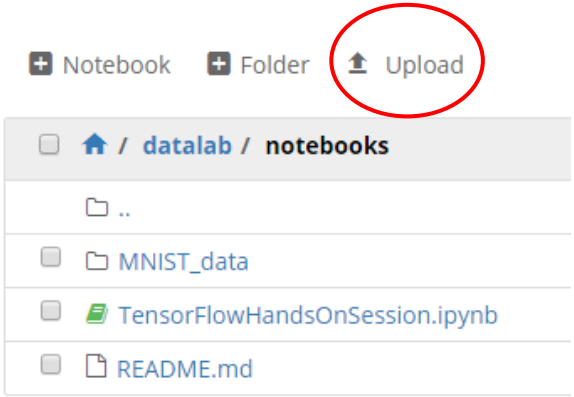
How to get started

- Step 1: Create a Google Cloud Platform Account linked with your visa (12 month free trial)
- Step 2: Follow this tutorial to set up Cloud Datalab
- Step 3: Download the IPython notebooks containing the assignments and the solution code
- Step 4: Upload the notebook with the code in Datalab



** The slides of the presentation will appear here after the session*

ML6

Hands-On Introduction to TensorFlow

Frederic Stallaert - Data Scientist @ML6 - Contact me! @ **frederic.stallaert@ml6.eu**
Nicolas DeRuytter - CEO @ML6
Erwin Huizenga - Machine Learning Specialist @Google

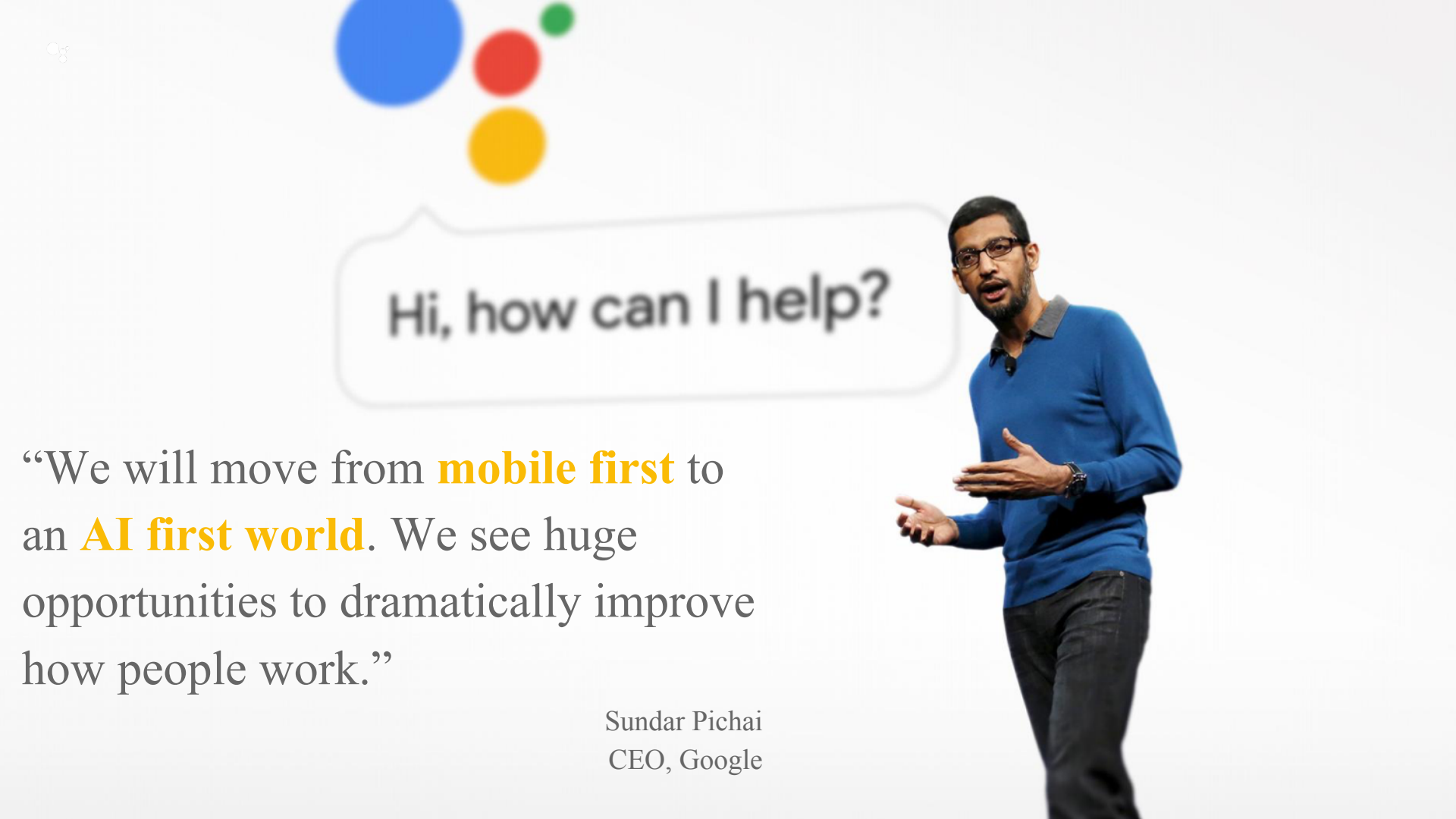
bit.ly/2yg1nD0

Google Cloud

Agenda

1. Machine Learning on Google Cloud Platform
2. Introduction to TensorFlow
3. tf.in.practice

Google Cloud



"We will move from **mobile first** to an **AI first world**. We see huge opportunities to dramatically improve how people work."

Sundar Pichai
CEO, Google

Vision Explorer Demo

At Google, we've been applying Machine Learning for a long time

Search
Search ranking
Speech recognition

Android
Keyboard & speech input

Play
App recommendations
Game developer experience

Gmail
Smart Reply
Spam classification

Drive
Suggested visualization and insights

Chrome
Search by Image

Photos
Photos search

YouTube
Video recommendations
Better thumbnails

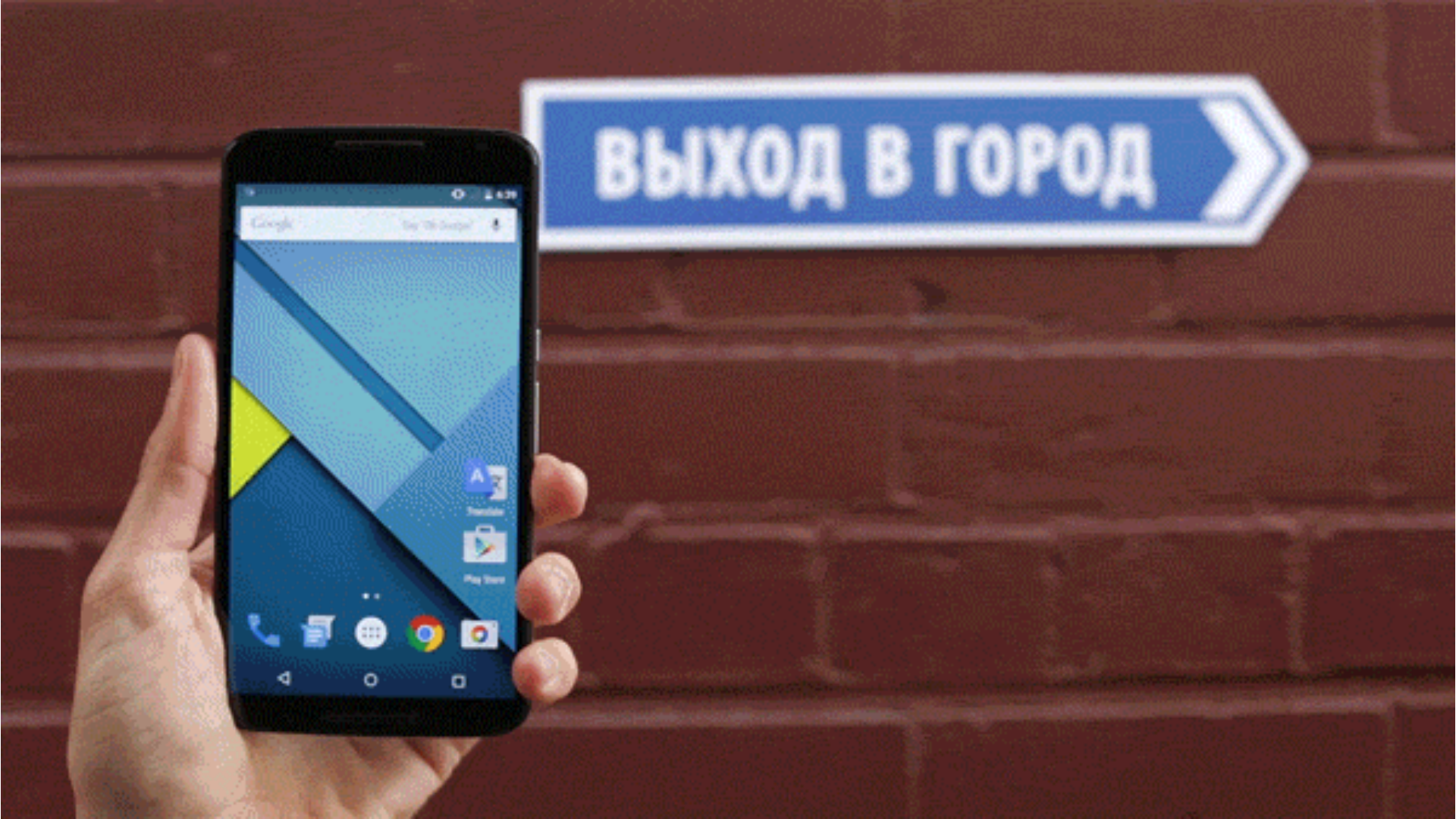
Maps
Street View image
Parsing Local Search

Translate
Text, graphic, and speech translations

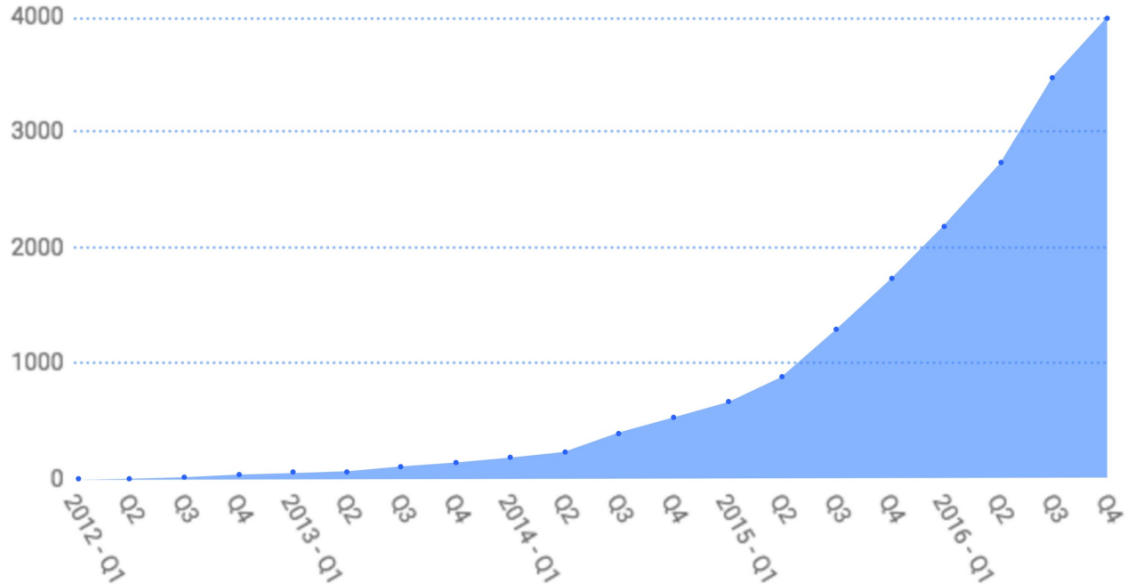
Cardboard
Smart stitching

Ads
Richer Text Ads
Automated Bidding

Google Cloud

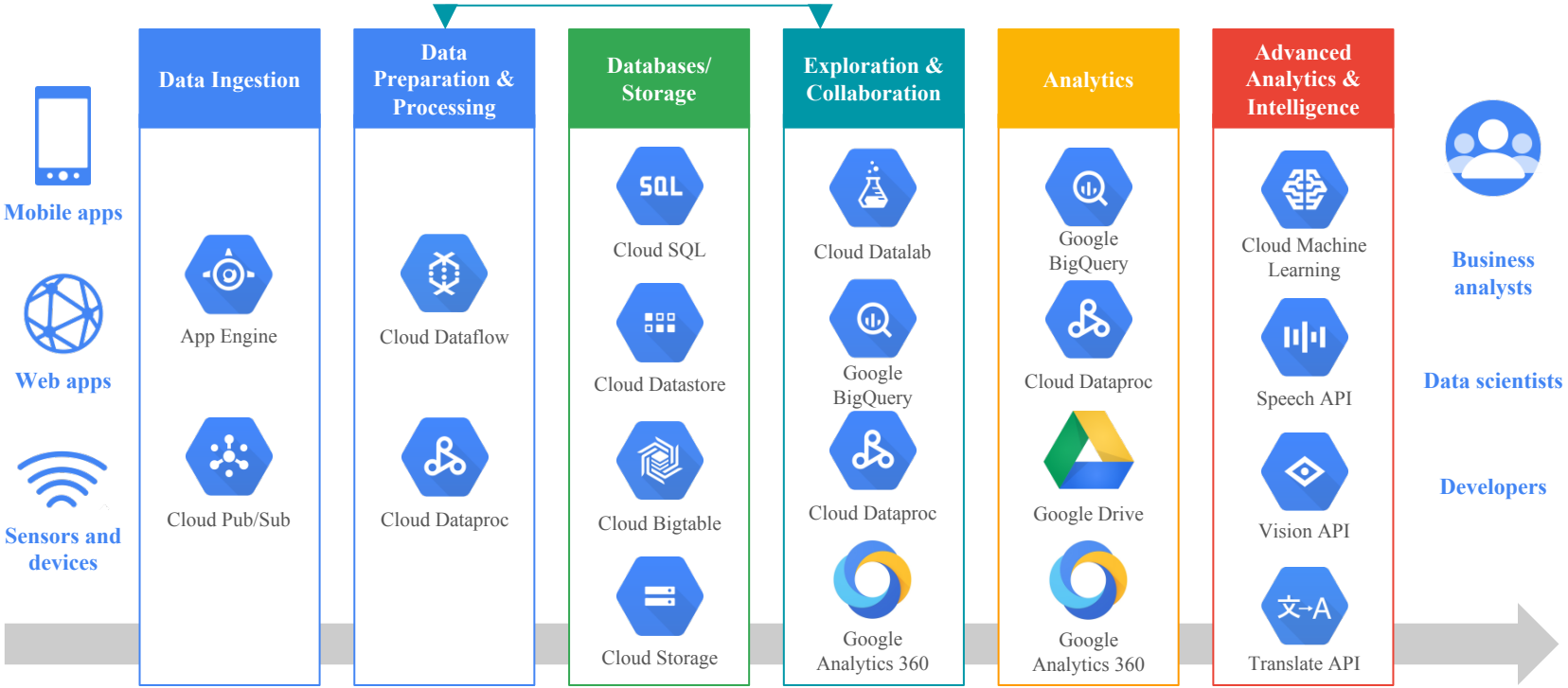

ВЫХОД В ГОРОД

Rapid accelerated use of Machine Learning at Google



Google Cloud

Transform Data into Actions



Google Cloud

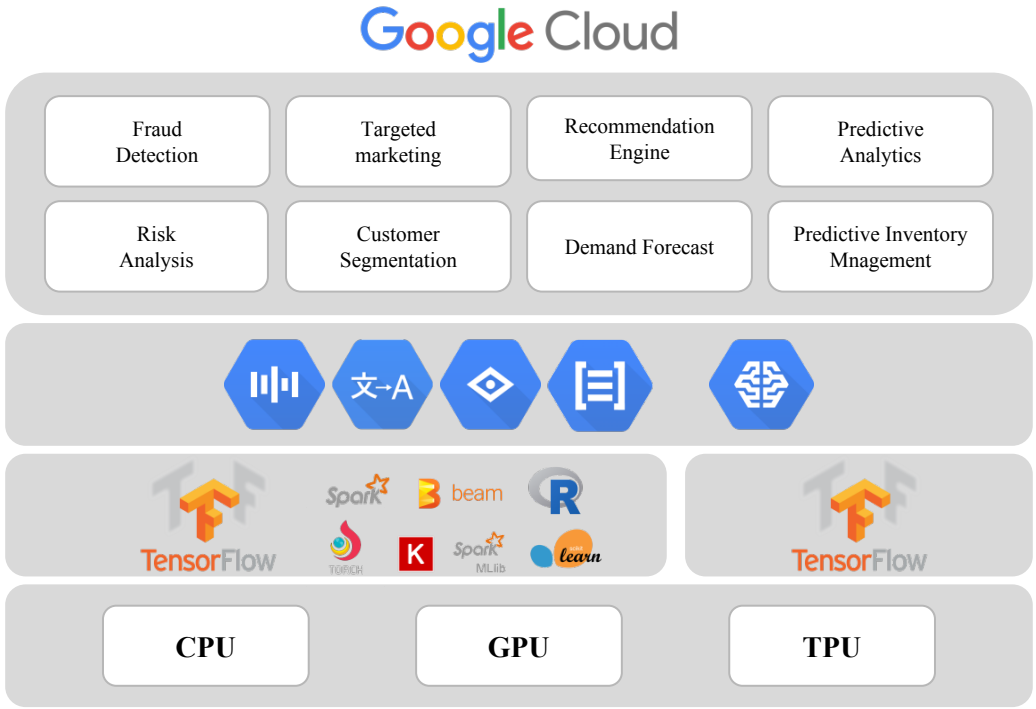
Google Cloud the end-to-end AI Platform

Industry Use-cases
In-loop inferencing for trained models

Cloud AI products
Pre-trained ML APIs to
Building custom ML models

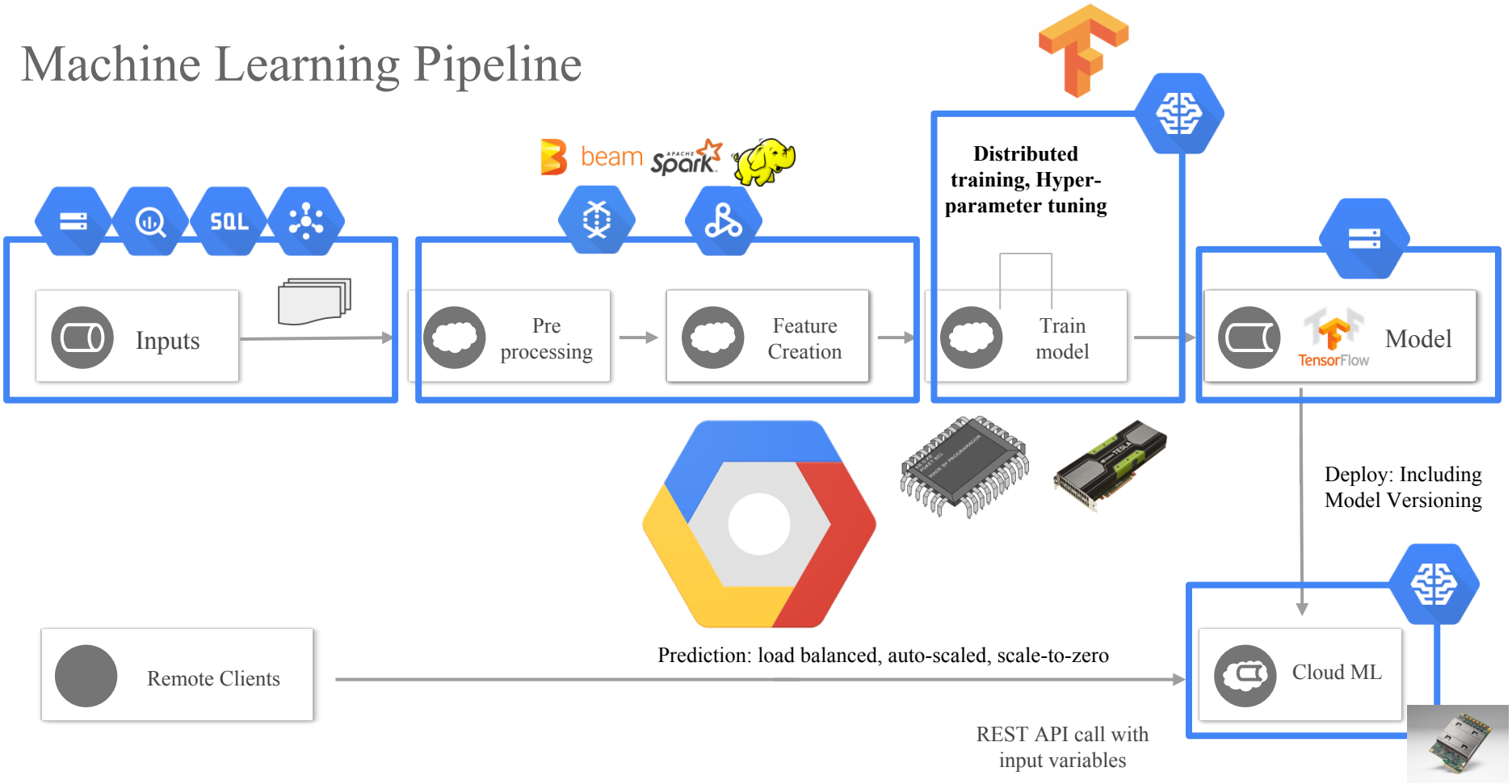
ML Framework
Industry-standard & widely adopted

Infrastructure
Best-in class processors for ML/DL



Google Cloud

Machine Learning Pipeline



Google Cloud

Results Enabled by Google Cloud

- **Faster Time to Insights** - Customer and product analytics up to 35x faster
- **Increased Customer Stickiness** - Used to only have weekly playlists. Now users can get up to seven playlists customized for them per day.

"Queries that used to take 20 minutes now take seconds. Faster results means more queries can be done, resulting in more finely tuned recommendations."

Wouter De Bie, Data Architect, Spotify


GO DATA FEST
16-20 OCT. 2017
XXX
THURSDAY 19 OCT. - GOOGLE
Google Cloud Platform

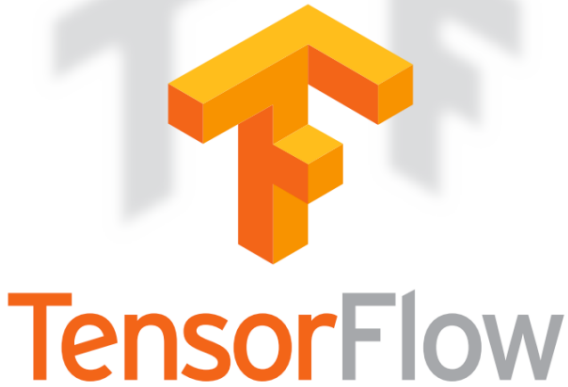
ML6

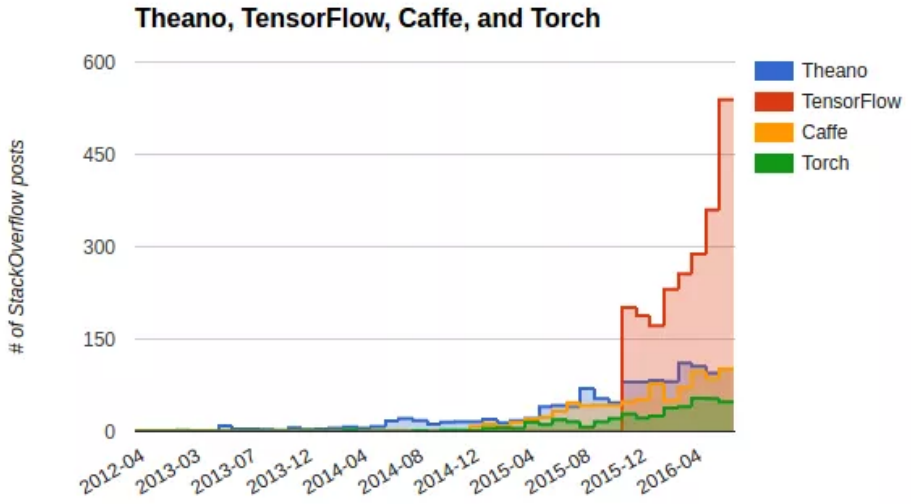
Who is ML6?

+ Team of Data Engineers, Data Scientists and Machine Learning Experts
+ Partner of Google Cloud
+ Artificial Intelligence Agency
+ Agile prototypes and robust solutions across various industries

info@ml6.eu
www.ML6.eu

Use case to illustrate what we actually do.



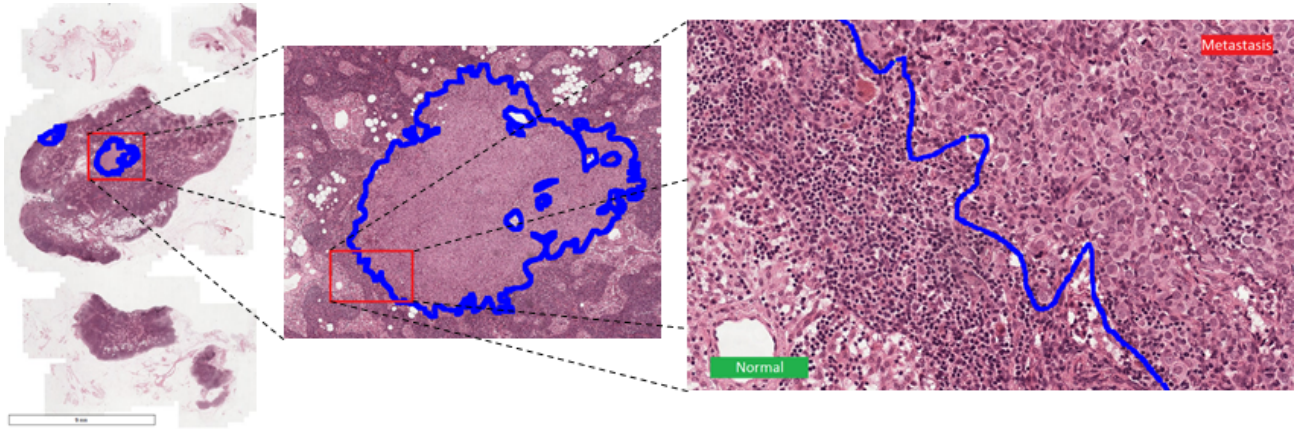


MEDICAL IMAGE CLASSIFICATION

TECHNICAL DETAILS - DATA

Camelyon16: ISBI challenge on cancer metastasis detection in lymph node

Training & Evaluation: 110 tumor slides, 160 normal slides, 130 evaluation slides



MEDICAL IMAGE CLASSIFICATION

TECHNICAL DETAILS - CHALLENGE BREAKDOWN

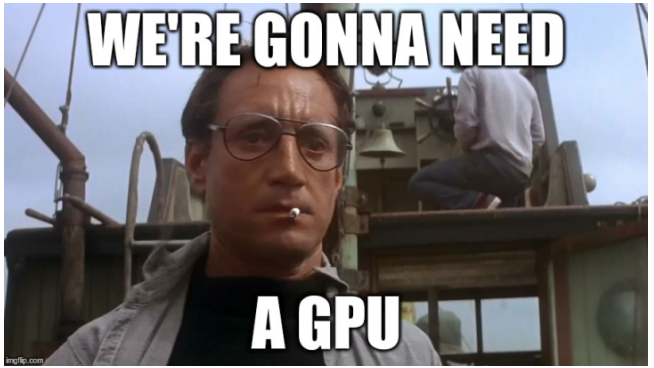
Not just another standard image classifier…

Training set construction

Computing environment

Network architecture

Post-processing (classification and segmentation)

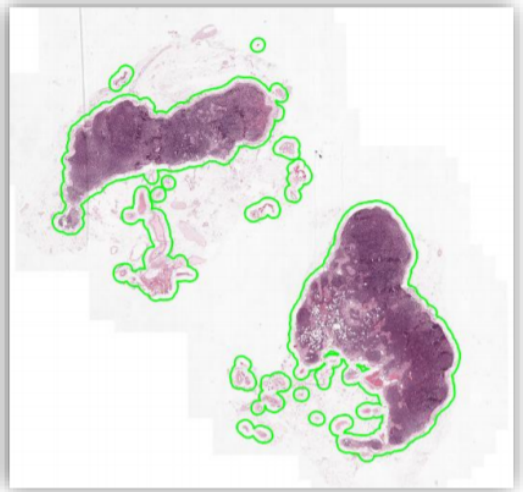


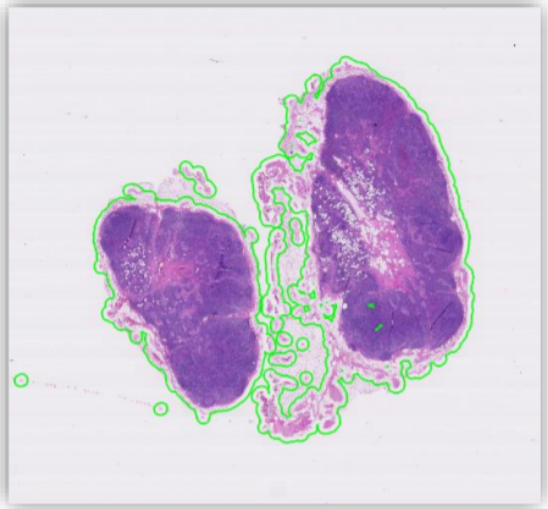
ML6

MEDICAL IMAGE CLASSIFICATION

TECHNICAL DETAILS - PREPROCESSING

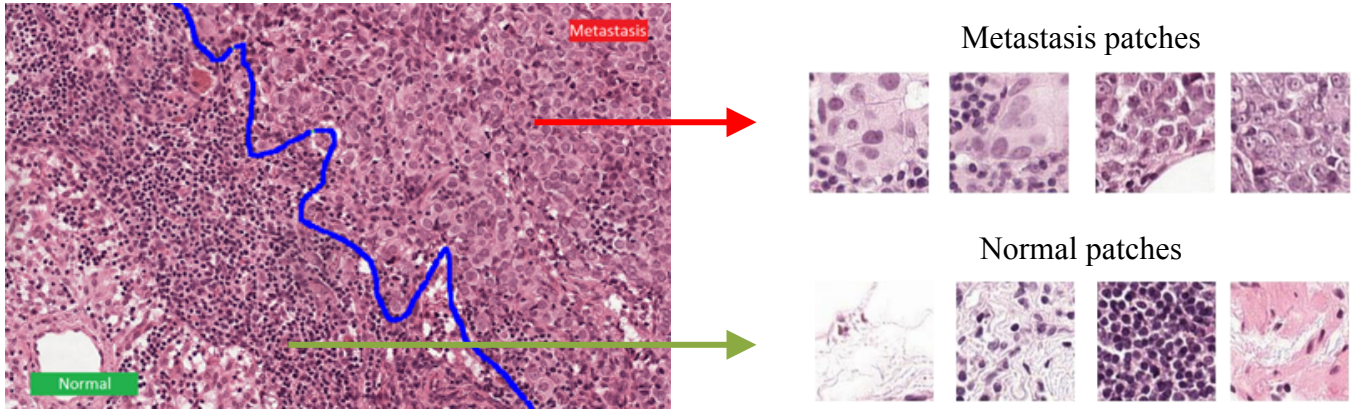
Tissue region segmentation with Otsu's method
allows for removing > 80% of image region on average





MEDICAL IMAGE CLASSIFICATION

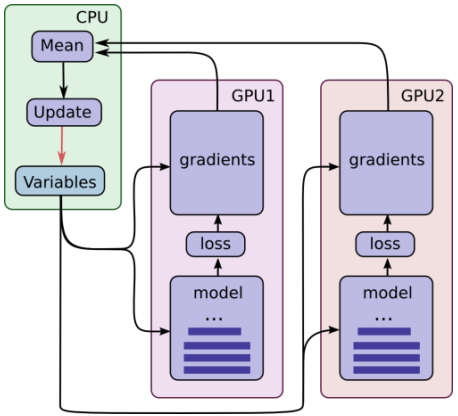
TECHNICAL DETAILS - PREPROCESSING



Millions of training examples are generated by sampling normal and metastasis regions of high-resolution whole-slide-images

MEDICAL IMAGE CLASSIFICATION

TECHNICAL DETAILS - MULTIPLE GPUs



Multiple GPUs were used to calculate **model updates**

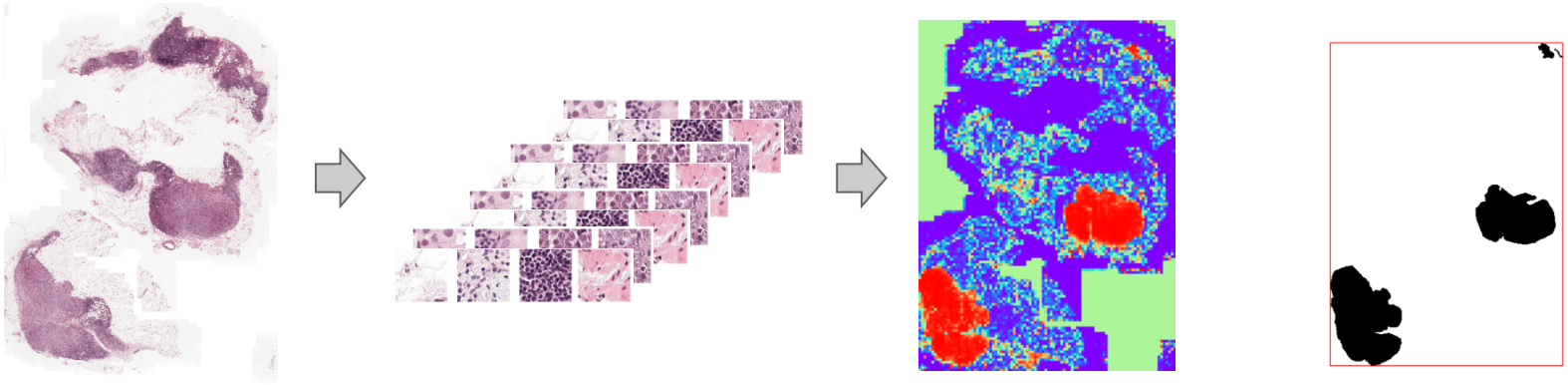
- individual model replica on each GPU
- all GPUs process a batch of data and send update of model parameters to CPU, which performs a synchronized update

Approach was extended to **Cloud ML Engine GPUs** during Healthcare Hackathon to build a dermatology POC in 1 day (by starting from a.o. pretrained VGG16)

MEDICAL IMAGE CLASSIFICATION

TECHNICAL DETAILS - FIRST RESULTS

By **scanning the patches** of a WSI, a **heatmap** can be generated



whole-slide-image | patches to classify | predicted cancer **heatmap** | ground-truth mask

First results of short training period seem **promising**

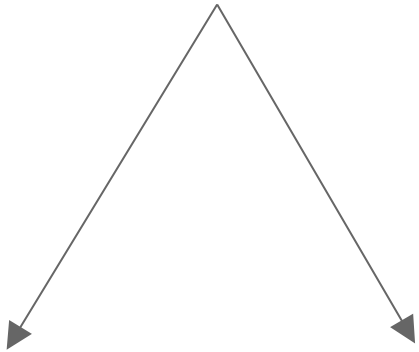
Longer training can lead to better results + extended **postprocessing** necessary

(e.g. extract higher level features from low level heatmaps, for tumor probability and location estimation)

ML6

Key Machine Learning Concepts

Supervised vs. Unsupervised Learning



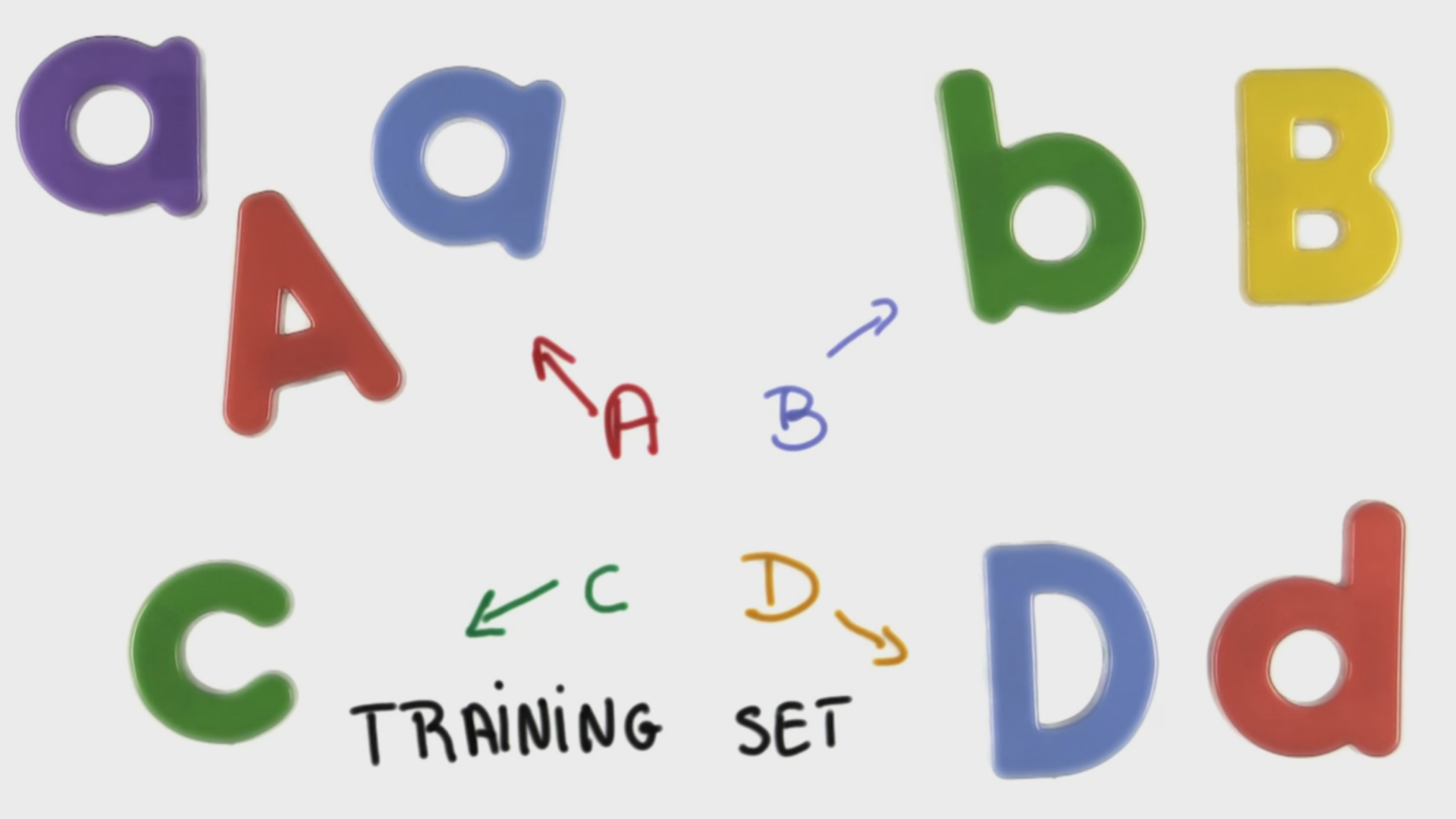
Regression **Classification**

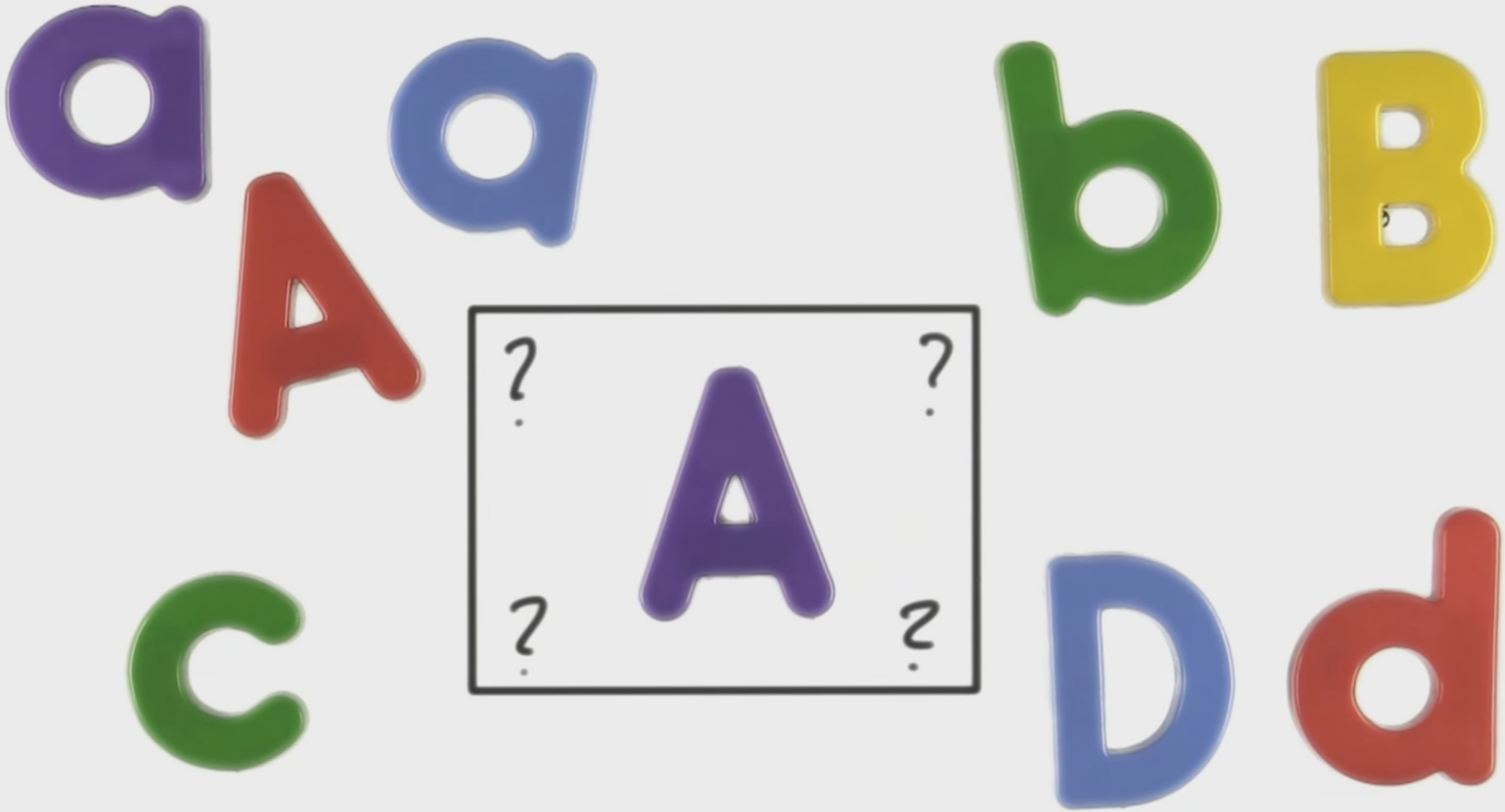
Google Cloud

TRAINING

VALIDATION

TESTING


A
B
C
D
TRAINING SET


a
a
A
b
B
?
?
A
?
?
c
D
d

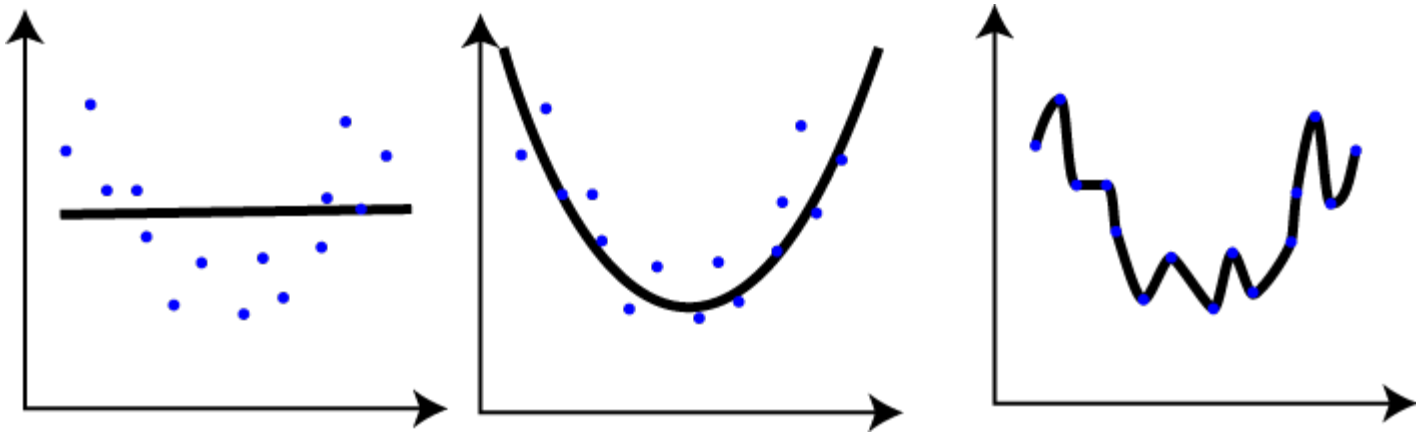
LOGISTIC CLASSIFIER

$$WX + b = Y$$

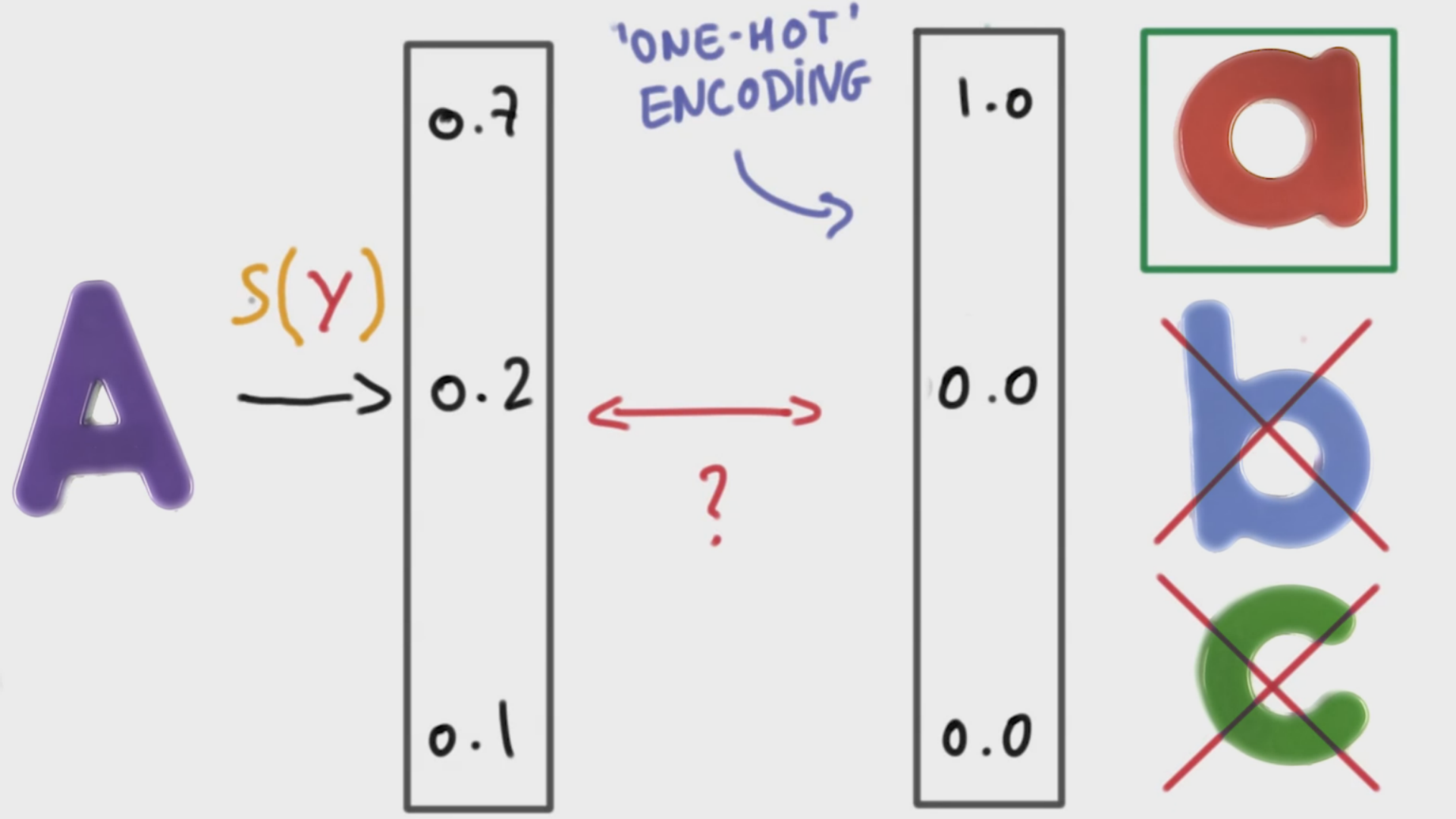
Model Evaluation

Google Cloud

Under-/Overfitting

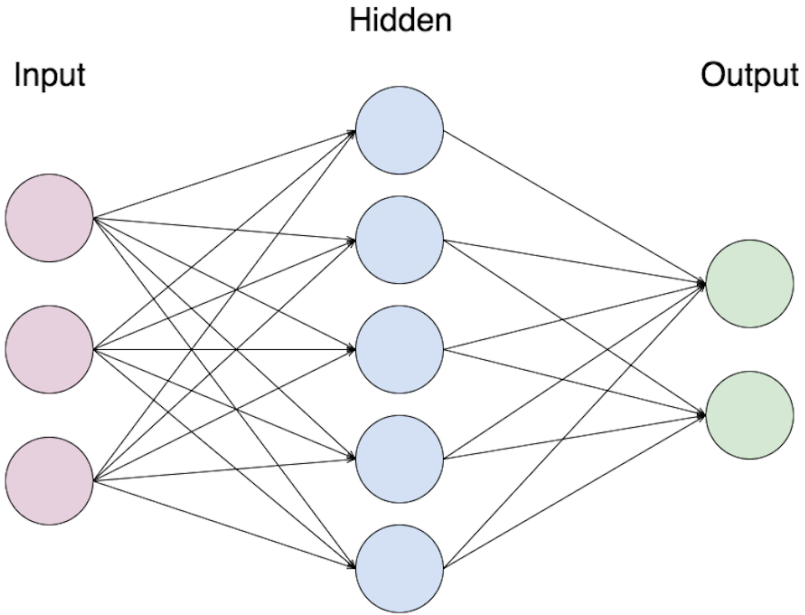


Google Cloud


'ONE-HOT' ENCODING
A
S(Y)
0.7
0.2
0.1
?
1.0
0.0
0.0

Neural Networks

- Multiple layers of nodes
- Multiple nodes in each layer
- Node = function with parameters
- Training in 2 phases
  - feed forward
  - backpropagation
- Optimizer



Google Cloud

IS THIS A

CAT or DOG?

CAT DOG

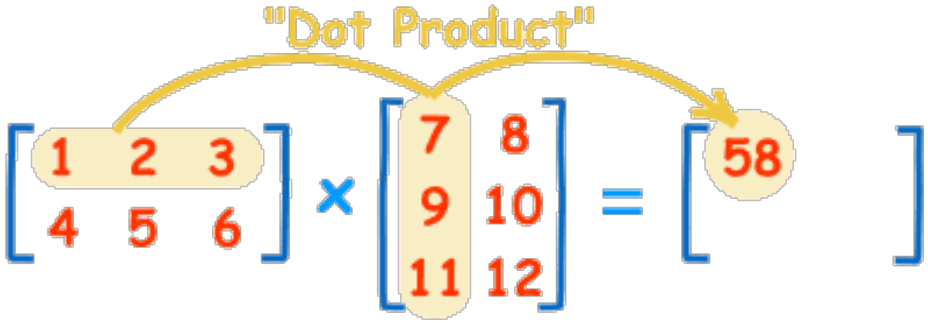
OUTPUT LAYER

ACTIVATED NEURONS

INPUT LAYER

DEEP *neural* NETWORK

Calculations with Matrices
Basic Recap



$$\begin{bmatrix} 1 & 4 \\ -2 & 3 \end{bmatrix} \cdot \begin{bmatrix} 1 & 3 & -4 \\ 6 & 8 & 0 \end{bmatrix}$$

$(2 \times \mathbf{2}) \cdot (\mathbf{2} \times 3)$

same

$$\begin{bmatrix} 5 & -3 \\ 4 & 1 \\ 7 & -2 \end{bmatrix} \cdot \begin{bmatrix} 8 \\ 2 \end{bmatrix}$$

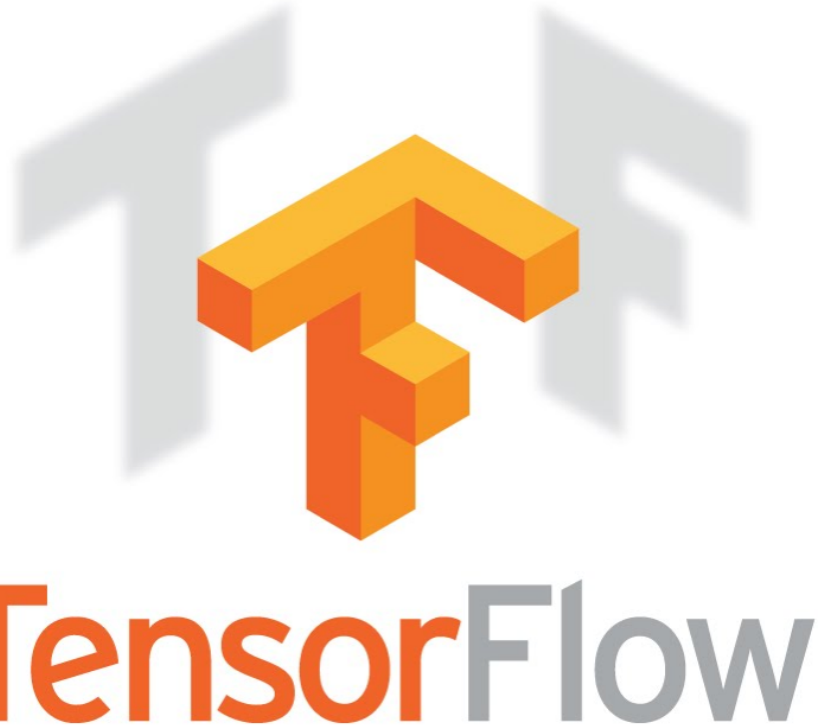
$(3 \times \mathbf{2}) \cdot (\mathbf{2} \times 1)$

same

Google Cloud

Google

- Open-source machine- and deep learning library
- Numerical computations using data flow graphs
  - Nodes: Mathematical operations
  - Edges: Multidimensional data arrays called tensors
- Flexible architecture allows computations on multiple CPU's or GPU's



Are you ready?

Google

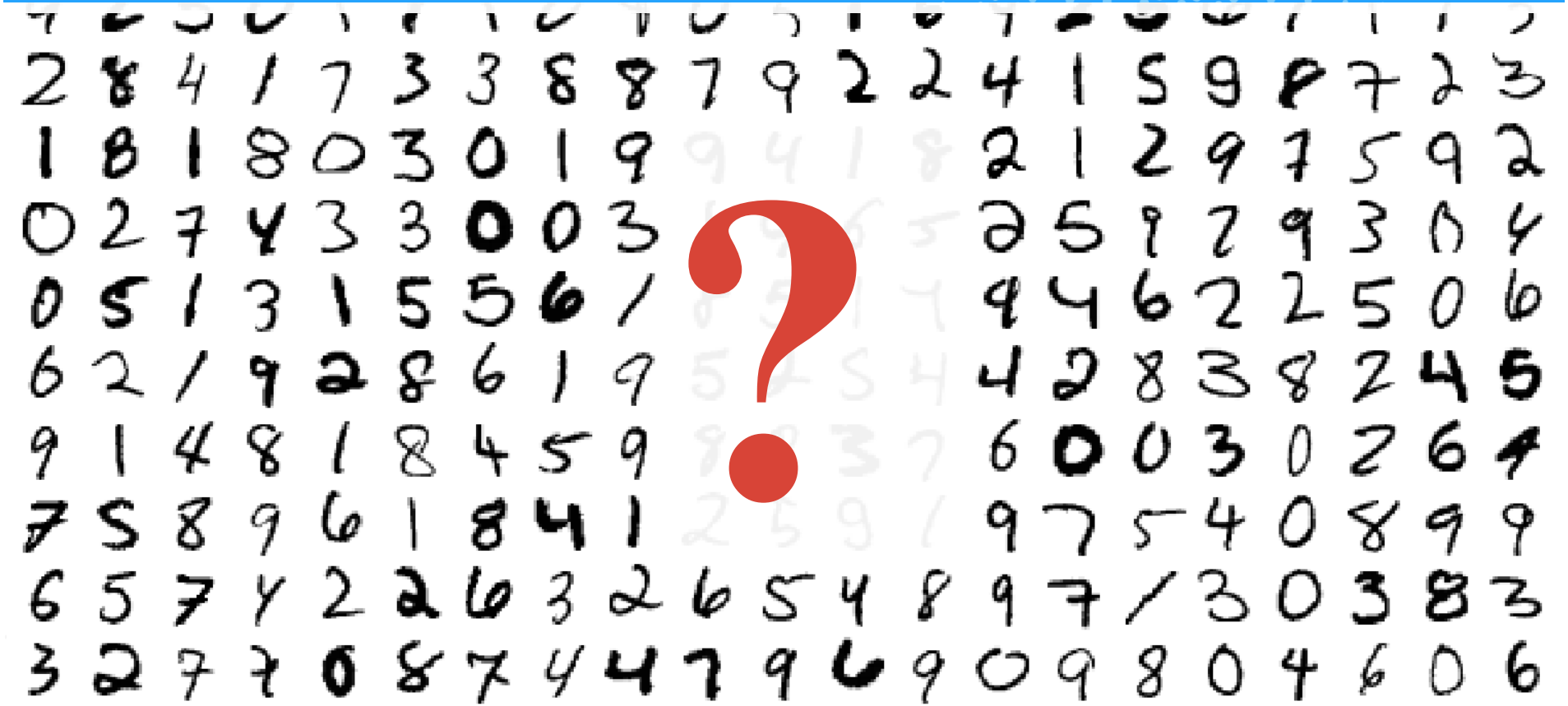
Let’s get started!

- Martin Görner

- How to set everything up?

  bit.ly/2yg1nD0

Hello World: handwritten digits classification - MNIST



MNIST = Mixed National Institute of Standards and Technology - Download the dataset at http://yann.lecun.com/exdb/mnist/

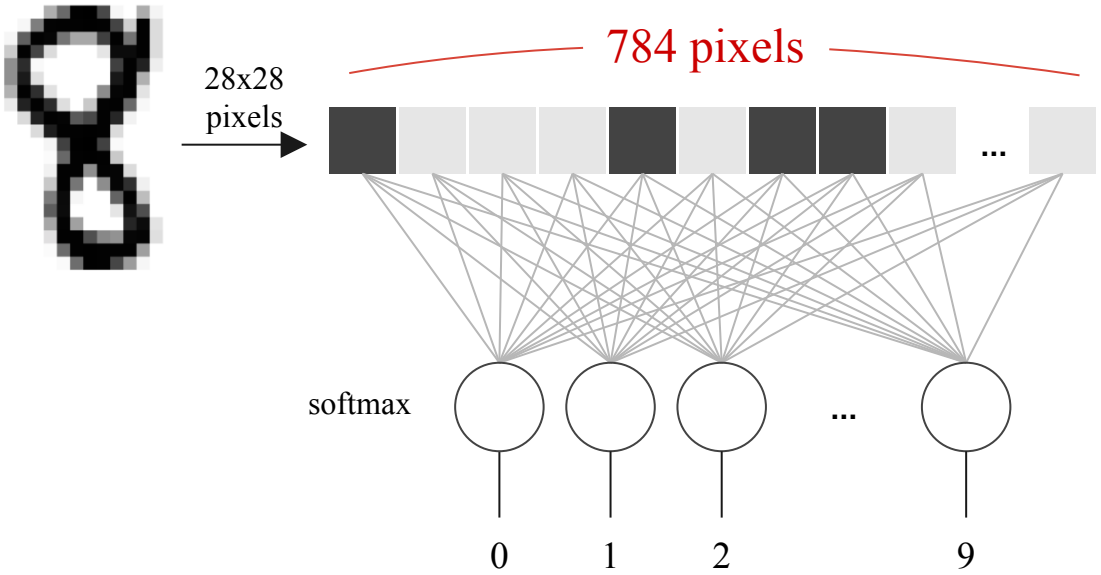
HOW A DEEP NEURAL NETWORK SEES

Image

“Audi A7”

Image source: “Unsupervised Learning of Hierarchical Representations with Convolutional Deep Belief Networks” ICML 2009 & Comm. ACM 2011.
Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Ng.

Very simple model: softmax classification

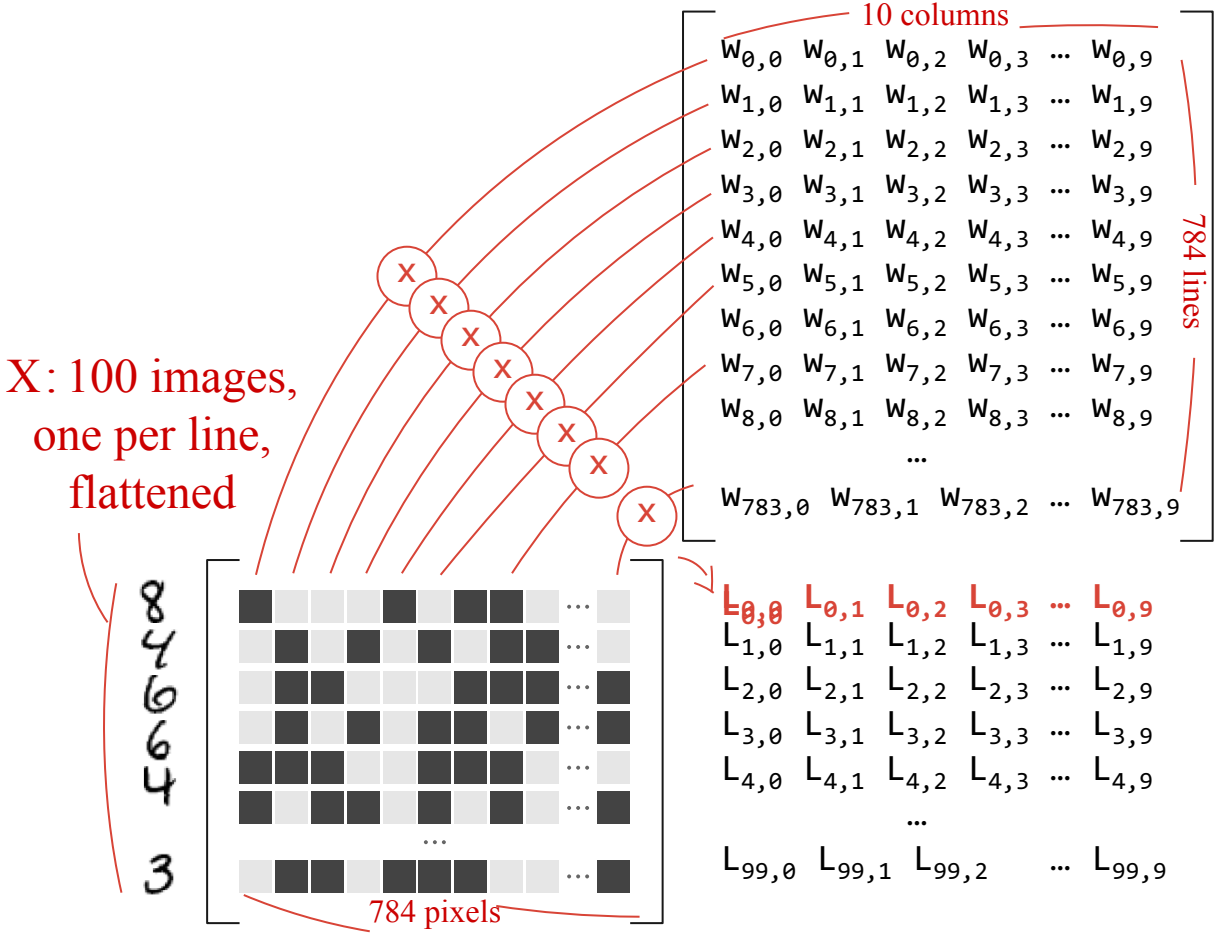


weighted sum of all pixels + bias

$$softmax(L_n) = \frac{e^{L_n}}{||e^{L}||}$$

neuron outputs

In matrix notation, 100 images at a time



Softmax, on a batch of images

Predictions Y[100, 10]

Images X[100, 784]

Weights W[784,10]

Biases b[10]

$$Y = softmax(X.W + b)$$

applied line by line

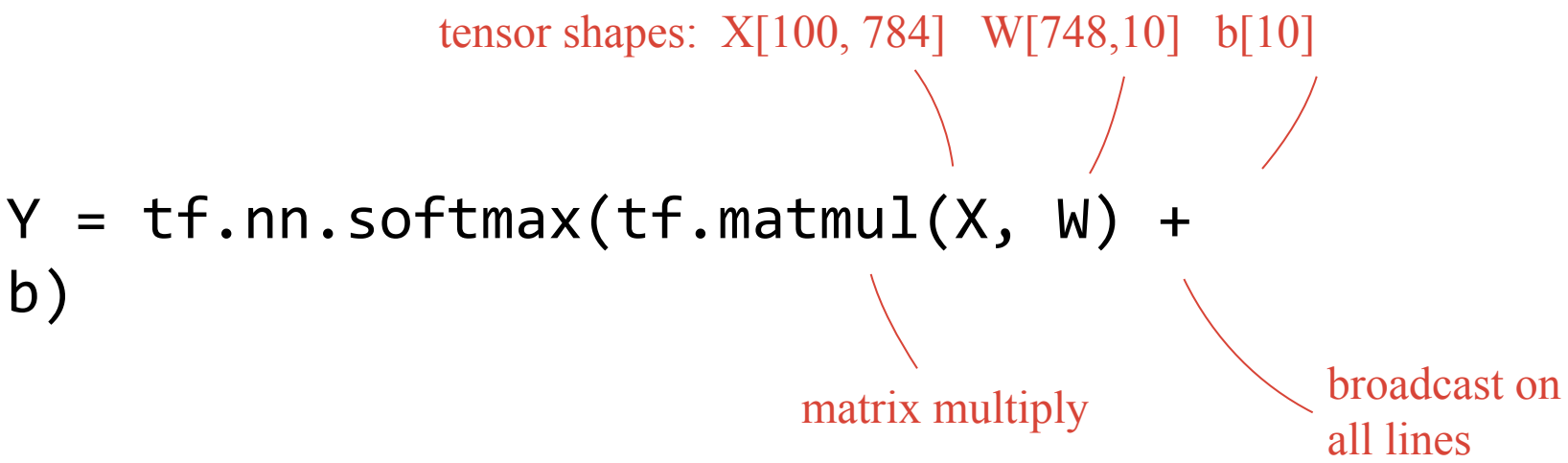
matrix multiply

broadcast on all lines

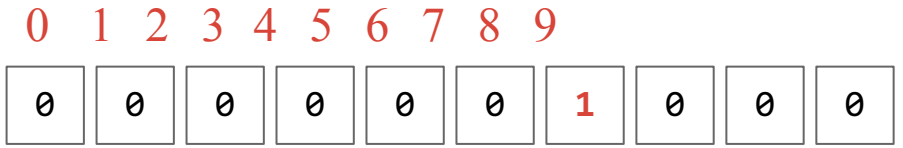
tensor shapes in [ ]

Google Cloud

Now in TensorFlow (Python)



Success ?

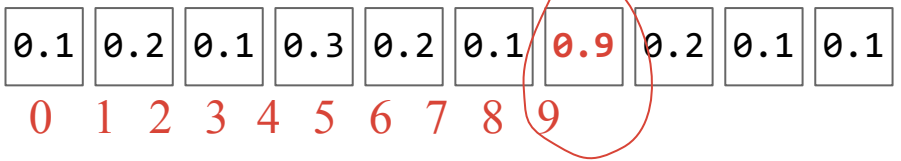


actual probabilities, “one-hot” encoded

Cross entropy: $-\sum Y_i' . log(Y_i)$

computed probabilities

this is a “6”



Google Cloud

TensorFlow - initialisation

```
import tensorflow as tf

X = tf.placeholder(tf.float32, [None, 28, 28, 1])
W = tf.Variable(tf.zeros([784, 10]))
b = tf.Variable(tf.zeros([10]))

init = tf.initialize_all_variables()
```

this will become the batch size, 100

28 x 28 grayscale images

Training = computing variables W and b

TensorFlow - success metrics

flattening images

```
# model
Y = tf.nn.softmax(tf.matmul(tf.reshape(X, [-1, 784]), W) + b)
# placeholder for correct answers
Y_ = tf.placeholder(tf.float32, [None, 10])

# loss function
cross_entropy = -tf.reduce_sum(Y_ * tf.log(Y))

# % of correct answers found in batch
is_correct = tf.equal(tf.argmax(Y,1), tf.argmax(Y_,1))
accuracy = tf.reduce_mean(tf.cast(is_correct, tf.float32))
```

"one-hot" encoded

"one-hot" decoding

Google Cloud

TensorFlow - training

learning rate

```
optimizer = tf.train.GradientDescentOptimizer(0.003)
train_step = optimizer.minimize(cross_entropy)
```

loss function

TensorFlow - run !

```
sess = tf.Session()
sess.run(init)

for i in range(1000):
    # load batch of images and correct answers
    batch_X, batch_Y = mnist.train.next_batch(100)
    train_data={X: batch_X, Y_: batch_Y}

    # train
    sess.run(train_step, feed_dict=train_data)

    # success ?
    a,c = sess.run([accuracy, cross_entropy], feed_dict=train_data)

    # success on test data ?
    test_data={X: mnist.test.images, Y_: mnist.test.labels}
    a,c = sess.run([accuracy, cross_entropy], feed=test_data)
```

running a Tensorflow computation, feeding placeholders

Tip: do this every 100 iterations

TensorFlow - full python code

```
import tensorflow as tf


X = tf.placeholder(tf.float32, [None, 28, 28, 1])
W = tf.Variable(tf.zeros([784, 10]))
b = tf.Variable(tf.zeros([10]))
init = tf.initialize_all_variables()

# model
Y=tf.nn.softmax(tf.matmul(tf.reshape(X,[-1, 784]), W) + b)

# placeholder for correct answers
Y_ = tf.placeholder(tf.float32, [None, 10])

# loss function
cross_entropy = -tf.reduce_sum(Y_ * tf.log(Y))

# % of correct answers found in batch
is_correct = tf.equal(tf.argmax(Y,1), tf.argmax(Y_,1))
accuracy = tf.reduce_mean(tf.cast(is_correct,tf.float32))
```

initialisation

model

success metrics

training step

```
optimizer = tf.train.GradientDescentOptimizer(0.003)
train_step = optimizer.minimize(cross_entropy)


sess = tf.Session()
sess.run(init)

for i in range(10000):
      # load batch of images and correct answers
      batch_X, batch_Y = mnist.train.next_batch(100)
      train_data={X: batch_X, Y_: batch_Y}

      # train
      sess.run(train_step, feed_dict=train_data)

            # success ? add code to print it
      a,c = sess.run([accuracy, cross_entropy], feed=train_data)

            # success on test data ?
      test_data={X:mnist.test.images, Y_:mnist.test.labels}
      a,c = sess.run([accuracy, cross_entropy], feed=test_data)
```

Run

Google Cloud

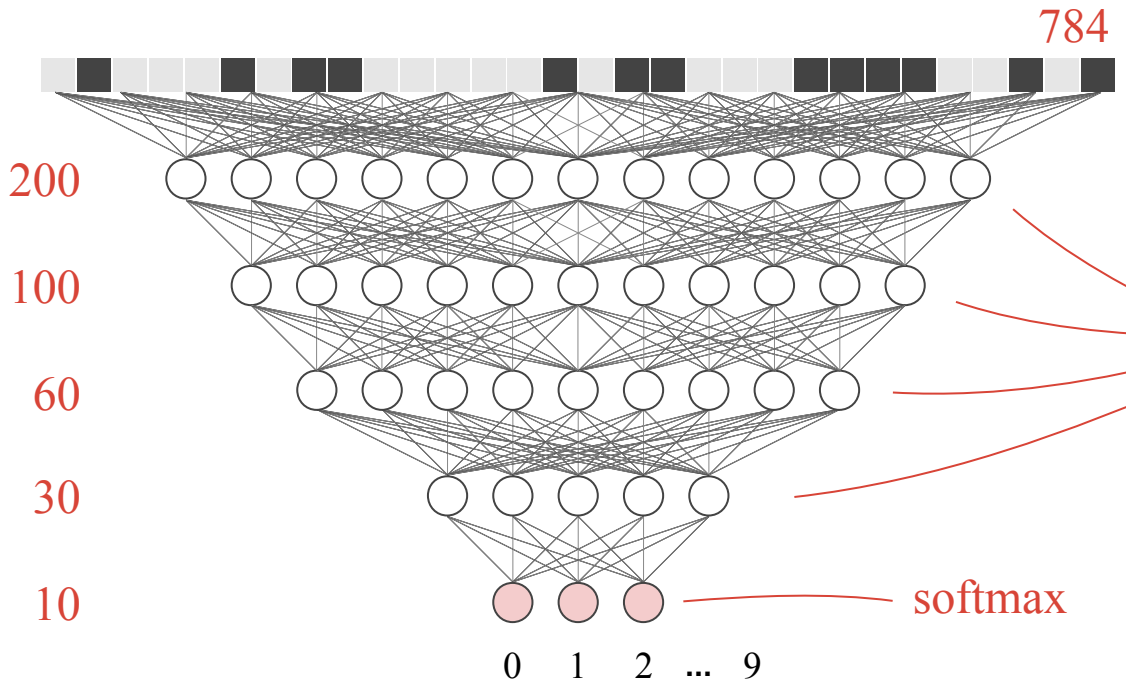
Cookbook

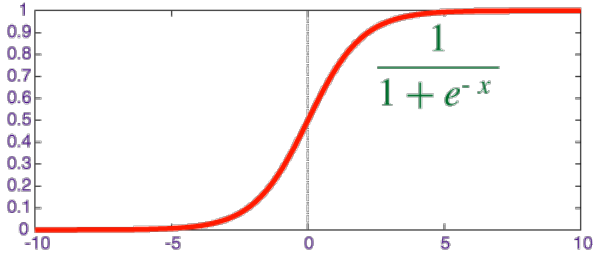
Softmax
Cross-entropy
Mini-batch


Go deep !

Let's try 5 fully-connected layers !

overkill





sigmoid function

Google Cloud

TensorFlow - initialisation

```
K = 200
L = 100
M = 60
N = 30
```

weights initialised with random values

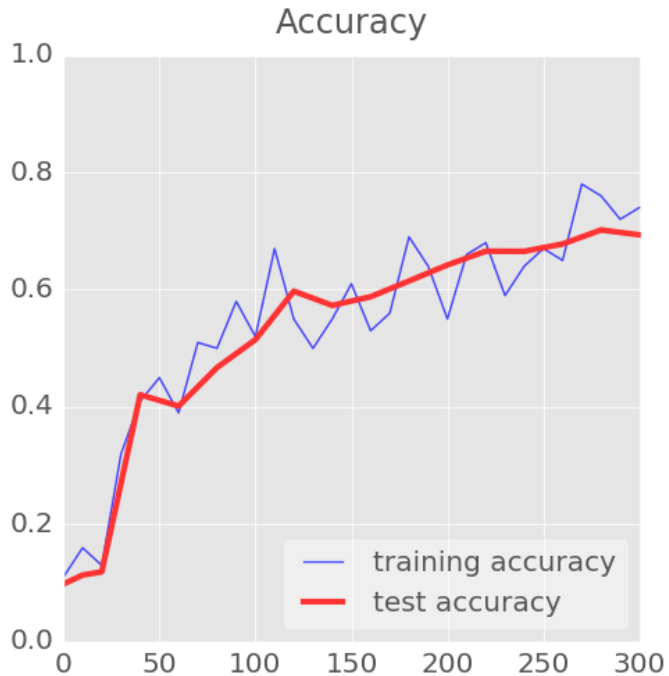
```
W1 = tf.Variable(tf.truncated_normal([28*28, K] ,stddev=0.1))
B1 = tf.Variable(tf.zeros([K]))

W2 = tf.Variable(tf.truncated_normal([K, L], stddev=0.1))
B2 = tf.Variable(tf.zeros([L]))

W3 = tf.Variable(tf.truncated_normal([L, M], stddev=0.1))
B3 = tf.Variable(tf.zeros([M]))
W4 = tf.Variable(tf.truncated_normal([M, N], stddev=0.1))
B4 = tf.Variable(tf.zeros([N]))
W5 = tf.Variable(tf.truncated_normal([N, 10], stddev=0.1))
B5 = tf.Variable(tf.zeros([10]))
```

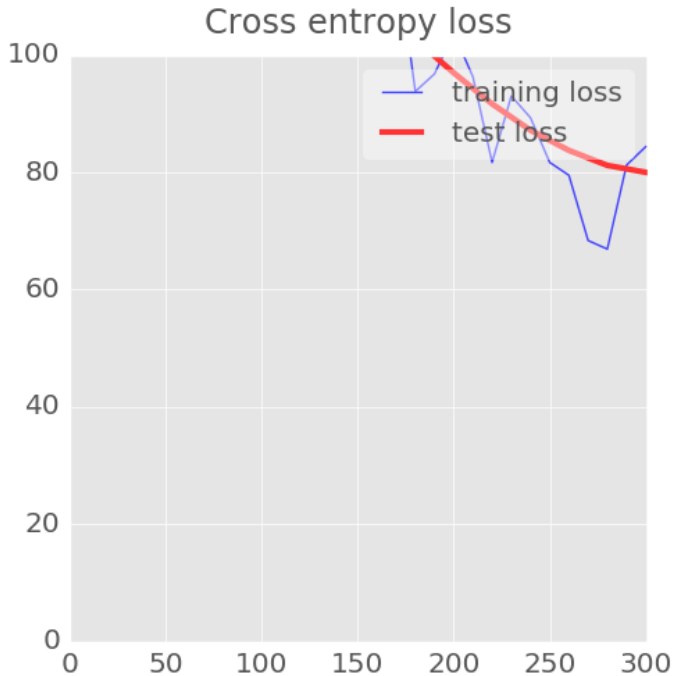
TensorFlow - the model

weights and biases

```
X = tf.reshape(X, [-1, 28*28])

Y1 = tf.nn.sigmoid(tf.matmul(X, W1) + B1)
Y2 = tf.nn.sigmoid(tf.matmul(Y1, W2) + B2)
Y3 = tf.nn.sigmoid(tf.matmul(Y2, W3) + B3)
Y4 = tf.nn.sigmoid(tf.matmul(Y3, W4) + B4)
Y = tf.nn.softmax(tf.matmul(Y4, W5) + B5)
```

Demo - slow start ?



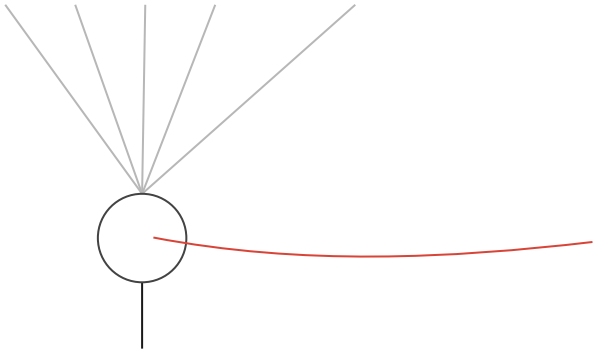


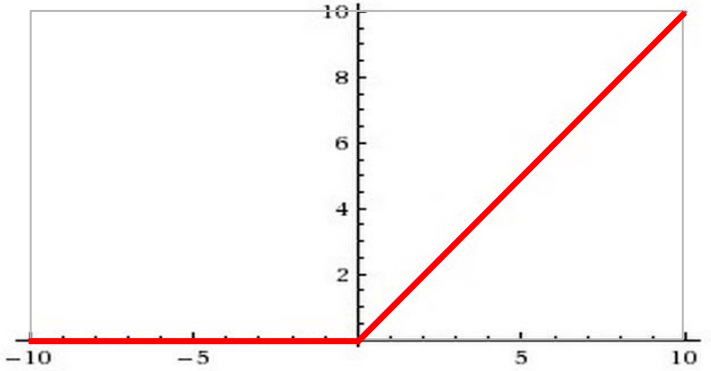
Google Cloud


Relu !

RELU

RELU = Rectified Linear Unit

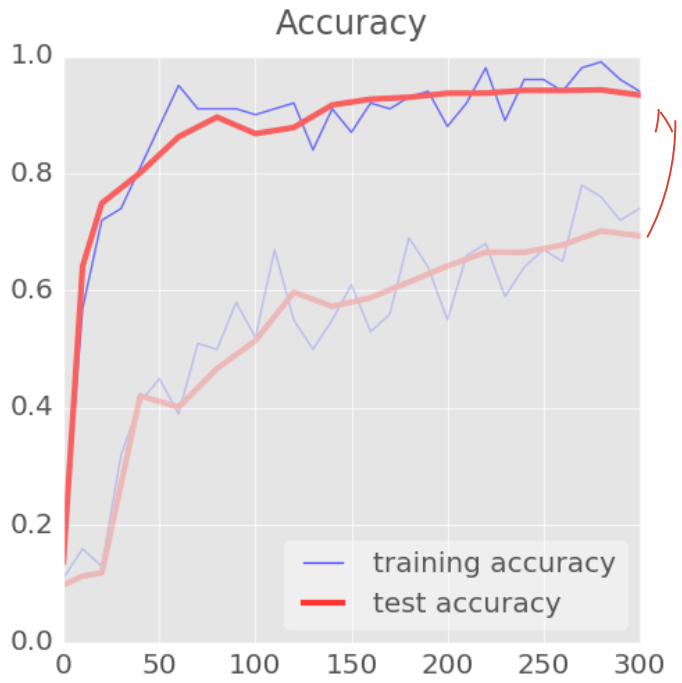


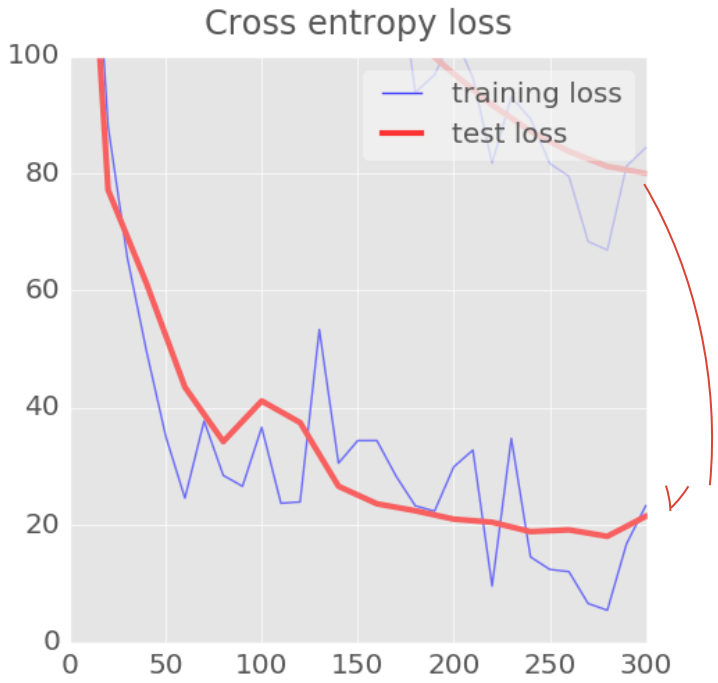



```
Y = tf.nn.relu(tf.matmul(X, W) + b)
```

Google Cloud

RELU

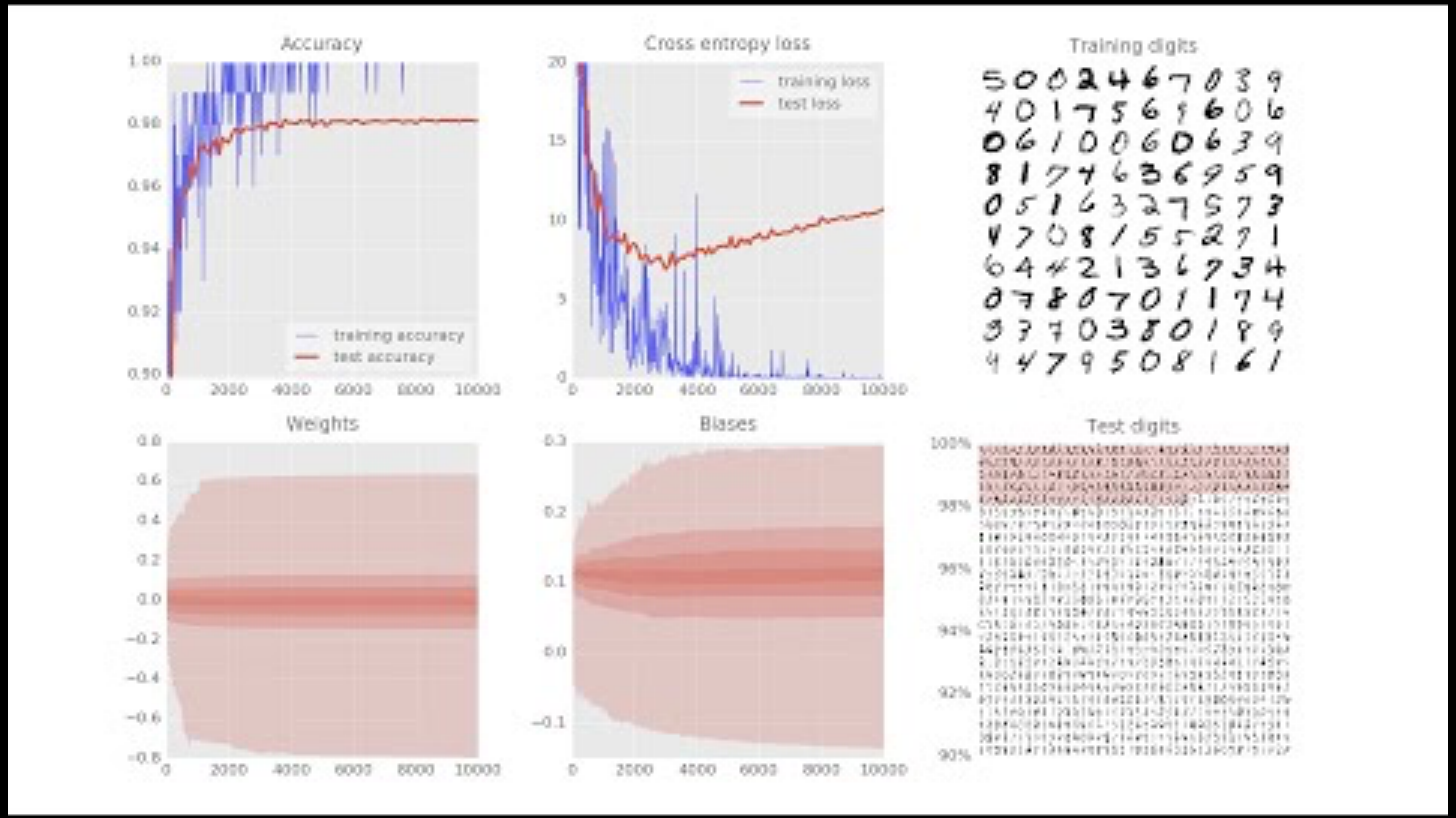


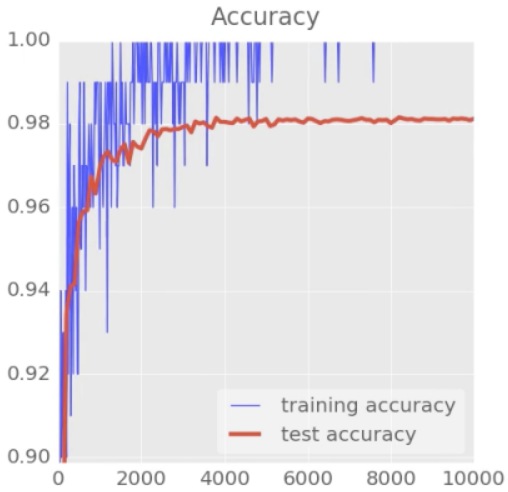


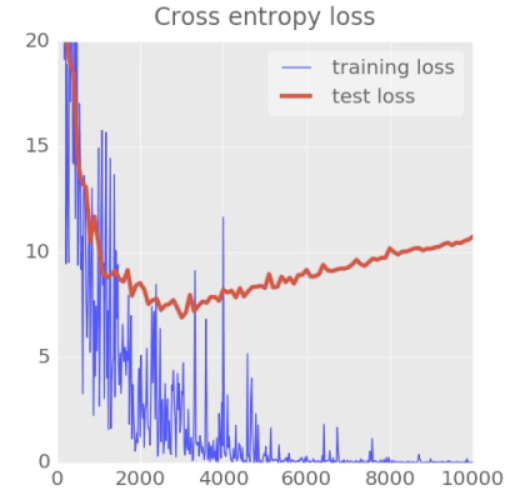
Google Cloud

Slow down . . .

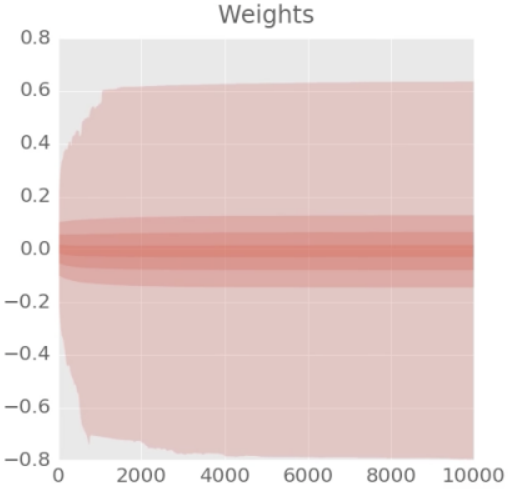
Demo

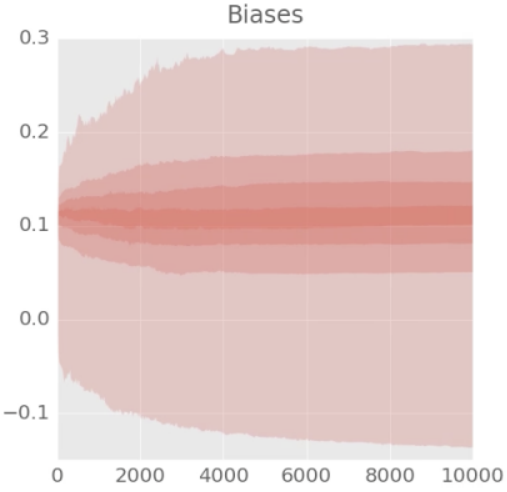

Accuracy
training accuracy
test accuracy
Cross entropy loss
training loss
test loss
Training digits
Weights
Biases
Test digits

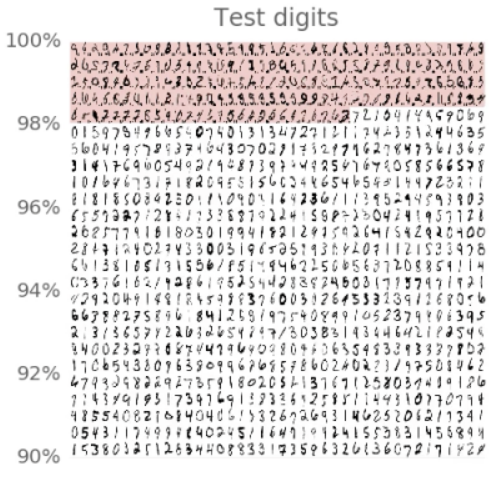

Accuracy
1.00
0.98
0.96
0.94
0.92
0.90
0
2000
4000
6000
8000
10000
training accuracy
test accuracy


Cross entropy loss
20
15
10
5
0
0
2000
4000
6000
8000
10000
training loss
test loss


Training digits

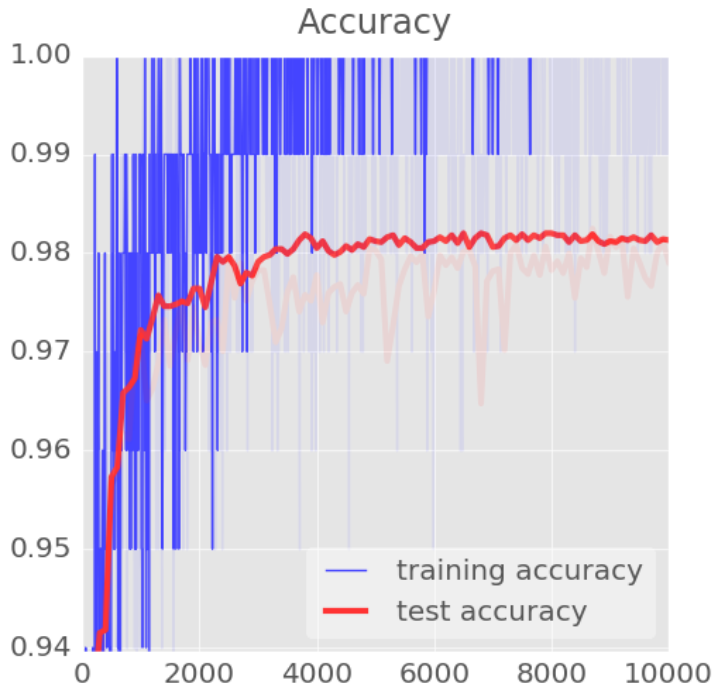

Weights
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8
0
2000
4000
6000
8000
10000

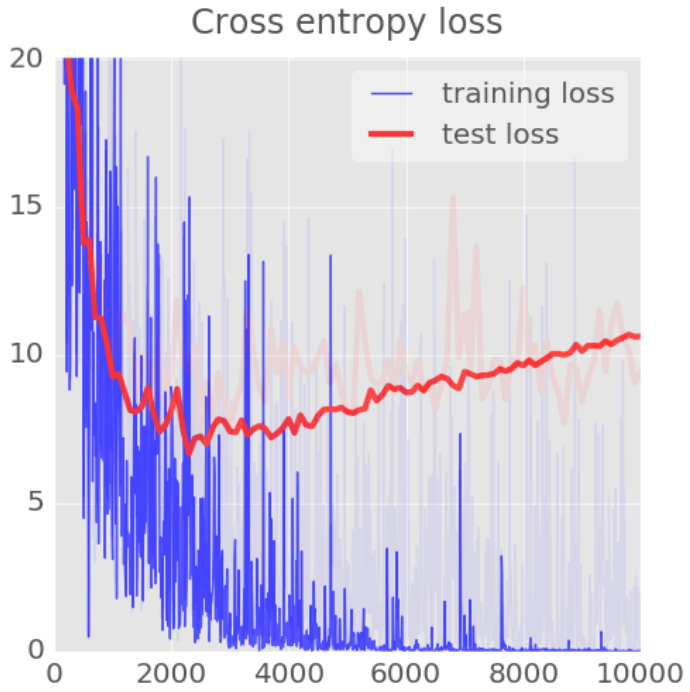

Biases
0.3
0.2
0.1
0.0
−0.1
0
2000
4000
6000
8000
10000


Test digits
100%
98%
96%
94%
92%
90%

98%

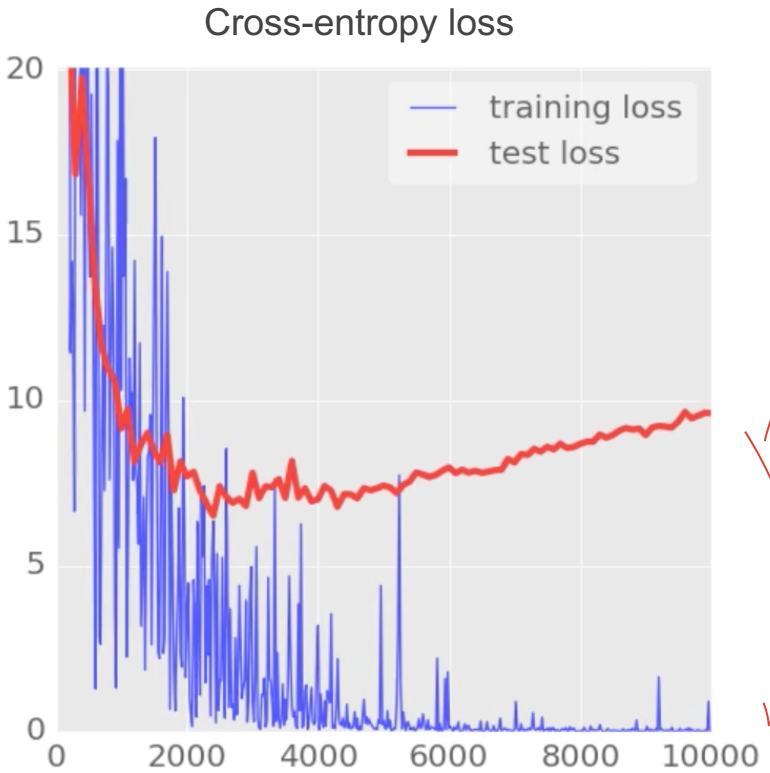
Learning rate decay





Learning rate 0.003 at start then dropping exponentially to 0.0001

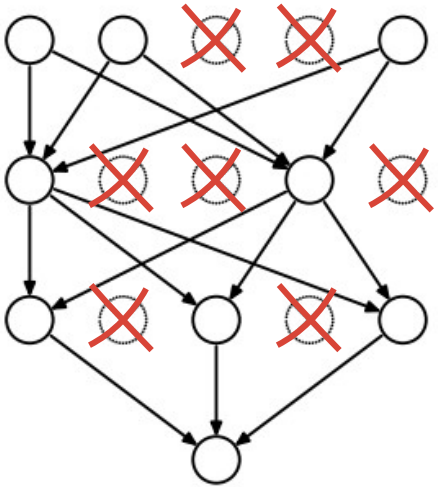
Overfitting



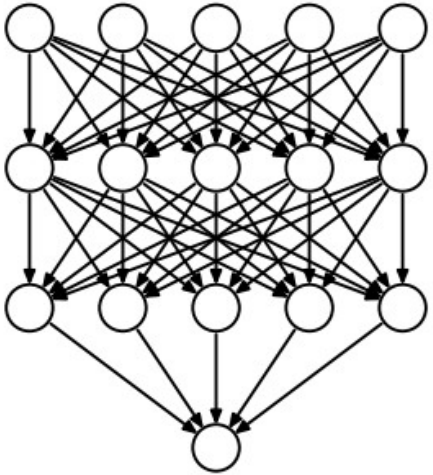
Overfitting ?

Dropout

Dropout



TRAINING
pkeep=0.75

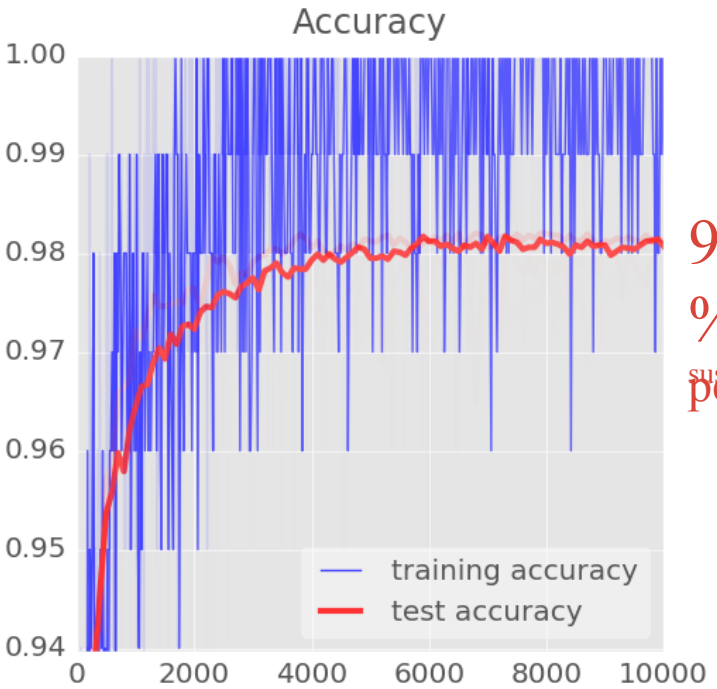


EVALUATION
pkeep=1

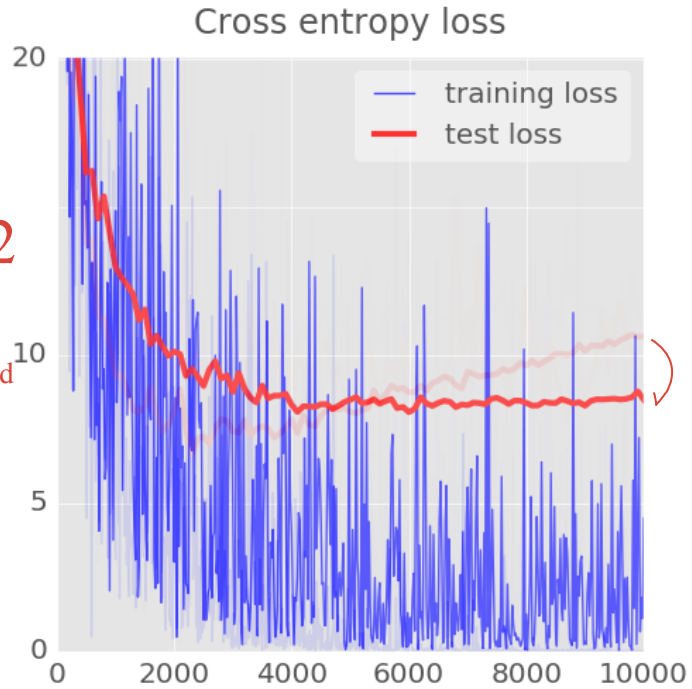
```
pkeep =
tf.placeholder(tf.float32)

Yf = tf.nn.relu(tf.matmul(X, W) + B)
Y  = tf.nn.dropout(Yf, pkeep)
```

Google Cloud

All the party tricks

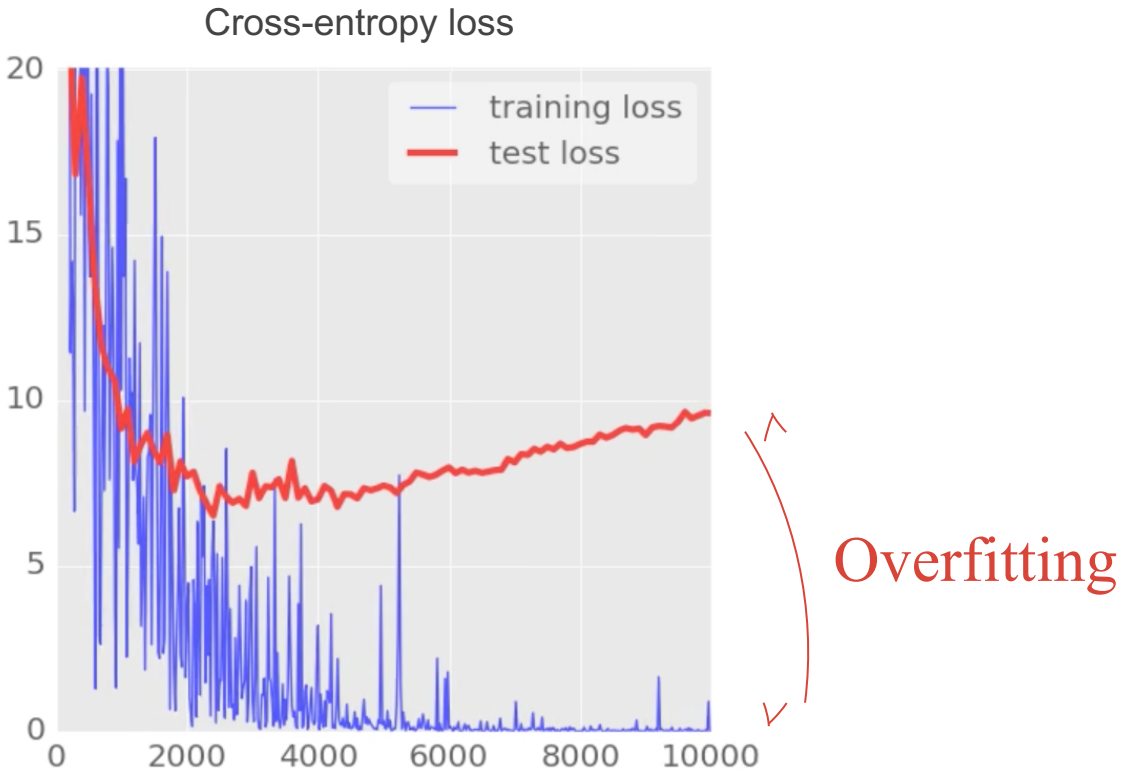


98.2 % sustained peak



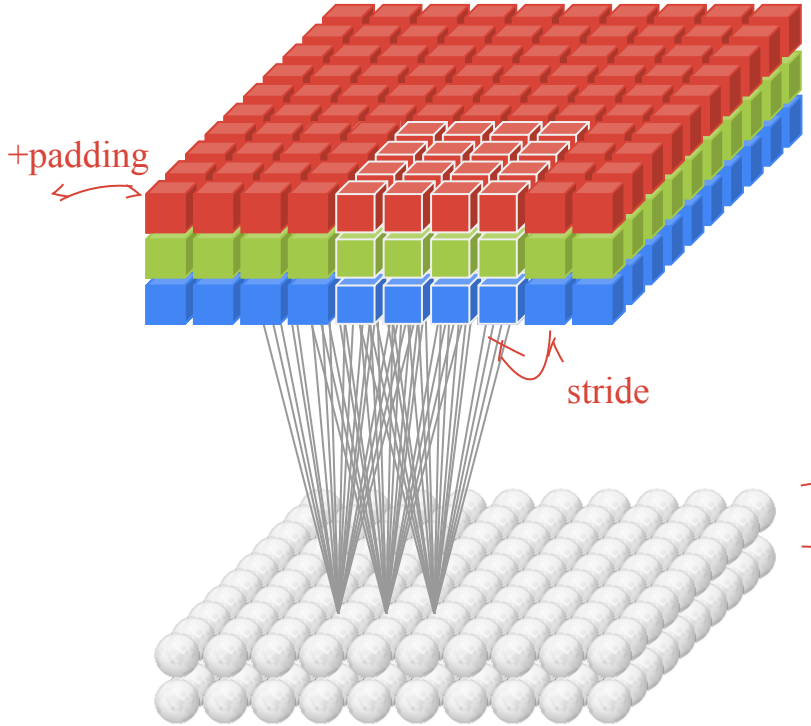
RELU, decaying learning rate 0.003 -> 0.0001 and dropout 0.75

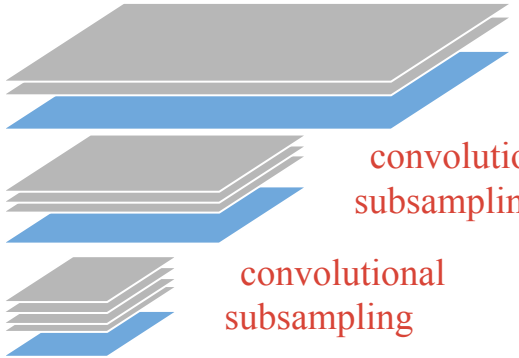
Overfitting




Overfitting
?!?
Too many
neurons
Not
enough
DATA
BAD
Network

Convolutional layer





$W_1[4, 4, 3]$

$W_2[4, 4, 3]$

$W[4, 4, 3, 2]$

filter size

input channels

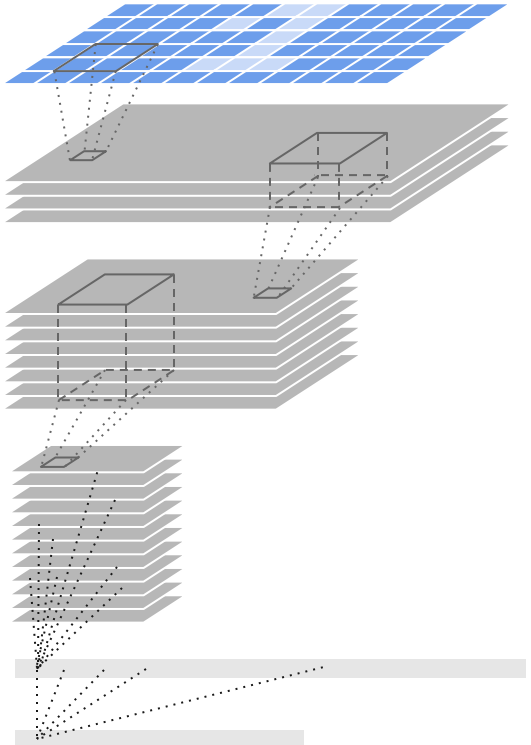
output channels

Google Cloud

Hacker’s tip

Convolutional neural network

+ biases on all layers



convolutional layer, 4 channels
W1[5, 5, 1, 4] stride 1

convolutional layer, 8 channels
W2[4, 4, 4, 8] stride 2

convolutional layer, 12 channels
W3[4, 4, 8, 12] stride 2

fully connected layer W4[7x7x12, 200]
softmax readout layer W5[200, 10]

Tensorflow - initialisation

filter size

input channels

output channels

```
K=4
L=8
M=12

W1 = tf.Variable(tf.truncated_normal([5, 5, 1, K] ,stddev=0.1))
B1 = tf.Variable(tf.ones([K])/10)
W2 = tf.Variable(tf.truncated_normal([5, 5, K, L] ,stddev=0.1))
B2 = tf.Variable(tf.ones([L])/10)
W3 = tf.Variable(tf.truncated_normal([4, 4, L, M] ,stddev=0.1))
B3 = tf.Variable(tf.ones([M])/10)

N=200

W4 = tf.Variable(tf.truncated_normal([7*7*M, N] ,stddev=0.1))
B4 = tf.Variable(tf.ones([N])/10)
W5 = tf.Variable(tf.truncated_normal([N, 10] ,stddev=0.1))
B5 = tf.Variable(tf.zeros([10])/10)
```

weights initialised with random values

Tensorflow - the model

input image batch
X[100, 28, 28, 1]

weights

stride

biases

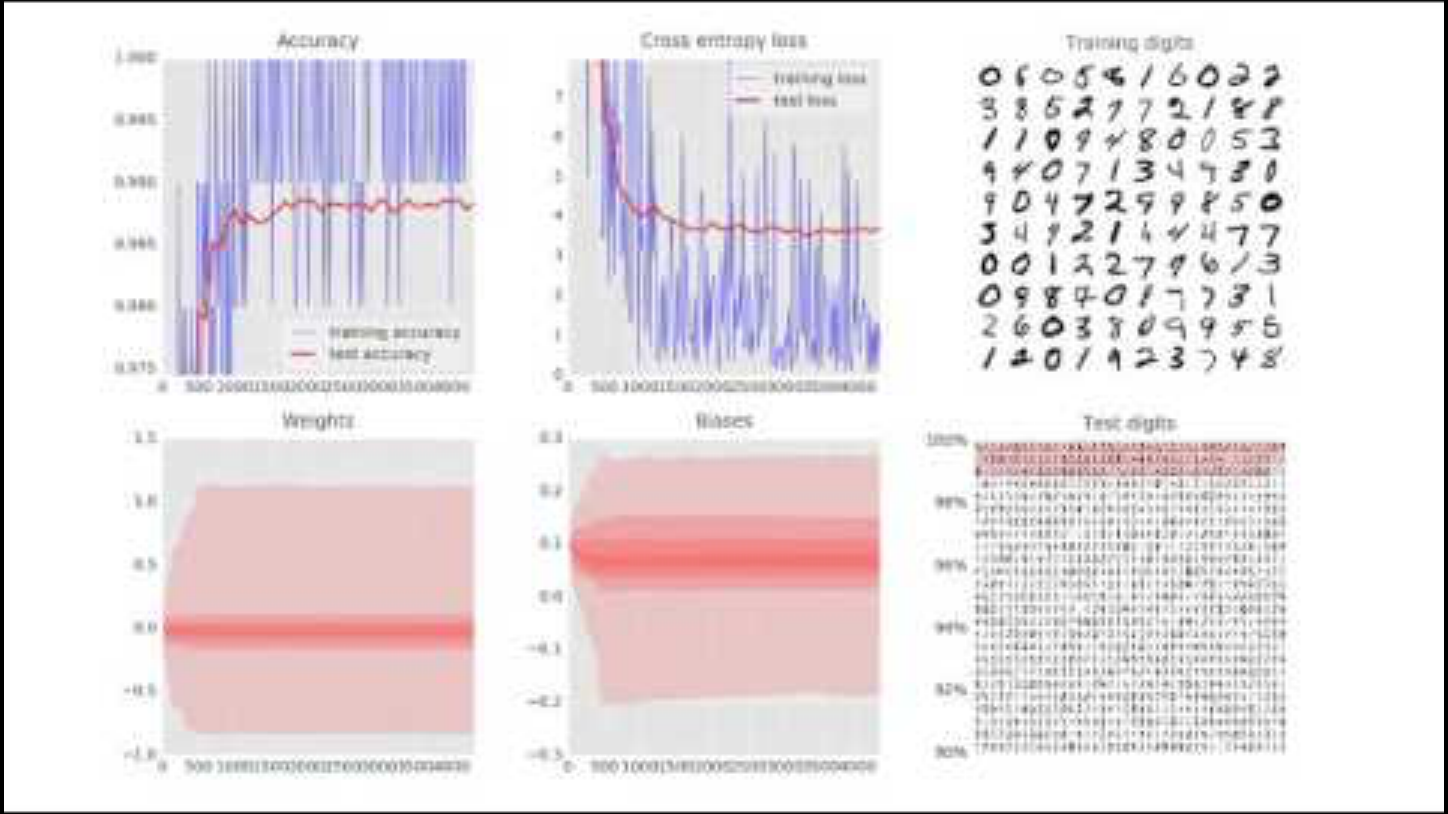
```
Y1 = tf.nn.relu(tf.nn.conv2d(X, W1, strides=[1, 1, 1, 1], padding='SAME') + B1)
Y2 = tf.nn.relu(tf.nn.conv2d(Y1, W2, strides=[1, 2, 2, 1], padding='SAME') + B2)
Y3 = tf.nn.relu(tf.nn.conv2d(Y2, W3, strides=[1, 2, 2, 1], padding='SAME') + B3)

YY = tf.reshape(Y3, shape=[-1, 7 * 7 * M])

Y4 = tf.nn.relu(tf.matmul(YY, W4) + B4)
Y  = tf.nn.softmax(tf.matmul(Y4, W5) + B5)
```

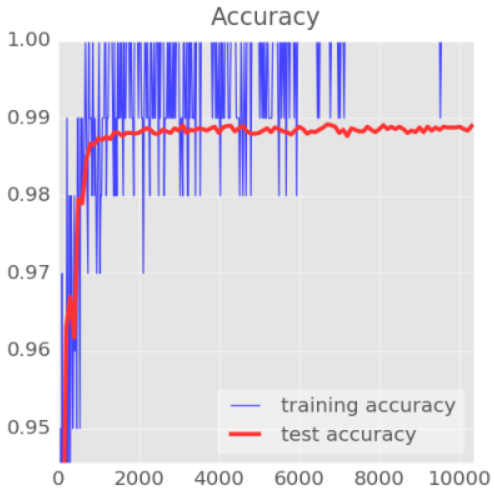
flatten all values for fully connected layer

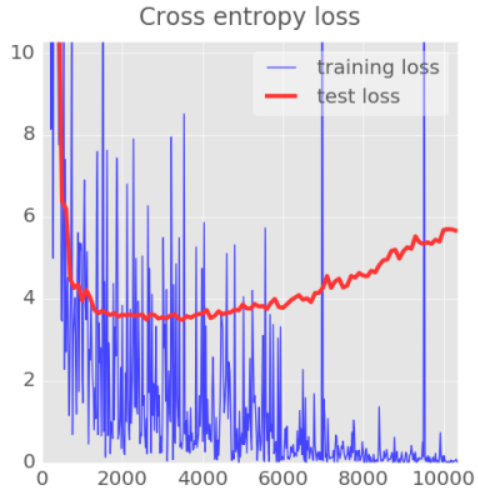
Y3 [100, 7, 7, 12]

YY [100, 7x7x12]

Demo

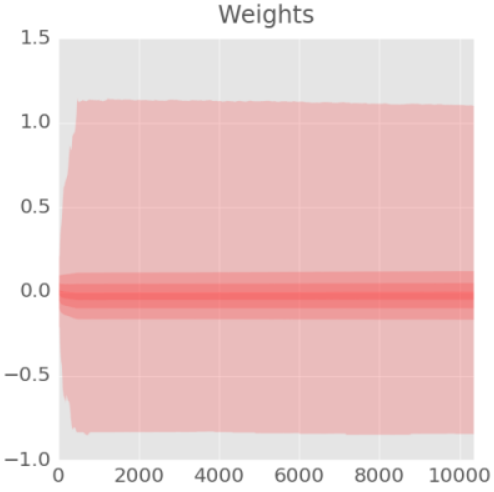

Accuracy
training accuracy
test accuracy
Cross entropy loss
training loss
test loss
Training digits
Weights
Biases
Test digits

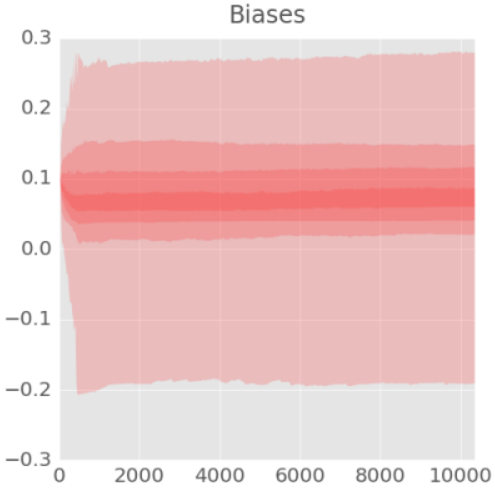

Accuracy
1.00
0.99
0.98
0.97
0.96
0.95
0
2000
4000
6000
8000
10000
training accuracy
test accuracy

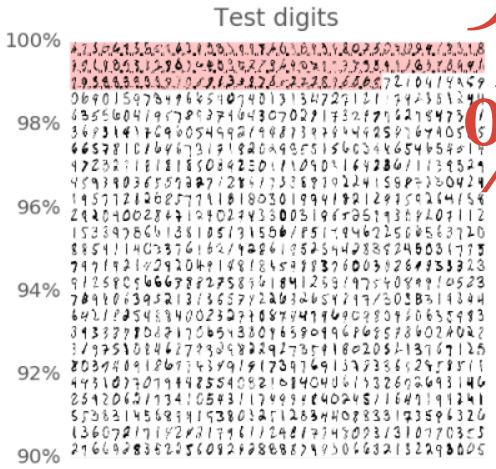

Cross entropy loss
10
8
6
4
2
0
2000
4000
6000
8000
10000
training loss
test loss


Training digits

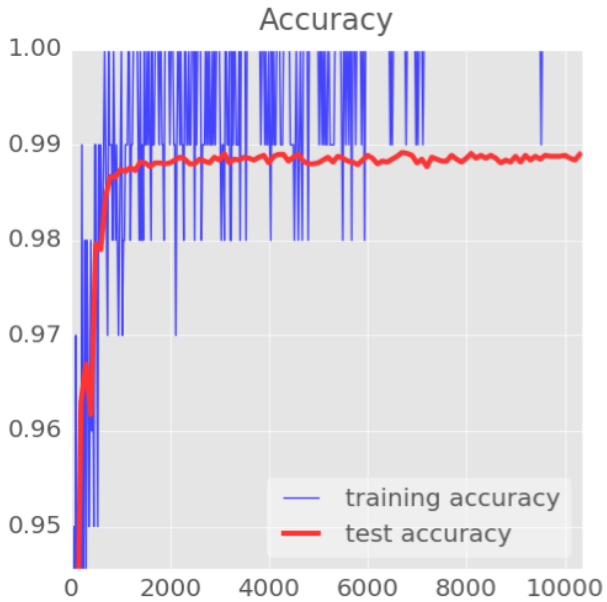
98.9%

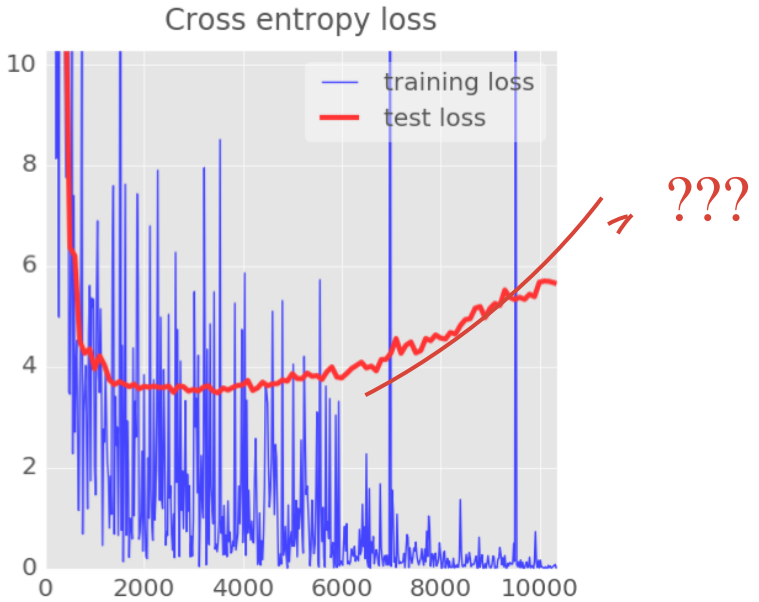

Weights
1.5
1.0
0.5
0.0
−0.5
−1.0
0
2000
4000
6000
8000
10000


Biases
0.3
0.2
0.1
0.0
−0.1
−0.2
−0.3
0
2000
4000
6000
8000
10000


Test digits
100%
98%
96%
94%
92%
90%

WTFH ???



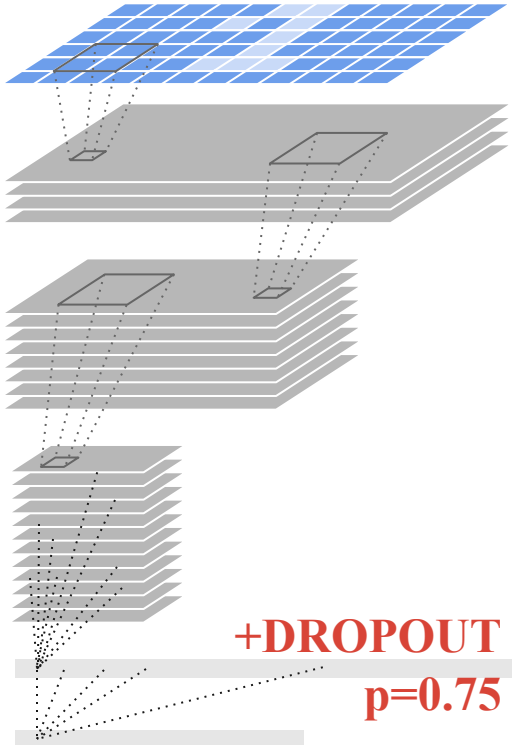


Google Cloud

@martin_gorner

Bigger convolutional network + dropout

+ biases on all layers



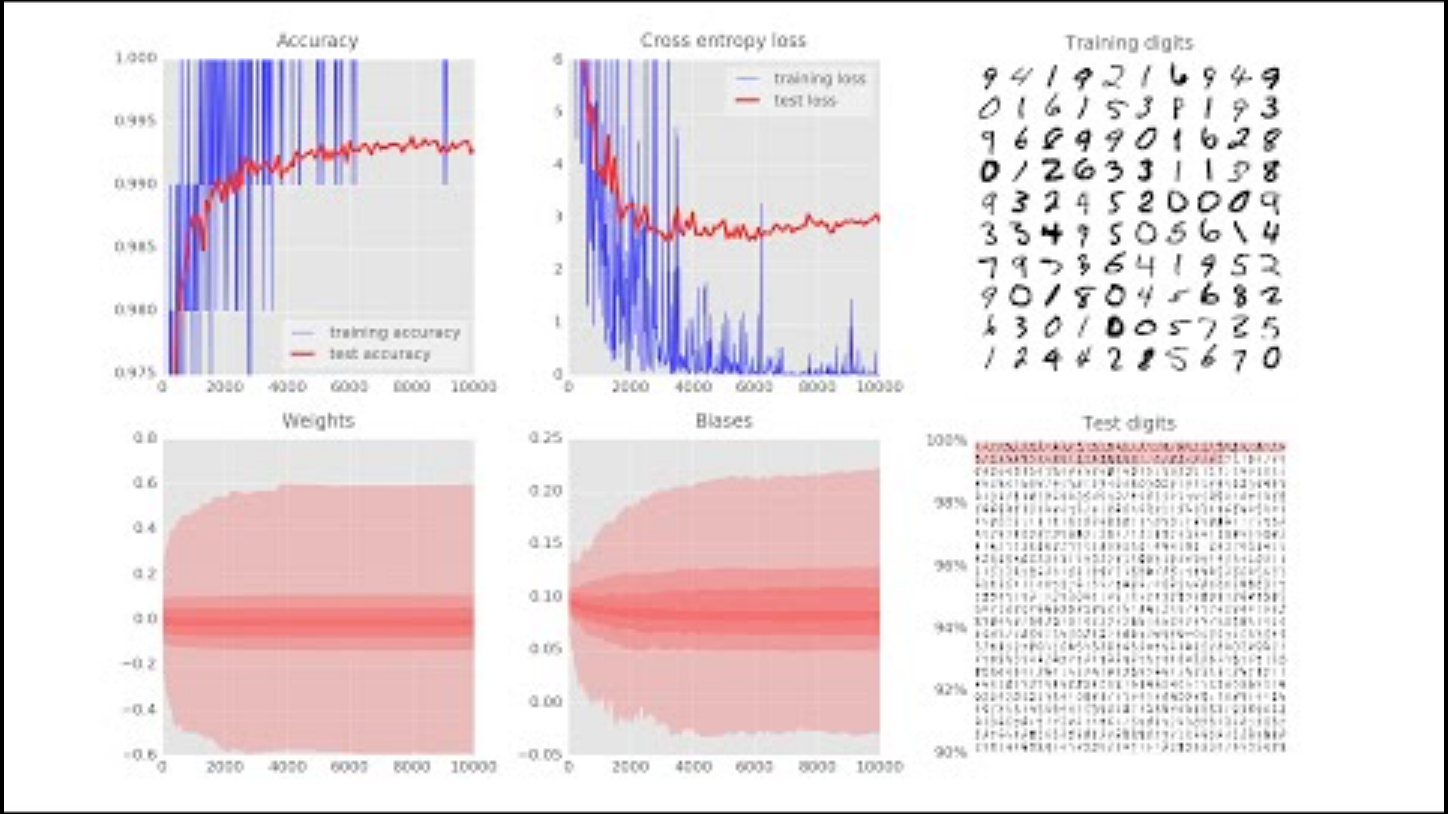
convolutional layer, **6 channels**
W1[**6, 6**, 1, 6] stride 1

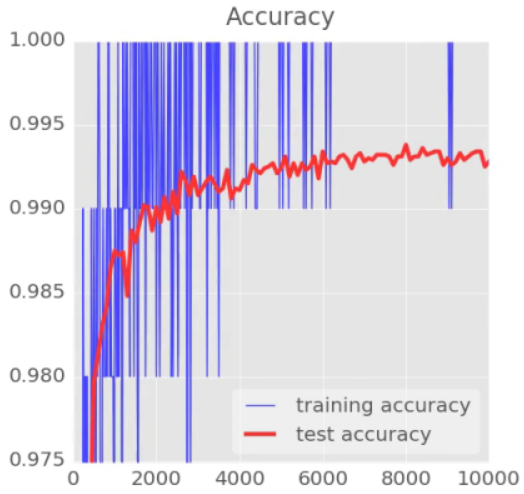
convolutional layer, **12 channels**
W2[**5, 5**, 6, 12] stride 2

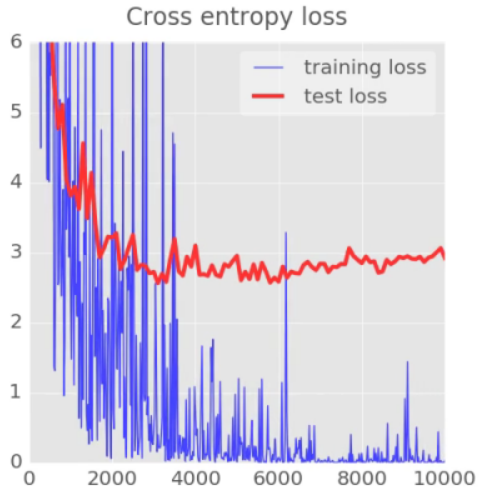
convolutional layer, **24 channels**
W3[**4, 4**, 12, 24] stride 2

fully connected layer W4[7x7x24, 200]
softmax readout layer W5[200, 10]

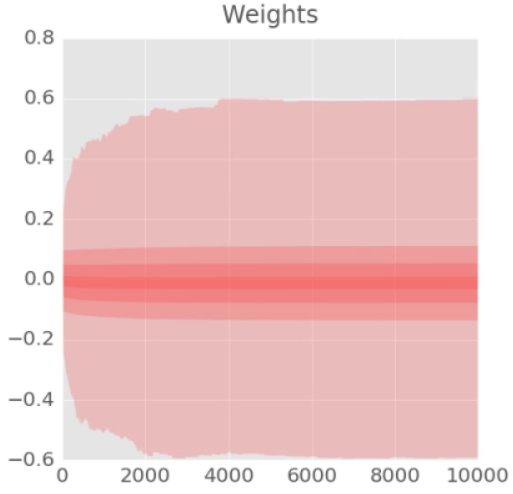
Demo

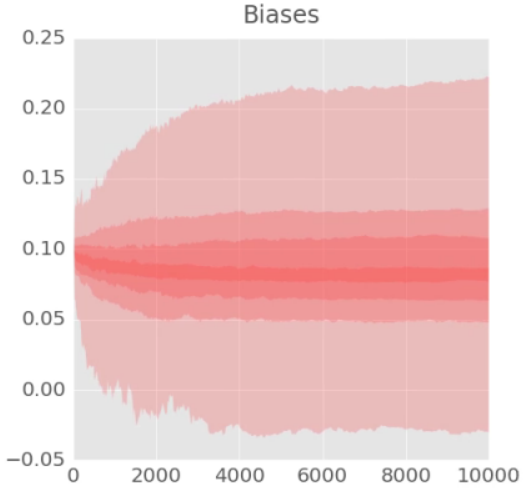

Accuracy
training accuracy
test accuracy
Cross entropy loss
training loss
test loss
Training digits
Weights
Biases
Test digits

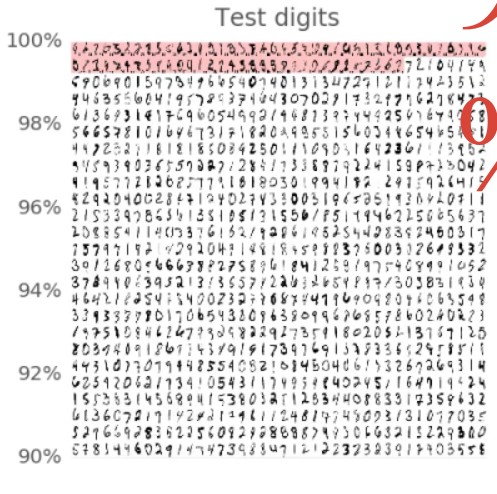

Accuracy
1.000
0.995
0.990
0.985
0.980
0.975
0
2000
4000
6000
8000
10000
training accuracy
test accuracy


Cross entropy loss
6
5
4
3
2
1
0
2000
4000
6000
8000
10000
training loss
test loss


Training digits

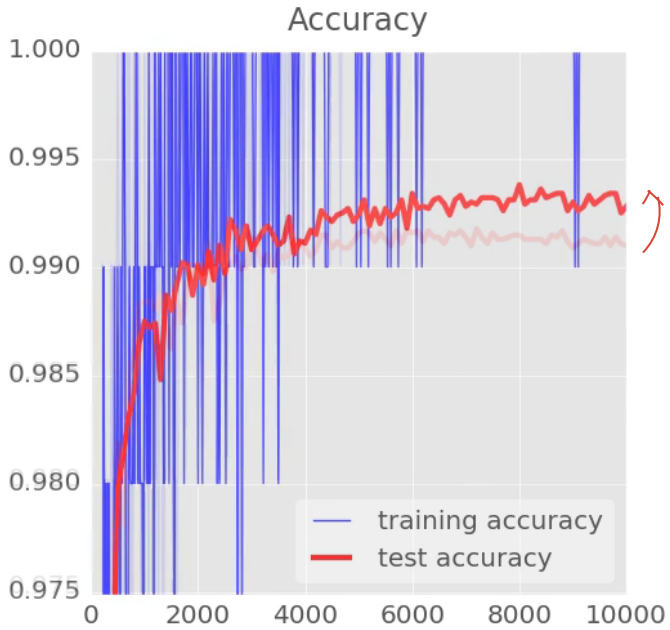

Weights
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
0
2000
4000
6000
8000
10000

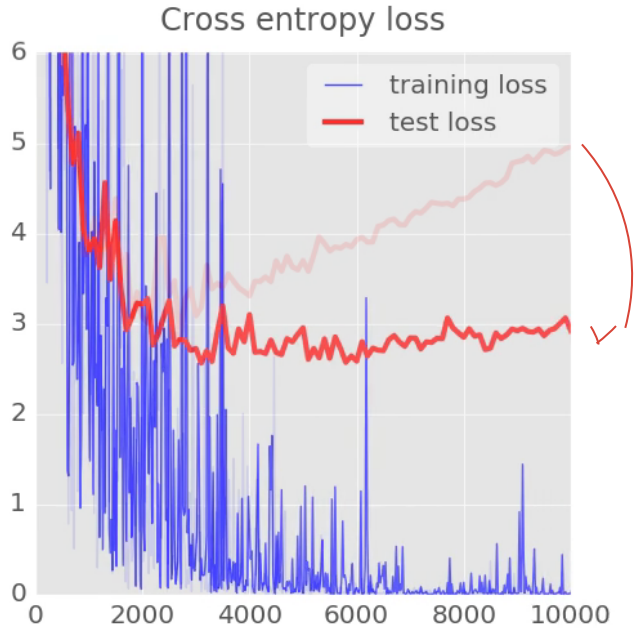

Biases
0.25
0.20
0.15
0.10
0.05
0.00
−0.05
0
2000
4000
6000
8000
10000


Test digits
100%
98%
96%
94%
92%
90%

99.3%

YEAH !



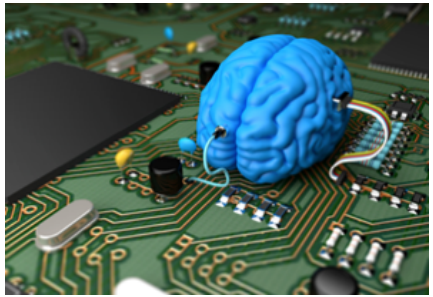


with dropout


THIS IS YOUR MACHINE LEARNING SYSTEM?
YUP! YOU POUR THE DATA INTO THIS BIG PILE OF LINEAR ALGEBRA, THEN COLLECT THE ANSWERS ON THE OTHER SIDE.
WHAT IF THE ANSWERS ARE WRONG?
JUST STIR THE PILE UNTIL THEY START LOOKING RIGHT.
DATA
ANSWERS

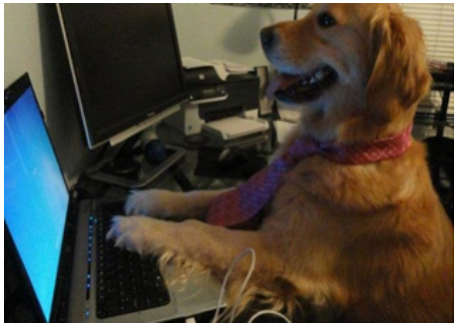
THANKS!

What society thinks I do



What my friends think I do

What other computer scientists think I do



What mathematicians think I do

What I think I do

```
from tensorflow import *
```

ML6

What I actually do

Where to start?

http://bit.ly/2yg1nD0

Tensorflow

tensorflow.org

ML6

ml6.eu

Developer codelabs

codelabs.developers.google.com

GitHub

Find Examples

github.com/tensorflow