Begin your Deep Learning Journey with Art

Josh Gordon
@random_forests

IS THIS A

CAT or DOG?

CAT DOG

OUTPUT LAYER

ACTIVATED NEURONS

INPUT LAYER

DEEP *neural* NETWORK

Deep Learning

Current state of the art in many tasks:

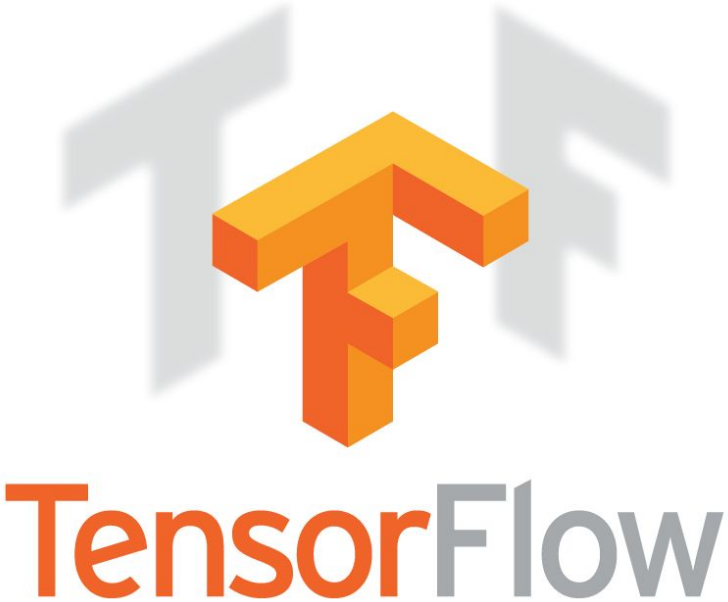
Image: *classification, captioning, colorization*

Text: *translation, parsing, summarization*

Speech: *recognition, generation*

Games: *AlphaGo, Atari, Mario Kart*

Many more.



- **Fast, flexible, and scalable open-source machine learning library**
- **For research and production**
- **Runs on CPU, GPU, Android, iOS, Raspberry PI, Datacenters**
- **Apache 2.0 license**

https://research.googleblog.com/2016/11/celebrating-tensorflows-first-year.html

Reproducible Research

Music https://magenta.tensorflow.org/2016/11/09/tuning-recurrent-networks-with-reinforcement-learning/

WaveNet https://deepmind.com/blog/wavenet-generative-model-raw-audio/

Translation https://research.googleblog.com/2016/09/a-neural-network-for-machine.html

Summarization https://research.googleblog.com/2016/08/text-summarization-with-tensorflow.html

Show and Tell https://research.googleblog.com/2016/09/show-and-tell-image-captioning-open.html

Inception https://research.googleblog.com/2016/08/improving-inception-and-image.html

Parsey McParseface https://research.googleblog.com/2016/05/announcing-syntaxnet-worlds-most.html

Many more: https://github.com/tensorflow/models/

Images

Inception

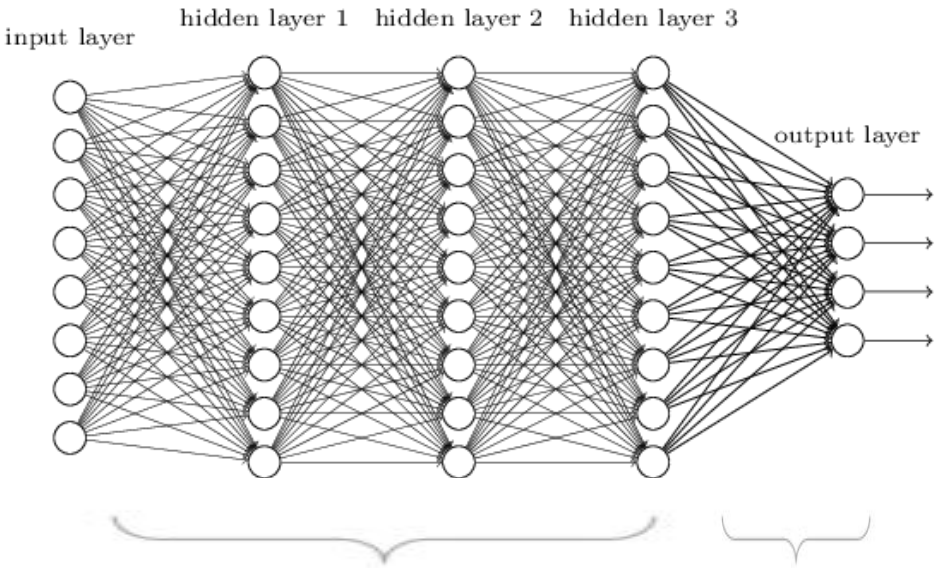
An Alaskan Malamute (left) and a Siberian Husky (right). Images from Wikipedia.

https://research.googleblog.com/2016/08/improving-inception-and-image.html

Create your own image classifier


goo.gl/xGsB9d
Monet - Sunflowers: 0.992
Claude Monet - Bouquet of Sunflowers
Images from the Metropolitan Museum of Art (with permission)
For more: @random_forests

Transfer Learning



Most of the network is unchanged

Only the last layer of weights and the outputs are updated

Image: neuralnetworksanddeeplearning.com

TensorFlow for Poets

Codelab goo.gl/xGsB9d

Video goo.gl/KewA03

Show and Tell

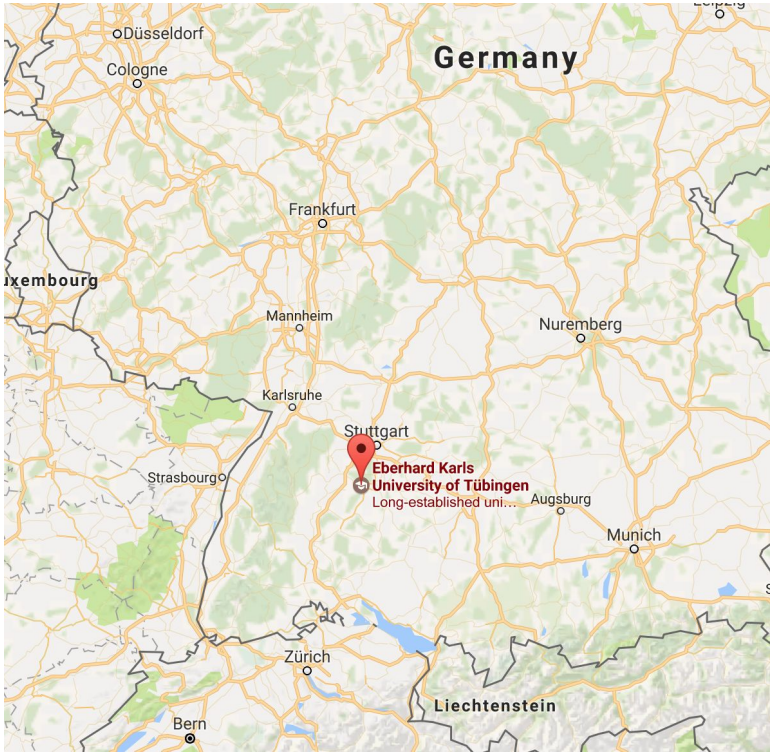
https://research.googleblog.com/2016/09/show-and-tell-image-captioning-open.html

Pretrained model - goo.gl/ExBjhS

A dog with a frisbee
in its mouth

Create your own artwork

A Neural Algorithm of Artistic Style



https://arxiv.org/pdf/1508.06576v2.pdf


Messe Karlsruhe
KASSE

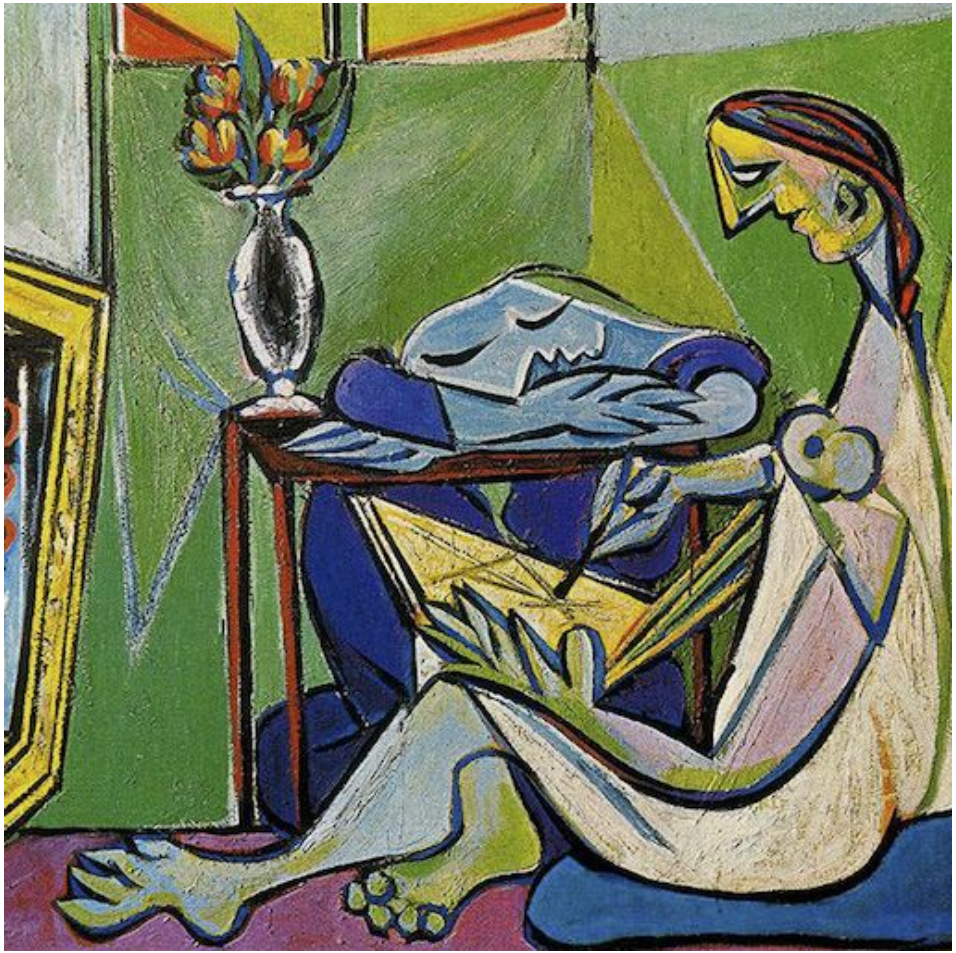
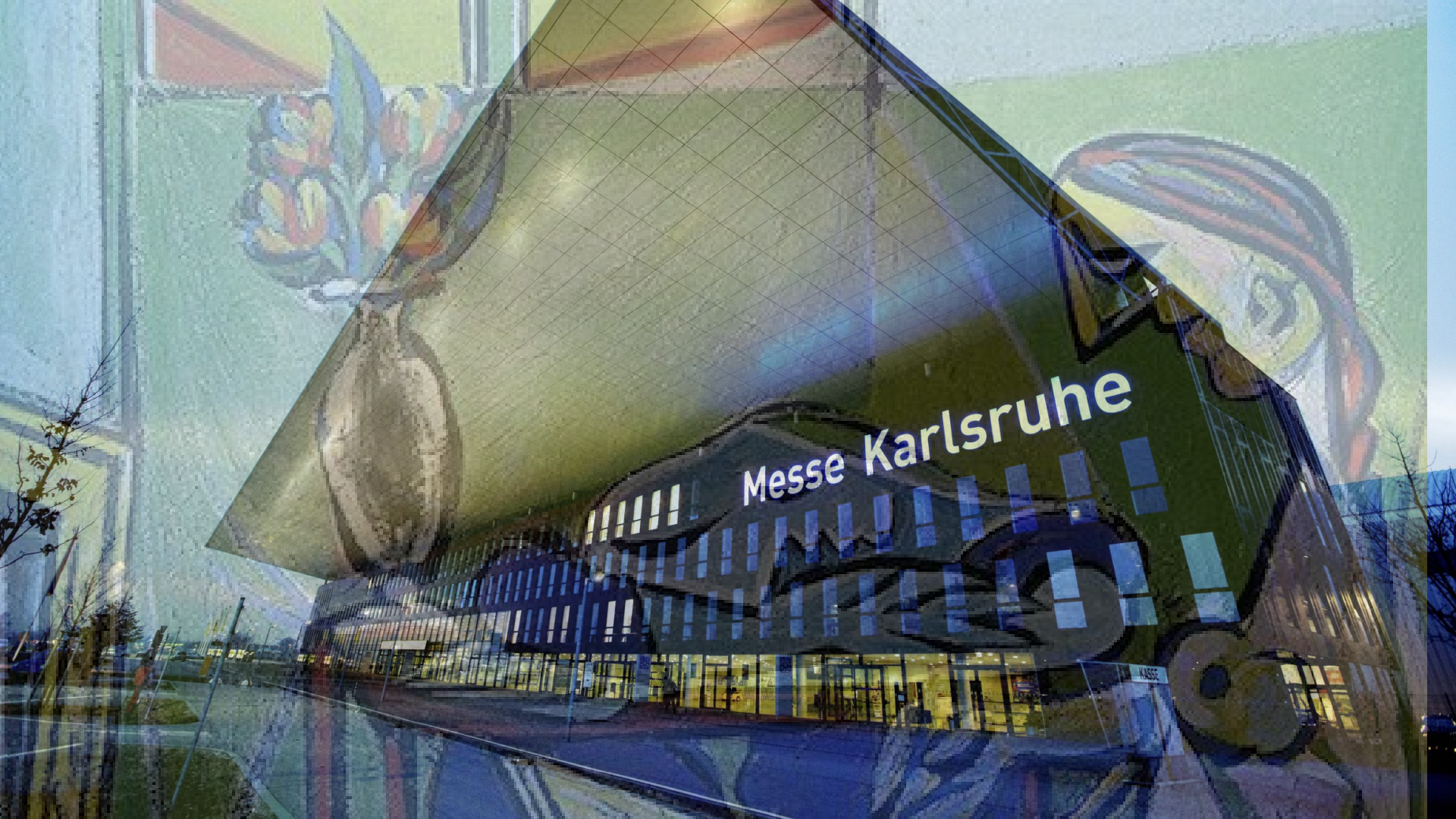
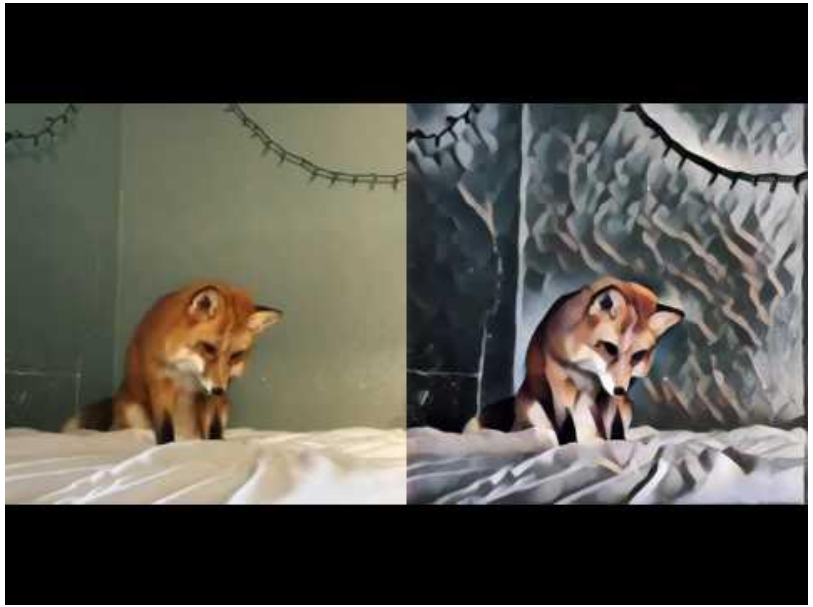


Image source - WikiArt


Messe Karlsruhe
KASSE

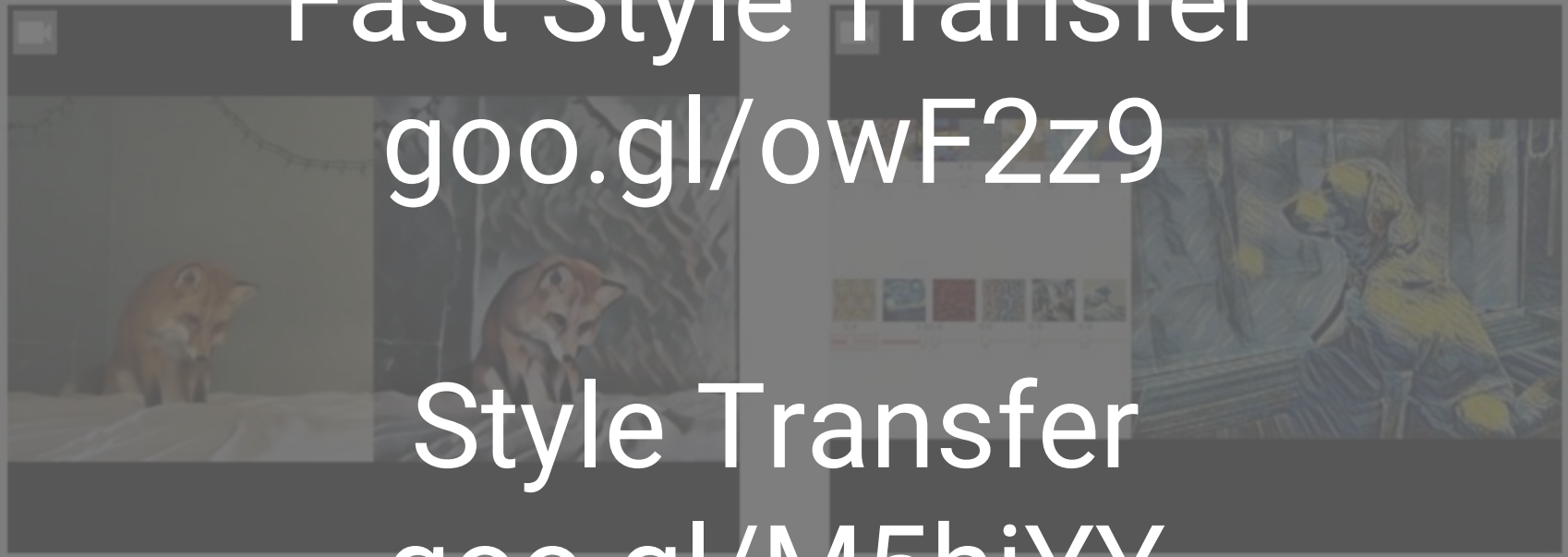

Messe Karlsruhe


Messe Karlsruhe



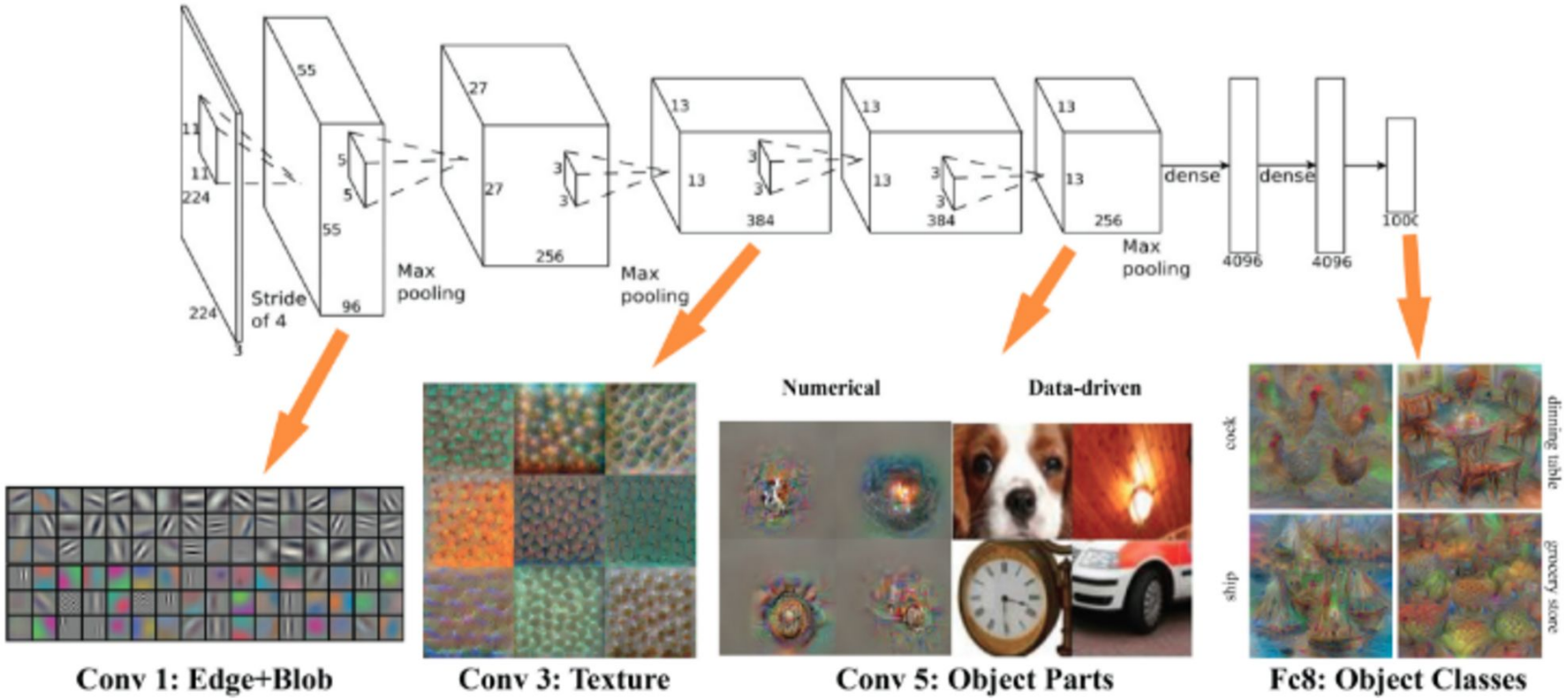
https://github.com/lengstrom/fast-style-transfer/
https://research.googleblog.com/2016/10/supercharging-style-transfer.html

Fast Style Transfer
goo.gl/owF2z9



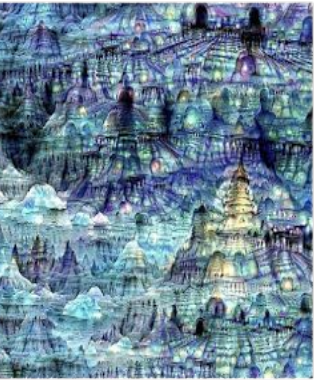
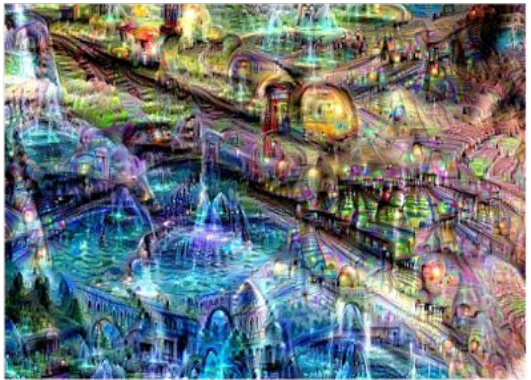
Style Transfer
goo.gl/M5hiYY

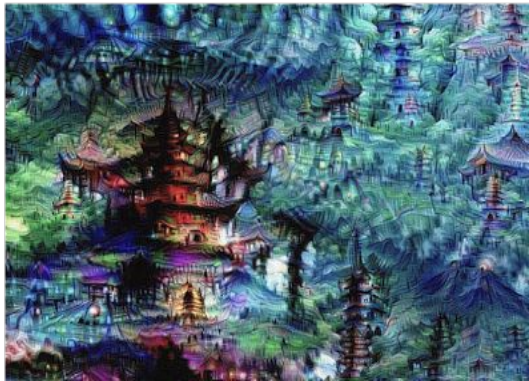
https://github.com/lengstrom/fast-style-transfer/
https://research.googleblog.com/2016/10/supercharging-style-transfer.html



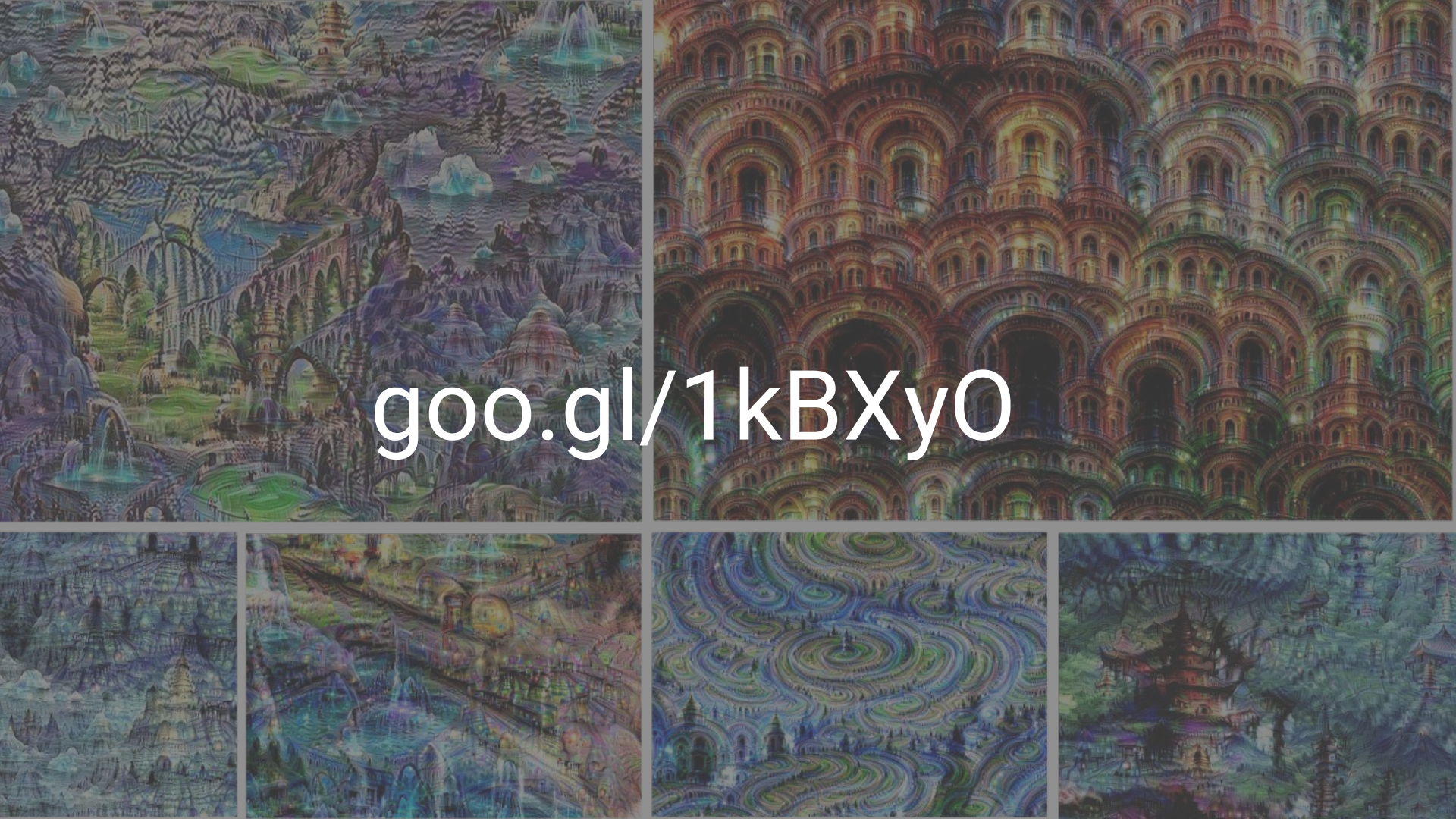
From: A Matlab Plugin to Visualize Neurons from Deep Models, Donglai Wei et. al.





https://www.youtube.com/watch?v=DgPaCWJL7XI
https://research.googleblog.com/2015/06/inceptionism-going-deeper-into-neural.html

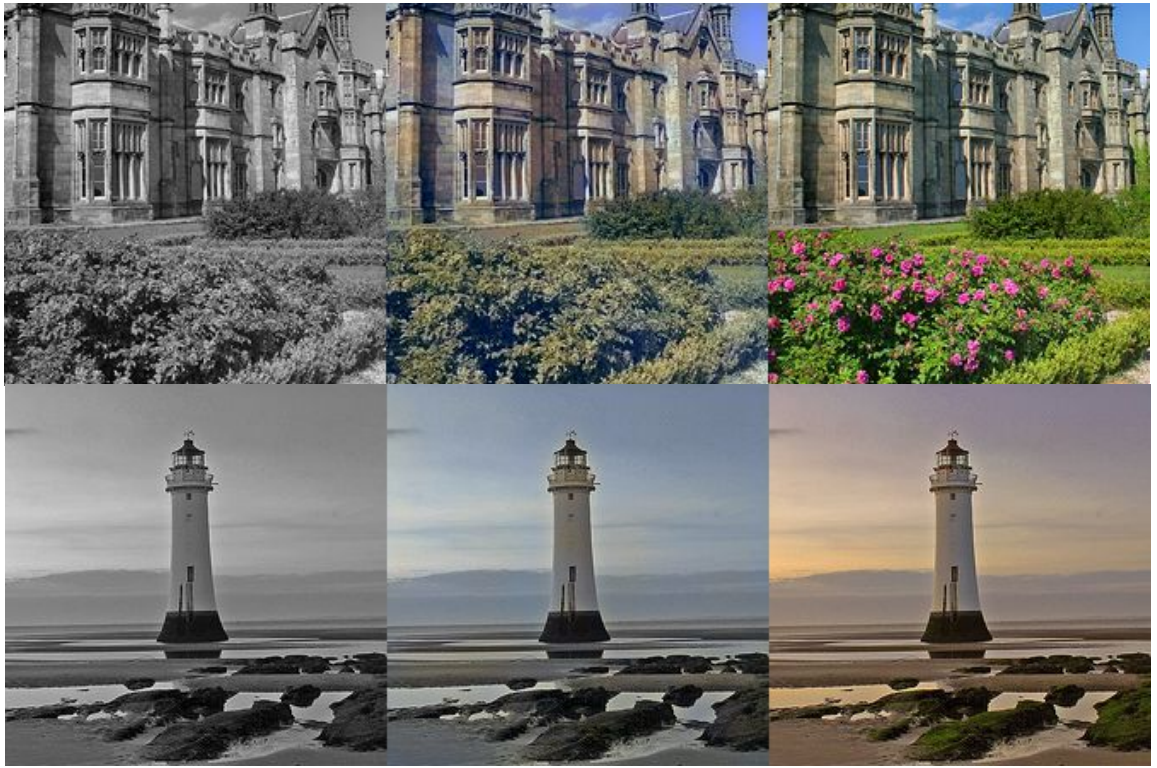

goo.gl/1kBXyO

ColorNet

Grayscale 灰度图

Prediction 预测

Ground Truth 地面实况



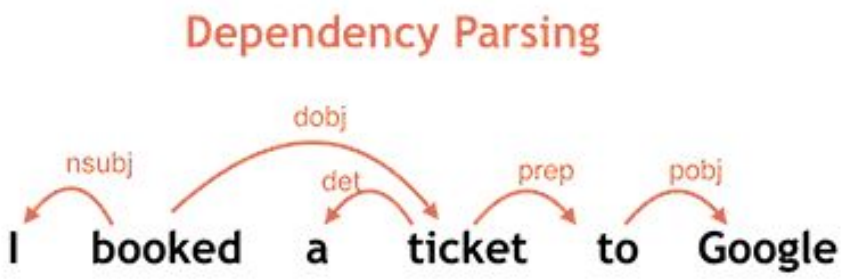
https://github.com/pavelgonchar/colornet

Image Super-Resolution

https://github.com/david-gpu/srez

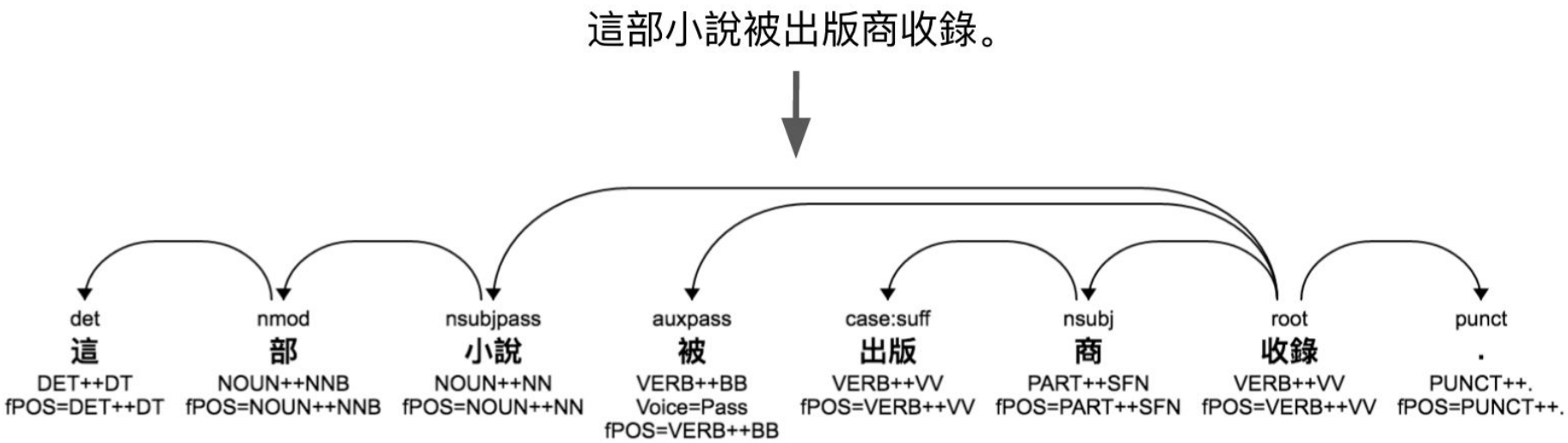
Text

Parsey McParseface



https://research.googleblog.com/2016/05/announcing-syntaxnet-worlds-most.html

Meet Parsey's Cousins: Syntax for 40 languages



https://research.googleblog.com/2016/08/meet-parseys-cousins-syntax-for-40.html

Summarization

Original text

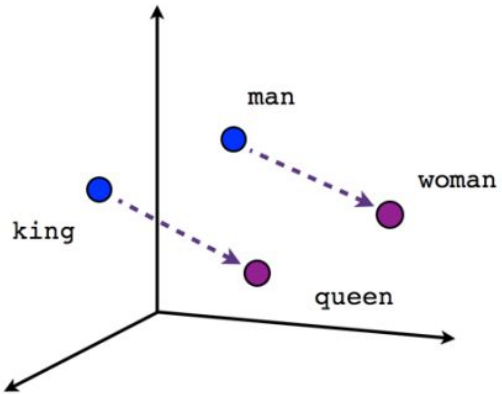
- *Alice and Bob took the train to visit the zoo. They saw a **baby giraffe, a lion, and a flock of colorful tropical birds**.*

Abstractive summary

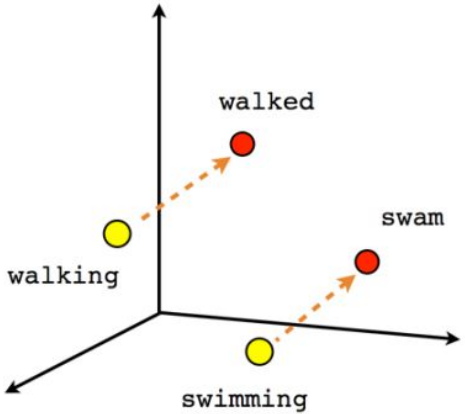
- *Alice and Bob visited the zoo and saw **animals and birds**.*

https://research.googleblog.com/2016/08/text-summarization-with-tensorflow.html

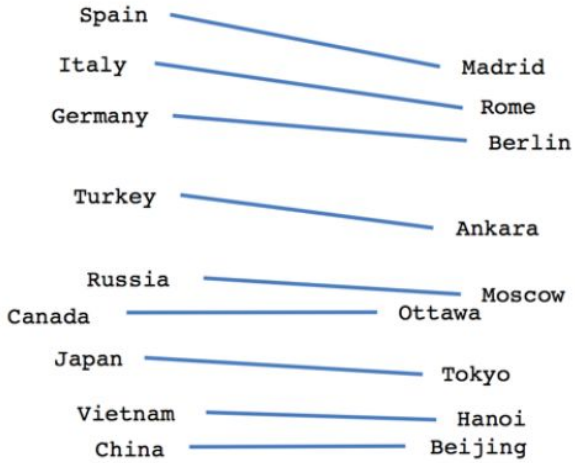
Word Embeddings



Male-Female



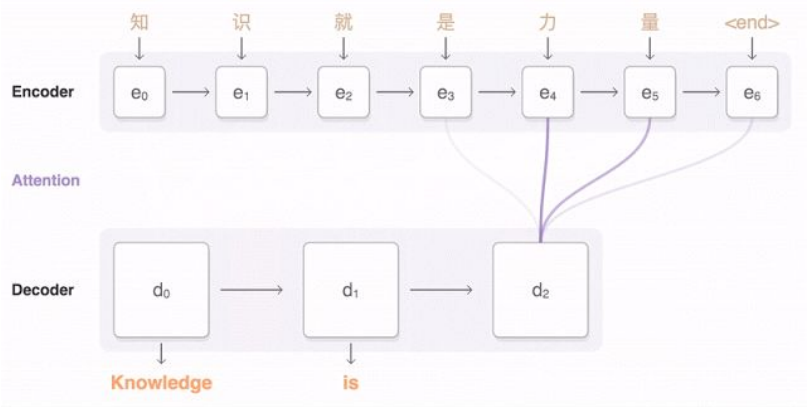
Verb tense



Country-Capital

https://www.tensorflow.org/versions/r0.12/tutorials/word2vec/index.html#vector-representations-of-words

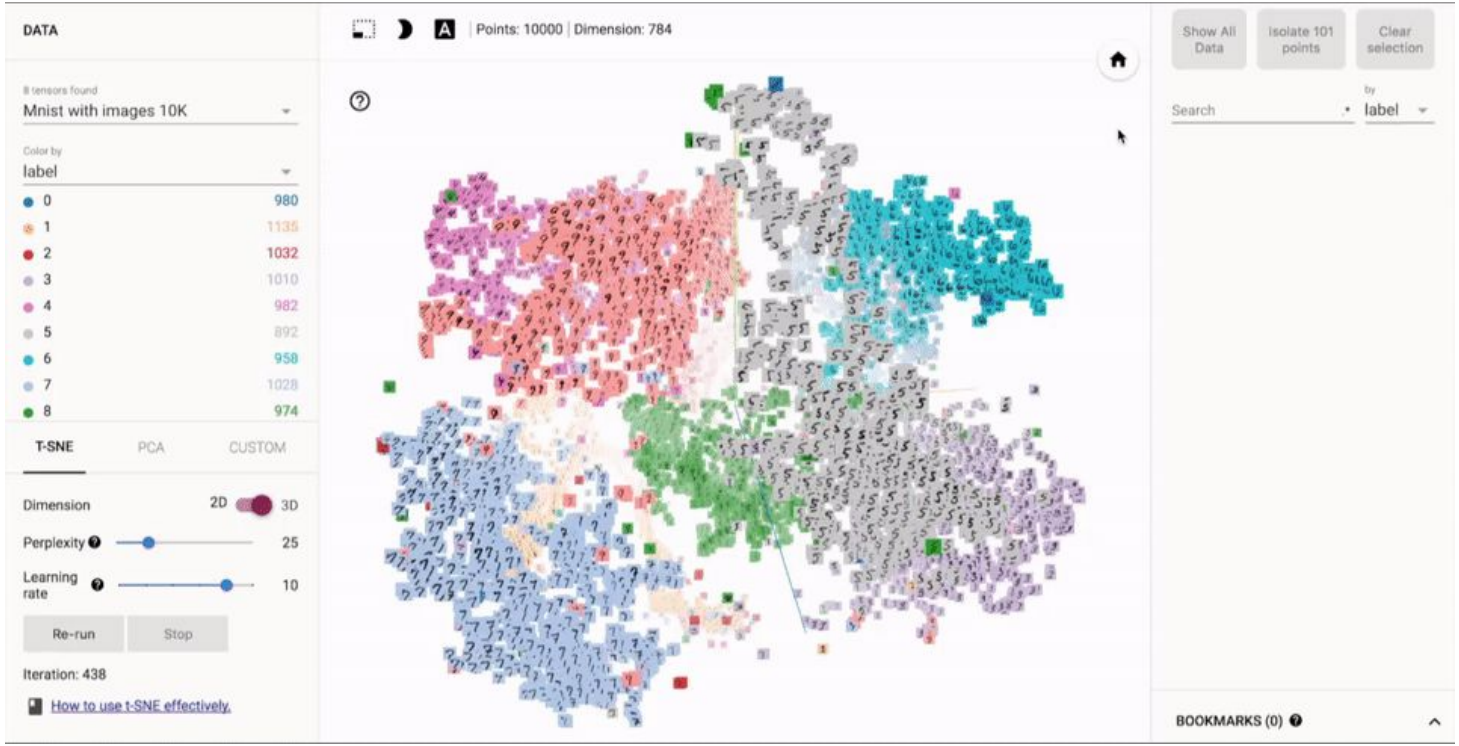
Translation



Chinese->English	100年前，预测引力波的爱因斯坦或许都无法想象人类可以直接观测到引力波。	100 years ago, the prediction of Einstein's gravitational waves probably can not imagine humans can directly observe gravitational waves.	100 years ago, Einstein predicted gravitational waves may not be able to imagine humans can directly observe the gravitational waves.	100 years ago, Einstein who predicted gravitational waves may not be able to imagine that humans can directly observe the gravitational waves.

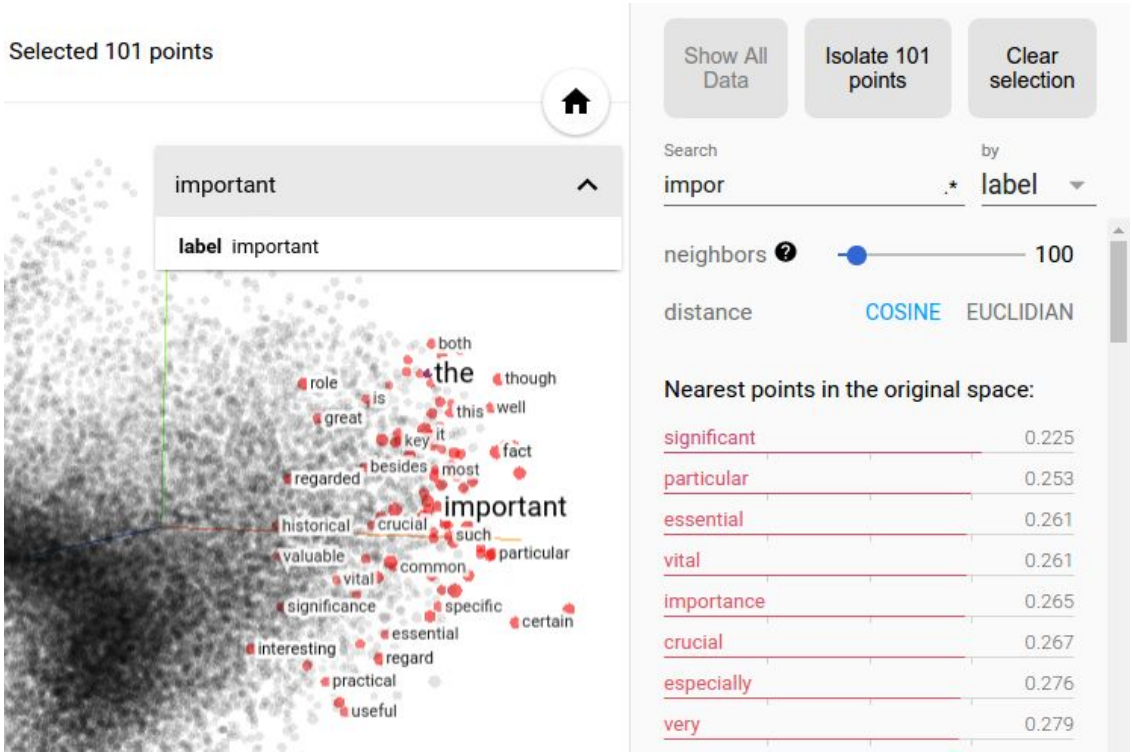
https://research.googleblog.com/2016/09/a-neural-network-for-machine.html

TensorBoard



https://www.tensorflow.org/versions/r0.12/how_tos/embedding_viz/index.html

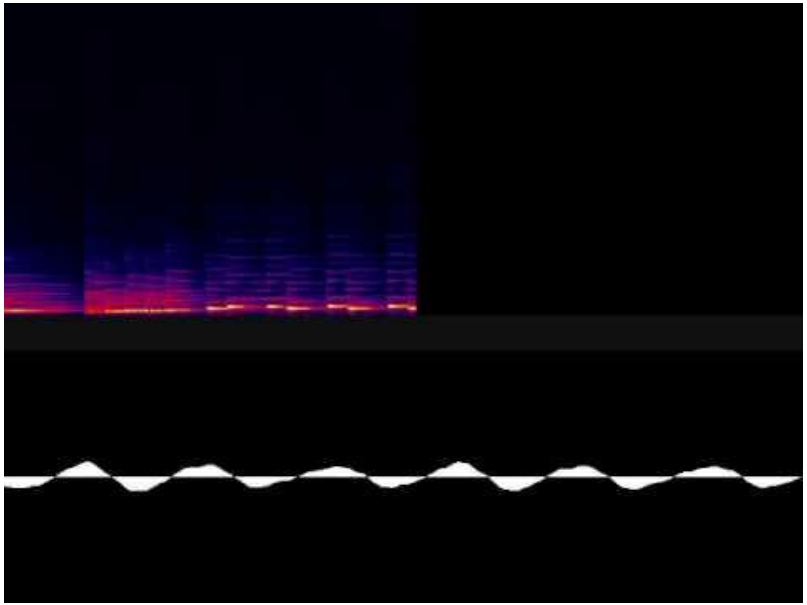
TensorBoard



https://www.tensorflow.org/versions/r0.12/how_tos/embedding_viz/index.html

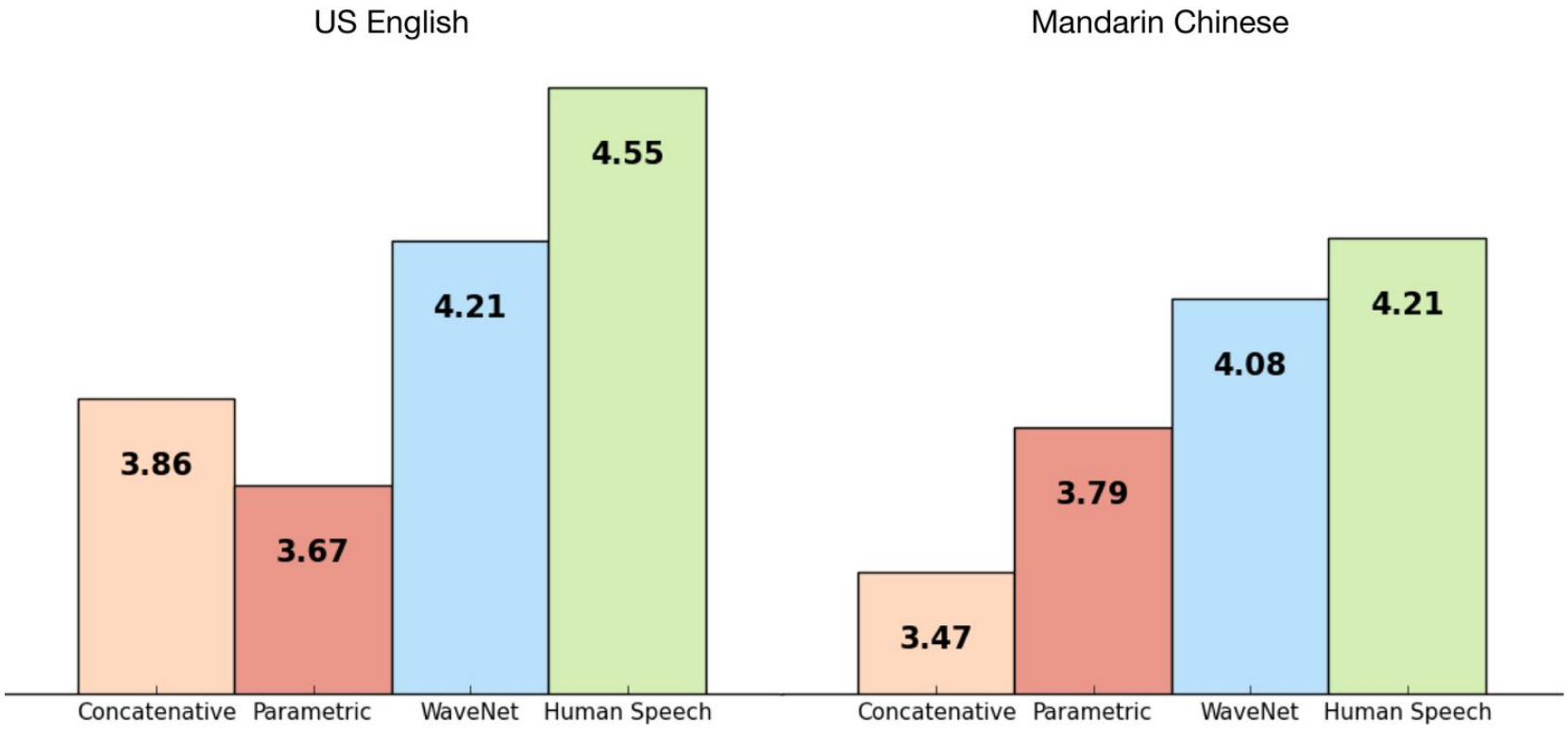
Sound

MAGENTA



https://magenta.tensorflow.org/2016/11/09/tuning-recurrent-networks-with-reinforcement-learning/

WaveNet



https://deepmind.com/blog/wavenet-generative-model-raw-audio/

Generating Speech and Music

English – Parametric 英文 – 参数化
English – Concatenative 英文- 拼接
English – WaveNet 英文 - WaveNet
Mandarin – Parametric 汉语 - 参数化
Mandarin – Concatenative 汉语 – 拼接
Mandarin – WaveNet 汉语 - WaveNet
Bonus - on hold 福利时间 – 听一段铃声

https://deepmind.com/blog/wavenet-generative-model-raw-audio/

https://magenta.tensorflow.org/2016/11/09/tuning-recurrent-networks-with-reinforcement-learning/

From the community

A TensorFlow implementation of DeepMind's WaveNet paper
https://github.com/ibab/tensorflow-wavenet

Speech-to-Text-WaveNet : End-to-end sentence level English speech recognition based on DeepMind's WaveNet and TensorFlow
https://github.com/buriburisuri/speech-to-text-wavenet

Games

AlphaGo

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

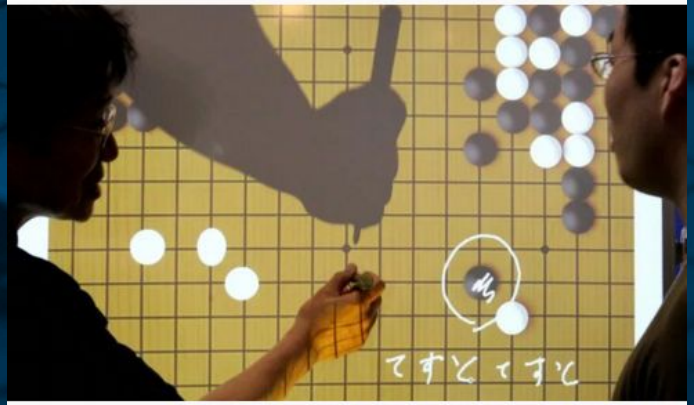
How did they do it?

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



Going up. (Reuters/Kiyoshi Ota)

Computers have slowly started to encroach on activities we previously believed only the brilliantly sophisticated human brain could handle. IBM's Deep Blue supercomputer 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.

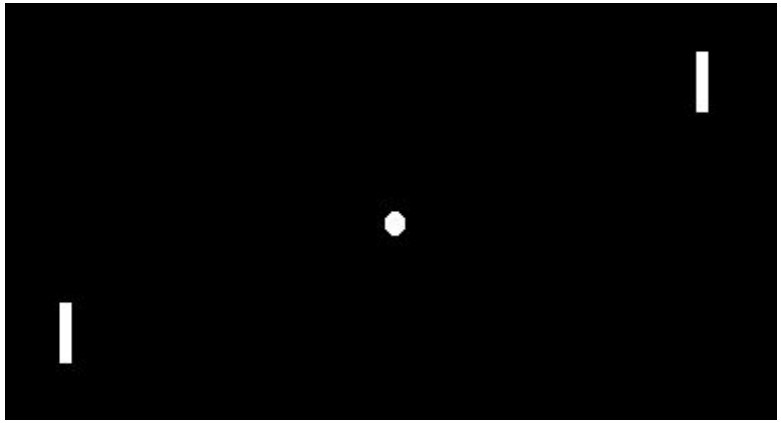


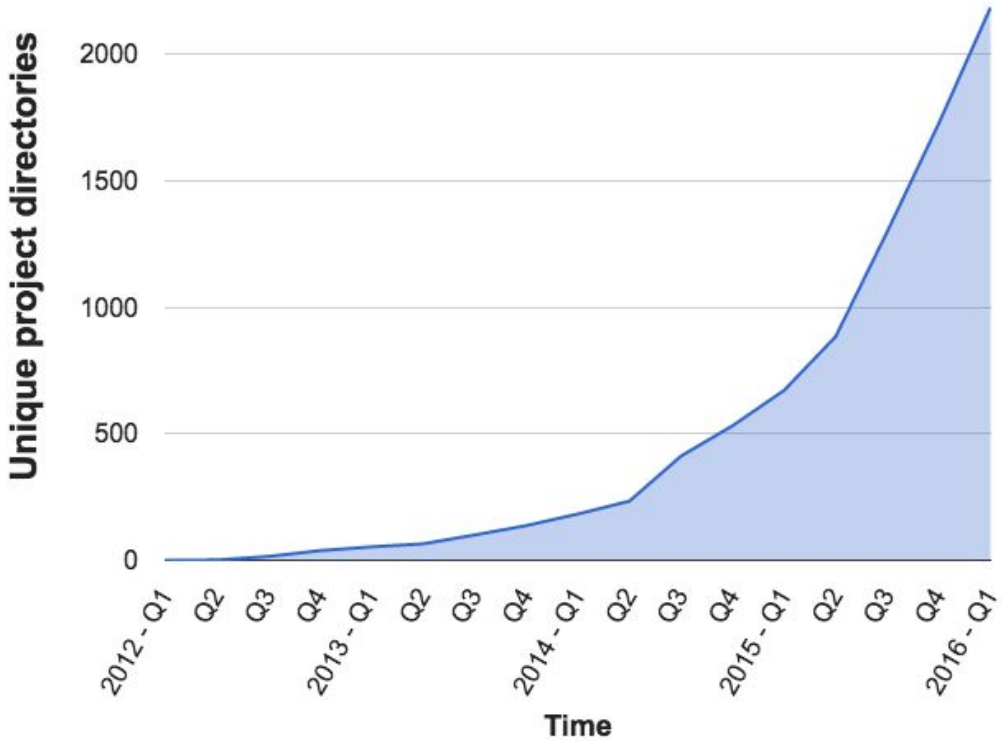

image.display
000 5 1

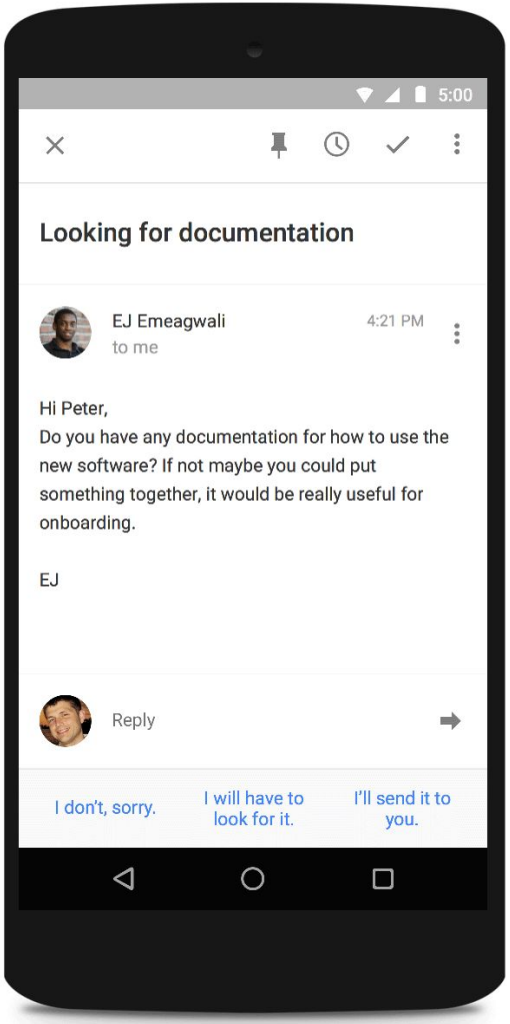
http://kevinhughes.ca/blog/tensor-kart

Deep Learning at Google

Growing Use of Deep Learning at Google

of directories containing model description files





Smart reply in Inbox by Gmail

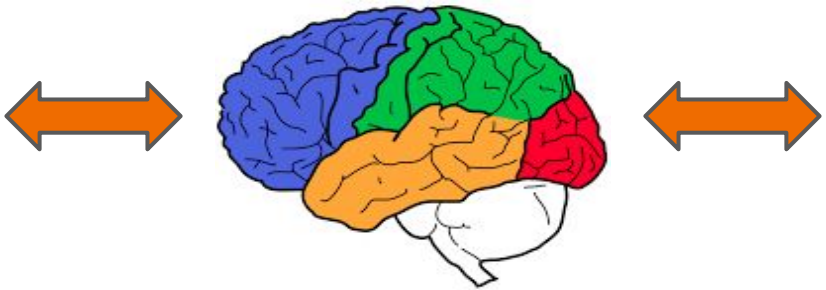
10%

of all responses sent on mobile

In conclusion

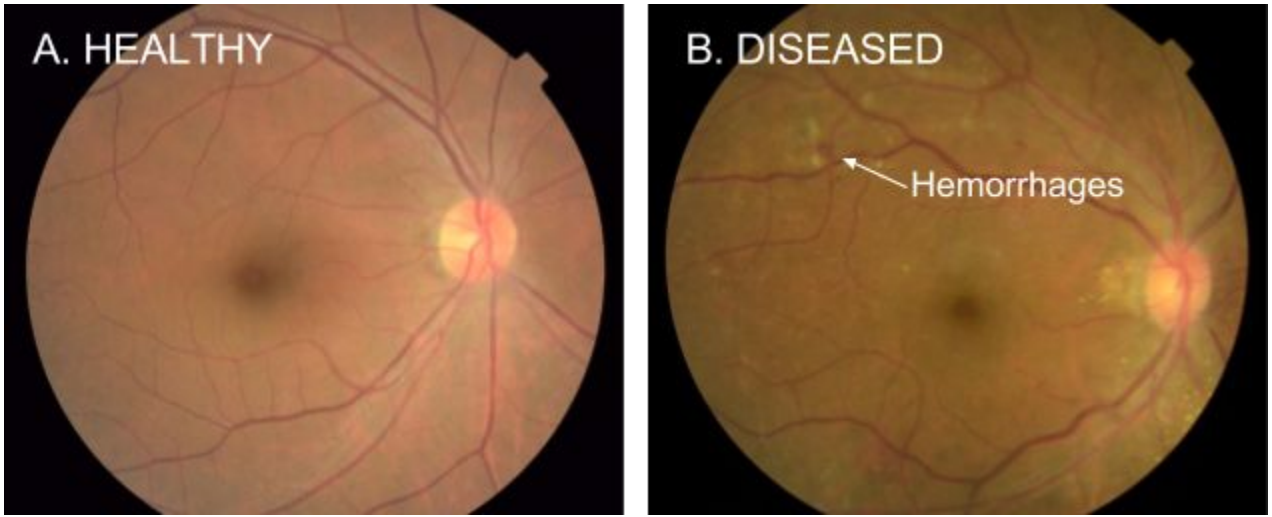
“Universal” Machine Learning

Speech
Text
Search
Queries
Images
Videos
Labels
Entities
Words
Audio
Features



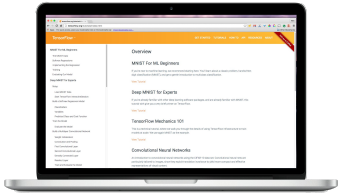
Speech
Text
Search
Queries
Images
Videos
Labels
Entities
Words
Audio
Features

Detection of Diabetic Eye Disease



https://research.googleblog.com/2016/11/deep-learning-for-detection-of-diabetic.html

Next steps



Tutorials and code
tensorflow.org

Totally new to ML?
Recipes goo.gl/KewA03

Intro to Deep Learning with TensorFlow
Udacity class goo.gl/iHssII

Stanford's CS231n
cs231n.github.io

Udacity's Machine Learning Nanodegree
goo.gl/ODpXj4

These slides + exercises
goo.gl/nrdsxM

TensorFlow Playground
goo.gl/mXhncM

Chris Olah's blog
colah.github.io

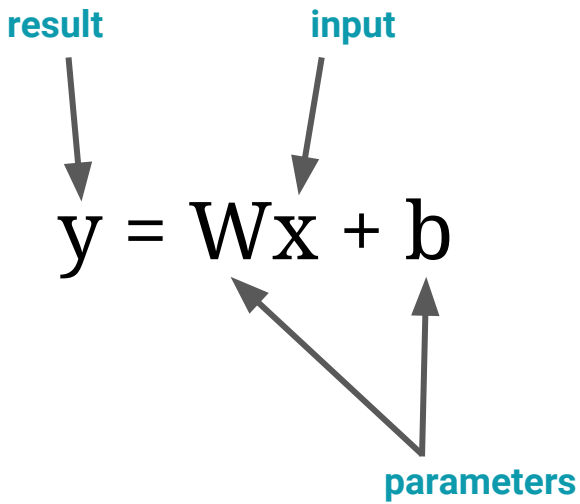
Michael Nielsen's book
neuralnetworksanddeeplearning.com

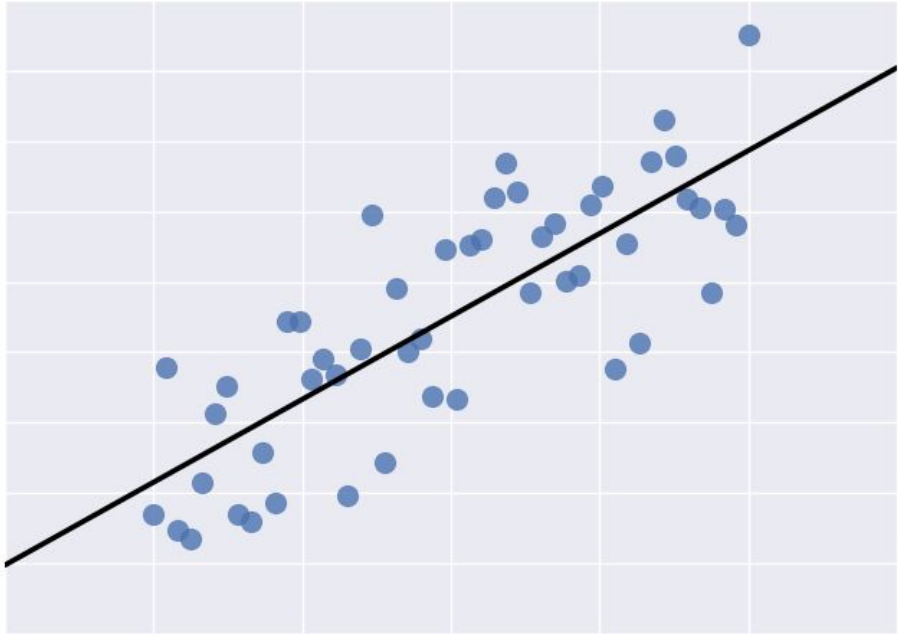
Thank you and have fun!

Josh Gordon
@random_forests

Extras

Linear Regression





Let's code this up

1. **Inference** (“given x, this is the code to predicts y”)

2. **Loss** (“quantify how bad our prediction was”)

3. **Optimize** (“update variables to improve the prediction”)

Inference: putting it all together

```
import tensorflow as tf
x = tf.placeholder(shape=[None],
                   dtype=tf.float32,
                   name='x')
W = tf.Variable(tf.random_normal([1], name="W"))
b = tf.Variable(tf.random_normal([1], name="b"))
y = W * x + b

with tf.Session() as sess:

  sess.run(tf.initialize_all_variables())

  print(sess.run(y, feed_dict={x: x_in}))
```

Build the graph

Prepare execution env

Initialize variables

Run the computation

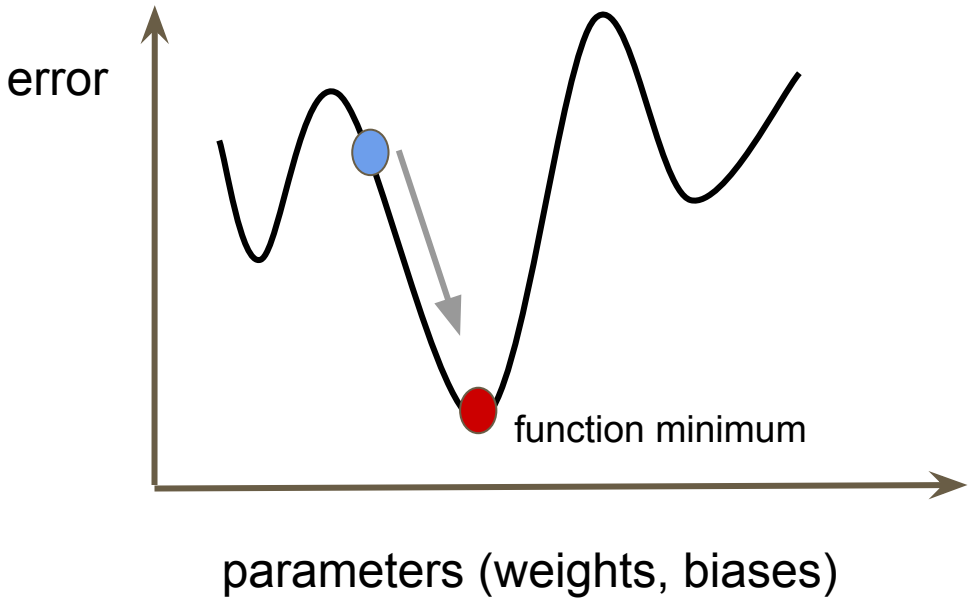
Define a Loss

Given y, y_train compute a loss, for instance:

$$loss = (y - y_{train})^2$$

```
# create an operation that calculates loss
loss = tf.reduce_mean(tf.square(y - y_train))
```

Minimize loss using an optimizer



tf.train.GradientDescentOptimizer

Or, use one of:

AdadeltaOptimizer

AdagradOptimizer

AdagradDAOptimizer

AdamOptimizer

...

Automatic Differentiation

Feed (x, y_{label}) pairs and adjust W and b to decrease the loss.

$$W \leftarrow W - \eta \left(dL/dW \right)$$

$$b \leftarrow b - \eta \left(dL/db \right)$$

TensorFlow computes gradients automatically

```
# Create an optimizer

optimizer = tf.train.GradientDescentOptimizer(0.5)

# Create an operation that minimizes loss.

train = optimizer.minimize(loss)
```

Learning rate

Putting it all together

```
loss = tf.reduce_mean(tf.square(y - y_train))
optimizer = tf.train.GradientDescentOptimizer(0.5)
train = optimizer.minimize(loss)
with tf.Session() as sess:
  sess.run(tf.initialize_all_variables())
  for i in range(1000):
    sess.run(train, feed_dict={x: x_data[i],
                               y_label: y_data[i]})
```

Define a loss

Create an optimizer

Op to minimize the loss

Initialize variables

Iteratively run the training op

Questions?

Progress last year

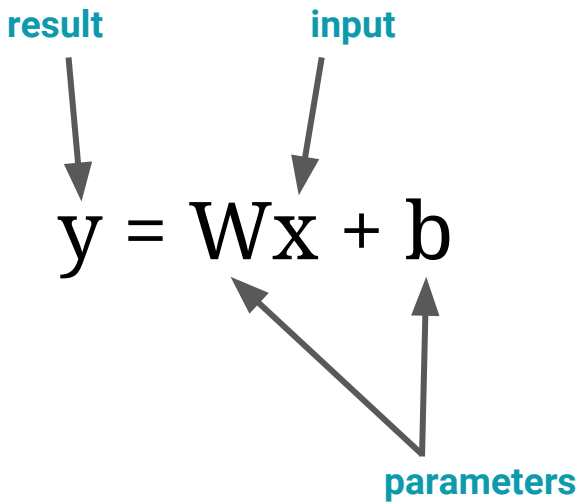
Release	Milestone
November 2015	Initial release
December (0.6)	Faster on GPUs; Python 3.3+
February (0.7)	TensorFlow Serving
April (0.8)	Distributed TensorFlow
June (0.9)	iOS; Mac GPU
August (0.10)	Slim
October (0.11)	HDFS; CUDA 8, CuDNN 5
November (0.12)	Windows 7, 10, and Server 2016

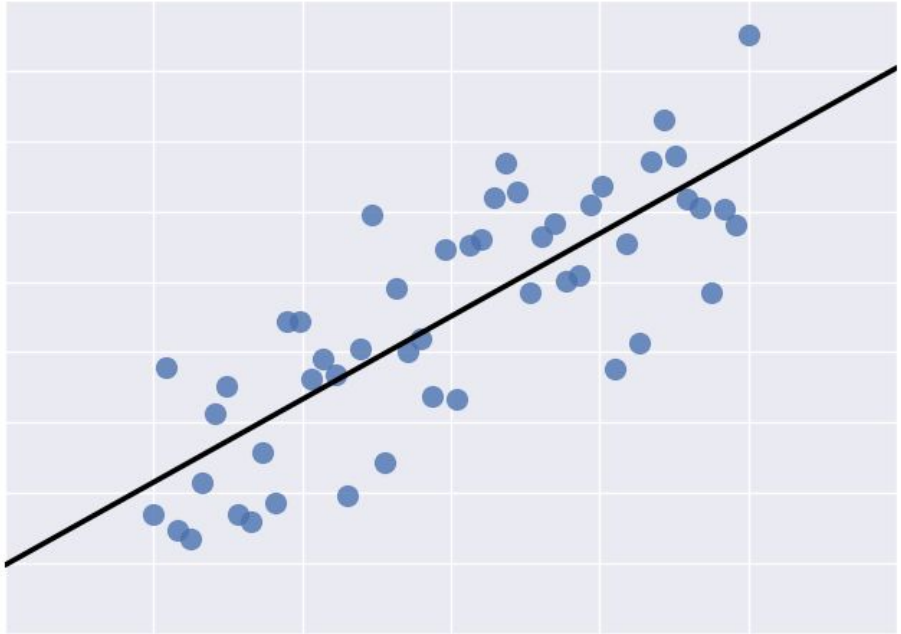
https://github.com/tensorflow/tensorflow/releases

Linear Regression

Inference, Loss, Optimize

Linear Regression





What are we trying to do?

Mystery equation: `y = 0.1 * x + 0.3 + noise`

Model: `y = W * x + b`

Objective: Given enough (*x*, *y*) samples, figure out the value of *W* and *b*.

Let's code this up

1. **Inference** (“given x, this is the code to predicts y”)

2. **Loss** (“quantify how bad our prediction was”)

3. **Optimize** (“update variables to improve the prediction”)

y = Wx + b in TensorFlow

```
import tensorflow as tf
```

y = Wx + b in TensorFlow

```
import tensorflow as tf

x = tf.placeholder(shape=[None],
                   dtype=tf.float32, name="x")
```

y = Wx + b in TensorFlow

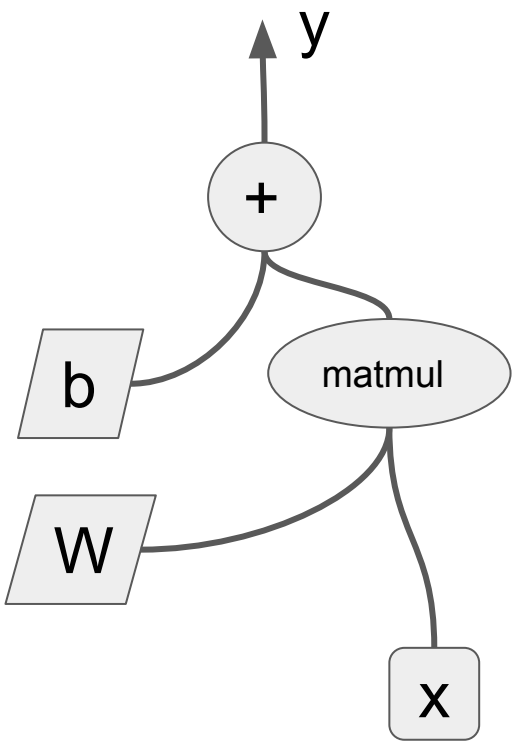
```
import tensorflow as tf

x = tf.placeholder(shape=[None],
                   dtype=tf.float32, name="x")

W = tf.Variable(tf.random_normal([1], name="W")
```

y = Wx + b in TensorFlow

```
import tensorflow as tf

x = tf.placeholder(shape=[None],
                   dtype=tf.float32, name="x")

W = tf.Variable(tf.random_normal([1], name="W")

b = tf.Variable(tf.random_normal([1], name="b")
```

y = Wx + b in TensorFlow

```
import tensorflow as tf

x = tf.placeholder(shape=[None],
                   dtype=tf.float32, name="x")

W = tf.Variable(tf.random_normal([1], name="W")

b = tf.Variable(tf.random_normal([1], name="b")

y = W * x + b
```



Variables Must be Initialized

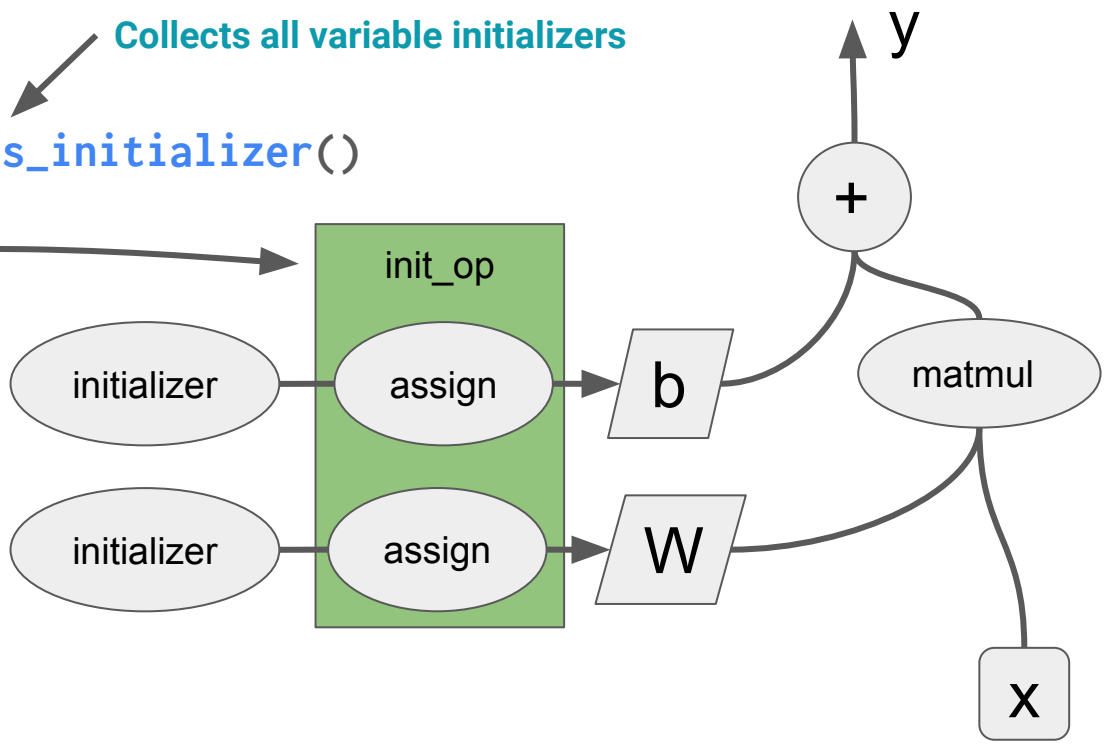
Collects all variable initializers

```
init_op = tf.global_variables_initializer()
```

Makes an execution environment

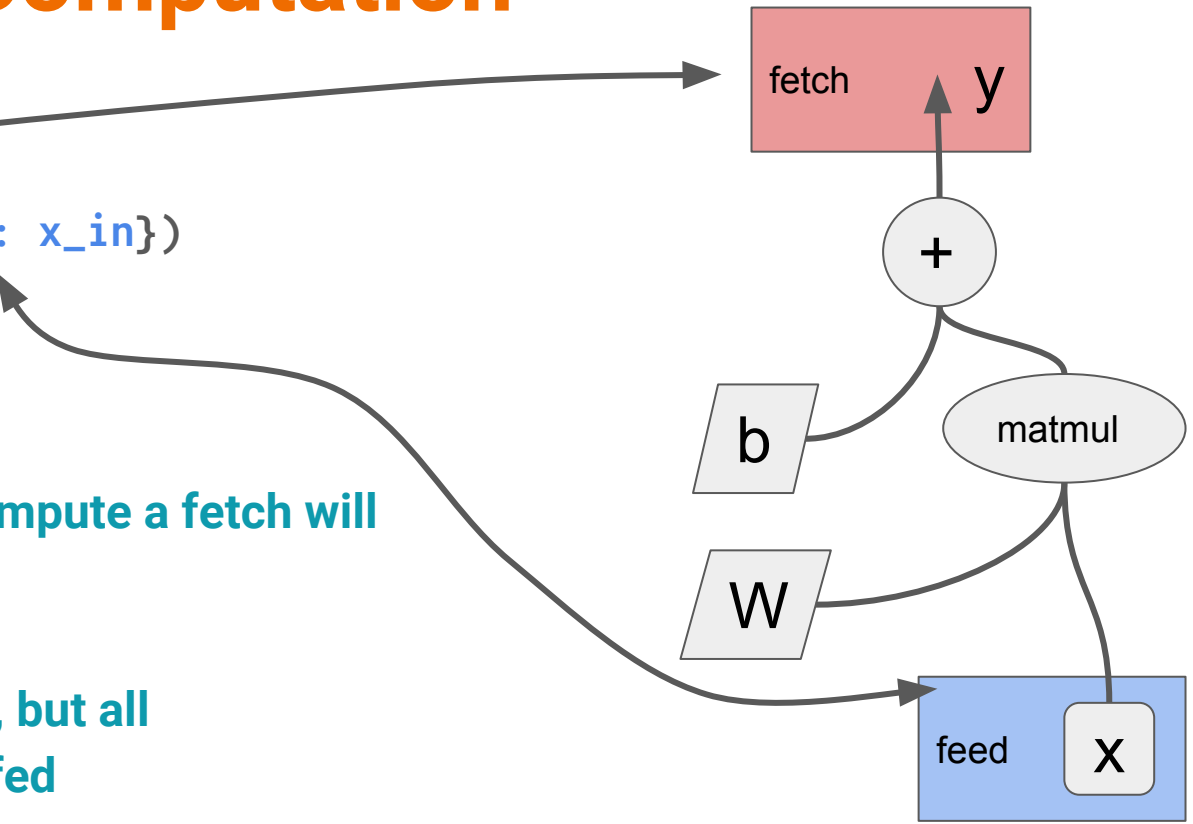
```
sess = tf.Session()

sess.run(init_op)
```

Actually initialize the variables



Running the Computation

```
x_in = 3

sess.run(y, feed_dict={x: x_in})
```



- Only what's used to compute a fetch will be evaluated
- All Tensors can be fed, but all placeholders must be fed

Inference: putting it all together

```
import tensorflow as tf
x = tf.placeholder(shape=[None],
                   dtype=tf.float32,
                   name='x')
W = tf.Variable(tf.random_normal([1], name="W"))
b = tf.Variable(tf.random_normal([1], name="b"))
y = W * x + b

with tf.Session() as sess:

  sess.run(tf.initialize_all_variables())

  print(sess.run(y, feed_dict={x: x_in}))
```

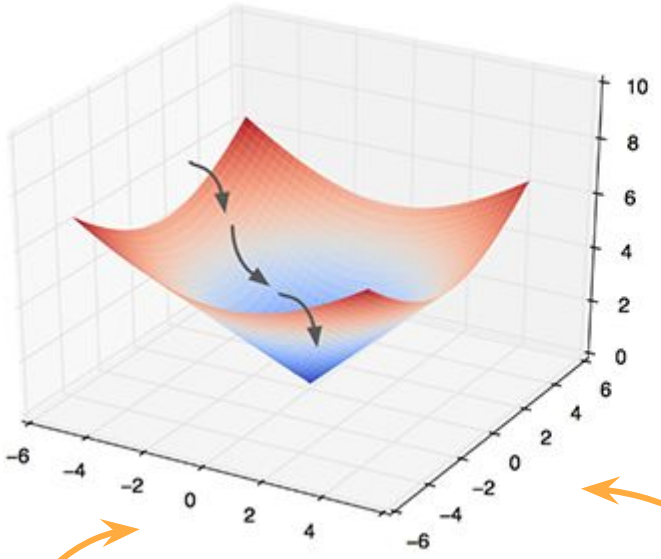
Build the graph

Prepare execution env

Initialize variables

Run the computation

Concepts: Loss and Gradient Descent



loss

parameter #2

parameter #1

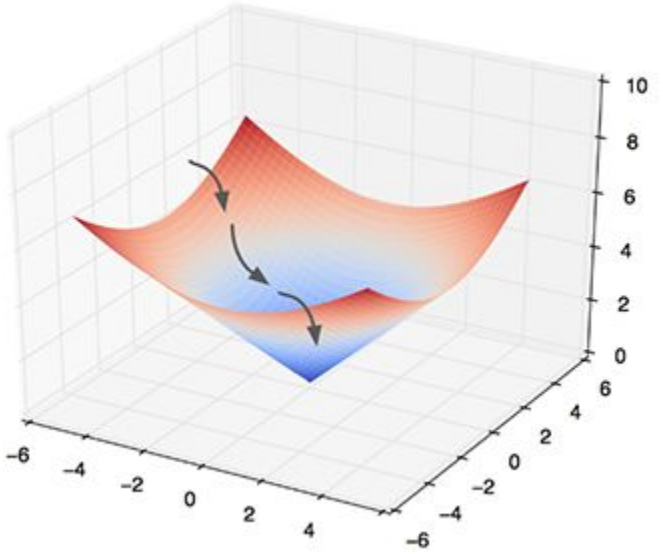
Define a Loss

Given y, y_train compute a loss, for instance:

$$loss = (y - y_{train})^2$$

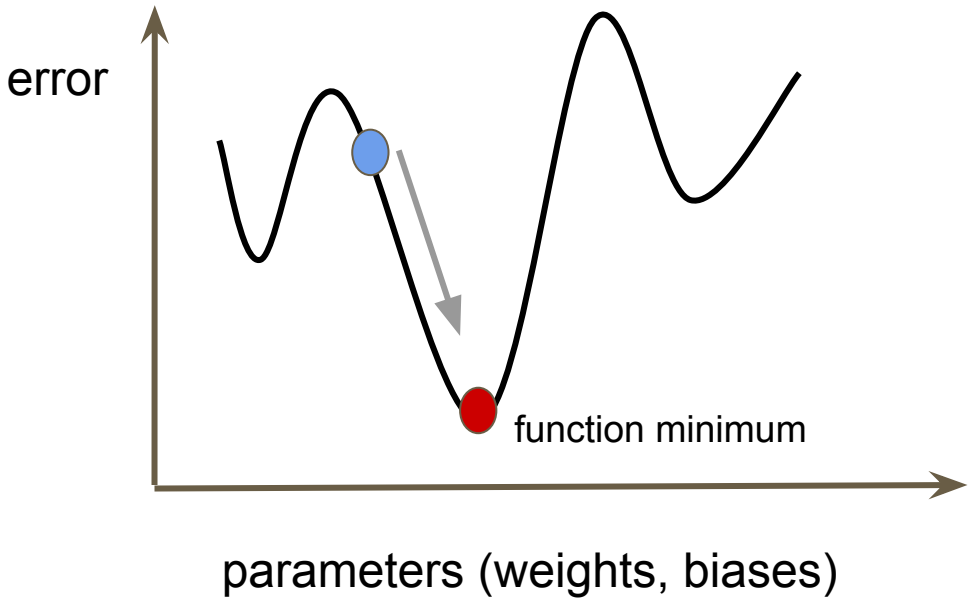
```
# create an operation that calculates loss
loss = tf.reduce_mean(tf.square(y - y_train))
```

Optimization



	-5	-4	-3	-2	-1	0	1	2	3	4	5
5	**50**	41	34	29	26	25	26	29	34	41	50
4	41	**32**	**25**	20	17	16	17	20	25	32	41
3	34	25	18	**13**	10	9	10	13	18	25	34
2	29	20	13	**8**	5	4	5	8	13	20	29
1	26	17	10	**5**	**2**	**1**	2	5	10	17	26
0	25	16	9		1	**0**	1	4	9	16	25
-1	26	17	10	5	2	1	2	5	10	17	26
-2	29	20	13	8	5	4	5	8	13	20	29
-3	34	25	18	13	10	9	10	13	18	25	34
-4	41	32	25	20	17	16	17	20	25	32	41
-5	50	41	34	29	26	25	26	29	34	41	50

Minimize loss using an optimizer



tf.train.GradientDescentOptimizer

Or, use one of:

AdadeltaOptimizer

AdagradOptimizer

AdagradDAOptimizer

AdamOptimizer

...

Automatic Differentiation

Feed (x, y_{label}) pairs and adjust W and b to decrease the loss.

$$W \leftarrow W - \eta \left(dL/dW \right)$$

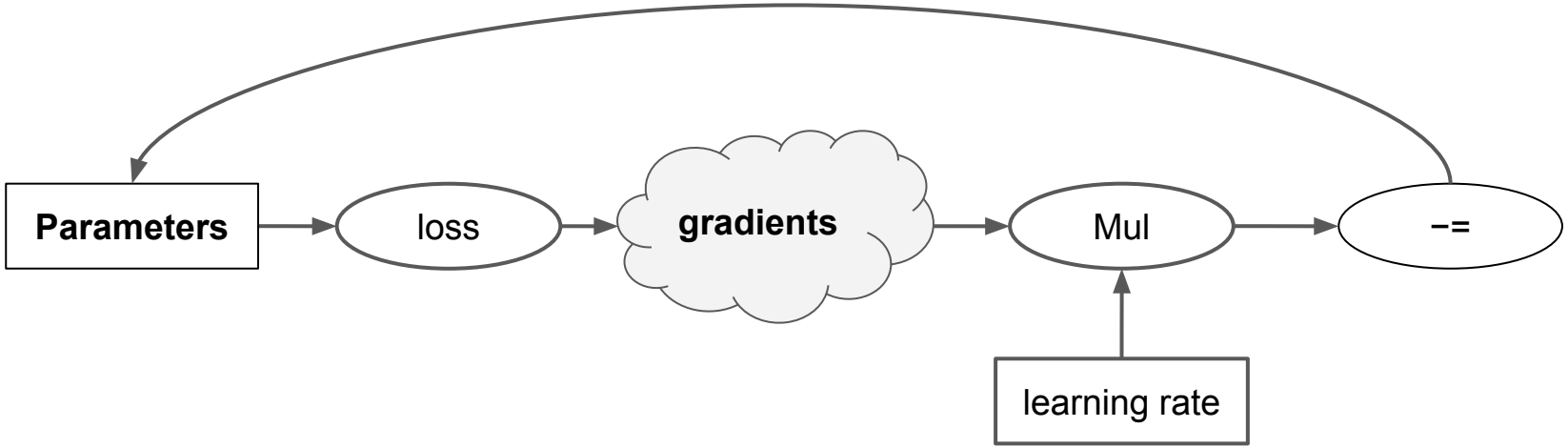
$$b \leftarrow b - \eta \left(dL/db \right)$$

TensorFlow computes gradients automatically

```
# Create an optimizer

optimizer = tf.train.GradientDescentOptimizer(0.5)

# Create an operation that minimizes loss.

train = optimizer.minimize(loss)
```

Learning rate

Behind the scenes



Putting it all together

```
loss = tf.reduce_mean(tf.square(y - y_train))
optimizer = tf.train.GradientDescentOptimizer(0.5)
train = optimizer.minimize(loss)
with tf.Session() as sess:
  sess.run(tf.initialize_all_variables())
  for i in range(1000):
    sess.run(train, feed_dict={x: x_data[i],
                               y_label: y_data[i]})
```

Define a loss

Create an optimizer

Op to minimize the loss

Initialize variables

Iteratively run the training op

Cumulative GitHub Stars