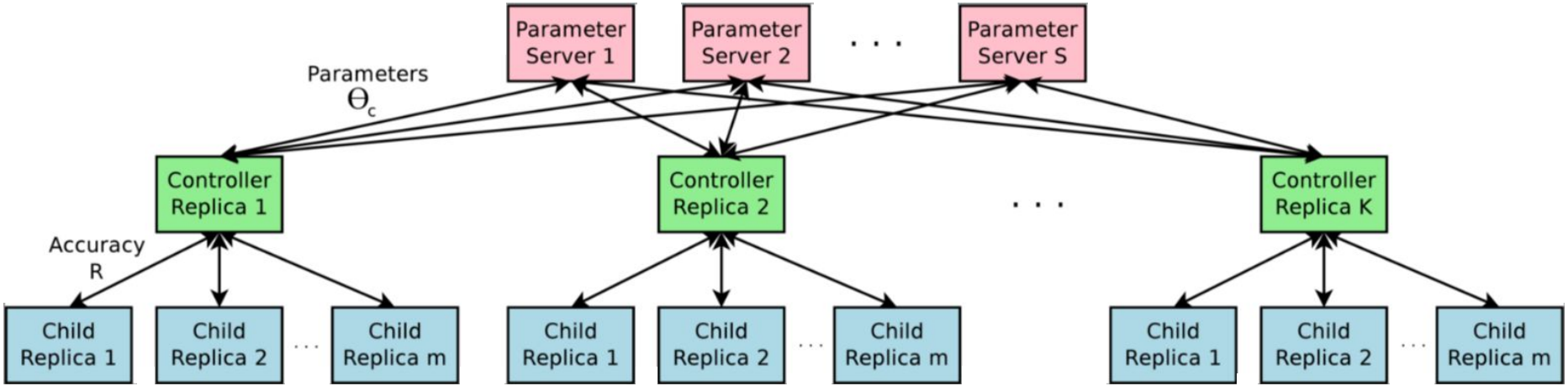
Filter Height Filter Width Stride Height Stride Width Number of Filters

Google

CIFAR-10 Experiment Details

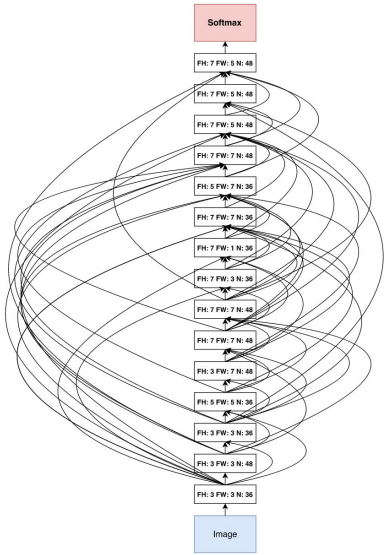
- Method uses 800 GPUs concurrently at one time
- Reward given to the Controller is the maximum validation accuracy
- Each child model was trained for 50 epochs
- Run for a total of **12,800** child models

Distributed Training



Google

Neural Architecture Search for 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

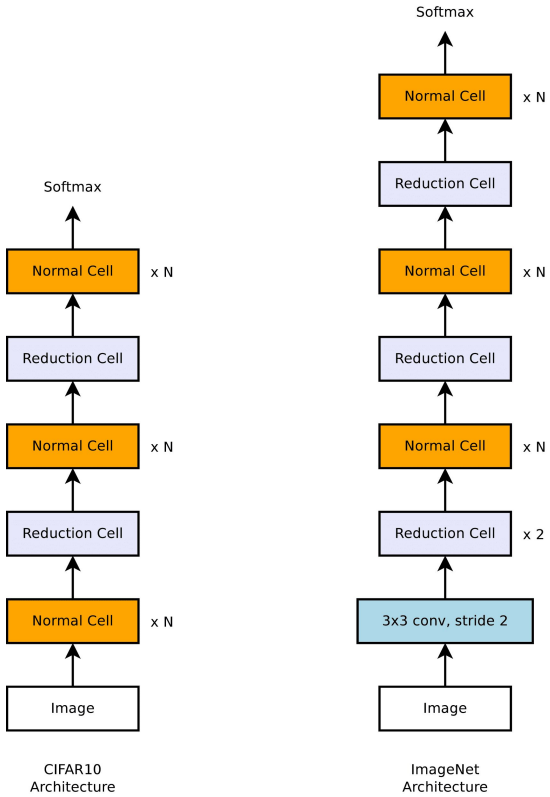
Best result of evolution (Real et al, 2017): 5.4%
Best result of Q-learning (Baker et al, 2017): 6.92%

Google

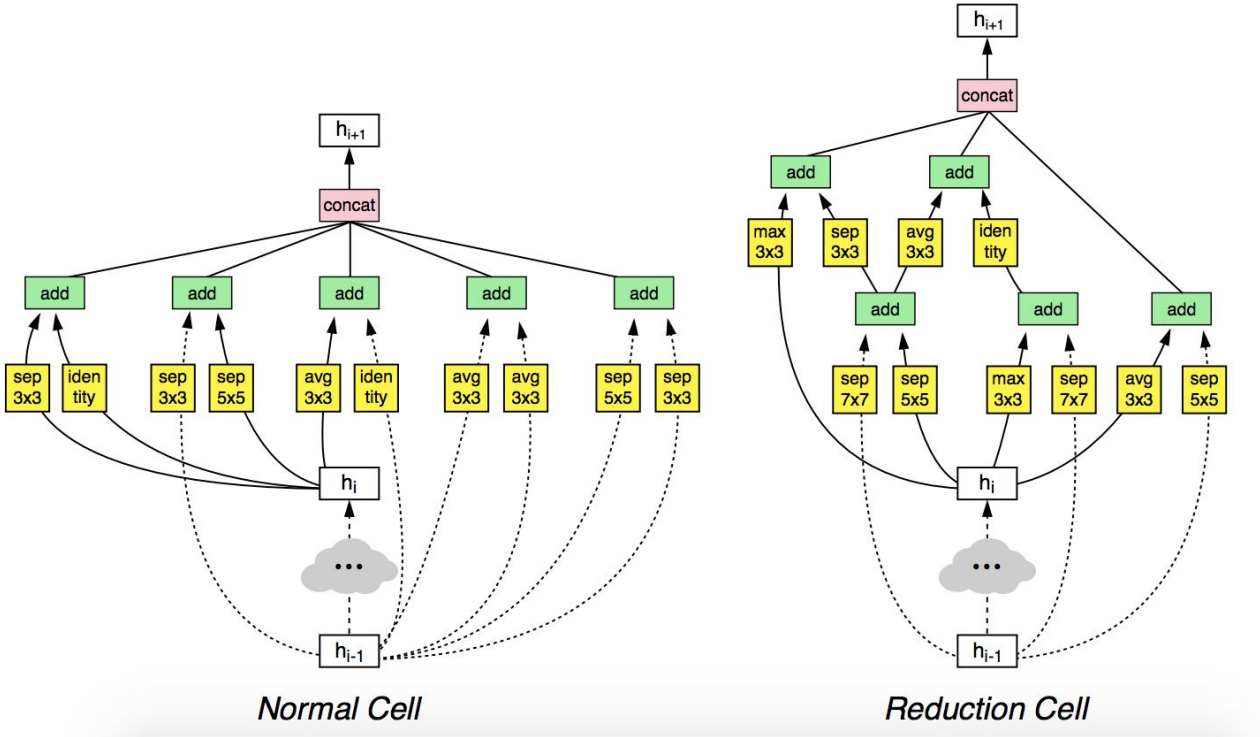
Neural Architecture Search for ImageNet

- Neural Architecture Search directly on ImageNet is expensive
- Key idea is to run Neural Architecture Search on CIFAR-10 to find a “cell”
- Construct a bigger net from the “cell” and train the net on ImageNet

Neural Architecture Search for ImageNet



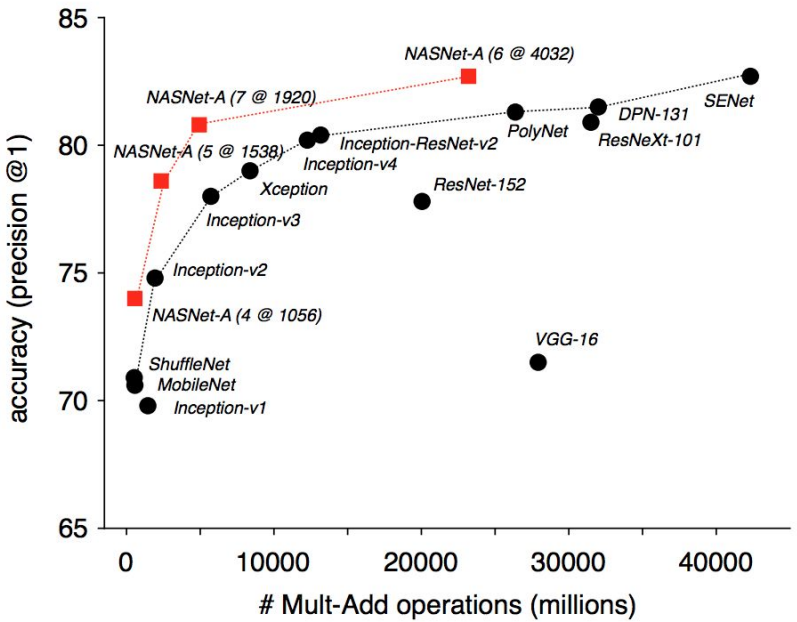
Neural Architecture Search for ImageNet

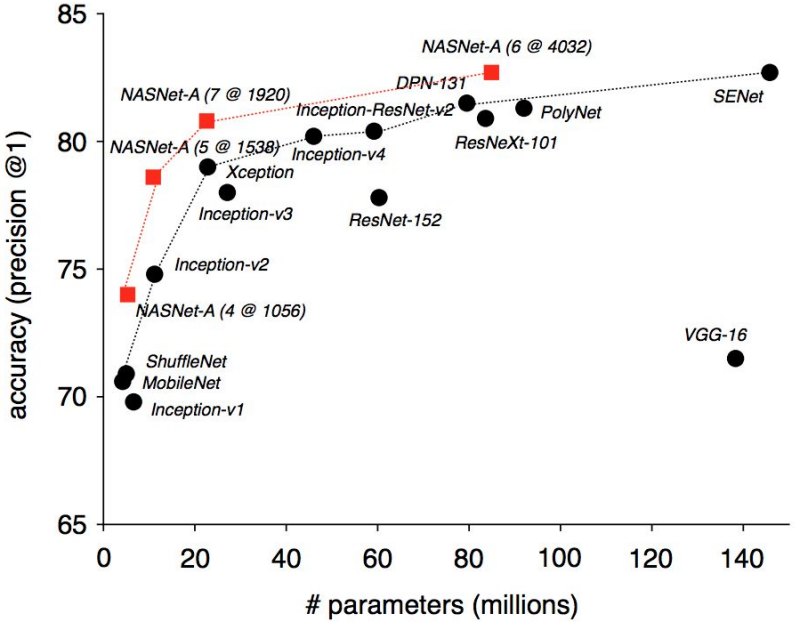


Performance of cell on CIFAR-10

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 [70]	39	7.1M	4.47
NAS v3 [70]	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

Performance of cell on ImageNet

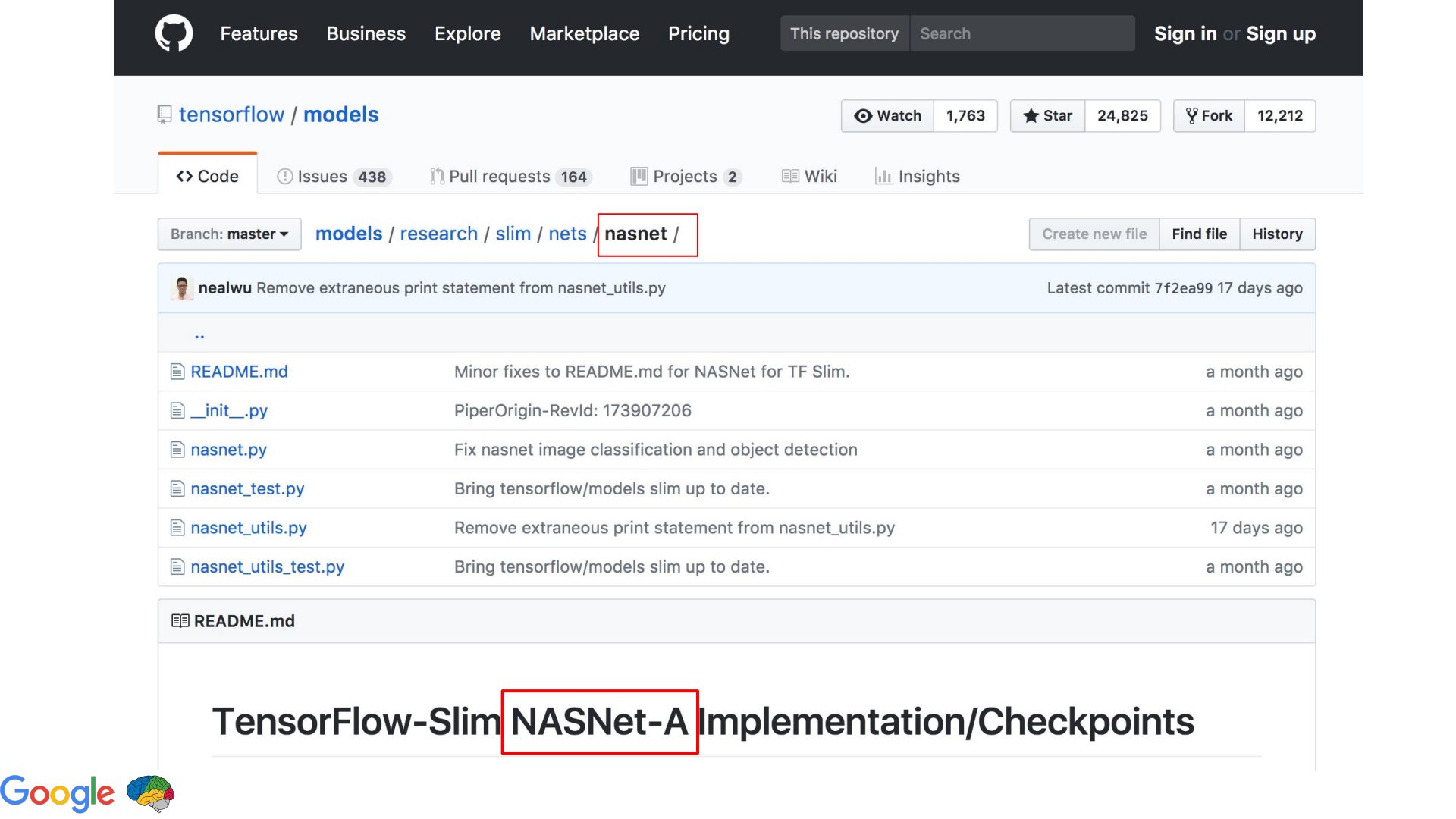




Google

Performance of cell on COCO (object detection)

Model	resolution	mAP (mini-val)	mAP (test-dev)
MobileNet-224 [24]	600 × 600	19.8%	-
ShuffleNet (2x) [69]	600 × 600	24.5%†	-
NASNet-A (4 @ 1056)	600 × 600	**29.6%**	-
ResNet-101-FPN [35]	800 (short side)	-	36.2%
Inception-ResNet-v2 (G-RMI) [28]	600 × 600	35.7%	35.6%
Inception-ResNet-v2 (TDM) [51]	600 × 1000	37.3%	36.8%
NASNet-A (6 @ 4032)	800 × 800	41.3%	40.7%
NASNet-A (6 @ 4032)	1200 × 1200	**43.2%**	**43.1%**
ResNet-101-FPN (RetinaNet) [36]	800 (short side)	-	39.1%


Features Business Explore Marketplace Pricing
This repository Search
Sign in or Sign up
tensorflow / models
Watch 1,763
Star 24,825
Fork 12,212
Code Issues 438 Pull requests 164 Projects 2 Wiki Insights
Branch: master
models / research / slim / nets / nasnet /
Create new file Find file History
nealwu Remove extraneous print statement from nasnet_utils.py
Latest commit 7f2ea99 17 days ago
..
README.md Minor fixes to README.md for NASNet for TF Slim. a month ago
__init__.py PiperOrigin-RevId: 173907206 a month ago
nasnet.py Fix nasnet image classification and object detection a month ago
nasnet_test.py Bring tensorflow/models slim up to date. a month ago
nasnet_utils.py Remove extraneous print statement from nasnet_utils.py 17 days ago
nasnet_utils_test.py Bring tensorflow/models slim up to date. a month ago
README.md
TensorFlow-Slim NASNet-A Implementation/Checkpoints