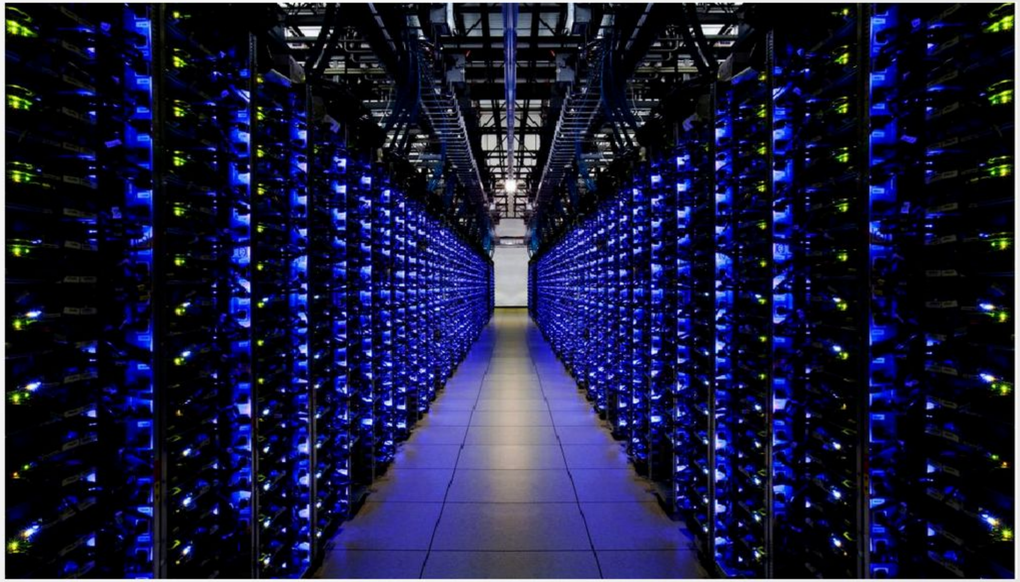
Distributed System In Deep Learning



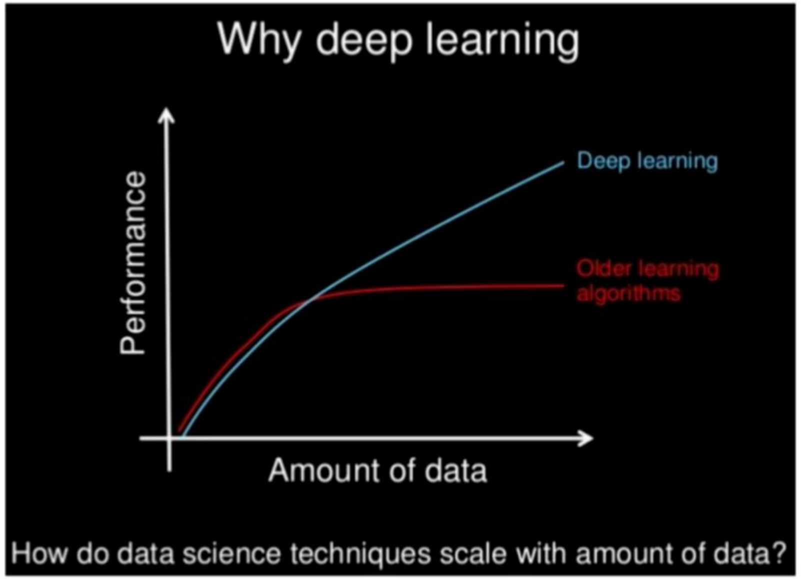
ashubly25

ashubly25

ashubly25

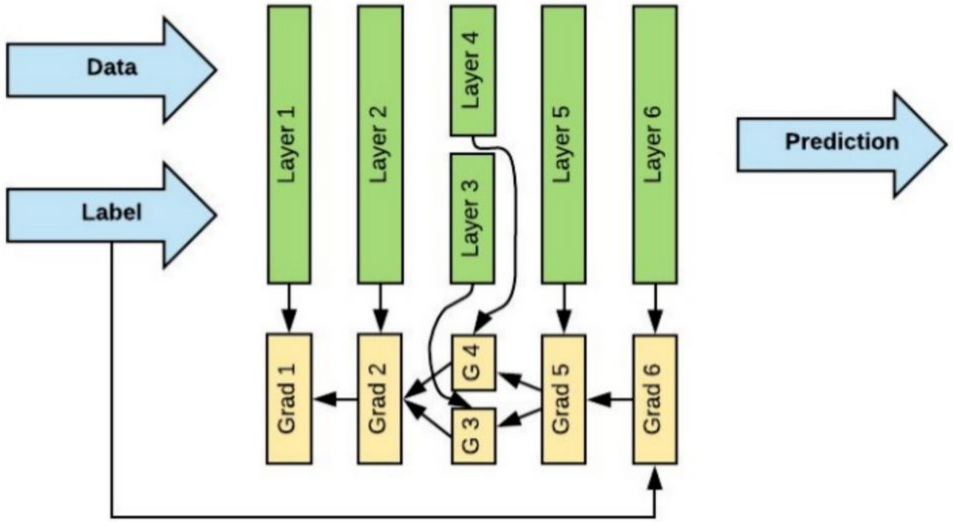
Ashutosh Singh

Deep Learning



Credit: Andrew Ng, https://www.slideshare.net/ExtractConf

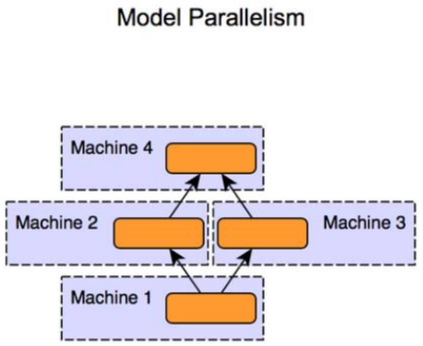
How does Deep Learning training work?

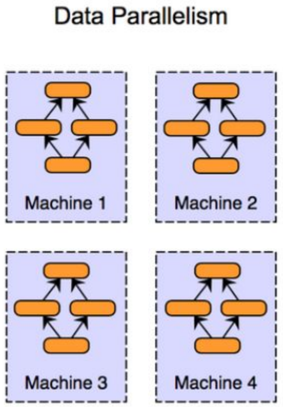


"All you need is lots and lots of data and lots of information about what the right answer is, and you'll be able to train a big neural net to do what you want."

Geoffrey Hinton

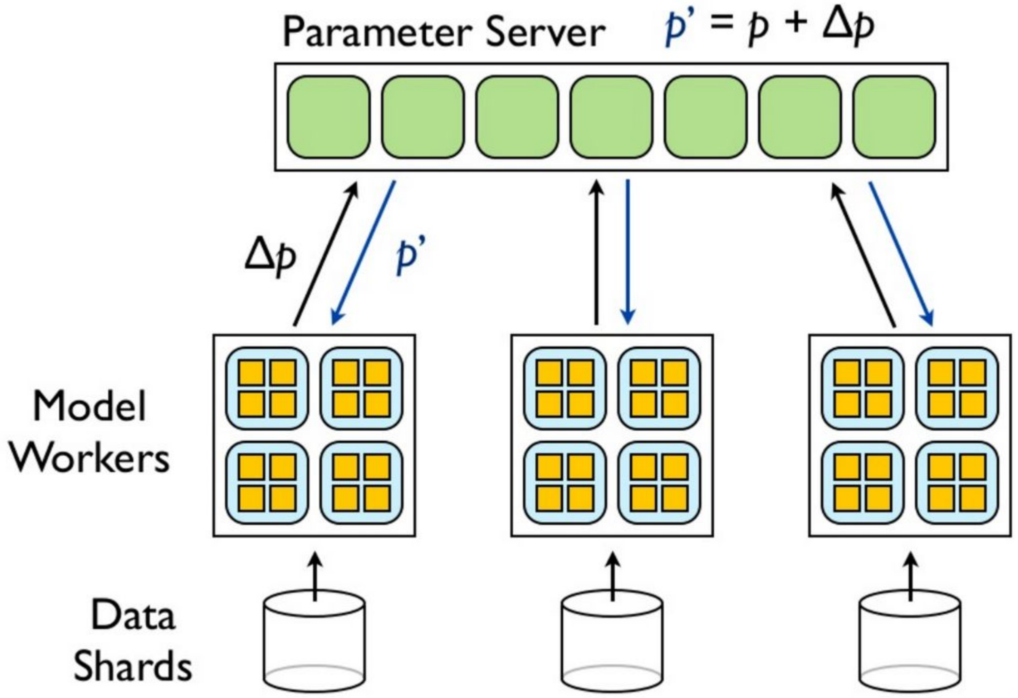
Scalable Computation



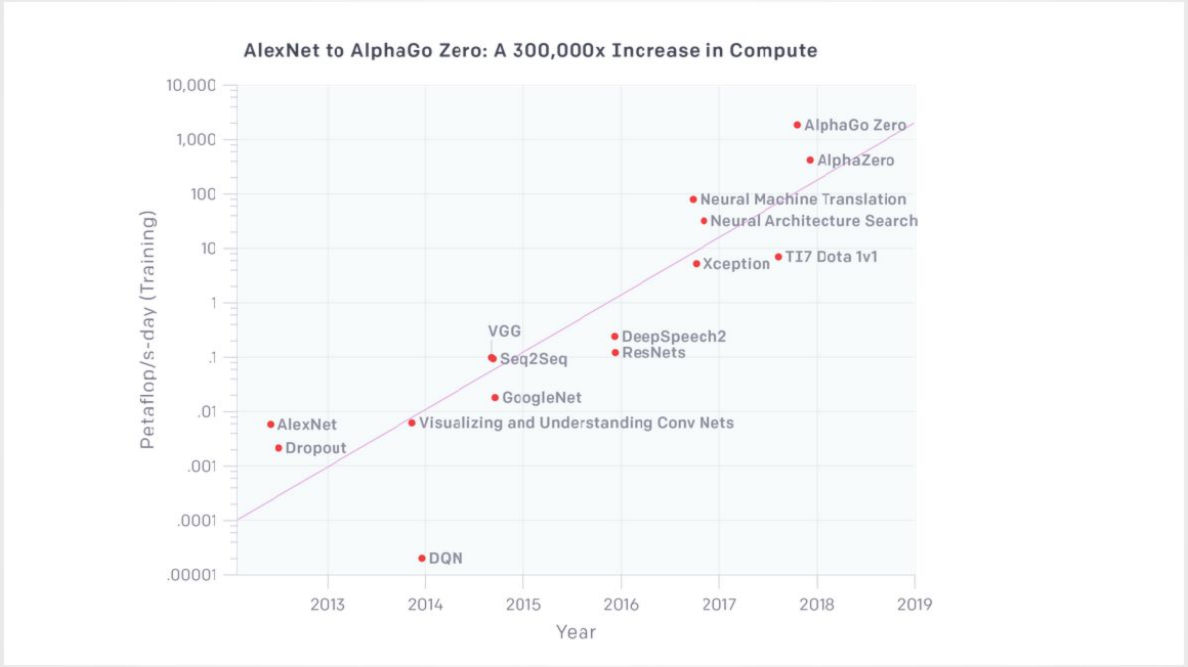


Data Parallelism:

Asynchronous Distributed Stochastic Gradient Descent

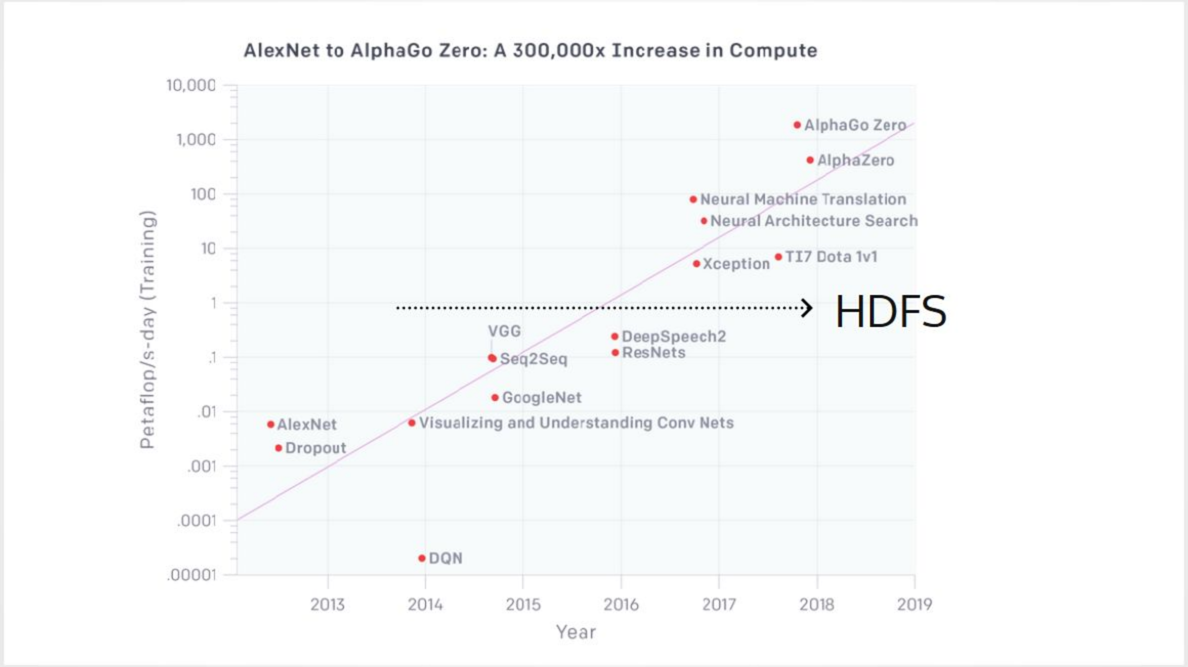


Computation



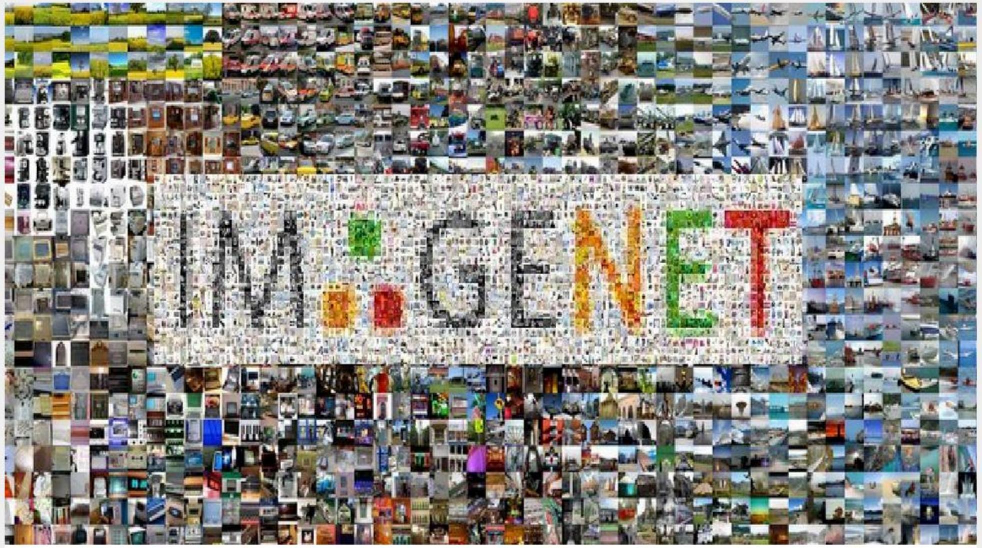
https://blog.openai.com/ai-and-compute/

Use of distributed system



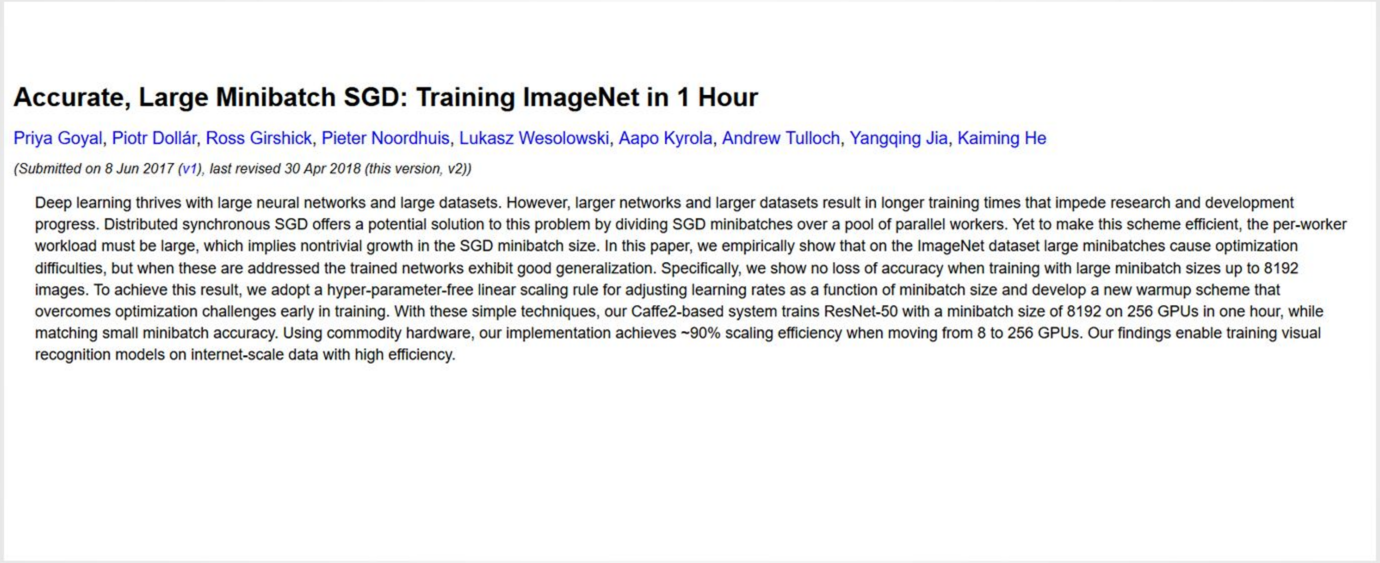
https://blog.openai.com/ai-and-compute/

Imagenet



"The **ImageNet** project is a large visual database designed for use in visual object recognition software research" -Wikipedia

Team Facebook



Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, Kaiming He

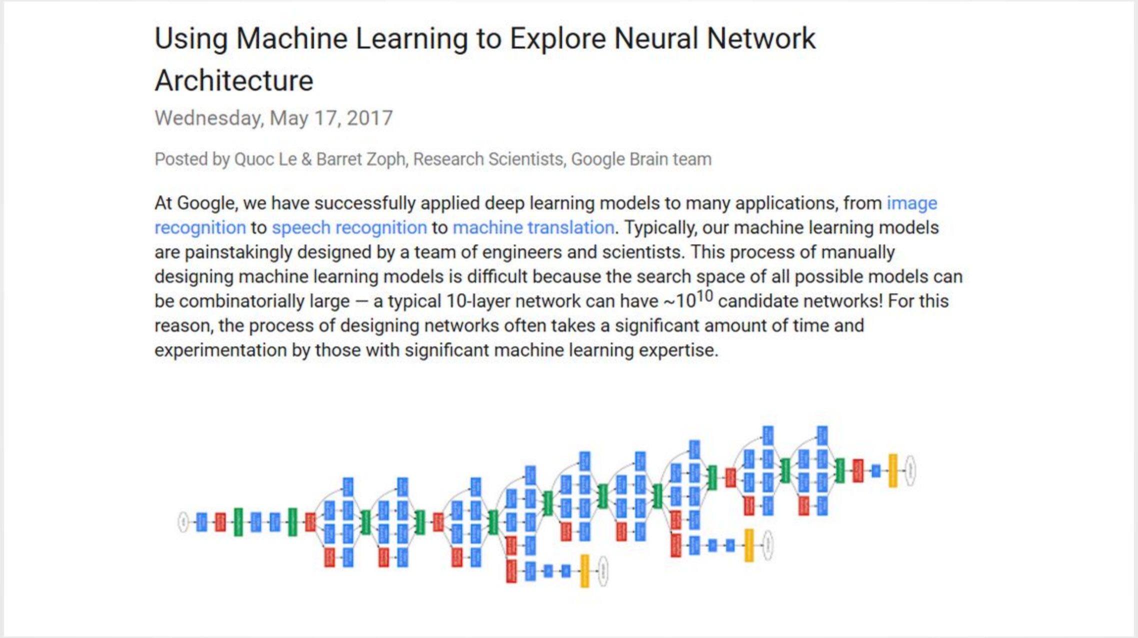
(Submitted on 8 Jun 2017 (v1), last revised 30 Apr 2018 (this version, v2))

Deep learning thrives with large neural networks and large datasets. However, larger networks and larger datasets result in longer training times that impede research and development progress. Distributed synchronous SGD offers a potential solution to this problem by dividing SGD minibatches over a pool of parallel workers. Yet to make this scheme efficient, the per-worker workload must be large, which implies nontrivial growth in the SGD minibatch size. In this paper, we empirically show that on the ImageNet dataset large minibatches cause optimization difficulties, but when these are addressed the trained networks exhibit good generalization. Specifically, we show no loss of accuracy when training with large minibatch sizes up to 8192 images. To achieve this result, we adopt a hyper-parameter-free linear scaling rule for adjusting learning rates as a function of minibatch size and develop a new warmup scheme that overcomes optimization challenges early in training. With these simple techniques, our Caffe2-based system trains ResNet-50 with a minibatch size of 8192 on 256 GPUs in one hour, while matching small minibatch accuracy. Using commodity hardware, our implementation achieves ~90% scaling efficiency when moving from 8 to 256 GPUs. Our findings enable training visual recognition models on internet-scale data with high efficiency.

Training Resnet-50 on Imagenet

Team Google

Auto ML with RL



Using Machine Learning to Explore Neural Network Architecture

Wednesday, May 17, 2017

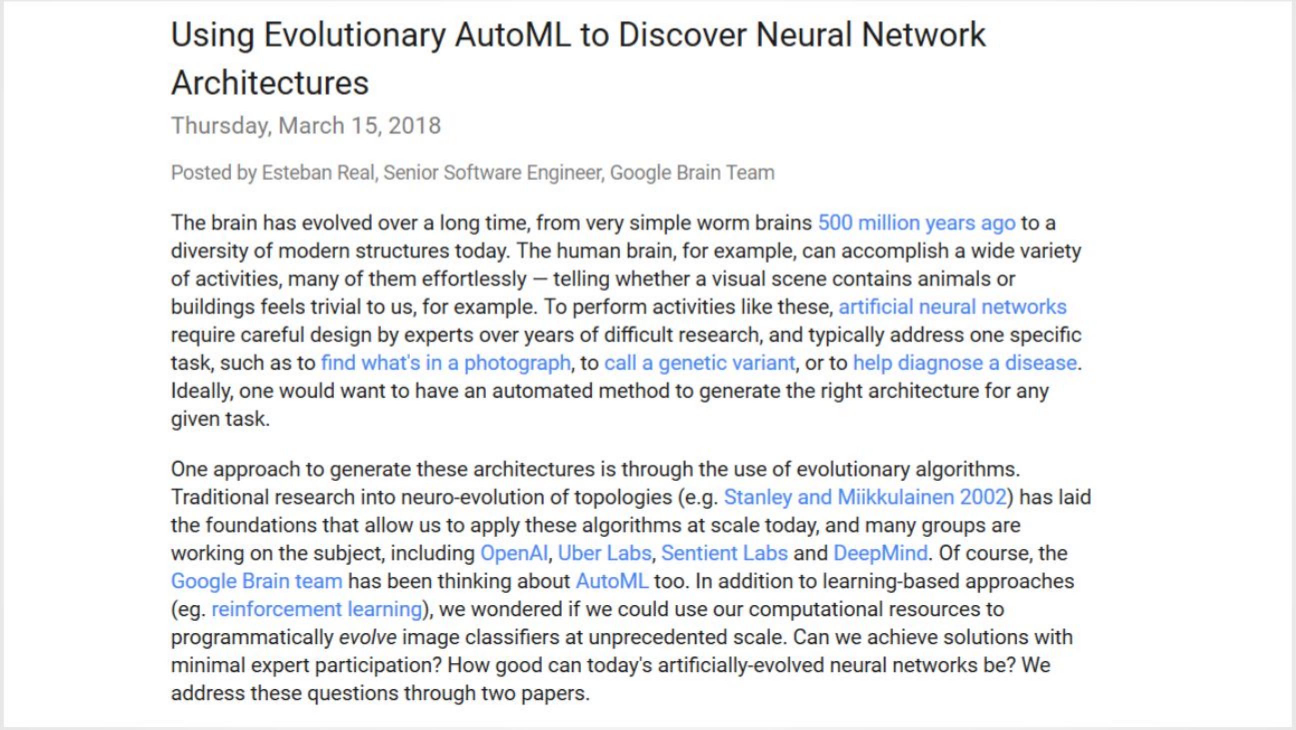
Posted by Quoc Le & Barret Zoph, Research Scientists, Google Brain team

At Google, we have successfully applied deep learning models to many applications, from image recognition to speech recognition to machine translation. Typically, our machine learning models are painstakingly designed by a team of engineers and scientists. This process of manually designing machine learning models is difficult because the search space of all possible models can be combinatorially large — a typical 10-layer network can have ~10^{10} candidate networks! For this reason, the process of designing networks often takes a significant amount of time and experimentation by those with significant machine learning expertise.

https://ai.googleblog.com/2017/05/

Team Google

Auto ML with Genetic Algorithm



Using Evolutionary AutoML to Discover Neural Network Architectures

Thursday, March 15, 2018

Posted by Esteban Real, Senior Software Engineer, Google Brain Team

The brain has evolved over a long time, from very simple worm brains 500 million years ago to a diversity of modern structures today. The human brain, for example, can accomplish a wide variety of activities, many of them effortlessly — telling whether a visual scene contains animals or buildings feels trivial to us, for example. To perform activities like these, artificial neural networks require careful design by experts over years of difficult research, and typically address one specific task, such as to find what's in a photograph, to call a genetic variant, or to help diagnose a disease. Ideally, one would want to have an automated method to generate the right architecture for any given task.

One approach to generate these architectures is through the use of evolutionary algorithms. Traditional research into neuro-evolution of topologies (e.g. Stanley and Miikkulainen 2002) has laid the foundations that allow us to apply these algorithms at scale today, and many groups are working on the subject, including OpenAI, Uber Labs, Sentient Labs and DeepMind. Of course, the Google Brain team has been thinking about AutoML too. In addition to learning-based approaches (eg. reinforcement learning), we wondered if we could use our computational resources to programmatically *evolve* image classifiers at unprecedented scale. Can we achieve solutions with minimal expert participation? How good can today's artificially-evolved neural networks be? We address these questions through two papers.

https://ai.googleblog.com/2017/03/

Architecture that exceeds imagenet and cifar-10 models

Learning Transferable Architectures for Scalable Image Recognition

Barret Zoph
Google Brain
barretzoph@google.com

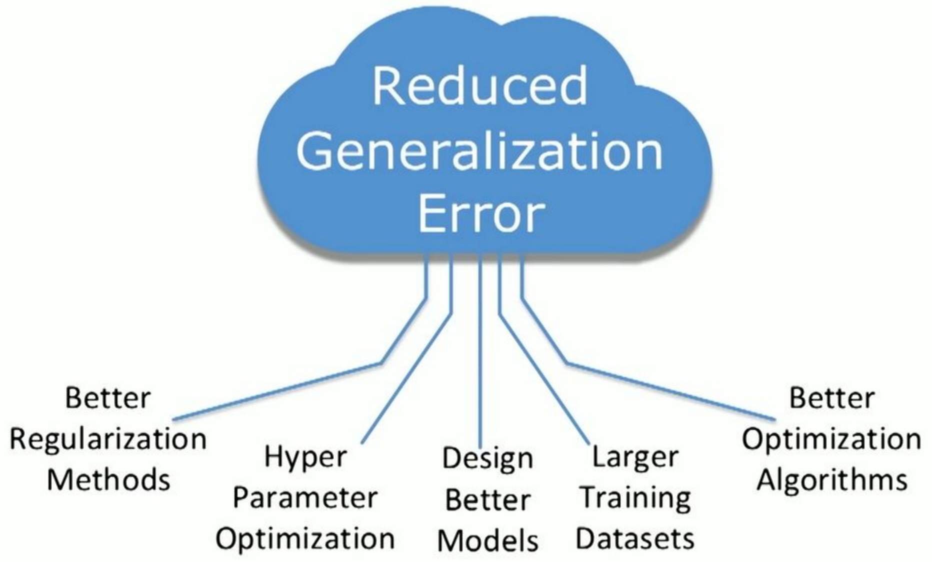
Vijay Vasudevan
Google Brain
vrv@google.com

Jonathon Shlens
Google Brain
shlens@google.com

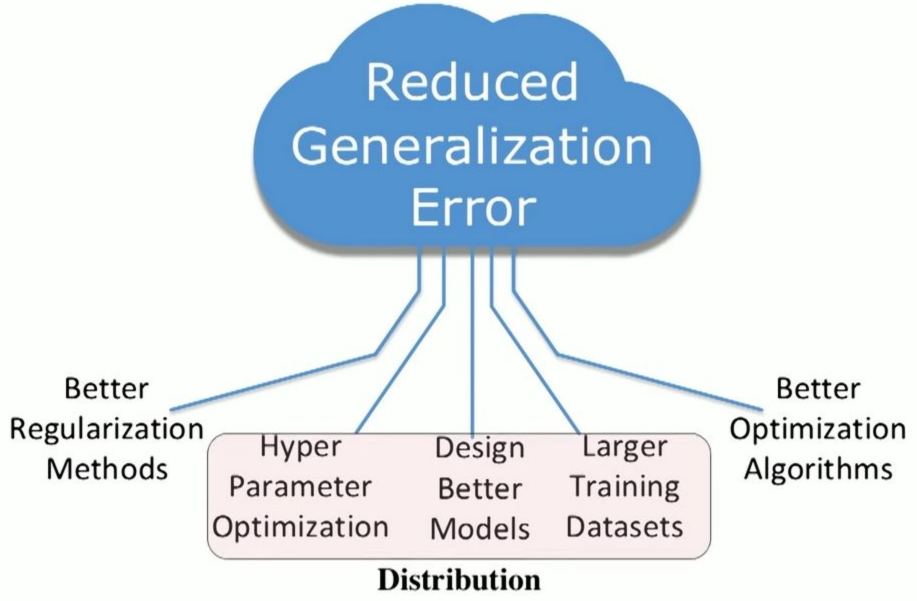
Quoc V. Le
Google Brain
qvl@google.com

https://arxiv.org/pdf/1707.07012.pdf

Methods for Improving Model Accuracy


Reduced Generalization Error
Better Regularization Methods
Hyper Parameter Optimization
Design Better Models
Larger Training Datasets
Better Optimization Algorithms

Methods for Improving Model Accuracy


Reduced Generalization Error
Better Regularization Methods
Hyper Parameter Optimization
Design Better Models
Larger Training Datasets
Better Optimization Algorithms
Distribution

Advantages of Distributed Deep Learning over conventional Deep Learning System

- *Minimize training time*
- *Maximize Batch Size*

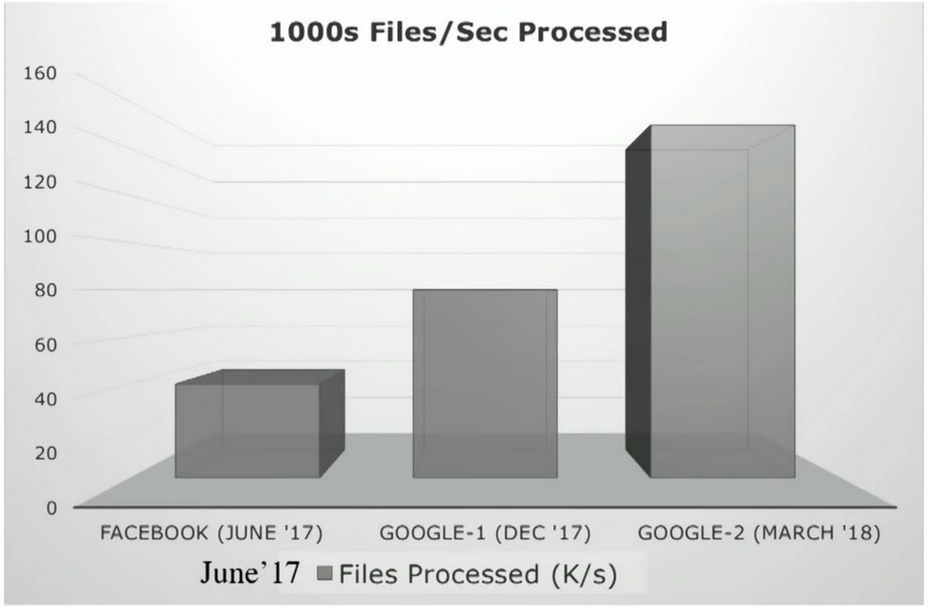
Improvements

- Facebook - 60 Minutes(June 2017)
- Google - 30 minutes(May 2017)
- Google - 10 minutes(March 2018)

Improvements – Number of GPU/TPU

- Facebook - 60 Minutes(June 2017) (256 GPU Nvidia P100s)
- Google - 30 minutes(May 2017) (256 TPU v2
- Google - 10 minutes(March 2018)

ImageNet – Files/Sec Processed



FB - https://goo.gl/ERpJyr

Google-1: https://goo.gl/EV7Xv1

Google-2:https://goo.gl/eidnyQ

GridSearch with TensorFlow/Spark

```
def train(learning_rate, dropout):

 [TensorFlow Code here]

args_dict = {'learning_rate': [0.001, 0.005, 0.01],
          'dropout': [0.5, 0.6]}

experiment.launch(spark, train, args_dict)
```

Launch 6 Spark Executors

Model Architecture Search on TensorFlow

```
def train_cifar10(learning_rate, dropout, num_layers):

 [TensorFlow Code here]

dict = {'learning_rate': [0.005, 0.00005],
'dropout': [0.01, 0.99], 'num_layers': [1,3]}

experiment.evolutionary_search(spark, train, dict, direction='max', popsize=10,
generations=3, crossover=0.7,  mutation=0.5)
```

Over the cloud training

Amazon SageMaker

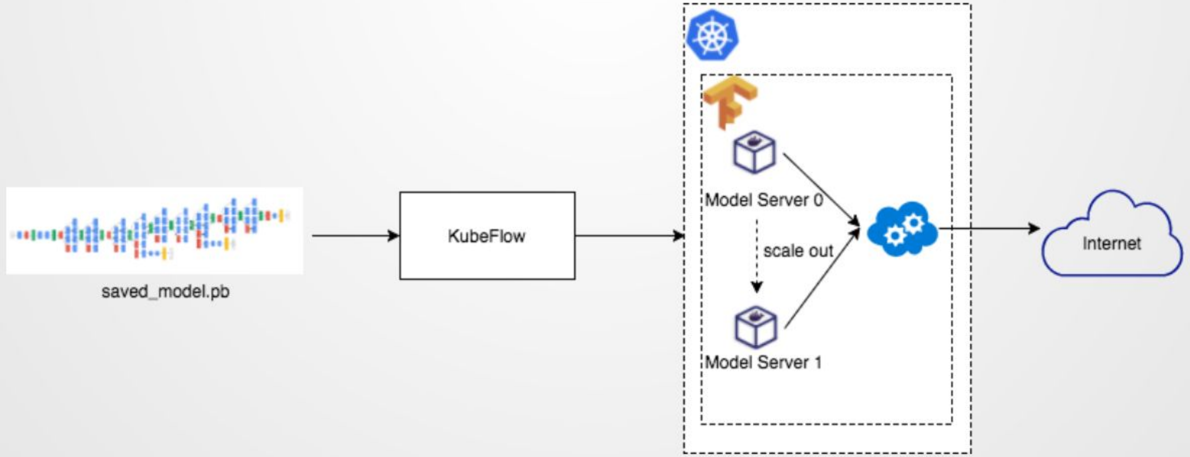
FLOYDHUB

The fastest way to build, train, and deploy deep learning models

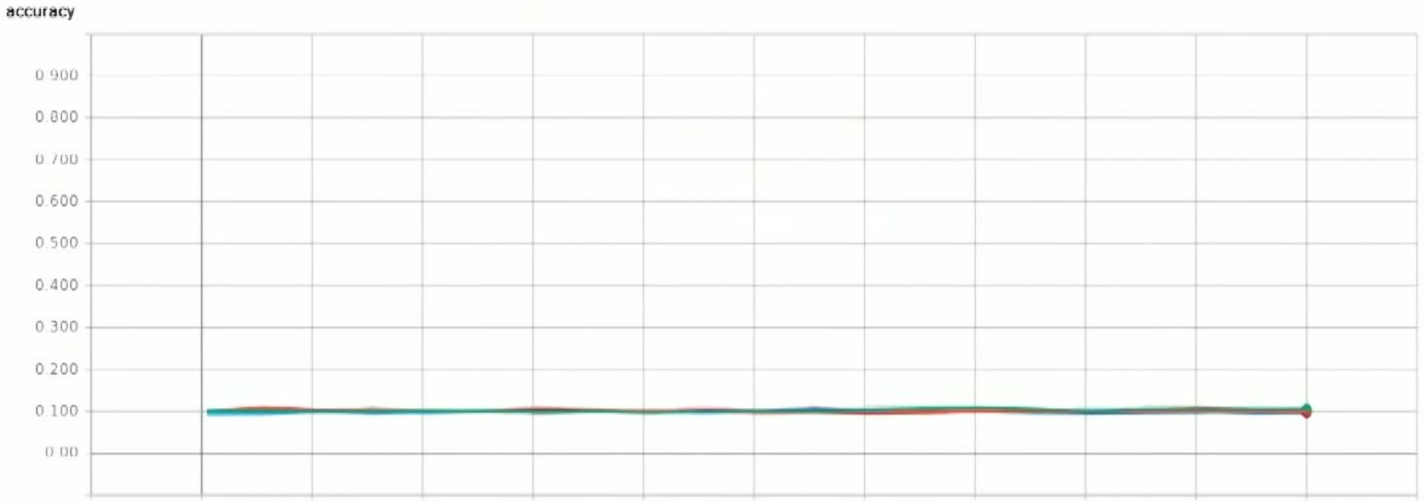
PyTorch jupyter Caffe learn

On premise

Kubeflow

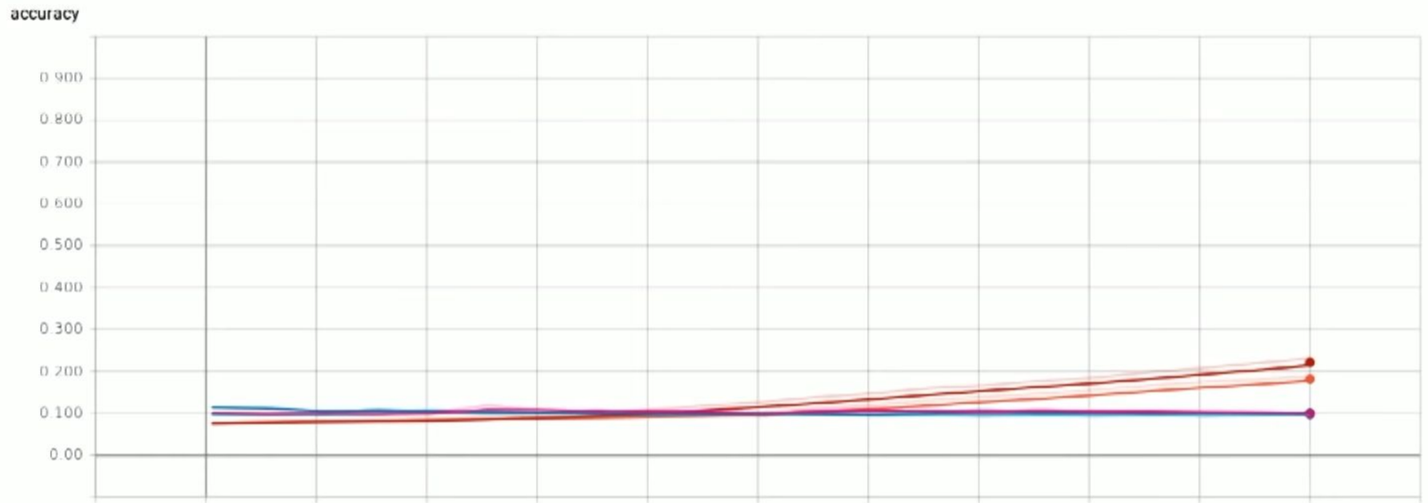


Differential Evolution in Tensorboard



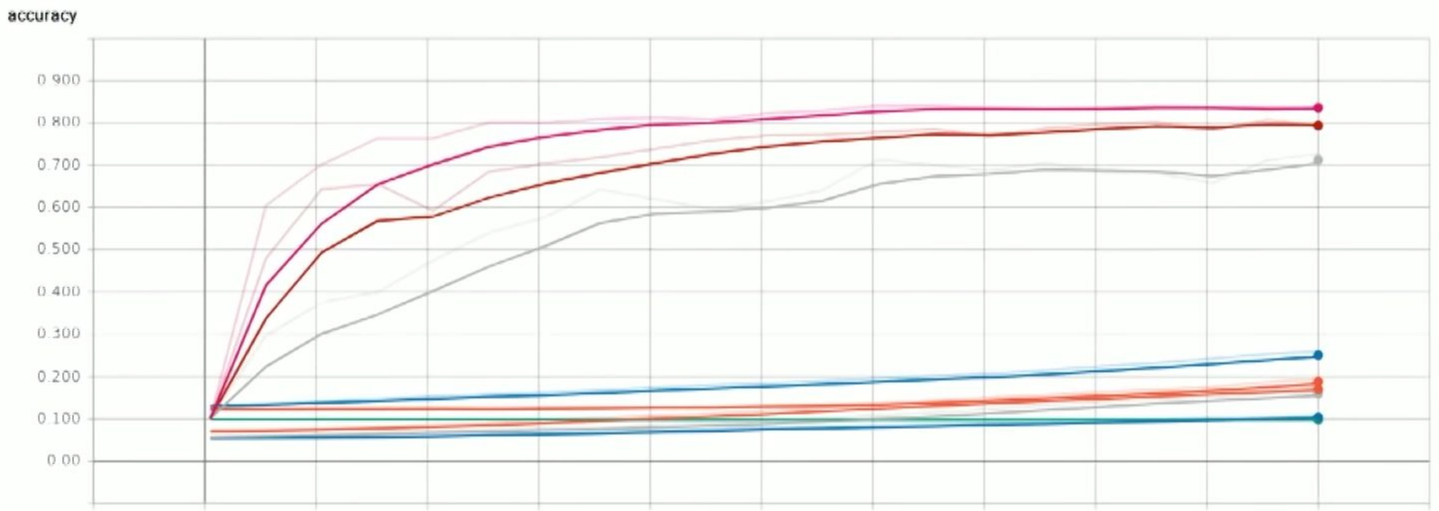
CIFAR-10

Differential Evolution in Tensorboard



CIFAR-10

Differential Evolution in Tensorboard



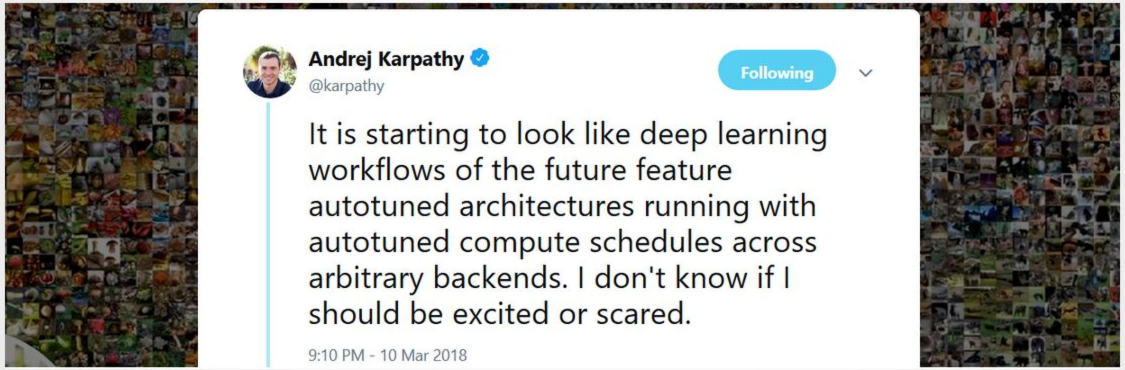
CIFAR-10

Cons of Distributed System

- Reading Hadoop Distributed File System in Python
- Mapreduce
- Debugging at such a scale
- Most of the code we write are in Docker files or YAML
- Adding new server is clunky
- Modification of a single file have to be done over all the shared machine to do those changes.
- Unfamilarity

Conclusion

- The future is distribution(https://www.oreilly.com/ideas/distributed-tensorflow)



Andrej Karpathy
@karpathy

It is starting to look like deep learning workflows of the future feature autotuned architectures running with autotuned compute schedules across arbitrary backends. I don't know if I should be excited or scared.

9:10 PM - 10 Mar 2018

Further Readings

- https://medium.com/@Petuum/intro-to-distributed-deep-learning-systems-a2e45c6b8e7
- https://eng.uber.com/horovod/
- https://www.matroid.com/scaledml/2018/jeff.pdf
- http://jenaiz.com/2015/06/deep-learning-in-a-large-scale-distributed-systems/
- https://blog.skymind.ai/distributed-deep-learning-part-1-an-introduction-to-distributed-training-of-neural-networks/
- https://bair.berkeley.edu/blog/2018/01/09/ray/