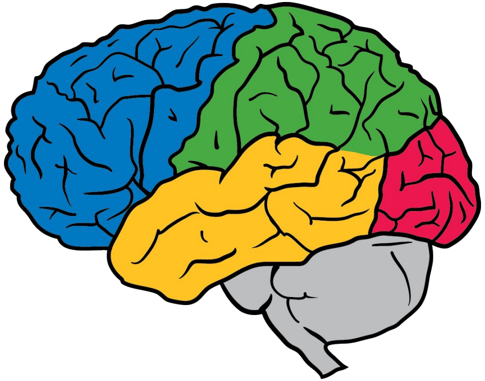


Large-Scale Deep Learning with TensorFlow for Building Intelligent Systems

Jeff Dean
Google Brain Team
g.co/brain

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

We can now store and perform computation on large datasets, using things like MapReduce, BigTable, Spanner, Flume, Pregel, or open-source variants like Hadoop, HBase, Cassandra, Giraph, ...

But what we really want is not just raw data,
but computer systems that **understand** this data

Where are we?

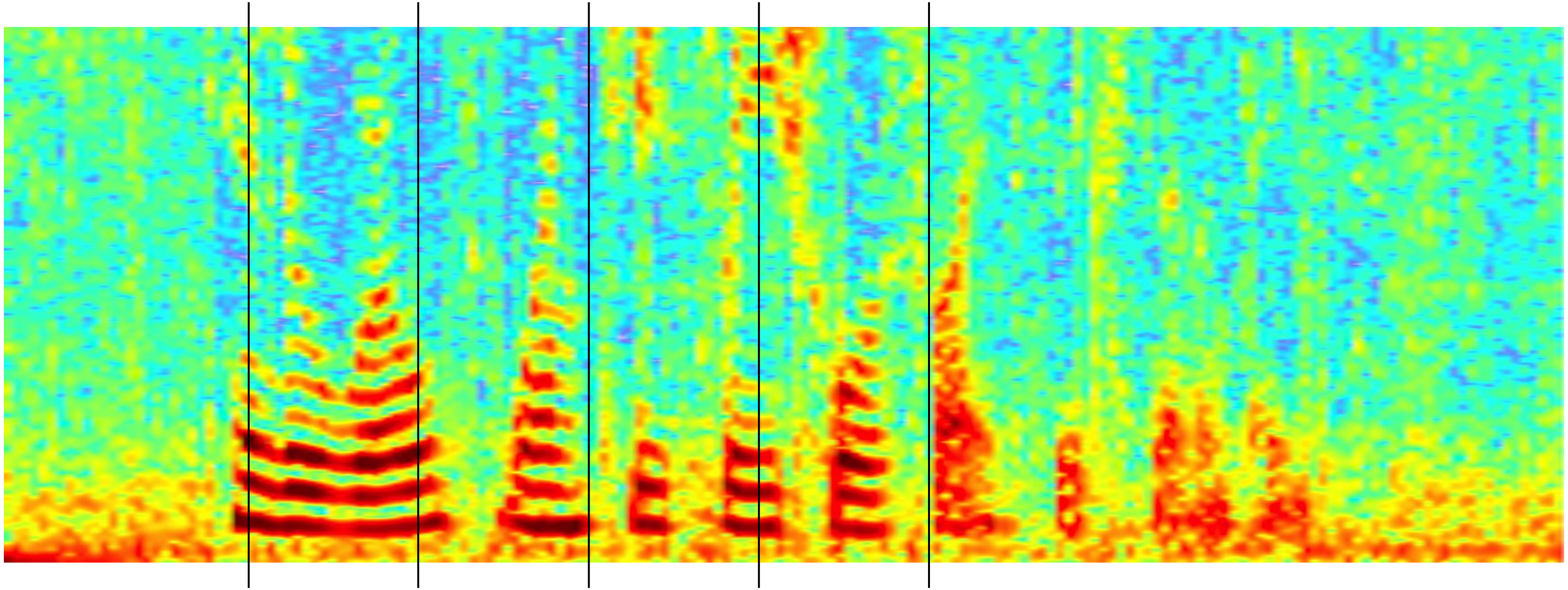
- Good handle on systems to store and manipulate data

- What we really care about now is understanding

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

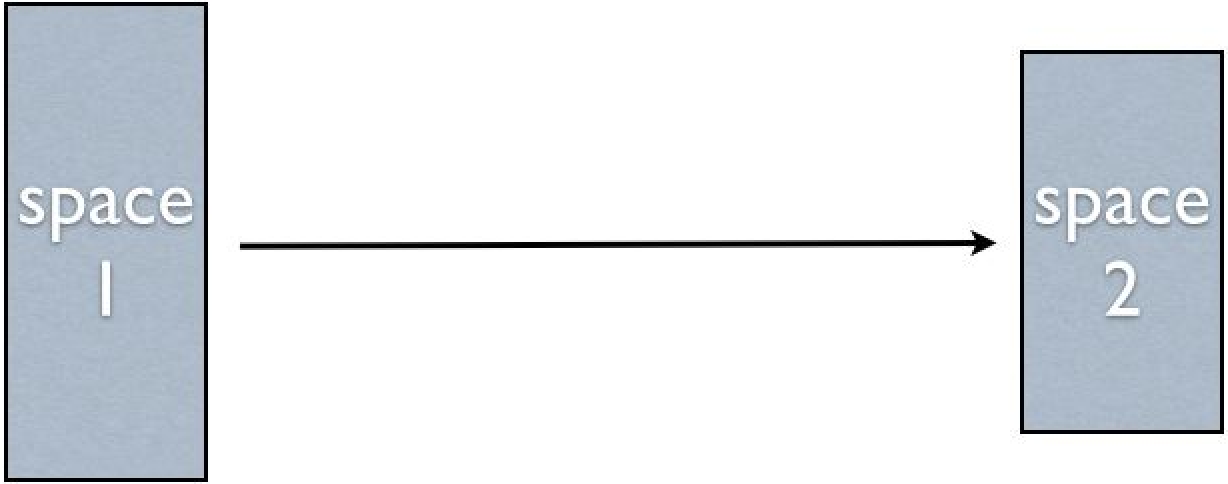
Selling all kinds of automobile and pickup truck parts, engines, and transmissions.

Example Queries 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*

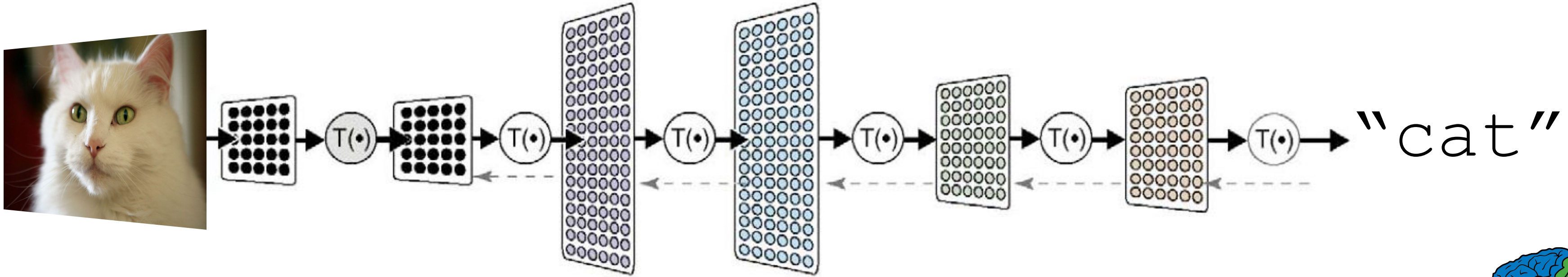
Neural Networks

- Learn a complicated function from data



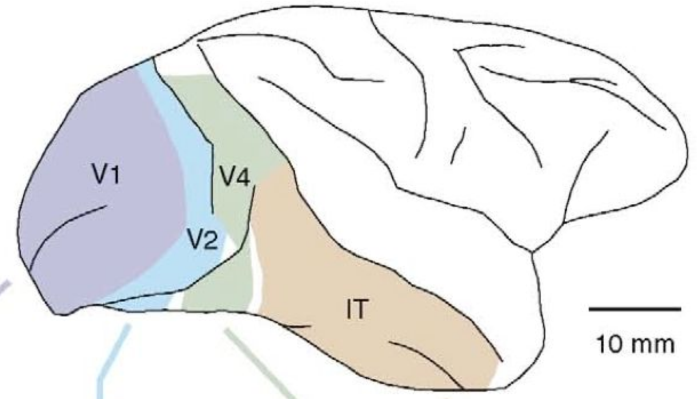
What is Deep Learning?

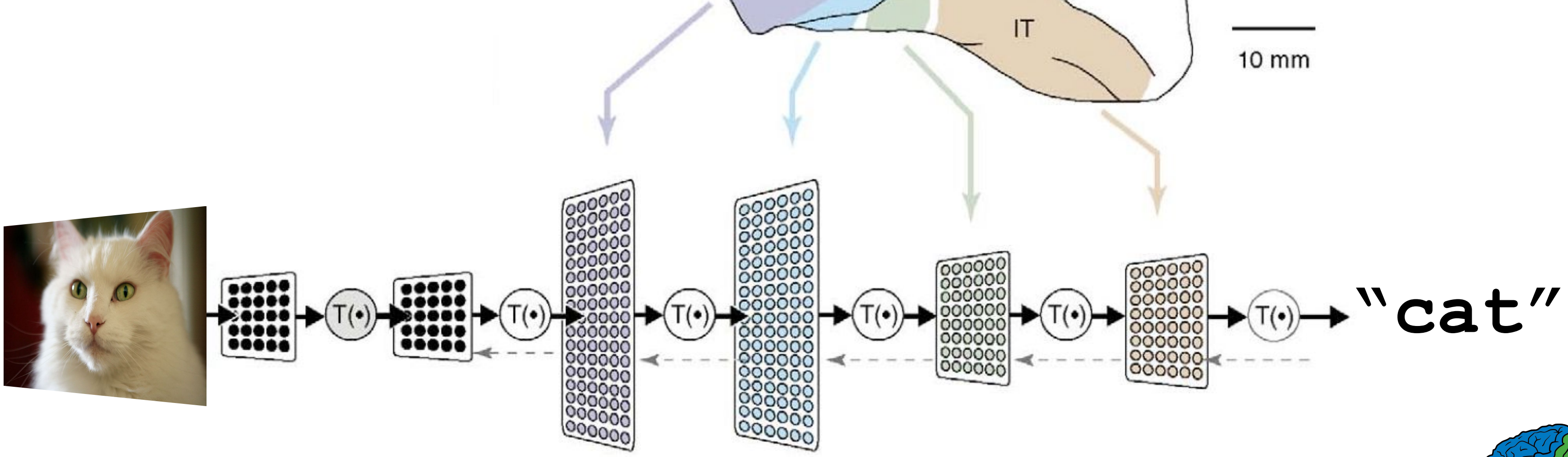
- A powerful class of machine learning model
- Modern reincarnation of artificial neural networks
- Collection of simple, trainable mathematical functions
- Compatible with many variants of machine learning



What is Deep Learning?

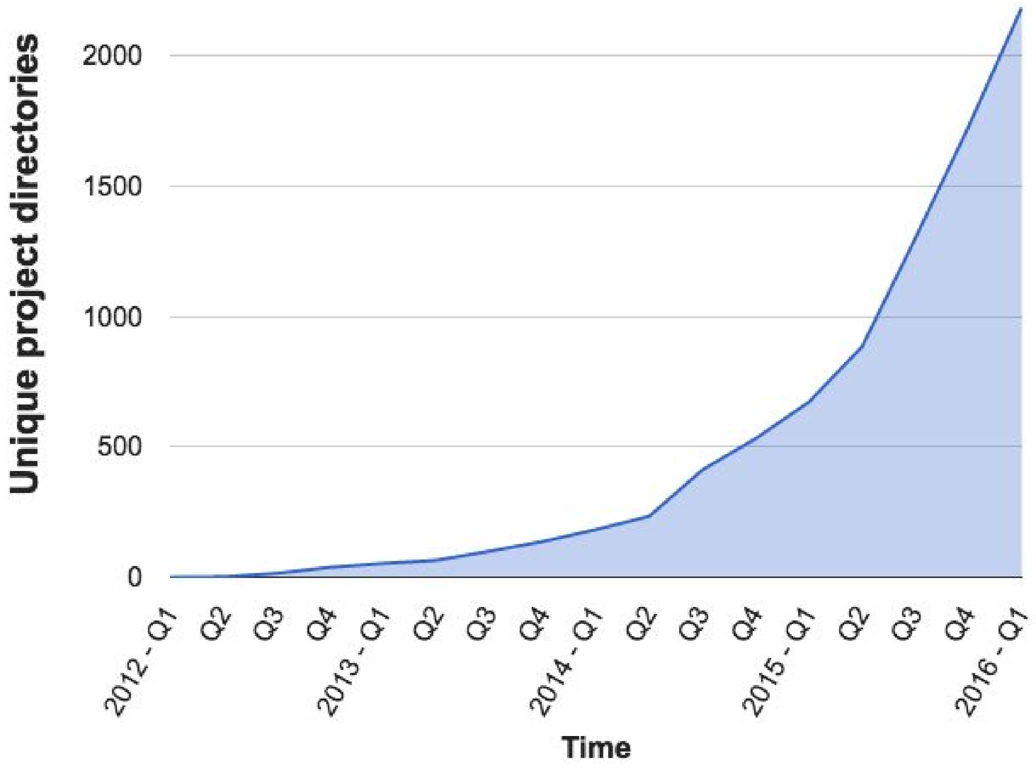
- Loosely based on (what little) we know about the brain





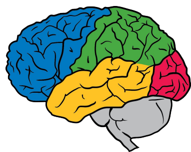
Growing Use of Deep Learning at Google

of directories containing model description files



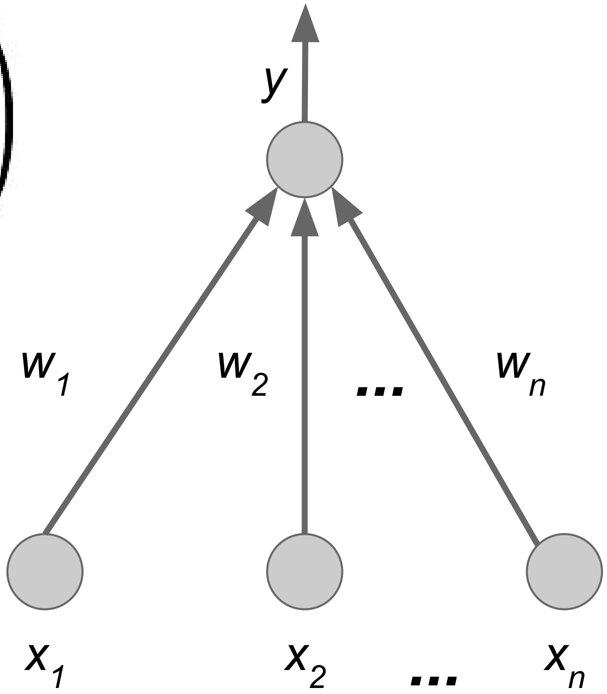
Across many products/areas:

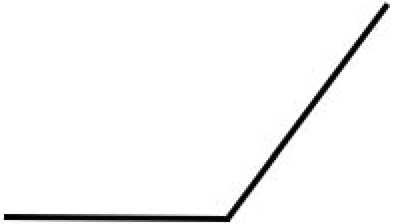
Android
Apps
drug discovery
Gmail
Image understanding
Maps
Natural language understanding
Photos
Robotics research
Speech
Translation
YouTube
… many others ...



The Neuron

$$y = F\left(\sum_i w_i x_i\right)$$

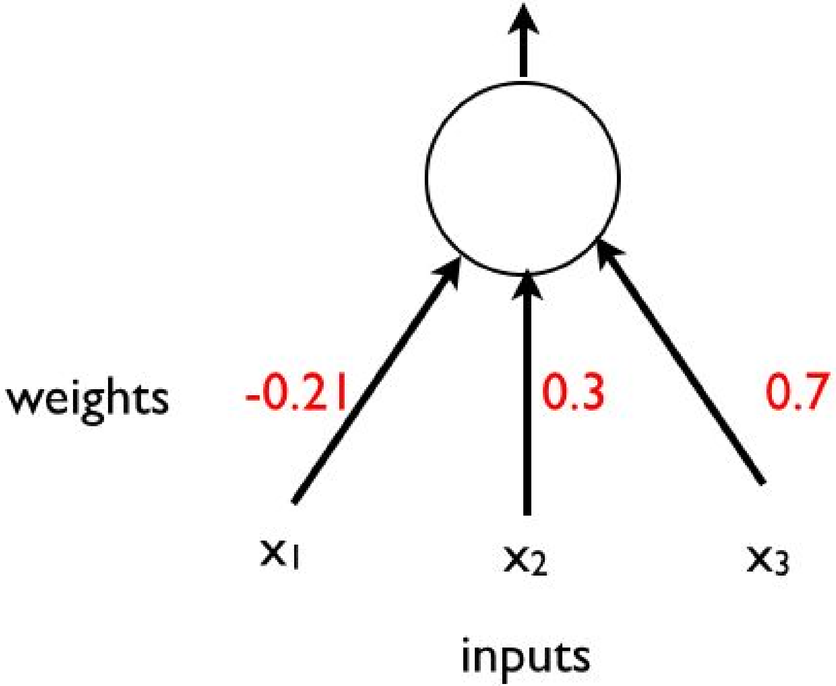


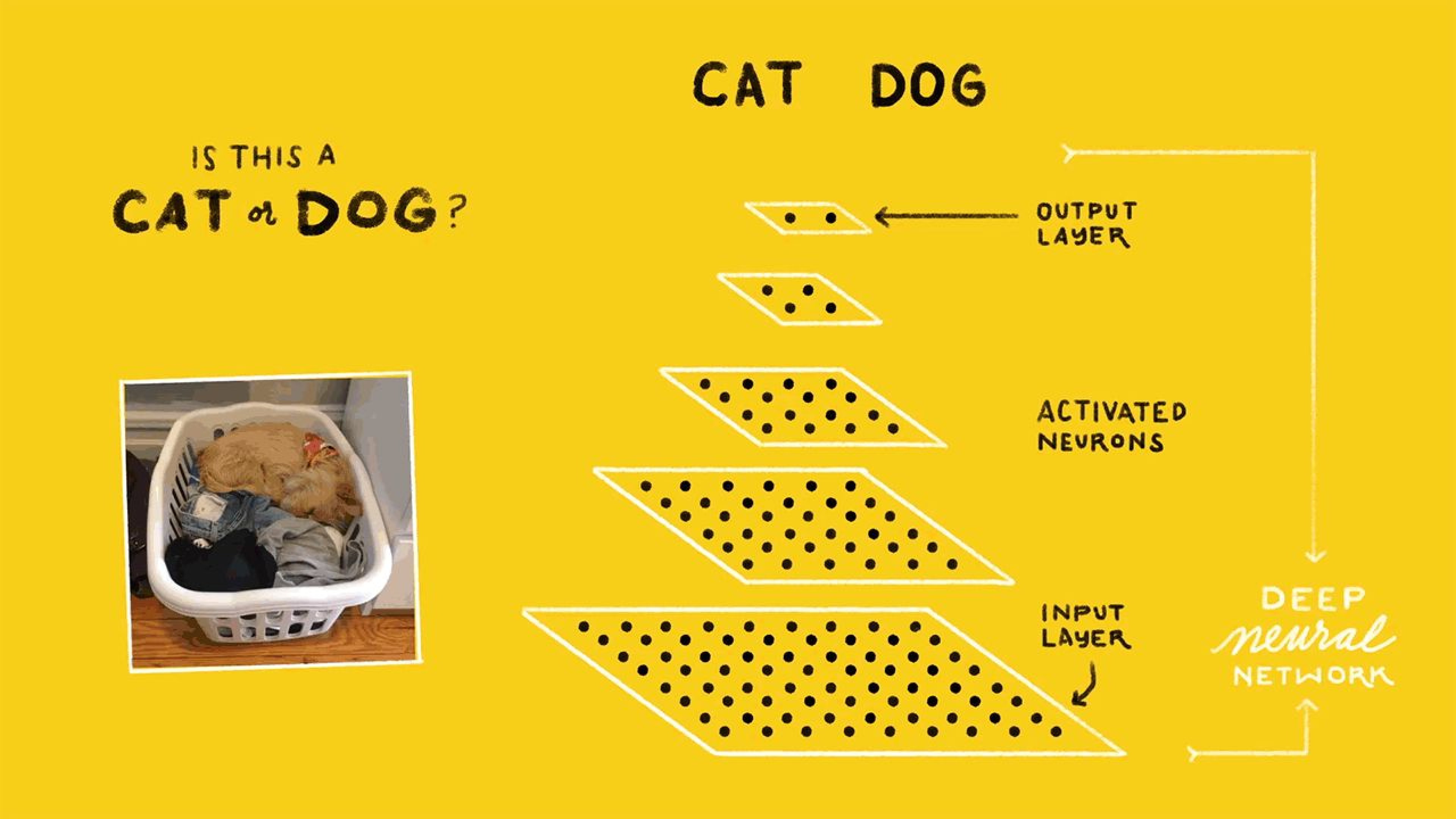


$F(x) = \max(0, x)$

F: a non-linear differentiable function

$$y = \max(0, -0.21 * x_1 + 0.3 * x_2 + 0.7 * x_3)$$




IS THIS A
CAT or DOG?
CAT DOG
OUTPUT
LAYER
ACTIVATED
NEURONS
INPUT
LAYER
DEEP
neural
NETWORK

Learning algorithm

While not done:

    Pick a random training example “(input, output)”
    Run neural network on “input”
    Adjust weights on edges to make output closer to “output”

Learning algorithm

While not done:

    Pick a random training example “(input, output)”

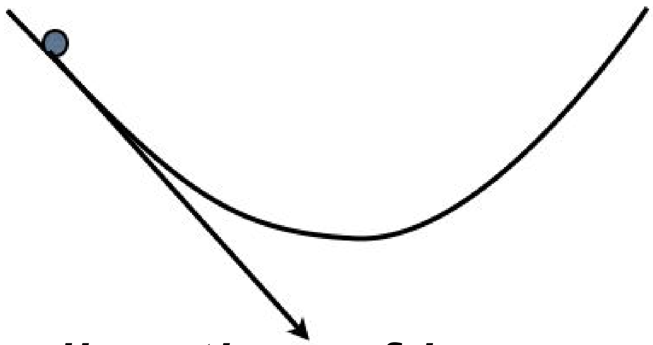
    Run neural network on “input”

    Adjust weights on edges to make output closer to “output”

Backpropagation

Use partial derivatives along the paths in the neural net

Follow the gradient of the error w.r.t. the connections

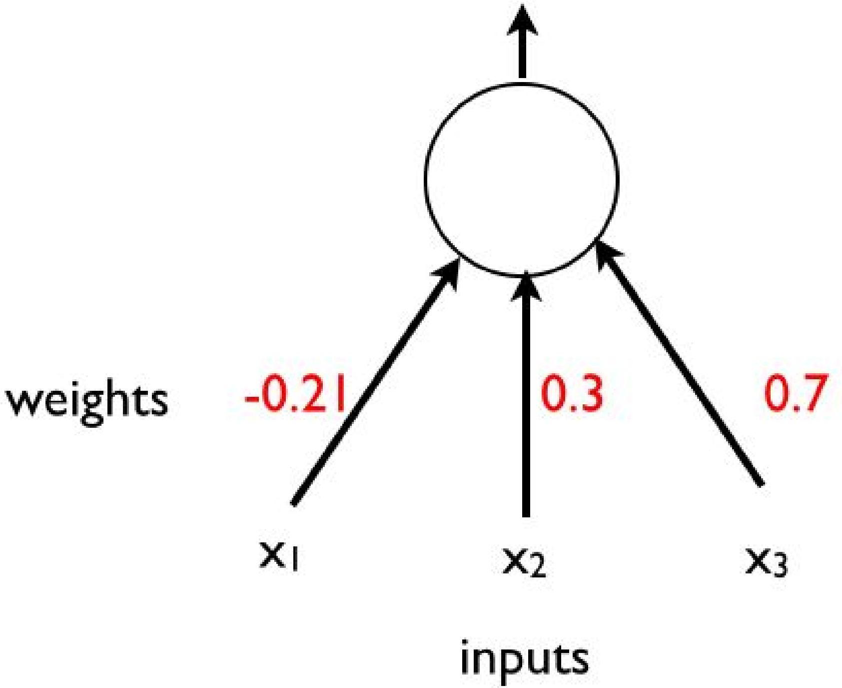


Gradient points in direction of improvement

Good description: **“Calculus on Computational Graphs: Backpropagation"**

http://colah.github.io/posts/2015-08-Backprop/

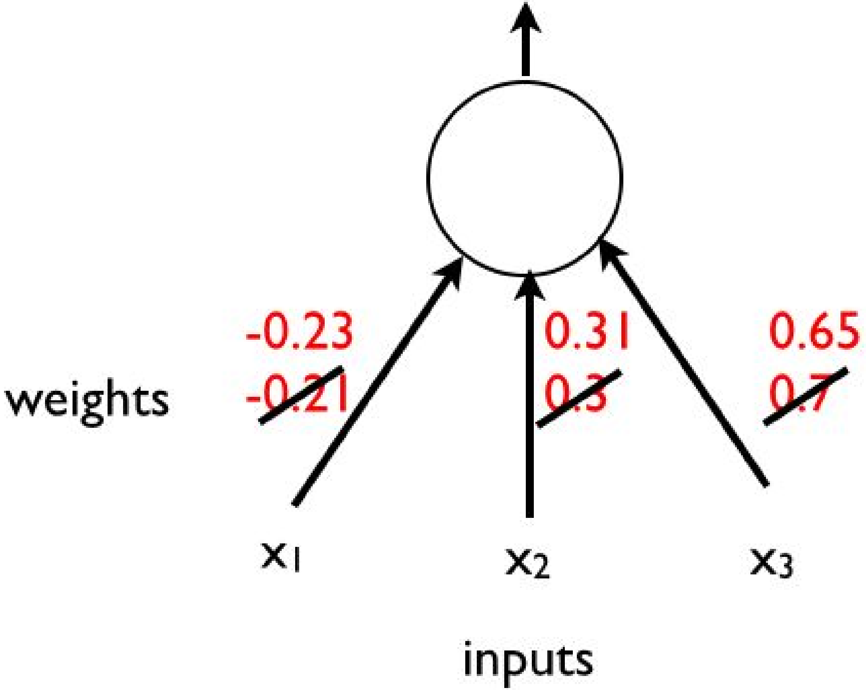
$$y = \max(0, -0.21 * x_1 + 0.3 * x_2 + 0.7 * x_3)$$



next time:

$$\text{output} = \max(0, -0.23*x_1 + 0.31*x_2 + 0.65*x_3)$$

$$\cancel{\text{output} = \max(0, -0.21*x_1 + 0.3*x_2 + 0.7*x_3)}$$

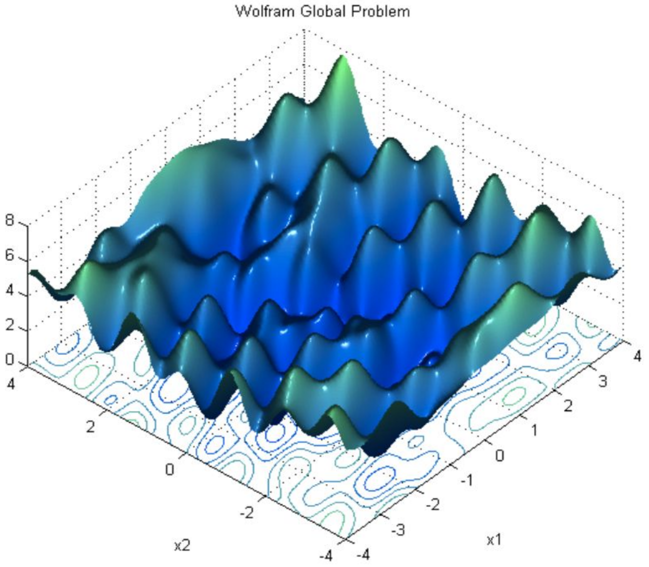


Non-convexity

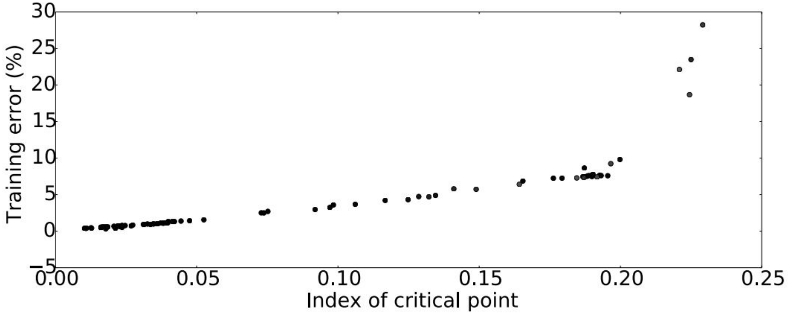
-Low-D => local minima

-High-D => saddle points

-Most local minima are close to the global minima



This shows a function of 2 variables: real neural nets are functions of hundreds of millions of variables!



Slide Credit: Yoshua Bengio

Plenty of raw data

- **Text**: trillions of words of English + other languages
- **Visual data**: billions of images and videos
- **Audio**: tens of thousands of hours of speech per day
- **User activity**: queries, marking messages spam, etc.
- **Knowledge graph**: billions of labelled relation triples
- ...

How can we build systems that truly understand this data?

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

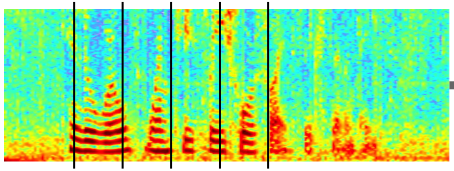
What changed? We now have:

sufficient computational resources
large enough interesting datasets

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

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

Speech Recognition



Acoustic Input

Deep
Recurrent
Neural Network

"How cold is it outside?"

Text Output

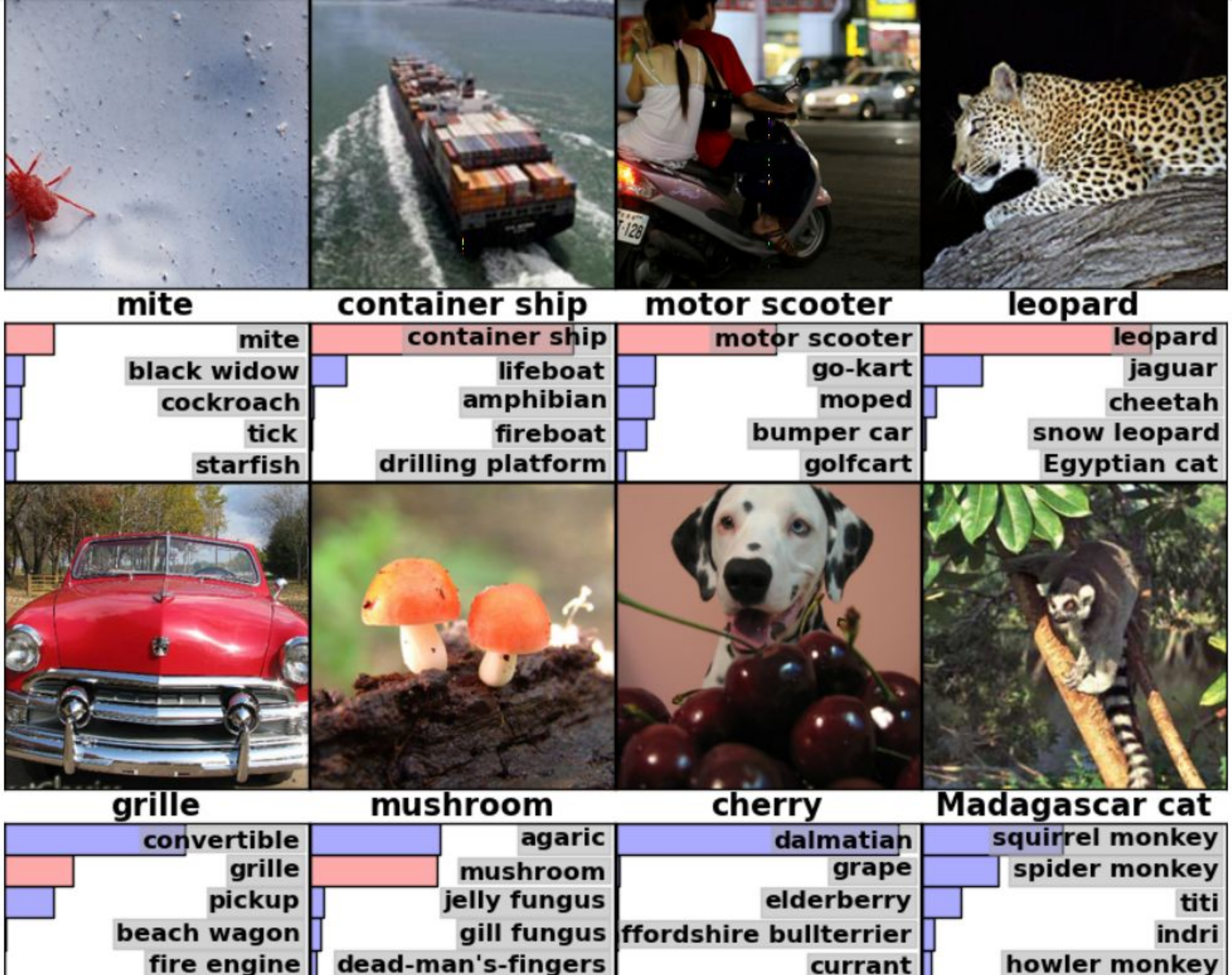
Reduced word errors by more than 30%

Google Research Blog - August 2012, August 2015

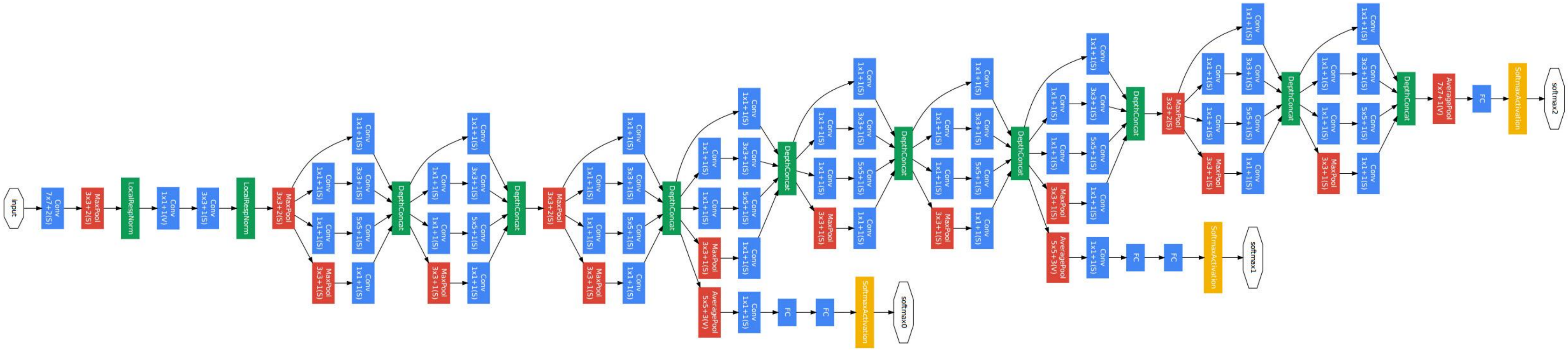
ImageNet Challenge

Given an image, predict one of 1000 different classes

Image credit: www.cs.toronto.edu/~fritz/absps/imagenet.pdf



The Inception Architecture (GoogLeNet, 2014)



Going Deeper with Convolutions

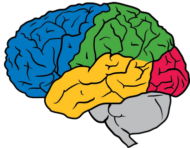
Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich

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



Good Fine-Grained Classification

“hibiscus”

“dahlia”

Good Generalization

Both recognized as “meal”

Sensible Errors

“snake”

“dog”

Google Photos Search

Your Photo

Deep Convolutional Neural Network

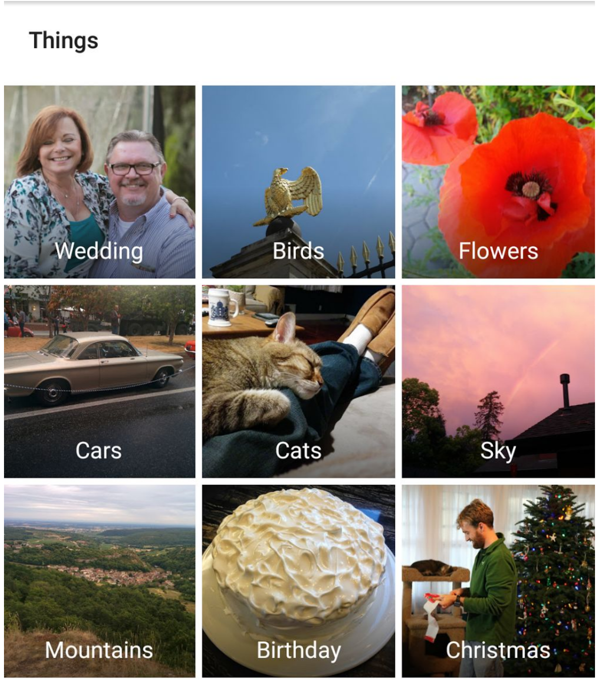
"ocean"

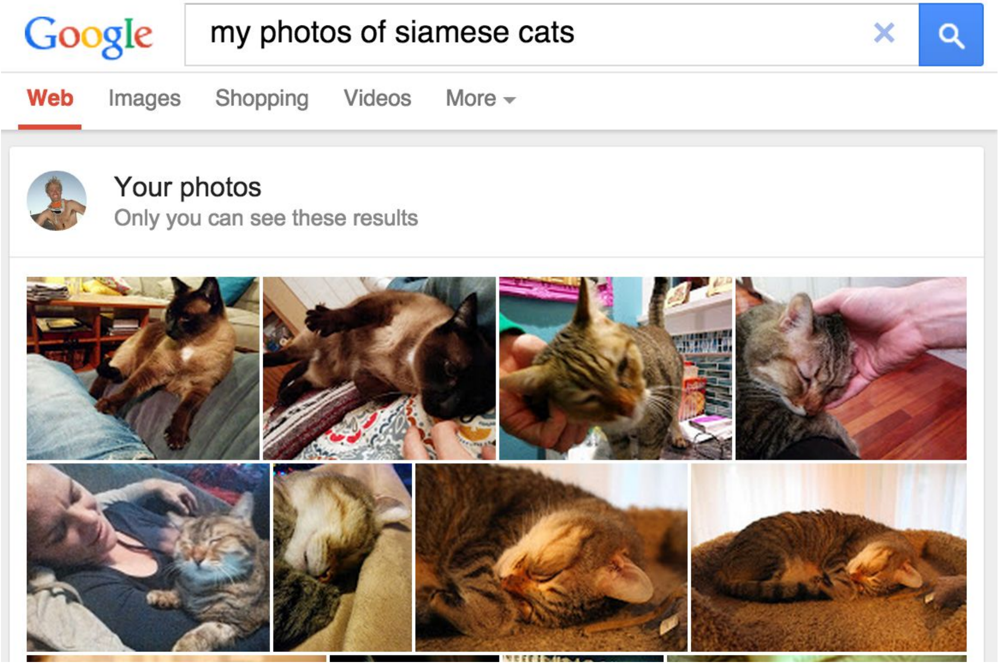
Automatic Tag

Search personal photos without tags.

Google Research Blog - June 2013

Google Photos Search

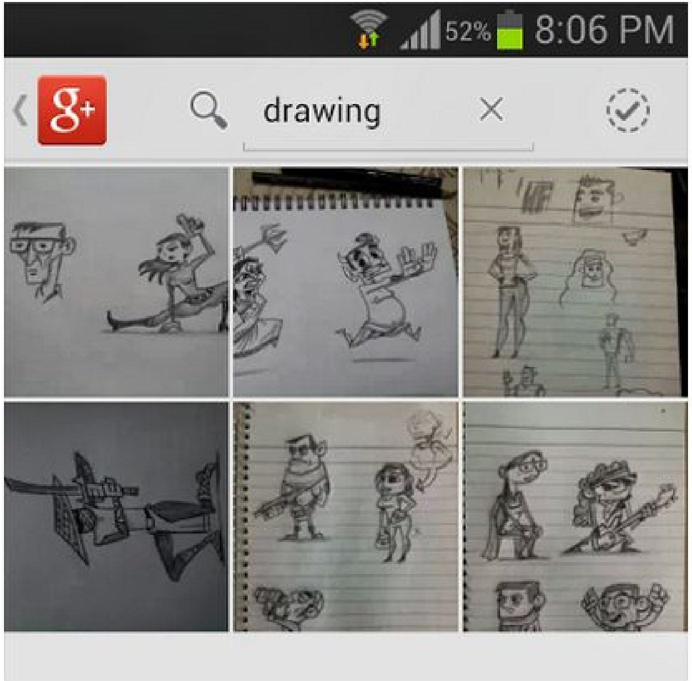


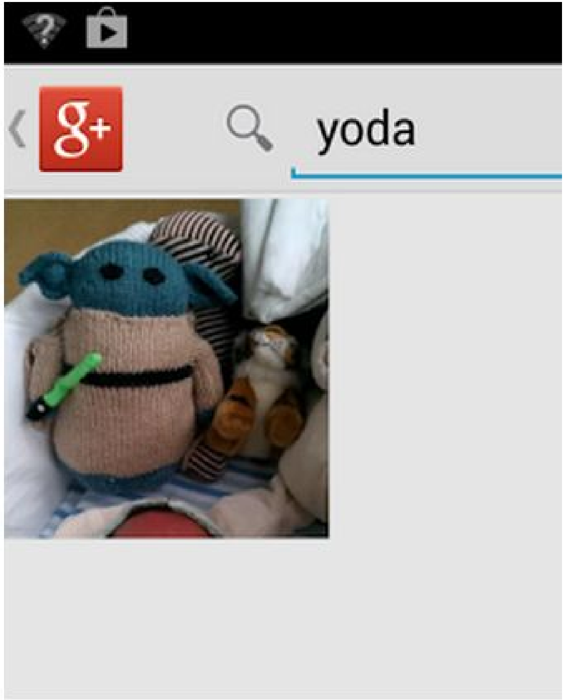


Research at Google

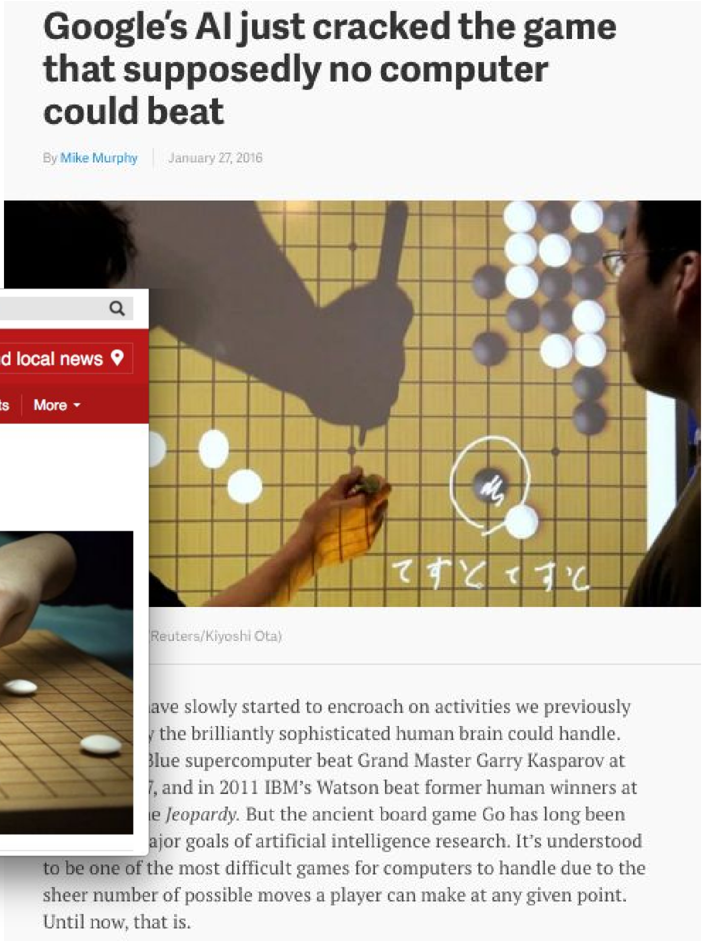
Google Photos Search

Google Plus photo search is awesome. Searched with keyword 'Drawing' to find all my scribbles at once :D





“Seeing” Go


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

By Mike Murphy | January 27, 2016

Reuters/Kiyoshi Ota)

ave slowly started to encroach on activities we previously
the brilliantly sophisticated human brain could handle.
Blue supercomputer beat Grand Master Garry Kasparov at
, and in 2011 IBM’s Watson beat former human winners at
e *Jeopardy*. But the ancient board game Go has long been
ajor 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.


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

Mastering the Game of Go with Deep Neural Networks and Tree Search, Silver *et al.*, Nature, vol. 529 (2016), pp. 484-503

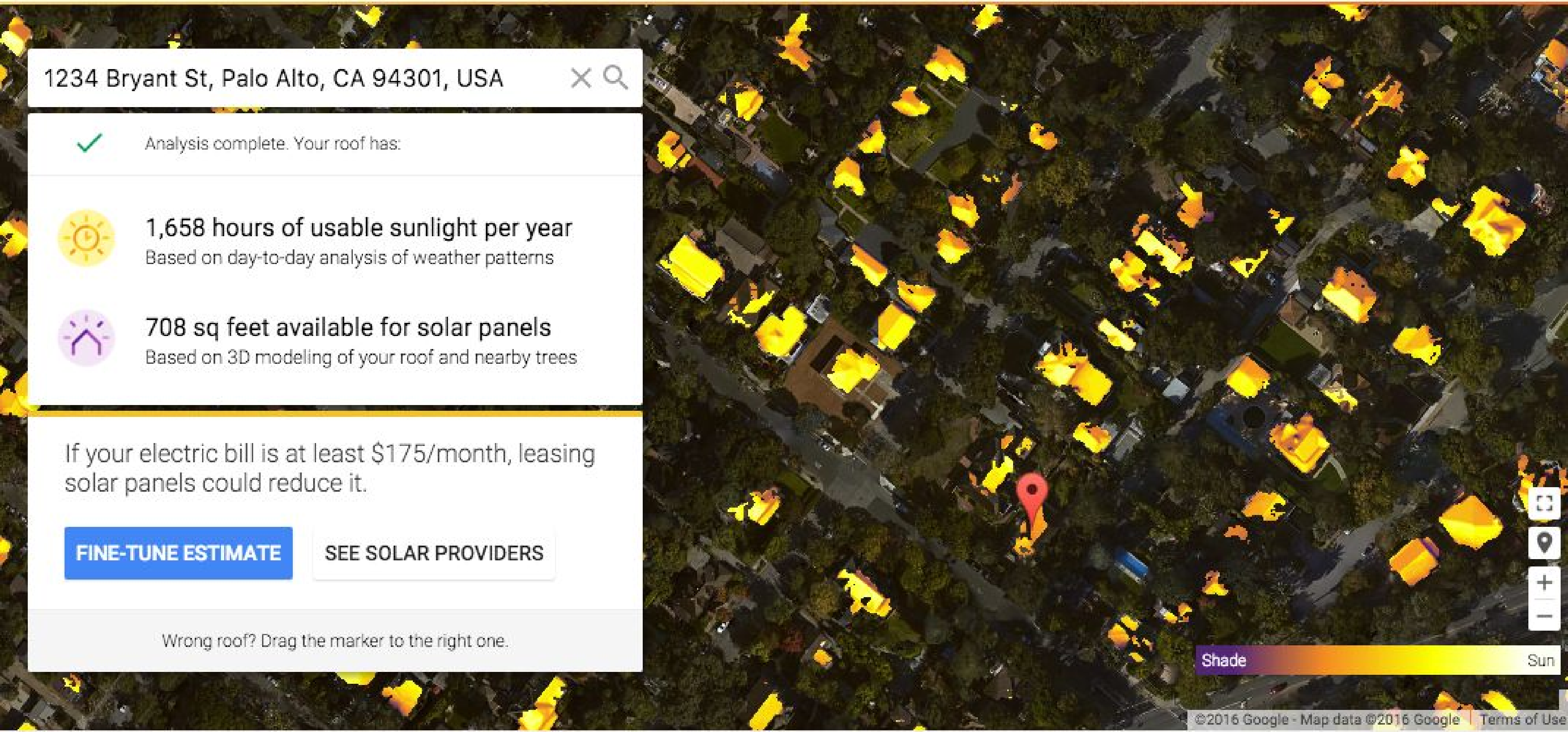
Reuse same model for completely different problems

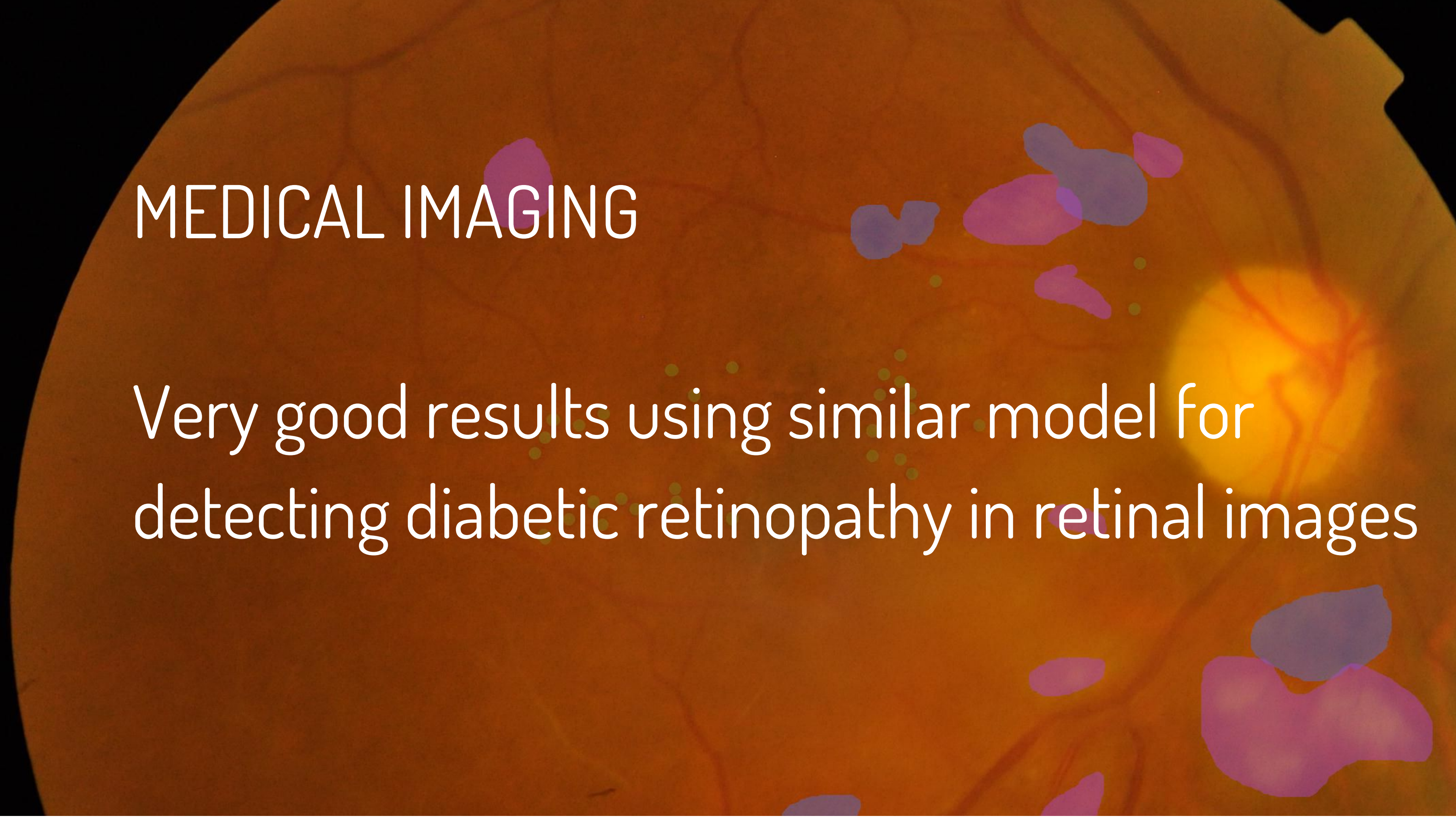
Same basic model structure
(e.g. given image, predict interesting parts of image)
trained on different data,
useful in **completely different contexts**


ASIAWIDE TRAVEL
Tel: (02) 9745 3355 1st Floor, 240 BURWOOD RD
Maria'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
RADIATORS • FILTERS • AIR CONDITIONING • EFI TUNING
FUEL INJECTION SERVICING • BATTERIES • AUTO ELECTRICAL
Factory Trained Technicians

Google Project Sunroof





MEDICAL IMAGING

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

Language Understanding

Query

[car parts for sale]

Document 1

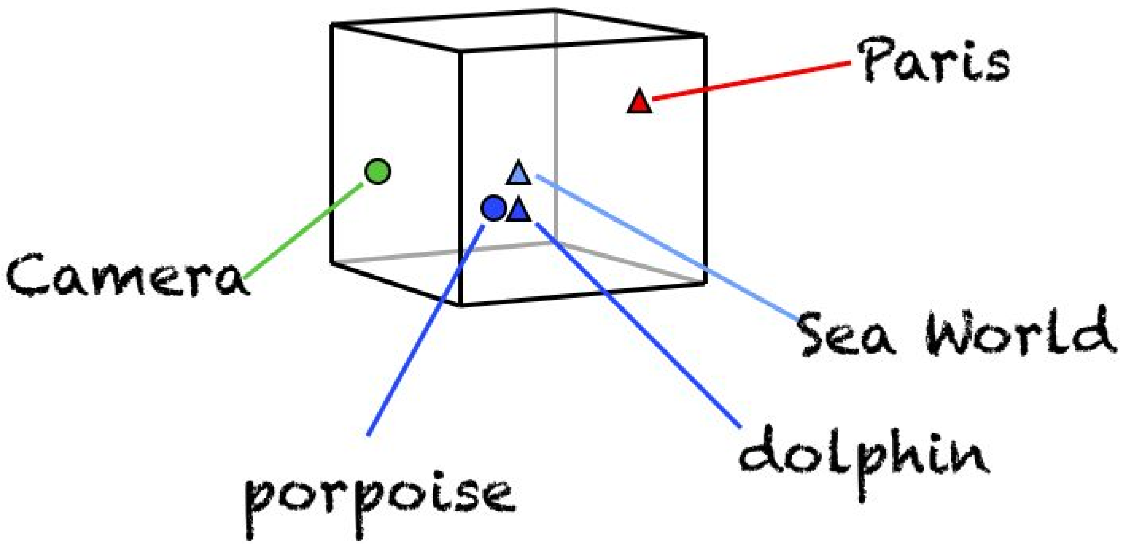
… car parking available for a small fee.
… parts of our floor model inventory for sale.

Document 2

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

How to deal with Sparse Data?

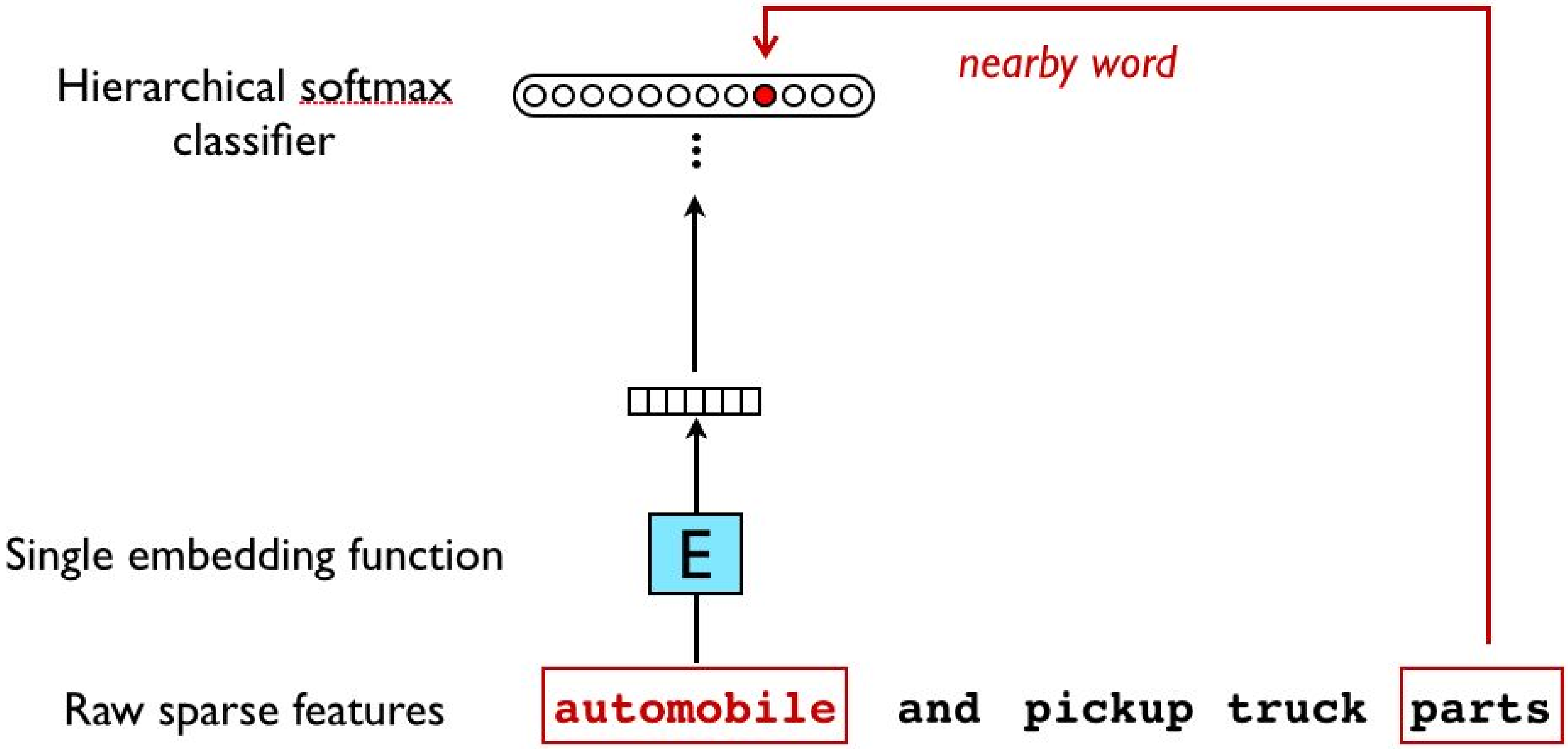
3-D embedding space



Embedding Function: A look-up-table that maps sparse features into dense floating point vectors.

Usually use many more than 3 dimensions (e.g. 100D, 1000D)

Embeddings Can be Trained With Backpropagation



Mikolov, Sutskever, Chen, Corrado and Dean. *Distributed Representations of Words and Phrases and Their Compositionality*, NIPS 2013.

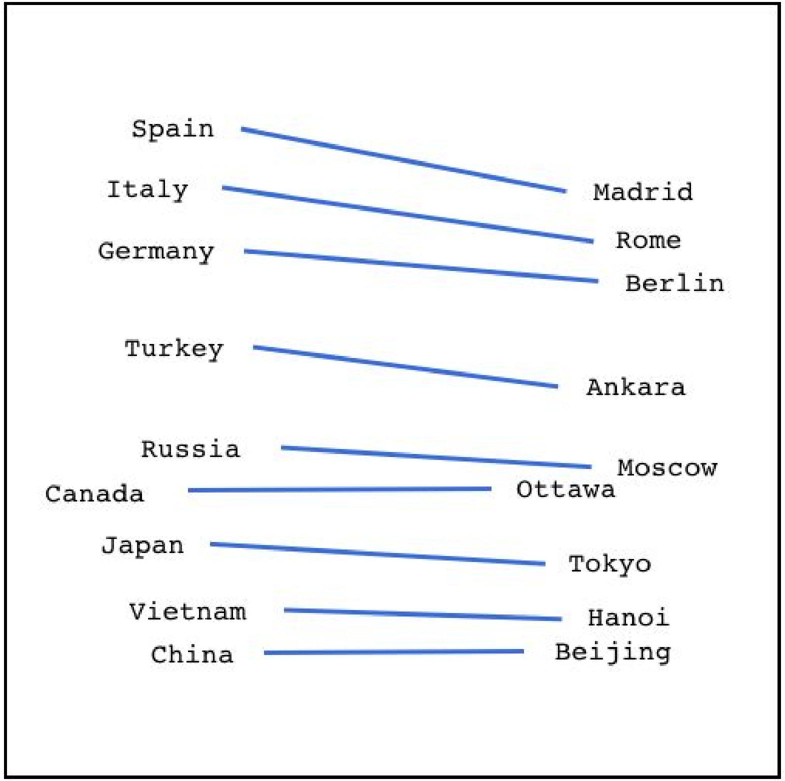
Nearest Neighbors are Closely Related Semantically

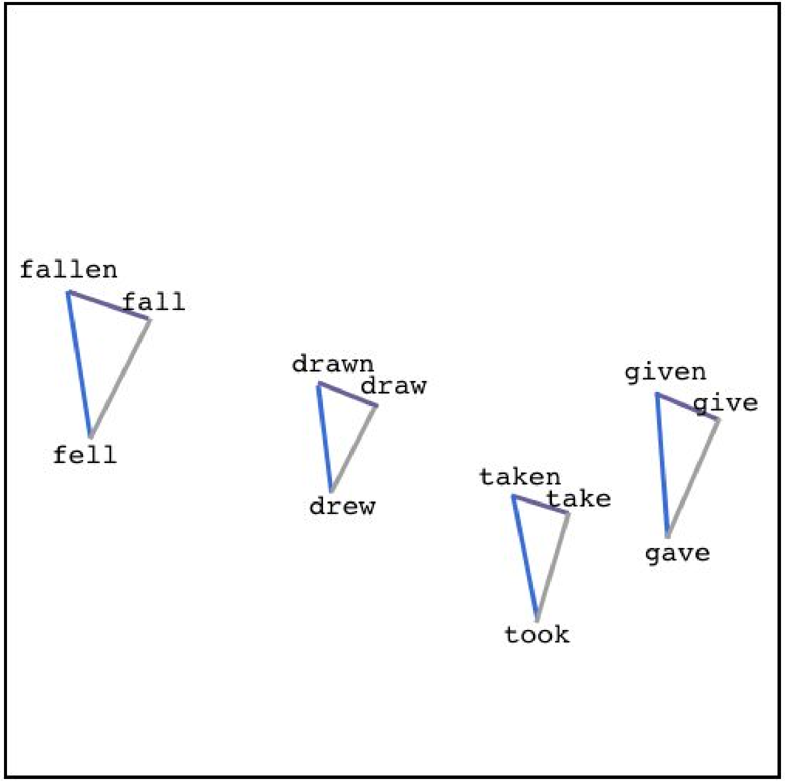
Trained language model on Wikipedia

tiger shark	**car**	**new york**
bull shark	cars	new york city
blacktip shark	muscle car	brooklyn
shark	sports car	long island
oceanic whitetip shark	compact car	syracuse
sandbar shark	autocar	manhattan
dusky shark	automobile	washington
blue shark	pickup truck	bronx
requiem shark	racing car	yonkers
great white shark	passenger car	poughkeepsie
lemon shark	dealership	new york state

* 5.7M docs, 5.4B terms, 155K unique terms, 500-D embeddings

Directions are Meaningful





Solve analogies with vector arithmetic!

$$V(\texttt{queen}) - V(\texttt{king}) \approx V(\texttt{woman}) - V(\texttt{man})$$

$$V(\texttt{queen}) \approx V(\texttt{king}) + (V(\texttt{woman}) - V(\texttt{man}))$$

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*"

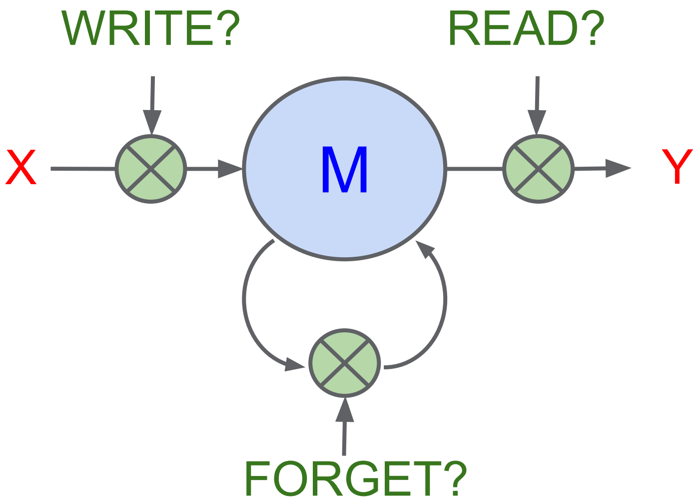
A Simple Model of Memory

Instruction Input Output

WRITE X, M

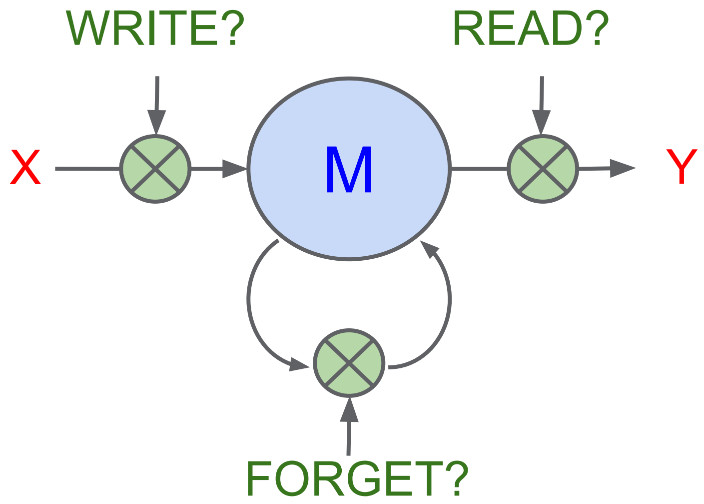
READ M, Y

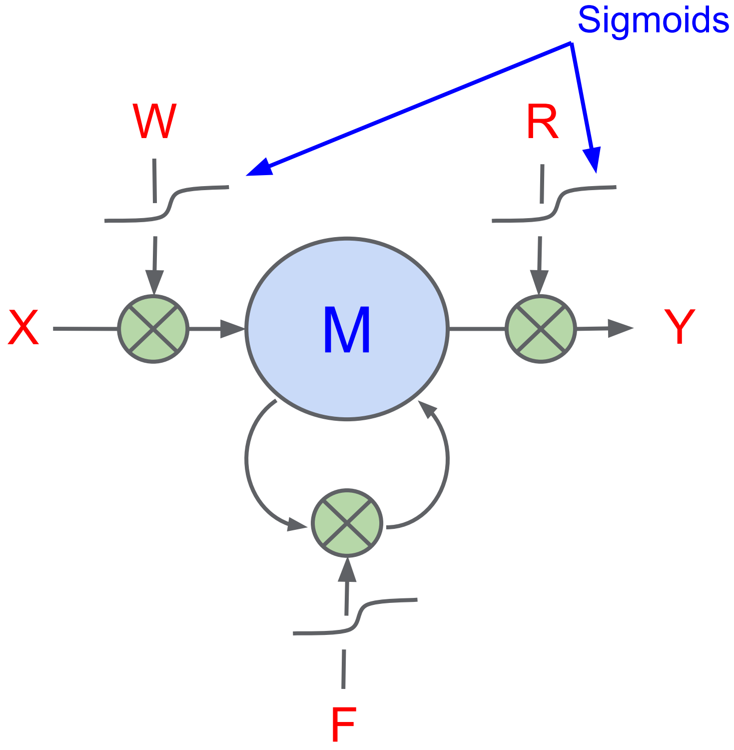
FORGET M



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

[Hochreiter & Schmidhuber, 1997]





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

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

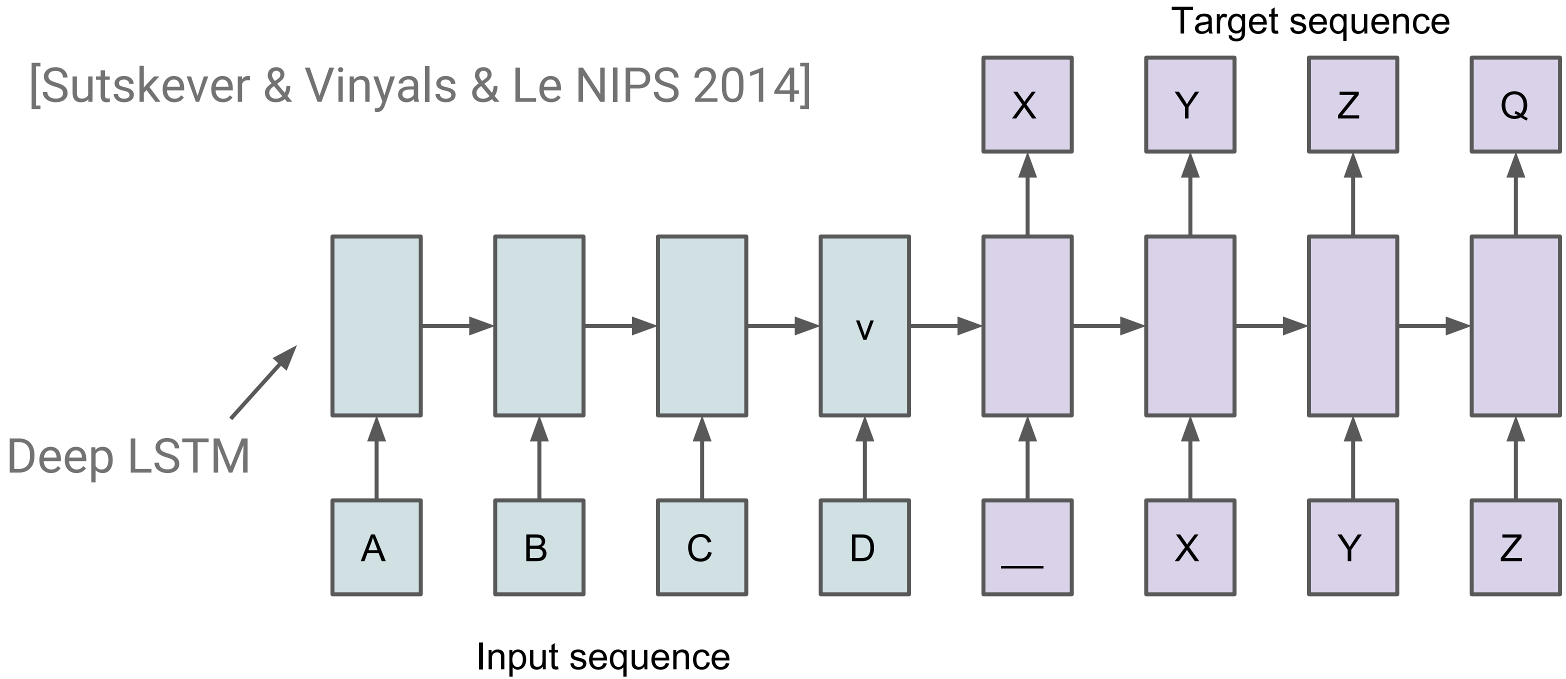
Enables long term dependencies to flow

TensorFlow

```
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])
```

Sequence-to-Sequence Model

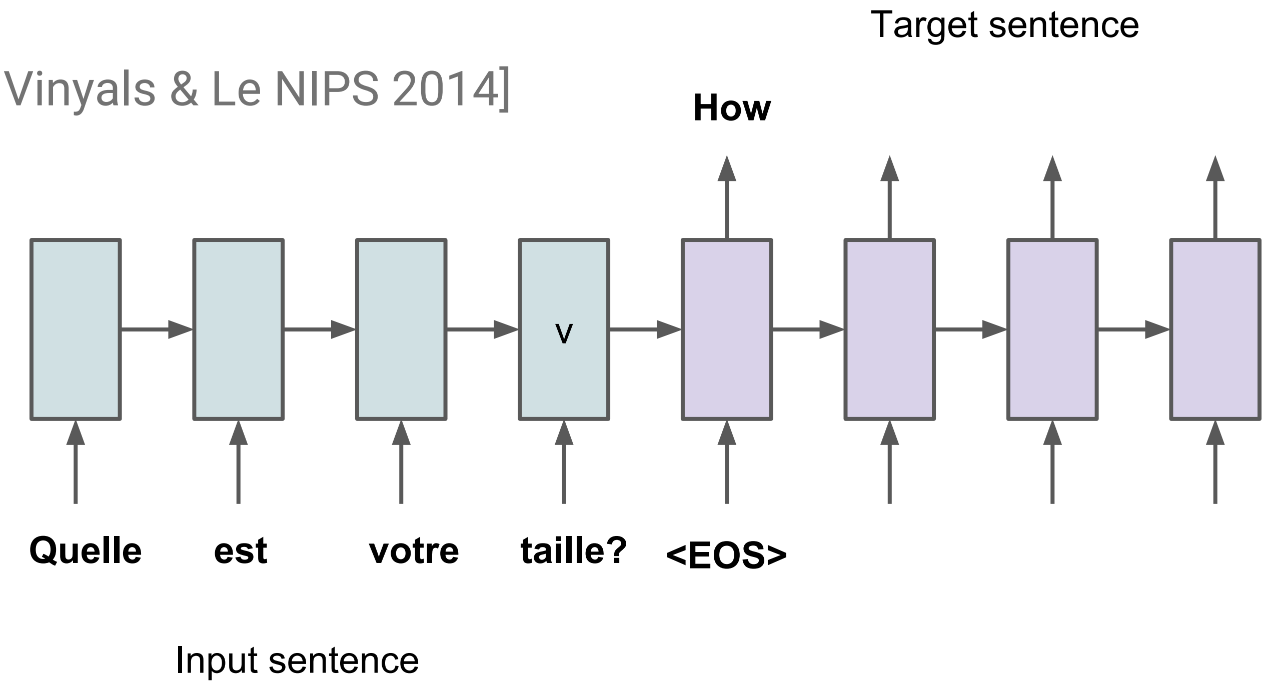
[Sutskever & Vinyals & Le NIPS 2014]



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

Sequence-to-Sequence Model: Machine Translation

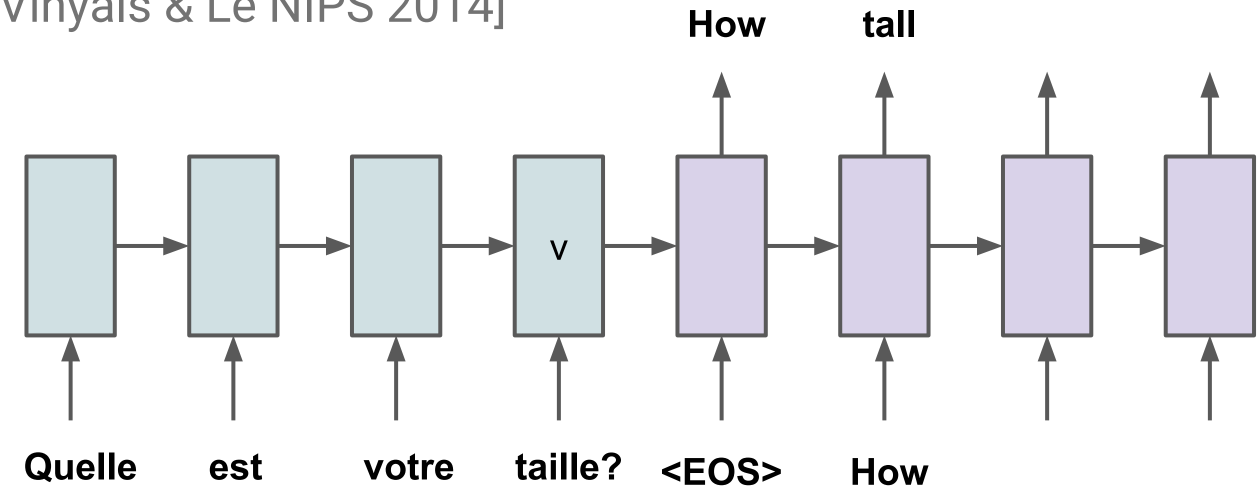
[Sutskever & Vinyals & Le NIPS 2014]



Sequence-to-Sequence Model: Machine Translation

[Sutskever & Vinyals & Le NIPS 2014]

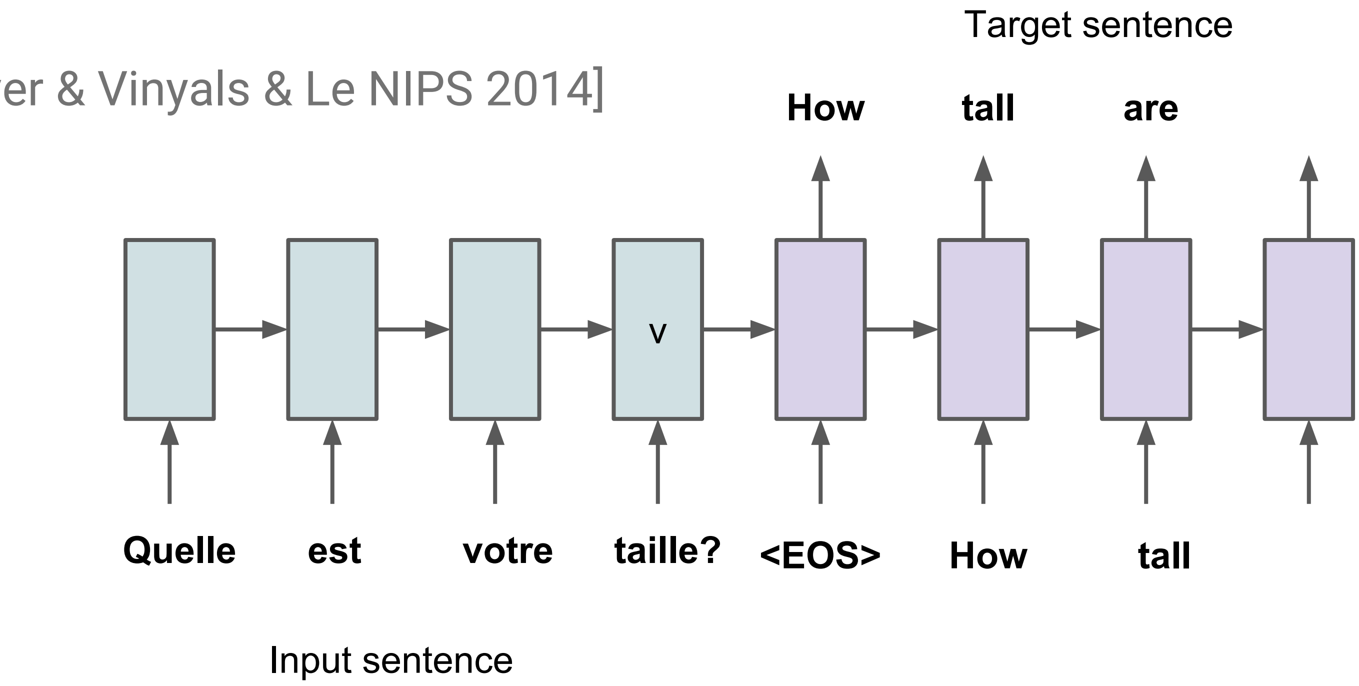
Target sentence



Input sentence

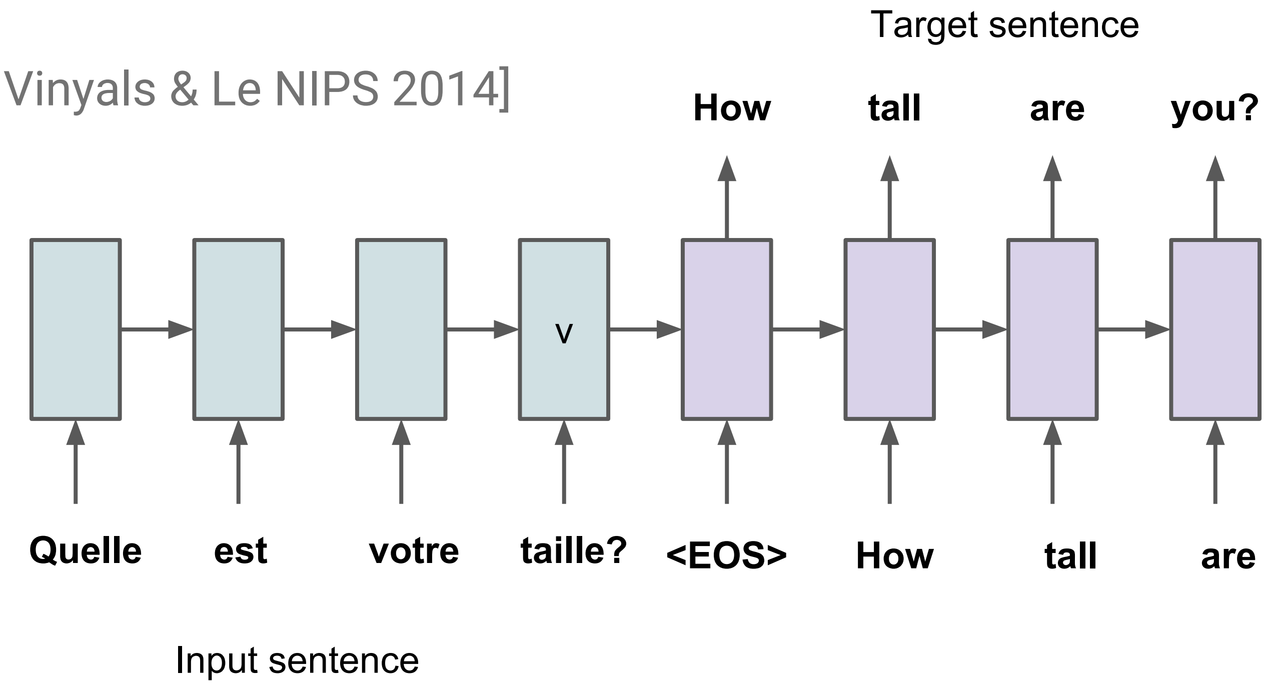
Sequence-to-Sequence Model: Machine Translation

[Sutskever & Vinyals & Le NIPS 2014]



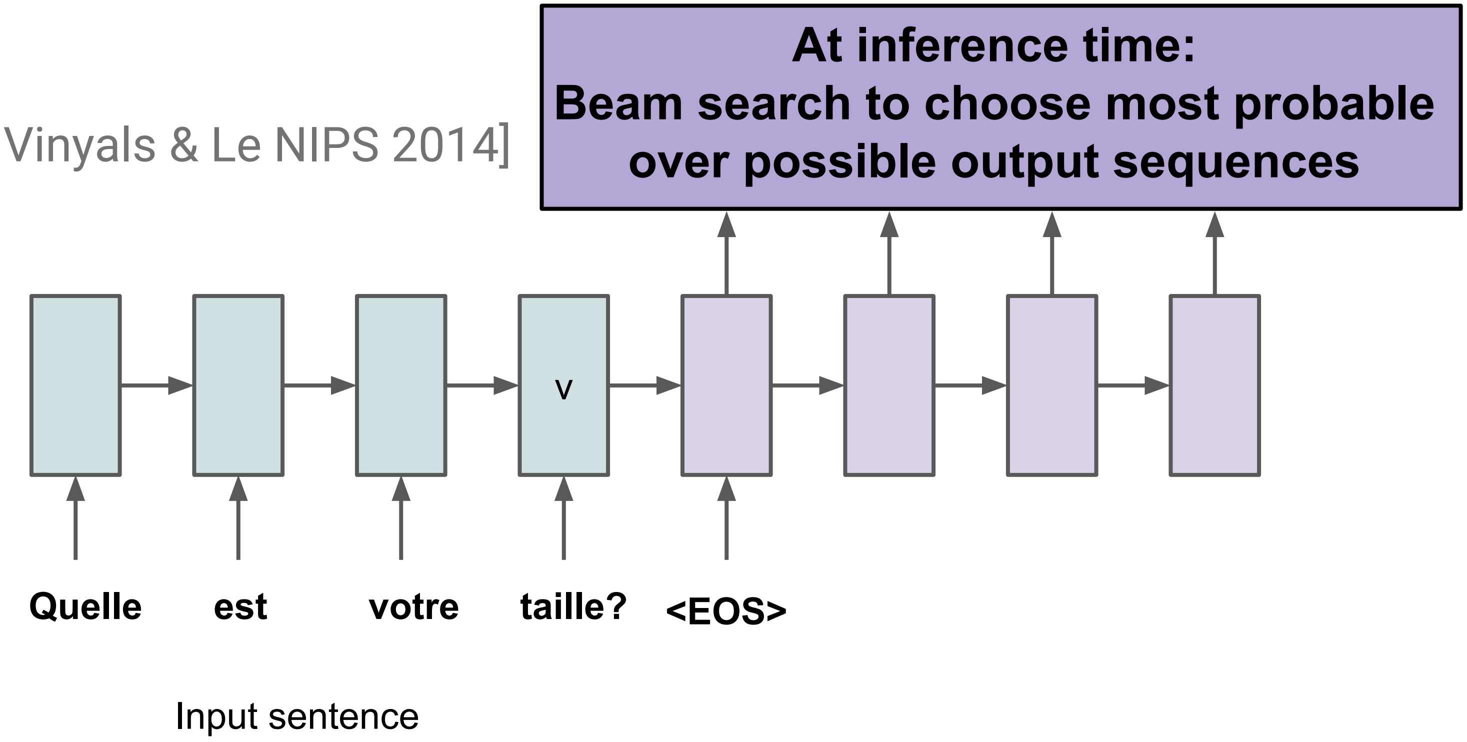
Sequence-to-Sequence Model: Machine Translation

[Sutskever & Vinyals & Le NIPS 2014]



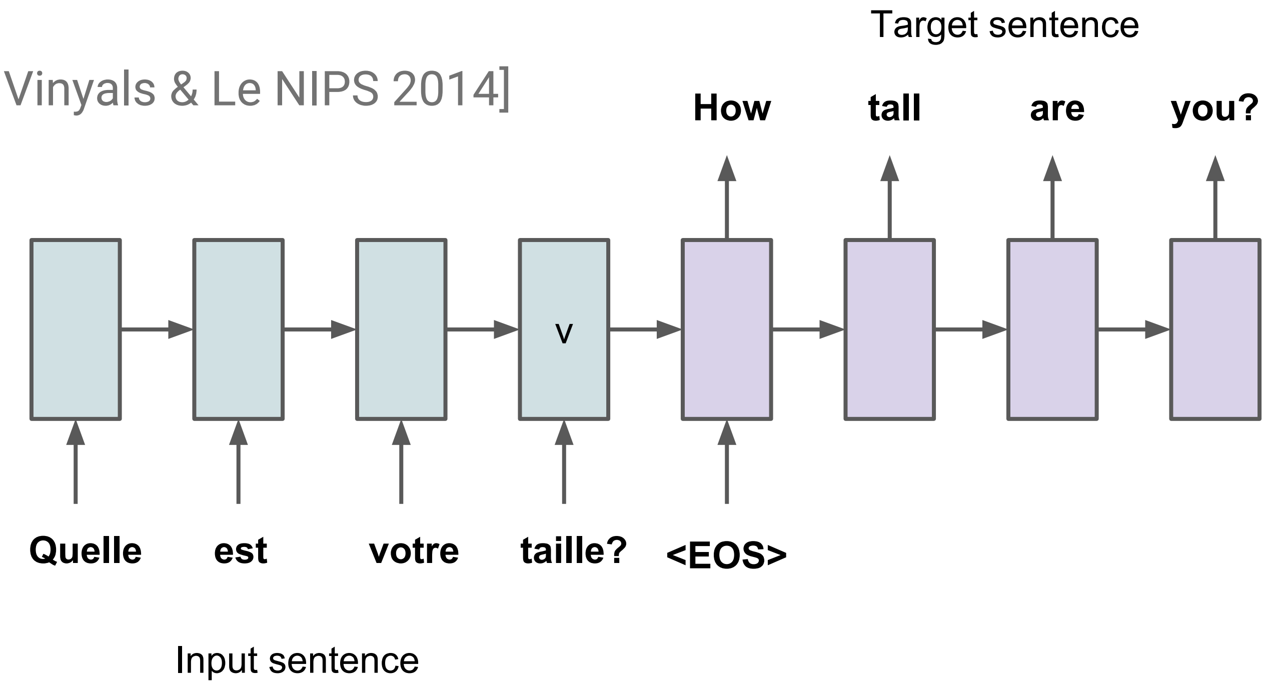
Sequence-to-Sequence Model: Machine Translation

[Sutskever & Vinyals & Le NIPS 2014]



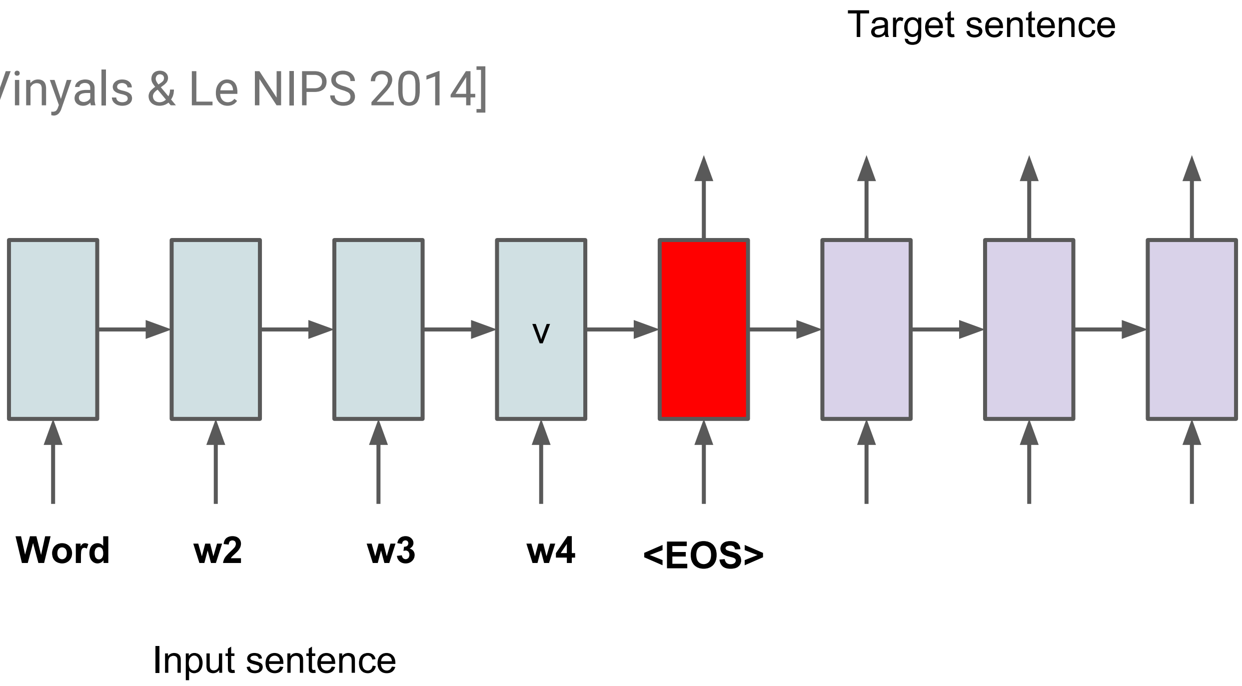
Sequence-to-Sequence Model: Machine Translation

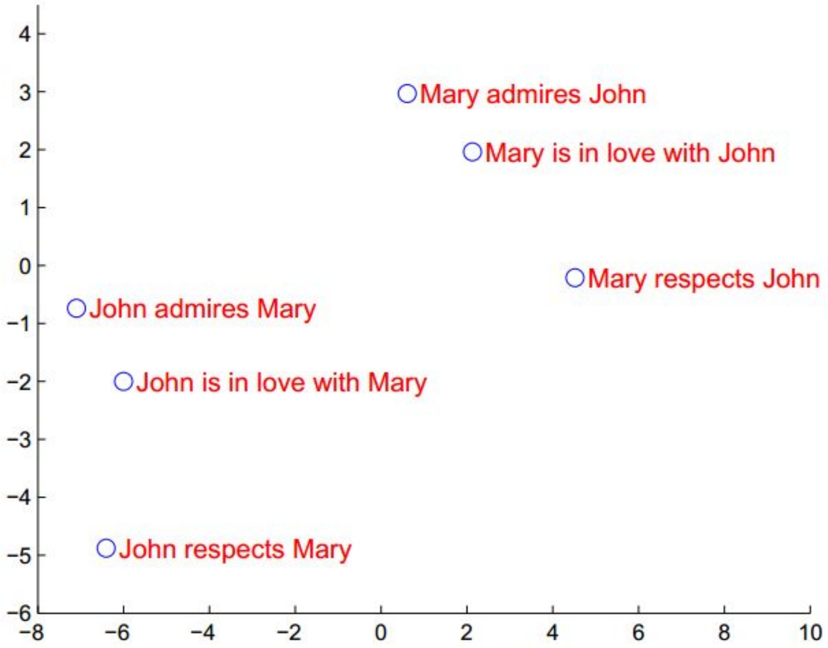
[Sutskever & Vinyals & Le NIPS 2014]

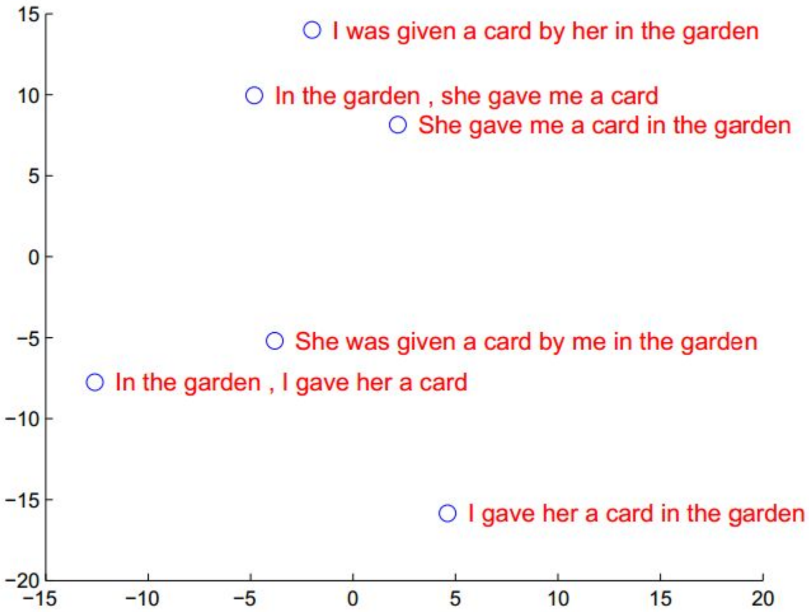


Sequence-to-Sequence Model: Machine Translation

[Sutskever & Vinyals & Le NIPS 2014]




Mary admires John
Mary is in love with John
Mary respects John
John admires Mary
John is in love with Mary
John respects Mary
4
3
2
1
0
−1
−2
−3
−4
−5
−6
−8
−6
−4
−2
0
2
4
6
8
10


I was given a card by her in the garden
In the garden , she gave me a card
She gave me a card in the garden
She was given a card by me in the garden
In the garden , I gave her a card
I gave her a card in the garden
15
10
5
0
−5
−10
−15
−20
−15
−10
−5
0
5
10
15
20

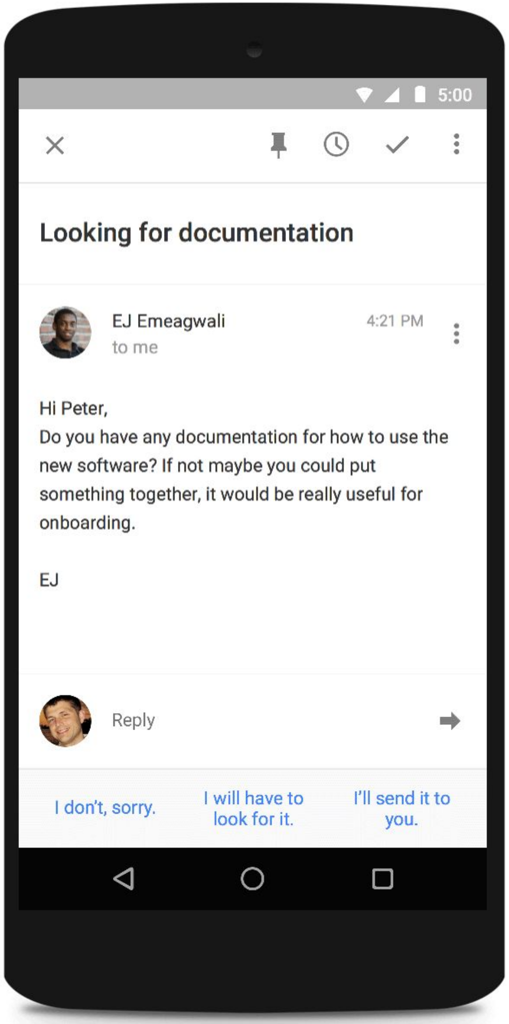
Inbox
by Gmail

Smart Reply

April 1, 2009: **April Fool's Day joke**

Nov 5, 2015: **Launched Real Product**

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

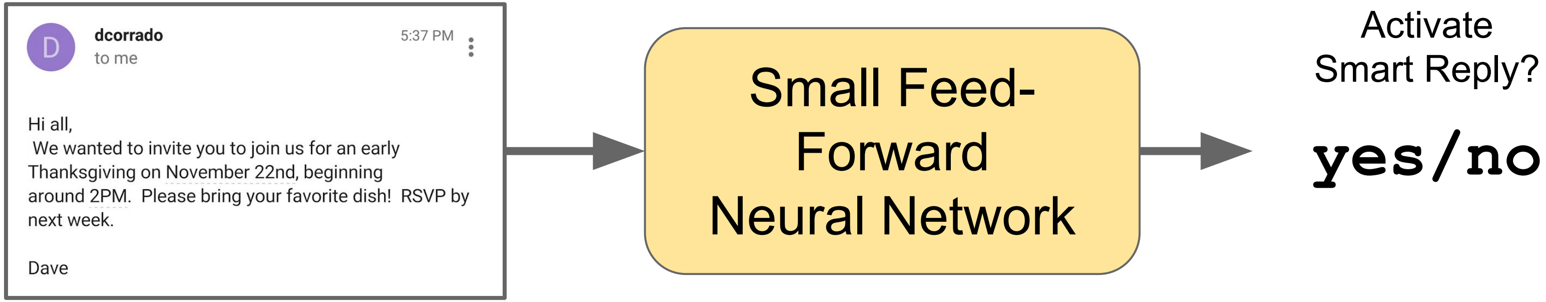


Smart Reply

Google Research Blog
- Nov 2015

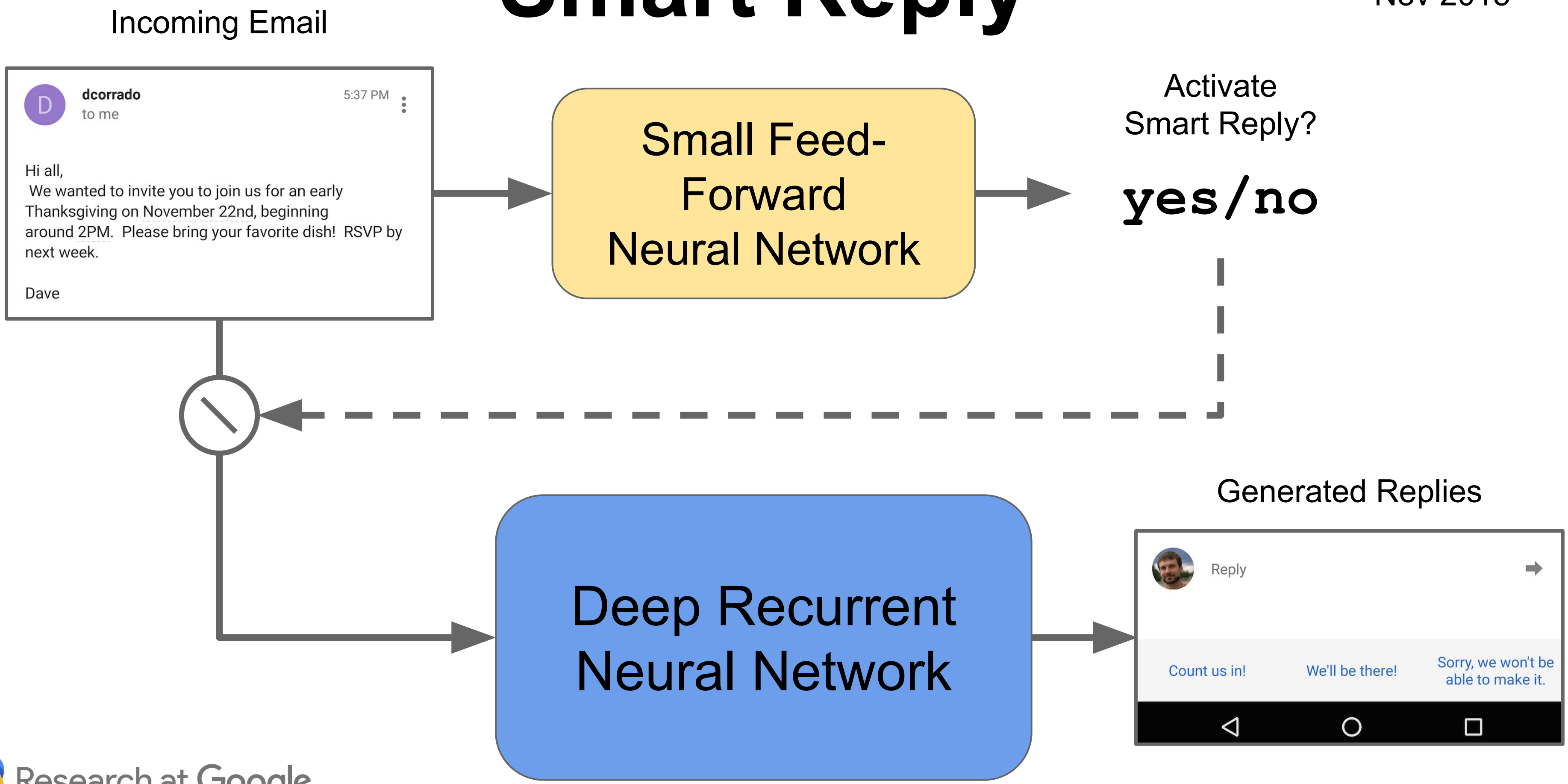
Incoming Email

Activate
Smart Reply?



Smart Reply

Google Research Blog
- Nov 2015



Sequence-to-Sequence

- **Translation:** [Kalchbrenner *et al.*, EMNLP 2013][Cho *et al.*, EMLP 2014][Sutskever & Vinyals & Le, NIPS 2014][Luong *et al.*, ACL 2015][Bahdanau *et al.*, ICLR 2015]
- **Image captions:** [Mao *et al.*, ICLR 2015][Vinyals *et al.*, CVPR 2015][Donahue *et al.*, CVPR 2015][Xu *et al.*, ICML 2015]
- **Speech:** [Chorowsky *et al.*, NIPS DL 2014][Chan *et al.*, arxiv 2015]
- **Language Understanding:** [Vinyals & Kaiser *et al.*, NIPS 2015][Kiros *et al.,* NIPS 2015]
- **Dialogue:** [Shang *et al.*, ACL 2015][Sordoni *et al.*, NAACL 2015][Vinyals & Le, ICML DL 2015]
- **Video Generation:** [Srivastava *et al.*, ICML 2015]
- **Algorithms:** [Zaremba & Sutskever, arxiv 2014][Vinyals & Fortunato & Jaitly, NIPS 2015][Kaiser & Sutskever, arxiv 2015][Zaremba *et al.*, arxiv 2015]

Image Captioning

[Vinyals *et al.*, CVPR 2015]

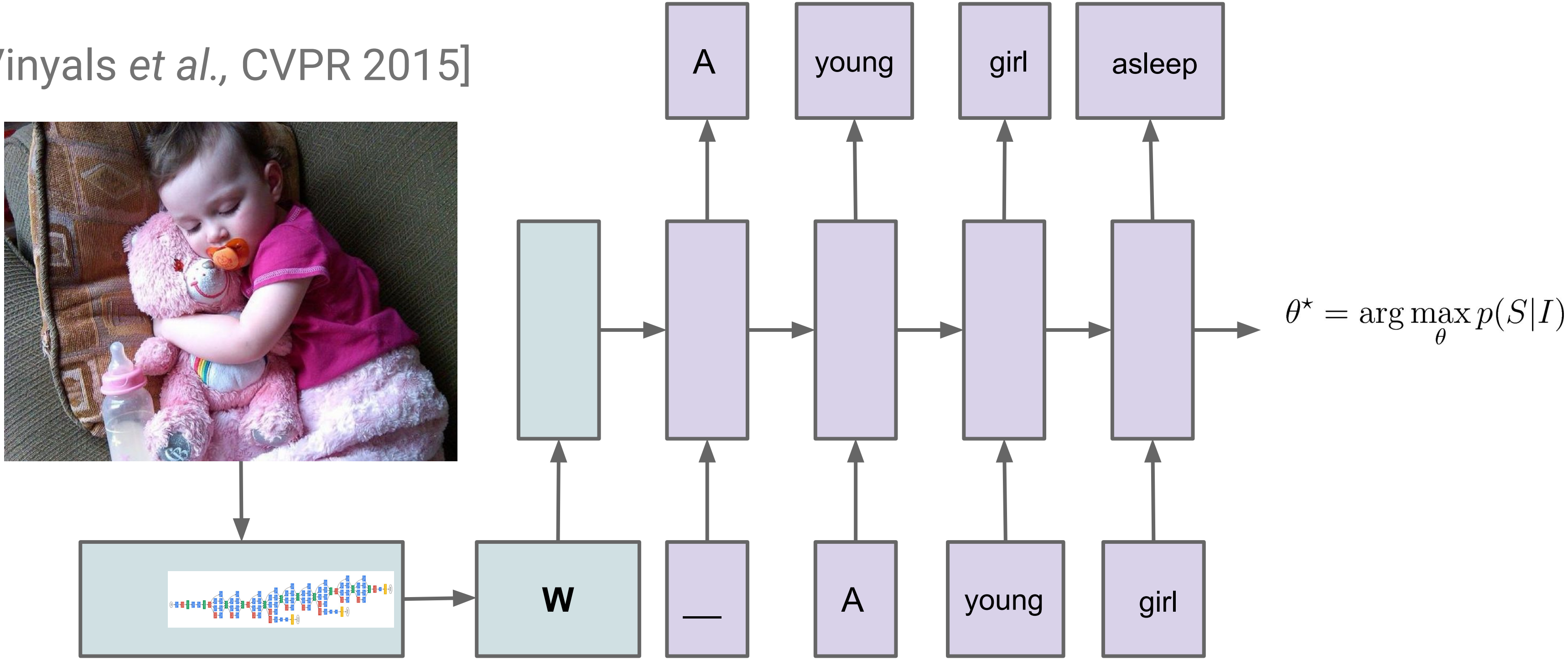
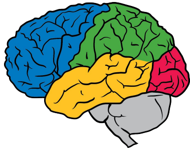


Image Captioning

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

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

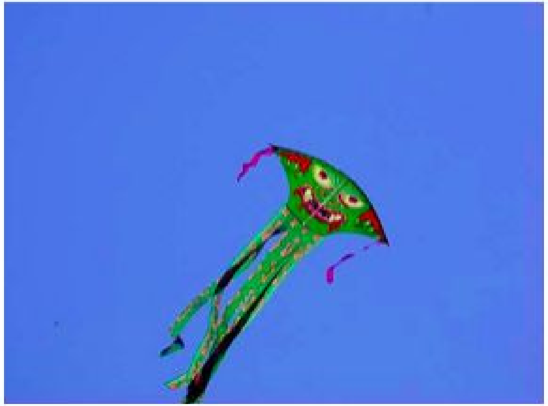
Model: A baby is asleep next to a teddy bear.



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

Combined Vision + Translation

Turnaround Time and Effect on Research

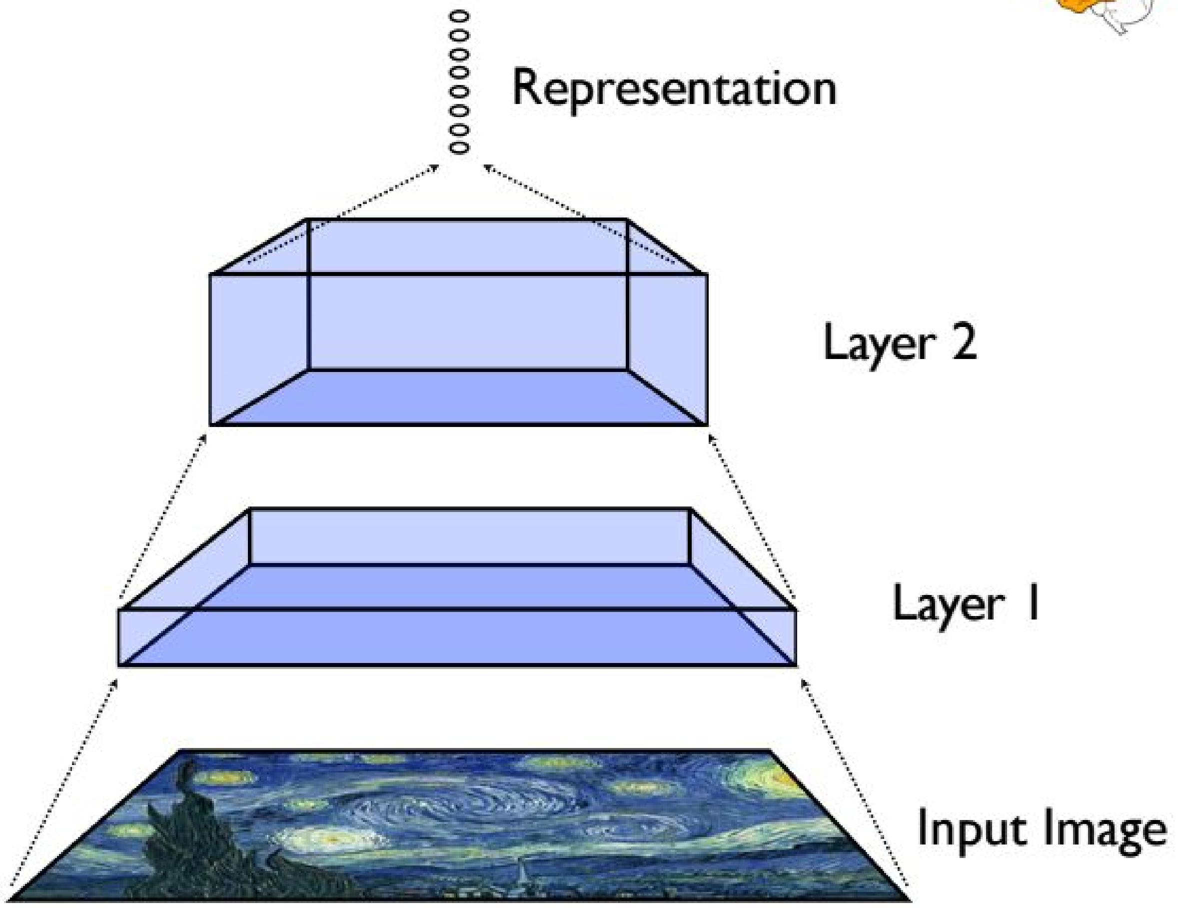
- Minutes, Hours:
  - **Interactive research! Instant gratification!**
- 1-4 days
  - Tolerable
  - Interactivity replaced by running many experiments in parallel
- 1-4 weeks:
  - High value experiments only
  - Progress stalls
- >1 month
  - Don’t even try

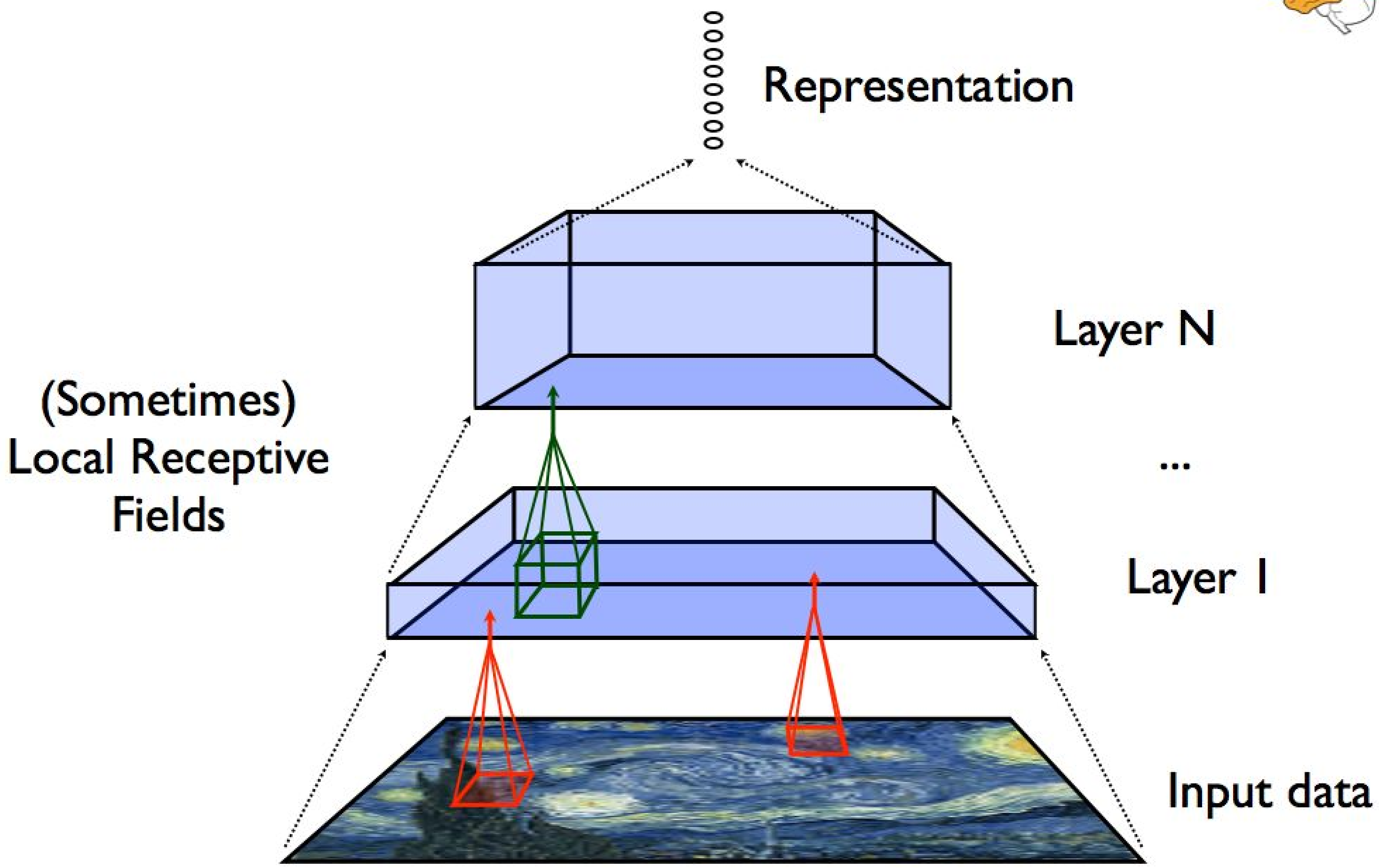
Train in a day what would take a single GPU card 6 weeks

How Can We Train Large, Powerful Models Quickly?

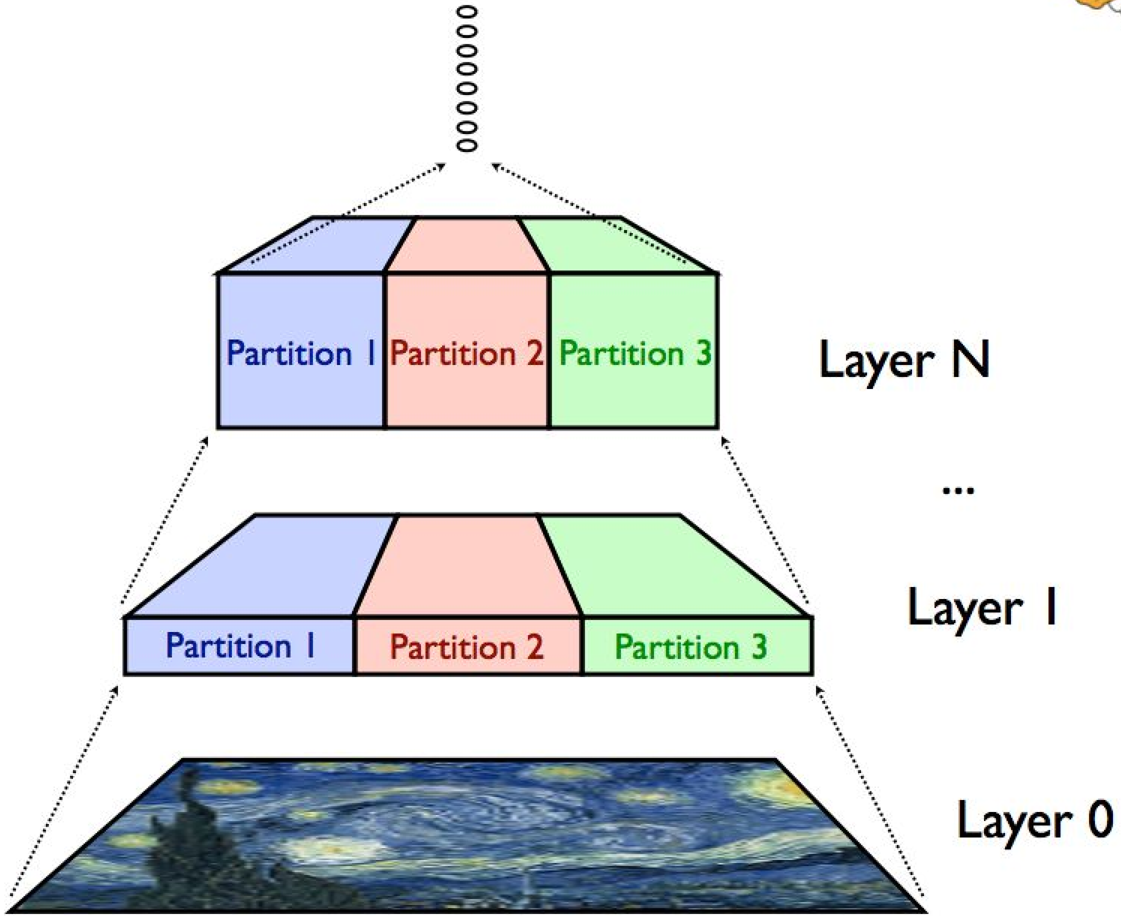
- Exploit many kinds of parallelism
  - Model parallelism
  - Data parallelism

Model Parallelism

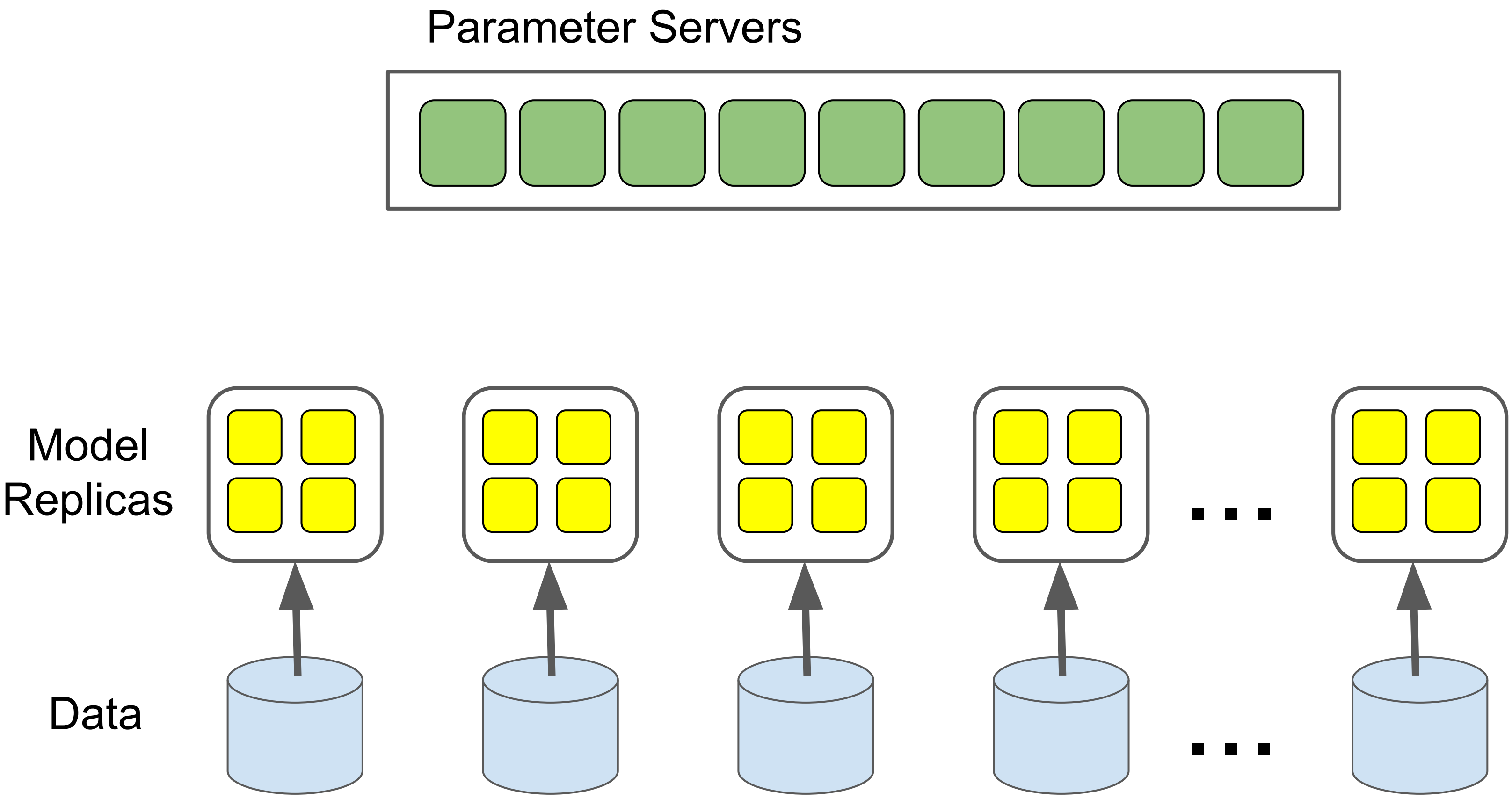



Representation
Layer N
(Sometimes)
Local Receptive
Fields
...
Layer 1
Input data

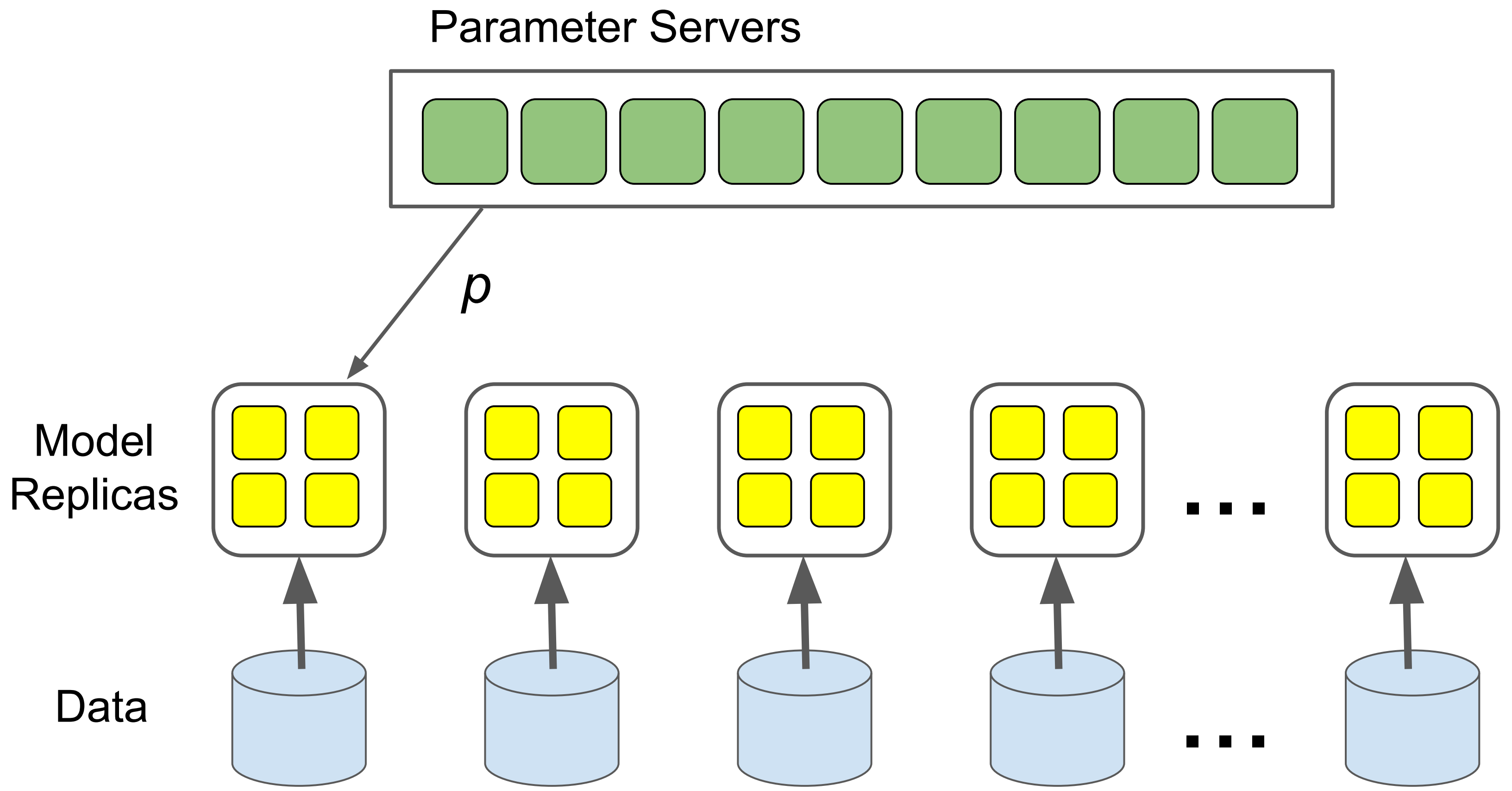
Model Parallelism: Partition model across machines



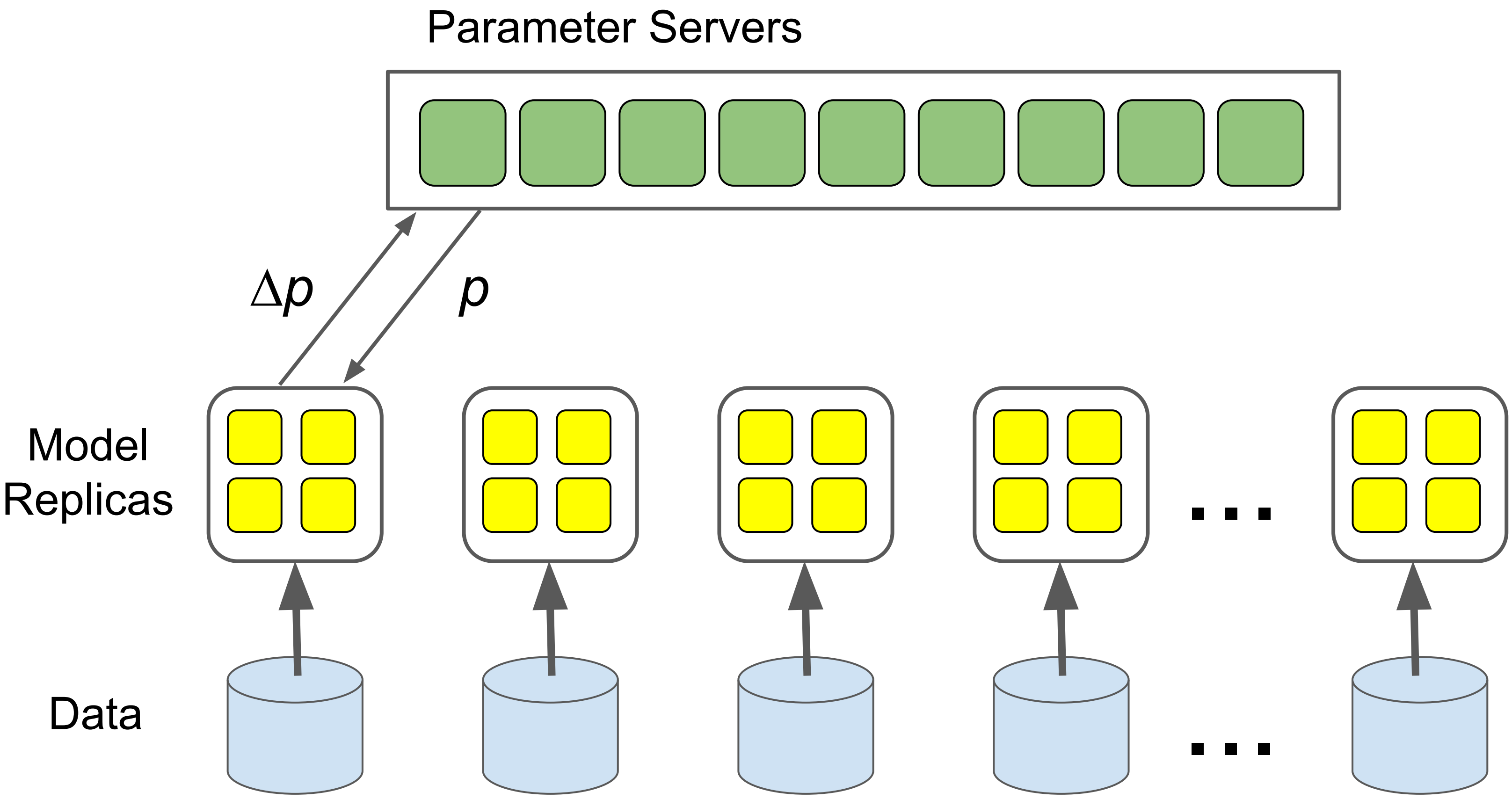
Data Parallelism



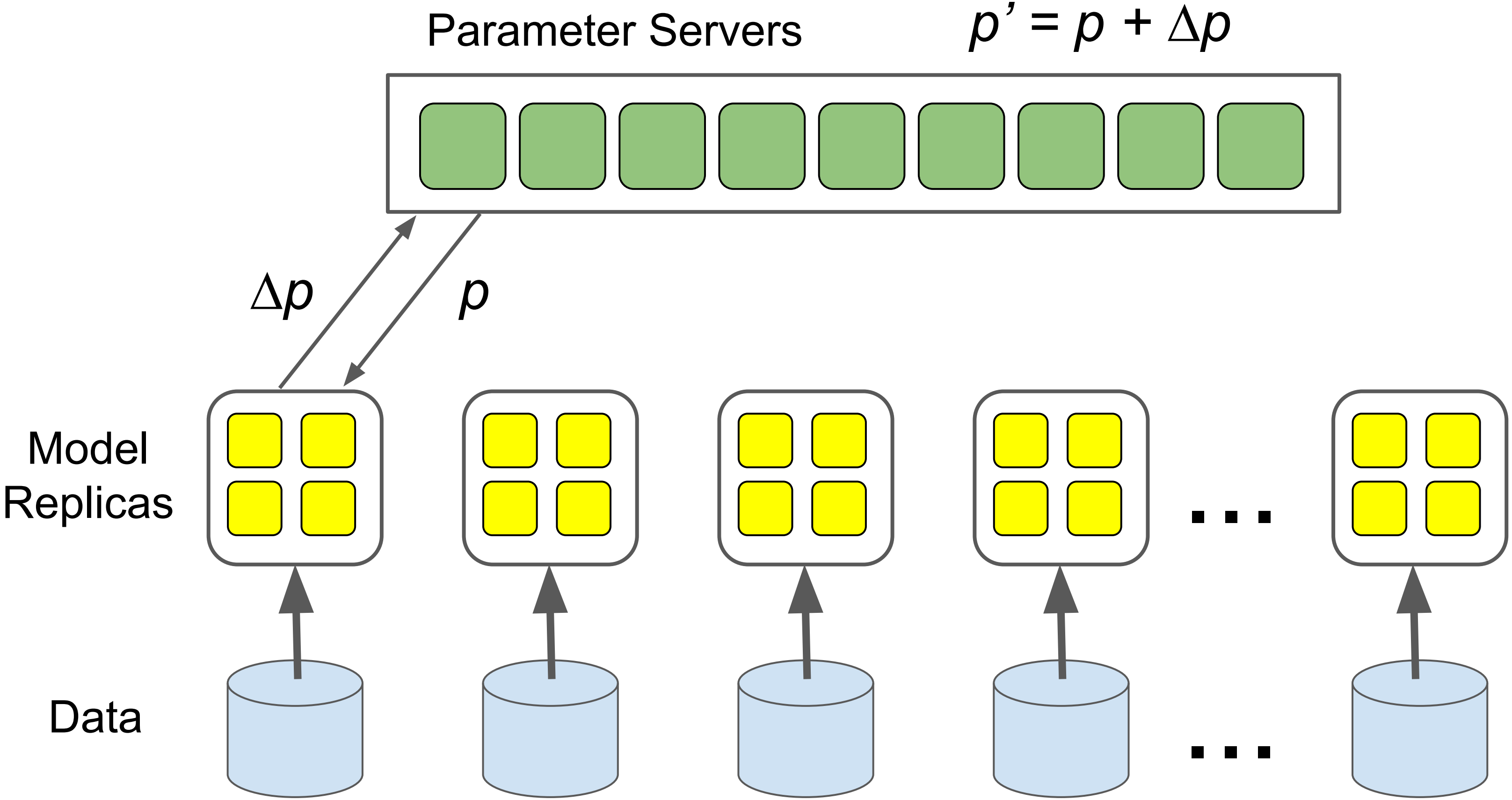
Data Parallelism



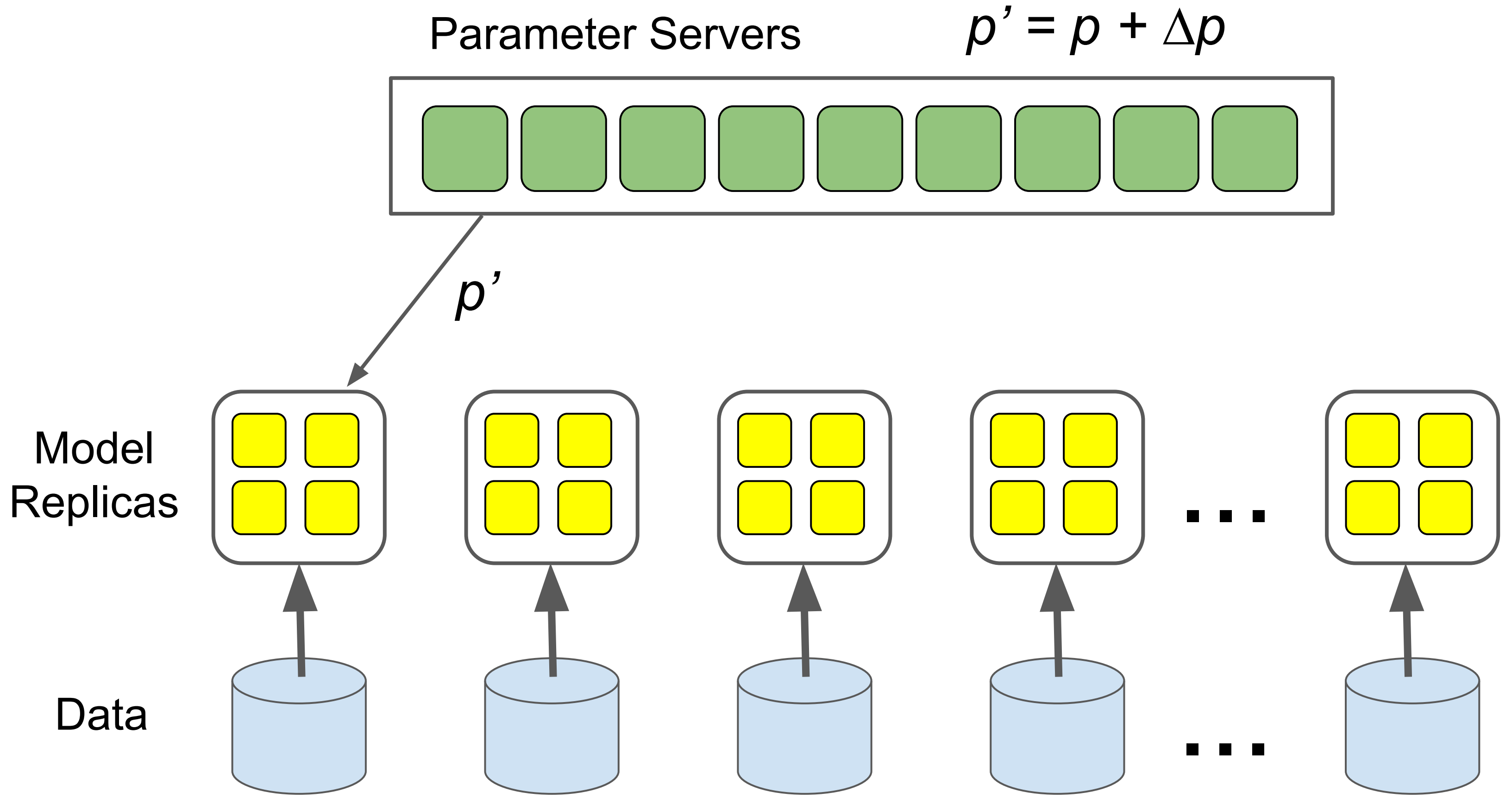
Data Parallelism



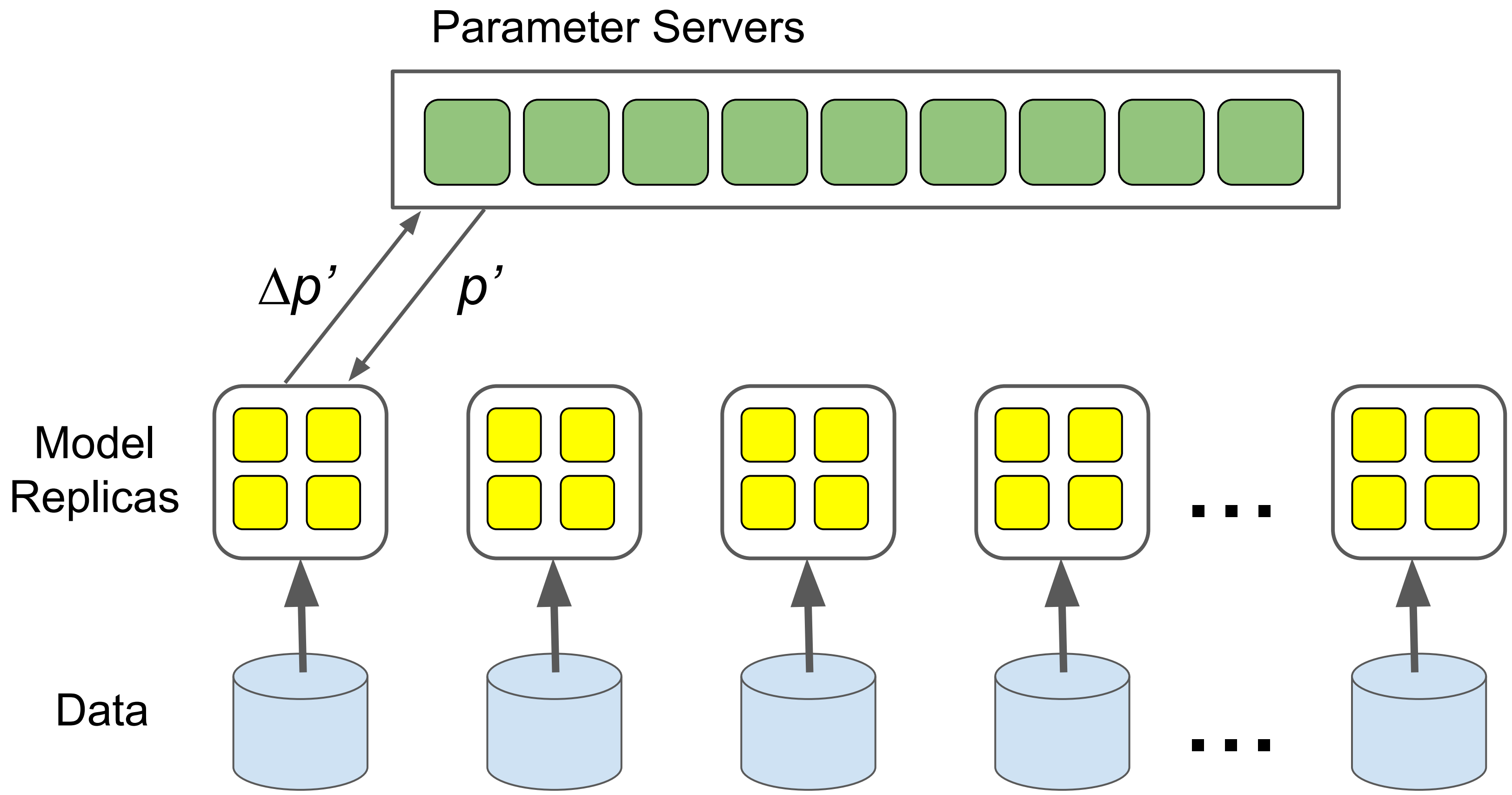
Data Parallelism



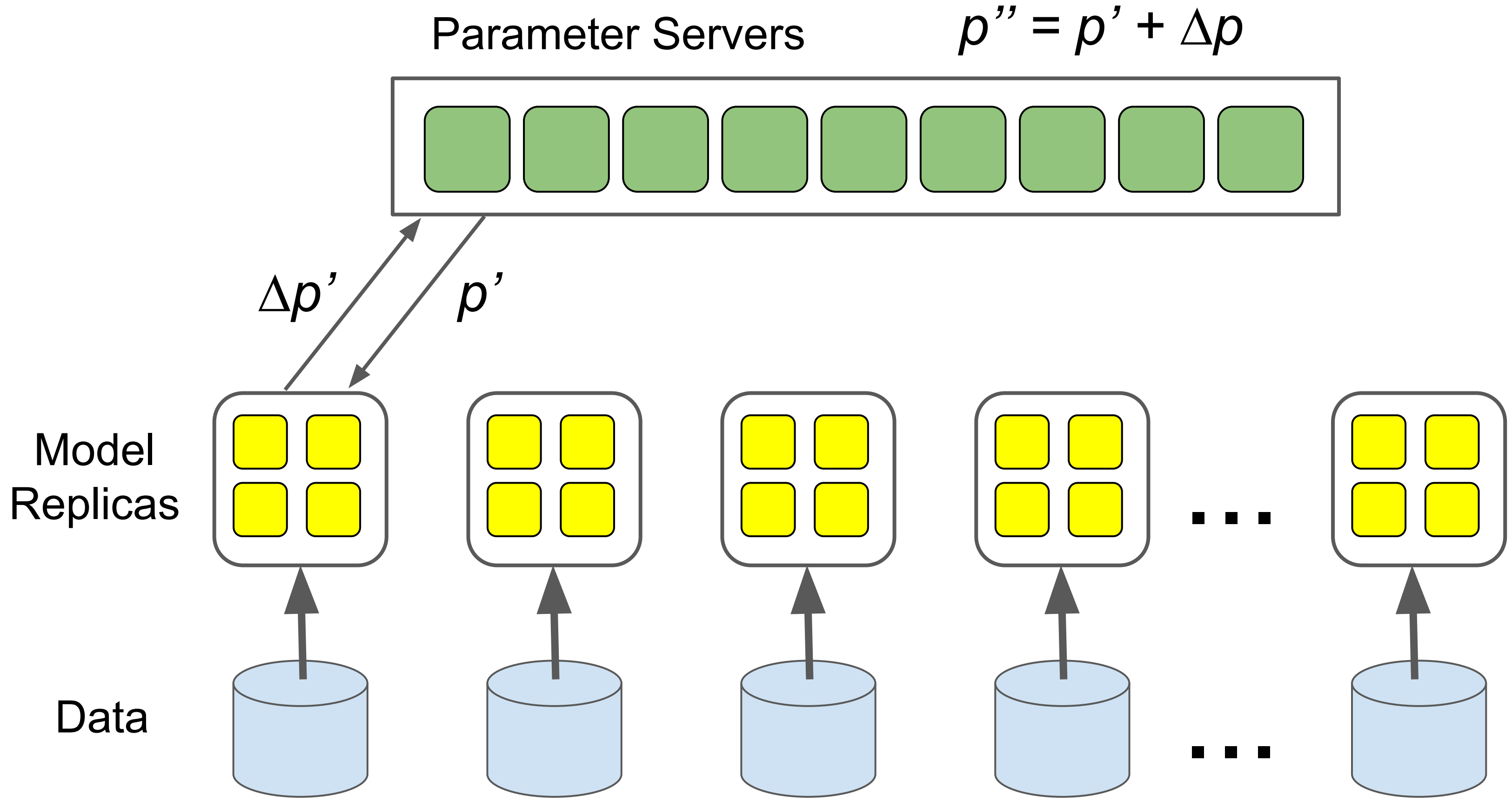
Data Parallelism



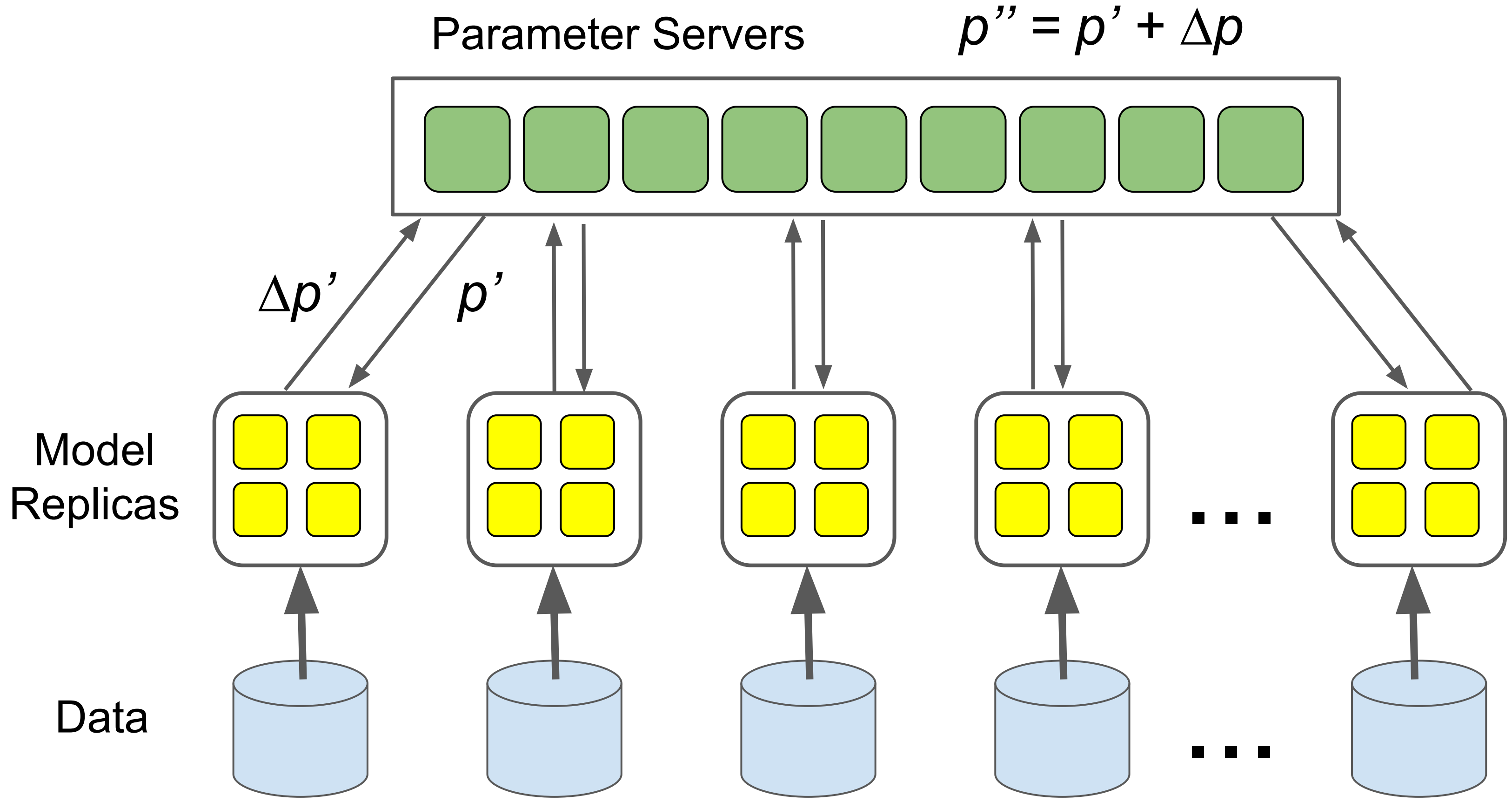
Data Parallelism



Data Parallelism



Data Parallelism



Data Parallelism Choices

Can do this **synchronously**:

- **N replicas** equivalent to an **N times larger batch size**
- Pro: No noise
- Con: Less fault tolerant (requires some recovery if any single machine fails)

Can do this **asynchronously**:

- Con: Noise in gradients
- Pro: Relatively fault tolerant (failure in model replica doesn't block other replicas)

(Or **hybrid**: M asynchronous groups of N synchronous replicas)

Image Model Training Time

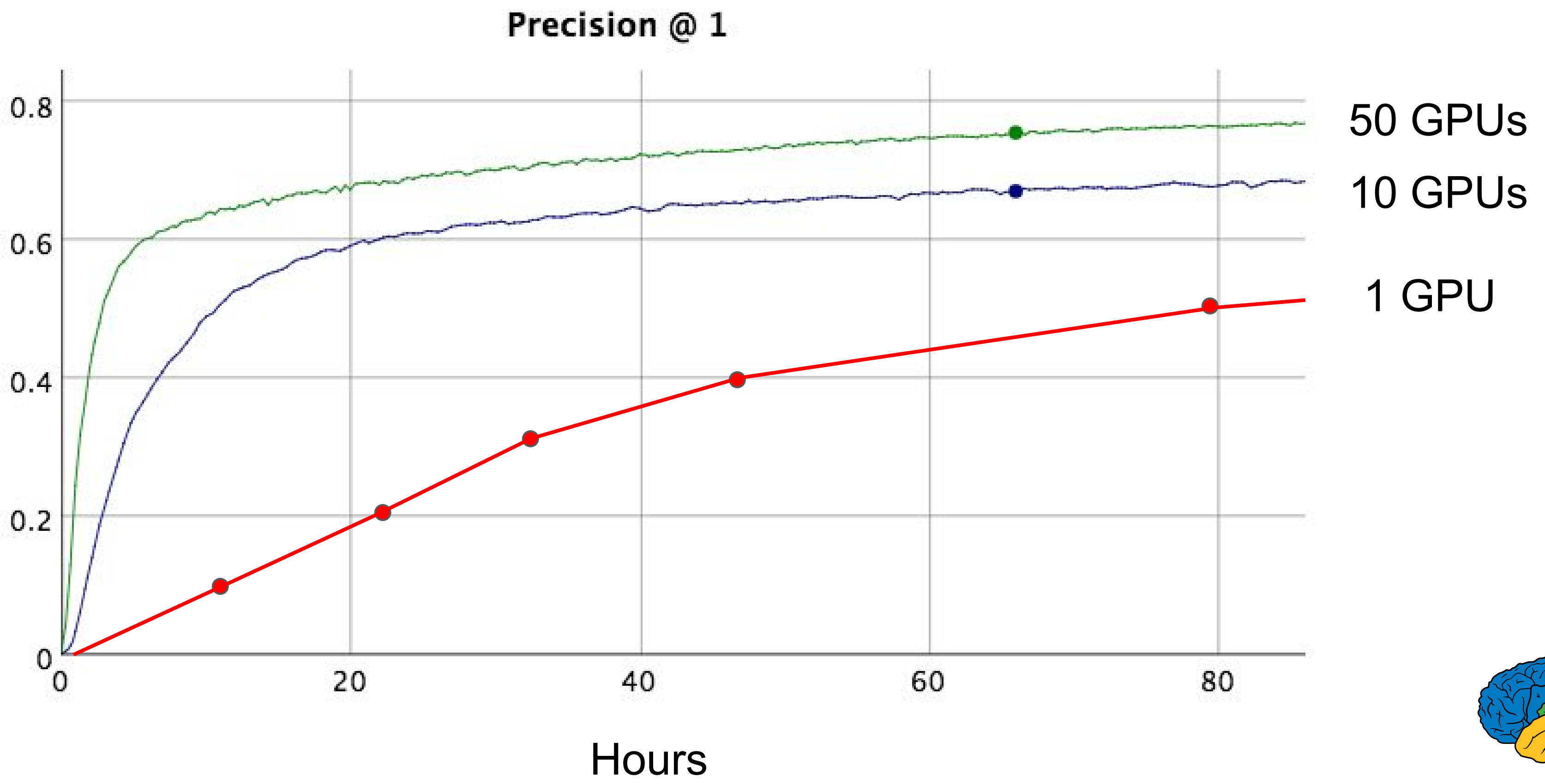
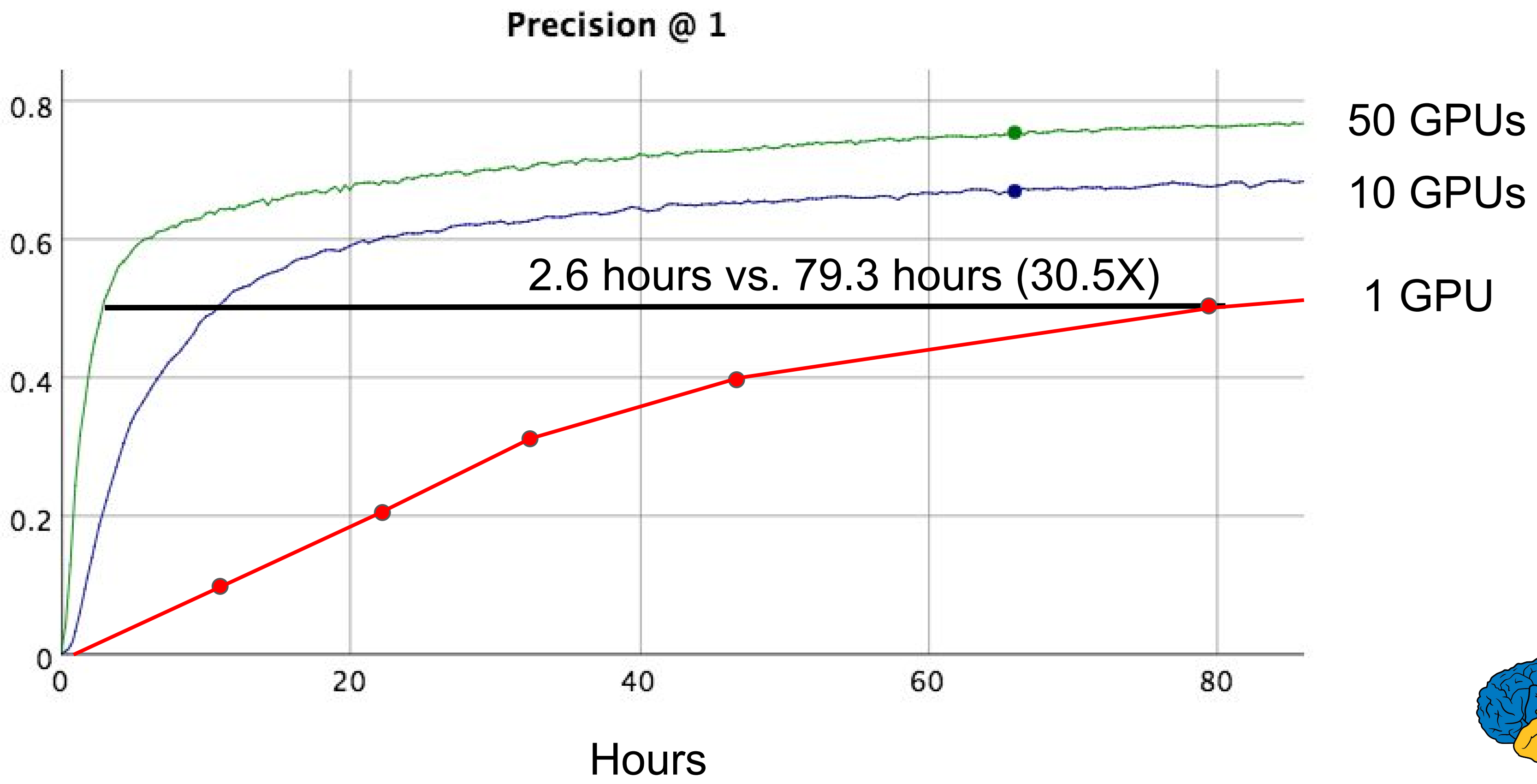


Image Model Training Time



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*

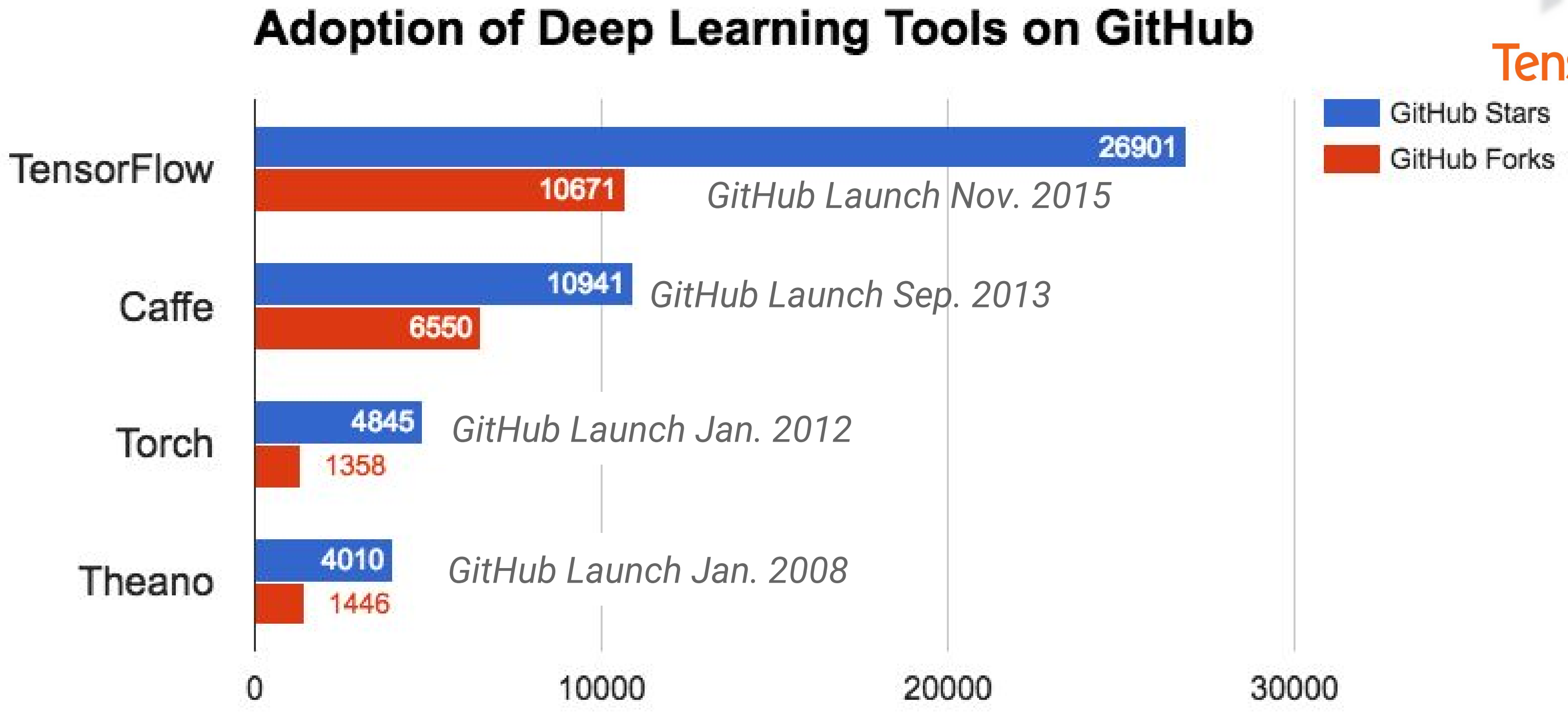
Abstract

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

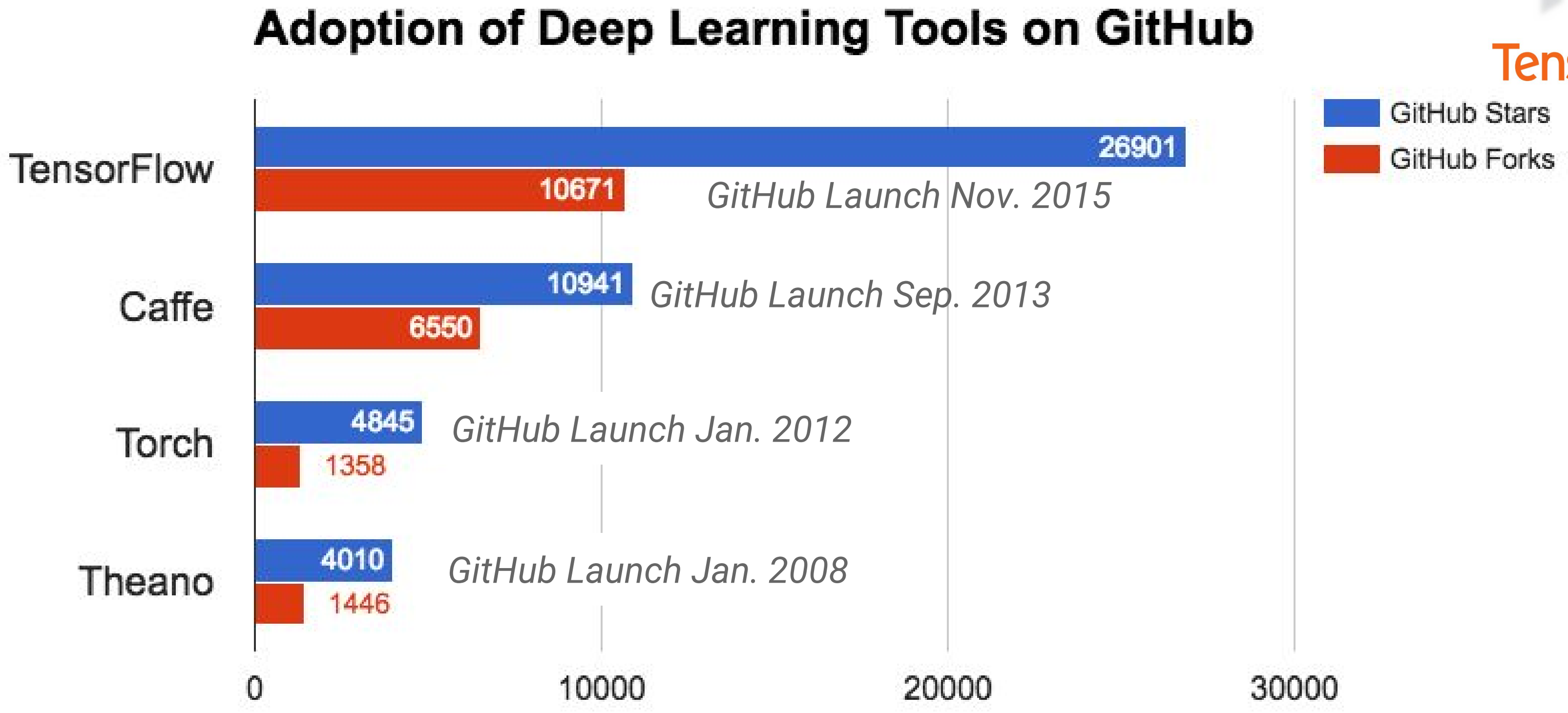
http://tensorflow.org/whitepaper2015.pdf

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 repository on GitHub in 2015 (despite only being available in Nov, '15)

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

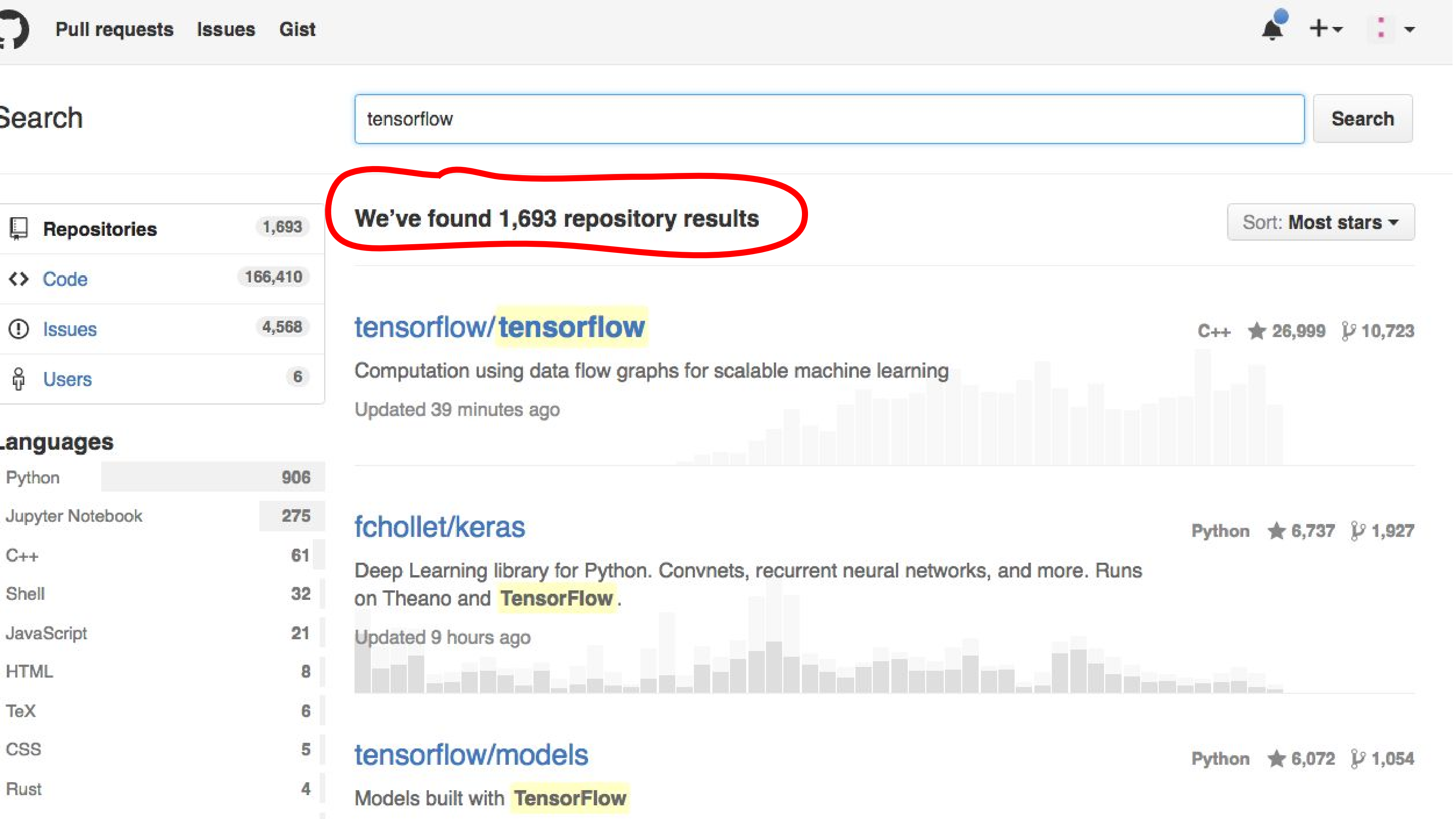
An introduction to RNNs, wherein we train an LSTM network to predict the next word in an English sentence. (A task sometimes called language modeling.)

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


Pull requests
Issues
Gist
Search
tensorflow
Search
We've found 1,693 repository results
Sort: Most stars
Repositories 1,693
Code 166,410
Issues 4,568
Users 6
Languages
Python 906
Jupyter Notebook 275
C++ 61
Shell 32
JavaScript 21
HTML 8
TeX 6
CSS 5
Rust 4
tensorflow/tensorflow
C++ 26,999 10,723
Computation using data flow graphs for scalable machine learning
Updated 39 minutes ago
fchollet/keras
Python 6,737 1,927
Deep Learning library for Python. Convnets, recurrent neural networks, and more. Runs on Theano and TensorFlow.
Updated 9 hours ago
tensorflow/models
Python 6,072 1,054
Models built with TensorFlow

Motivations

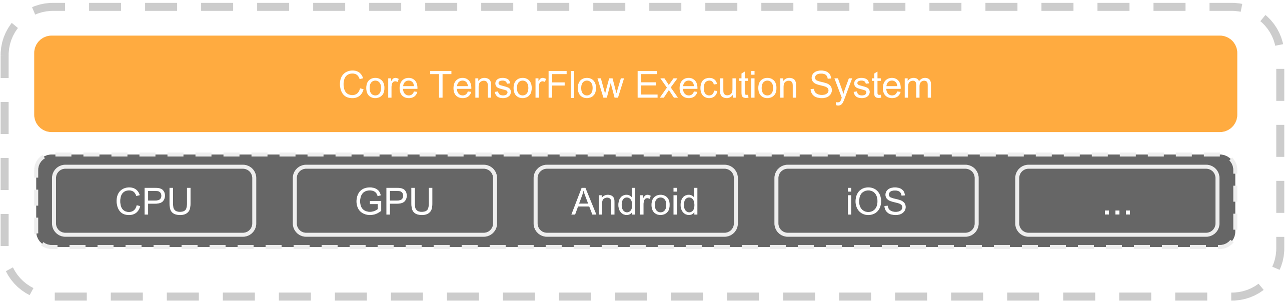
DistBelief (1st system) was great for scalability, and production training of basic kinds of models

Not as flexible as we wanted for research purposes

Better understanding of problem space allowed us to make some dramatic simplifications

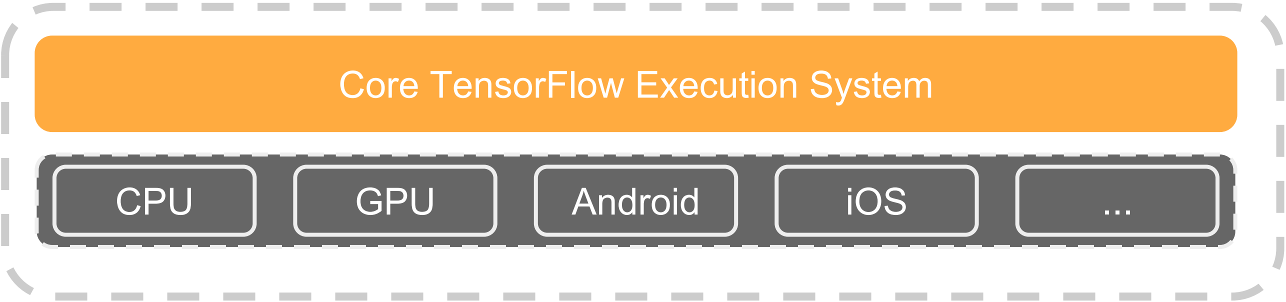
TensorFlow: Expressing High-Level ML Computations

- Core in C++
  - Very low overhead



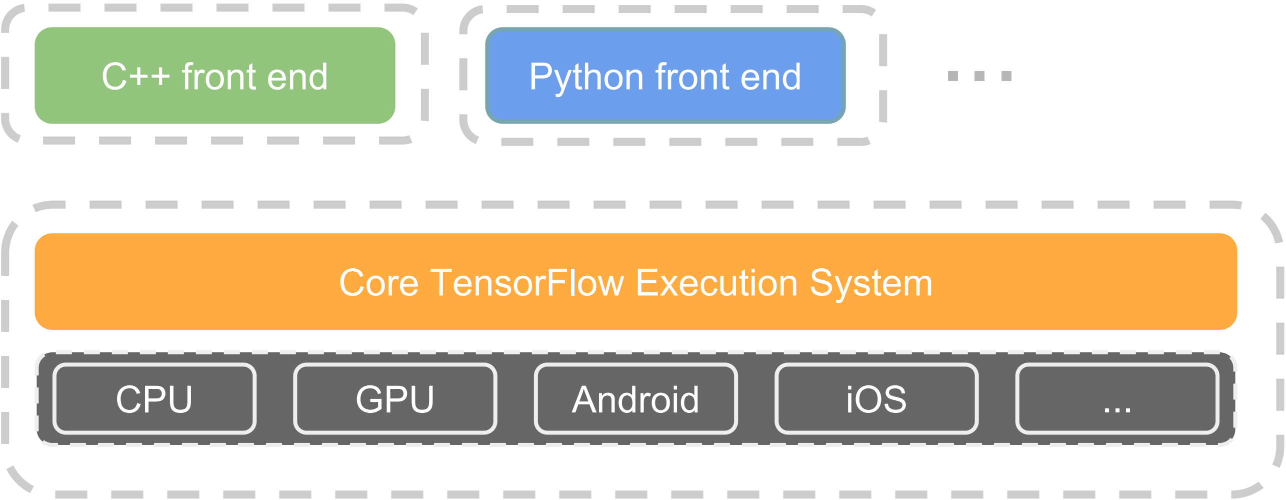
TensorFlow: Expressing High-Level ML Computations

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



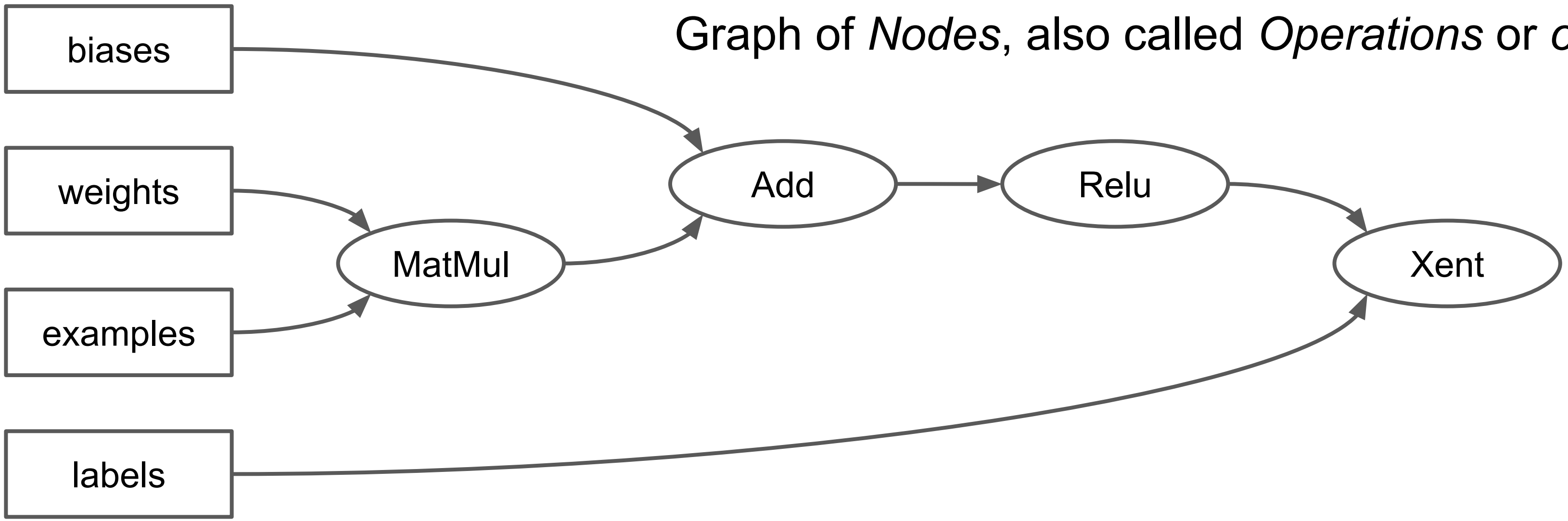
TensorFlow: Expressing High-Level ML Computations

- 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

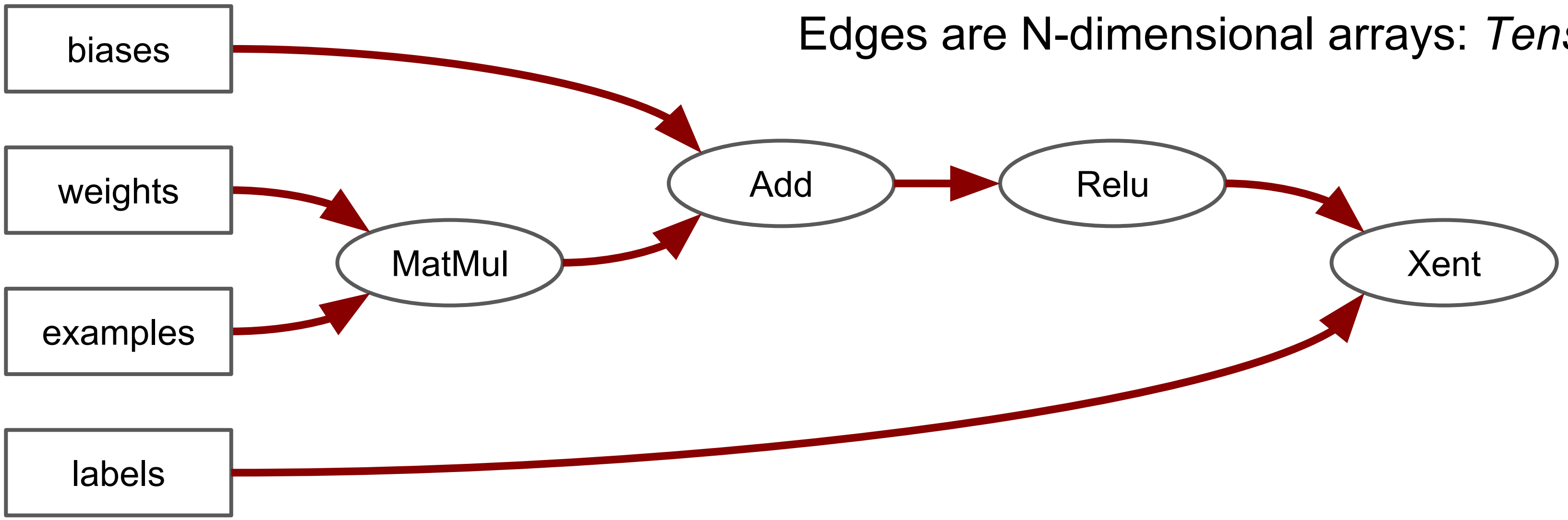
Graph of *Nodes*, also called *Operations* or *ops.*



Computation is a dataflow graph

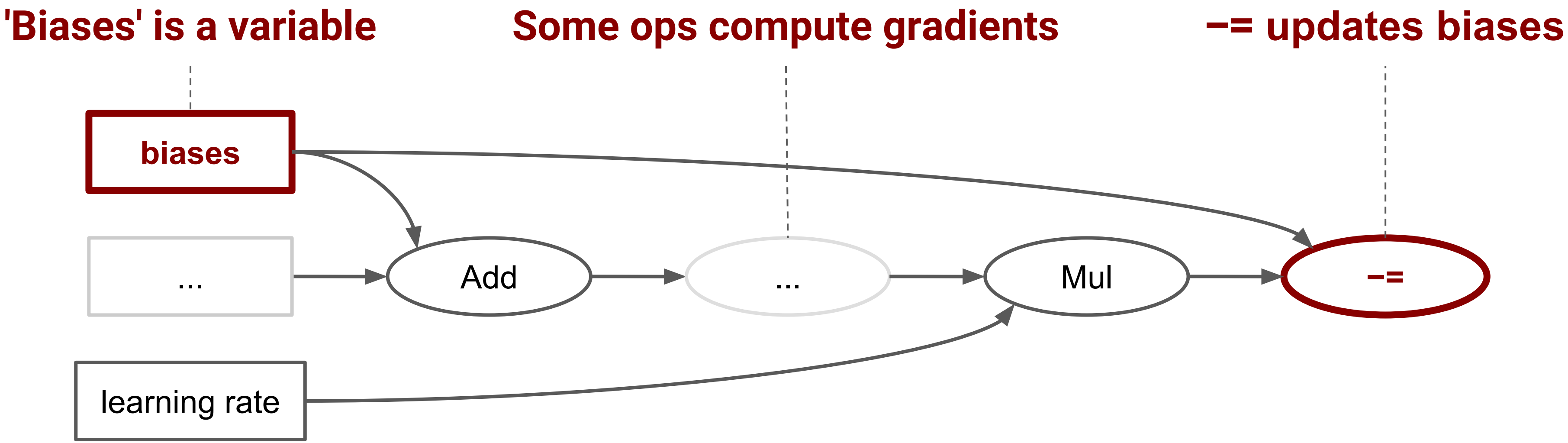
with tensors

Edges are N-dimensional arrays: *Tensors*



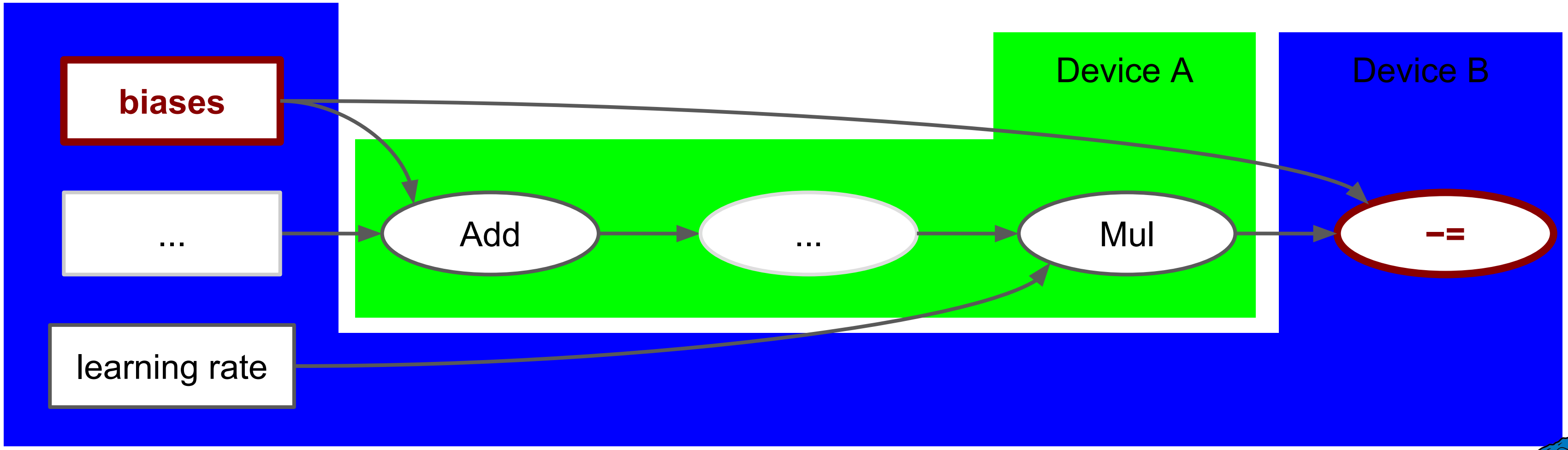
Computation is a dataflow graph

with state



Computation is a dataflow graph

distributed



Devices: Processes, Machines, GPUs, etc

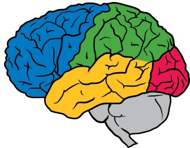
TensorFlow: Expressing High-Level ML Computations

Automatically runs models on range of platforms:

from **phones** ...

to **single machines** (CPU and/or GPUs) …

to **distributed systems** of many 100s of GPU cards



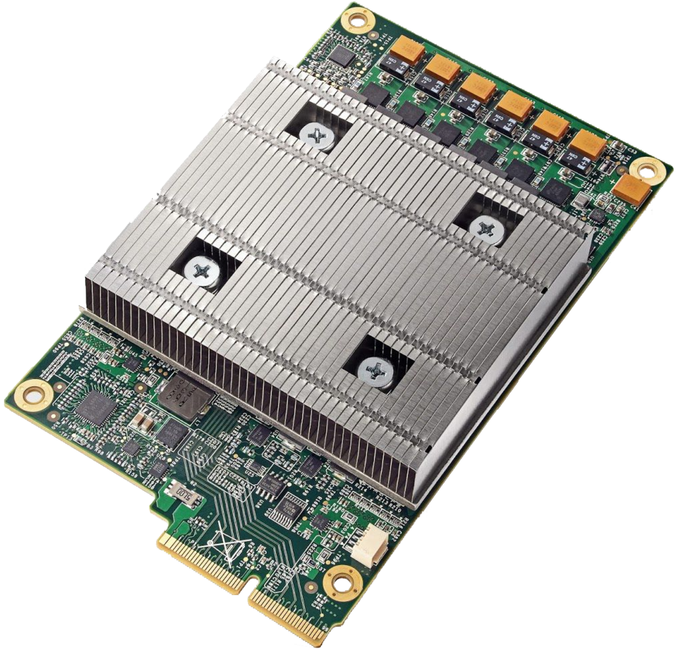
Trend: Much More Heterogeneous hardware

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 >14 months: used on every search query, used for AlphaGo match, ...

Using TensorFlow for Parallelism

Trivial to express both model parallelism as well as data parallelism

- Very minimal changes to single device model code

Example: LSTM

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

Example: Deep LSTM

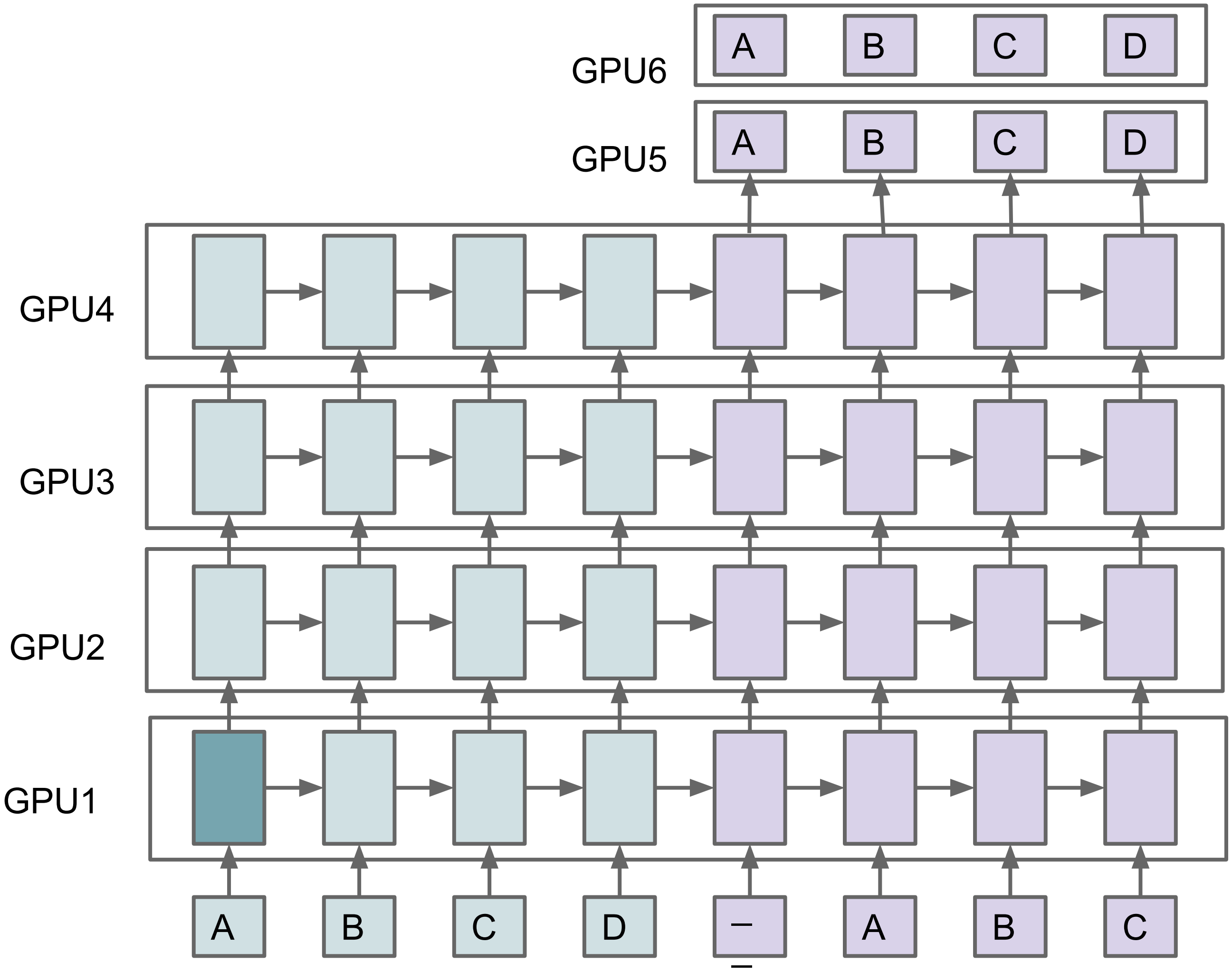
```
for i in range(20):
  for d in range(4): # d is depth
    input = x[i] if d is 0 else m[d-1]
    m[d], c[d] = LSTMCell(input, mprev[d], cprev[d])
    mprev[d] = m[d]
    cprev[d] = c[d]
```

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]
```

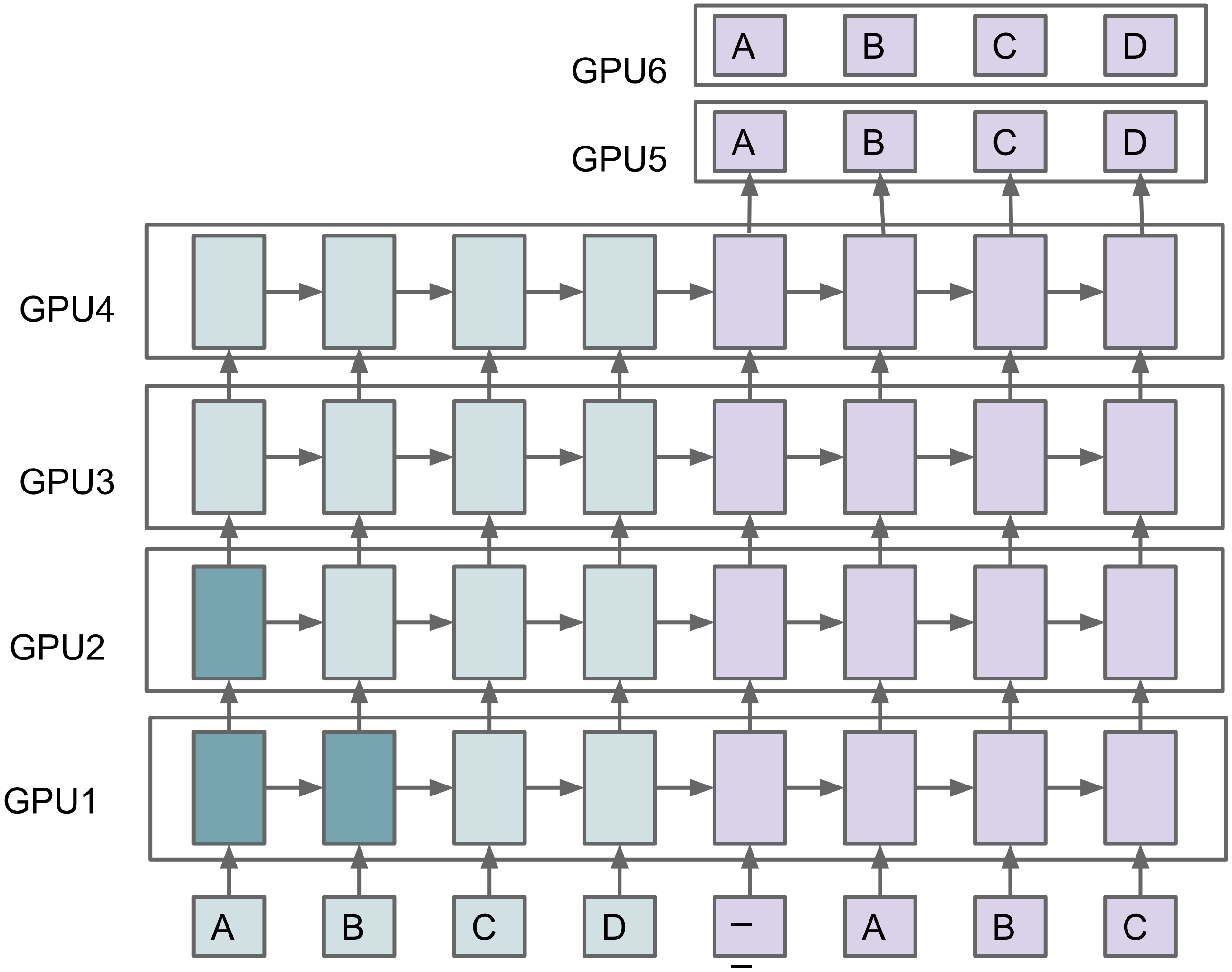


80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence

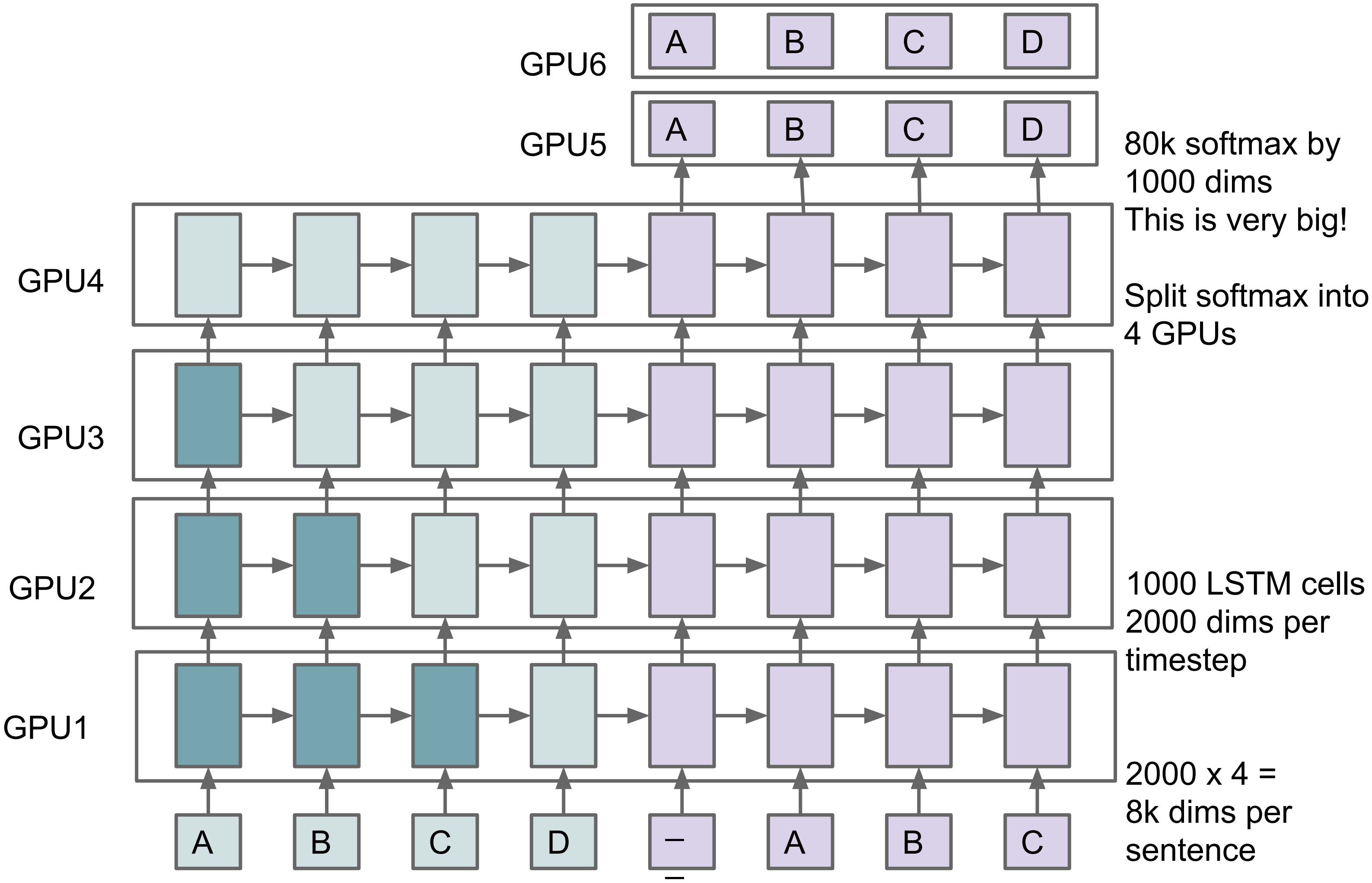


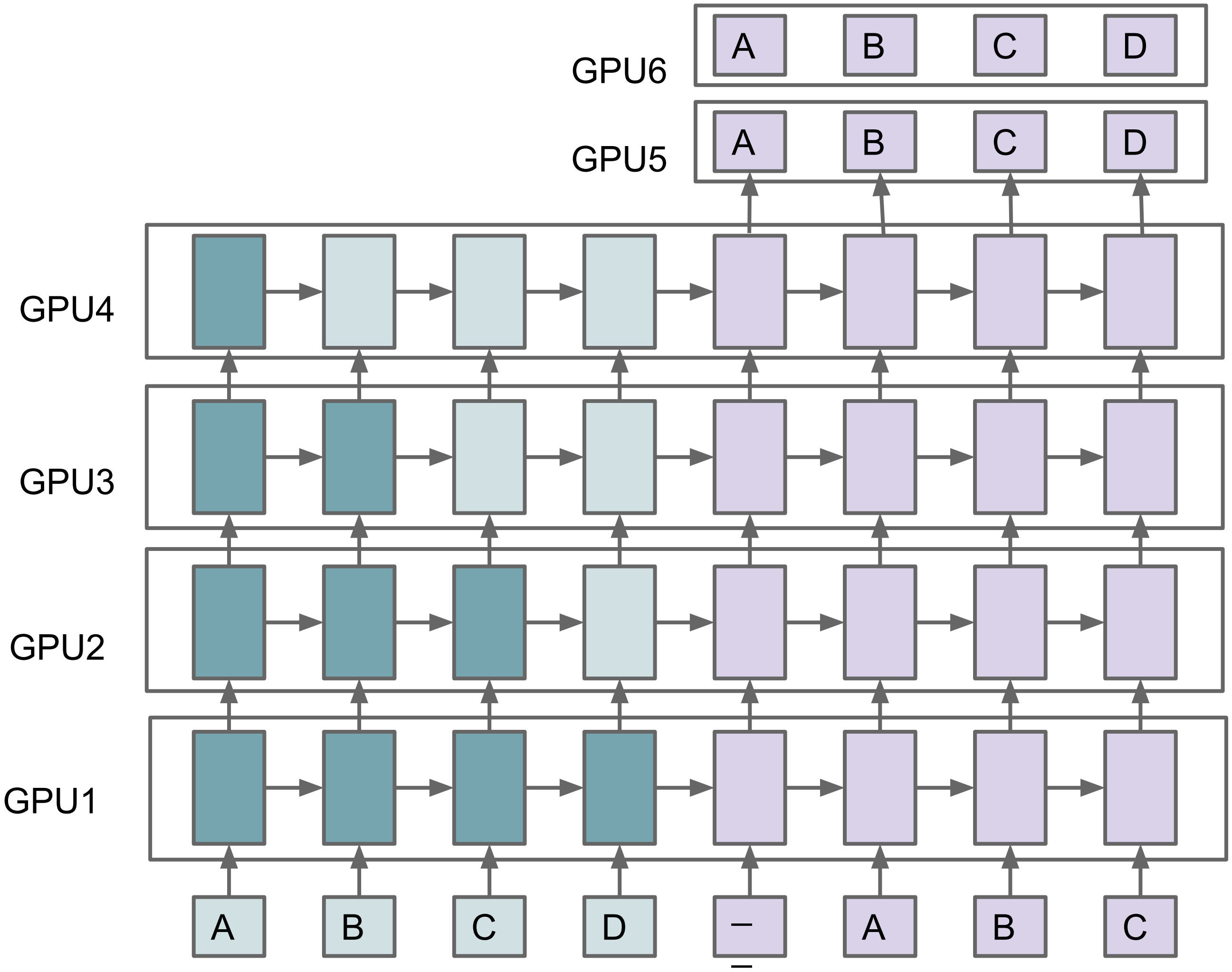
80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence


GPU6
A
B
C
D
GPU5
A
B
C
D
80k softmax by
1000 dims
This is very big!
Split softmax into
4 GPUs
GPU4
GPU3
GPU2
GPU1
1000 LSTM cells
2000 dims per
timestep
2000 x 4 =
8k dims per
sentence
A
B
C
D
_
A
B
C

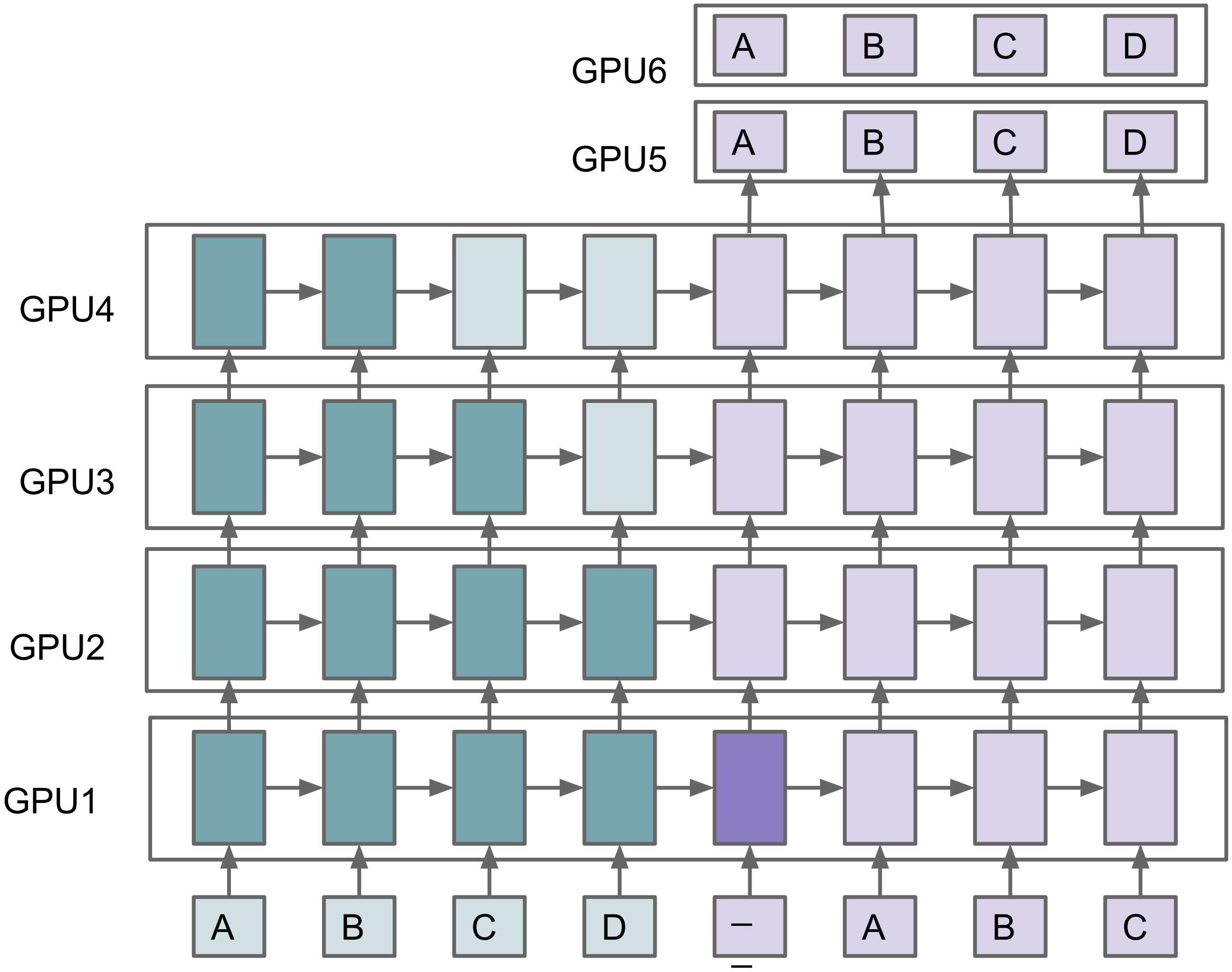


80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence

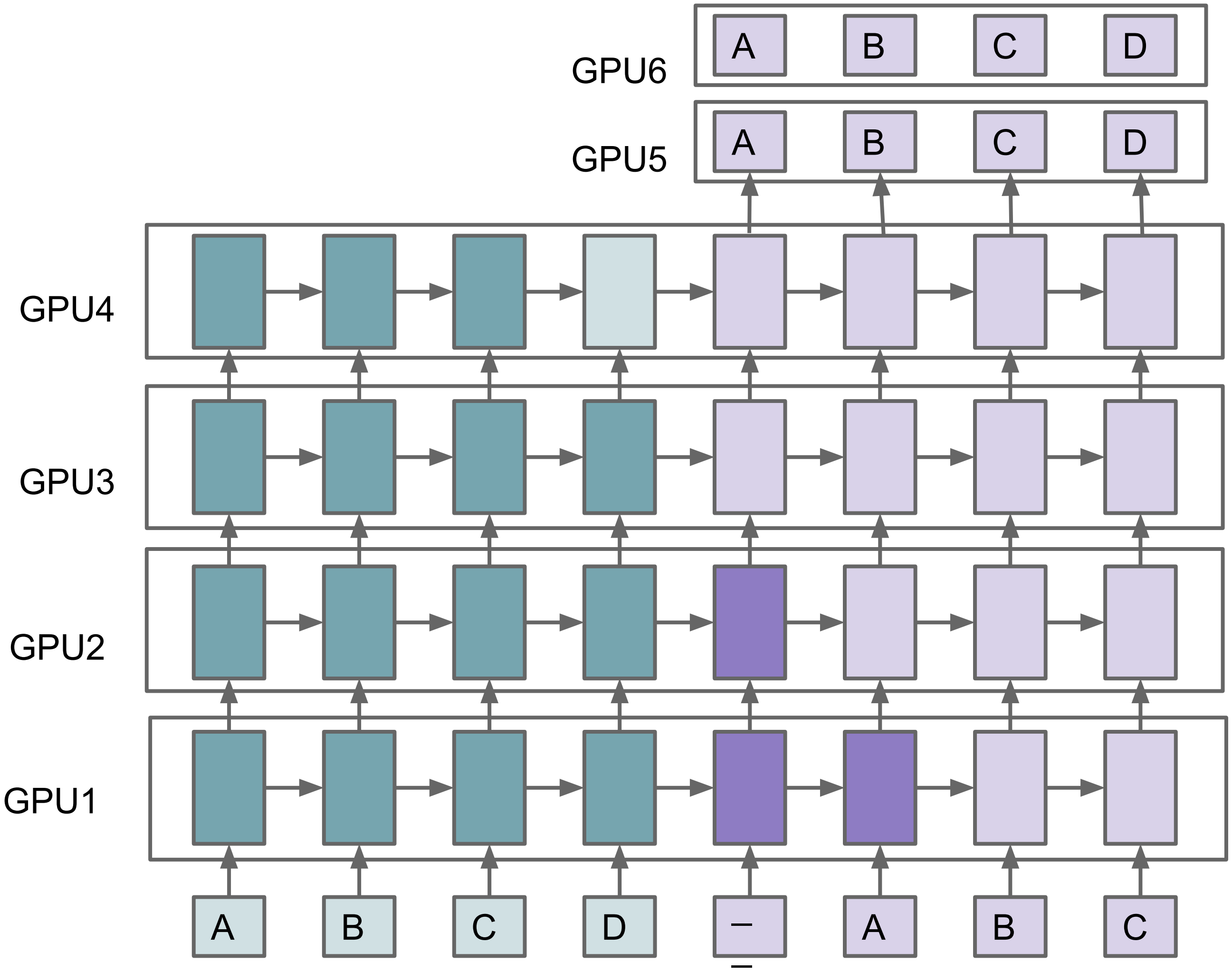


80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence

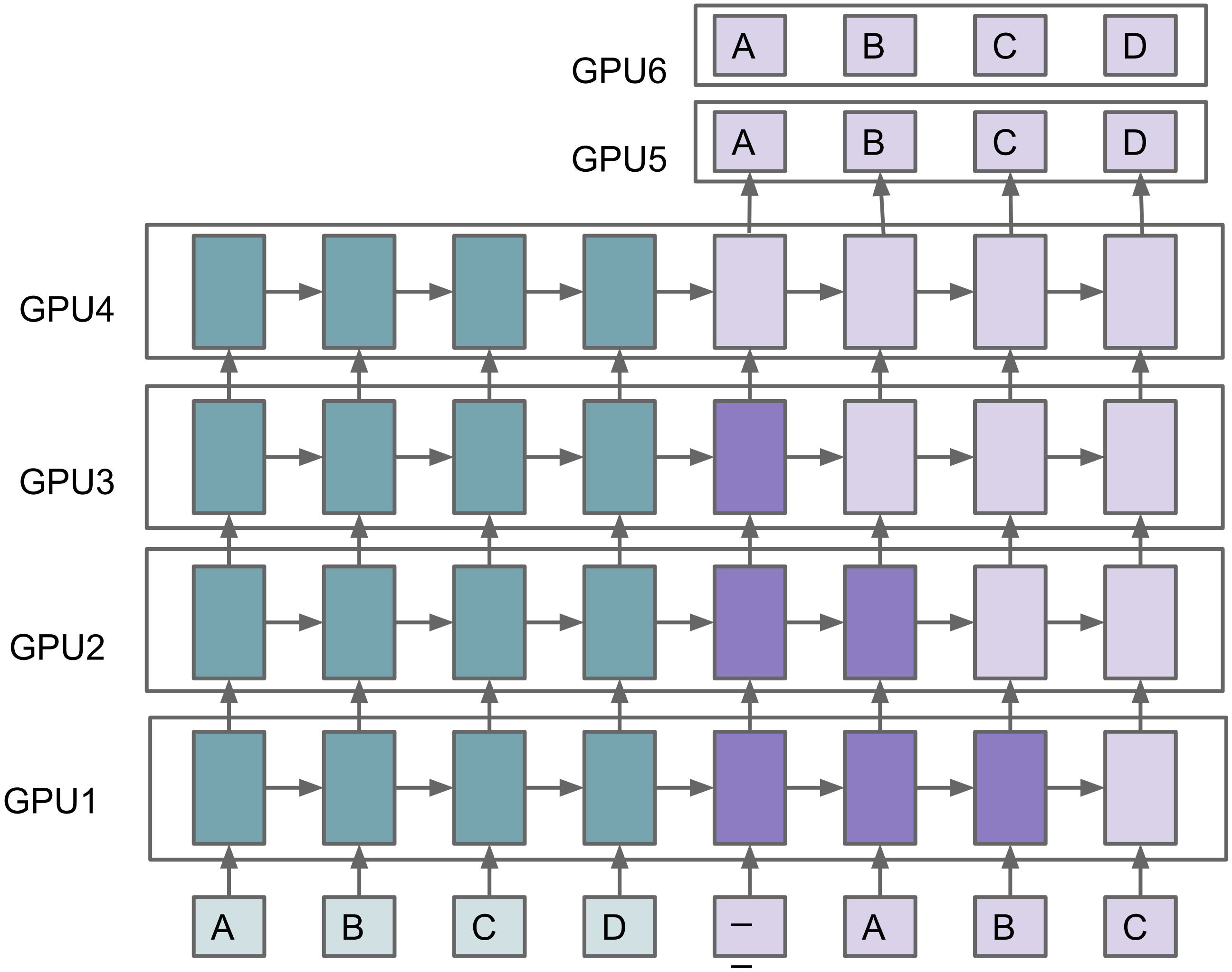


80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence

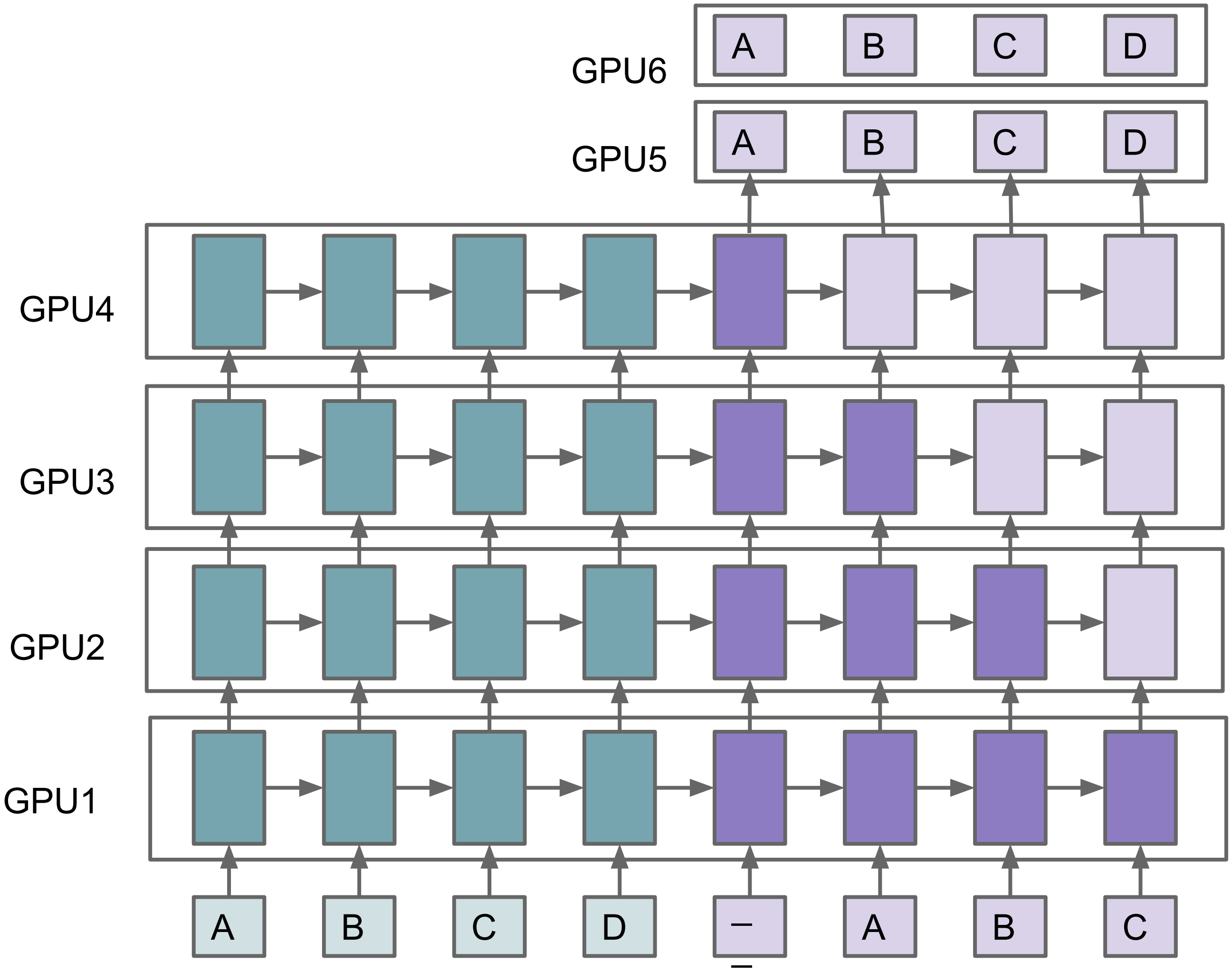


80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence

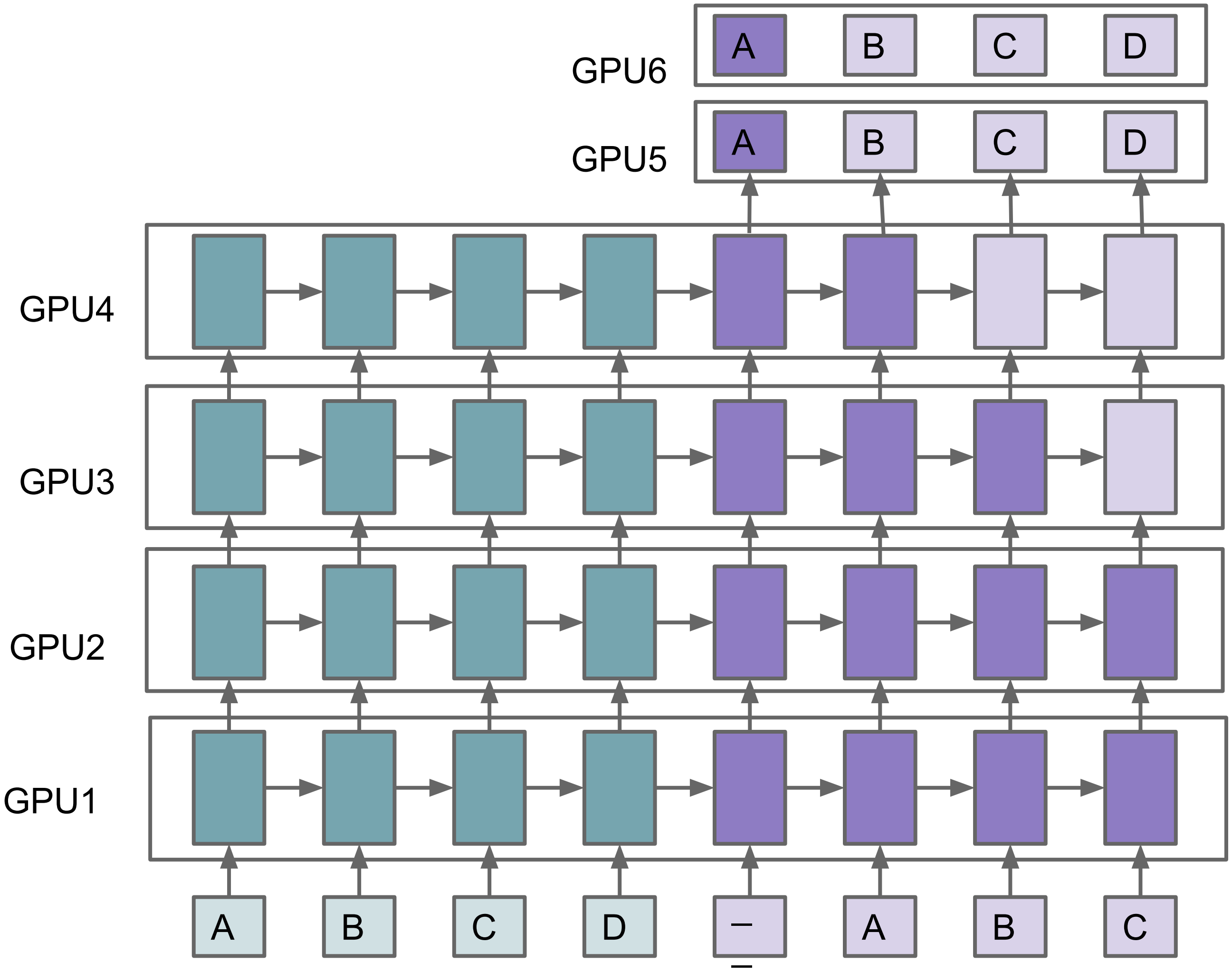


80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence

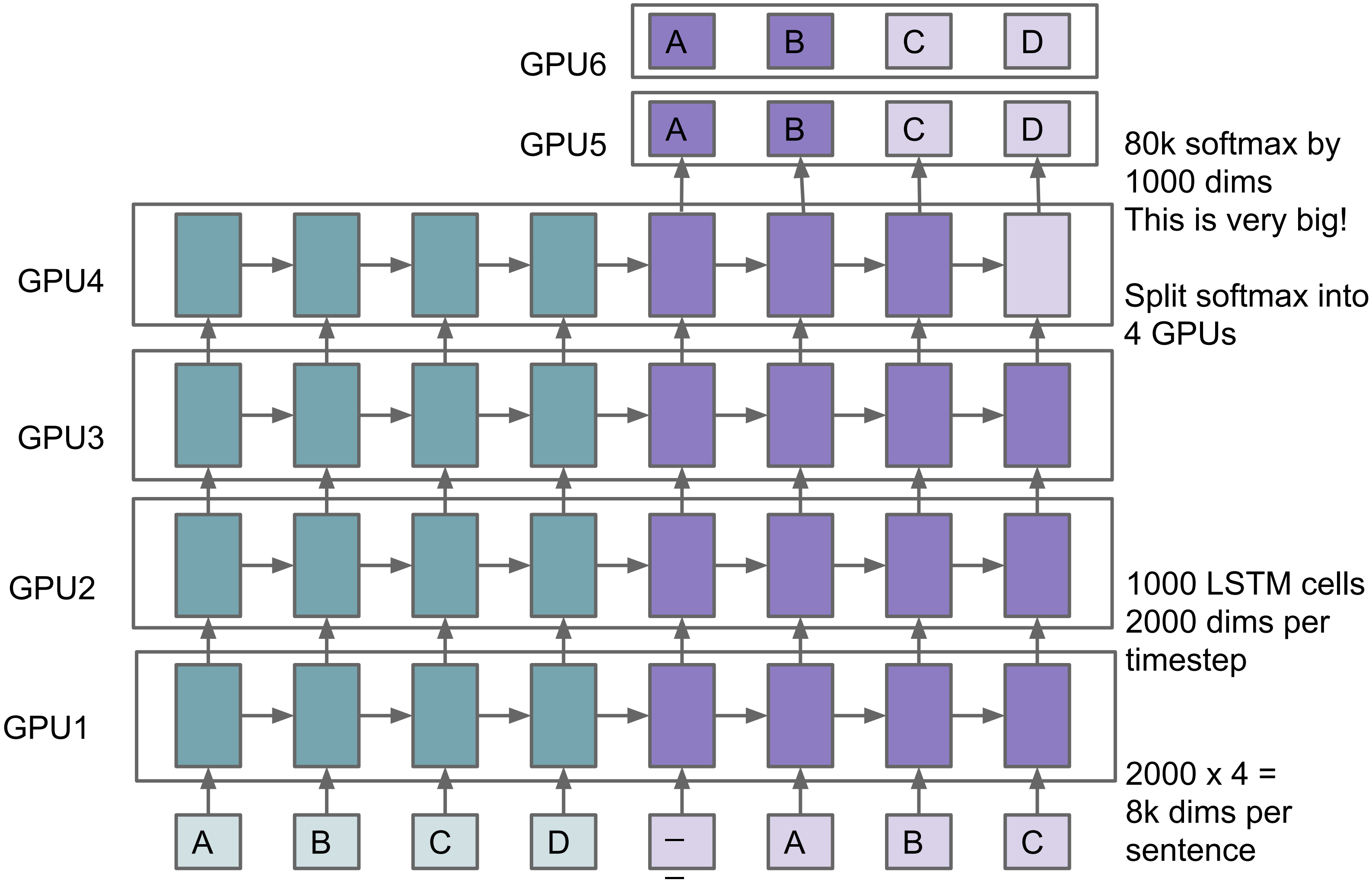


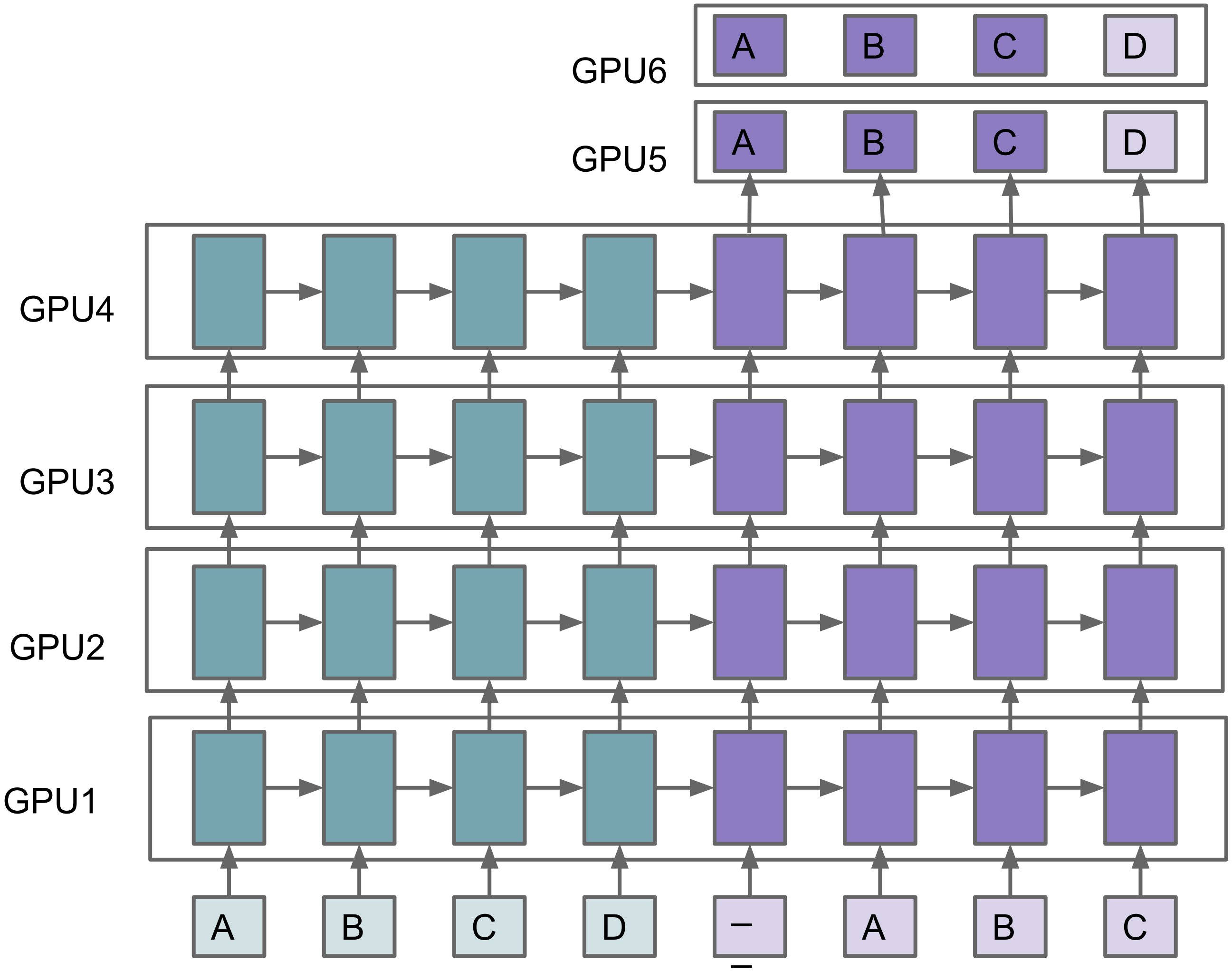
80k softmax by
1000 dims
This is very big!

Split softmax into
4 GPUs

1000 LSTM cells
2000 dims per
timestep

2000 x 4 =
8k dims per
sentence


GPU6
A
B
C
D
GPU5
A
B
C
D
80k softmax by
1000 dims
This is very big!
Split softmax into
4 GPUs
GPU4
GPU3
GPU2
1000 LSTM cells
2000 dims per
timestep
GPU1
2000 x 4 =
8k dims per
sentence
A
B
C
D
_
A
B
C

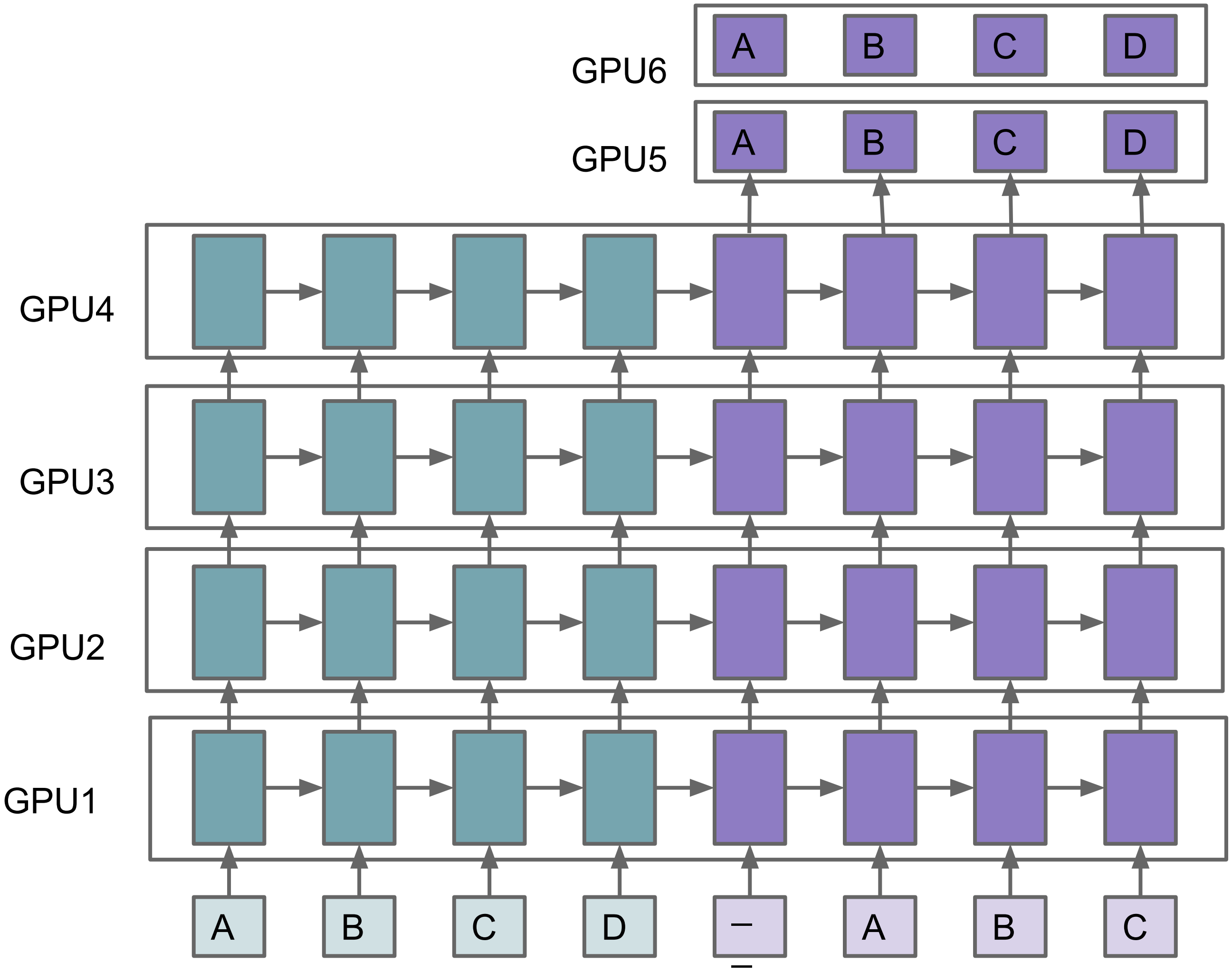


80k softmax by
1000 dims
This is very big!

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

Interesting Open Problems

ML:

unsupervised learning

reinforcement learning

highly multi-task and transfer learning

automatic learning of model structures

privacy preserving techniques in ML

...

Interesting Open Problems

Systems:

Use high level descriptions of ML computations and map these efficiently onto wide variety of different hardware

Integration of ML into more traditional data processing systems

Automated splitting of computations across mobile devices and datacenters

Use learning in lieu of traditional heuristics in systems

...

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
  - ...

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.org/abs/1603.02199

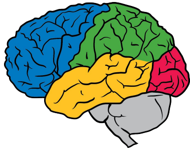
Conclusions

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

If you're not considering how to apply deep neural nets to your data, **you almost certainly should be**

TensorFlow makes it easy for everyone to experiment with these techniques

- Highly scalable design allows faster experiments, accelerates research
- Easy to share models and to publish code to give reproducible results
- Ability to go from research to production within same system



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)
research.google.com/people/jeff
research.google.com/pubs/BrainTeam.html

Questions?