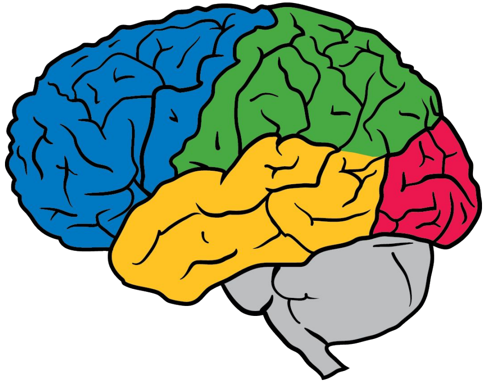


Large-Scale Deep Learning With TensorFlow

Jeff Dean
Google Brain team
g.co/brain

In collaboration with **many** other people at Google

What is the Google Brain Team?

- Research team focused on long term artificial intelligence research
    - Mix of computer systems and machine learning research expertise
    - Pure ML research, and research in context of emerging ML application areas:
        - robotics, language understanding, healthcare, ...

g.co/brain

We Disseminate Our Work in Many Ways

- By publishing our work
  - See papers at research.google.com/pubs/BrainTeam.html
- By releasing TensorFlow, our core machine learning research system, as an open-source project
- By releasing implementations of our research models in TensorFlow
- By collaborating with product teams at Google to get our research into real products

What Do We Really Want?

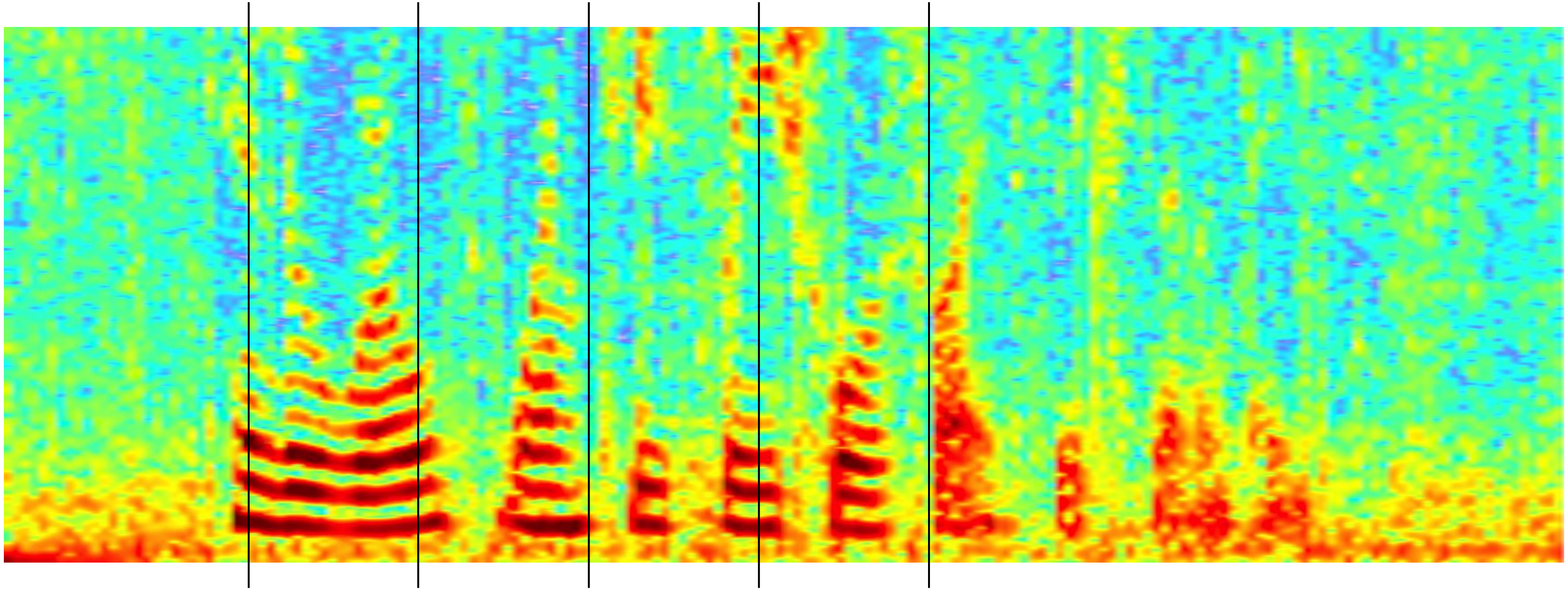
- Build artificial intelligence algorithms and systems that learn from experience

- Use those to solve difficult problems that benefit humanity

What do I mean by understanding?

What do I mean by understanding?

What do I mean by understanding?



What do I mean by understanding?

Query

[car parts for sale]

What do I mean by understanding?

Query

[car parts for sale]

Document 1

… car parking available for a small fee.
… parts of our floor model inventory for sale.

Document 2

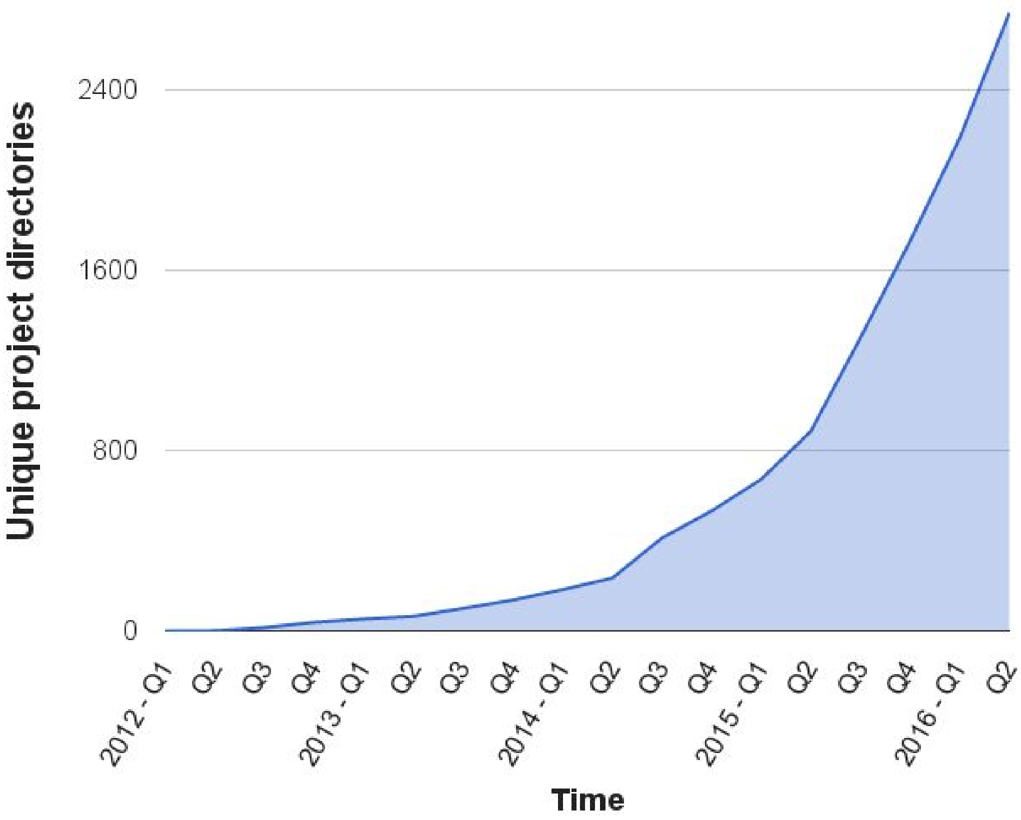
Selling all kinds of automobile and pickup truck parts, engines, and transmissions.

Example Needs of the Future

- *Which of these eye images shows symptoms of diabetic retinopathy?*
- *Find me all rooftops in North America*
- *Describe this video in Spanish*
- *Find me all documents relevant to reinforcement learning for robotics and summarize them in German*
- *Find a free time for everyone in the Smart Calendar project to meet and set up a videoconference*
- *Robot, please fetch me a cup of tea from the snack kitchen*

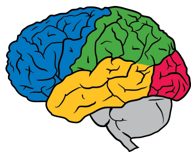
Growing Use of Deep Learning at Google

of directories containing model description files



Across many products/areas:

Android
Apps
drug discovery
Gmail
Image understanding
Maps
Natural language understanding
Photos
Robotics research
Speech
Translation
YouTube
… many others ...



Important Property of Neural Networks

Results get better with

more data +
bigger models +
more computation

(Better algorithms, new insights and improved techniques always help, too!)

Aside

Many of the techniques that are successful now were developed 20-30 years ago

What changed? We now have:

sufficient computational resources
large enough interesting datasets

Use of large-scale parallelism lets us look ahead many generations of hardware improvements, as well

What do you want in a machine learning system?

- **Ease of expression**: for lots of crazy ML ideas/algorithms
- **Scalability**: can run experiments quickly
- **Portability**: can run on wide variety of platforms
- **Reproducibility**: easy to share and reproduce research
- **Production readiness**: go from research to real products

TensorFlow

http://tensorflow.org/

and

https://github.com/tensorflow/tensorflow

Open, standard software for general machine learning

Great for Deep Learning in particular

First released Nov 2015

Apache 2.0 license

TensorFlow:
Large-Scale Machine Learning on Heterogeneous Distributed Systems
(Preliminary White Paper, November 9, 2015)

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng
Google Research*

Abstract

TensorFlow [1] is an interface for expressing machine learning algorithms, and an implementation for executing such algorithms. A computation expressed using TensorFlow can be executed with little or no change on a wide variety of heterogeneous systems, ranging from mobile devices such as phones

sequence prediction [47], move selection for Go [34], pedestrian detection [2], reinforcement learning [38], and other areas [17, 5]. In addition, often in close collaboration with the Google Brain team, more than 50 teams at Google and other Alphabet companies have deployed deep neural networks using DistBelief in a wide variety

http://tensorflow.org/whitepaper2015.pdf

TensorFlow: A system for large-scale machine learning

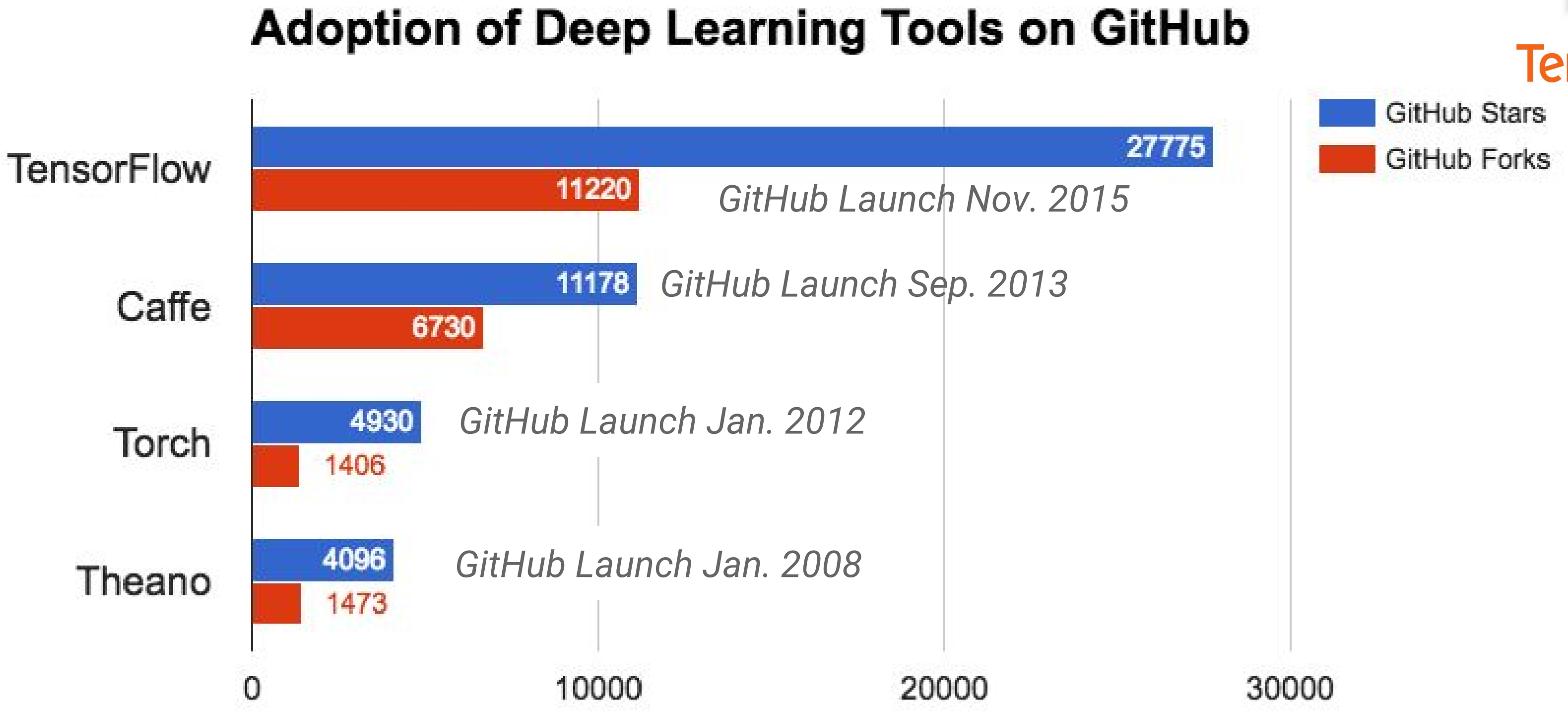
Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng

Google Brain

Preprint: arxiv.org/abs/1605.08695

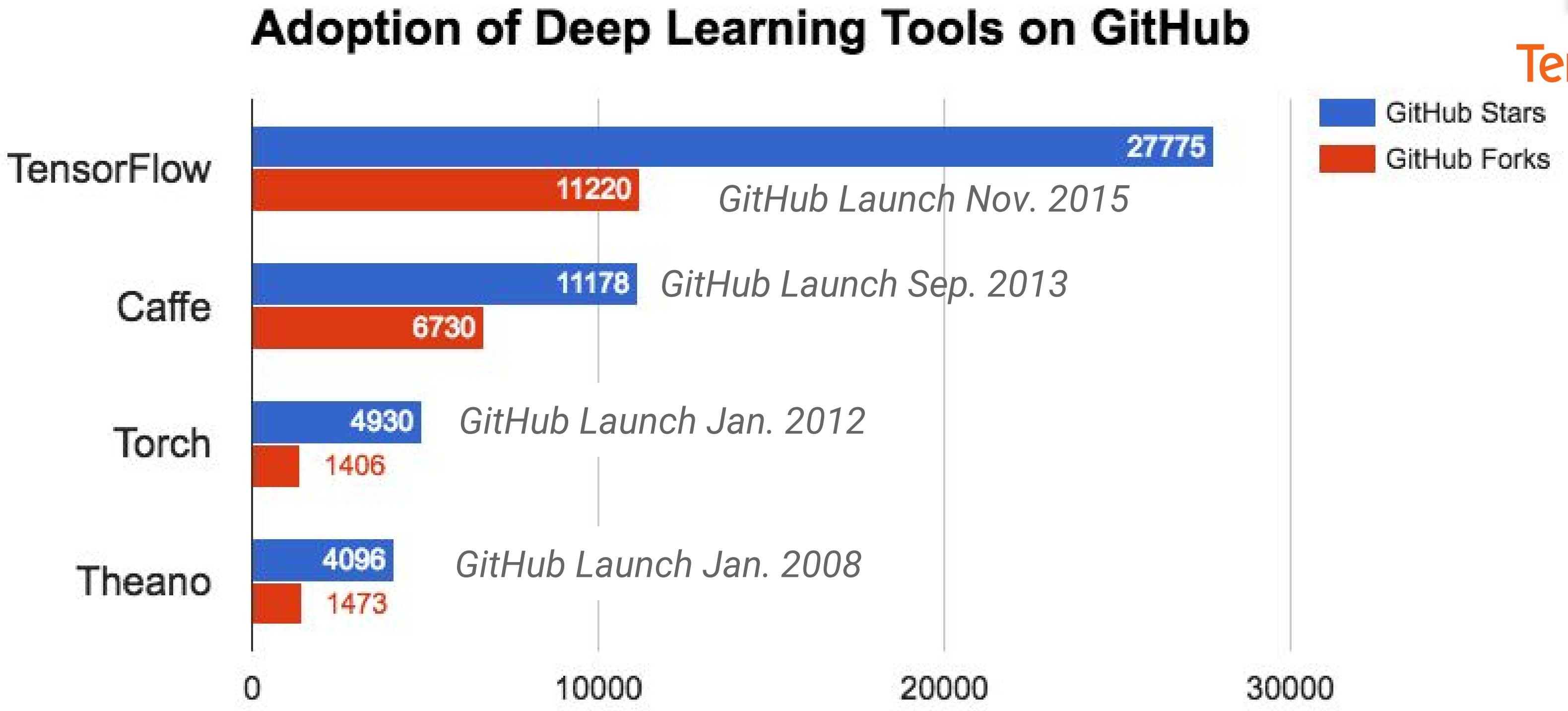
Updated version will appear in OSDI 2016

Strong External Adoption



50,000+ binary installs in 72 hours, 500,000+ since November, 2015

Strong External Adoption



50,000+ binary installs in 72 hours, 500,000+ since November, 2015

Most forked new repo on GitHub in 2015 (despite only being available in Nov, '15)

Version: master

TensorFlow Mechanics 101

This is a technical tutorial, where we walk you through the details of using TensorFlow infrastructure to train models at scale. We use again MNIST as the example.

View Tutorial

Convolutional Neural Networks

An introduction to convolutional neural networks using the CIFAR-10 data set. Convolutional neural nets are particularly tailored to images, since they exploit translation invariance to yield more compact and effective representations of visual content.

View Tutorial

Vector Representations of Words

This tutorial motivates why it is useful to learn to represent words as vectors (called word embeddings). It introduces the word2vec model as an efficient method for learning embeddings. It also covers the high-level details behind noise-contrastive training methods (the biggest recent advance in training embeddings).

View Tutorial

Recurrent Neural Networks

An introduction to RNNs, wherein we train an LSTM network to predict the next word in an English sentence. (A task sometimes called language modeling.)

View Tutorial

Sequence-to-Sequence Models

A follow on to the RNN tutorial, where we assemble a sequence-to-sequence model for machine translation. You will learn to build your own English-to-French translator, entirely machine learned, end-to-end.

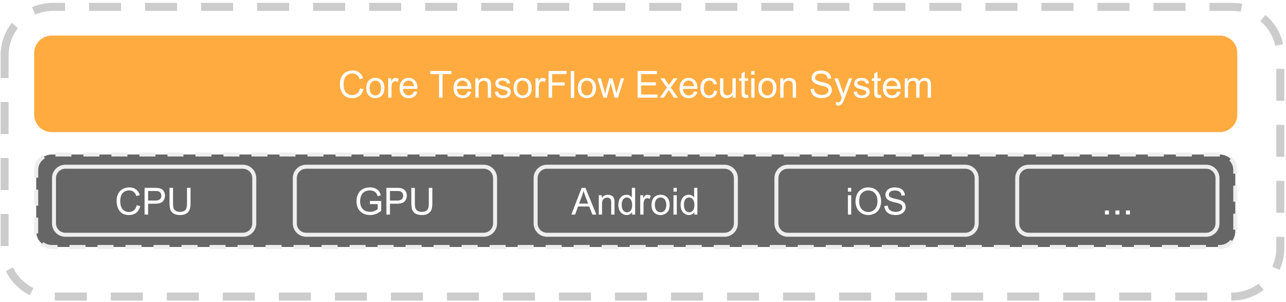
View Tutorial

Motivations

- DistBelief (our 1st system) was the first scalable deep learning system, but not as flexible as we wanted for research purposes
- Better understanding of problem space allowed us to make some dramatic simplifications
- Define the industrial standard for machine learning
- Short circuit the MapReduce/Hadoop inefficiency

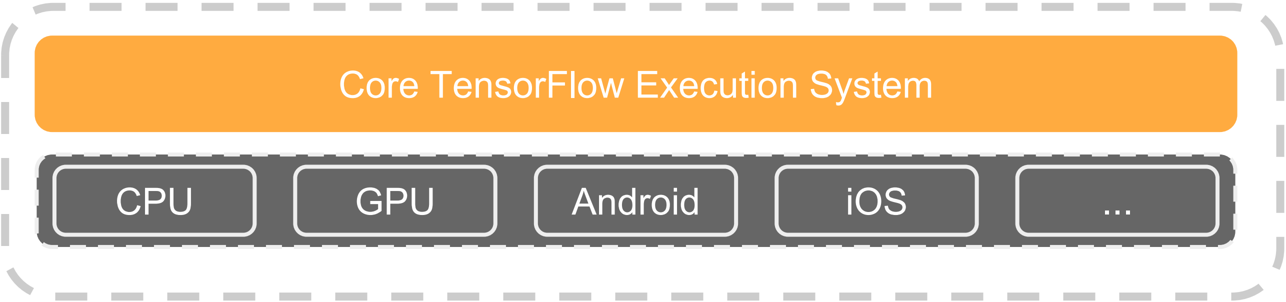
TensorFlow: Expressing High-Level ML Computations

- Core in C++
  - Very low overhead



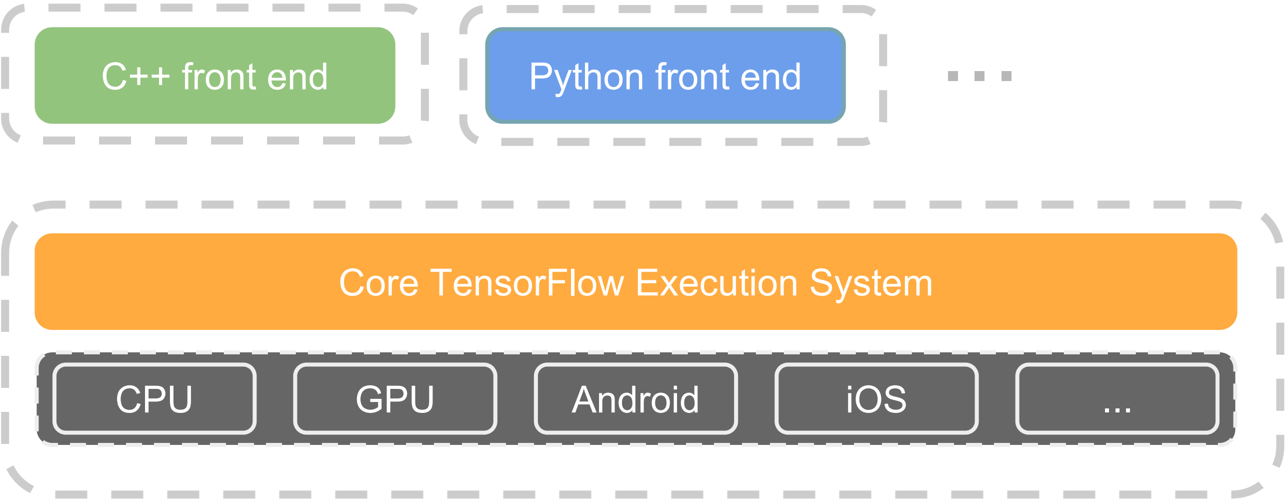
TensorFlow: Expressing High-Level ML Computations

- Core in C++
  - Very low overhead
- Different front ends for specifying/driving the computation
  - Python and C++ today, easy to add more



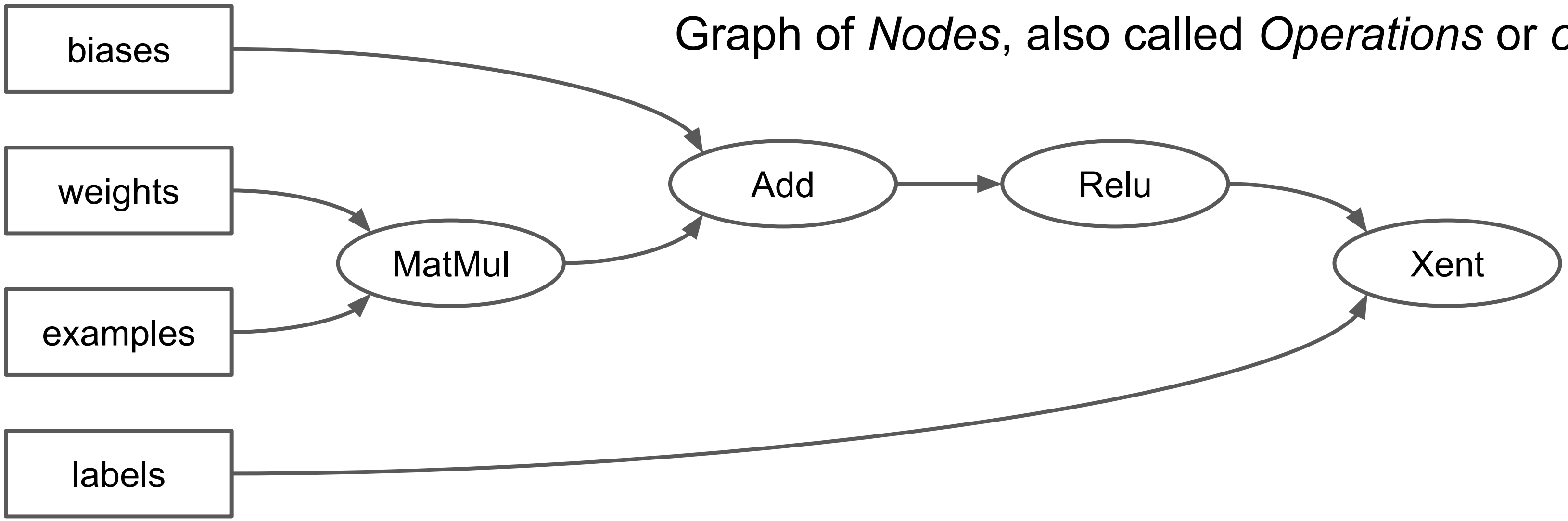
TensorFlow: Expressing High-Level ML Computations

- Core in C++
  - Very low overhead
- Different front ends for specifying/driving the computation
  - Python and C++ today, easy to add more



Computation is a dataflow graph

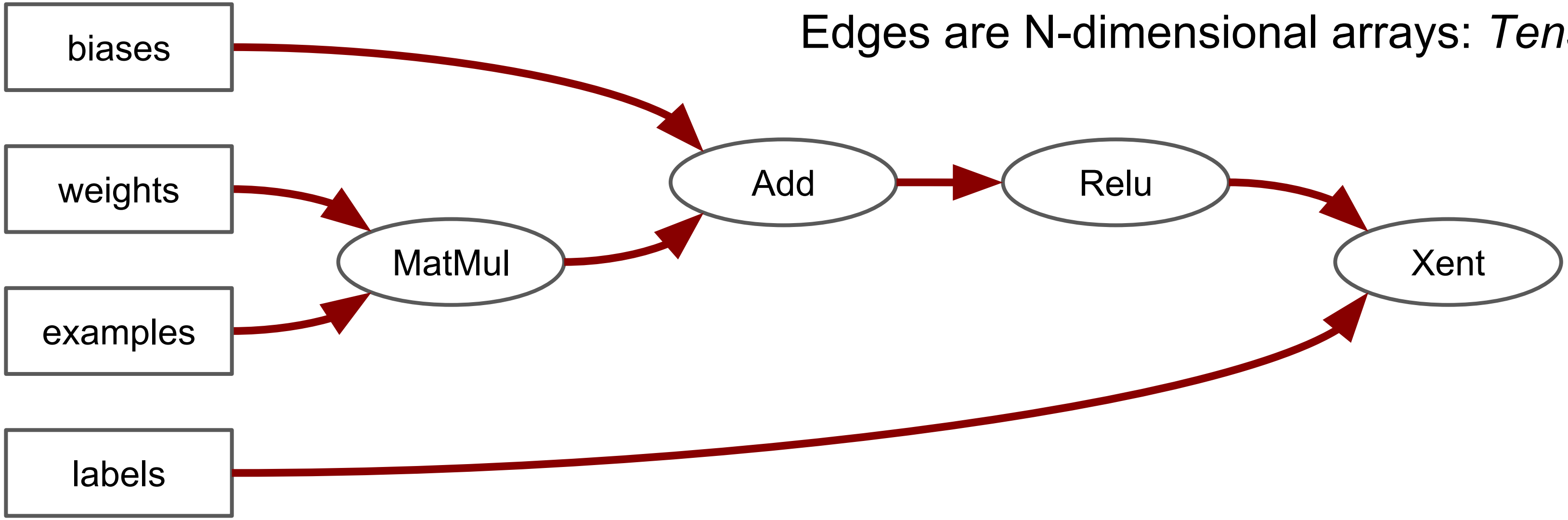
Graph of *Nodes*, also called *Operations* or *ops.*



Computation is a dataflow graph

with tensors

Edges are N-dimensional arrays: *Tensors*



Example TensorFlow fragment

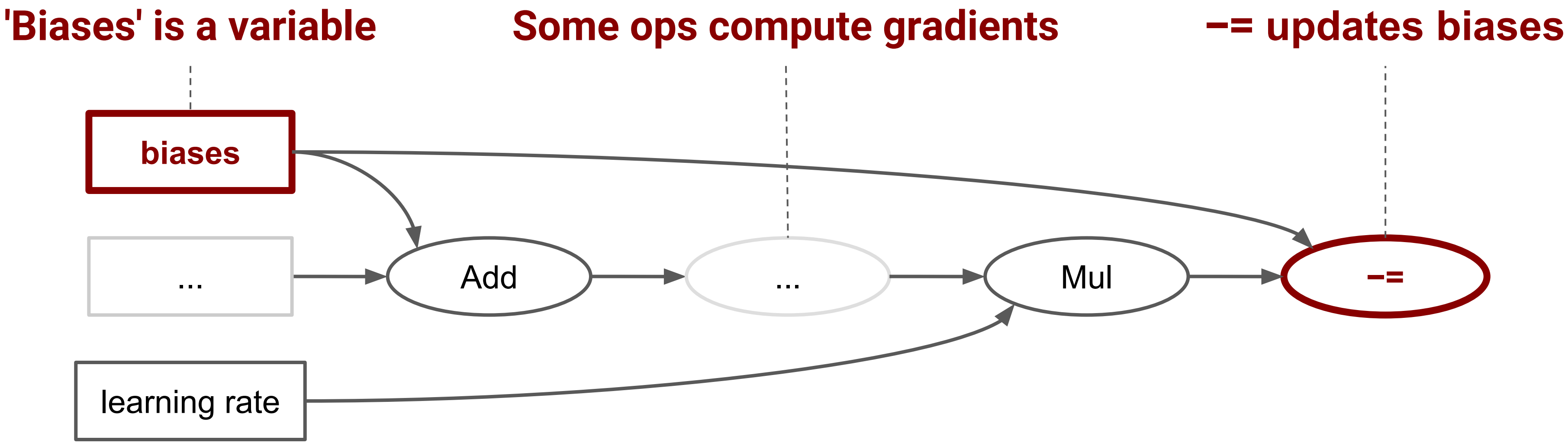
- Build a graph computing a neural net inference.

```
import tensorflow as tf
from tensorflow.examples.tutorials.mnist import input_data

mnist = input_data.read_data_sets('MNIST_data', one_hot=True)
x = tf.placeholder("float", shape=[None, 784])
W = tf.Variable(tf.zeros([784,10]))
b = tf.Variable(tf.zeros([10]))
y = tf.nn.softmax(tf.matmul(x, W) + b)
```

Computation is a dataflow graph

with state



Symbolic Differentiation

- Automatically add ops to calculate symbolic gradients of variables w.r.t. loss function.
- Apply these gradients with an optimization algorithm

```
y_ = tf.placeholder(tf.float32, [None, 10])
cross_entropy = -tf.reduce_sum(y_ * tf.log(y))
opt = tf.train.GradientDescentOptimizer(0.01)
train_op = opt.minimize(cross_entropy)
```

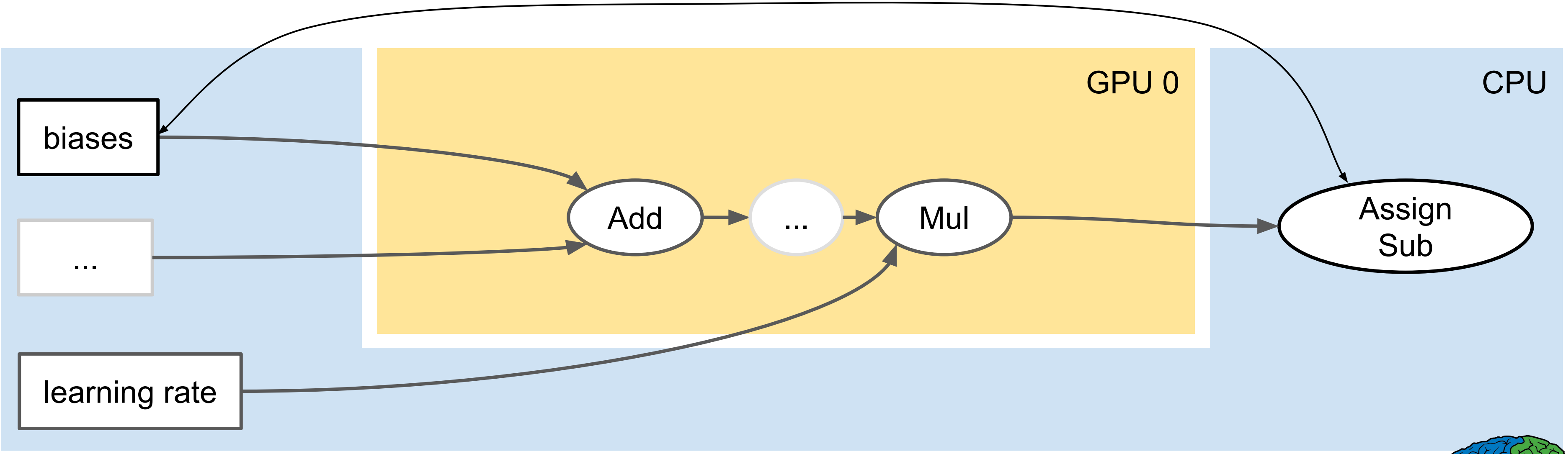
Define graph and then execute it repeatedly

- Launch the graph and run the training ops in a loop

```
init = tf.initialize_all_variables()
sess = tf.Session()
sess.run(init)
for i in range(1000):
  batch_xs, batch_ys = mnist.train.next_batch(100)
  sess.run(train_step, feed_dict={x: batch_xs, y_: batch_ys})
```

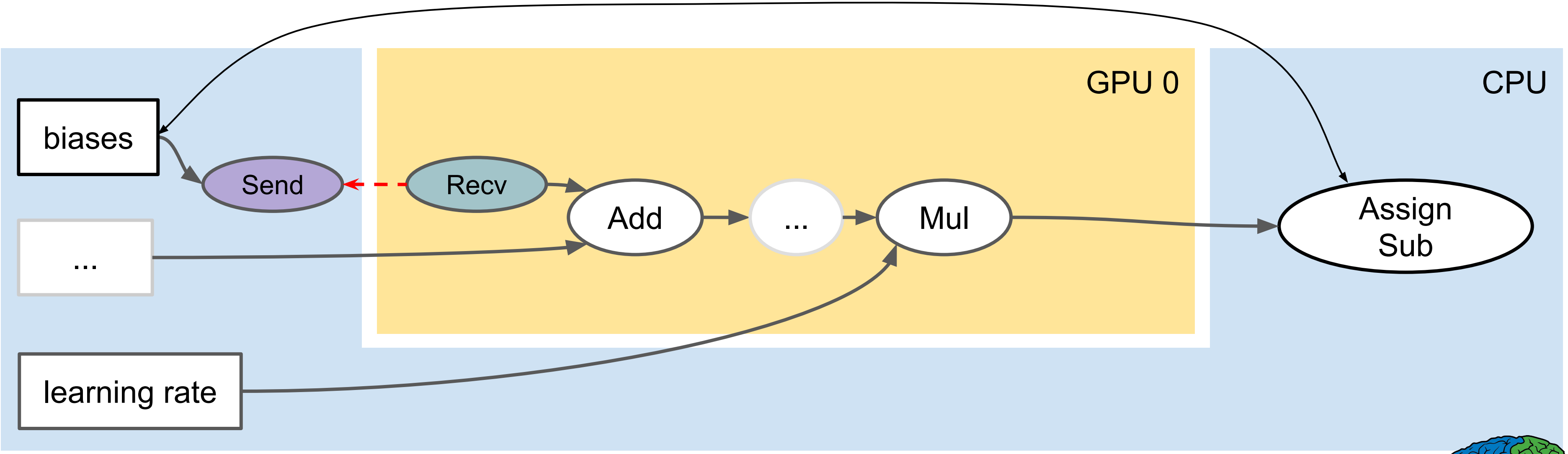
Computation is a dataflow graph

distributed



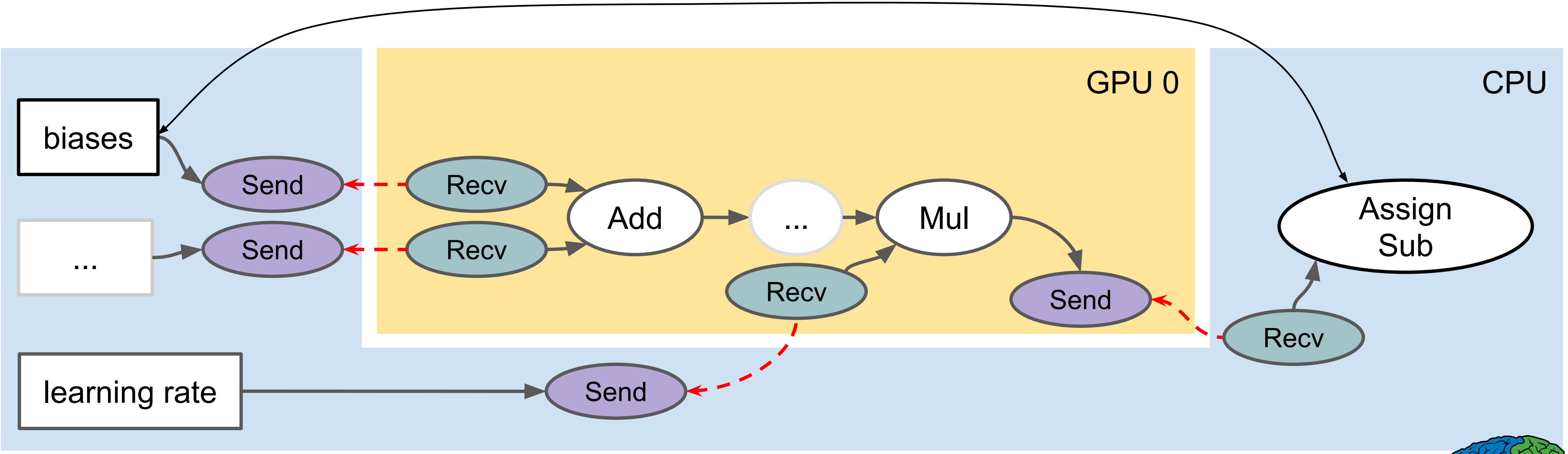
Assign *Devices* to Ops

- TensorFlow inserts *Send/Recv* Ops to transport tensors across devices
- *Recv* ops pull data from *Send* ops



Assign *Devices* to Ops

- TensorFlow inserts *Send/Recv* Ops to transport tensors across devices
- *Recv* ops pull data from *Send* ops



November 2015

Release 0.5.0

Initial release of TensorFlow.

December 2015

Release 0.6.0

Major Features and Improvements

- Python 3.3+ support via changes to python codebase and ability to specify python version via ./configure.
- Some improvements to GPU performance and memory usage: convnet benchmarks roughly equivalent with native cudnn v2 performance. Improvements mostly due to moving to 32-bit indices, faster shuffling kernels. More improvements to come in later releases.

Bug Fixes

- Lots of fixes to documentation and tutorials, many contributed by the public.
- 271 closed issues on github issues.

February 2016

Release 0.7.0

Major Features and Improvements

- Allow using any installed Cuda >= 7.0 and cuDNN >= R2, and add support for cuDNN R4
- Added a `contrib/` directory for unsupported or experimental features, including higher level `layers` module
- Added an easy way to add and dynamically load user-defined ops
- Built out a good suite of tests, things should break less!
- Added `MetaGraphDef` which makes it easier to save graphs with metadata
- Added assignments for "Deep Learning with TensorFlow" udacity course

Bug Fixes and Other Changes

- Added a versioning framework for `GraphDef` s to ensure compatibility
- Enforced Python 3 compatibility
- Internal changes now show up as sensibly separated commits
- Open-sourced the doc generator

April 2016

Release 0.8.0

Major Features and Improvements

- Added a distributed runtime using GRPC
- Move skflow to `contrib/learn`
- Better linear optimizer in `contrib/linear_optimizer`
- Random forest implementation in `contrib/tensor_forest`
- CTC loss and decoders in `contrib/ctc`
- Basic support for `half` data type
- Better support for loading user ops (see examples in `contrib/`)
- Allow use of (non-blocking) Eigen threadpool with `TENSORFLOW_USE_EIGEN_THREADPOOL` define
- Add an extension mechanism for adding network file system support
- TensorBoard displays metadata stats (running time, memory usage and device used) and tensor shapes

Big Fixes and Other Changes

- Utility for inspecting checkpoints
- Basic tracing and timeline support
- Allow building against cuDNN 5 (not incl. RNN/LSTM support)

June 2016

Release 0.9.0

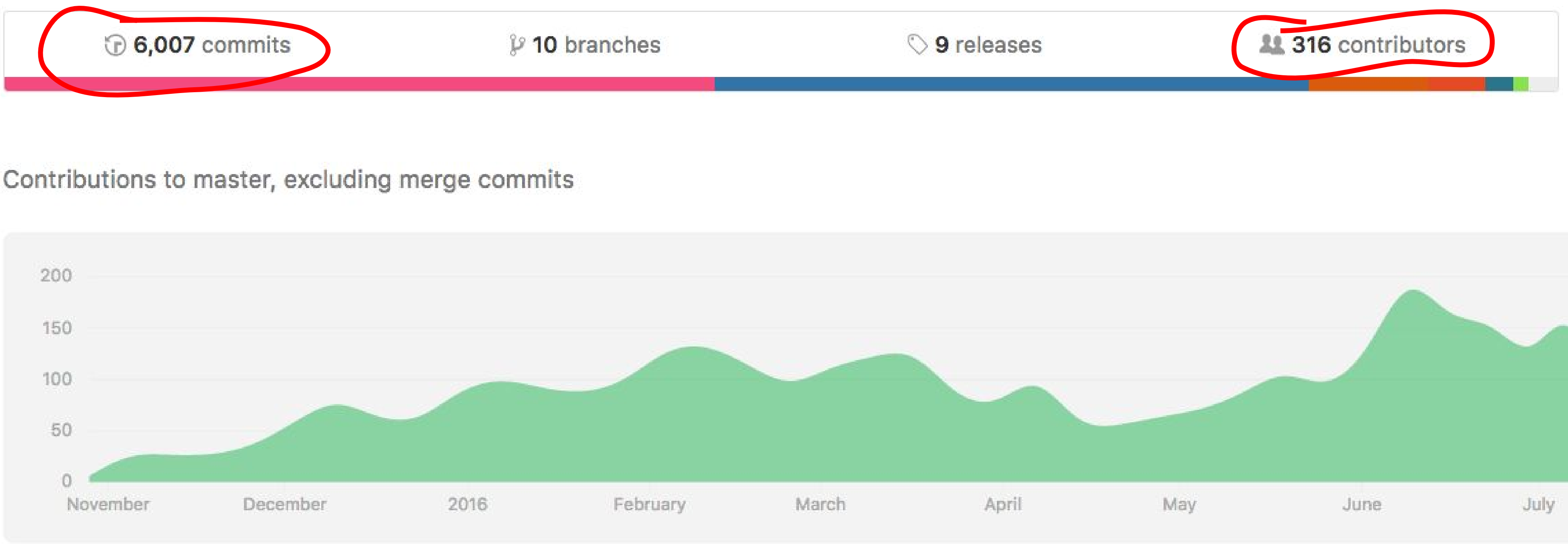
Major Features and Improvements

- Python 3.5 support and binaries
- Added iOS support
- Added support for processing on GPUs on MacOS
- Added makefile for better cross-platform build support (C API only)
- fp16 support and improved complex128 support for many ops
- Higher level functionality in contrib.{layers,losses,metrics,learn}
- More features to Tensorboard
- Improved support for string embedding and sparse features
- The RNN api is finally "official" (see, e.g., `tf.nn.dynamic_rnn`, `tf.nn.rnn`, and the classes in `tf.nn.rnn_cell`).
- TensorBoard now has an Audio Dashboard, with associated audio summaries.

Big Fixes and Other Changes

- Turned on CuDNN Autotune.
- Added support for using third-party Python optimization algorithms (contrib.opt).
- Google Cloud Storage filesystem support.
- HDF5 support

Activity

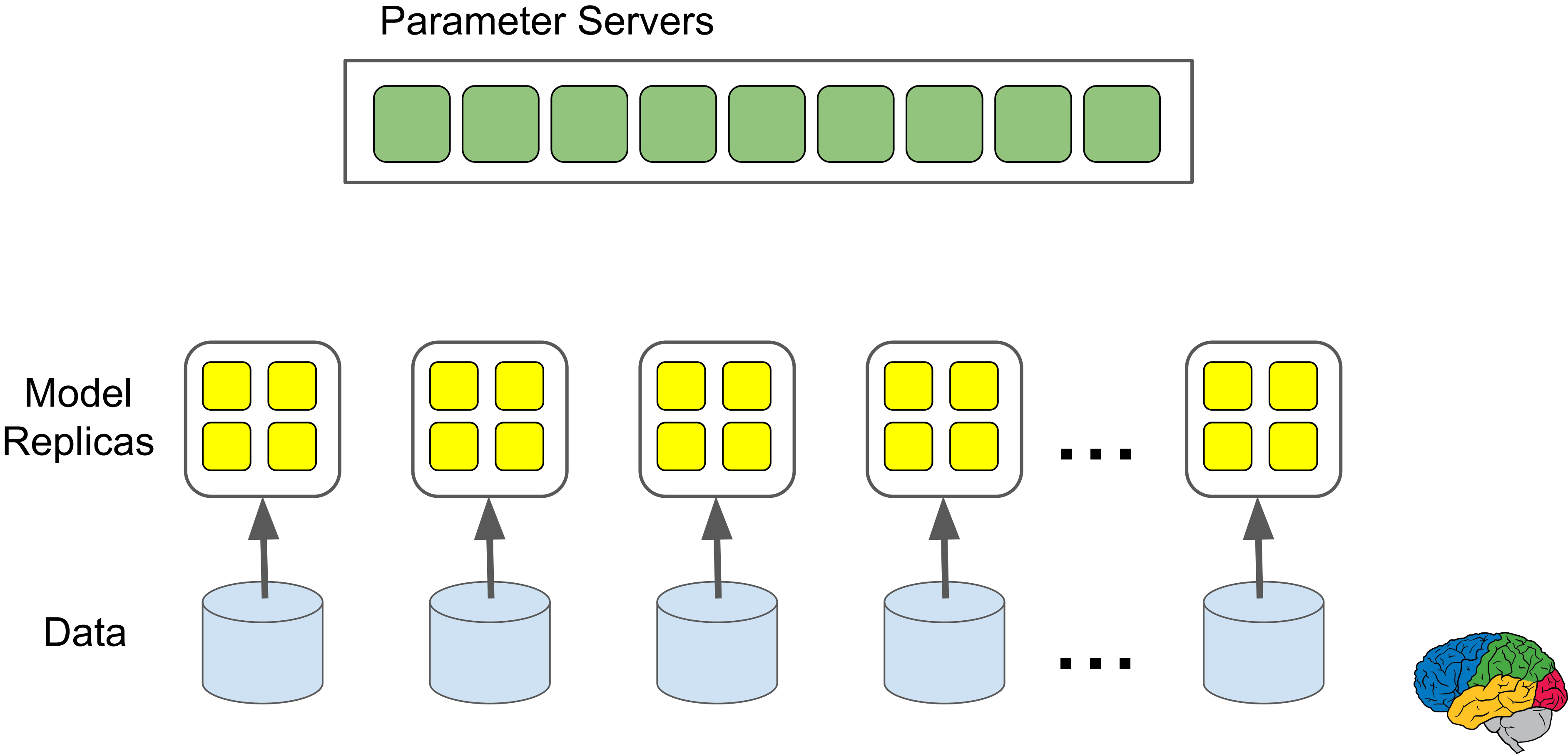

6,007 commits
10 branches
9 releases
316 contributors
Contributions to master, excluding merge commits
200
150
100
50
0
November
December
2016
February
March
April
May
June
July

Experiment Turnaround Time and Research Productivity

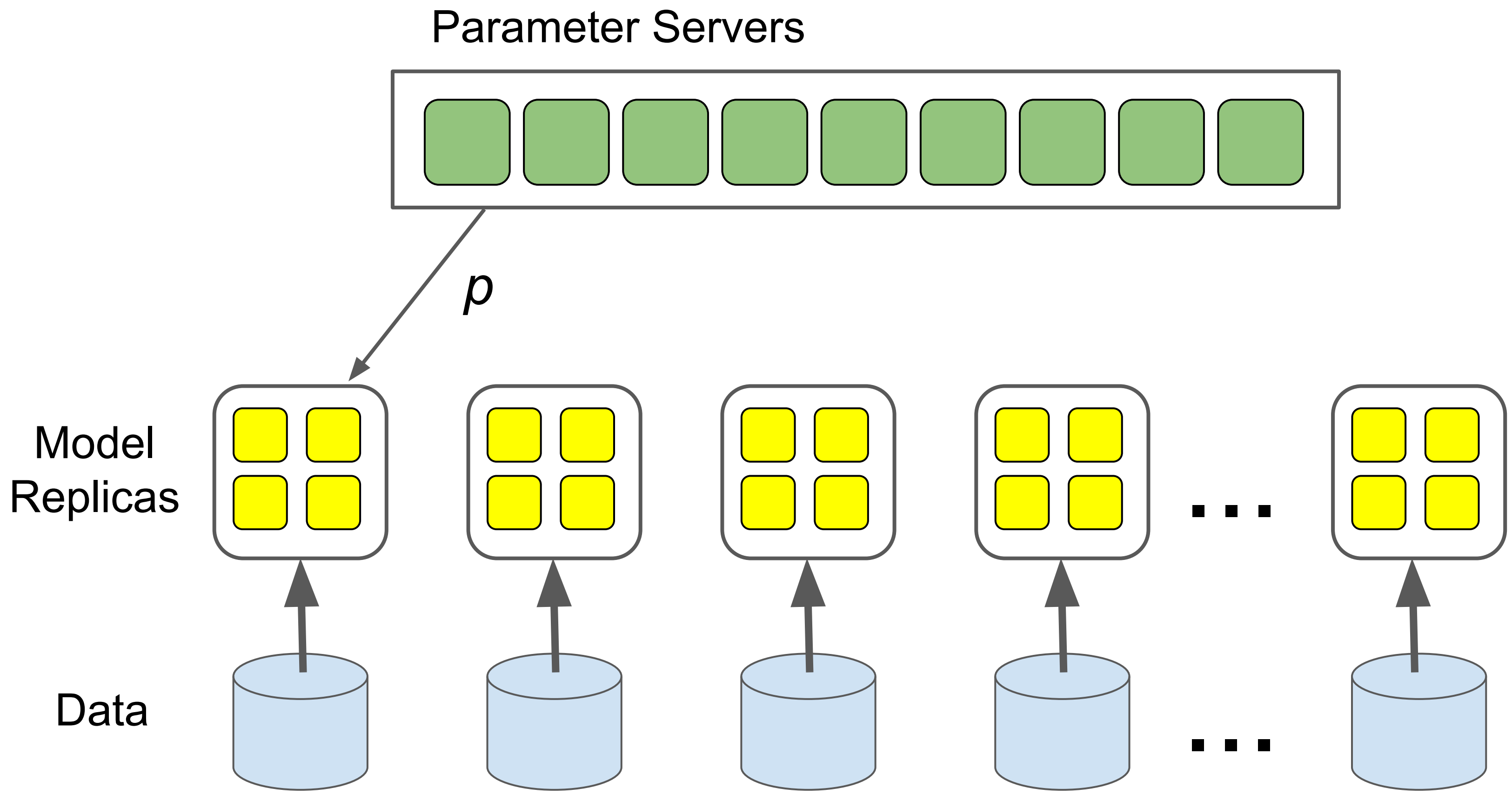
- **Minutes, Hours**:
  - **Interactive research! Instant gratification!**
- **1-4 days**
  - Tolerable
  - Interactivity replaced by running many experiments in parallel
- **1-4 weeks**
  - High value experiments only
  - Progress stalls
- **>1 month**
  - Don’t even try


Google
google.com/datacenters

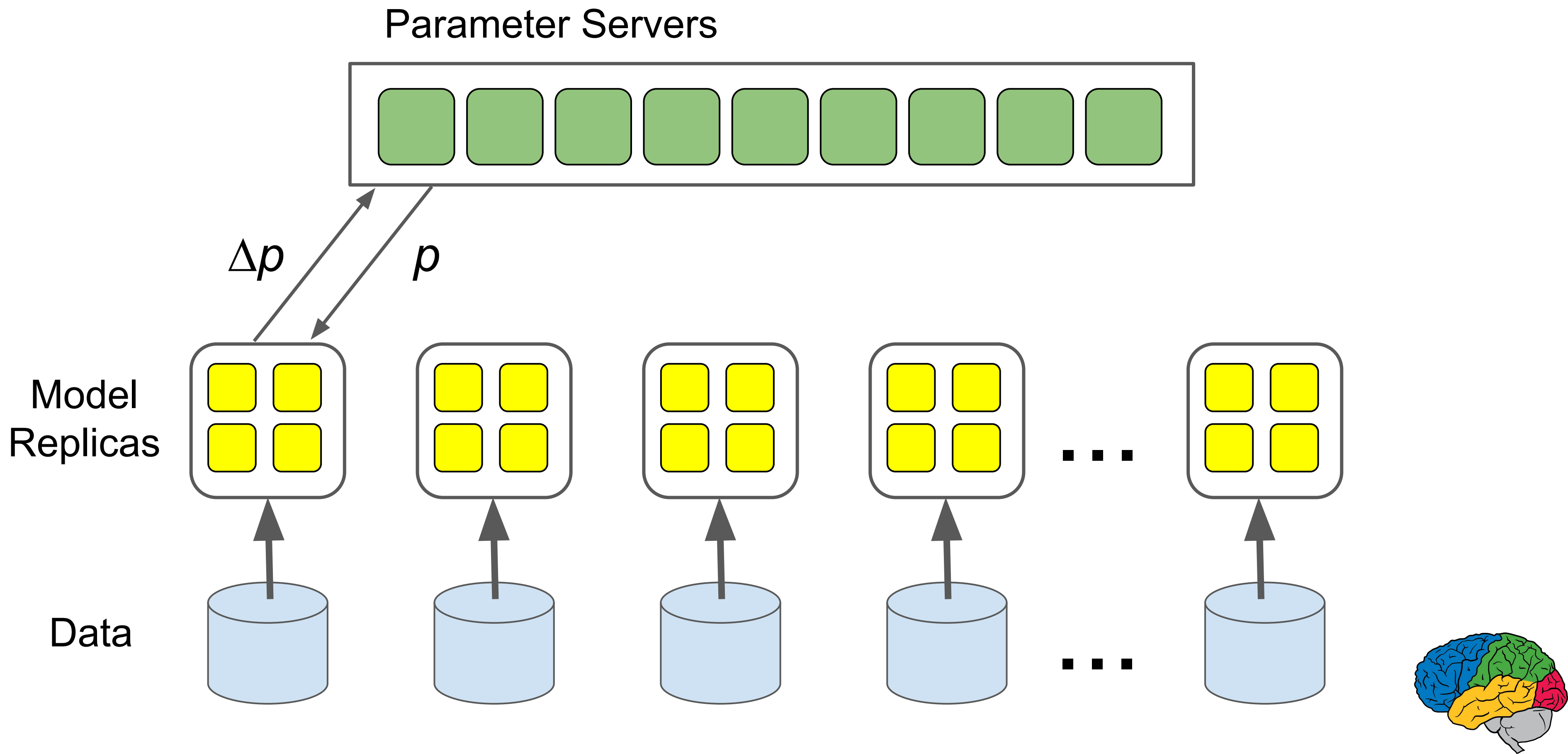
Data Parallelism



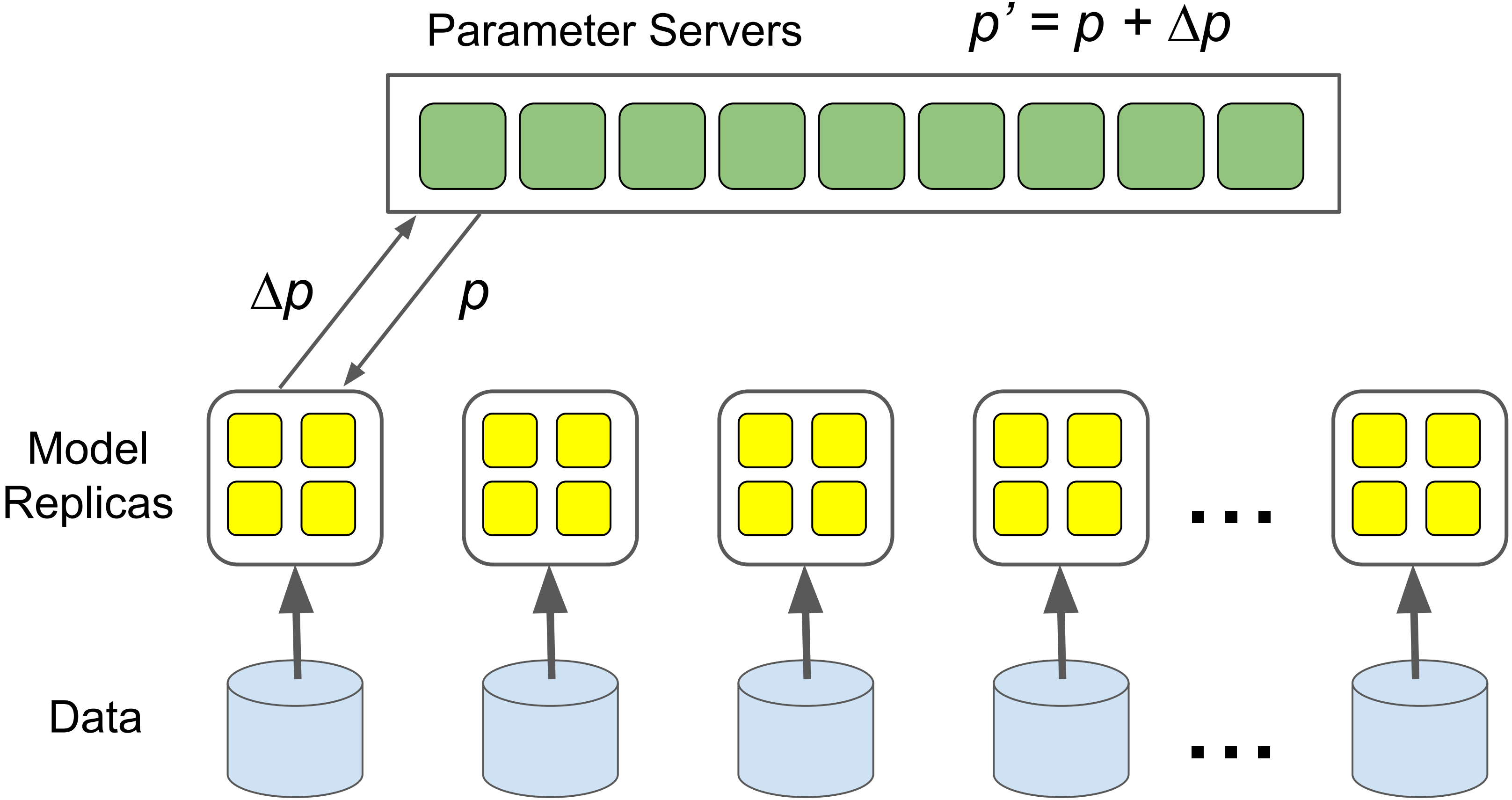
Data Parallelism



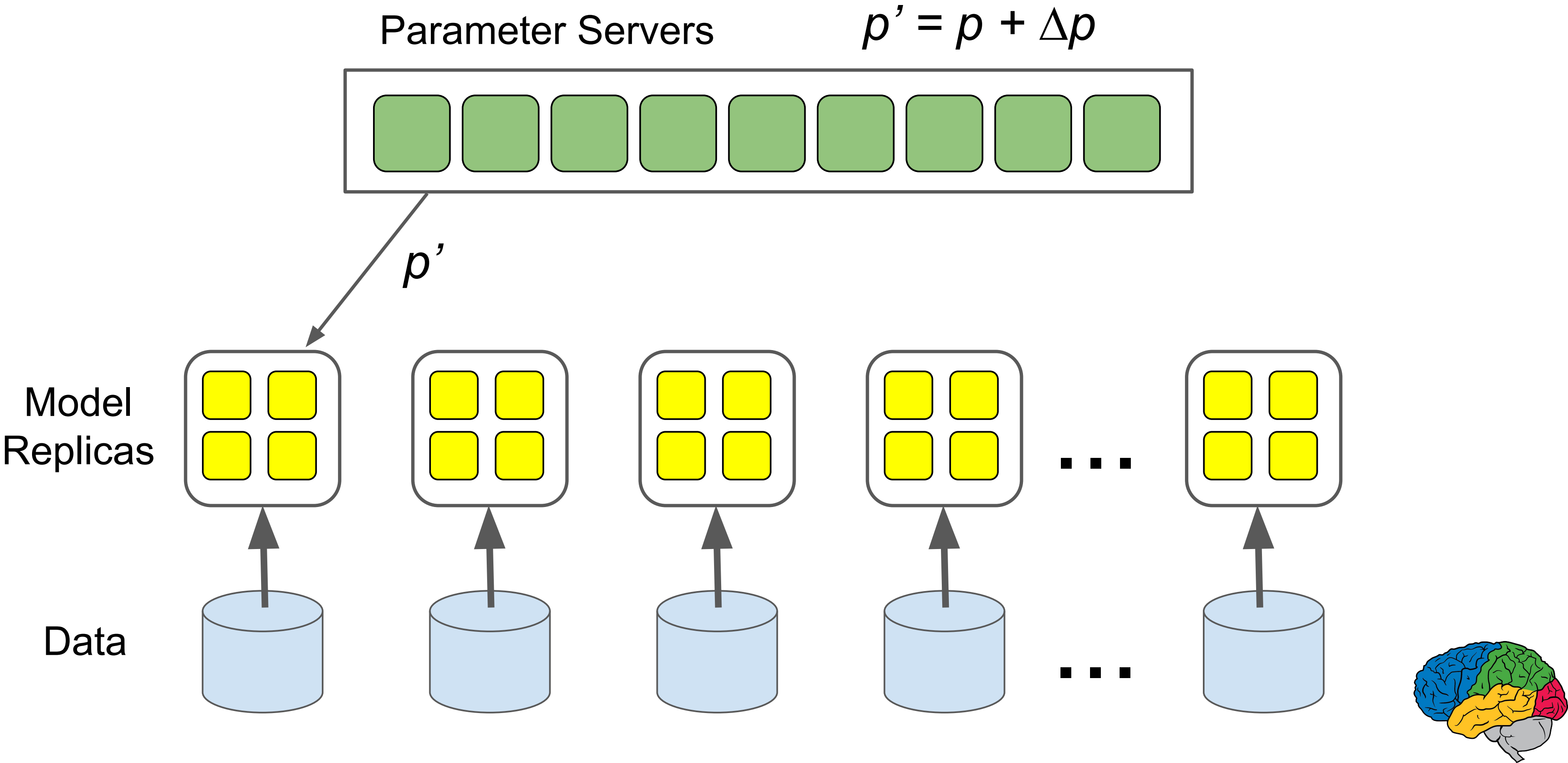
Data Parallelism



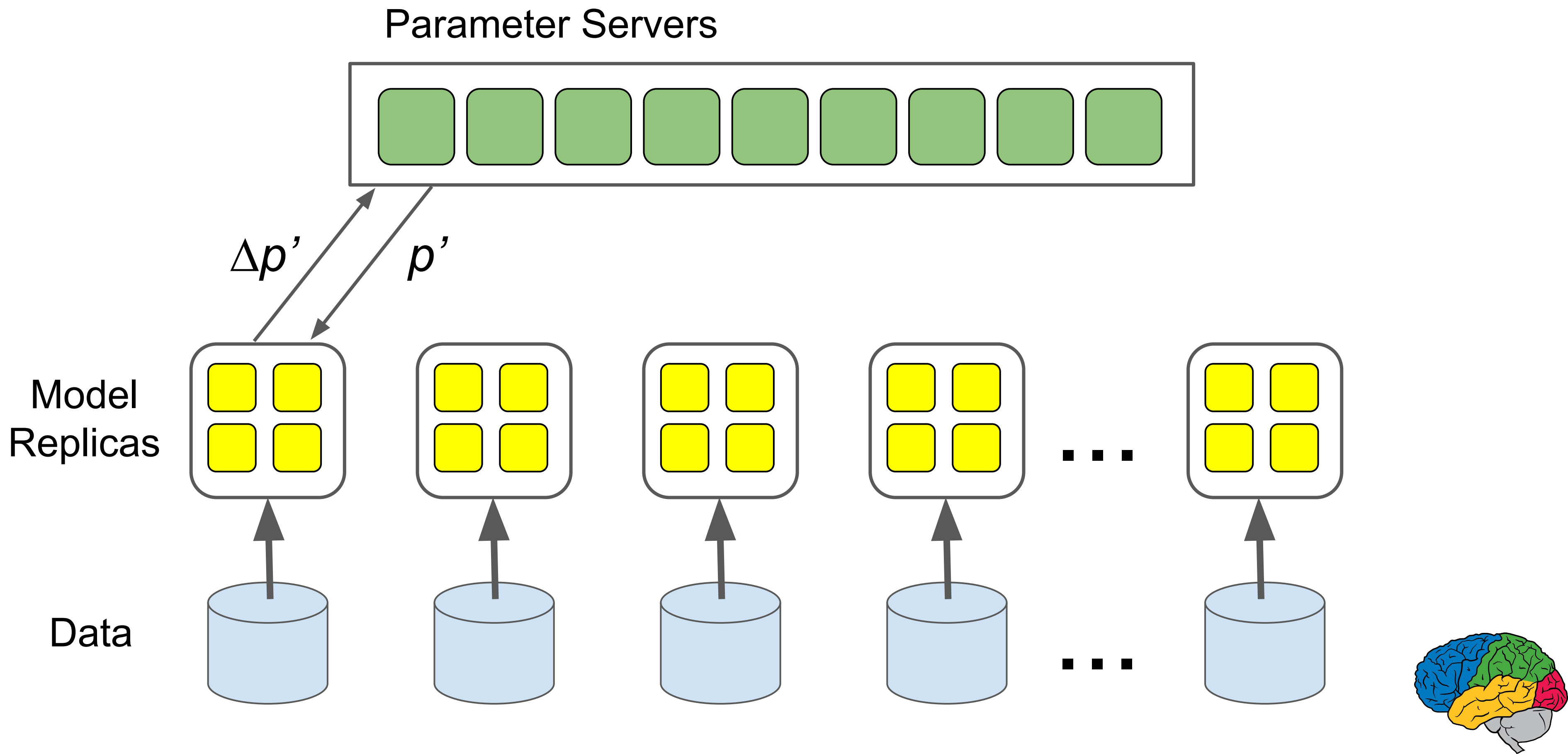
Data Parallelism



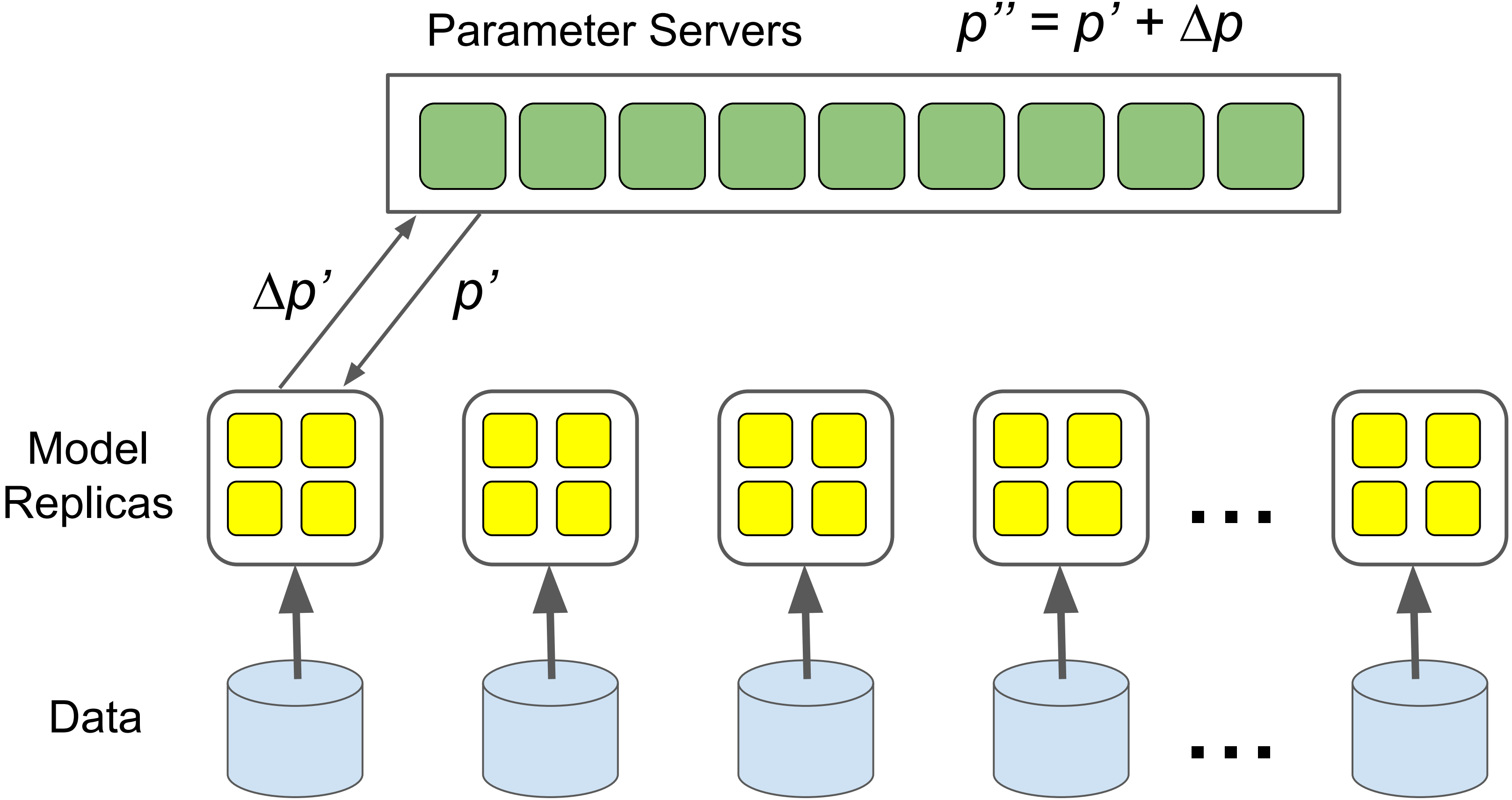
Data Parallelism



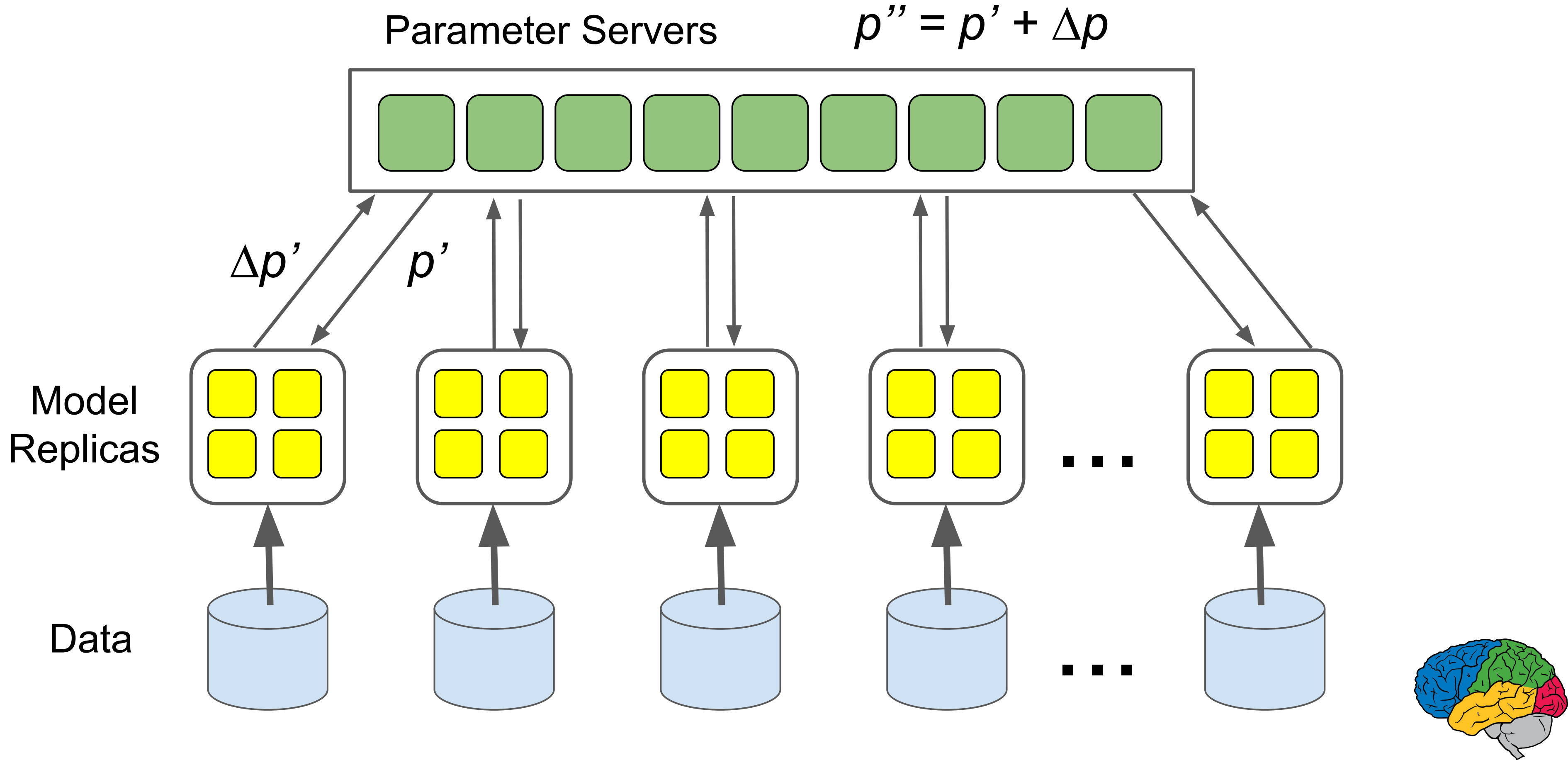
Data Parallelism



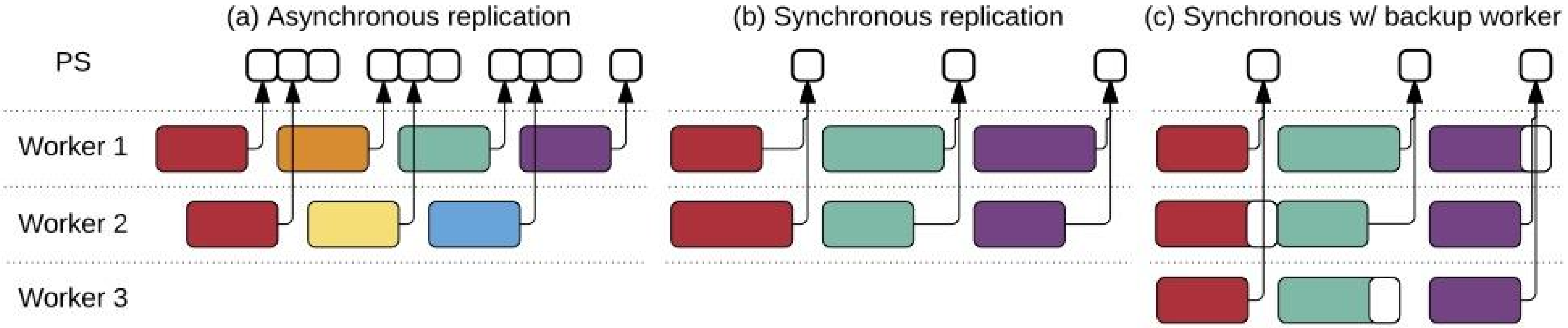
Data Parallelism



Data Parallelism



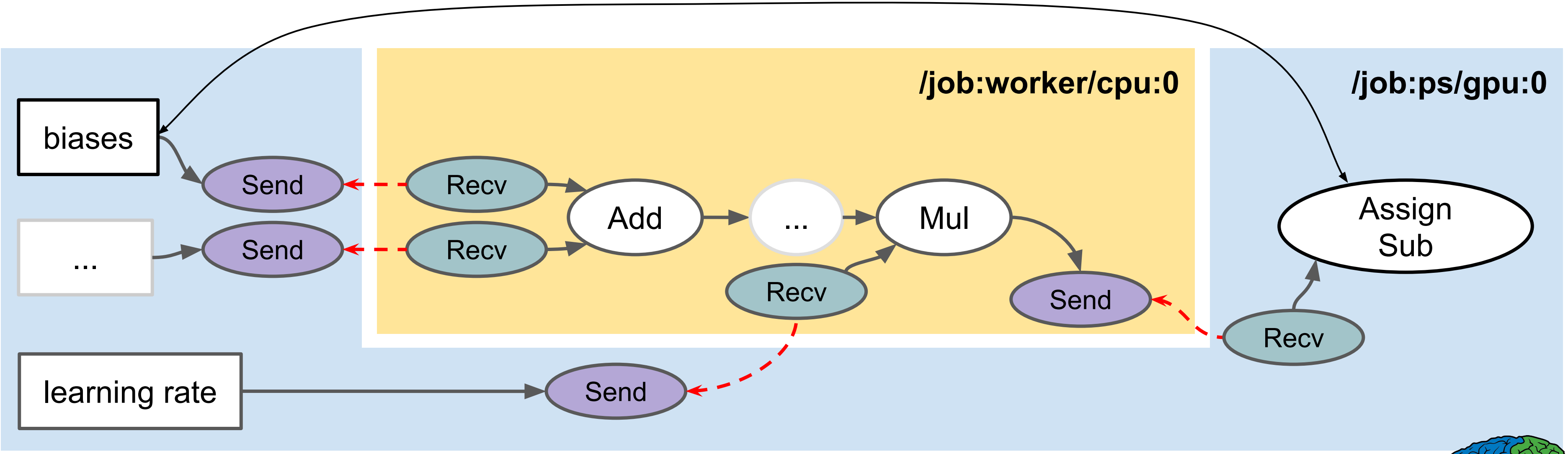
Distributed training mechanisms



Graph structure and low-level graph primitives (queues) allow us to play with synchronous vs. asynchronous update algorithms.

Cross process communication is the same!

- Communication across machines over the network abstracted identically to cross device communication.



No specialized parameter server subsystem!

Image Model Training Time

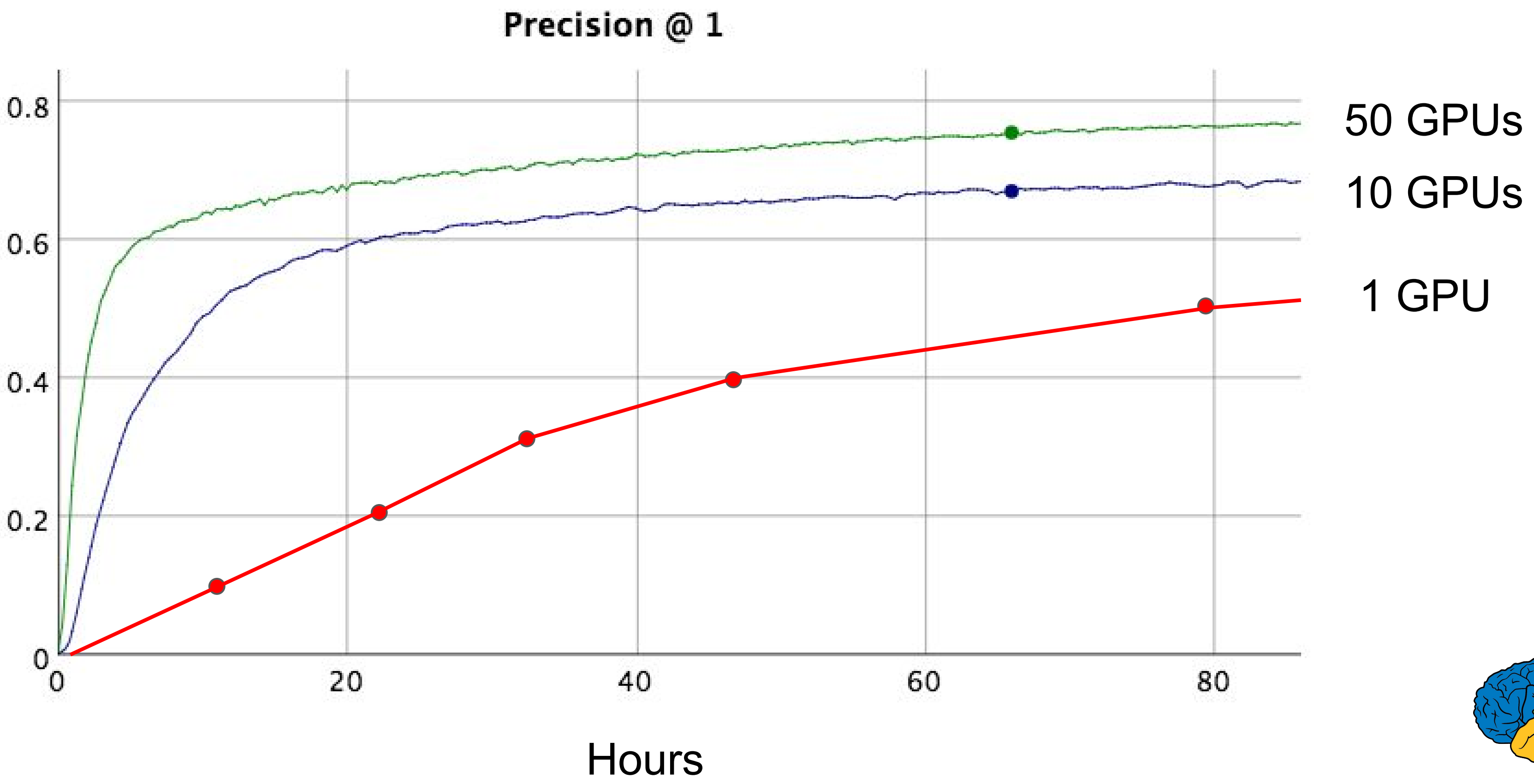
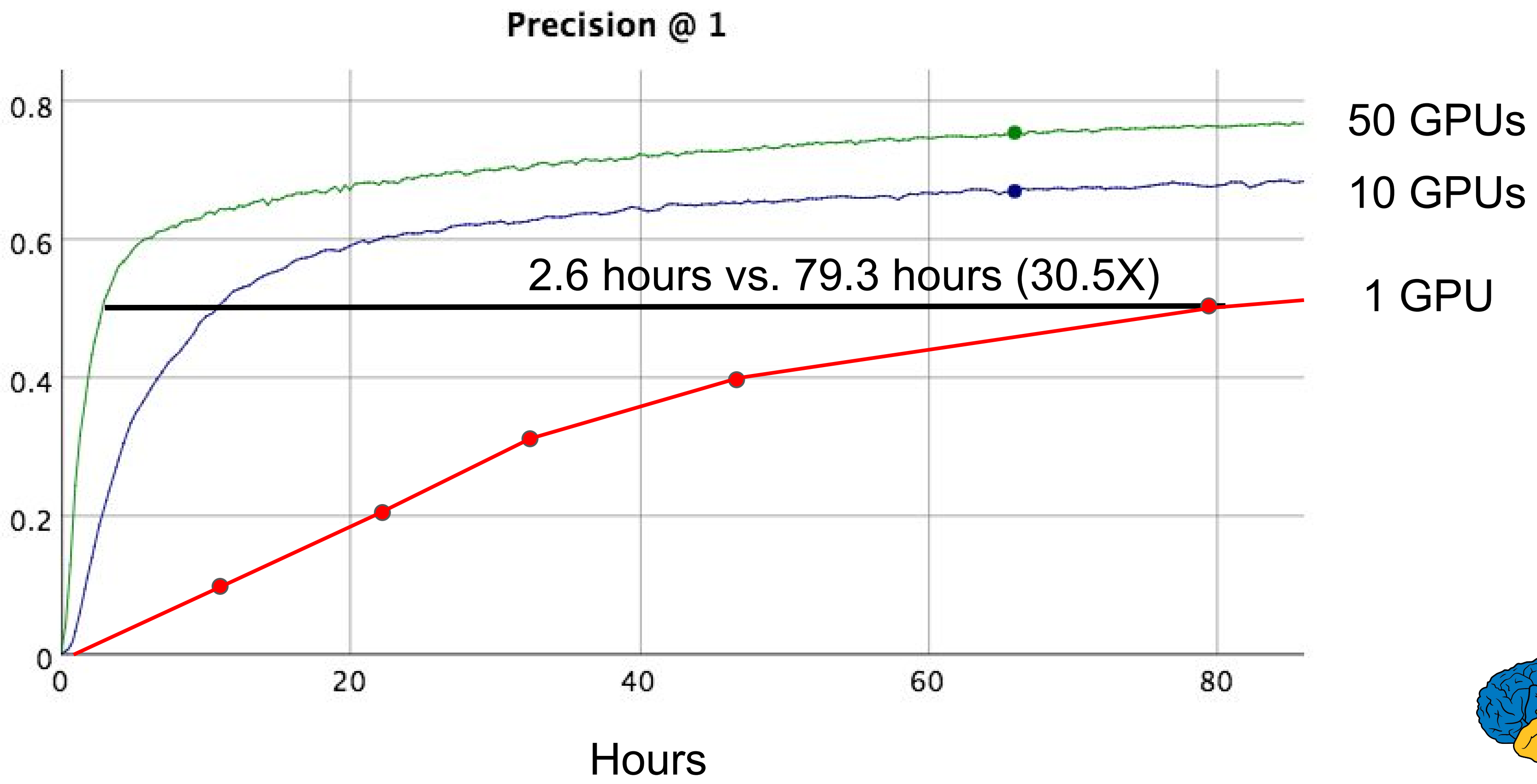
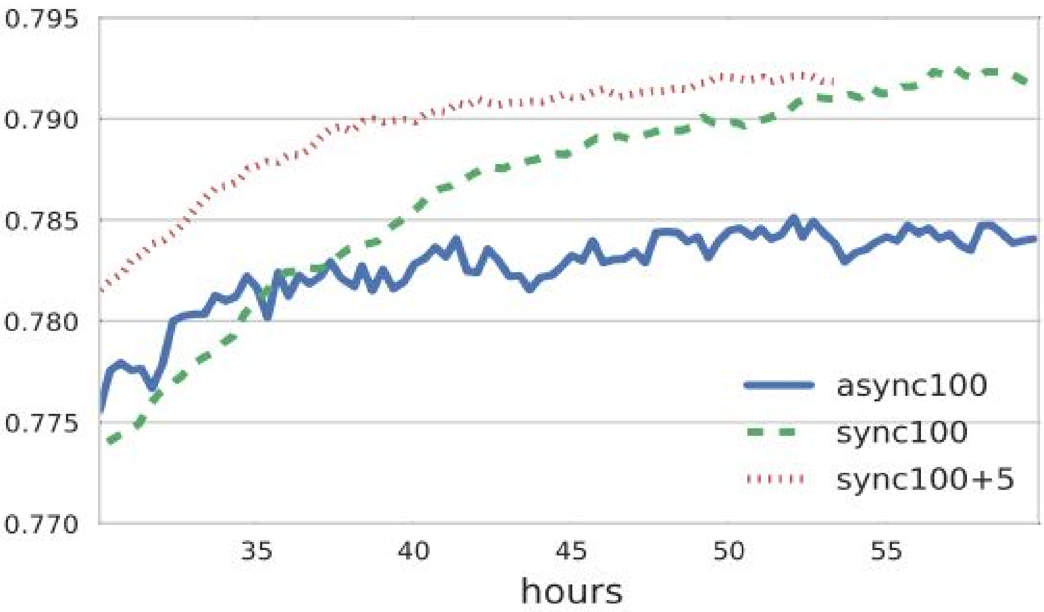


Image Model Training Time



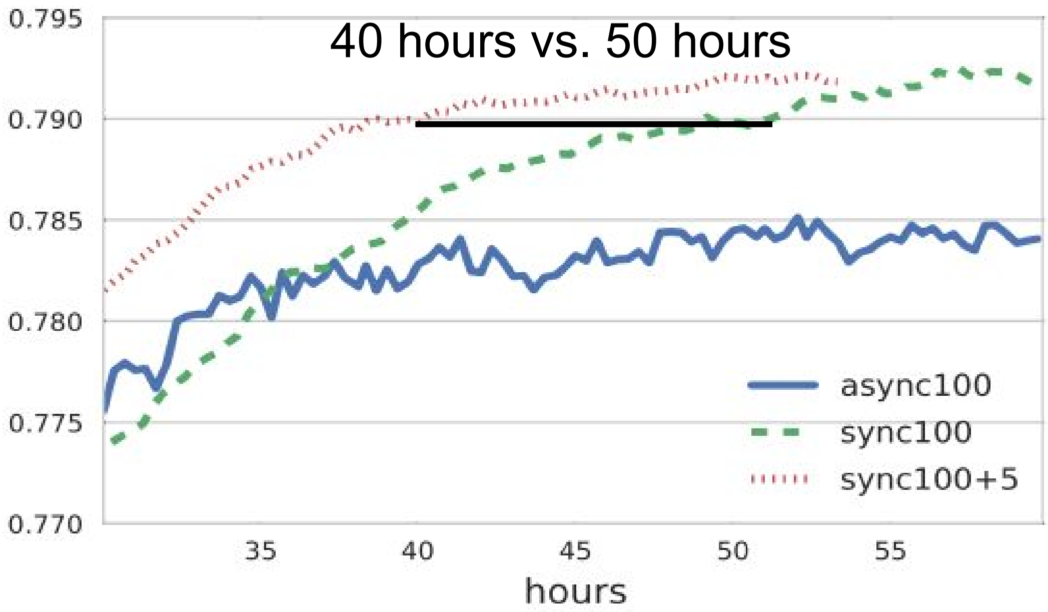
Sync converges faster (time to accuracy)



Synchronous updates (with backup workers) trains to higher accuracy faster
Better scaling to more workers (less loss of accuracy)

Revisiting Distributed Synchronous SGD, Jianmin Chen, Rajat Monga, Samy Bengio, Raal Jozefowicz, ICLR Workshop 2016, arxiv.org/abs/1604.00981

Sync converges faster (time to accuracy)



Synchronous updates (with backup workers) trains to higher accuracy faster
Better scaling to more workers (less loss of accuracy)

Revisiting Distributed Synchronous SGD, Jianmin Chen, Rajat Monga, Samy Bengio, Raal Jozefowicz, ICLR Workshop 2016, arxiv.org/abs/1604.00981

General Computations

Although we originally built TensorFlow for our uses around deep neural networks, it’s actually quite flexible

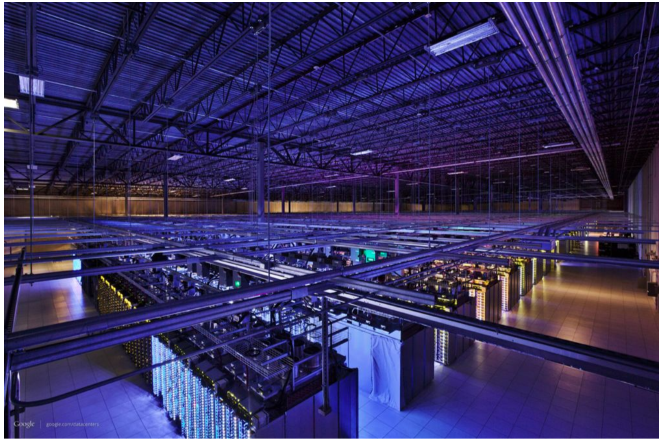
Wide variety of machine learning and other kinds of numeric computations easily expressible in the computation graph model

Runs on Variety of Platforms

phones

single machines (CPU and/or GPUs) …

distributed systems of 100s of machines and/or GPU cards



custom ML hardware

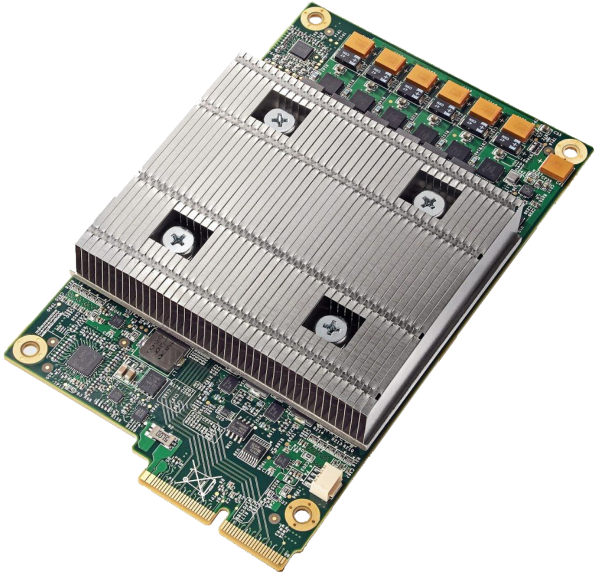
Trend: Much More Heterogeneous hardware

General purpose CPU performance scaling has slowed significantly

Specialization of hardware for certain workloads will be more important

Tensor Processing Unit

Custom machine learning ASIC

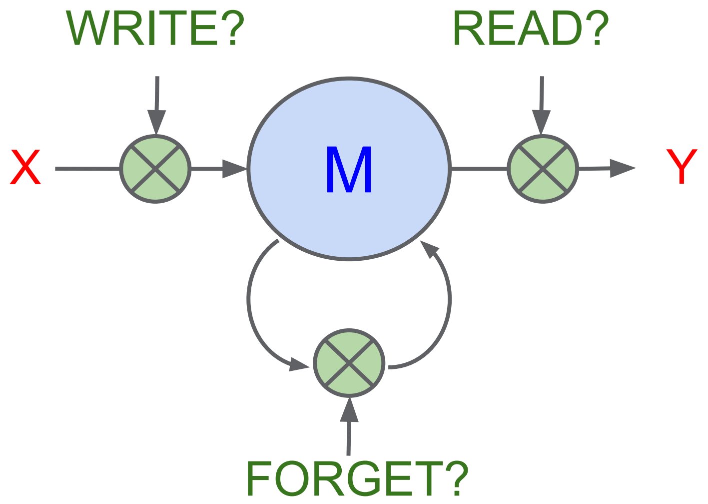


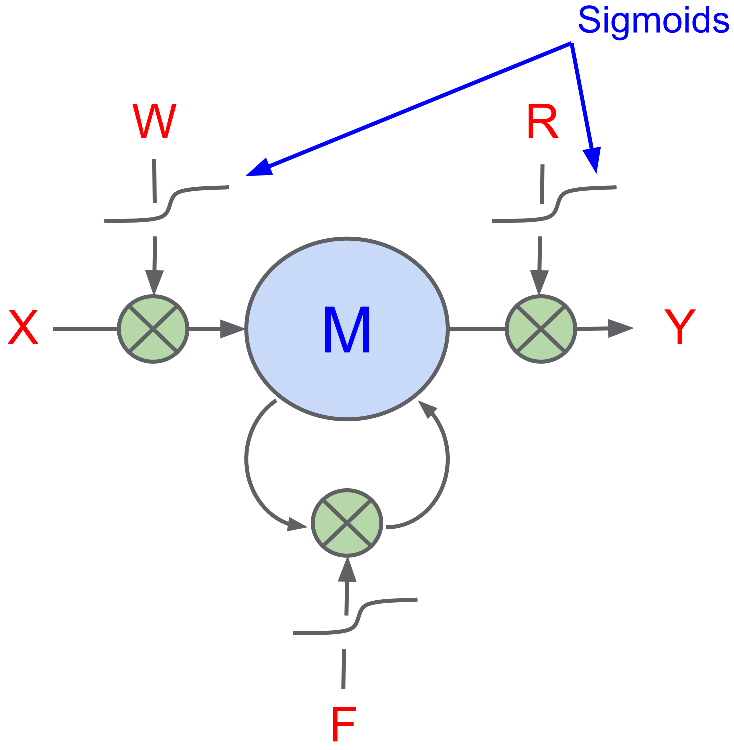
In production use for >16 months: used on every search query, used for AlphaGo match, ...

See Google Cloud Platform blog: [Google supercharges machine learning tasks with TPU custom chip](), by Norm Jouppi, May, 2016

Long Short-Term Memory (LSTMs):
Make Your Memory Cells Differentiable

[Hochreiter & Schmidhuber, 1997]





Example: LSTM [Hochreiter et al, 1997][Gers et al, 1999]

$$i_t = W_{ix}x_t + W_{ih}h_{t-1} + b_i$$
$$j_t = W_{jx}x_t + W_{jh}h_{t-1} + b_j$$
$$f_t = W_{fx}x_t + W_{fh}h_{t-1} + b_f$$
$$o_t = W_{ox}x_t + W_{oh}h_{t-1} + b_o$$
$$c_t = \boxed{\sigma(f_t) \odot c_{t-1}} + \sigma(i_t) \odot \tanh(j_t)$$
$$h_t = \sigma(o_t) \odot \tanh(c_t)$$

Enables long term dependencies to flow

TensorFlow

```python
def __call__(self, inputs, state, scope=None):
  """Long short-term memory cell (LSTM)."""
  with vs.variable_scope(scope or type(self).__name__):  # "BasicLSTMCell"
    # Parameters of gates are concatenated into one multiply for efficiency.
    c, h = array_ops.split(1, 2, state)
    concat = linear([inputs, h], 4 * self._num_units, True)

    # i = input_gate, j = new_input, f = forget_gate, o = output_gate
    i, j, f, o = array_ops.split(1, 4, concat)

    new_c = c * sigmoid(f + self._forget_bias) + sigmoid(i) * tanh(j)
    new_h = tanh(new_c) * sigmoid(o)

  return new_h, array_ops.concat(1, [new_c, new_h])
```

Example: LSTM

```
for i in range(20):
    m, c = LSTMCell(x[i], mprev, cprev)
    mprev = m
    cprev = c
```

Example: Deep LSTM

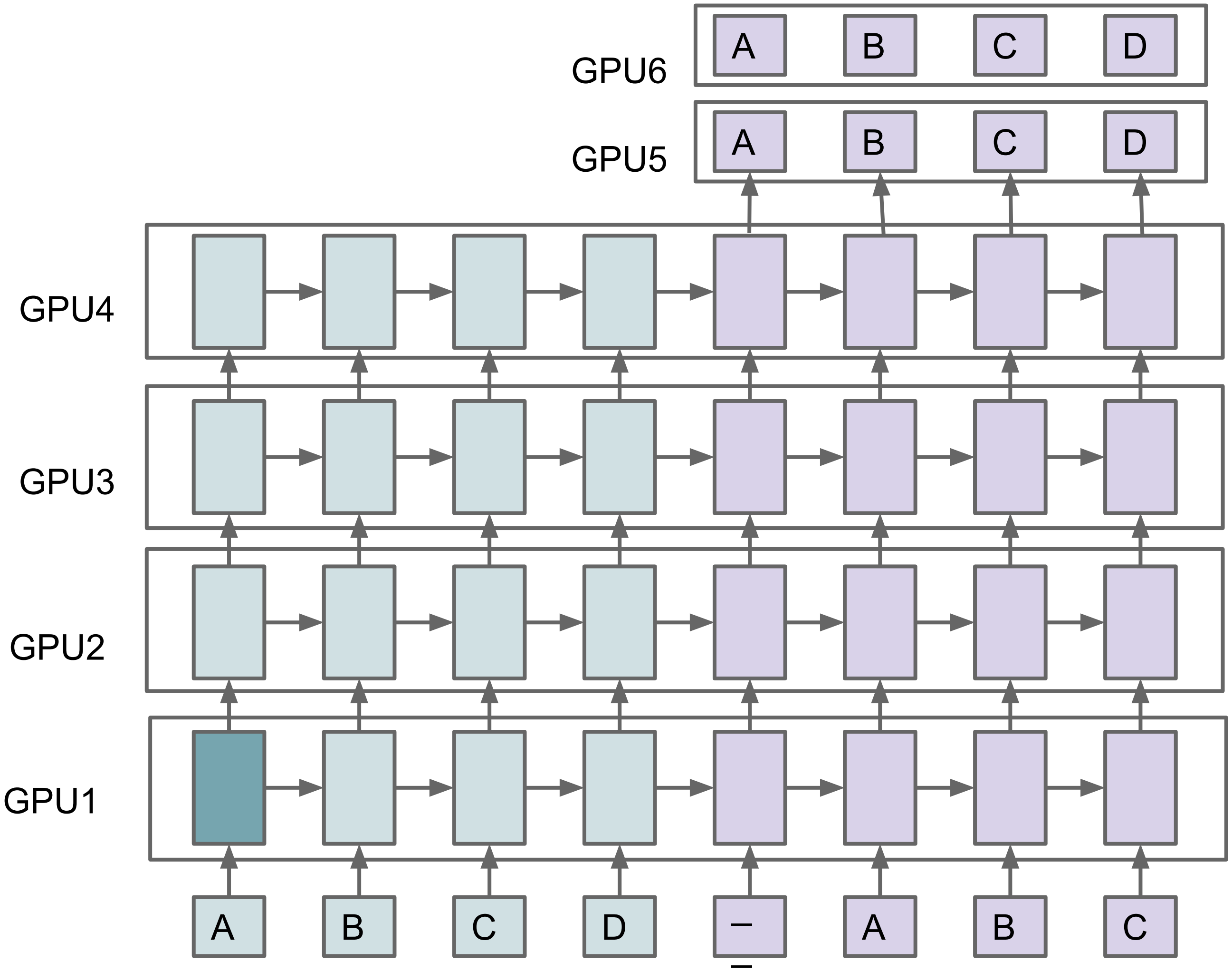
```
for i in range(20):
  for d in range(4): # d is depth
      input = x[i] if d is 0 else m[d-1]
      m[d], c[d] = LSTMCell(input, mprev[d], cprev[d])
      mprev[d] = m[d]
      cprev[d] = c[d]
```

Example: Deep LSTM

```
for i in range(20):
  for d in range(4): # d is depth
      input = x[i] if d is 0 else m[d-1]
      m[d], c[d] = LSTMCell(input, mprev[d], cprev[d])
      mprev[d] = m[d]
      cprev[d] = c[d]
```

Example: Deep LSTM

```
for i in range(20):
  for d in range(4): # d is depth
    with tf.device("/gpu:%d" % d):
      input = x[i] if d is 0 else m[d-1]
      m[d], c[d] = LSTMCell(input, mprev[d], cprev[d])
      mprev[d] = m[d]
      cprev[d] = c[d]
```

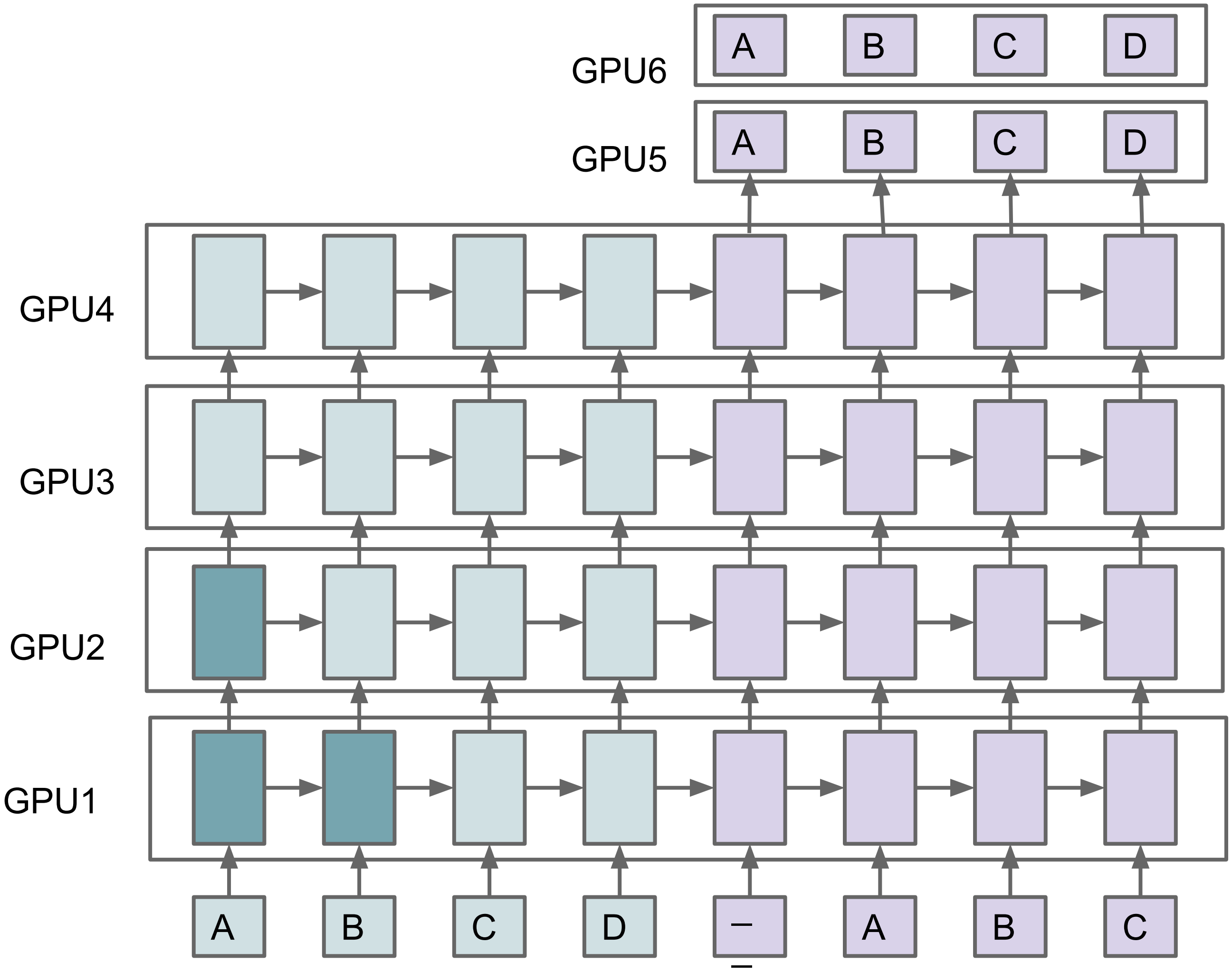


80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence

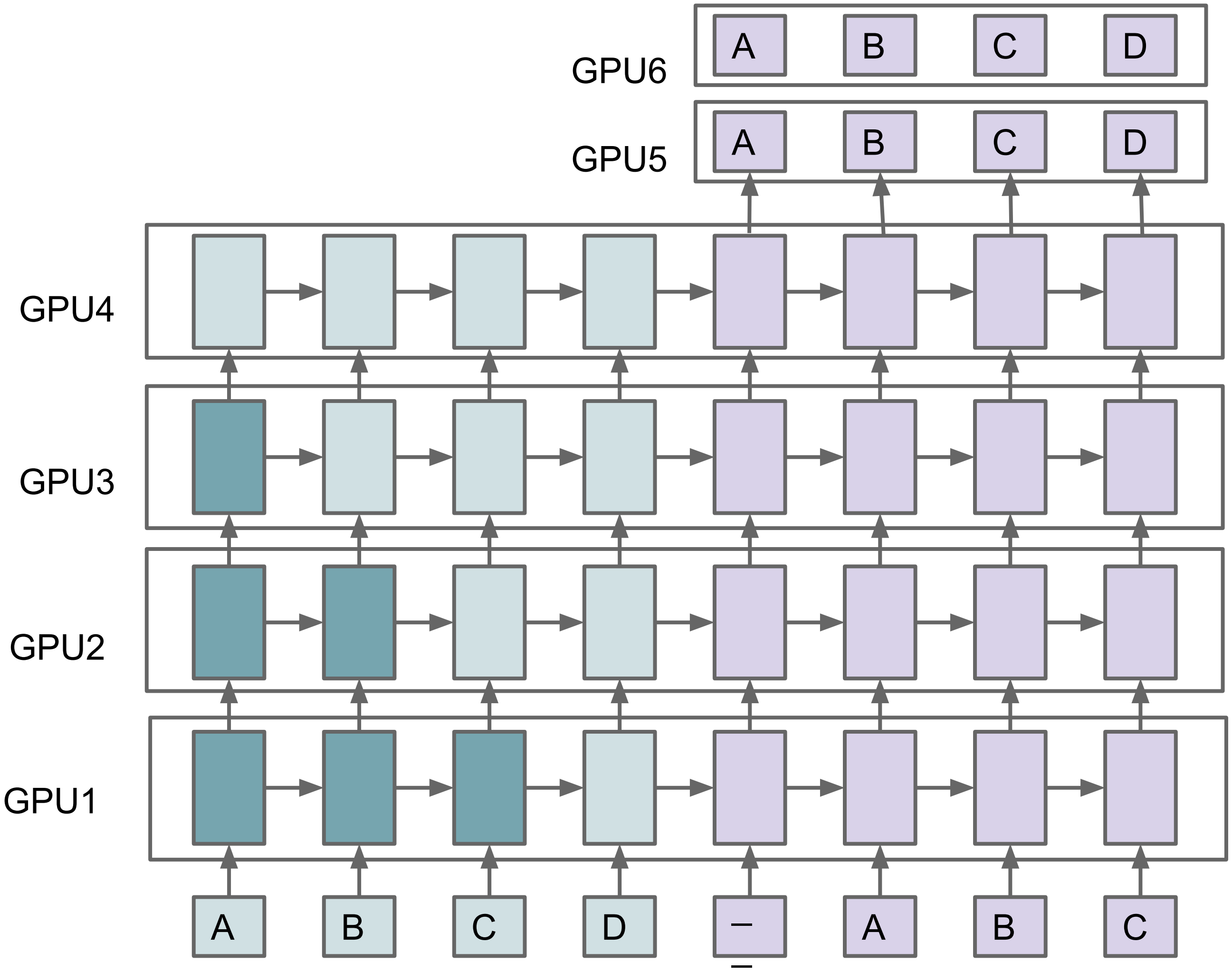


80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence

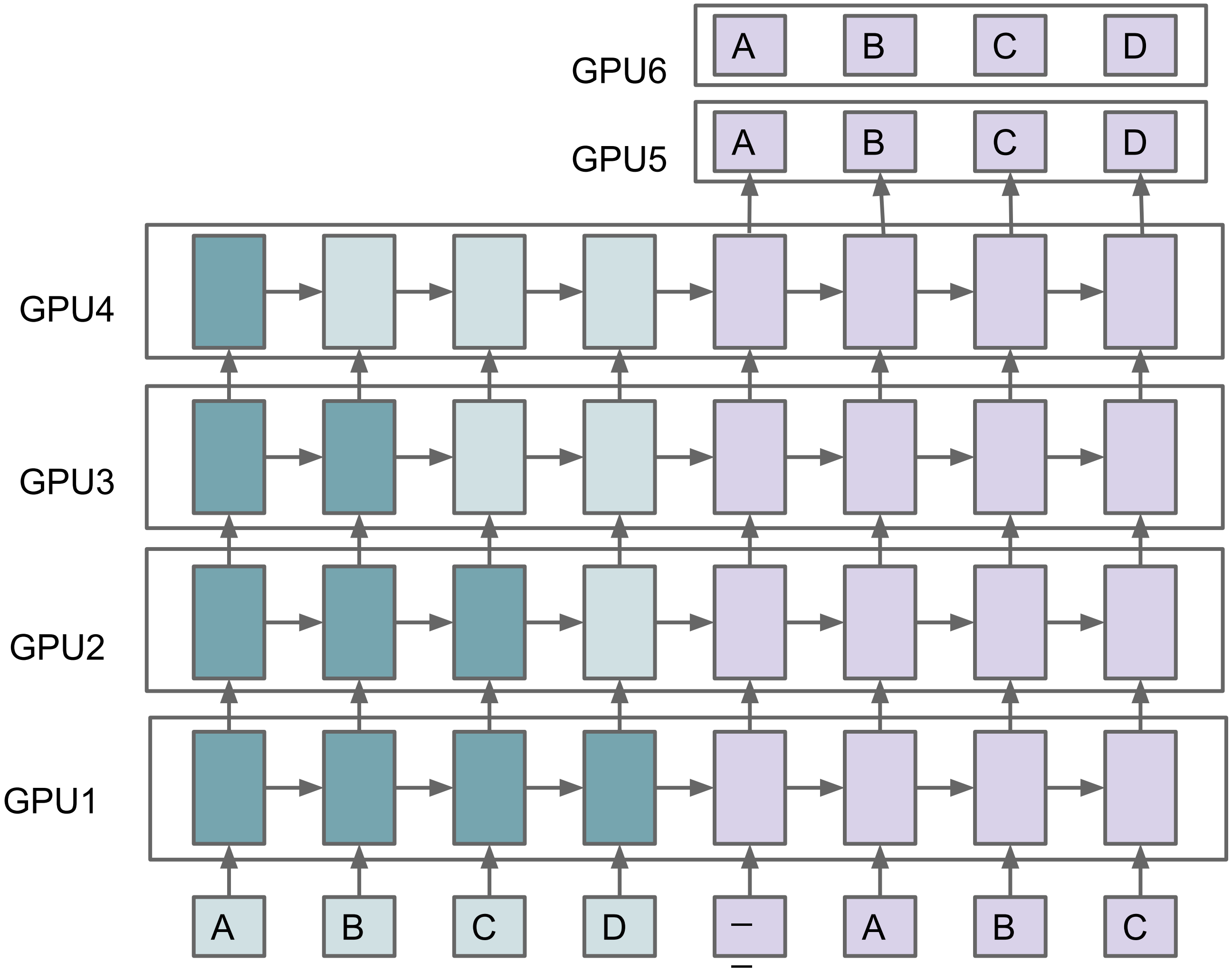


80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence

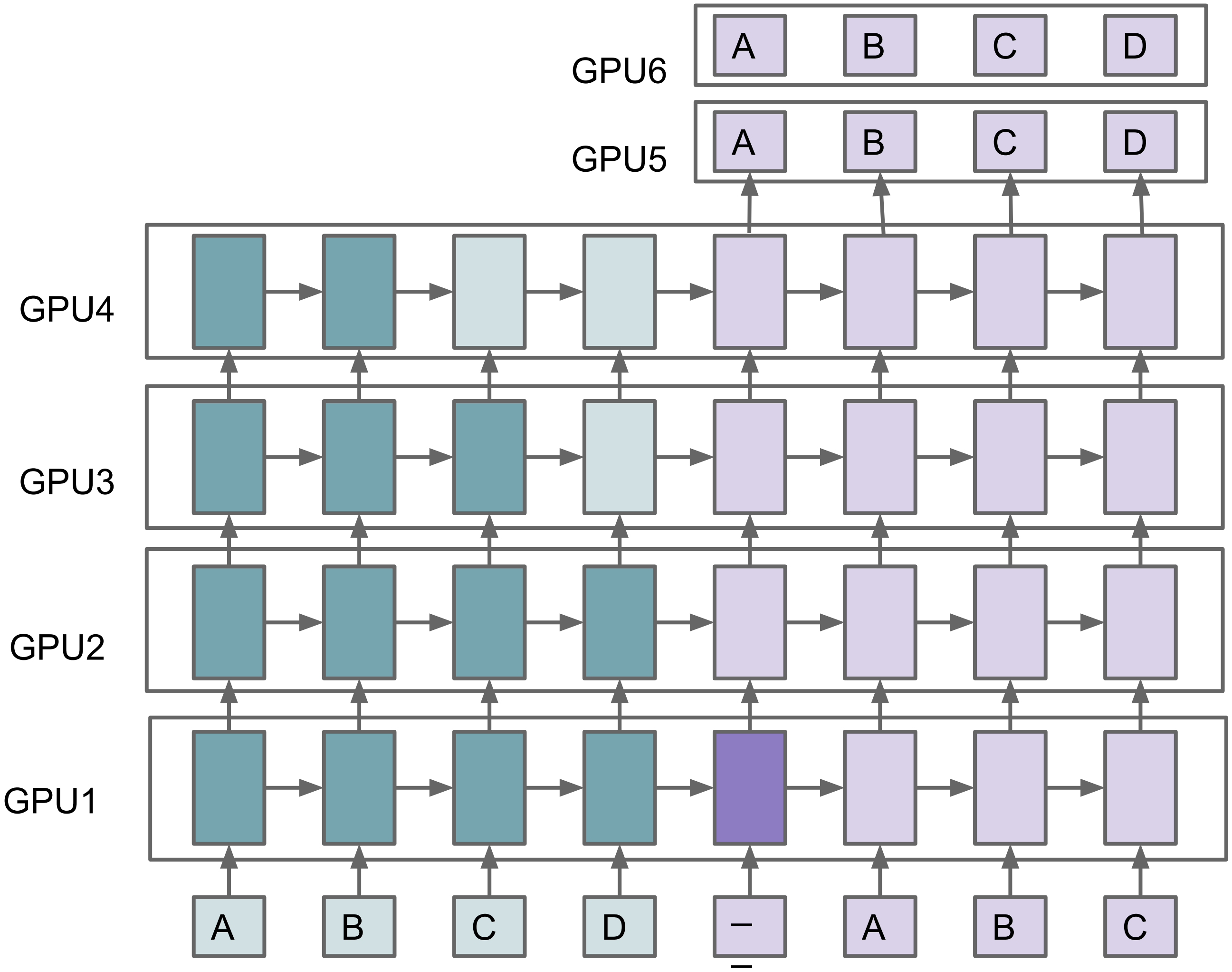


80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence

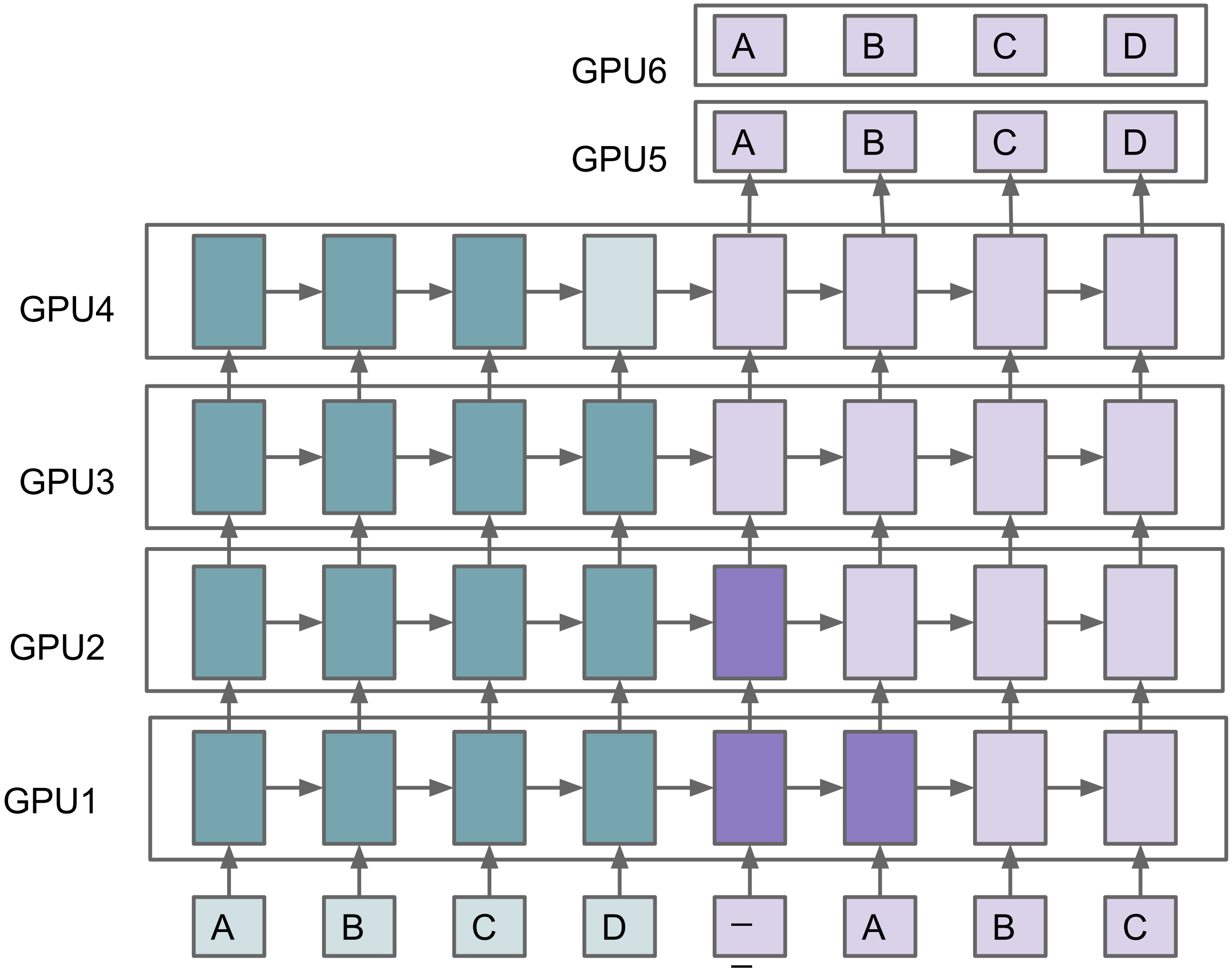


80k softmax by 1000 dims
This is very big!

Split softmax into 4 GPUs

1000 LSTM cells
2000 dims per timestep

2000 x 4 =
8k dims per sentence

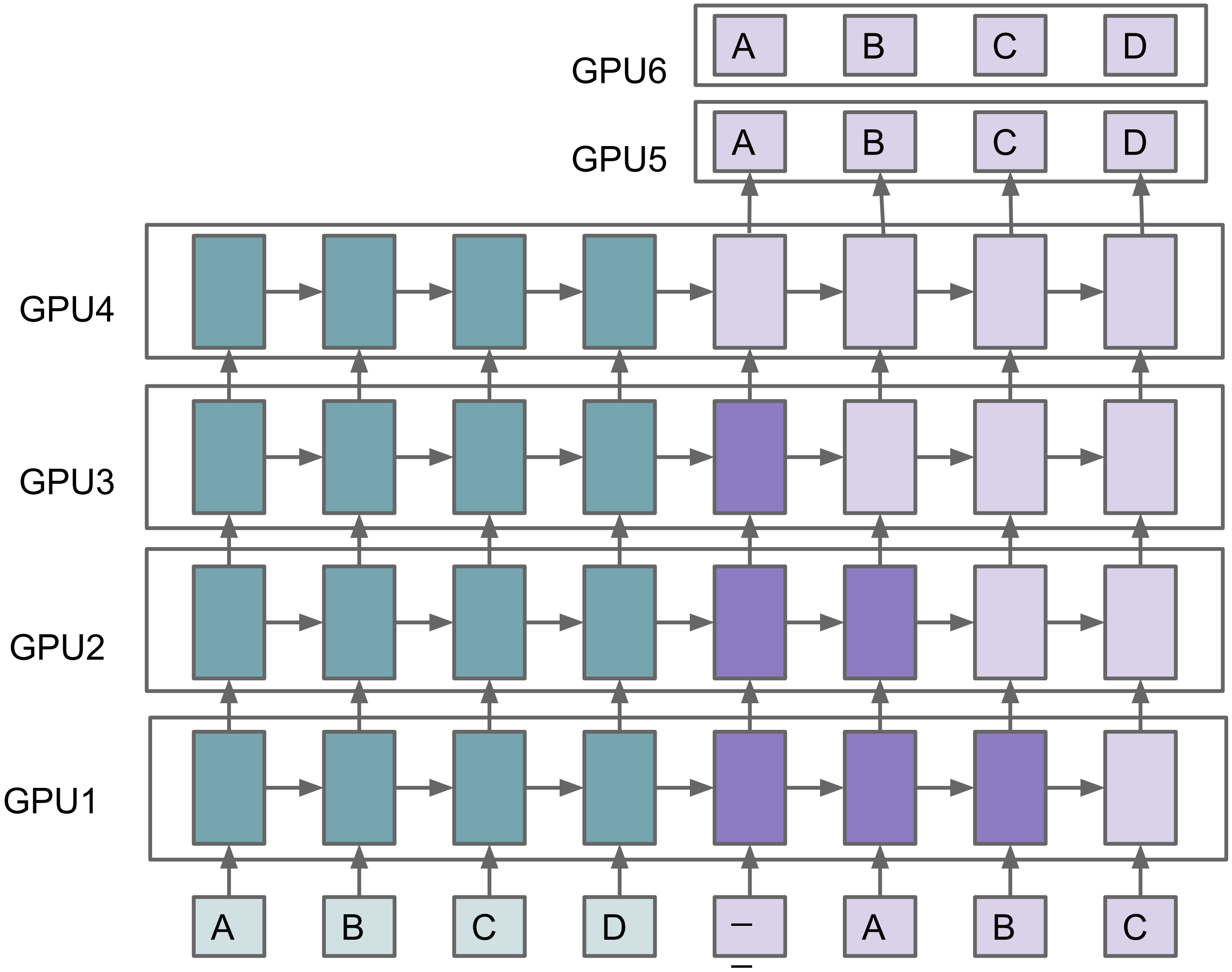


80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence

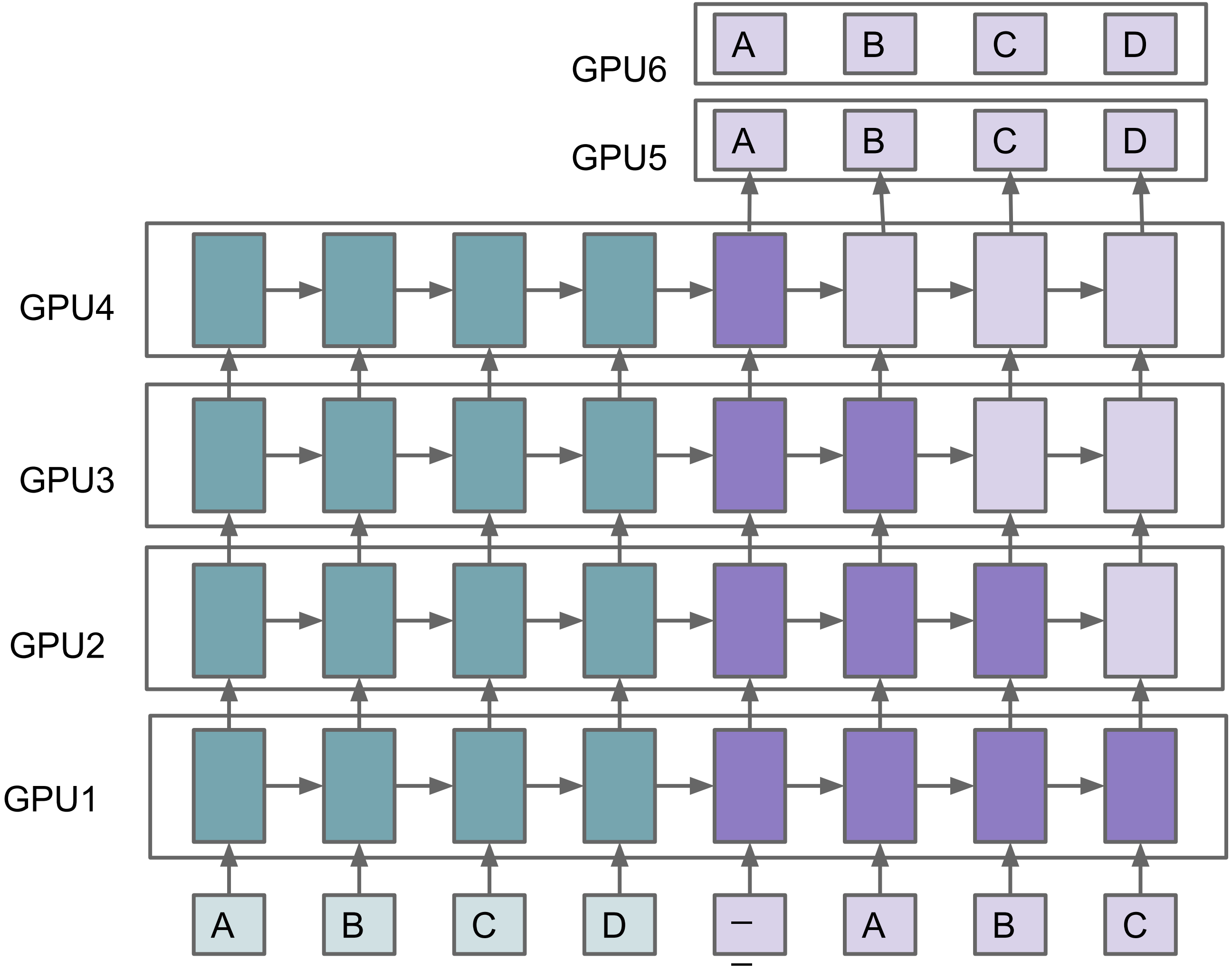


80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence

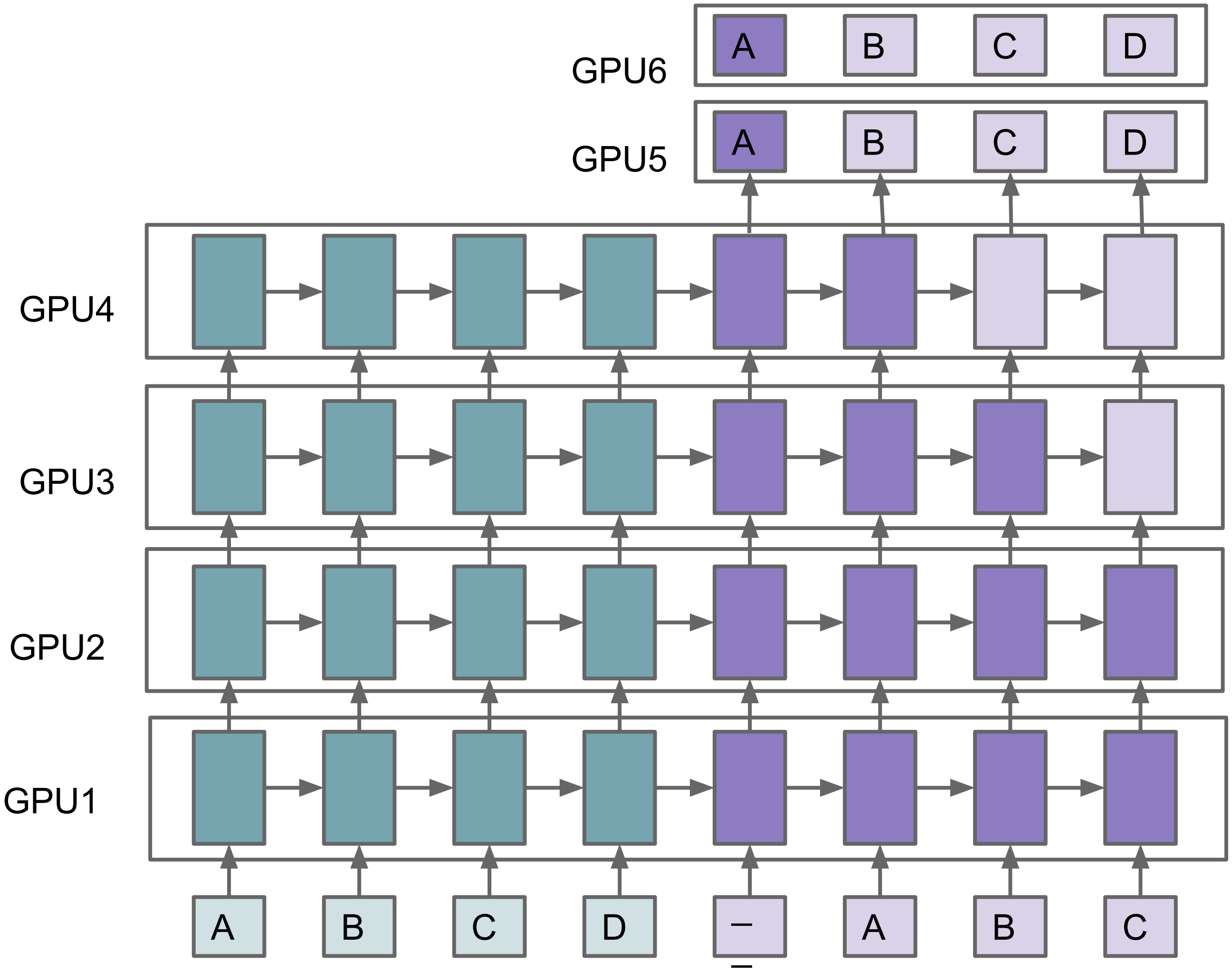


80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence

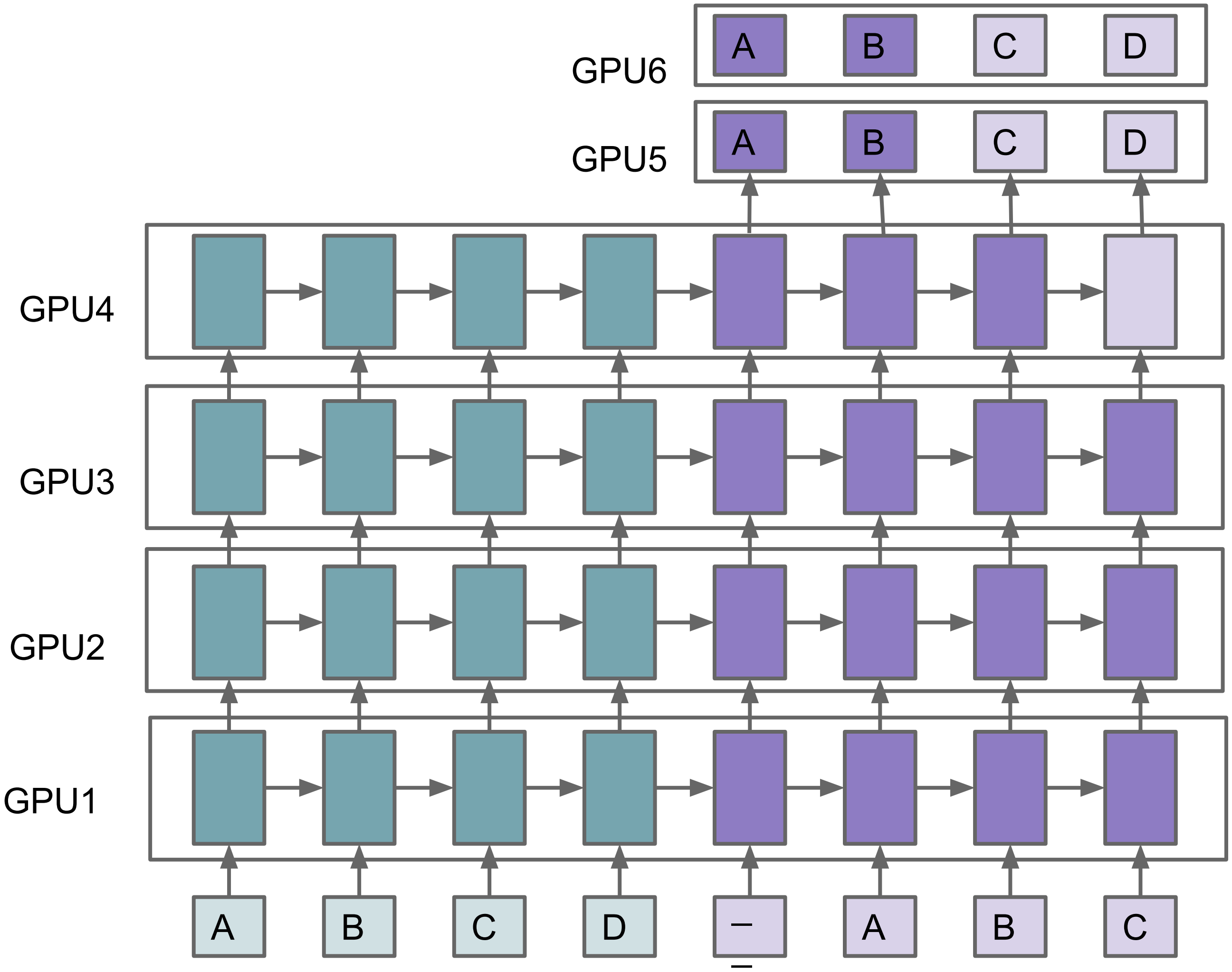


80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence

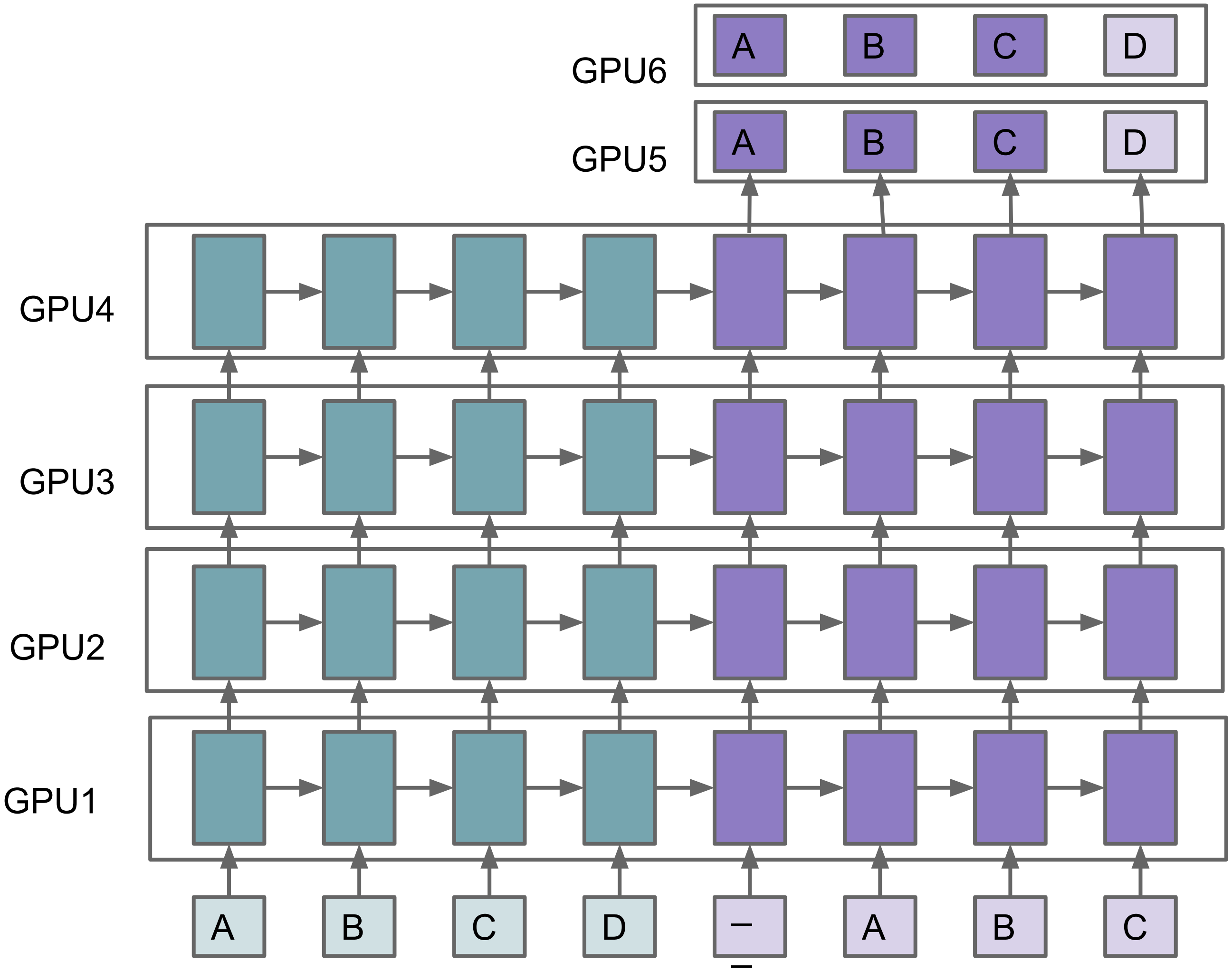


80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence

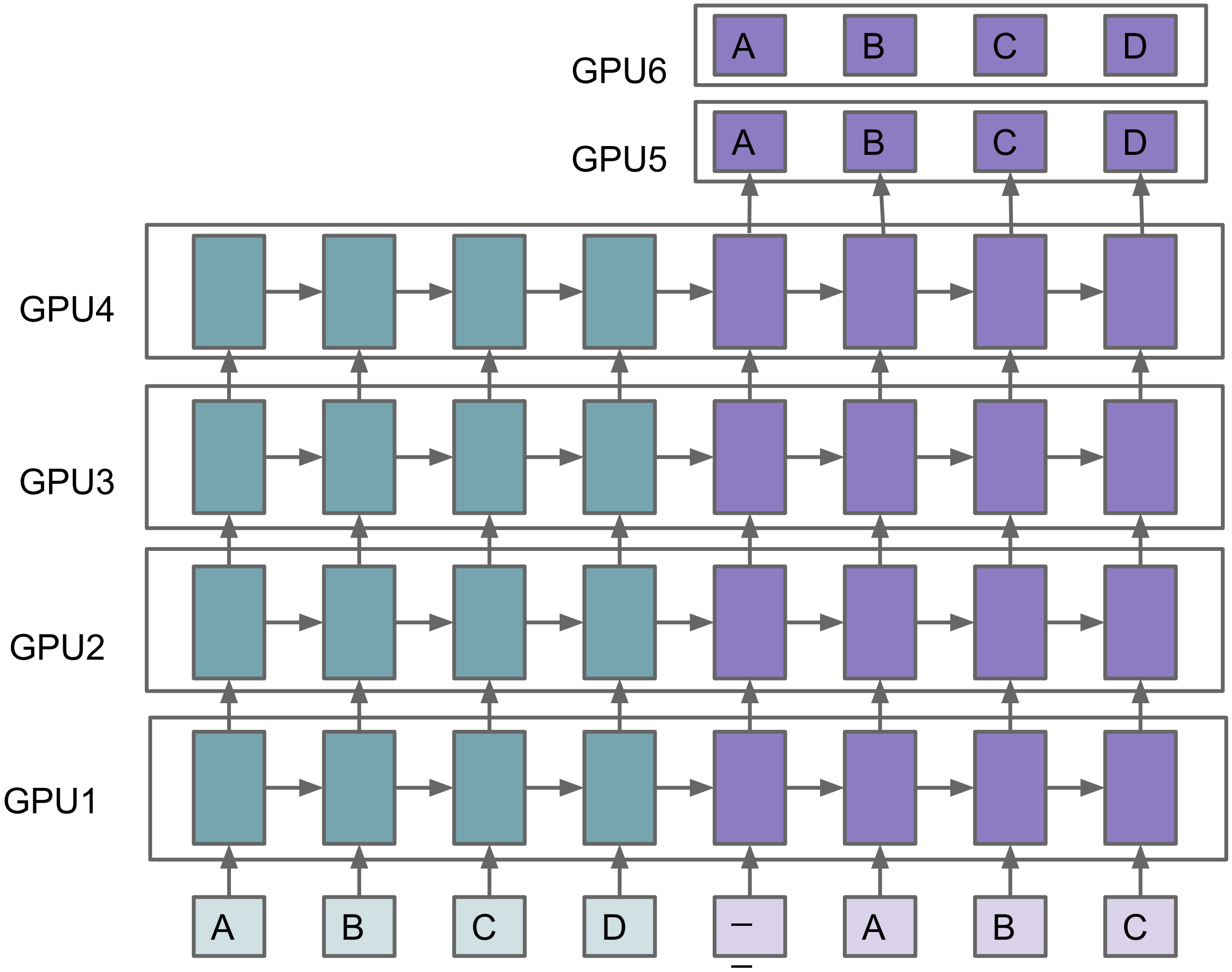


80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence



80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

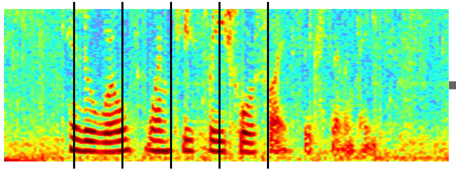
1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence

What are some ways that deep learning is having a significant impact at Google?

All of these examples implemented using TensorFlow
or our predecessor system

Speech Recognition



Acoustic Input

Deep
Recurrent
Neural Network

"How cold is it outside?"

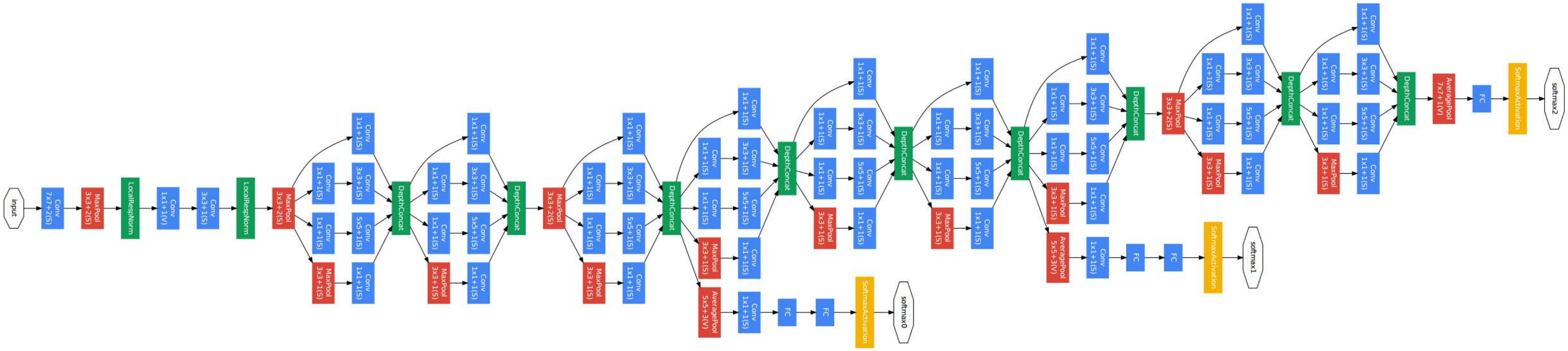
Text Output

Reduced word errors by more than 30%

Google Research Blog - August 2012, August 2015

Research at Google

The Inception Architecture (GoogLeNet, 2014)



Going Deeper with Convolutions

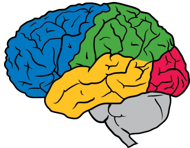
Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich

ArXiv 2014, CVPR 2015

Neural Nets: Rapid Progress in Image Recognition

Team	Year	Place	Error (top-5)
XRCE (pre-neural-net explosion)	2011	1st	25.8%
Supervision (AlexNet)	2012	1st	16.4%
Clarifai	2013	1st	11.7%
GoogLeNet (Inception)	2014	1st	6.66%
Andrej Karpathy (human)	2014	N/A	**5.1%**
BN-Inception (Arxiv)	2015	N/A	4.9%
Inception-v3 (Arxiv)	2015	N/A	3.46%

ImageNet challenge classification task



Google Photos Search

Your Photo

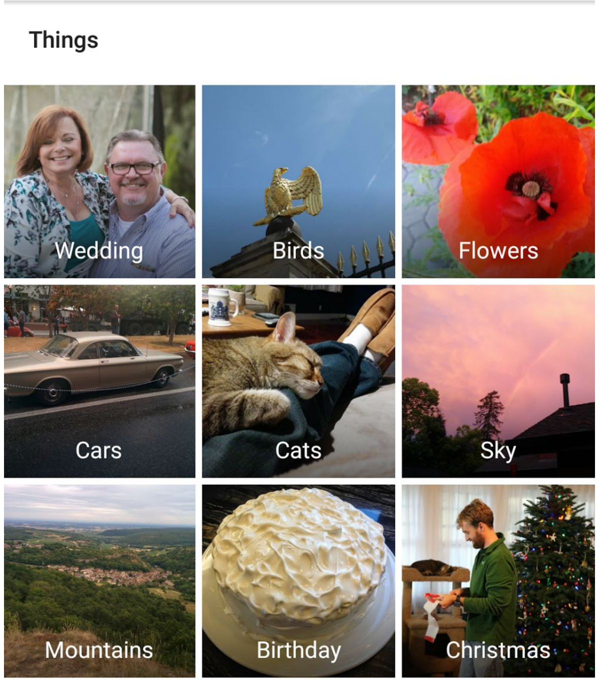
→ Deep Convolutional Neural Network → “ocean”

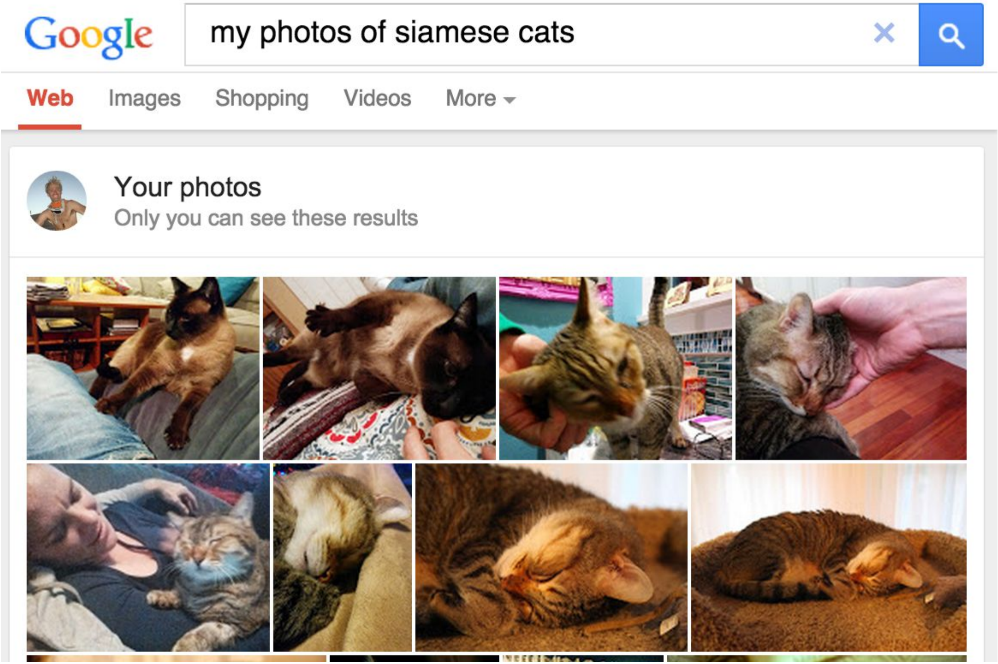
Automatic Tag

Search personal photos without tags.

Google Research Blog - June 2013

Google Photos Search





Reuse same model for completely different problems

Same basic model structure
trained on different data,
useful in **completely different contexts**

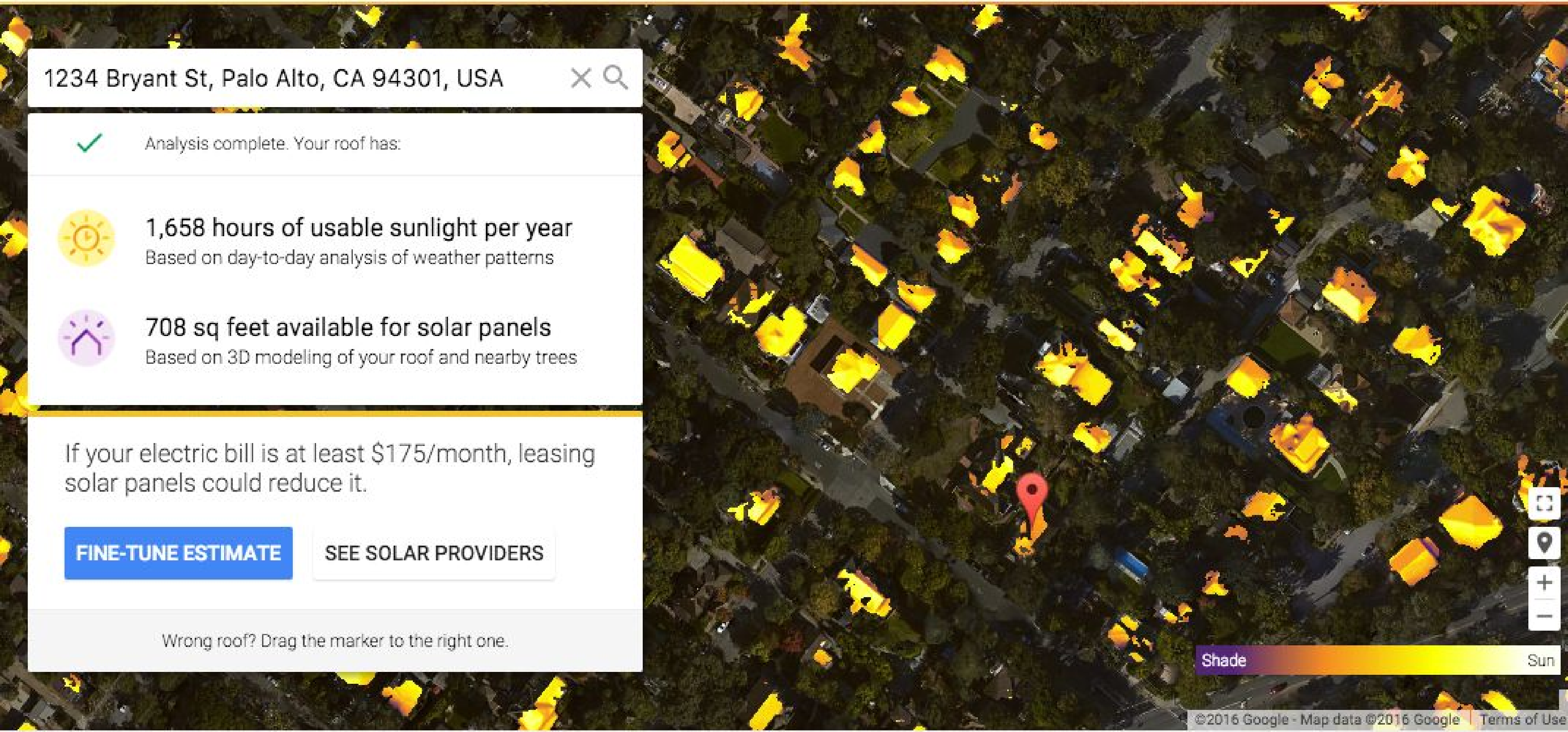
Example: given image → predict interesting pixels

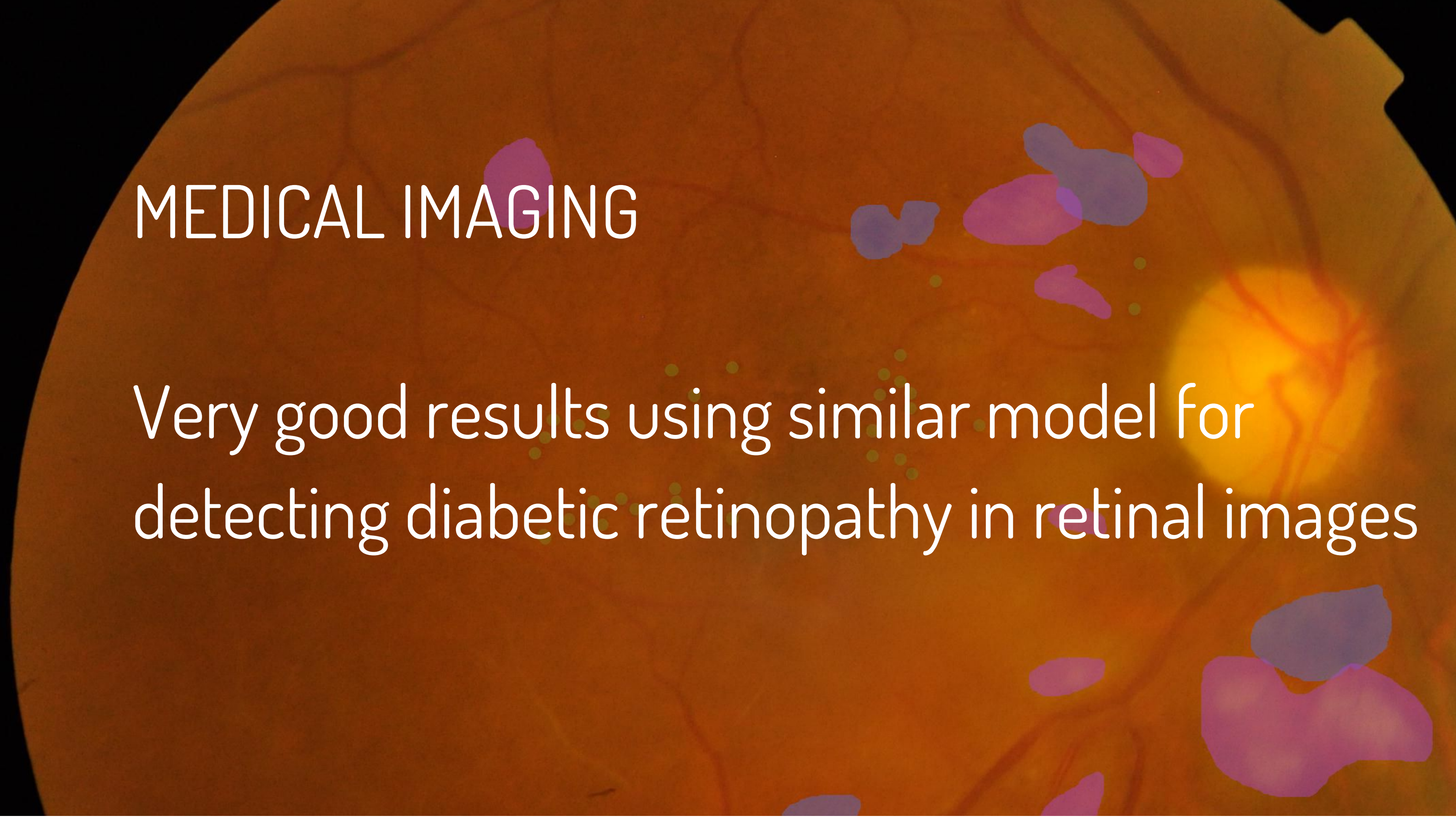

ASIAWIDE TRAVEL
(02) 9745 3355 1st Floor, 240 BURWOOD RD
aria's Bakery Inn 超羣餅屋
Maria's Bakery Inn 超羣餅屋


Corner
Cubbyhouse
CIANO MOTOR ENGINEERS
MECHANICAL REPAIRS TO ALL MAKES AND MODELS
Specialising In BMW, MINI & TOYOTA
8 REGATTA ROAD FIVE DOCK 9745 3173
88
LATEST DIAGNOSTIC EQUIPMENT • REGO INSPECTIONS
NEW CAR/LOGBOOK SERVICING • BRAKES • CLUTCHES
STEERING • SUSPENSION • TYRES • WHEEL ALIGNMENTS
FUEL INJECTION SERVICING • BATTERIES • AUTO ELECTRICAL
Factory Trained Technicians

Google Project Sunroof

www.google.com/sunroof




MEDICAL IMAGING
Very good results using similar model for detecting diabetic retinopathy in retinal images

“Seeing” Go

BBC | News | Sport | Weather | iPlayer | TV | Radio | More | Search

NEWS

Find local news

Home | UK | World | Business | Politics | Tech | Science | Health | Education | Entertainment & Arts | More

Technology

Google achieves AI 'breakthrough' at Go

An artificial intelligence program developed by Google beats Europe's top player at the ancient Chinese game of Go, about a decade earlier than expected.

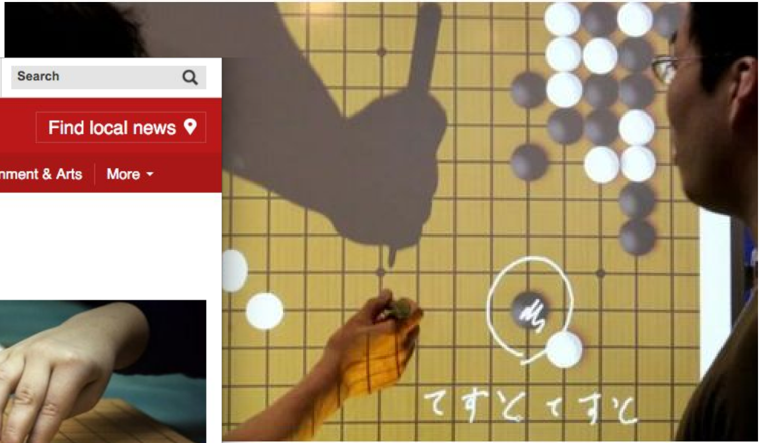
27 January 2016 | Technology

- How did they do it?
- What is the game Go?

Facebook trains AI to beat humans at Go

Google’s AI just cracked the game that supposedly no computer could beat

By Mike Murphy | January 27, 2016



Kiyoshi Ota)

…wly started to encroach on activities we previously …rilliantly sophisticated human brain could handle. …percomputer beat Grand Master Garry Kasparov at chess in 1997, and in 2011 IBM’s Watson beat former human winners at the quiz game *Jeopardy*. But the ancient board game Go has long been one of the major goals of artificial intelligence research. It’s understood to be one of the most difficult games for computers to handle due to the sheer number of possible moves a player can make at any given point. Until now, that is.

RankBrain in Google Search Ranking

Query: "car parts for sale",
Doc: "Rebuilt transmissions …"

Query & document features

→ Deep Neural Network →

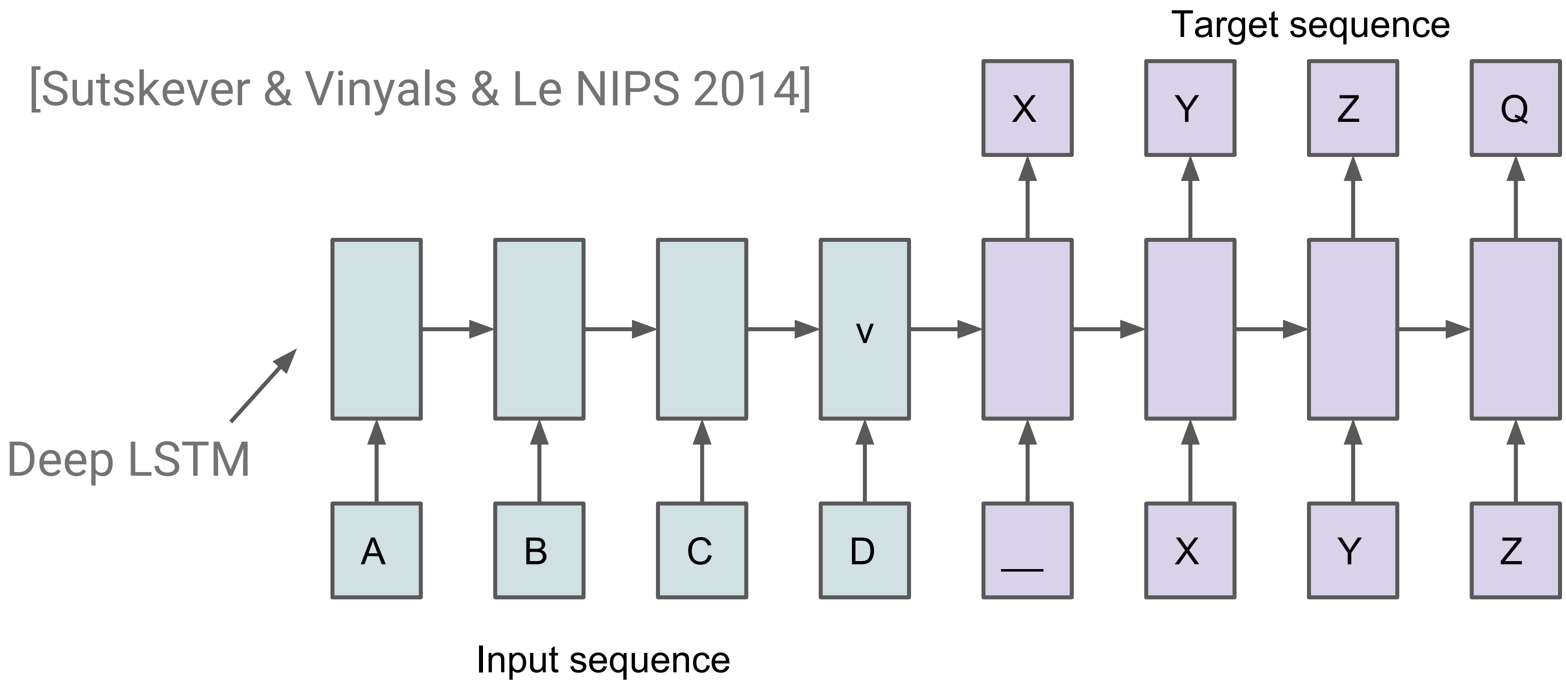
Score for doc,query pair

Launched in 2015
Third most important search ranking signal (of 100s)

Bloomberg, Oct 2015: "*Google Turning Its Lucrative Web Search Over to AI Machines*"

Sequence-to-Sequence Model

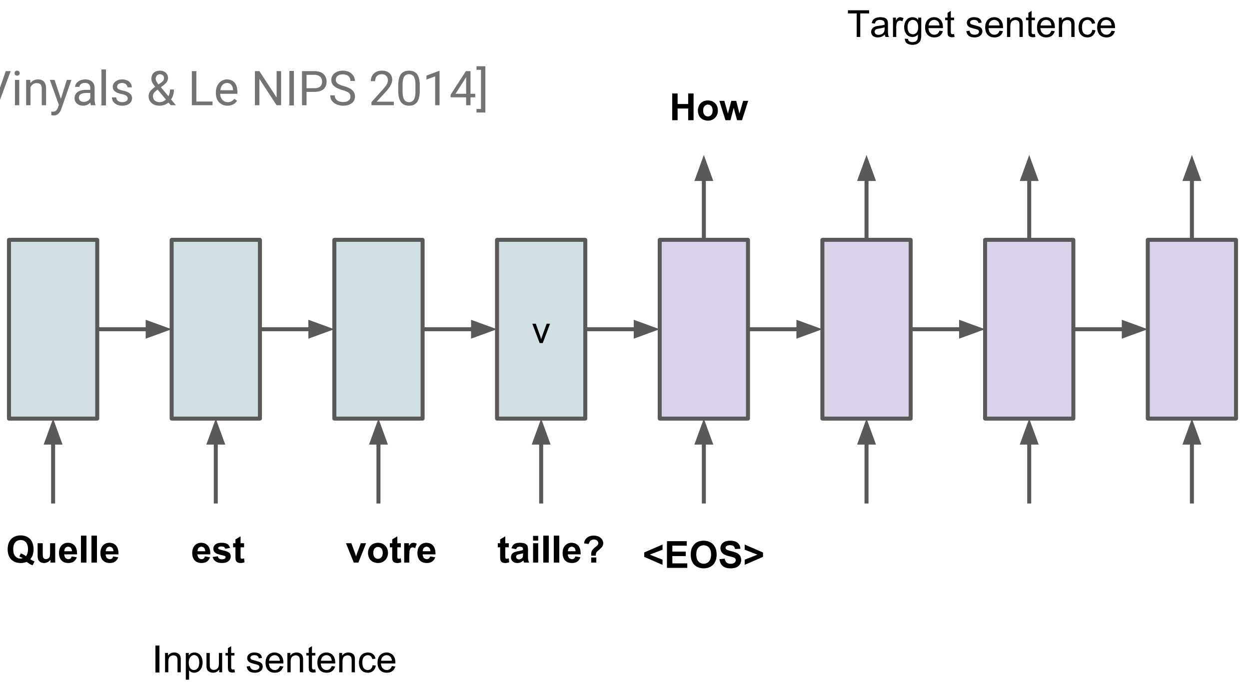
[Sutskever & Vinyals & Le NIPS 2014]



$$P(y_1, \ldots, y_{T'} | x_1, \ldots, x_T) = \prod_{t=1}^{T'} p(y_t | v, y_1, \ldots, y_{t-1})$$

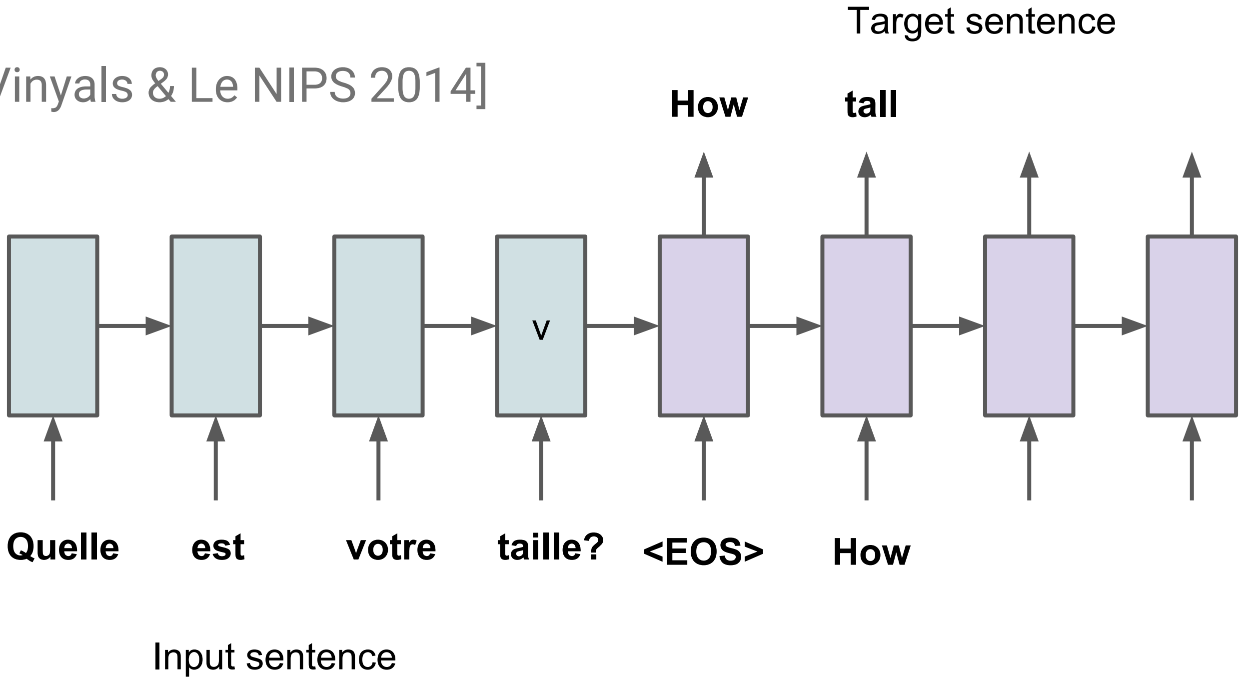
Sequence-to-Sequence Model: Machine Translation

[Sutskever & Vinyals & Le NIPS 2014]



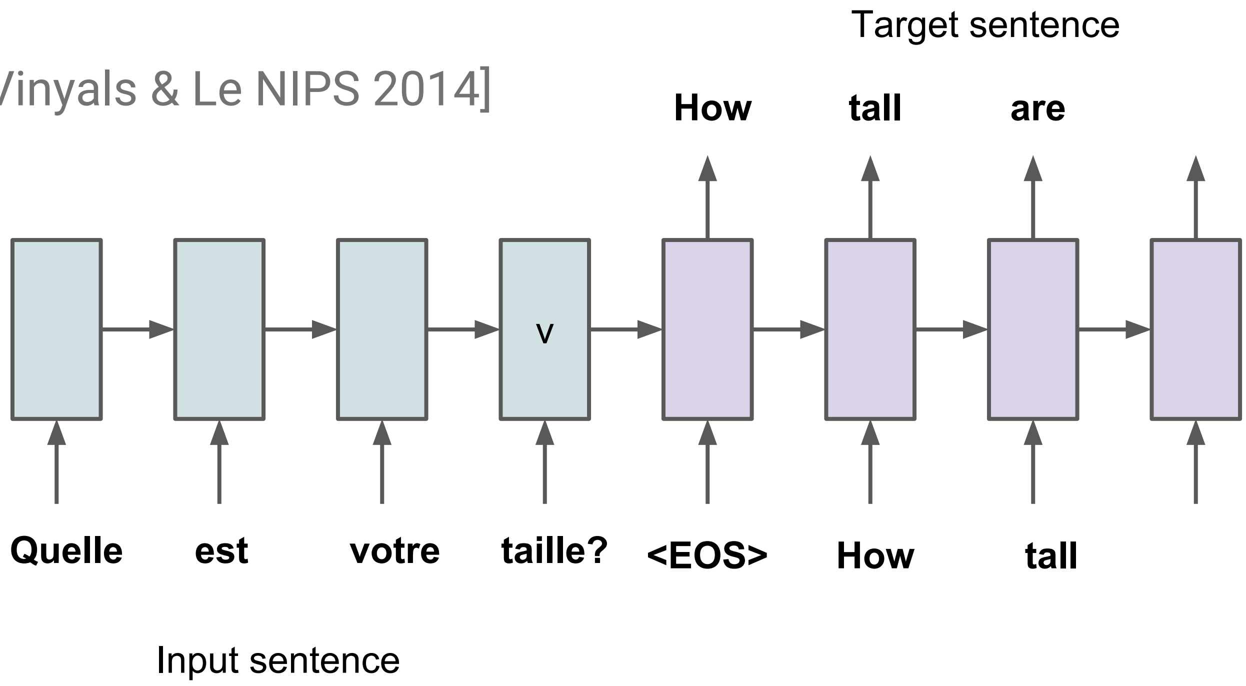
Sequence-to-Sequence Model: Machine Translation

[Sutskever & Vinyals & Le NIPS 2014]



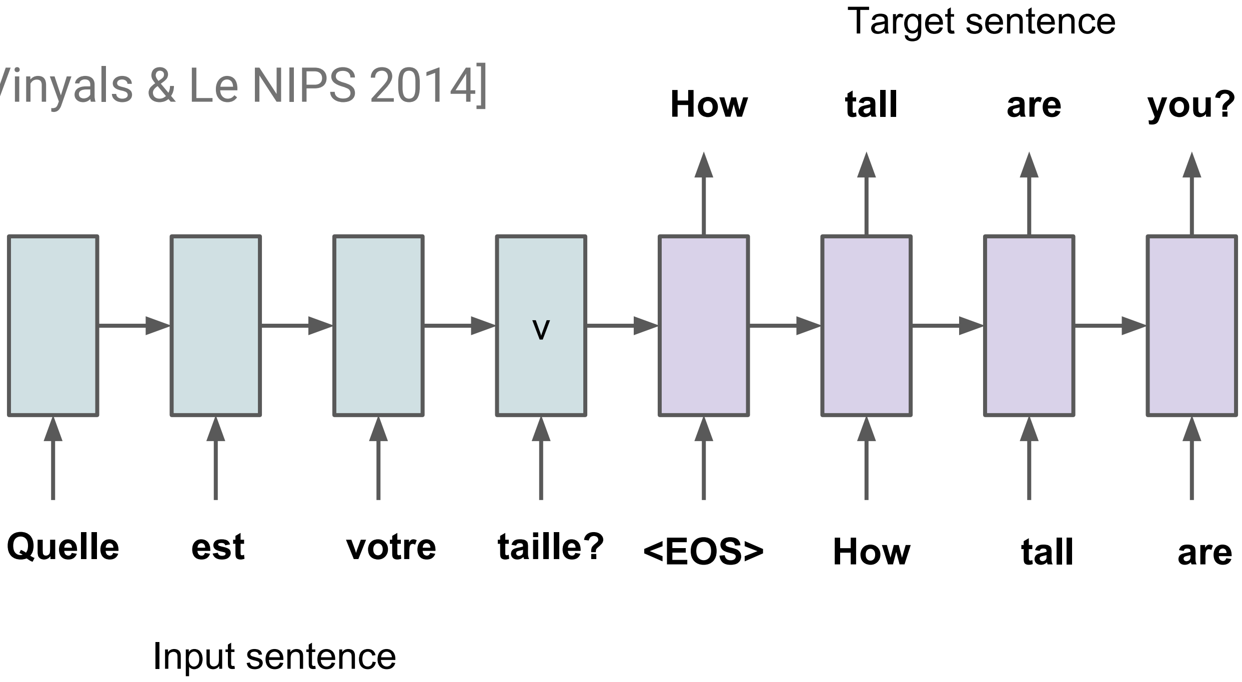
Sequence-to-Sequence Model: Machine Translation

[Sutskever & Vinyals & Le NIPS 2014]



Sequence-to-Sequence Model: Machine Translation

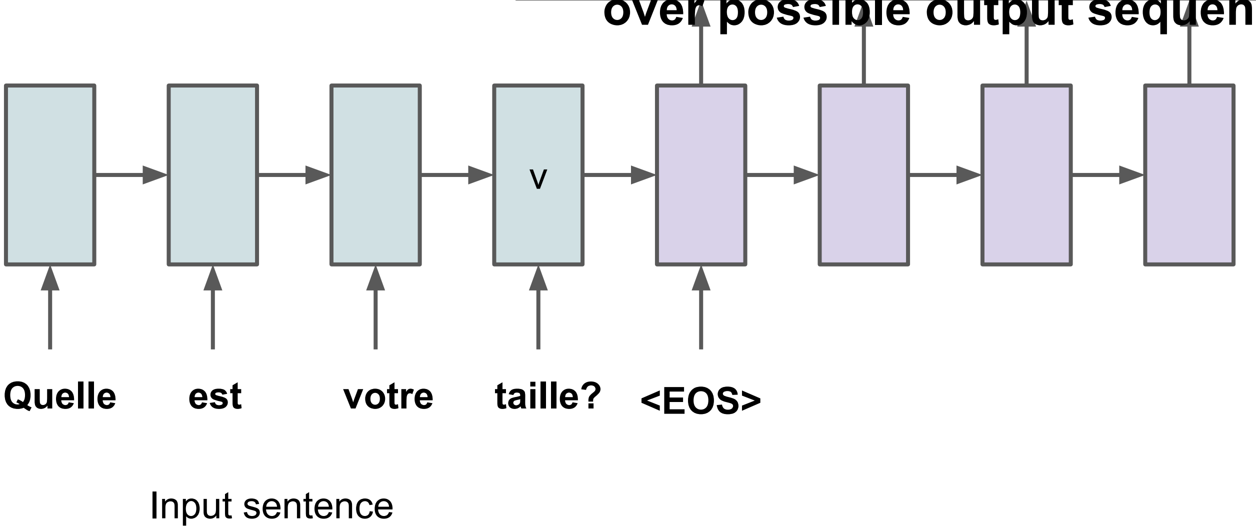
[Sutskever & Vinyals & Le NIPS 2014]



Sequence-to-Sequence Model: Machine Translation

[Sutskever & Vinyals & Le NIPS 2014]

At inference time:
Beam search to choose most probable
over possible output sequences



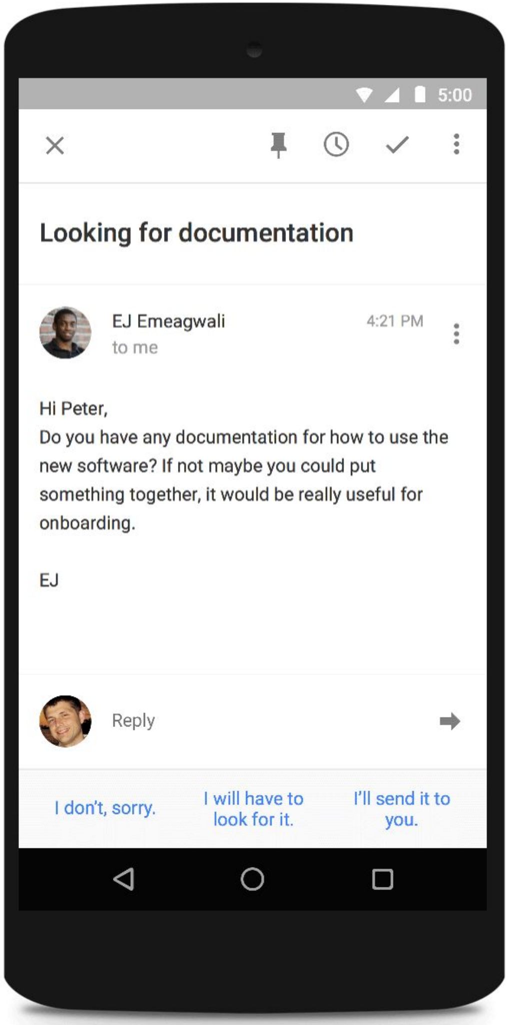
Inbox
by Gmail

Smart Reply

April 1, 2009: April Fool's Day joke

Nov 5, 2015: Launched Real Product

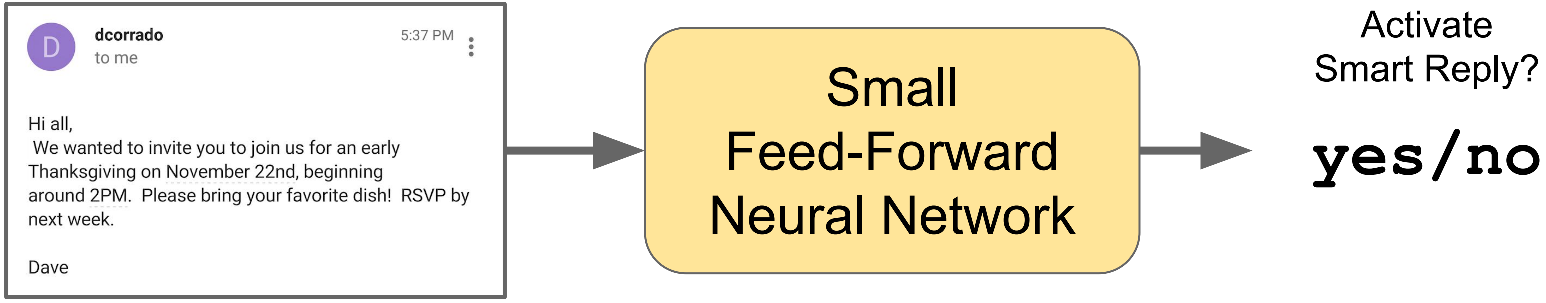
Feb 1, 2016: >10% of mobile Inbox replies



Smart Reply

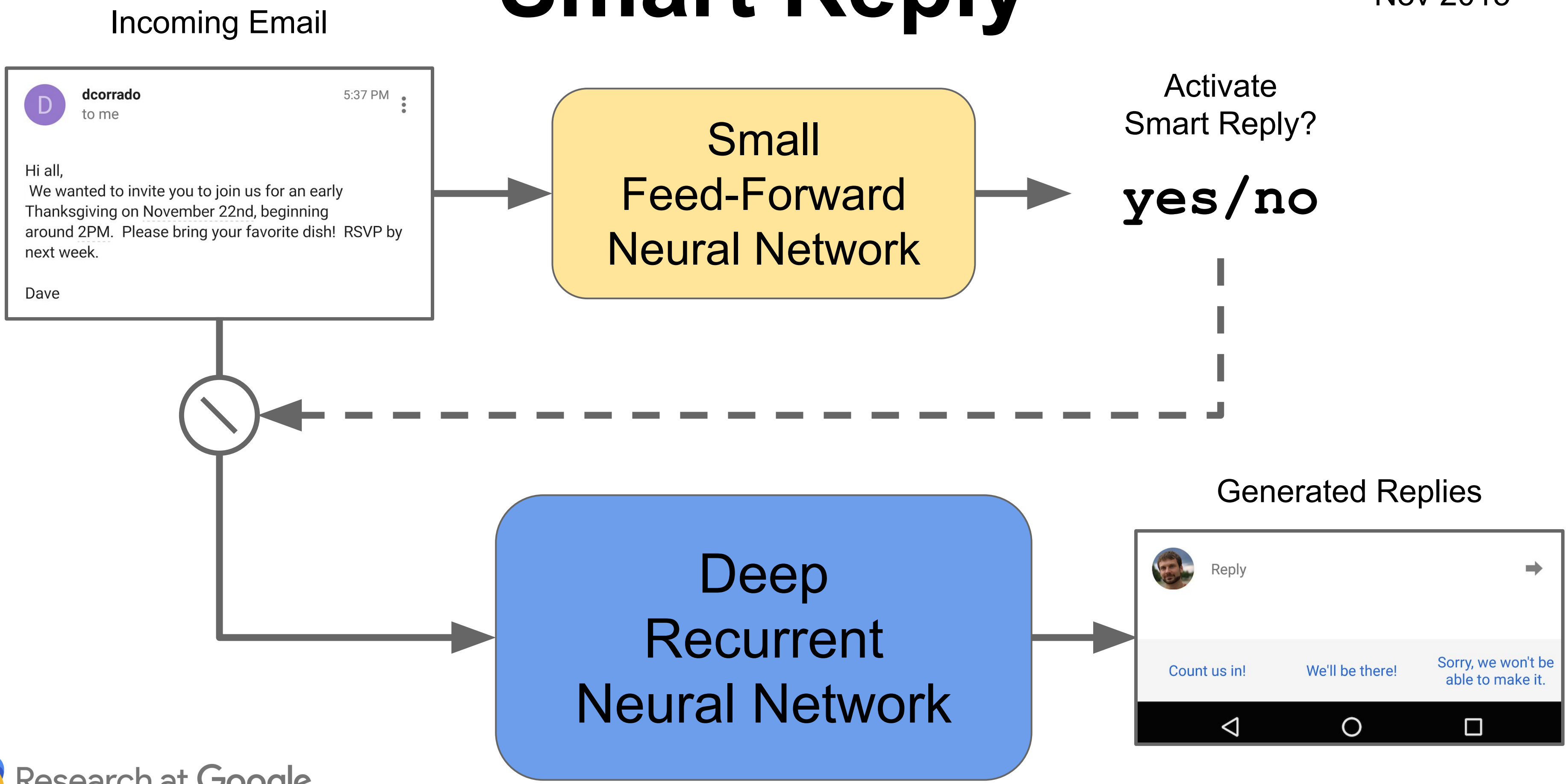
Google Research Blog
- Nov 2015

Incoming Email



Smart Reply

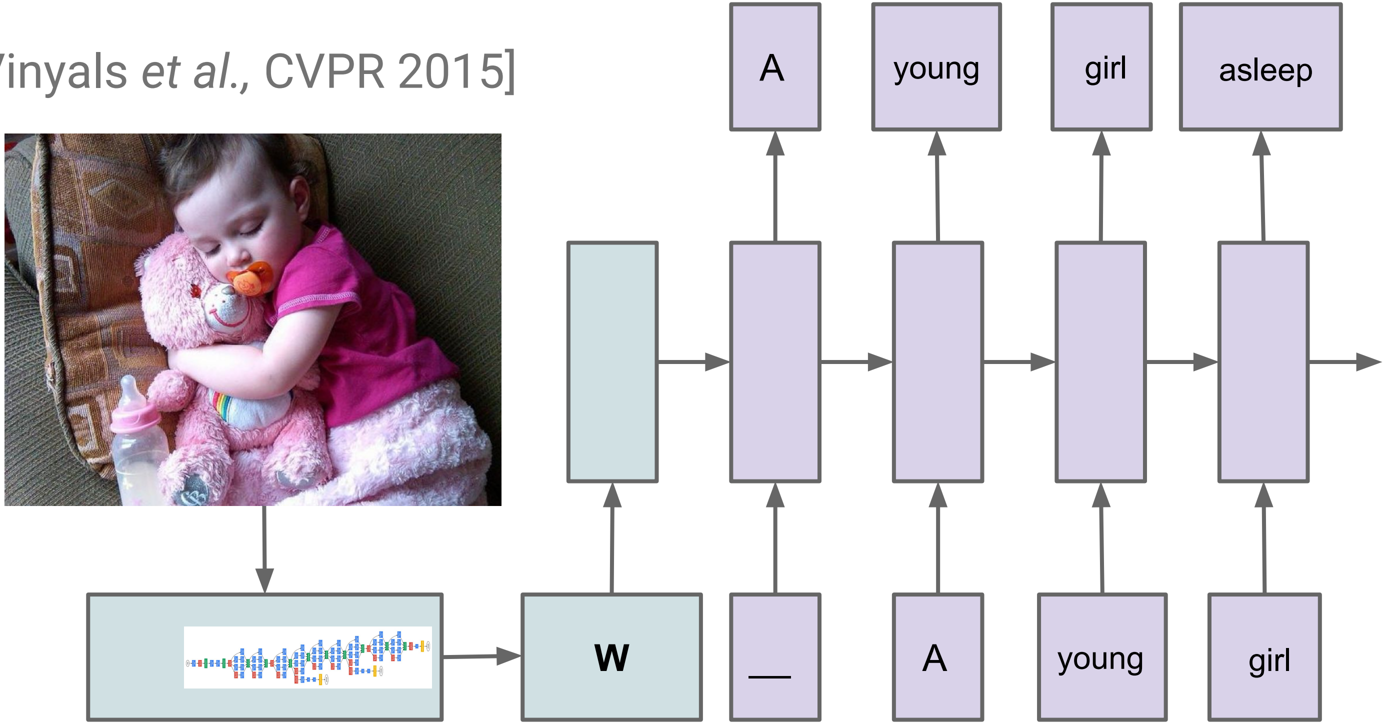
Google Research Blog
- Nov 2015



Research at Google

Image Captioning

[Vinyals *et al.*, CVPR 2015]



$$\theta^\star = \arg\max_\theta p(S|I)$$

Image Captions Research

Human: A young girl asleep on the sofa cuddling a stuffed bear.

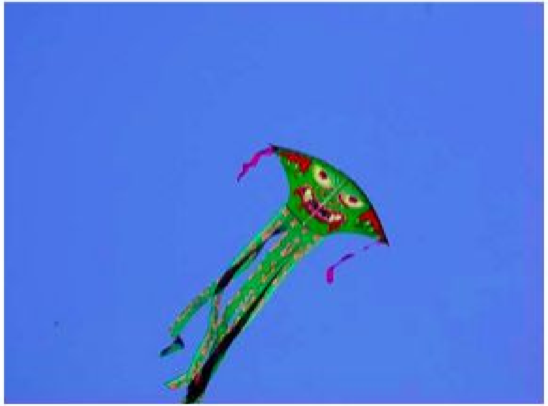
Model: A close up of a child holding a stuffed animal.

Model: A baby is asleep next to a teddy bear.

A man holding a tennis racquet on a tennis court.

Two pizzas sitting on top of a stove top oven

A group of young people playing a game of Frisbee



A man flying through the air while riding a snowboard

Combining Vision with Robotics

“Deep Learning for Robots: Learning from Large-Scale Interaction”, Google Research Blog, March, 2016

“Learning Hand-Eye Coordination for Robotic Grasping with Deep Learning and Large-Scale Data Collection”, Sergey Levine, Peter Pastor, Alex Krizhevsky, & Deirdre Quillen, Arxiv, arxiv.org/abs/1603.02199

How Can You Get Started with Machine Learning?

Three ways, with varying complexity:

(1) Use a Cloud-based API (Vision, Speech, etc.)
(2) Use an existing model architecture, and retrain it or fine tune on your dataset
(3) Develop your own machine learning models for new problems

More flexible, but more effort required

Use Cloud-based APIs

GOOGLE TRANSLATE API
Dynamically translate between thousands of available language pairs

cloud.google.com/translate

CLOUD SPEECH API ALPHA
Speech to text conversion powered by machine learning

cloud.google.com/speech

CLOUD VISION API
Derive insight from images with our powerful Cloud Vision API

cloud.google.com/vision

CLOUD TEXT API ALPHA
Use Cloud Text API for sentiment analysis and entity recognition in a piece of text.

cloud.google.com/text

Use Cloud-based APIs

GOOGLE TRANSLATE API
Dynamically translate between thousands of available language pairs

cloud.google.com/translate

CLOUD SPEECH API ALPHA
Speech to text conversion powered by machine learning

cloud.google.com/speech

CLOUD VISION API
Derive insight from images with our powerful Cloud Vision API

cloud.google.com/vision

CLOUD TEXT API ALPHA
Use Cloud Text API for sentiment analysis and entity recognition in a piece of text.

cloud.google.com/text

Google Cloud Vision API

https://cloud.google.com/vision/

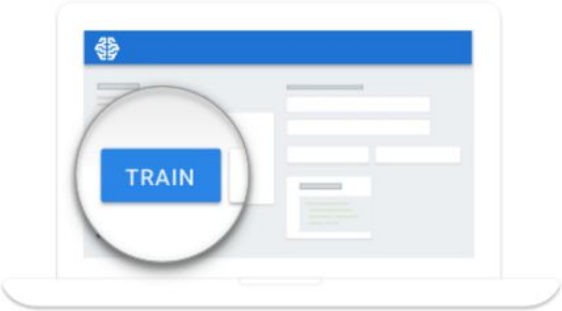
Google Cloud ML
Scaled service for training and inference w/TensorFlow

CLOUD MACHINE LEARNING ALPHA

Machine Learning on any data, of any size

SIGN UP FOR THE LIMITED PREVIEW

Managed scalable machine learning platform



Google Cloud Machine Learning is a managed platform that enables you to easily build machine learning models, that work on any type of data, of any size. Create your model with the powerful TensorFlow framework, that powers many Google products from Google Photos, to Google Cloud Speech. Build models of any size with our

A Few TensorFlow Community Examples
(From more than 2000 results for `'tensorflow'` on GitHub)

- DQN: github.com/nivwusquorum/tensorflow-deepq
- NeuralArt: github.com/woodrush/neural-art-tf
- Char RNN: github.com/sherjilozair/char-rnn-tensorflow
- Keras ported to TensorFlow: github.com/fchollet/keras
- Show and Tell: github.com/jazzsaxmafia/show_and_tell.tensorflow
- Mandarin translation: github.com/jikexueyuanwiki/tensorflow-zh

...

A Few TensorFlow Community Examples
(From more than ~~2000~~ 2100 results for `‘tensorflow’` on GitHub)

- DQN: github.com/nivwusquorum/tensorflow-deepq
- NeuralArt: github.com/woodrush/neural-art-tf
- Char RNN: github.com/sherjilozair/char-rnn-tensorflow
- Keras ported to TensorFlow: github.com/fchollet/keras
- Show and Tell: github.com/jazzsaxmafia/show_and_tell.tensorflow
- Mandarin translation: github.com/jikexueyuanwiki/tensorflow-zh

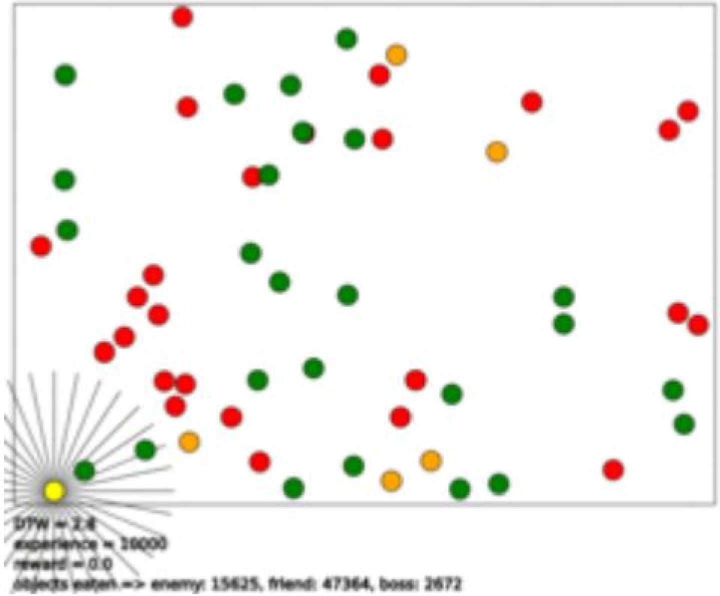
...

github.com/nivwusquorum/tensorflow-deepq

Reinforcement Learning using Tensor Flow

Quick start

Check out Karpathy game in `notebooks` folder.



The image above depicts a strategy learned by the DeepQ controller. Available actions are accelerating top, bottom, left or right. The reward signal is +1 for the green fellas, -1 for red and -5 for orange.

github.com/woodrush/neural-art-tf

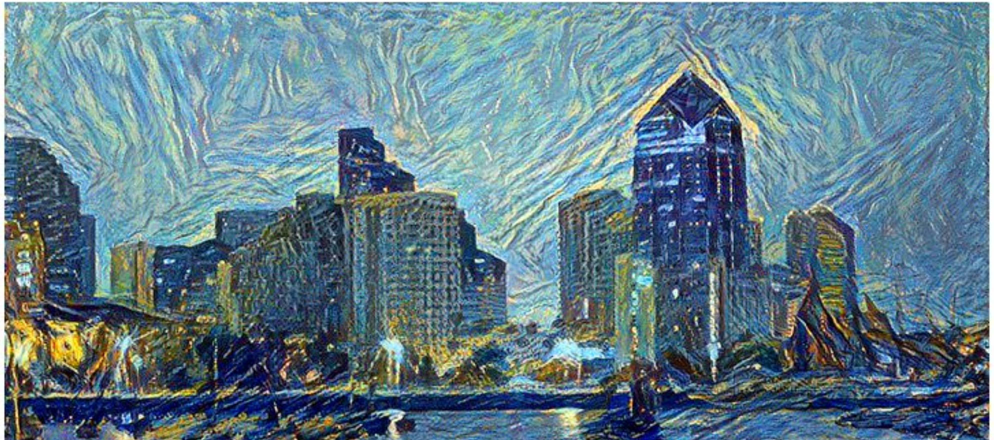
"Neural Art" in TensorFlow

An implementation of "A neural algorithm of Artistic style" in TensorFlow, for

- Introductory, hackable demos for TensorFlow, and
- Demonstrating the use of importing various Caffe cnn models (VGG and illustration2vec) in TF.

In this work, I put effort in putting the code simple as possible, for being a good introductory code to TF. For this reason, I also implemented very basic uses of TensorBoard (the visualizer). I also aimed on demonstrating the use of importing various Caffe models from *.caffemodel files into TensorFlow, especially models that seemed not to be imported by anybody yet in TF (as far as I know). Based on https://github.com/ethereon/caffe-tensorflow, I modified the importer so that it can import illustration2vec (http://illustration2vec.net/), which is another CNN available as a Caffe model. Using different CNNs yields different results, which reflects the characteristics of the model.

In the Neural Art problem setting, the weights of the CNN are fixed, and the input image into the CNN is the only "trainable" variable, making the code easy to understand (the optimized/trained image is the output image). I hope this example serves as a good introduction to TensorFlow as well as for entertainment purposes.



github.com/sherjilozair/char-rnn-tensorflow

char-rnn-tensorflow

Multi-layer Recurrent Neural Networks (LSTM, RNN) for character-level language models in Python using Tensorflow.

Inspired from Andrej Karpathy's char-rnn.

Requirements

- Tensorflow

Basic Usage

To train with default parameters on the tinyshakespeare corpus, run `python train.py` .

To sample from a checkpointed model, `python sample.py` .

github.com/fchollet/keras

Keras: Deep Learning library for Theano and TensorFlow

You have just found Keras.

Keras is a minimalist, highly modular neural networks library, written in Python and capable of running either on top of either TensorFlow or Theano. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result with the least possible delay is key to doing good research.

Use Keras if you need a deep learning library that:

- allows for easy and fast prototyping (through total modularity, minimalism, and extensibility).
- supports both convolutional networks and recurrent networks, as well as combinations of the two.
- supports arbitrary connectivity schemes (including multi-input and multi-output training).
- runs seamlessly on CPU and GPU.

Read the documentation at Keras.io.

Keras is compatible with: - **Python 2.7-3.5** with the Theano backend - **Python 2.7** with the TensorFlow backend github.com/jazzsaxmafia/show_and_tell.tensorflow

Neural Caption Generator

- Implementation of "Show and Tell" http://arxiv.org/abs/1411.4555
  - Borrowed some code and ideas from Andrej Karpathy's NeuralTalk.
- You need flickr30k data (images and annotations)

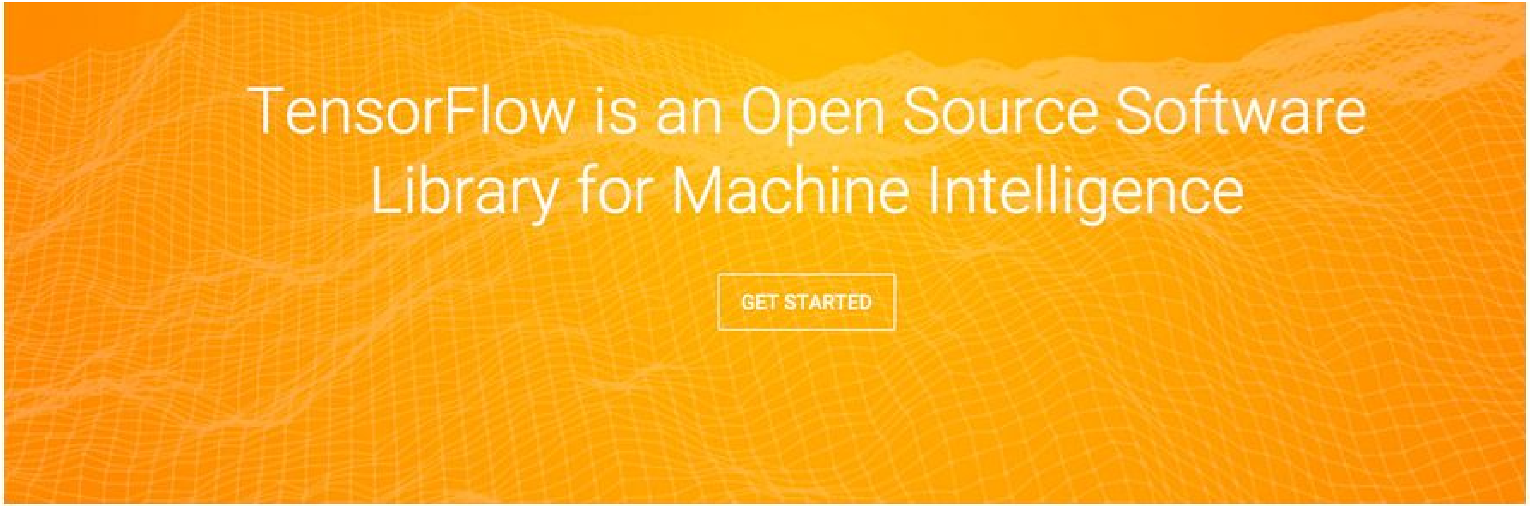
Code

- make_flickr_dataset.py : Extracting feats of flickr30k images, and save them in './data/feats.npy'
- model_tensorflow.py : TensorFlow Version
- model_theano.py : Theano Version

Usage

- Flickr30k Dataset Download
- Extract VGG Featues of Flicker30k images (make_flickr_dataset.py)
- Train: run train() in model_tensorflow.py or model_theano.py
- Test: run test() in model_tensorflow.py or model_theano.py.
  - parameters: VGG FC7 feature of test image, trained model path

github.com/jikexueyuanwiki/tensorflow-zh



你正在翻译的项目可能会比 Android 系统更加深远地影响着世界！

缘起

2015年11月9日，Google 官方在其博客上称，Google Research 宣布推出第二代机器学习系统 TensorFlow，针对先前的 DistBelief 的短板有了各方面的加强，更重要的是，它是开源的，任何人都可以用。

机器学习作为人工智能的一种类型，可以让软件根据大量的数据来对未来的情况进行阐述或预判。如今，领先的科技巨头无不在机器学习下予以极大投入。Facebook、苹果、微软，甚至国内的百度。Google 自然也在其中。「TensorFlow」是 Google

What Does the Future Hold?

Deep learning usage will continue to grow and accelerate:

- Across more and more fields and problems:
    - robotics, self-driving vehicles, ...
    - health care
    - video understanding
    - dialogue systems
    - personal assistance
    - ...

Conclusions

Deep neural networks are making significant strides in understanding:
In speech, vision, language, search, robotics, …

If you're not considering how to use deep neural nets to solve your vision or understanding problems, **you almost certainly should be**

Further Reading

- Dean, *et al.*, *Large Scale Distributed Deep Networks,* NIPS 2012, research.google.com/archive/large_deep_networks_nips2012.html.
- Mikolov, Chen, Corrado & Dean. *Efficient Estimation of Word Representations in Vector Space,* NIPS 2013, arxiv.org/abs/1301.3781.
- Sutskever, Vinyals, & Le, *Sequence to Sequence Learning with Neural Networks*, NIPS, 2014, arxiv.org/abs/1409.3215.
- Vinyals, Toshev, Bengio, & Erhan. *Show and Tell: A Neural Image Caption Generator*. CVPR 2015. arxiv.org/abs/1411.4555
- TensorFlow white paper, tensorflow.org/whitepaper2015.pdf (clickable links in bibliography)

g.co/brain (We're hiring! Also check out Brain Residency program at g.co/brainresidency)
www.tensorflow.org
research.google.com/people/jeff
research.google.com/pubs/BrainTeam.html

Questions?