

Social Computing and Big Data Analytics

社群運算與大數據分析

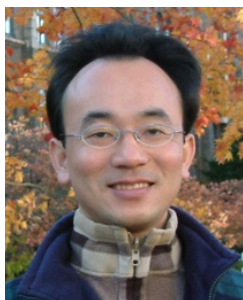
Deep Learning with Google TensorFlow

(Google TensorFlow 深度學習)

1052SCBDA10

MIS MBA (M2226) (8606)

Wed, 8,9, (15:10-17:00) (L206)



Min-Yuh Day

戴敏育

Assistant Professor

專任助理教授

Dept. of Information Management, Tamkang University

淡江大學 資訊管理學系

<http://mail.tku.edu.tw/myday/>

2017-05-03



課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
1	2017/02/15	Course Orientation for Social Computing and Big Data Analytics (社群運算與大數據分析課程介紹)
2	2017/02/22	Data Science and Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data (資料科學與大數據分析： 探索、分析、視覺化與呈現資料)
3	2017/03/01	Fundamental Big Data: MapReduce Paradigm, Hadoop and Spark Ecosystem (大數據基礎：MapReduce典範、 Hadoop與Spark生態系統)

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
4	2017/03/08	Big Data Processing Platforms with SMACK: Spark, Mesos, Akka, Cassandra and Kafka (大數據處理平台SMACK： Spark, Mesos, Akka, Cassandra, Kafka)
5	2017/03/15	Big Data Analytics with Numpy in Python (Python Numpy 大數據分析)
6	2017/03/22	Finance Big Data Analytics with Pandas in Python (Python Pandas 財務大數據分析)
7	2017/03/29	Text Mining Techniques and Natural Language Processing (文字探勘分析技術與自然語言處理)
8	2017/04/05	Off-campus study (教學行政觀摩日)

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
9	2017/04/12	Social Media Marketing Analytics (社群媒體行銷分析)
10	2017/04/19	期中報告 (Midterm Project Report)
11	2017/04/26	Deep Learning with Theano and Keras in Python (Python Theano 和 Keras 深度學習)
12	2017/05/03	Deep Learning with Google TensorFlow (Google TensorFlow 深度學習)
13	2017/05/10	Sentiment Analysis on Social Media with Deep Learning (深度學習社群媒體情感分析)

課程大綱 (Syllabus)

週次 (Week)	日期 (Date)	內容 (Subject/Topics)
14	2017/05/17	Social Network Analysis (社會網絡分析)
15	2017/05/24	Measurements of Social Network (社會網絡量測)
16	2017/05/31	Tools of Social Network Analysis (社會網絡分析工具)
17	2017/06/07	Final Project Presentation I (期末報告 I)
18	2017/06/14	Final Project Presentation II (期末報告 II)

Deep Learning with Google TensorFlow

Deep Learning Libraries: Tensorflow and Keras

Deep learning libraries: GitHub activity from February 11 to April 12, 2017

new contributors from 2017-02-11 to 2017-04-12			new forks from 2017-02-11 to 2017-04-12		
#1:	131	tensorflow/tensorflow	#1:	4192	tensorflow/tensorflow
#2:	63	fchollet/keras	#2:	991	fchollet/keras
#3:	51	pytorch/pytorch	#3:	810	BVLC/caffe
#4:	49	dmlc/mxnet	#4:	517	deeplearning4j/deeplearning4j
#5:	18	Theano/Theano	#5:	414	dmlc/mxnet
#6:	11	BVLC/caffe	#6:	307	pytorch/pytorch
#7:	11	Microsoft/CNTK	#7:	244	Microsoft/CNTK
#8:	9	tflearn/tflearn	#8:	211	tflearn/tflearn
#9:	9	pfnet/chainer	#9:	134	torch/torch7
#10:	8	torch/torch7	#10:	131	Theano/Theano
#11:	5	deeplearning4j/deeplearning4j	#11:	116	baidu/paddle
#12:	4	NVIDIA/DIGITS	#12:	88	NVIDIA/DIGITS
#13:	3	baidu/paddle	#13:	55	pfnet/chainer

new issues from 2017-02-11 to 2017-04-12			aggregate activity from 2017-02-11 to 2017-04-12		
#1:	1175	tensorflow/tensorflow	#1:	36.64	tensorflow/tensorflow
#2:	568	fchollet/keras	#2:	12.52	fchollet/keras
#3:	499	dmlc/mxnet	#3:	8.53	dmlc/mxnet
#4:	286	pytorch/pytorch	#4:	6.09	BVLC/caffe
#5:	257	Microsoft/CNTK	#5:	5.92	pytorch/pytorch
#6:	239	deeplearning4j/deeplearning4j	#6:	5.12	deeplearning4j/deeplearning4j
#7:	219	baidu/paddle	#7:	4.12	Microsoft/CNTK
#8:	173	Theano/Theano	#8:	2.93	Theano/Theano
#9:	171	BVLC/caffe	#9:	2.86	baidu/paddle
#10:	112	NVIDIA/DIGITS	#10:	2.17	tflearn/tflearn
#11:	84	tflearn/tflearn	#11:	1.68	NVIDIA/DIGITS
#12:	57	pfnet/chainer	#12:	1.38	torch/torch7
#13:	47	torch/torch7	#13:	1.12	pfnet/chainer



TensorFlow

Google TensorFlow

TensorFlow™

GET STARTED TUTORIALS HOW TO API RESOURCES ABOUT

Fork me on GitHub

TensorFlow is an Open Source Software
Library for Machine Intelligence

GET STARTED

About TensorFlow

TensorFlow™ is an open source software library for numerical computation using data flow graphs. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) communicated between them. The flexible architecture allows you to deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device with a single API.



<https://www.tensorflow.org/>

TensorFlow
is an
Open Source
Software Library
for
Machine Intelligence

numerical computation using data flow graphs

Tensor

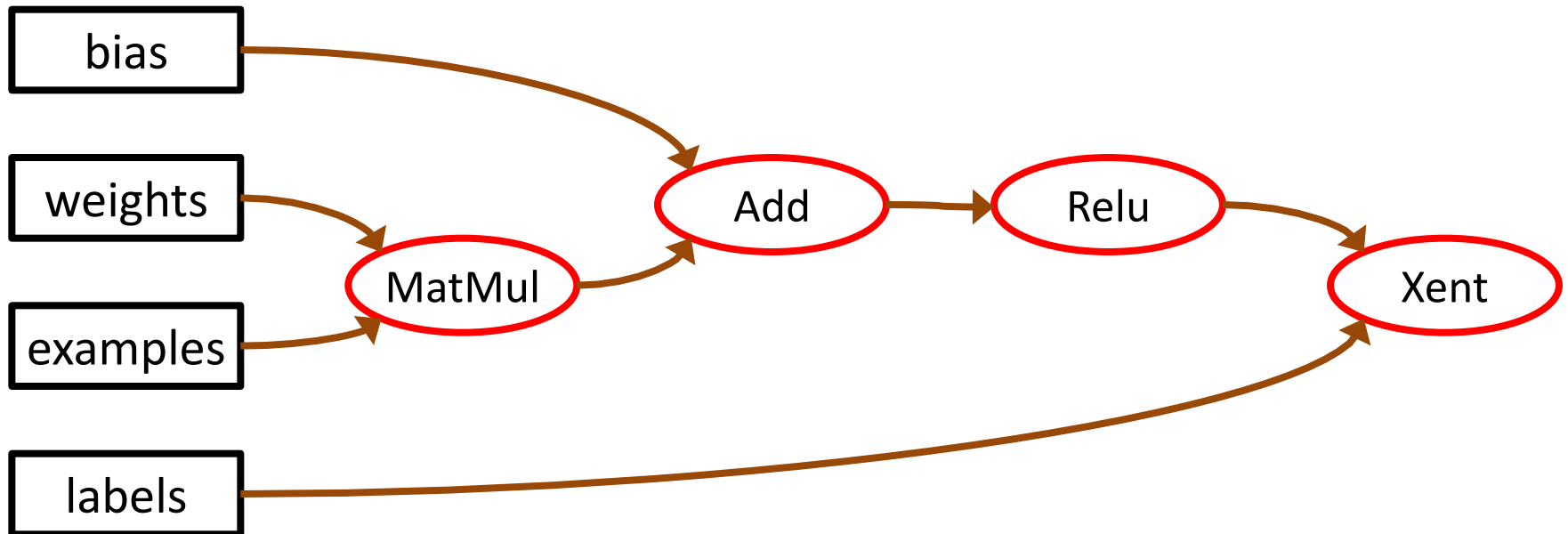
- 3
 - # a rank 0 tensor; this is a **scalar** with shape []
- [1., 2., 3.]
 - # a rank 1 tensor; this is a **vector** with shape [3]
- [[1., 2., 3.], [4., 5., 6.]]
 - # a rank 2 tensor; a **matrix** with shape [2, 3]
- [[[1., 2., 3.]], [[7., 8., 9.]]]
 - # a rank 3 **tensor** with shape [2, 1, 3]

Nodes:
mathematical operations

edges:
multidimensional data arrays
(tensors)
communicated between nodes

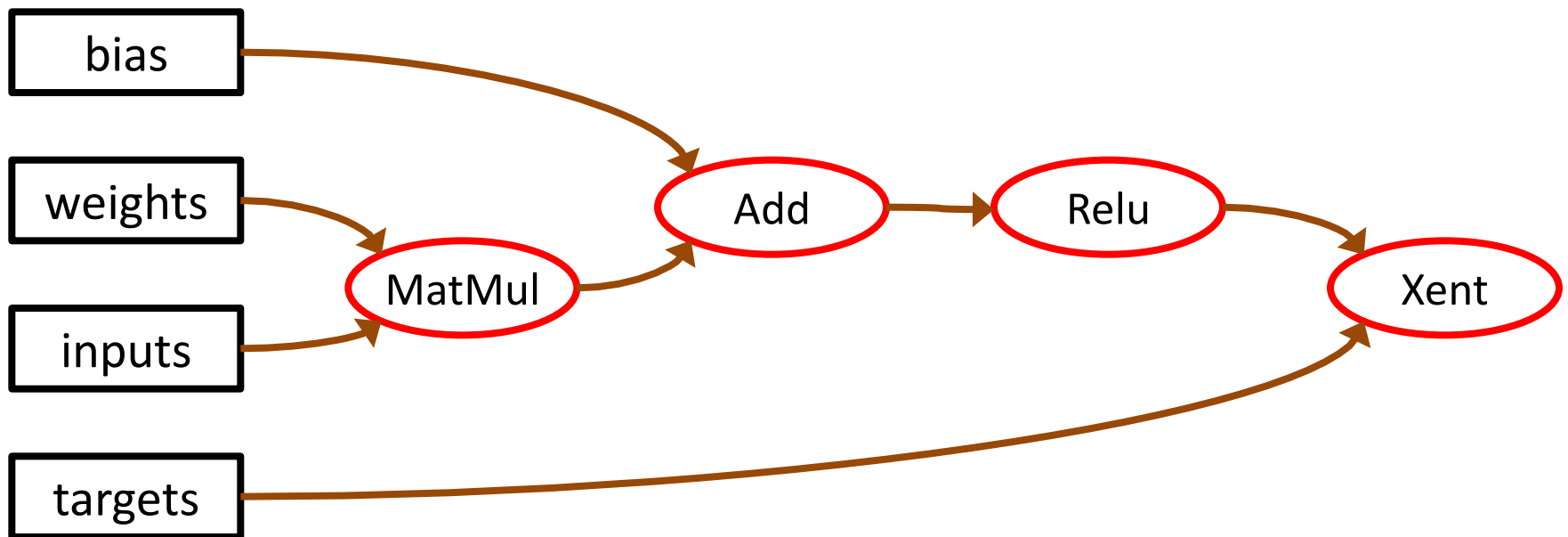
Computation is a Dataflow Graph

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

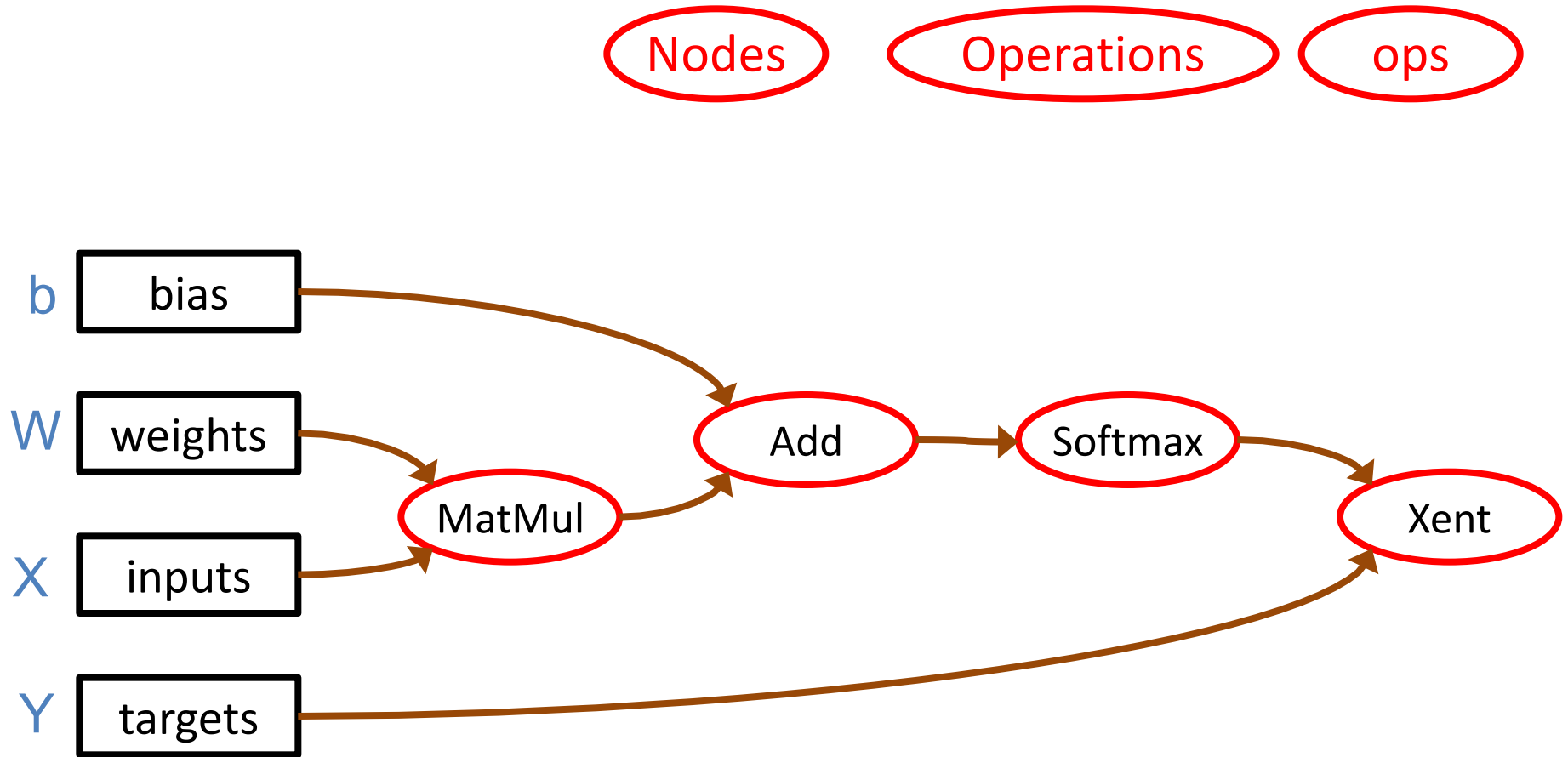


Computation is a Dataflow Graph

Edges are N-dimensional arrays: **Tensors**



Logistic Regression as Dataflow Graph

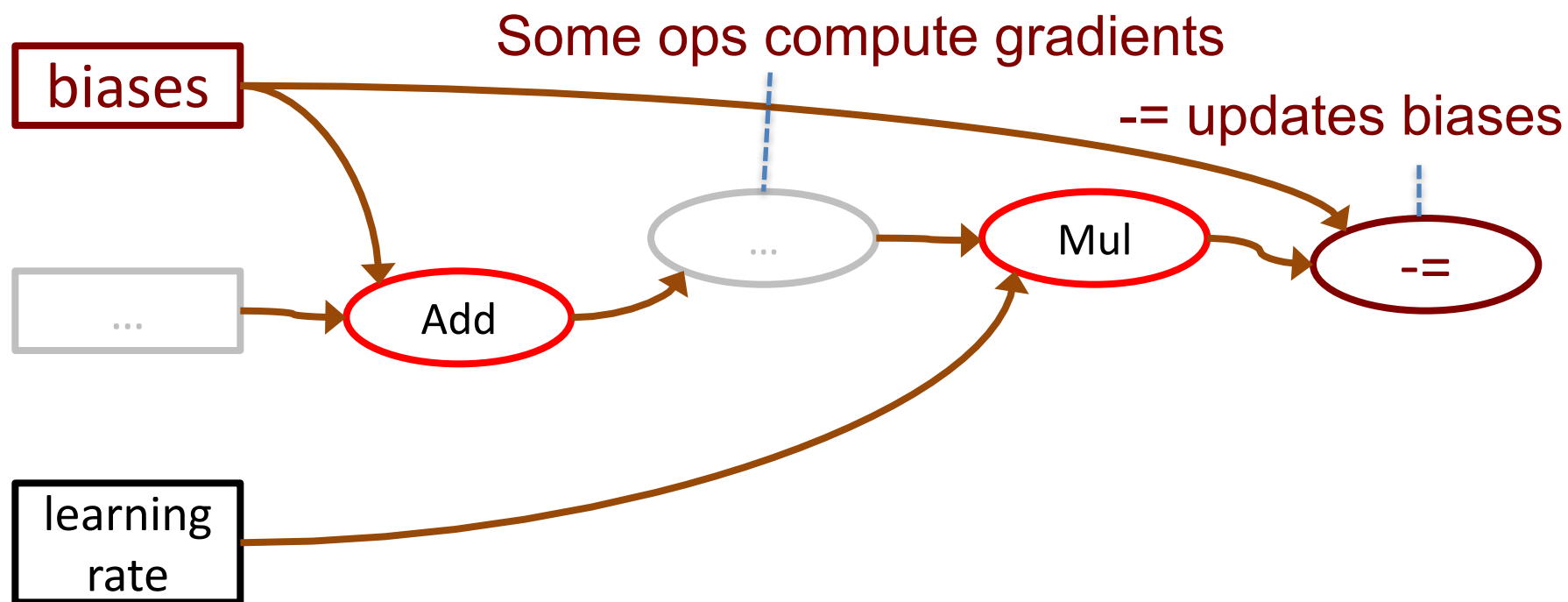


Edges are N-dimensional arrays: **Tensors**

Computation is a Dataflow Graph

with state

'Biases' is a variable

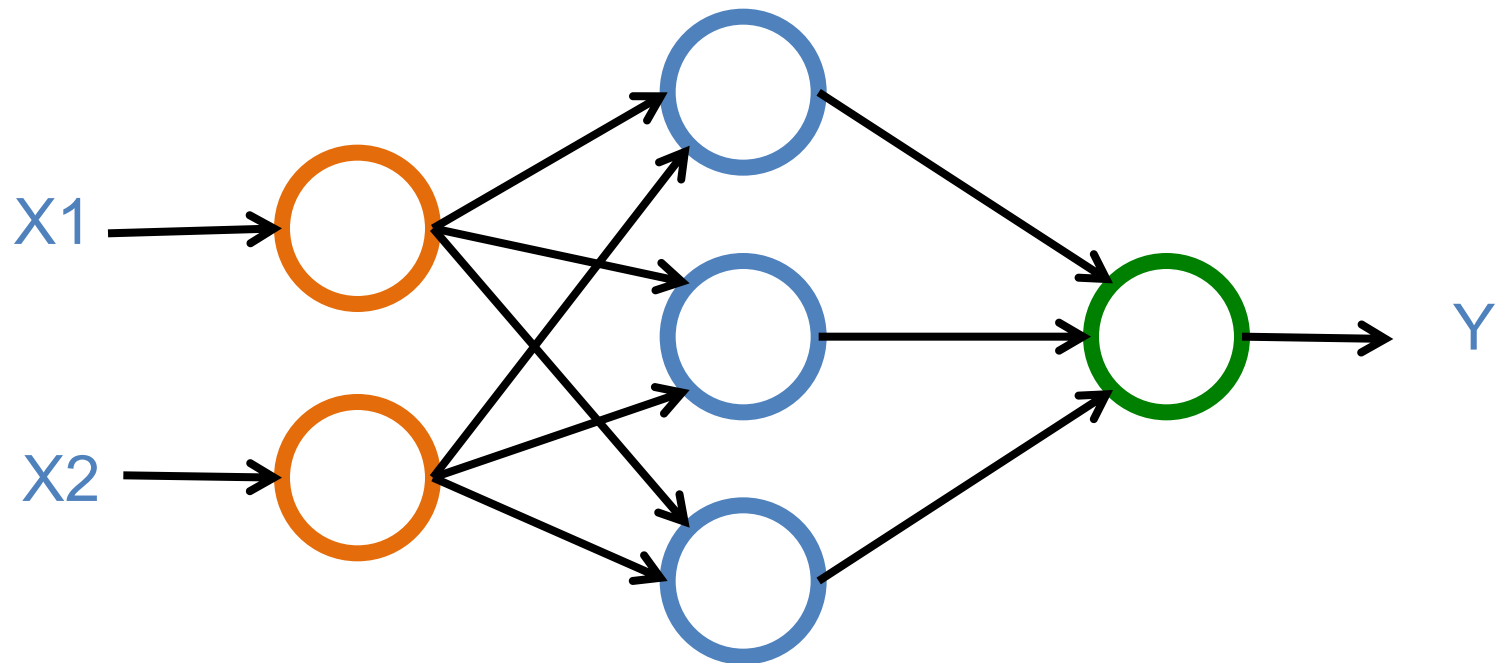


Neural Networks

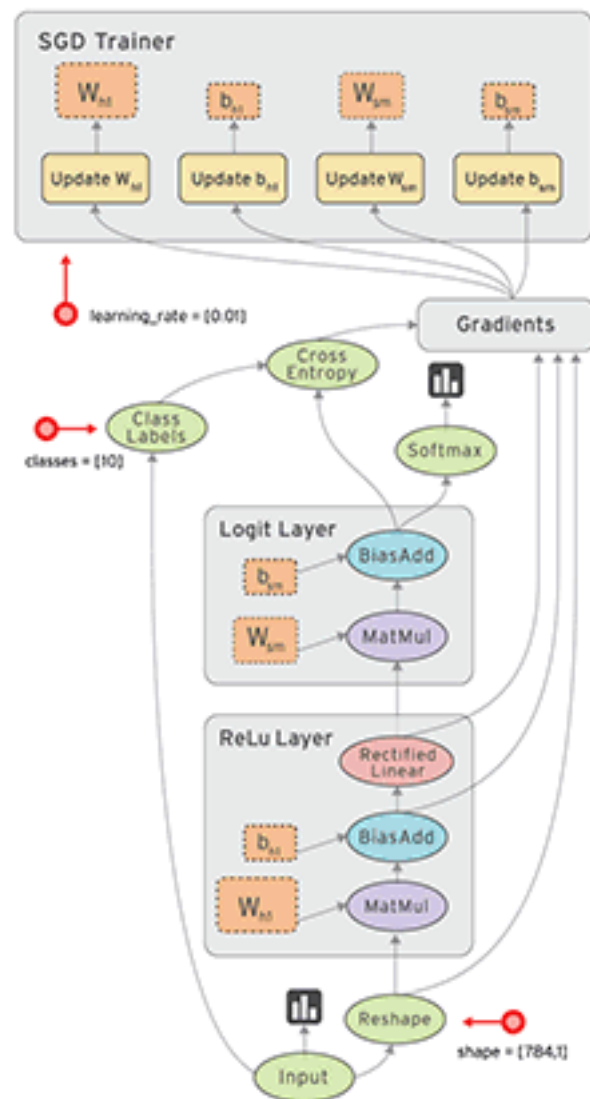
Input Layer
(X)

Hidden Layer
(H)

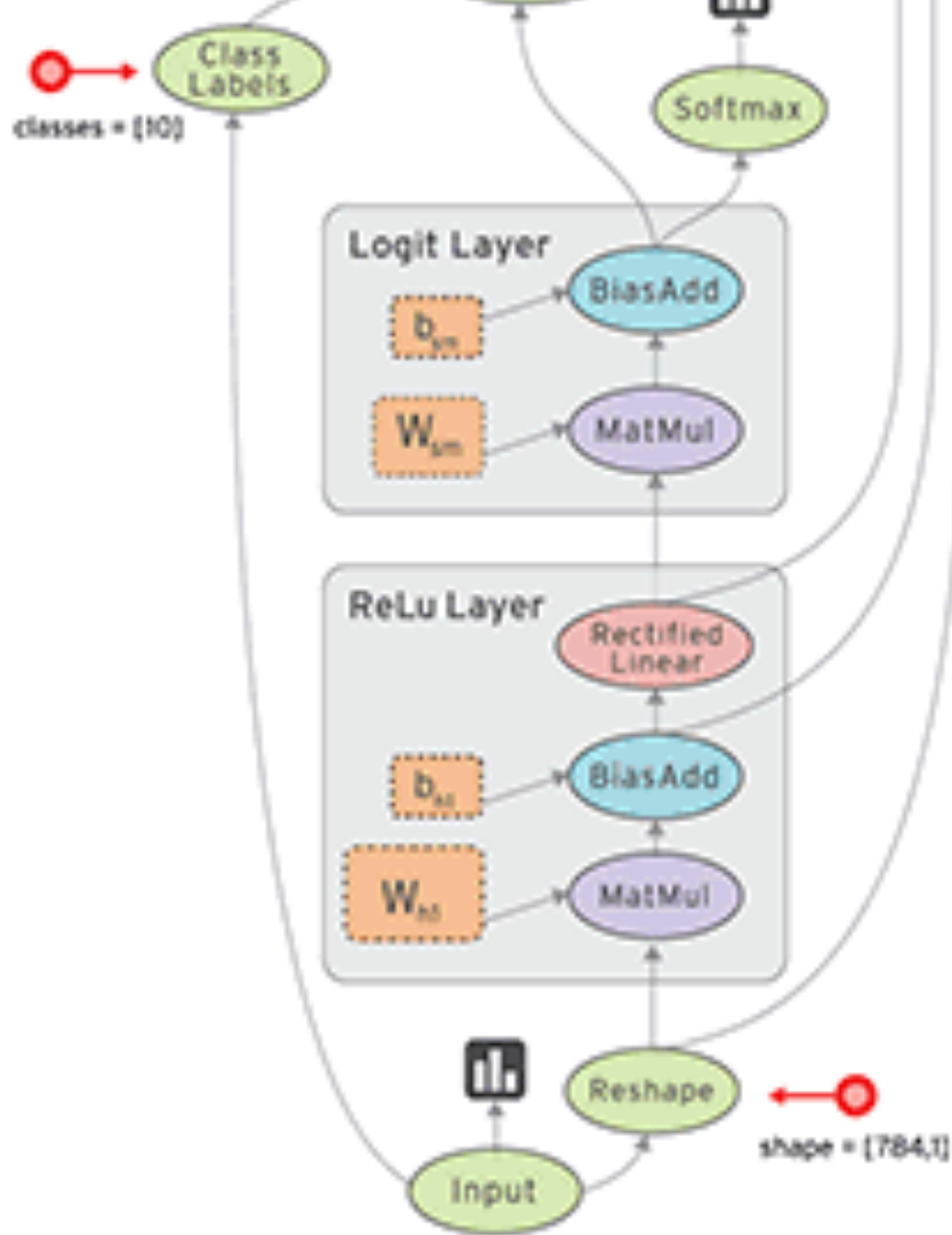
Output Layer
(Y)



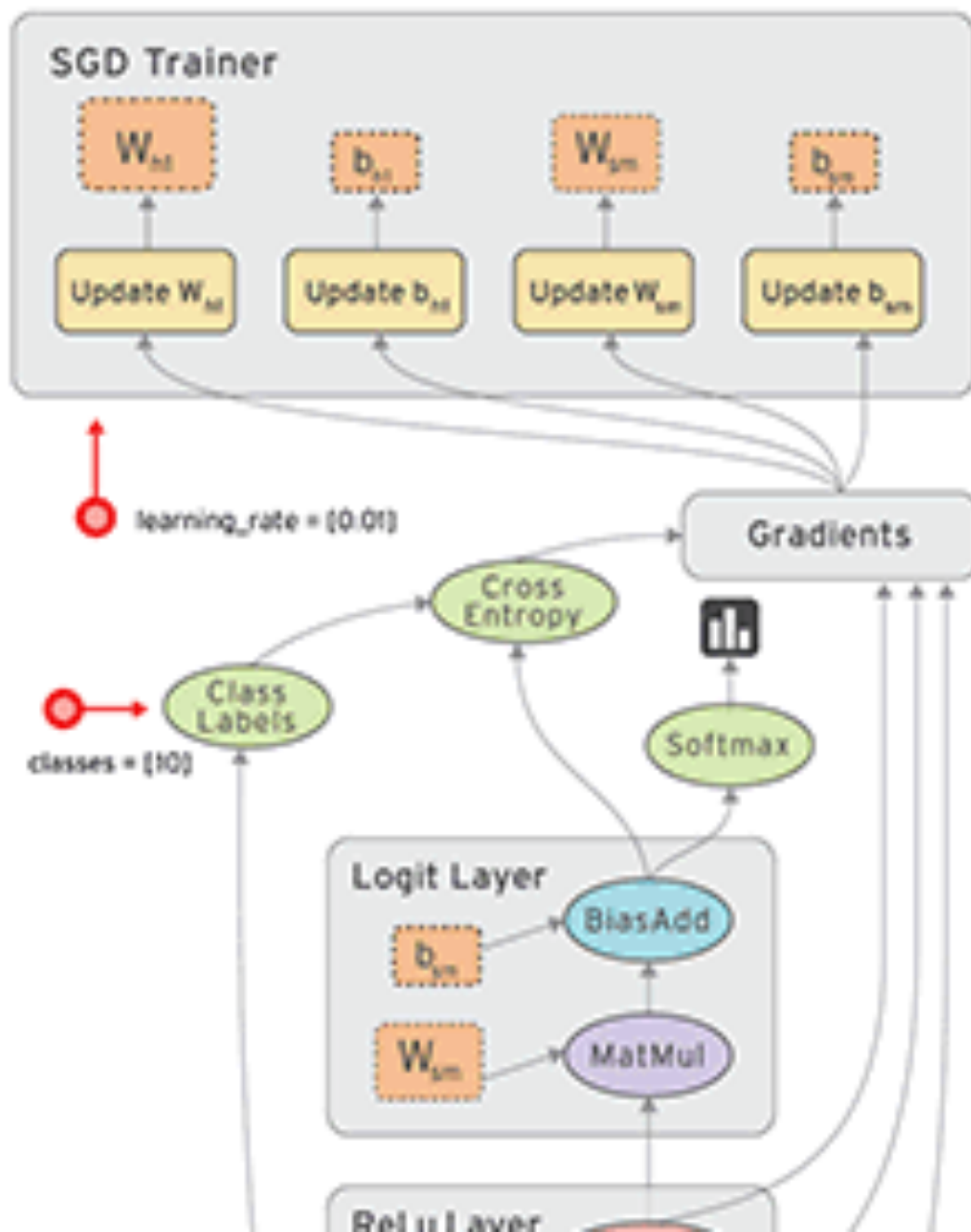
Data Flow Graph



Data Flow Graph



Data Flow Graph



TensorFlow Playground

Tinker With a **Neural Network** Right Here in Your Browser.
Don't Worry, You Can't Break It. We Promise.



Iterations
000,582

Learning rate

0.03

Activation

Tanh

Regularization

None

Regularization rate

0

Problem type

Classification

DATA

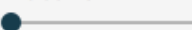
Which dataset do you want to use?



Ratio of training to test data: 50%



Noise: 0



Batch size: 10



INPUT

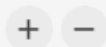
Which properties do you want to feed in?



3 HIDDEN LAYERS



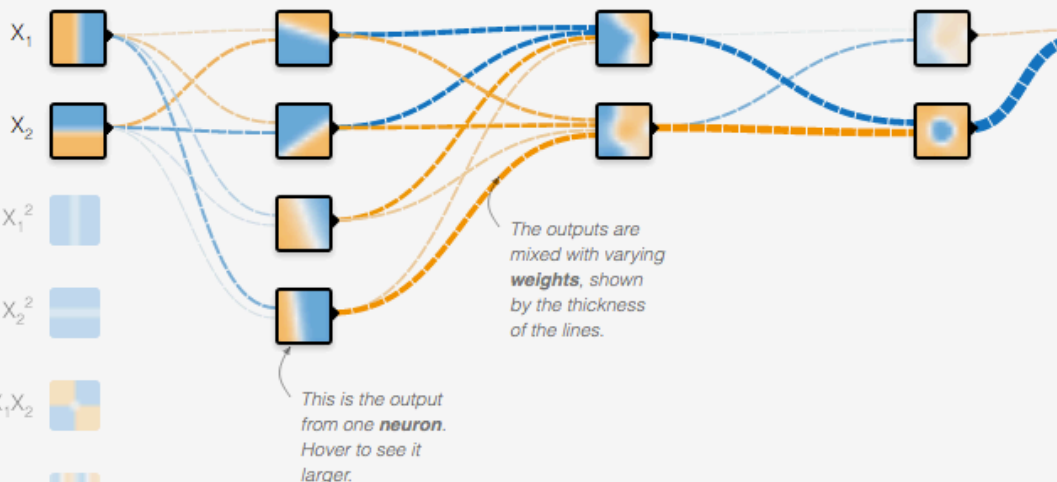
4 neurons



2 neurons



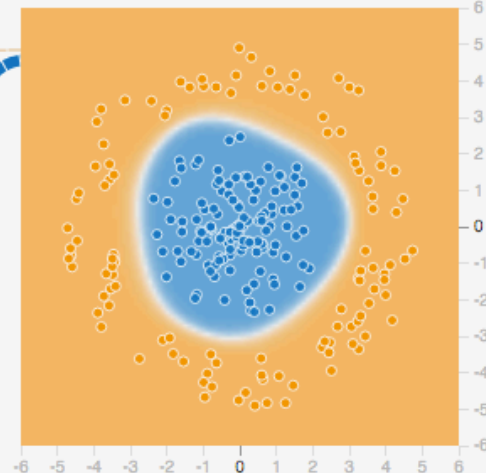
2 neurons



OUTPUT

Test loss 0.000

Training loss 0.000



TensorBoard

TensorBoard

EVENTS

IMAGES

GRAPHS

HISTOGRAMS



Fit to screen



Download PNG

Run train
(1)

Session runs (0)

Upload

Color ☒ Structure
☐ Device
color: same substructure
gray: unique substructure

Graph (* = expandable)



Namespace*



OpNode



Unconnected series*



Connected series*



Constant



Summary



Dataflow edge

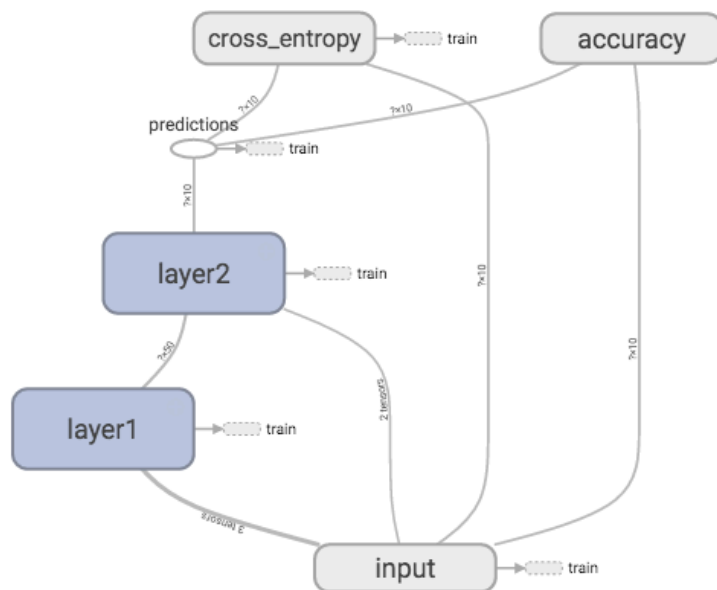


Control dependency edge

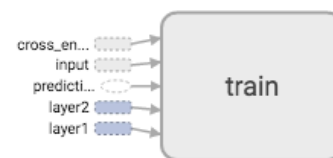


Reference edge

Main Graph



Auxiliary nodes



TensorFlow and Deep Learning

TensorFlow and Deep Learning

1 Overview

Preparation: Install

2 TensorFlow, get the sample code

3 Theory: train a neural network

4 Theory: a 1-layer neural network

5 Theory: gradient descent

6 Lab: let's jump into the code

7 Lab: adding layers

8 Lab: special care for deep networks

9 Lab: learning rate decay

10 Lab: dropout, overfitting

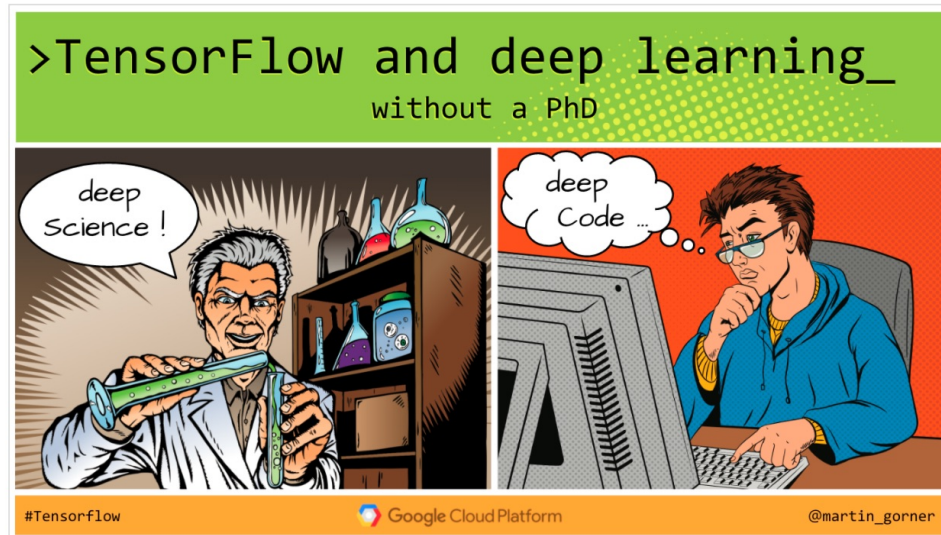
11 Theory: convolutional networks

Did you find a mistake? [Please file a bug.](#)

← TensorFlow and deep learning, without a PhD

🕒 149 min remaining

1. Overview



In this codelab, you will learn how to build and train a neural network that recognises handwritten digits. Along the way, as you enhance your neural network to achieve 99% accuracy, you will also discover the tools of the trade that deep learning professionals use to train their models efficiently.

This codelab uses the [MNIST](#) dataset, a collection of 60,000 labeled digits that has kept generations of PhDs busy for almost two decades. You will solve the problem with less than 100 lines of Python / TensorFlow code.

What you'll learn

TensorFlow MNIST Tutorial



Features Business Explore Pricing

This repository

Search

[Sign in](#) or [Sign up](#)

 [martin-gorner](#) / [tensorflow-mnist-tutorial](#)

 Watch

50

 Star

489

 Fork

204

 Code

 Issues **6**

 Pull requests **2**

 Projects **0**

 Pulse

 Graphs

Sample code for "Tensorflow and deep learning, without a PhD" presentation and code lab.

 **102** commits

 **1** branch

 **0** releases

 **4** contributors

 Apache-2.0

Branch: **master** ▾

[New pull request](#)

[Find file](#)

[Clone or download ▾](#)



martin-gorner committed on **GitHub** Update INSTALL.txt ...

Latest commit ed331aa 25 days ago

 mlengine	added example using the Tensorflow high level layers API	26 days ago
 .gitignore	small bug fix in batch norm	6 months ago
 CONTRIBUTING.md	initial commit 2	4 months ago
 INSTALL.txt	Update INSTALL.txt	25 days ago
 LICENSE	Initial commit	a year ago
 README.md	better image URL	3 months ago
 mnist_1.0_softmax.py	global_variables_initializer used everywhere instead of initalize_al...	2 months ago
 mnist_2.0_five_layers_sigmoid.py	Fix spacing in the network structure comment	a month ago
 mnist_2.1_five_layers_relu_lrdecay...	Fix spacing in the network structure comment	a month ago

TensorFlow and Deep Learning

- What is a neural network and how to train it
- How to build a basic 1-layer neural network using TensorFlow
- How to add more layers
- Training tips and tricks: overfitting, dropout, learning rate decay ...
- How to troubleshoot deep neural networks
- How to build convolutional networks

TensorFlow


```
conda info --envs
```

```
conda --version
```

```
python --version
```

```
conda list
```

```
conda create -n tensorflow python=3.5
```

```
source activate tensorflow
```

```
activate tensorflow
```

```
pip install Theano
```

```
conda install pydot
```

```
sudo pip install keras
```

```
pip install keras
```

```
pip install tensorflow
```

```
pip install ipython[all]
```

`pip install tensorflow`

```
bash-3.2$ pip install tensorflow
Collecting tensorflow
  Downloading tensorflow-1.1.0-cp36-cp36m-macosx_10_11_x86_64.whl (31.3MB)
    100% |████████████████████████████████████████| 31.3MB 23kB/s
Requirement already satisfied: wheel>=0.26 in ./anaconda/lib/python3.6/site-packages (from tensorflow)
Requirement already satisfied: six>=1.10.0 in ./anaconda/lib/python3.6/site-packages (from tensorflow)
Collecting protobuf>=3.2.0 (from tensorflow)
  Downloading protobuf-3.2.0-py2.py3-none-any.whl (360kB)
    100% |████████████████████████████████████████| 368kB 453kB/s
Requirement already satisfied: werkzeug>=0.11.10 in ./anaconda/lib/python3.6/site-packages (from tensorflow)
Requirement already satisfied: numpy>=1.11.0 in ./anaconda/lib/python3.6/site-packages (from tensorflow)
Requirement already satisfied: setuptools in ./anaconda/lib/python3.6/site-packages/setuptools-27.2.0-py3.6.egg (from protobuf>=3.2.0->tensorflow)
Installing collected packages: protobuf, tensorflow
Successfully installed protobuf-3.2.0 tensorflow-1.1.0
bash-3.2$
```

TensorFlow MNIST Tutorial

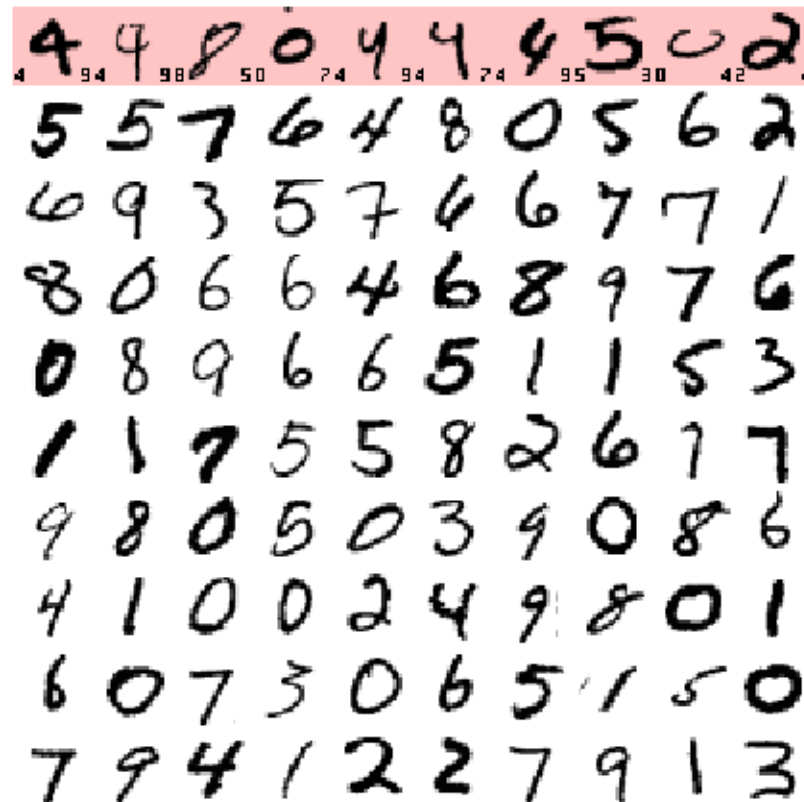
```
git clone https://github.com/martin-gorner/tensorflow-mnist-tutorial.git
```

```
cd tensorflow-mnist-tutorial
```

```
python3 mnist_1.0_softmax.py
```

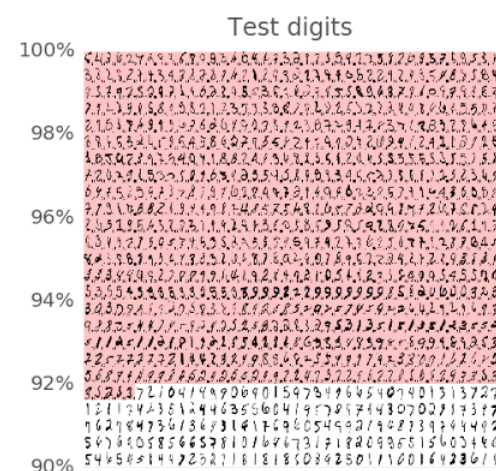
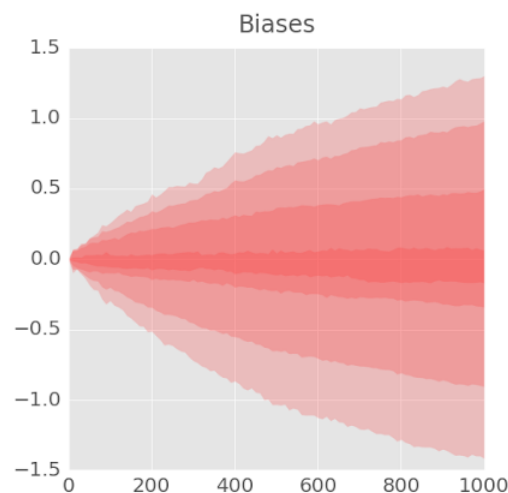
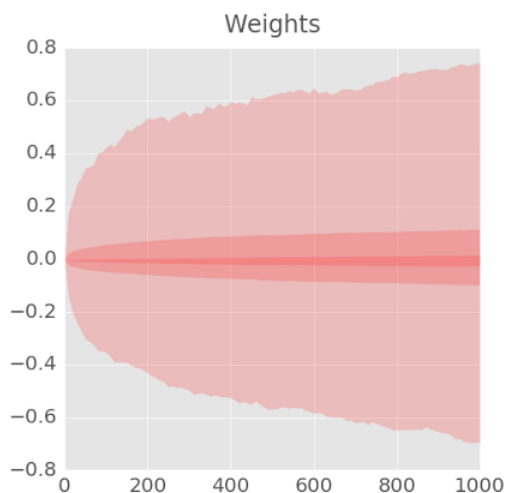
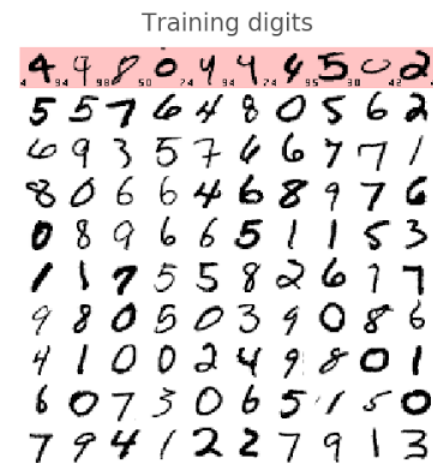
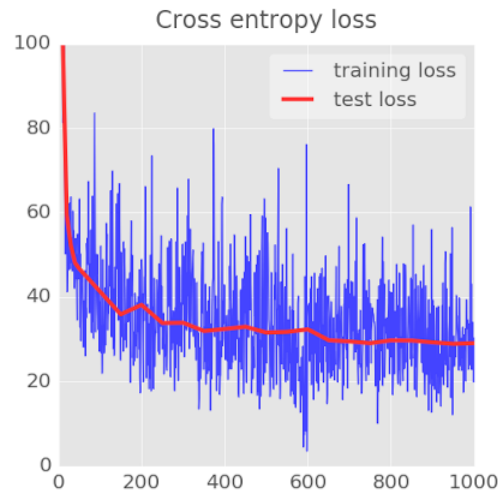
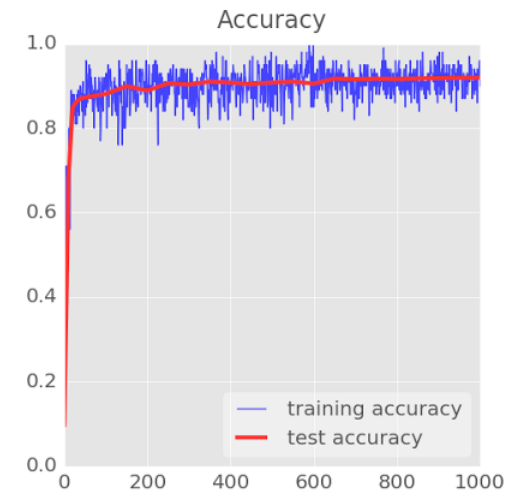
MNIST dataset: 60,000 labeled digits

Training digits



cd tensorflow-mnist-tutorial

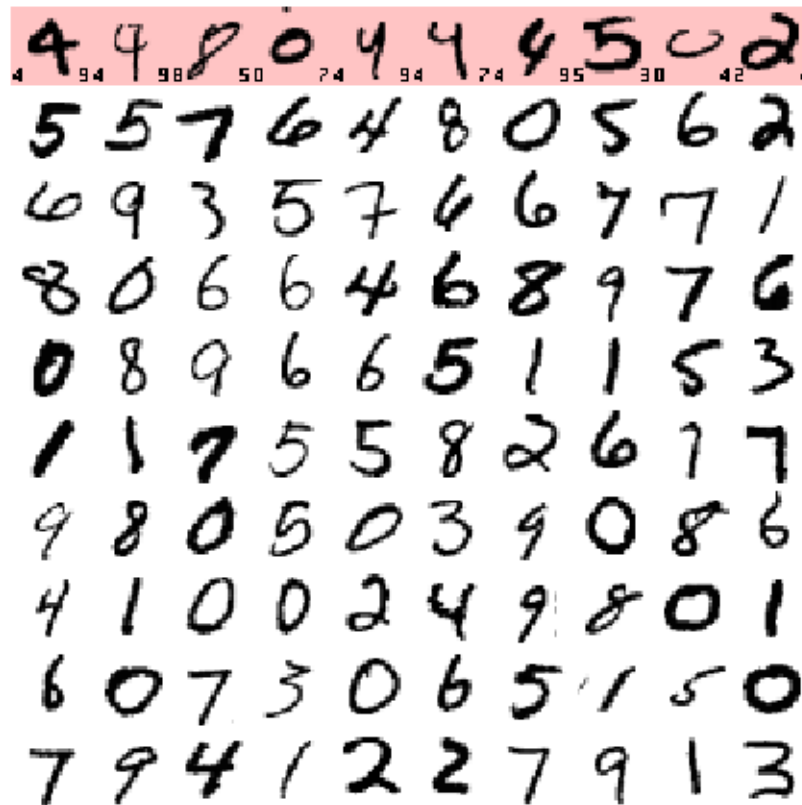
python3 mnist_1.0_softmax.py

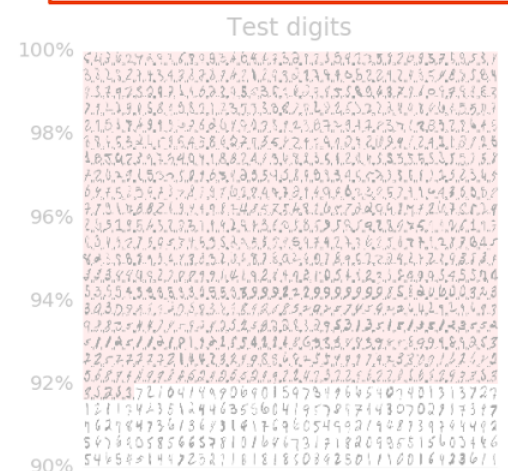
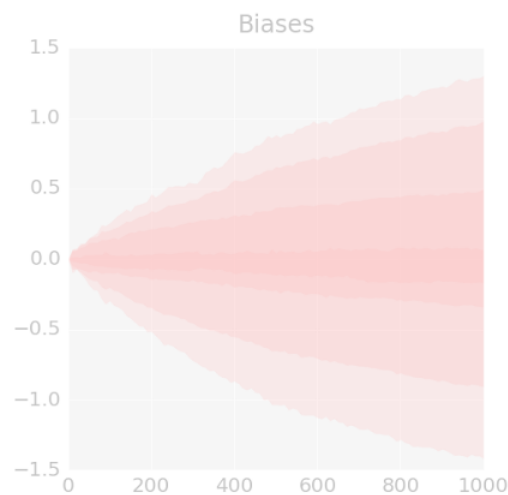
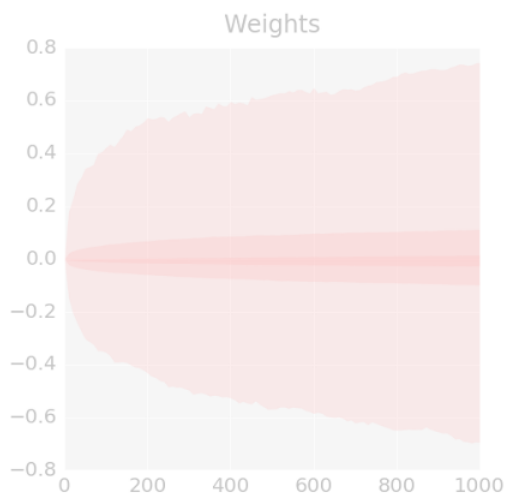
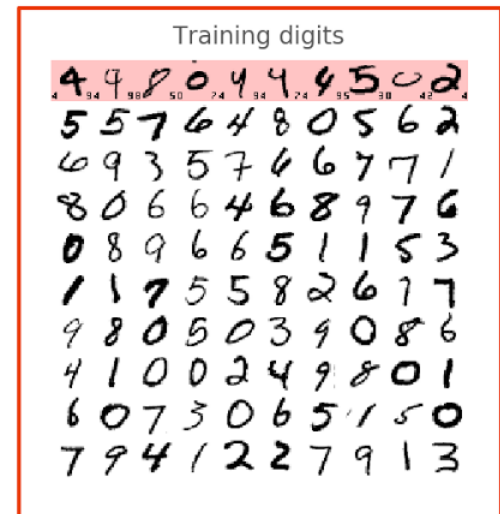
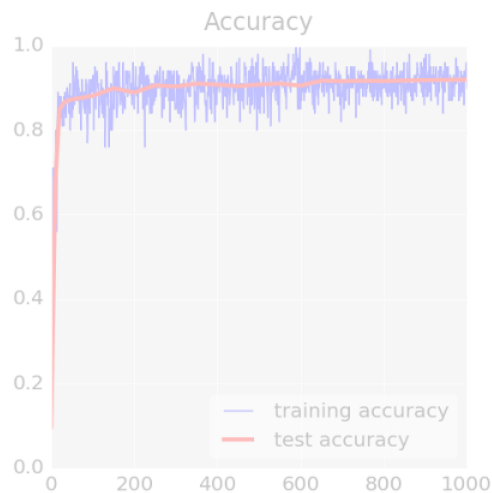


Train a Neural Network

Training digits
updates to **weights** and **biases** =>
better recognition (loop)

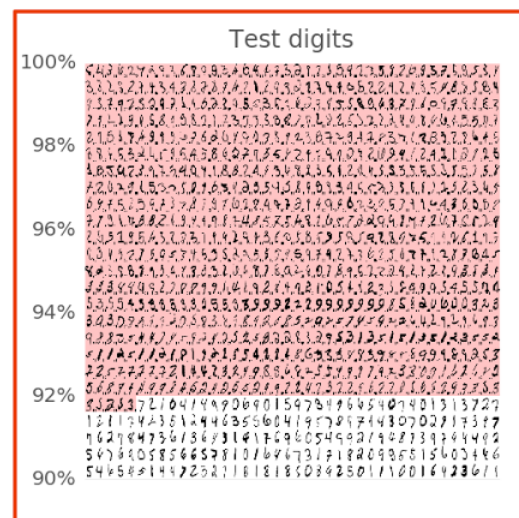
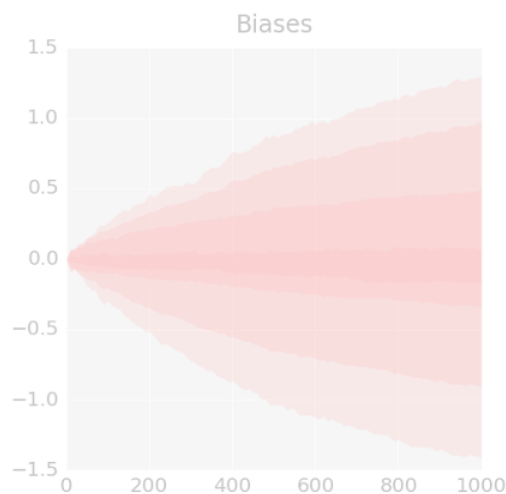
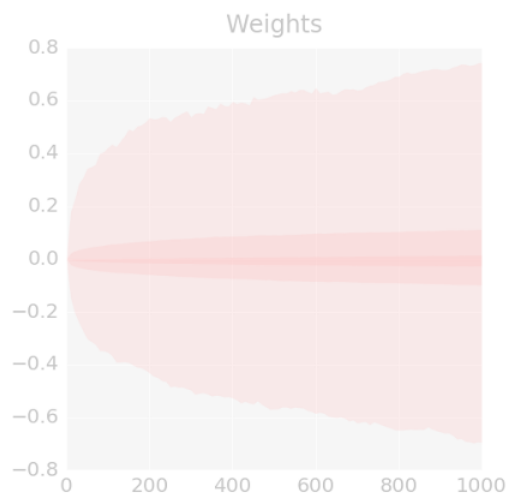
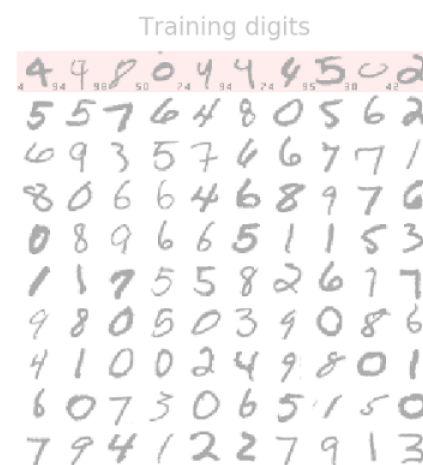
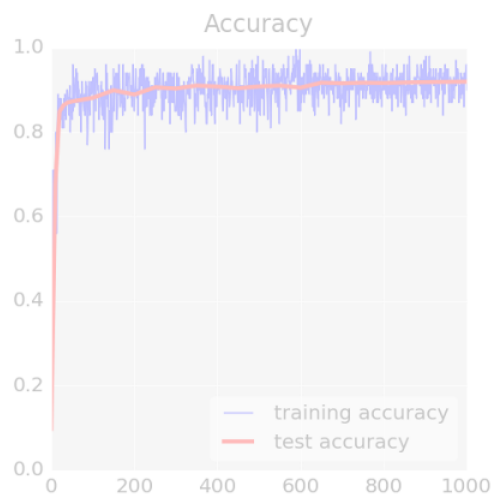
Training digits

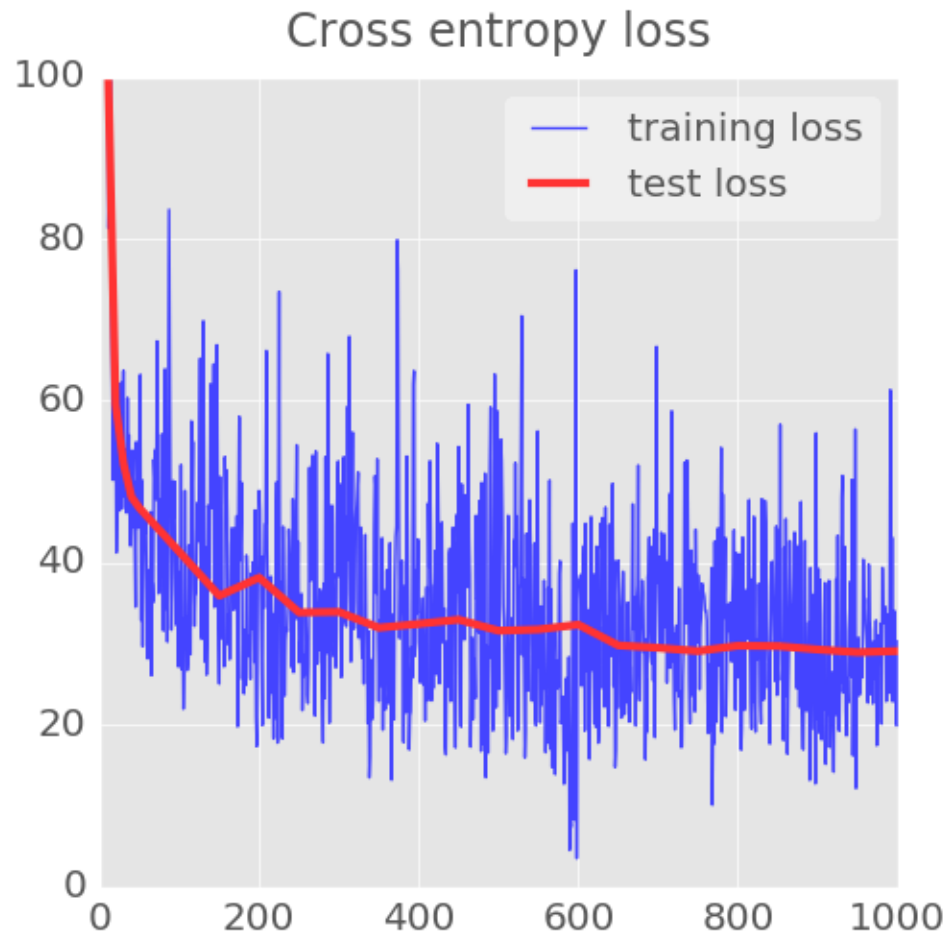


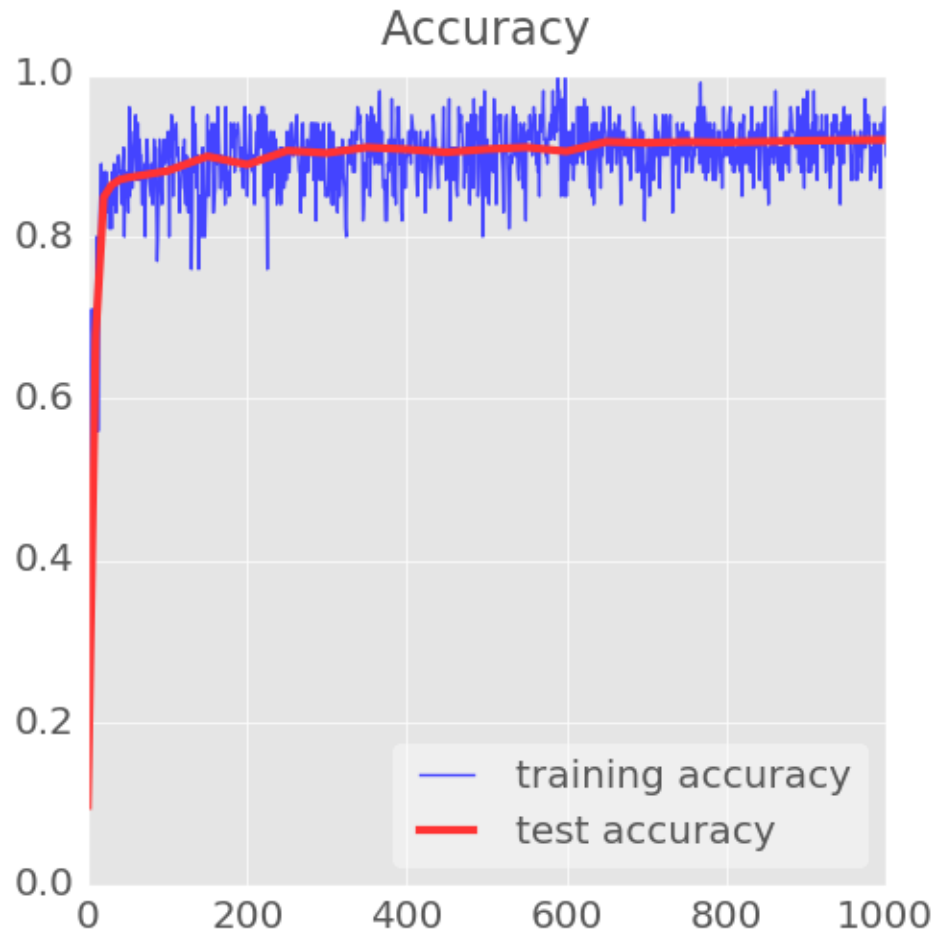


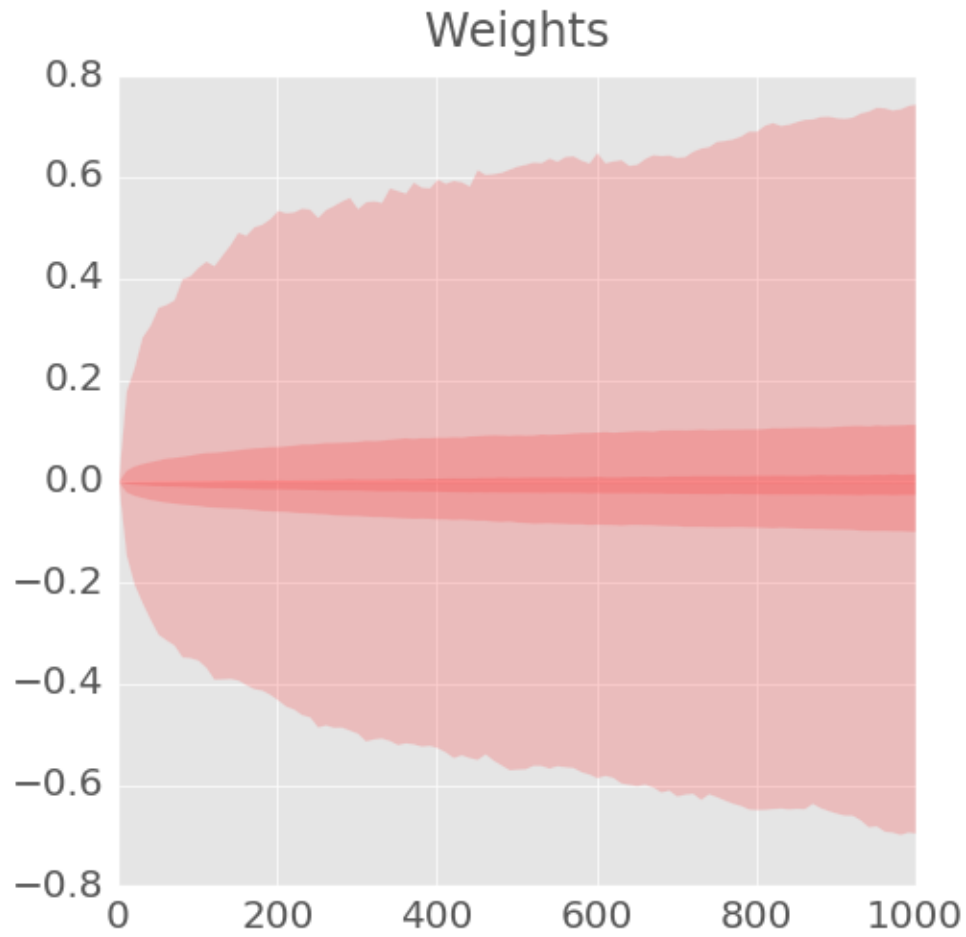
Test digits

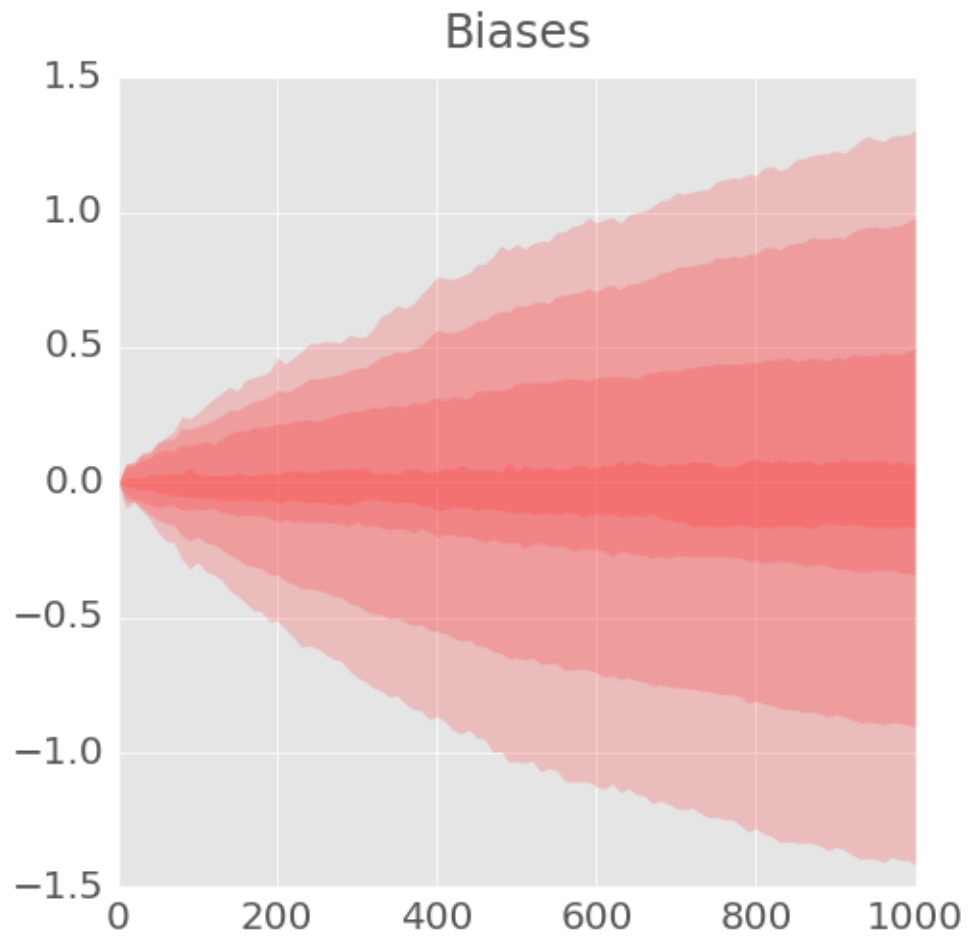


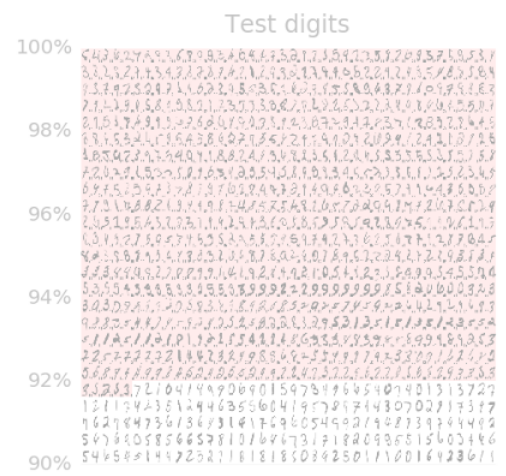
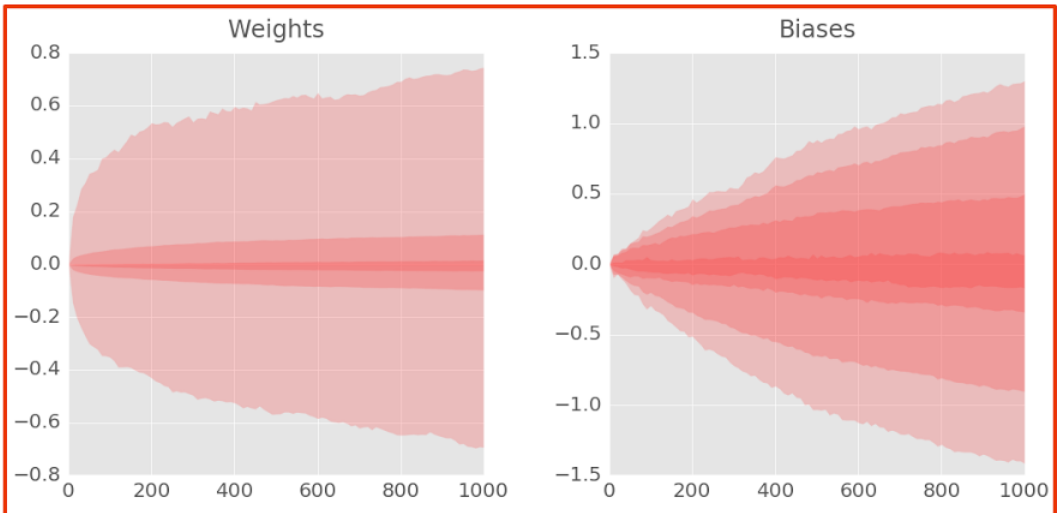
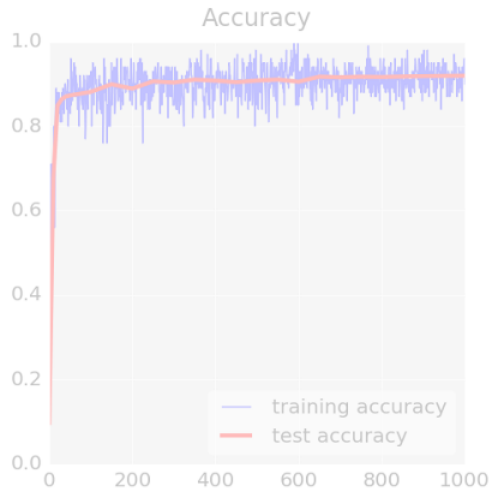










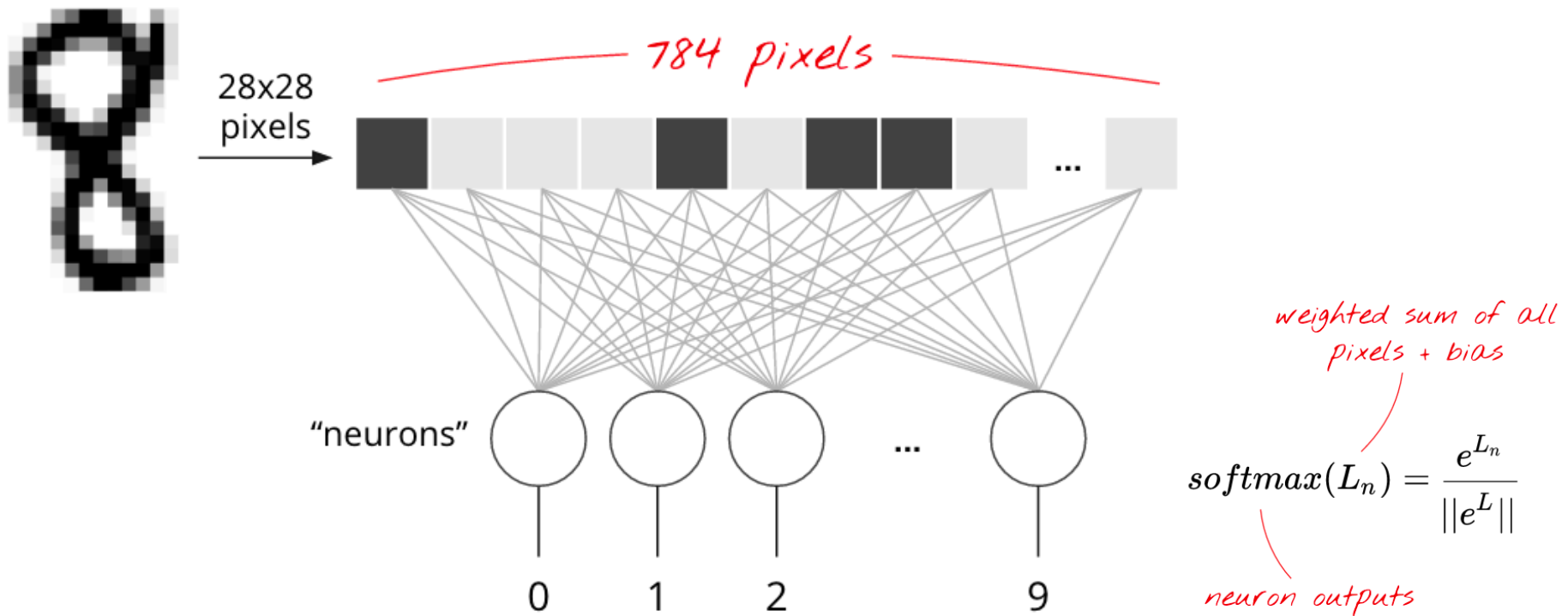


Cookbook

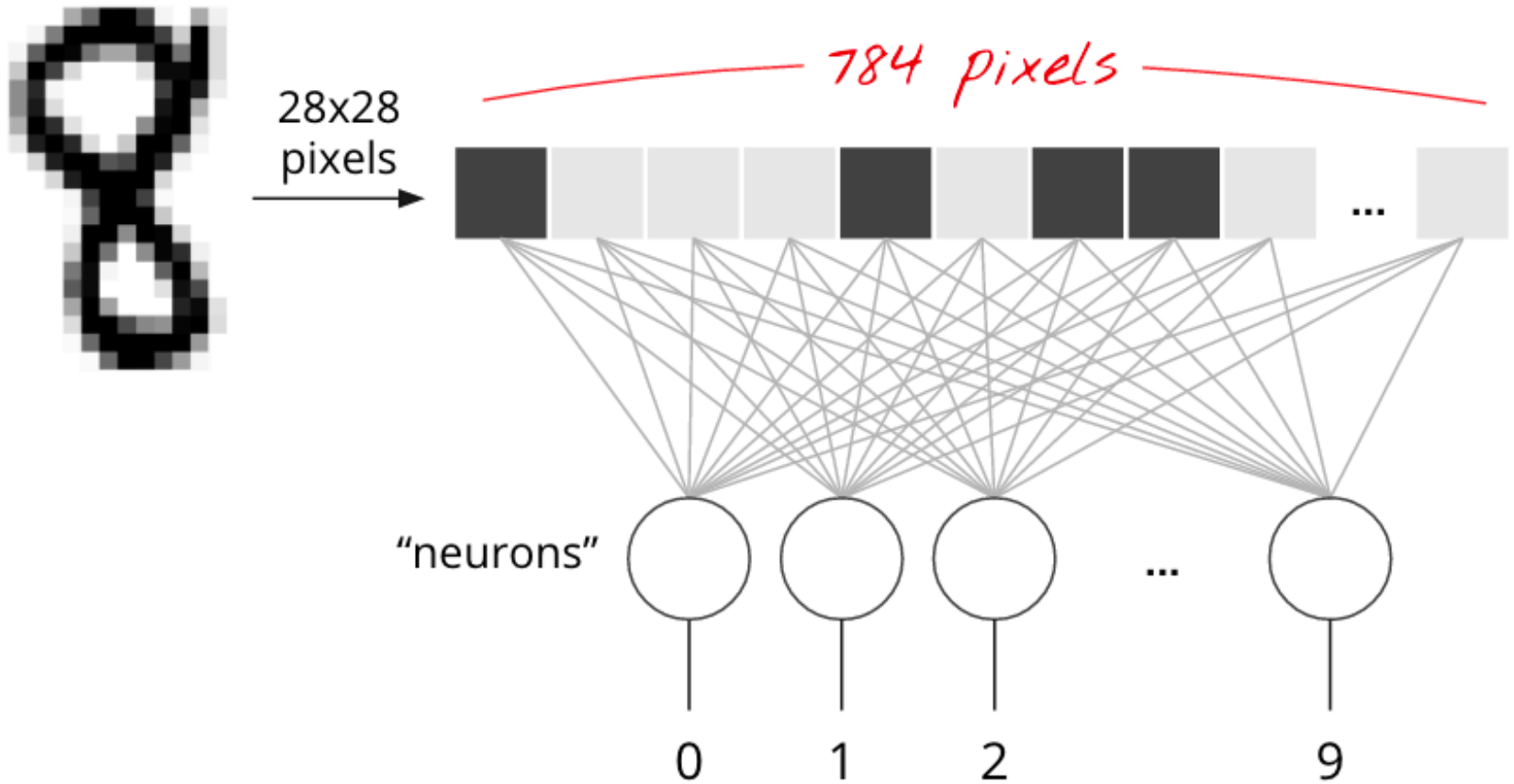
Softmax
Cross-entropy
Mini-batch



Very Simple Model: Softmax Classification



Very Simple Model: Softmax Classification



Very Simple Model: Softmax Classification

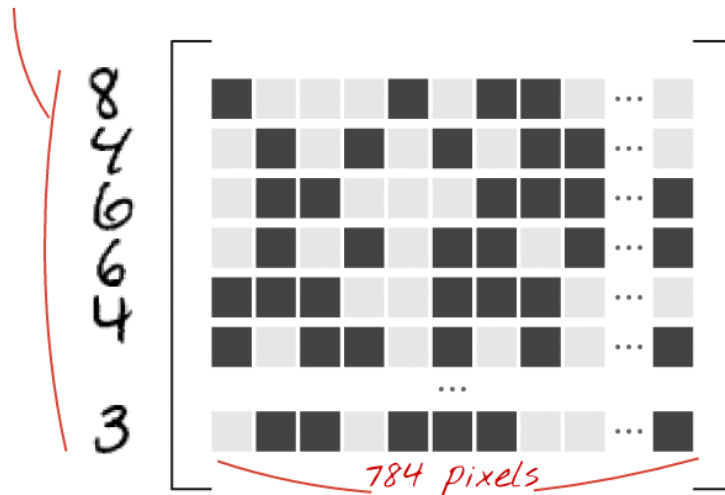
*weighted sum of all
pixels + bias*

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

neuron outputs

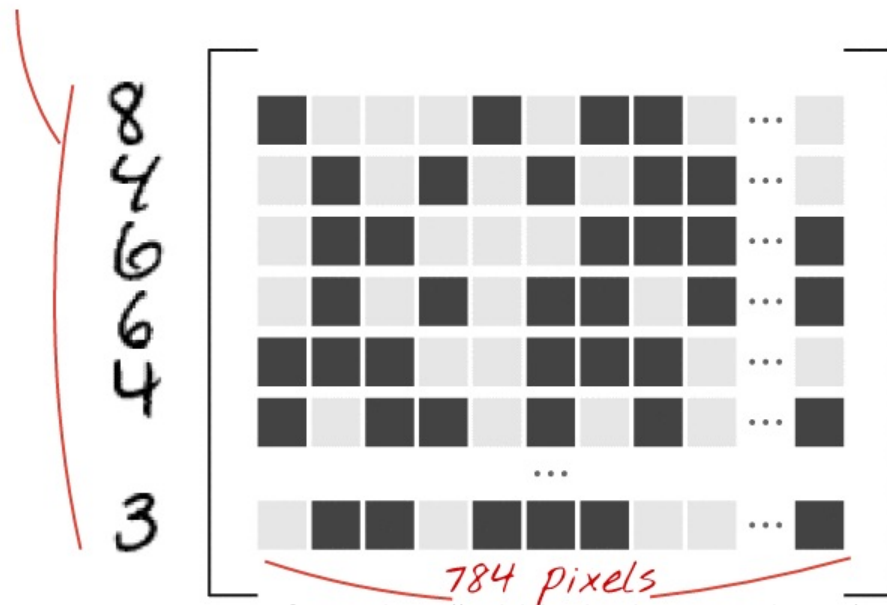
In Matrix notation, 100 images at a time

*X: 100 images,
one per line,
flattened*



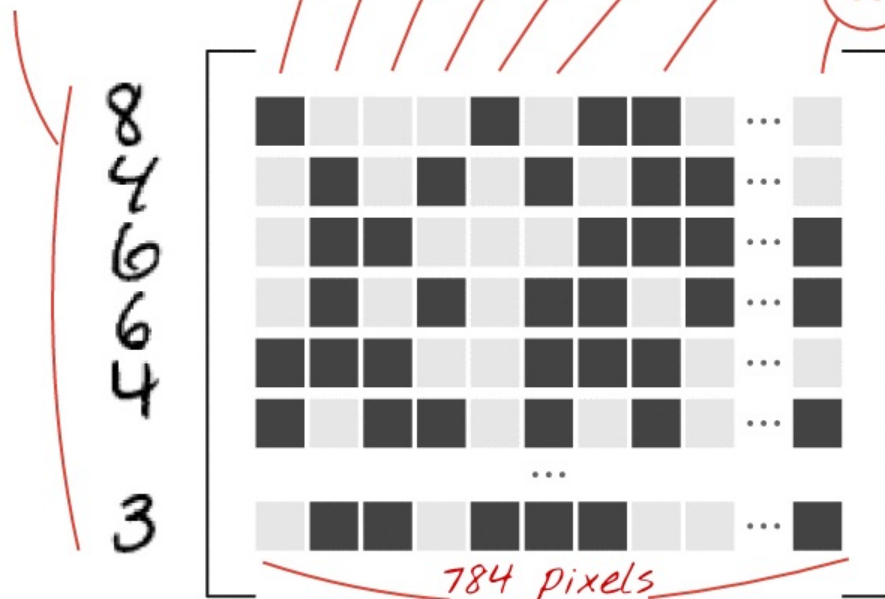
<i>10 columns</i>										<i>784 lines</i>
$W_{0,0}$	$W_{0,1}$	$W_{0,2}$	$W_{0,3}$	...	$W_{0,9}$					
$W_{1,0}$	$W_{1,1}$	$W_{1,2}$	$W_{1,3}$	...	$W_{1,9}$					
$W_{2,0}$	$W_{2,1}$	$W_{2,2}$	$W_{2,3}$	...	$W_{2,9}$					
$W_{3,0}$	$W_{3,1}$	$W_{3,2}$	$W_{3,3}$	...	$W_{3,9}$					
$W_{4,0}$	$W_{4,1}$	$W_{4,2}$	$W_{4,3}$	...	$W_{4,9}$					
$W_{5,0}$	$W_{5,1}$	$W_{5,2}$	$W_{5,3}$	...	$W_{5,9}$					
$W_{6,0}$	$W_{6,1}$	$W_{6,2}$	$W_{6,3}$	...	$W_{6,9}$					
$W_{7,0}$	$W_{7,1}$	$W_{7,2}$	$W_{7,3}$	...	$W_{7,9}$					
$W_{8,0}$	$W_{8,1}$	$W_{8,2}$	$W_{8,3}$	...	$W_{8,9}$					
...										
$W_{783,0}$	$W_{783,1}$	$W_{783,2}$	...	$W_{783,9}$						

*X: 100 images,
one per line,
flattened*



10 columns										
$W_{0,0}$	$W_{0,1}$	$W_{0,2}$	$W_{0,3}$	...	$W_{0,9}$	784 lines				
$W_{1,0}$	$W_{1,1}$	$W_{1,2}$	$W_{1,3}$	...	$W_{1,9}$					
$W_{2,0}$	$W_{2,1}$	$W_{2,2}$	$W_{2,3}$	...	$W_{2,9}$					
$W_{3,0}$	$W_{3,1}$	$W_{3,2}$	$W_{3,3}$	...	$W_{3,9}$					
$W_{4,0}$	$W_{4,1}$	$W_{4,2}$	$W_{4,3}$	...	$W_{4,9}$					
$W_{5,0}$	$W_{5,1}$	$W_{5,2}$	$W_{5,3}$	...	$W_{5,9}$					
$W_{6,0}$	$W_{6,1}$	$W_{6,2}$	$W_{6,3}$	...	$W_{6,9}$					
$W_{7,0}$	$W_{7,1}$	$W_{7,2}$	$W_{7,3}$	...	$W_{7,9}$					
$W_{8,0}$	$W_{8,1}$	$W_{8,2}$	$W_{8,3}$	...	$W_{8,9}$					
...										
$W_{783,0}$	$W_{783,1}$	$W_{783,2}$	...	$W_{783,9}$						

*X: 100 images,
one per line,
flattened*



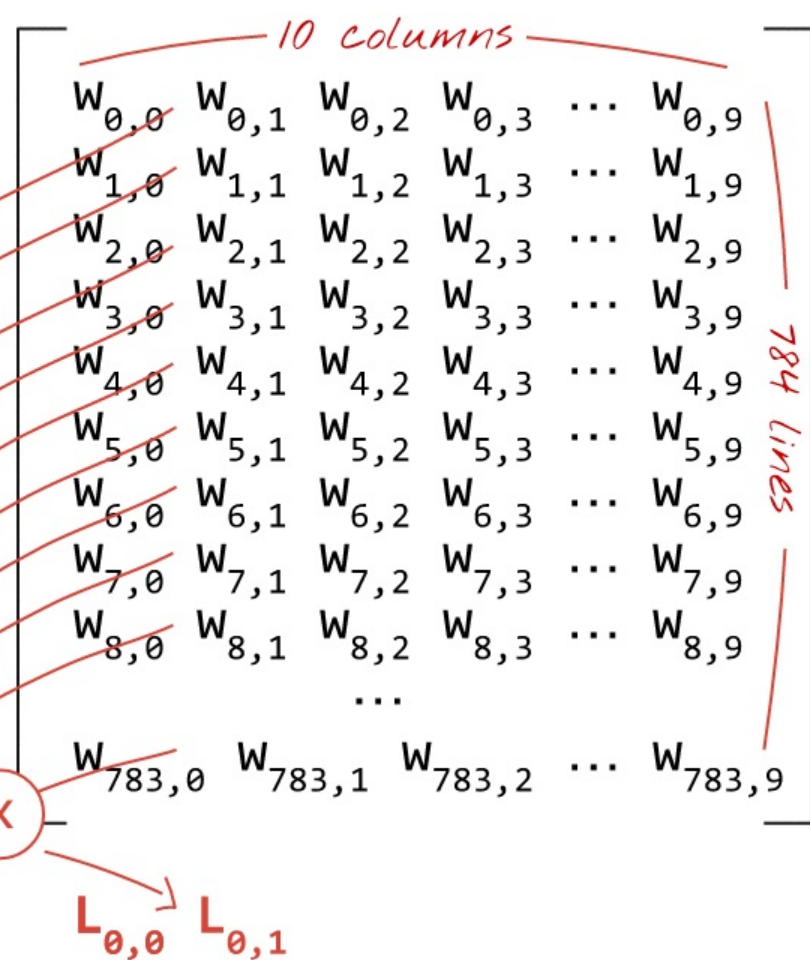
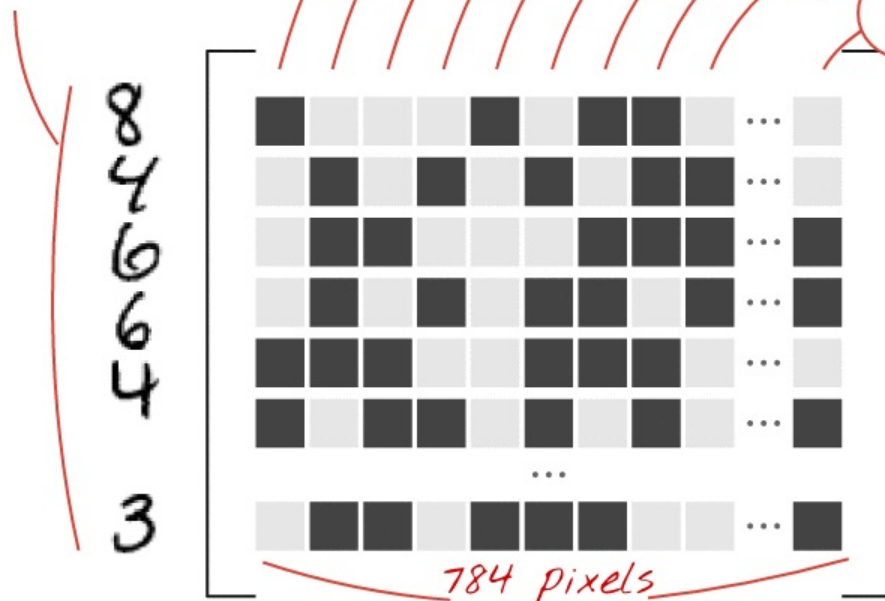
10 columns

$W_{0,0}$	$W_{0,1}$	$W_{0,2}$	$W_{0,3}$	...	$W_{0,9}$
$W_{1,0}$	$W_{1,1}$	$W_{1,2}$	$W_{1,3}$	...	$W_{1,9}$
$W_{2,0}$	$W_{2,1}$	$W_{2,2}$	$W_{2,3}$	...	$W_{2,9}$
$W_{3,0}$	$W_{3,1}$	$W_{3,2}$	$W_{3,3}$	...	$W_{3,9}$
$W_{4,0}$	$W_{4,1}$	$W_{4,2}$	$W_{4,3}$	...	$W_{4,9}$
$W_{5,0}$	$W_{5,1}$	$W_{5,2}$	$W_{5,3}$	...	$W_{5,9}$
$W_{6,0}$	$W_{6,1}$	$W_{6,2}$	$W_{6,3}$	...	$W_{6,9}$
$W_{7,0}$	$W_{7,1}$	$W_{7,2}$	$W_{7,3}$	...	$W_{7,9}$
$W_{8,0}$	$W_{8,1}$	$W_{8,2}$	$W_{8,3}$	...	$W_{8,9}$
...					
$W_{783,0}$	$W_{783,1}$	$W_{783,2}$	...		$W_{783,9}$

784 lines

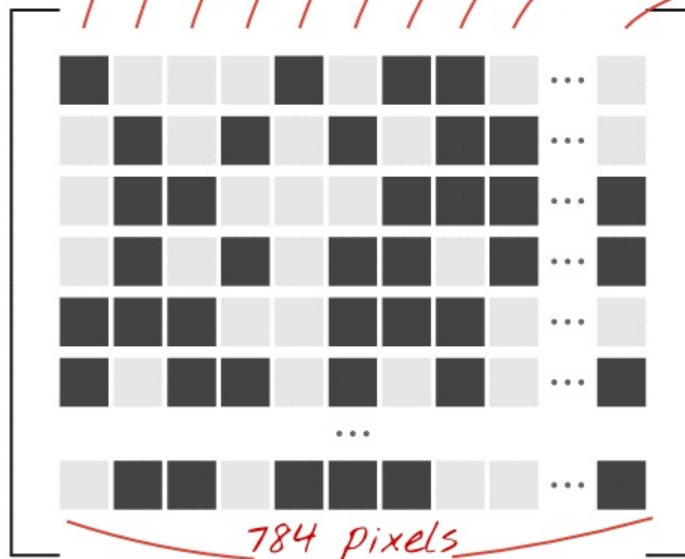
$L_{0,0}$

*X: 100 images,
one per line,
flattened*



*X: 100 images,
one per line,
flattened*

8
4
6
6
4
3



10 columns

$W_{0,0}$	$W_{0,1}$	$W_{0,2}$	$W_{0,3}$	...	$W_{0,9}$
$W_{1,0}$	$W_{1,1}$	$W_{1,2}$	$W_{1,3}$	...	$W_{1,9}$
$W_{2,0}$	$W_{2,1}$	$W_{2,2}$	$W_{2,3}$	...	$W_{2,9}$
$W_{3,0}$	$W_{3,1}$	$W_{3,2}$	$W_{3,3}$	...	$W_{3,9}$
$W_{4,0}$	$W_{4,1}$	$W_{4,2}$	$W_{4,3}$	...	$W_{4,9}$
$W_{5,0}$	$W_{5,1}$	$W_{5,2}$	$W_{5,3}$	...	$W_{5,9}$
$W_{6,0}$	$W_{6,1}$	$W_{6,2}$	$W_{6,3}$	...	$W_{6,9}$
$W_{7,0}$	$W_{7,1}$	$W_{7,2}$	$W_{7,3}$	...	$W_{7,9}$
$W_{8,0}$	$W_{8,1}$	$W_{8,2}$	$W_{8,3}$	...	$W_{8,9}$
...	...	...	...	...	...
$W_{783,0}$	$W_{783,1}$	$W_{783,2}$	...	...	$W_{783,9}$

784 lines

$L_{0,0}$	$L_{0,1}$	$L_{0,2}$	$L_{0,3}$	...	$L_{0,9}$
$L_{1,0}$	$L_{1,1}$	$L_{1,2}$	$L_{1,3}$	...	$L_{1,9}$
$L_{2,0}$	$L_{2,1}$	$L_{2,2}$	$L_{2,3}$	...	$L_{2,9}$
$L_{3,0}$	$L_{3,1}$	$L_{3,2}$	$L_{3,3}$	...	$L_{3,9}$
$L_{4,0}$	$L_{4,1}$	$L_{4,2}$	$L_{4,3}$	...	$L_{4,9}$
...	...	...	...	...	...
$L_{99,0}$	$L_{99,1}$	$L_{99,2}$	...	...	$L_{99,9}$

What are "weights" and "biases" ?

How is the "cross-entropy"
computed ?

How exactly does the training
algorithm work ?

$$Y = f(X)$$

Predictions

$Y[100, 10]$

Images

$X[100, 784]$

Weights

$W[784, 10]$

Biases

$b[10]$

$$Y = \text{softmax}(X.W + b)$$

applied line
by line

matrix multiply

broadcast
on all lines

tensor shapes in []

Cross Entropy

0	1	2	3	4	5	6	7	8	9
0	0	0	0	0	0	1	0	0	0

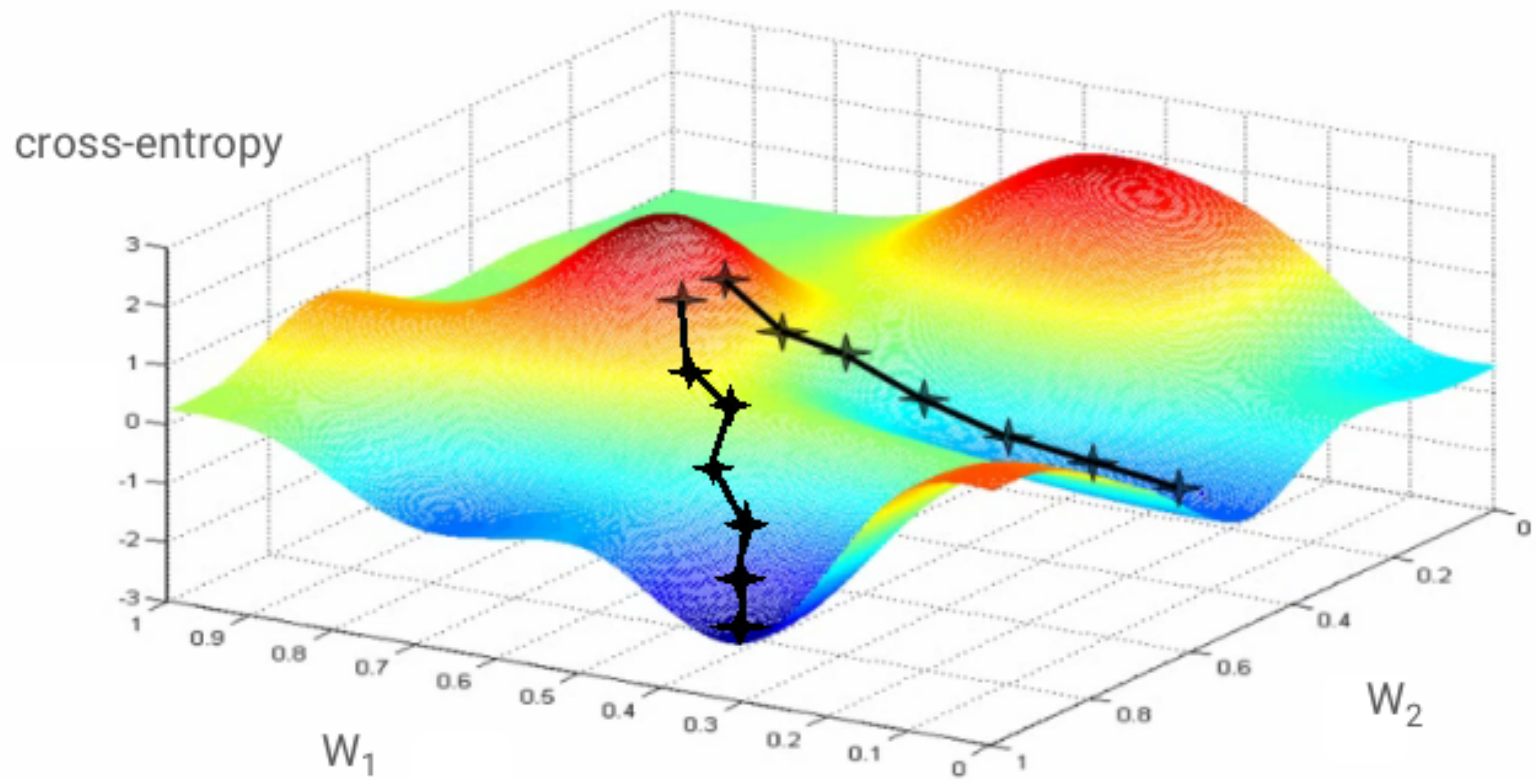
actual probabilities, "one-hot" encoded

Cross entropy: $-\sum Y'_i \cdot \log(Y_i)$

computed probabilities

0.1	0.2	0.1	0.3	0.2	0.1	0.9	0.2	0.1	0.1
0	1	2	3	4	5	6	7	8	9

this is a "6"



Training Loop

Training digits and labels

=> loss function

=> gradient (partial derivatives)

=> steepest descent

=> update weights and biases

**=> repeat with next mini-batch of
training images and labels**

**"mini-batches":
100 images and labels**

```
import tensorflow as tf
```


mnist_1.0_softmax.py

```
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()
```

mnist_1.0_softmax.py

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

mnist_1.0_softmax.py

```
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)
```

mnist_1.0_softmax.py

```
# success ?
```

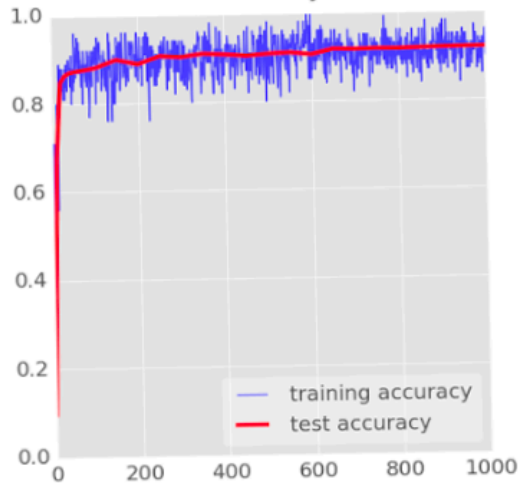
```
a,c = sess.run([accuracy, cross_entropy],  
feed_dict=train_data)
```

```
# success on test data ?
```

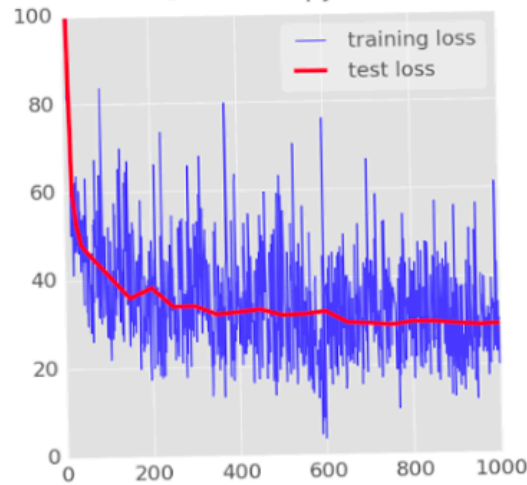
```
test_data={X: mnist.test.images, Y_: mnist.test.labels}  
a,c = sess.run([accuracy, cross_entropy], feed=test_data)
```

mnist_1.0_softmax.py

Accuracy



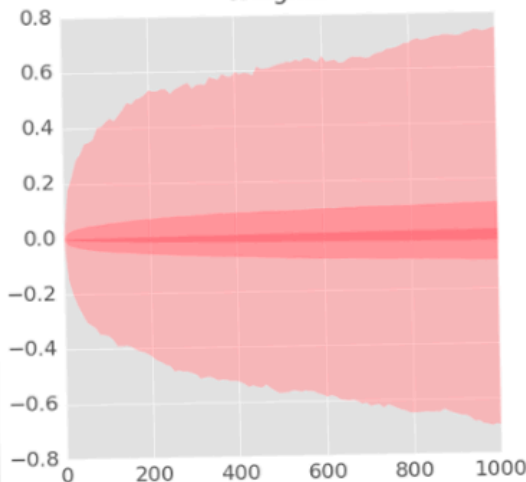
Cross entropy loss



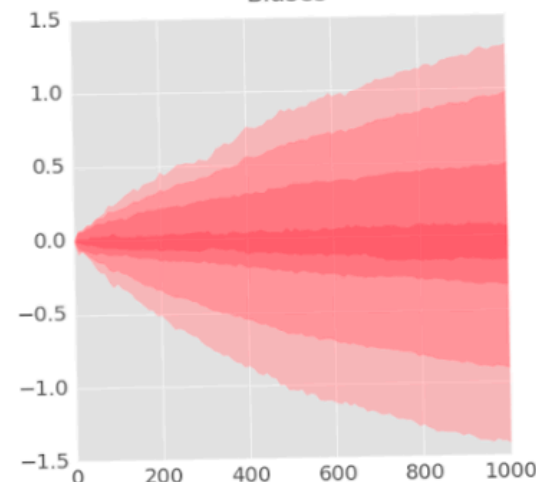
Training digits

4 4 8 0 4 4 4 5 0 2
5 5 7 6 4 8 0 5 6 2
6 9 3 5 7 6 6 7 7 1
8 0 6 6 4 6 8 9 7 6
0 8 9 6 6 5 1 1 5 3
1 1 7 5 5 8 2 6 7 7
9 8 0 5 0 3 9 0 8 6
4 1 0 0 2 4 9 8 0 1
6 0 7 3 0 6 5 1 5 0
7 9 4 1 2 2 7 9 1 3

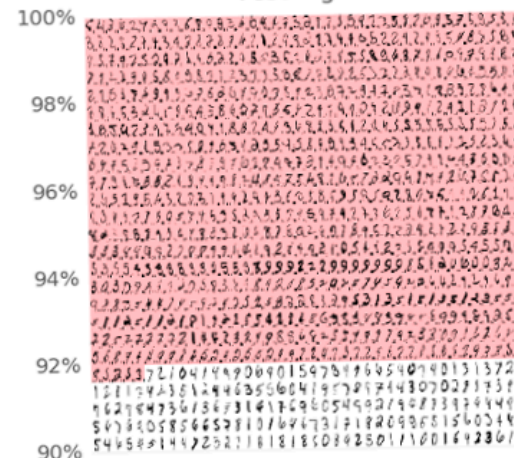
Weights



Biases



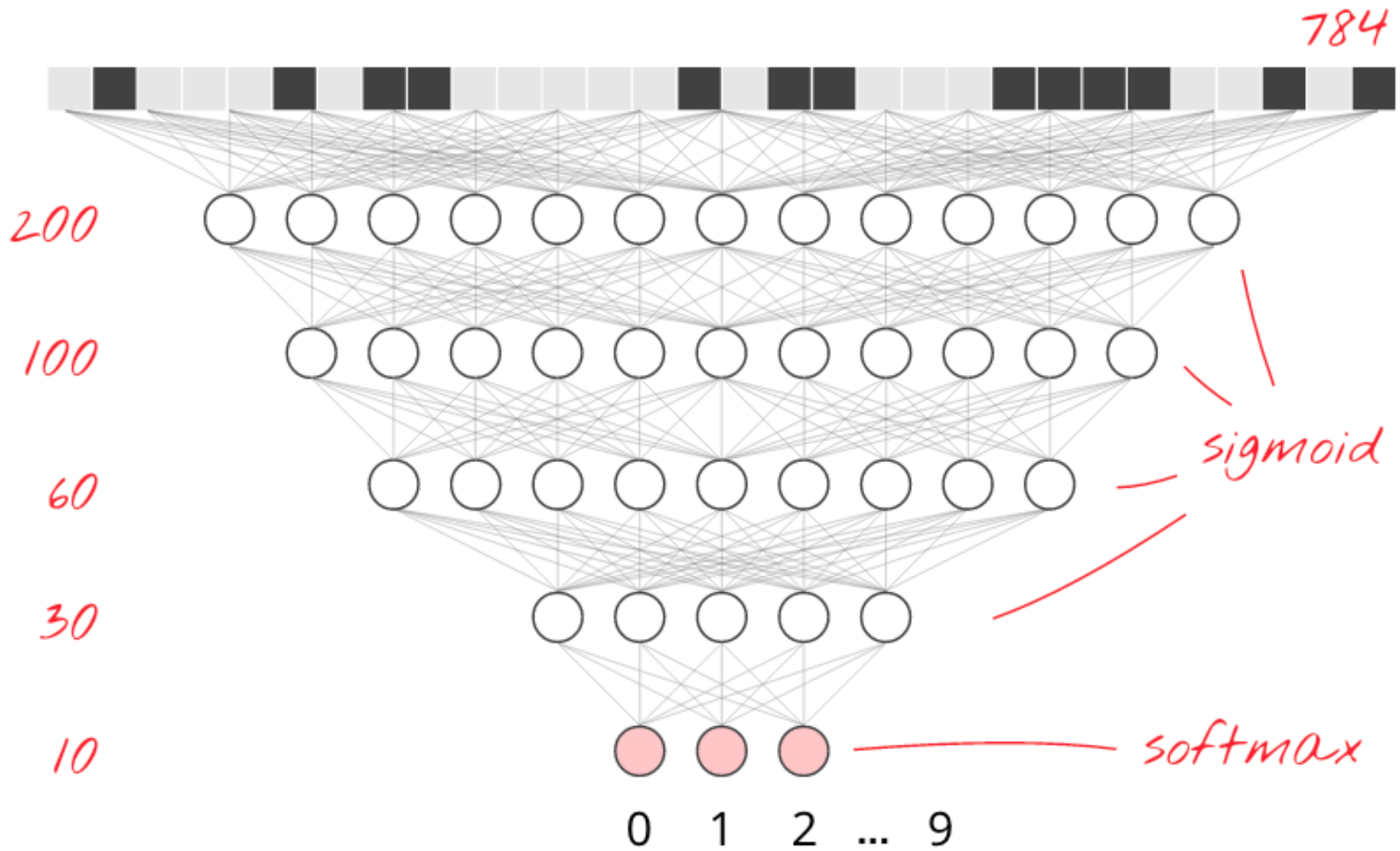
Test digits



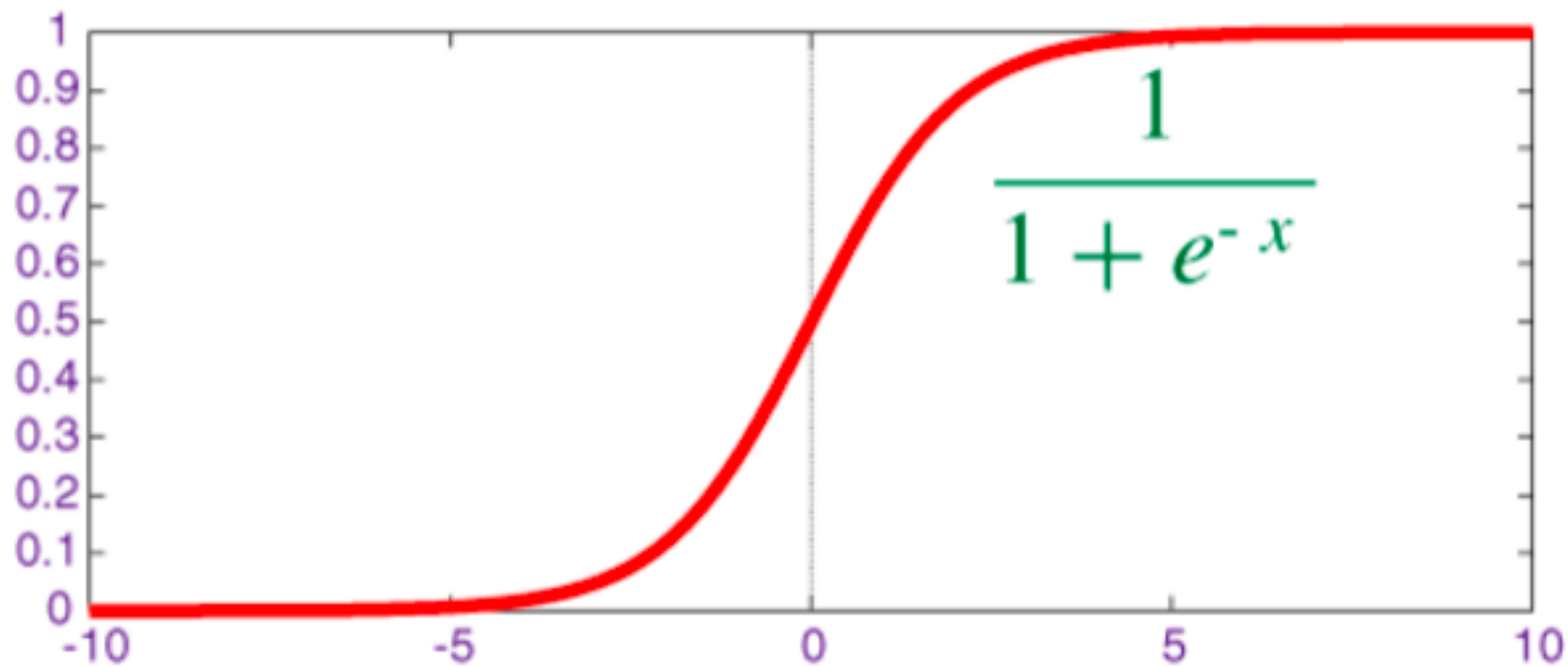
92!
😊

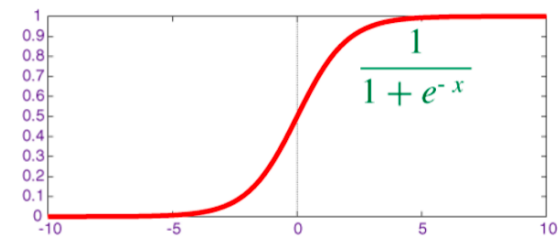
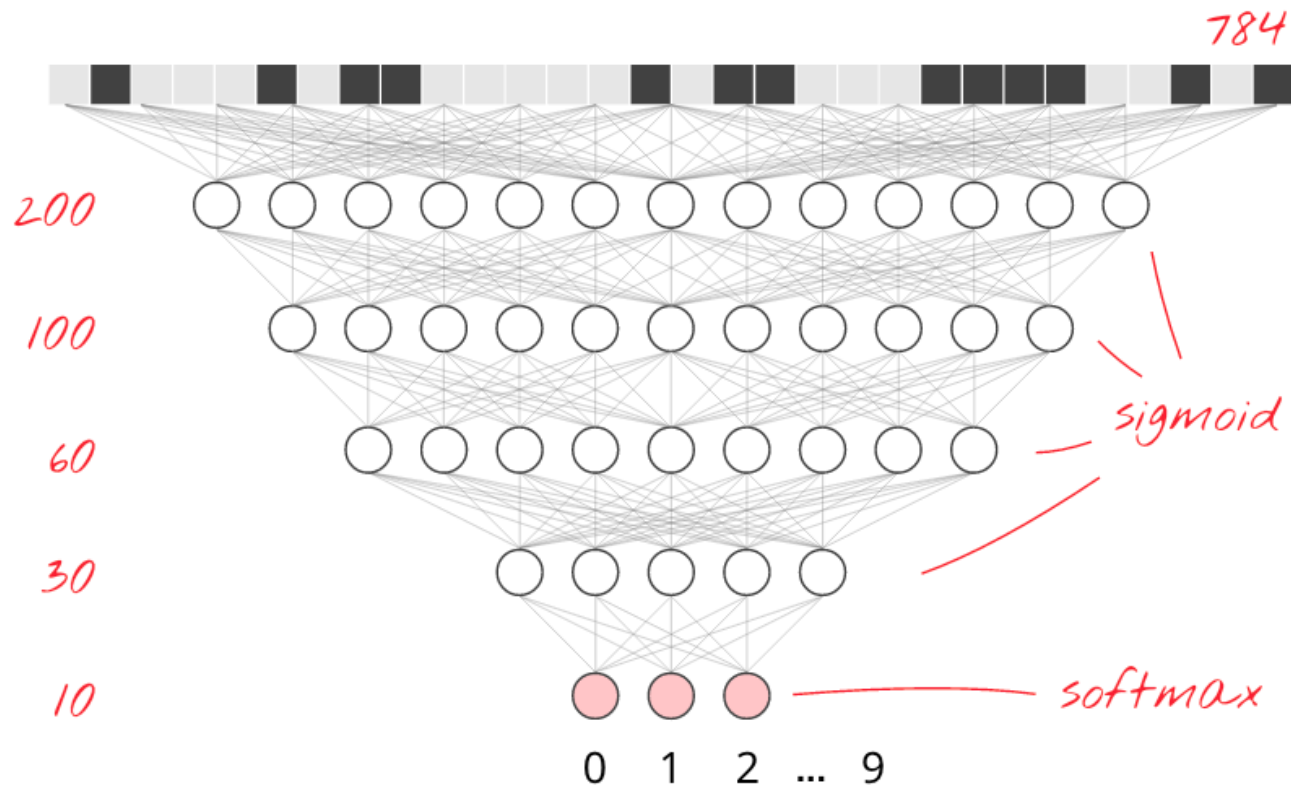
Deep Learning

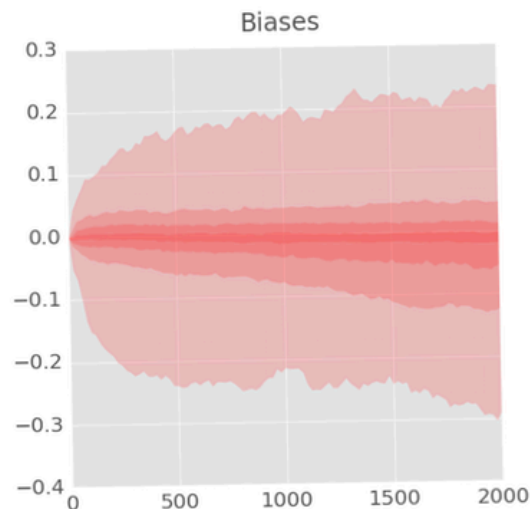
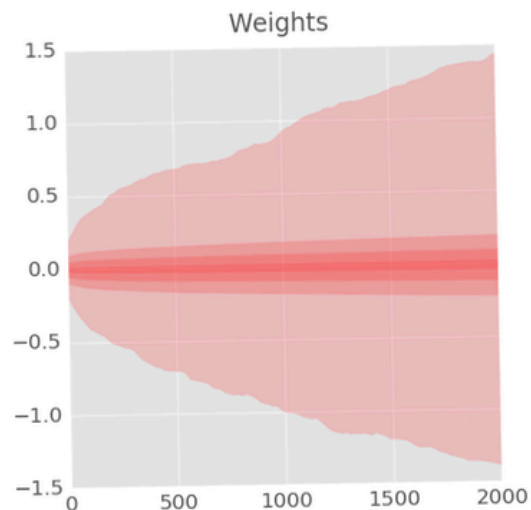
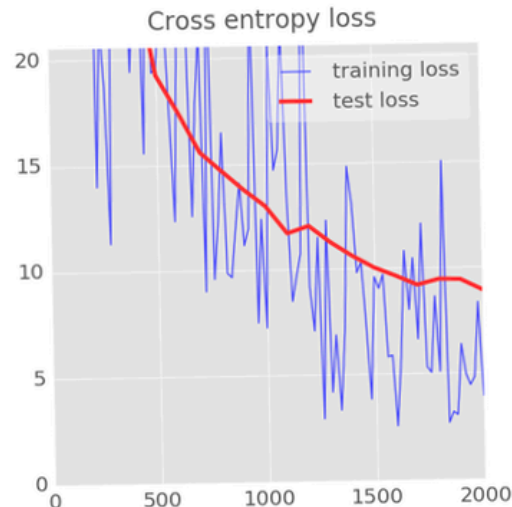
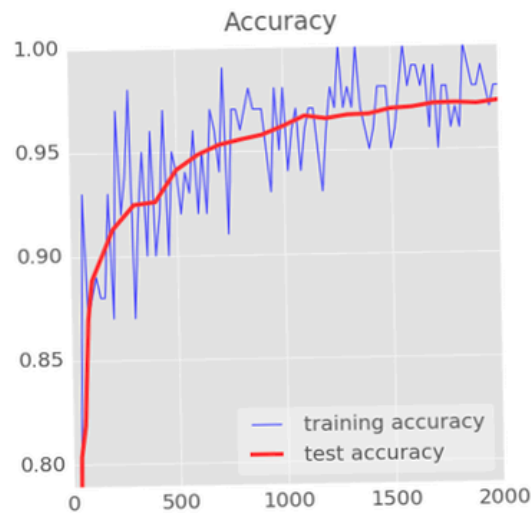




Sigmoid





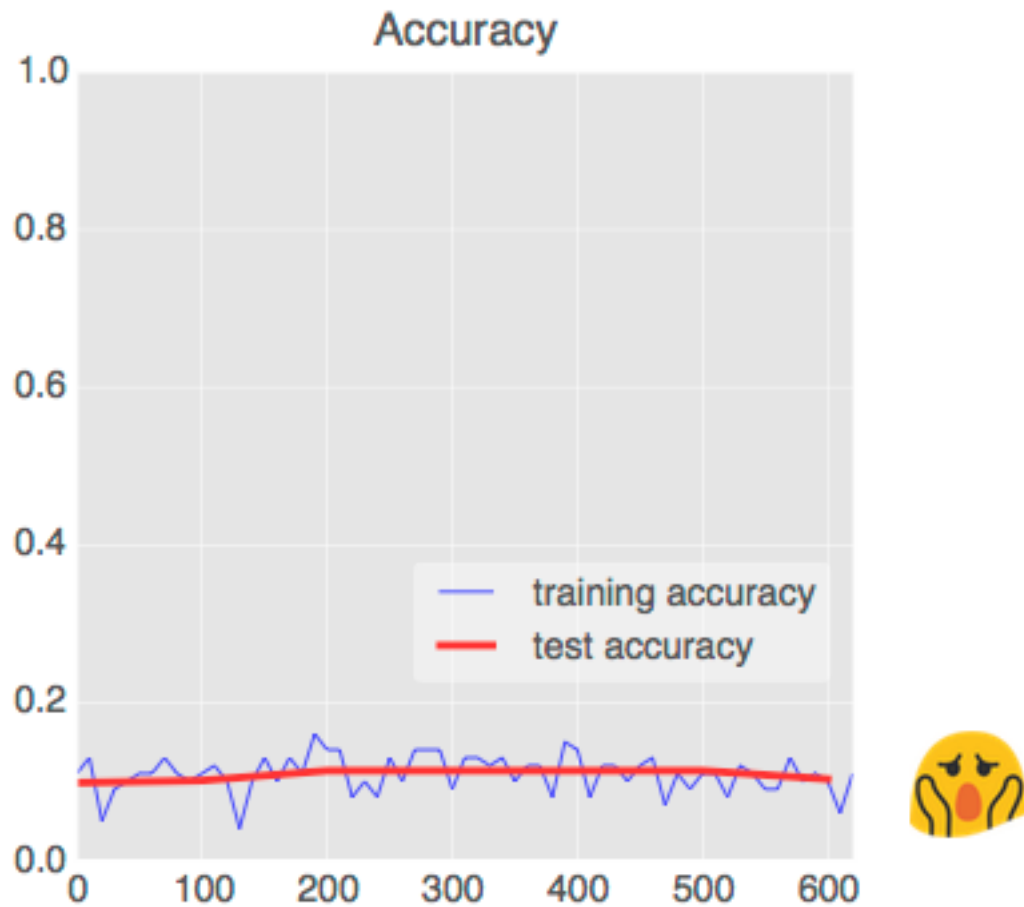


97%

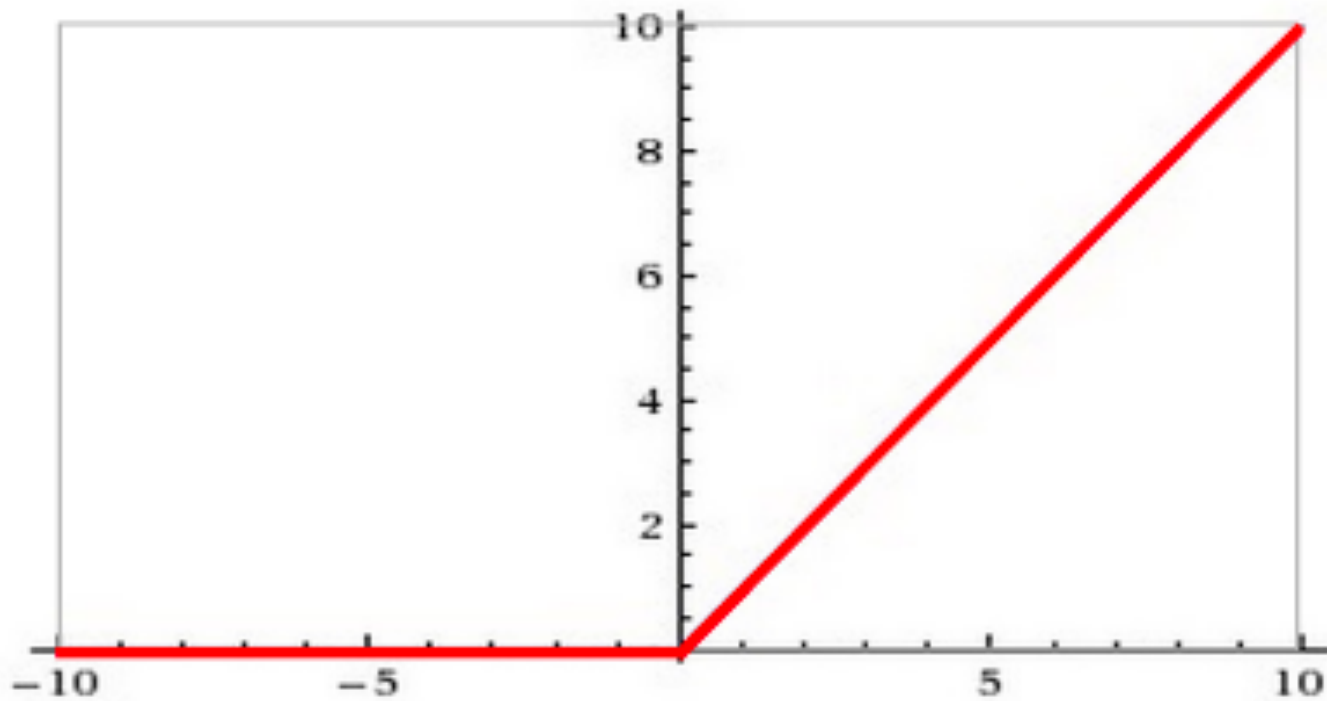


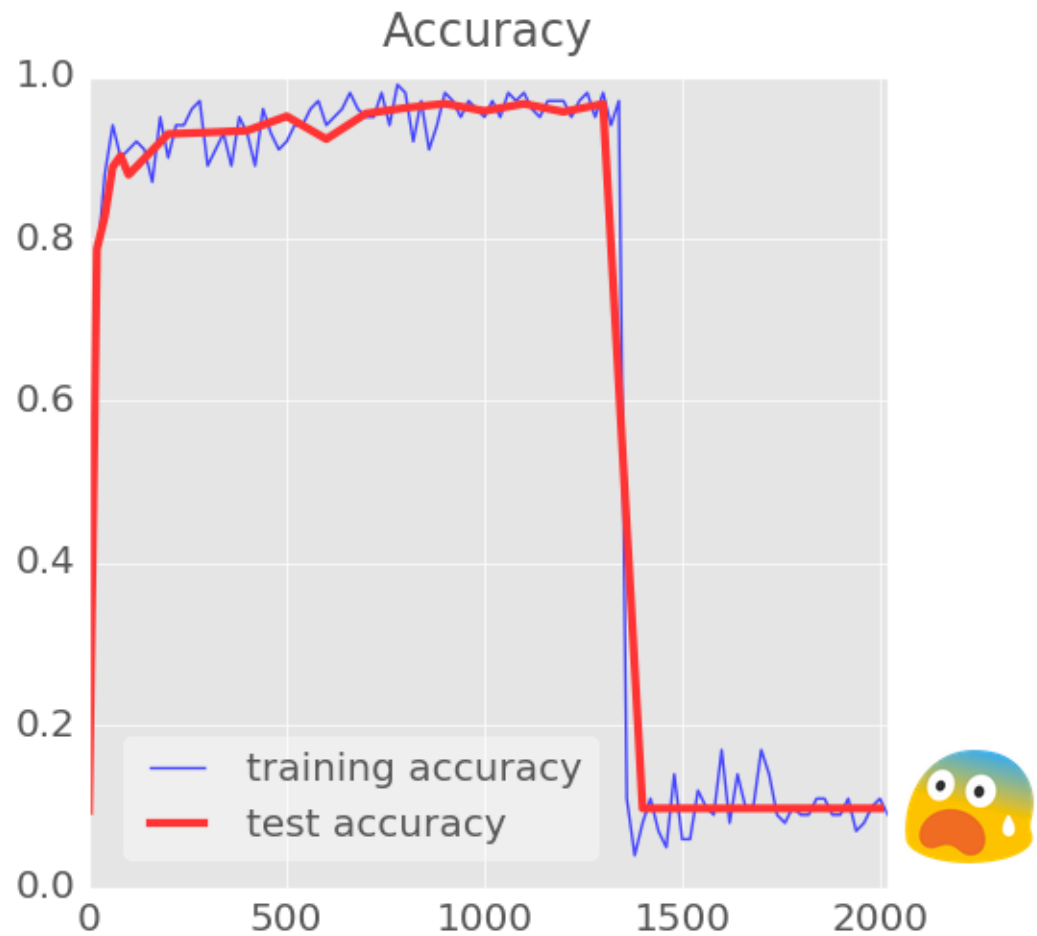
ReLU





ReLU





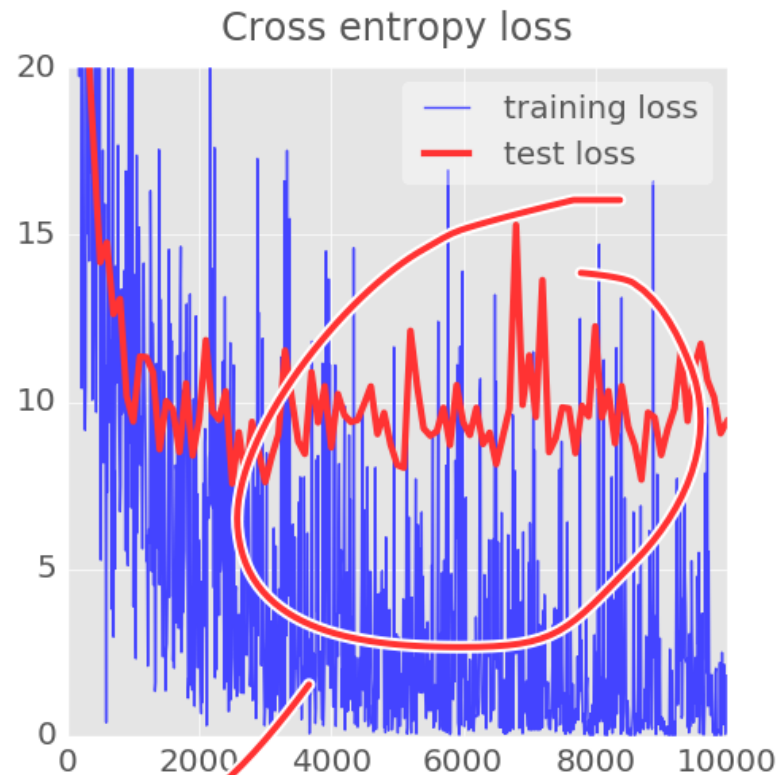
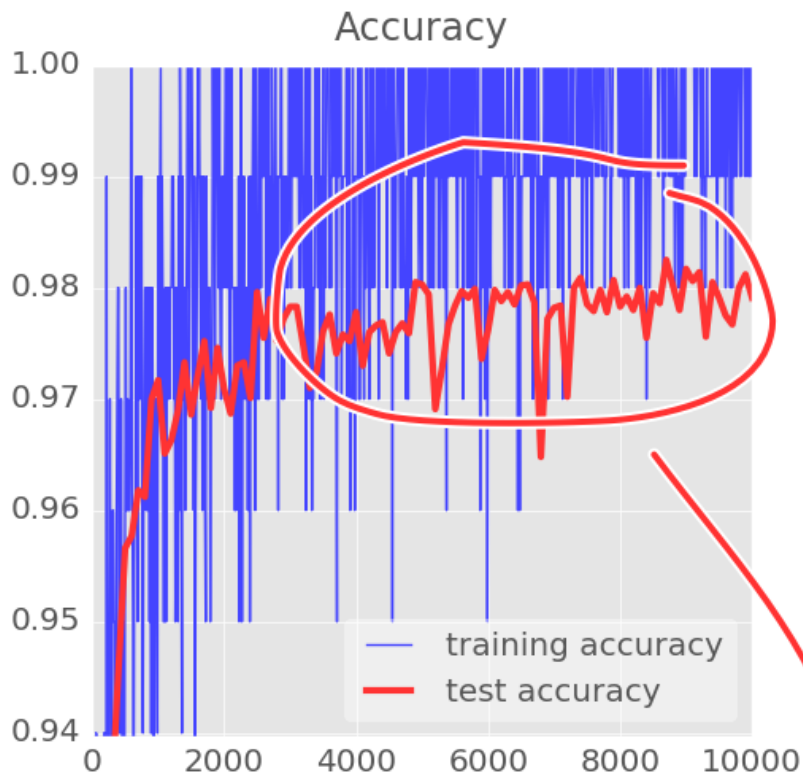
Learning Rate

Slow down...

Learning
rate decay

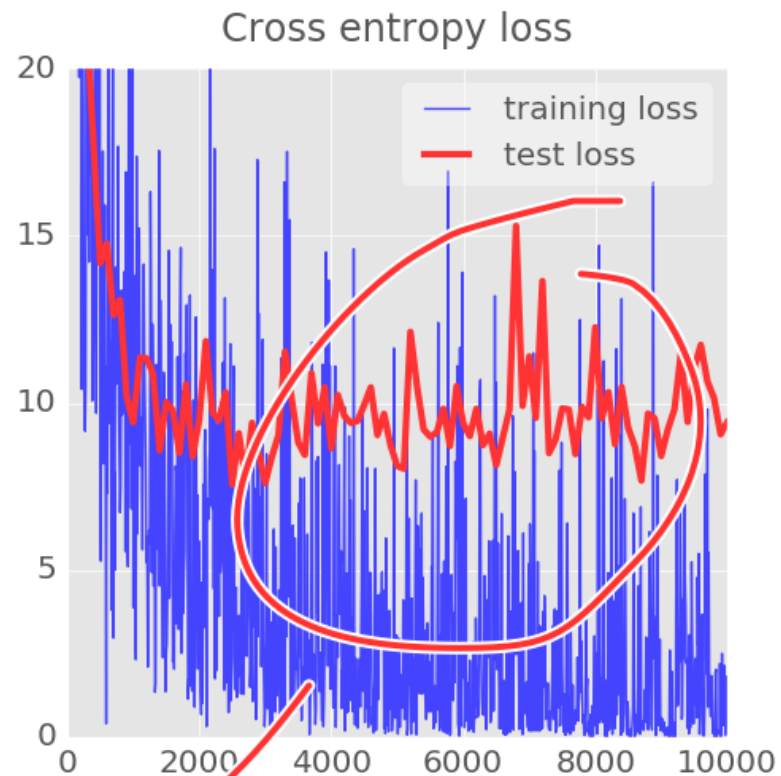
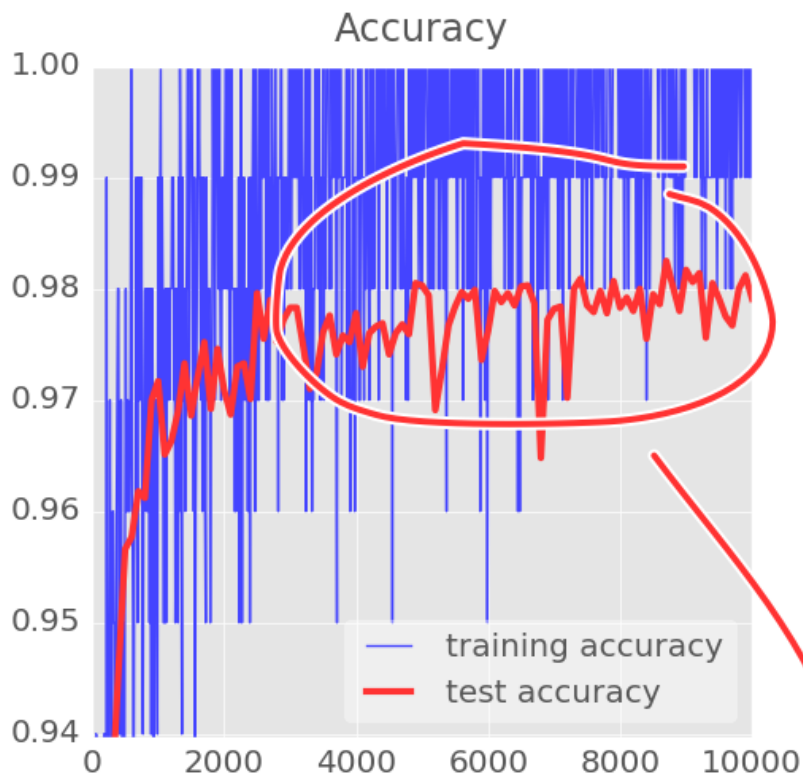


LR =
0.003

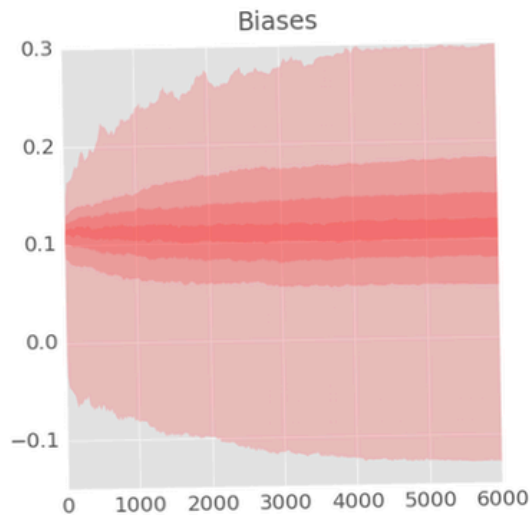
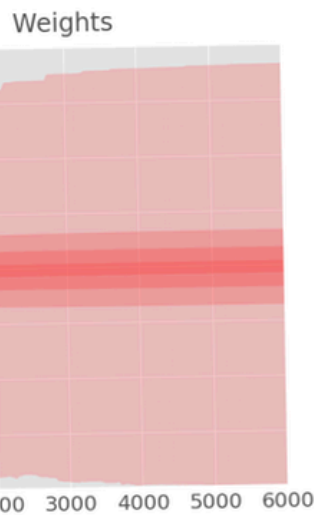
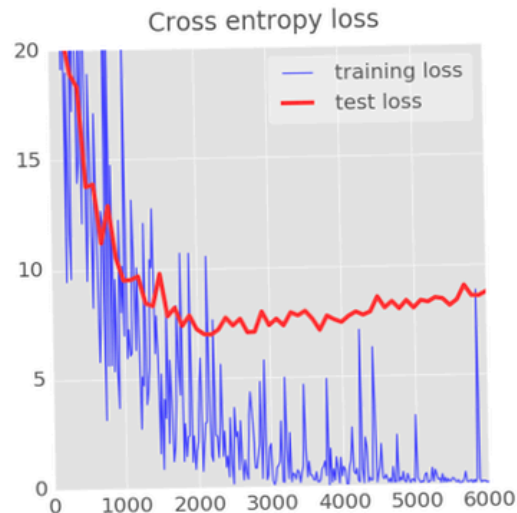
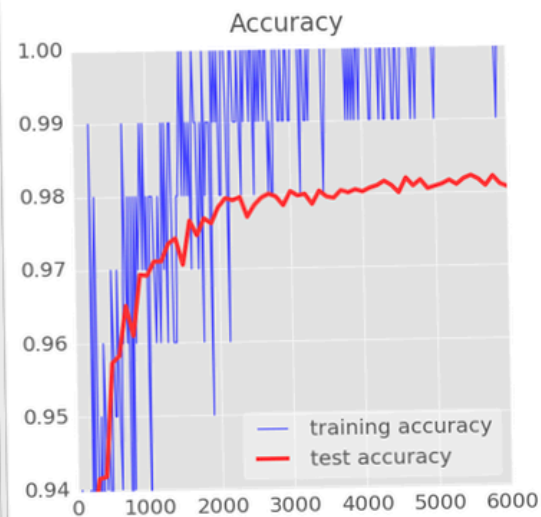


yuck!

LR =
0.003



yuck!



Training digits

```

5 8 0 6 2 8 2 8 6 5
8 4 4 / 6 7 9 9 4 5
0 7 8 7 8 0 7 1 0 7
5 5 7 5 5 8 3 8 5 9
6 6 2 4 0 8 6 7 5 1
3 9 1 0 5 4 0 5 0 1
6 4 8 3 7 3 7 8 1 1
1 6 8 8 0 2 6 5 1 0
0 6 6 7 5 4 8 6 4 5
1 8 7 2 5 0 1 5 2 9
  
```



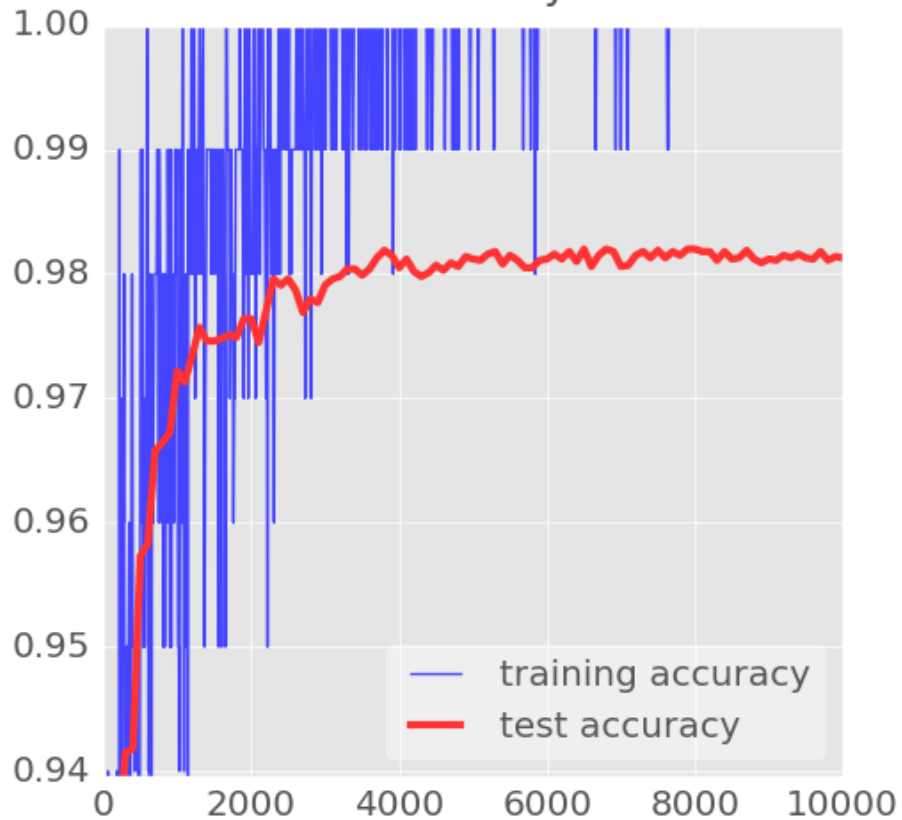
98% 😊

Dropout

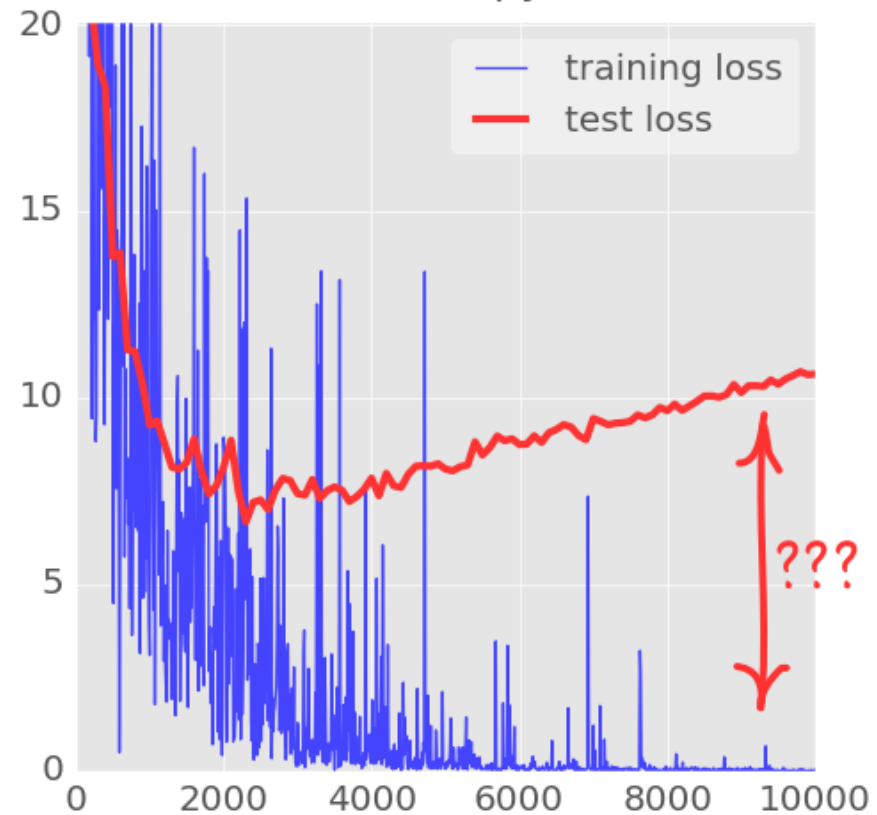
Dropout



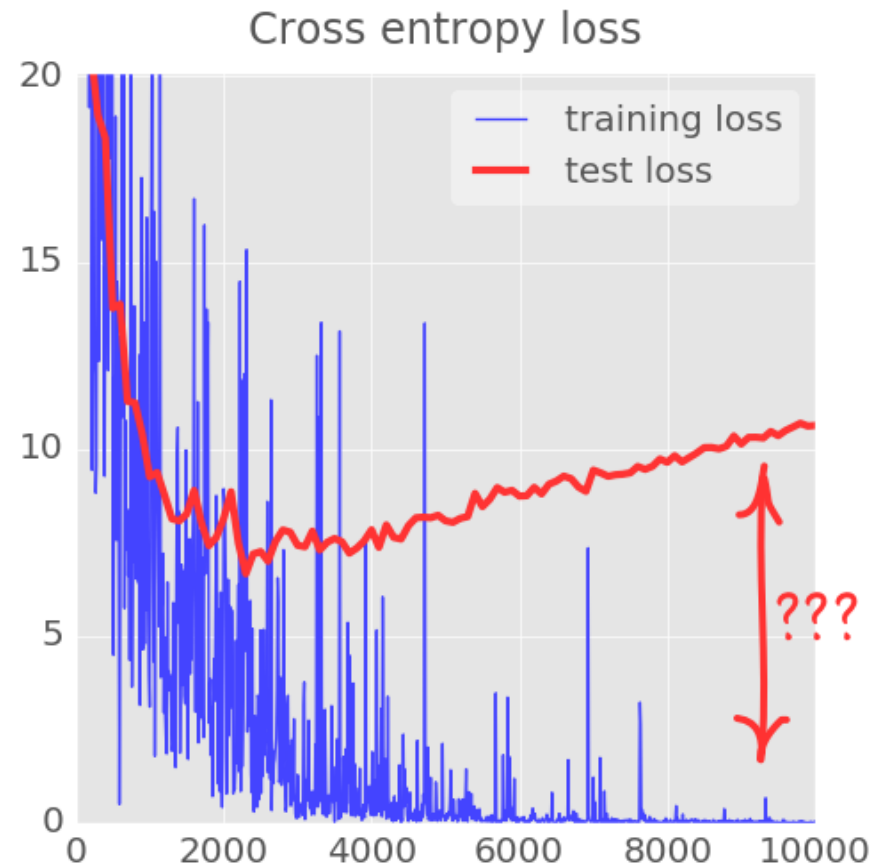
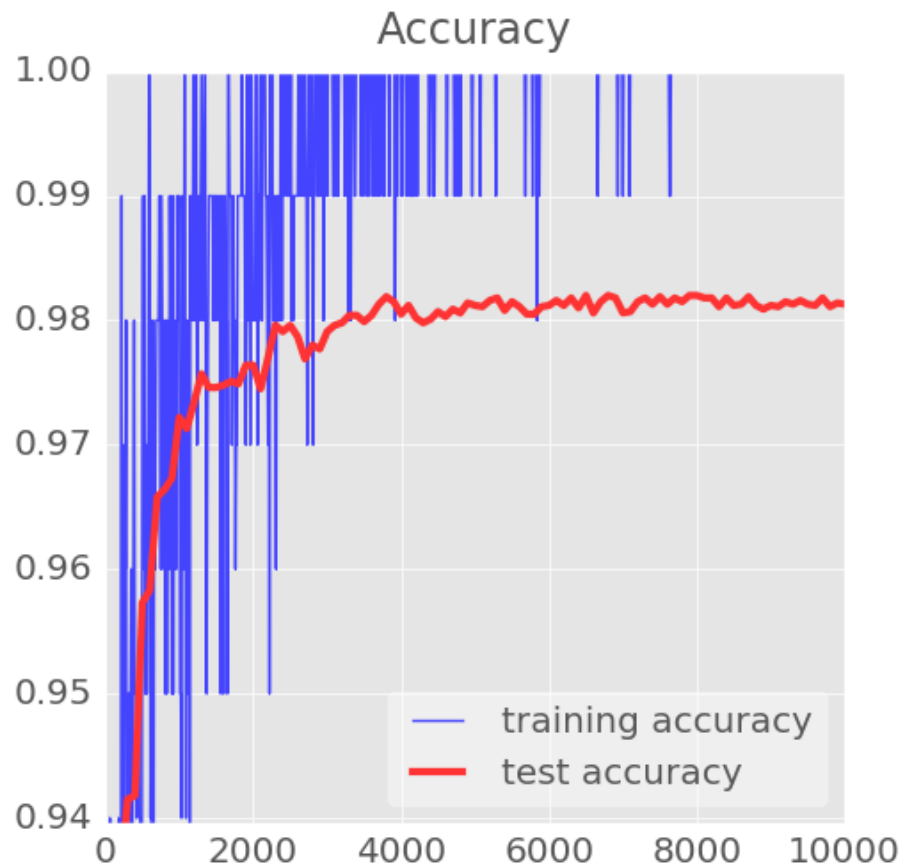
Accuracy



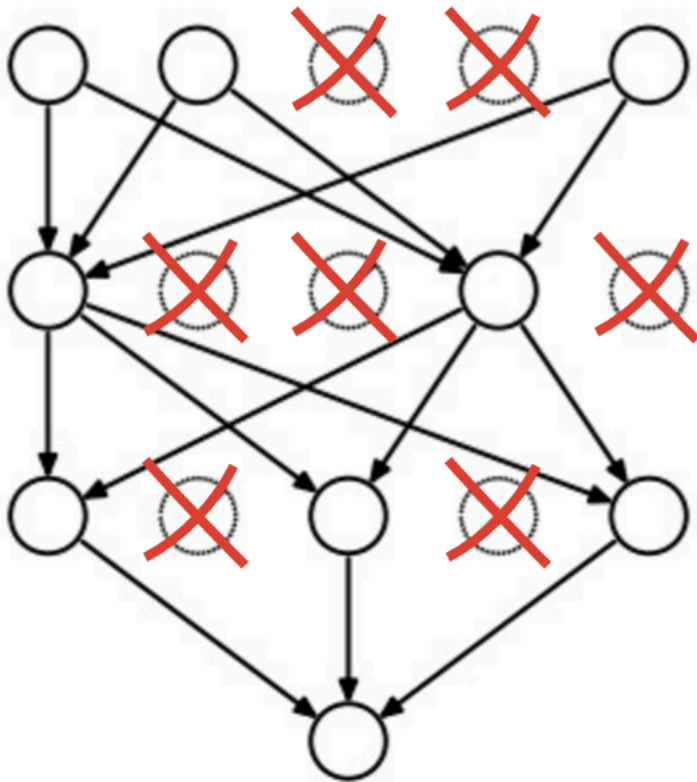
Cross entropy loss



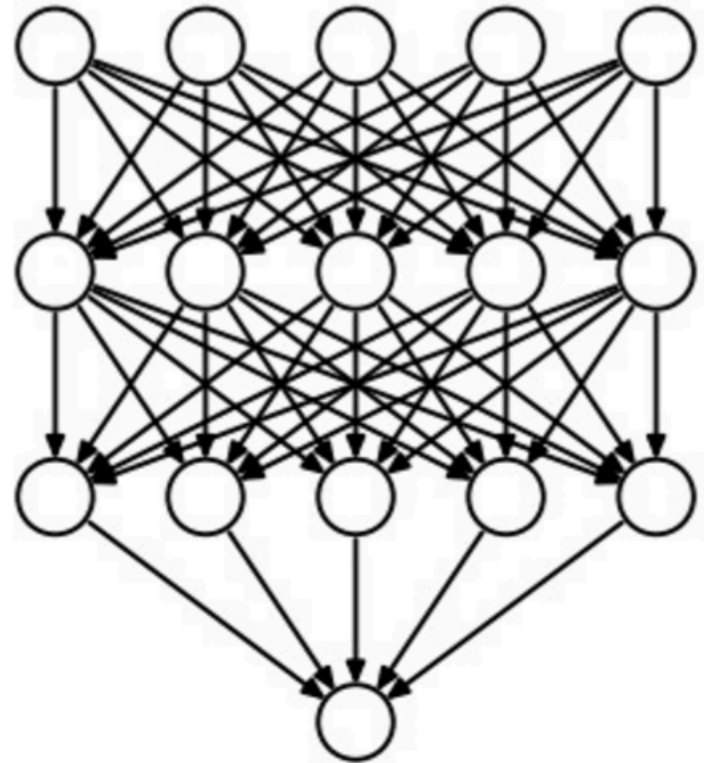
Overfitting



Dropout

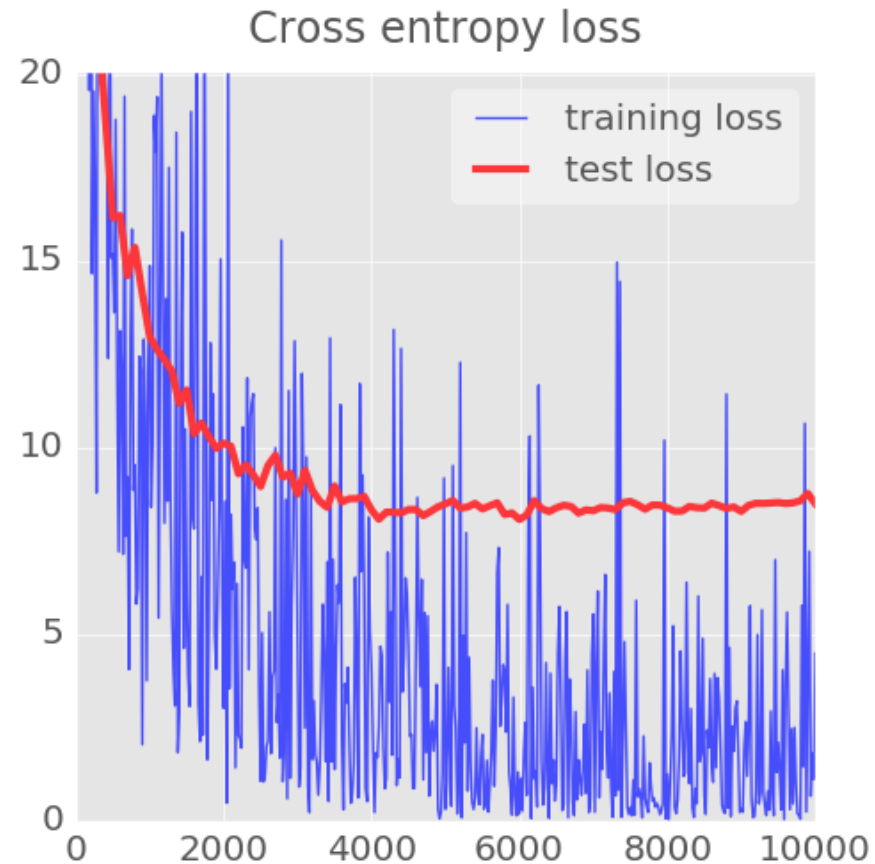
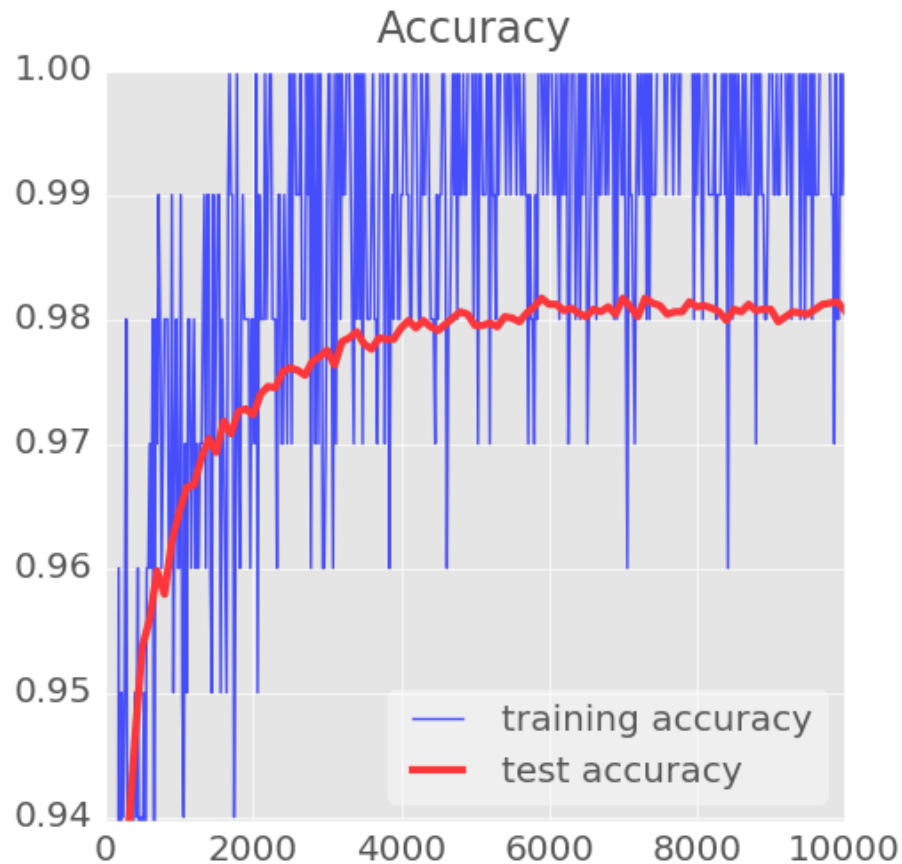


Training
 $p_{\text{keep}} = 0.75$

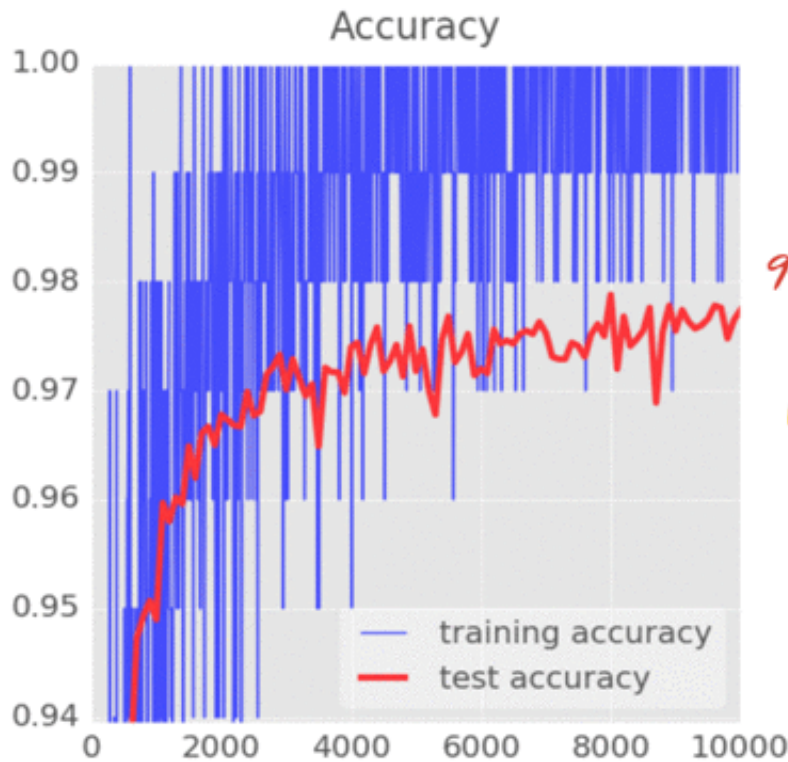


Test
 $p_{\text{keep}} = 1.0$

TensorFlow MNIST Tutorial

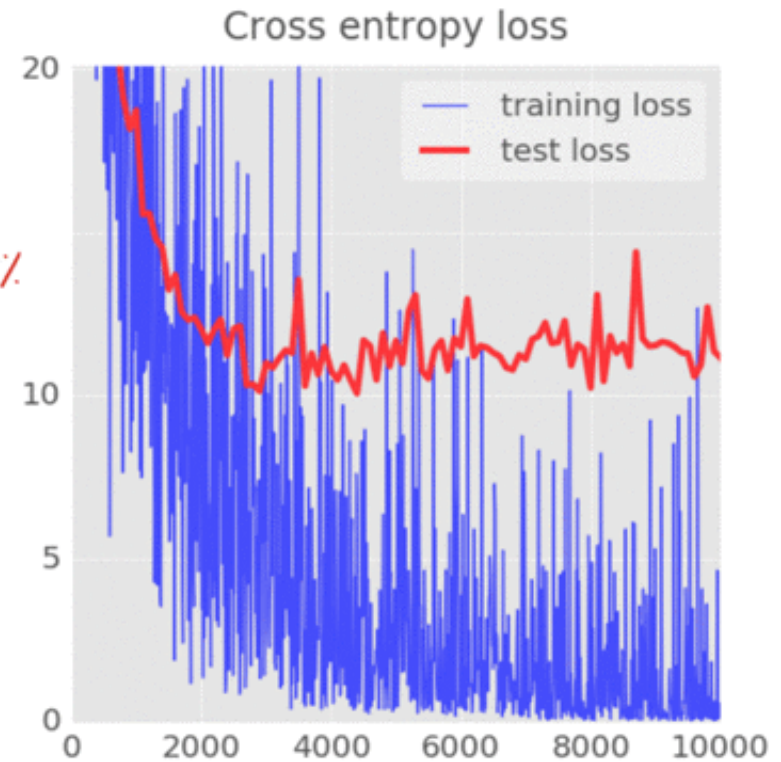


TensorFlow MNIST Tutorial

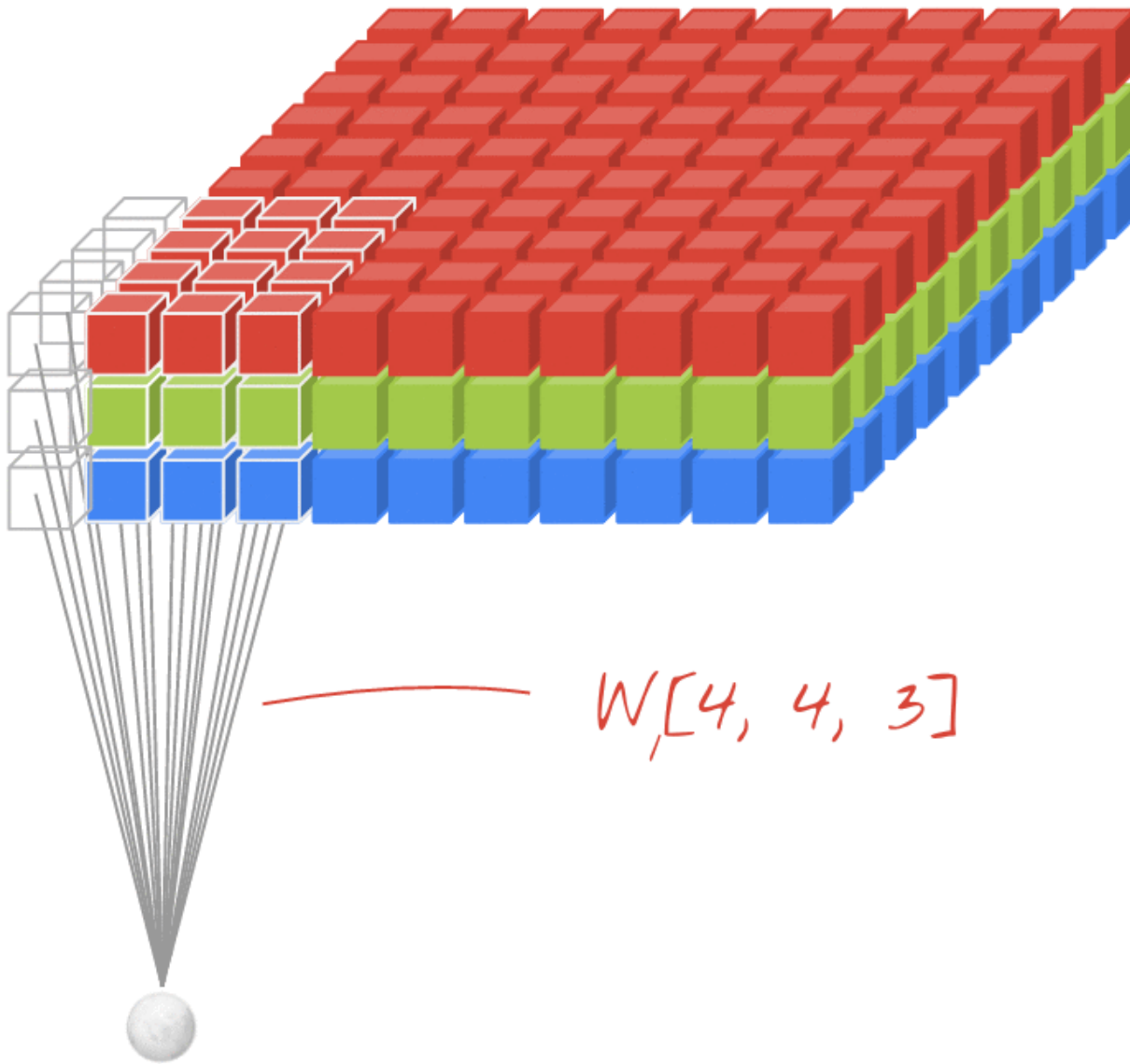


5 layers
sigmoid

97.9%



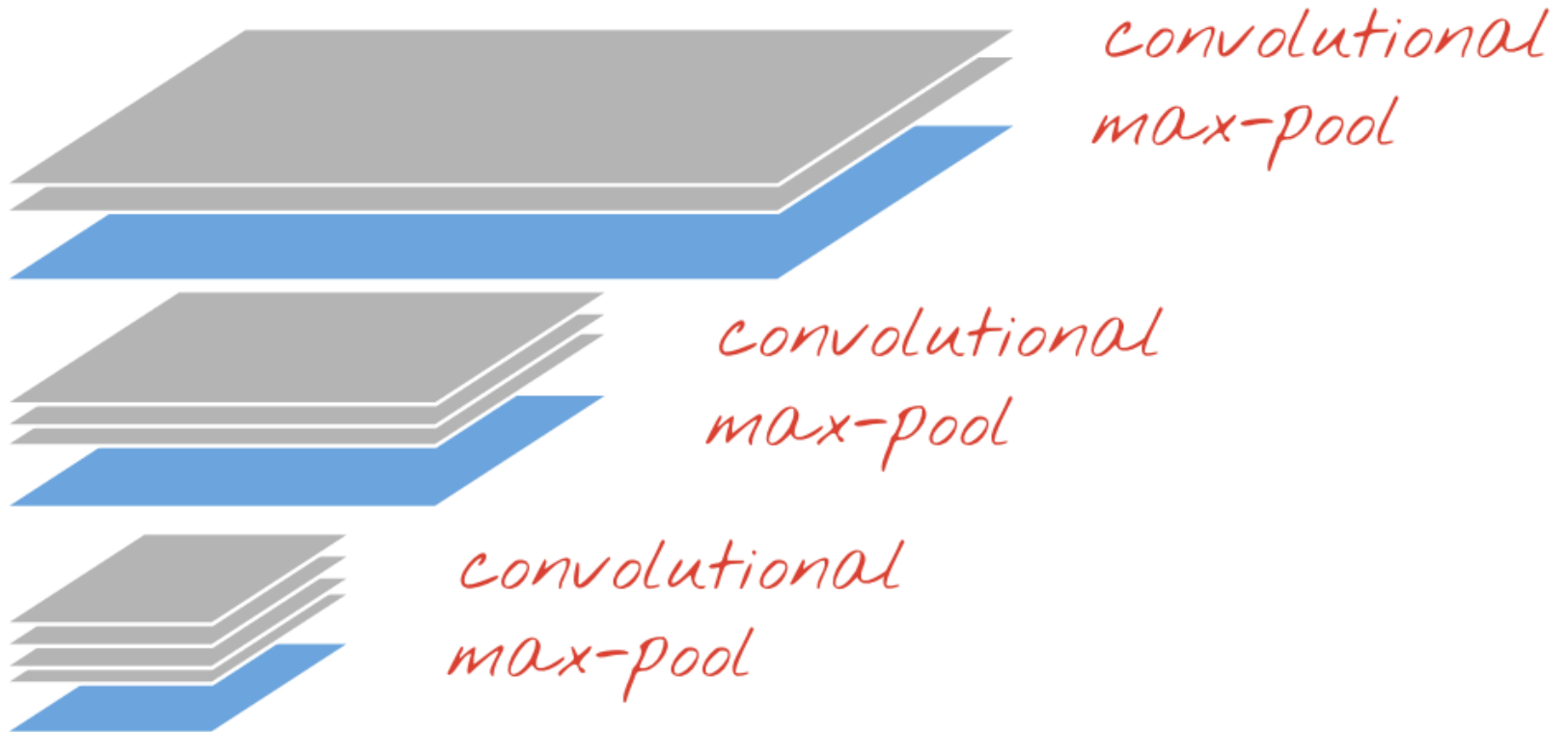




$W_1[4, 4, 3]$
 $W_2[4, 4, 3]$ | $W[4, 4, 3, 2]$

filter size input channels output channels

Convolutional Max-Pool



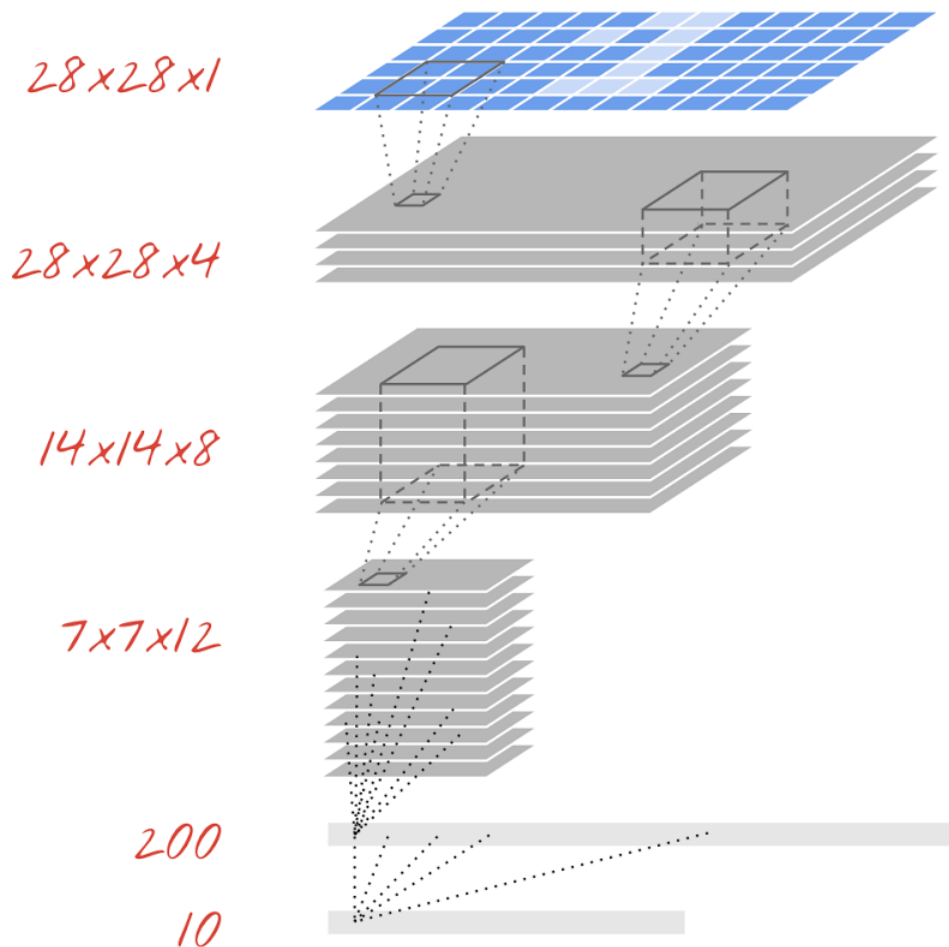
All Convolutional

Hacker's tip



ALL
Convolutional

+ biases on
all layers



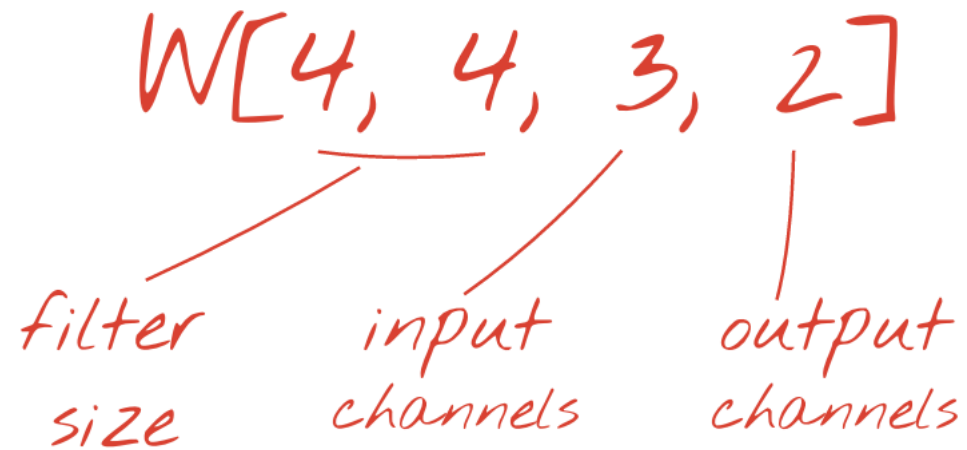
convolutional layer, 4 channels
 $W1[5, 5, 1, 4]$ stride 1

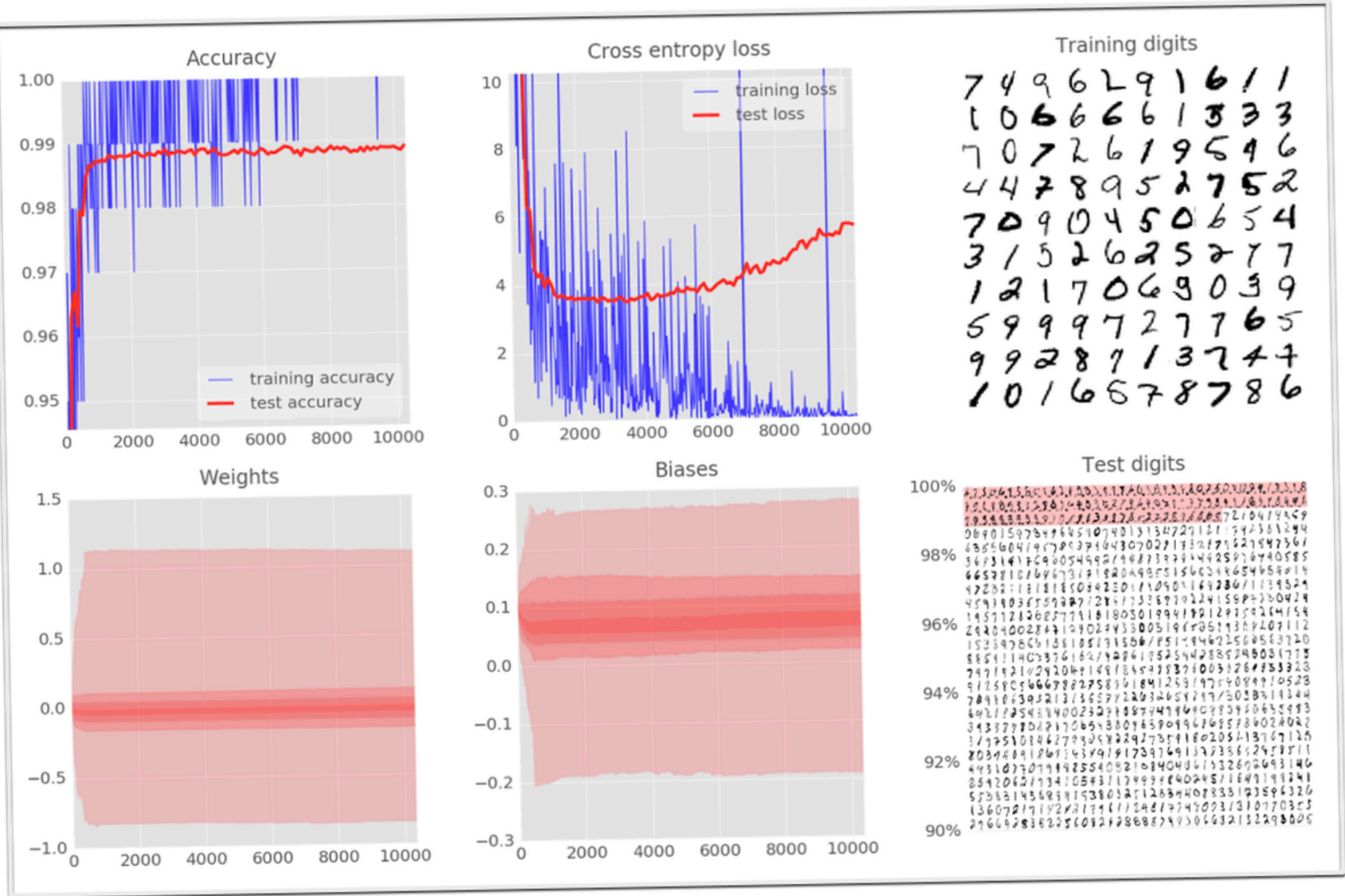
convolutional layer, 8 channels
 $W2[4, 4, 4, 8]$ stride 2

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

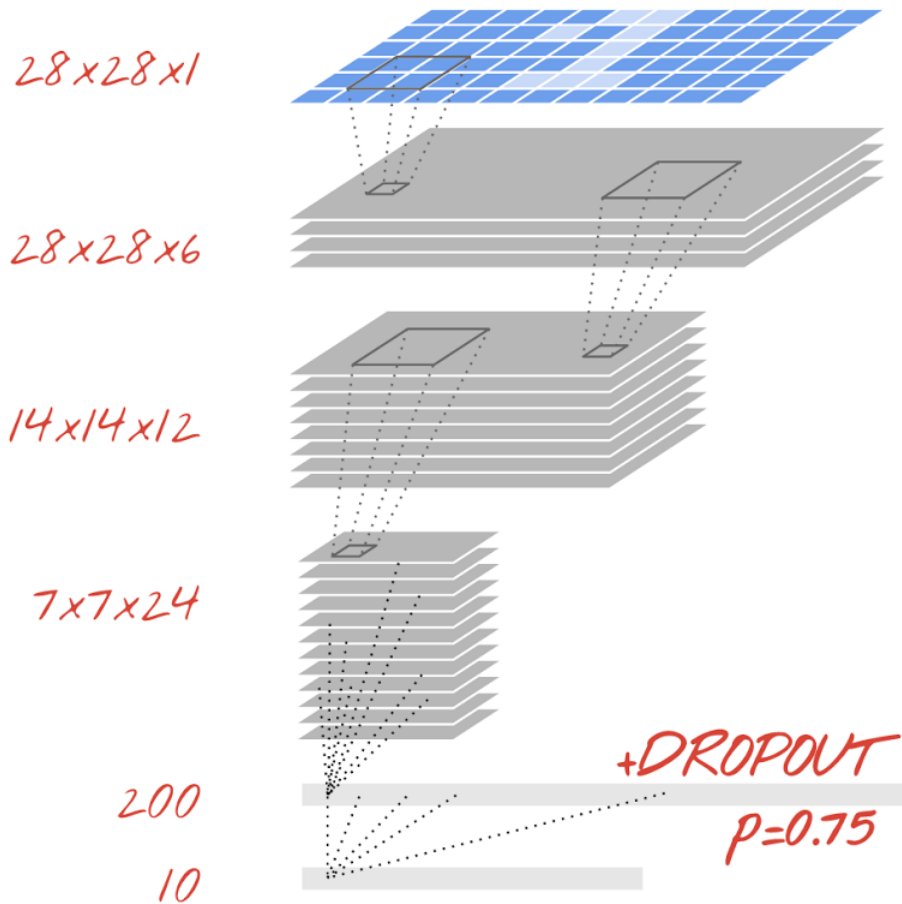
fully connected layer
 $W4[7 \times 7 \times 12, 200]$

softmax readout layer
 $W5[200, 10]$





+ biases on
all layers



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

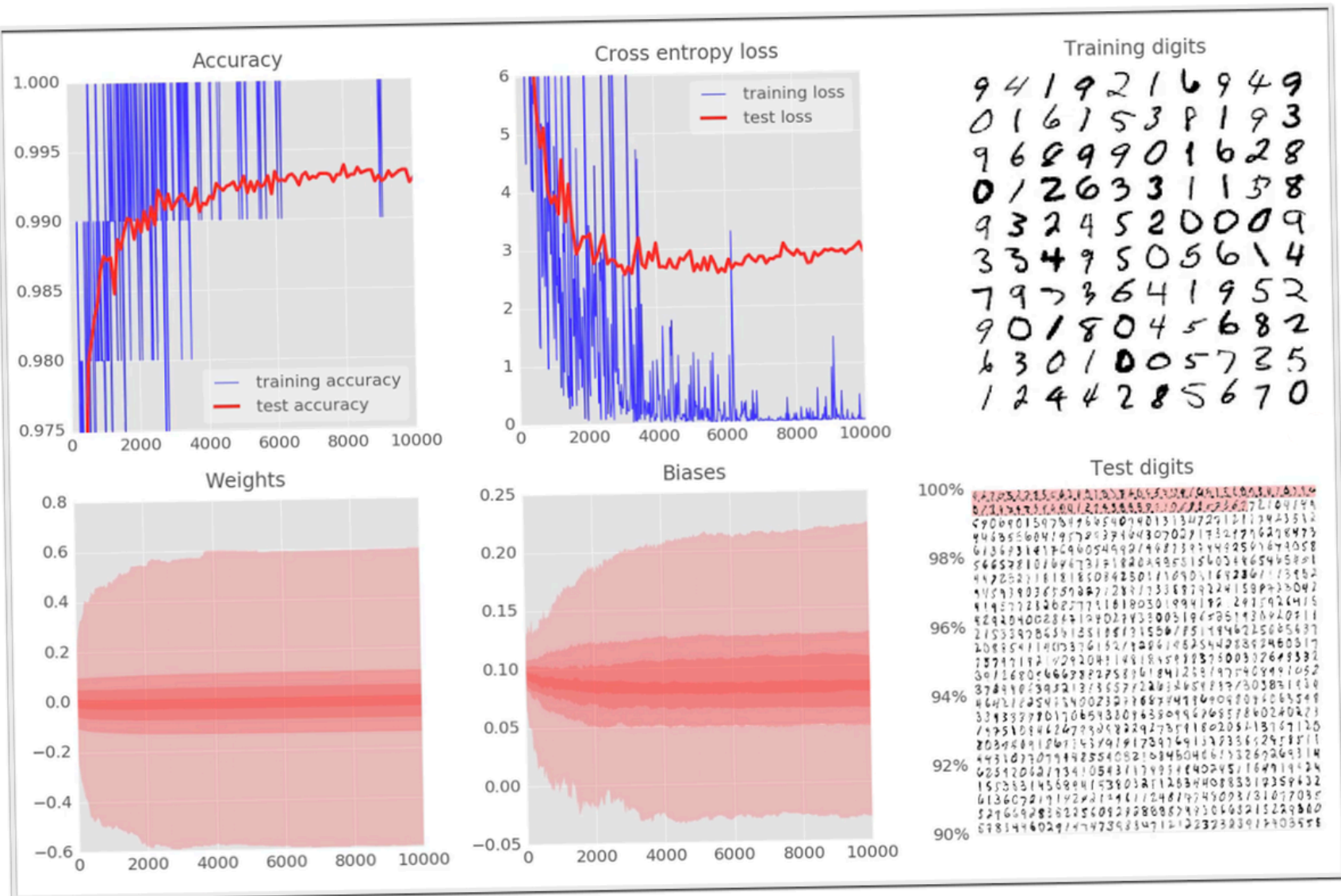
convolutional layer, 12 channels
 $W2[5, 5, 6, 12]$ stride 2

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

fully connected layer
 $W4[7 \times 7 \times 24, 200]$

softmax readout layer
 $W5[200, 10]$

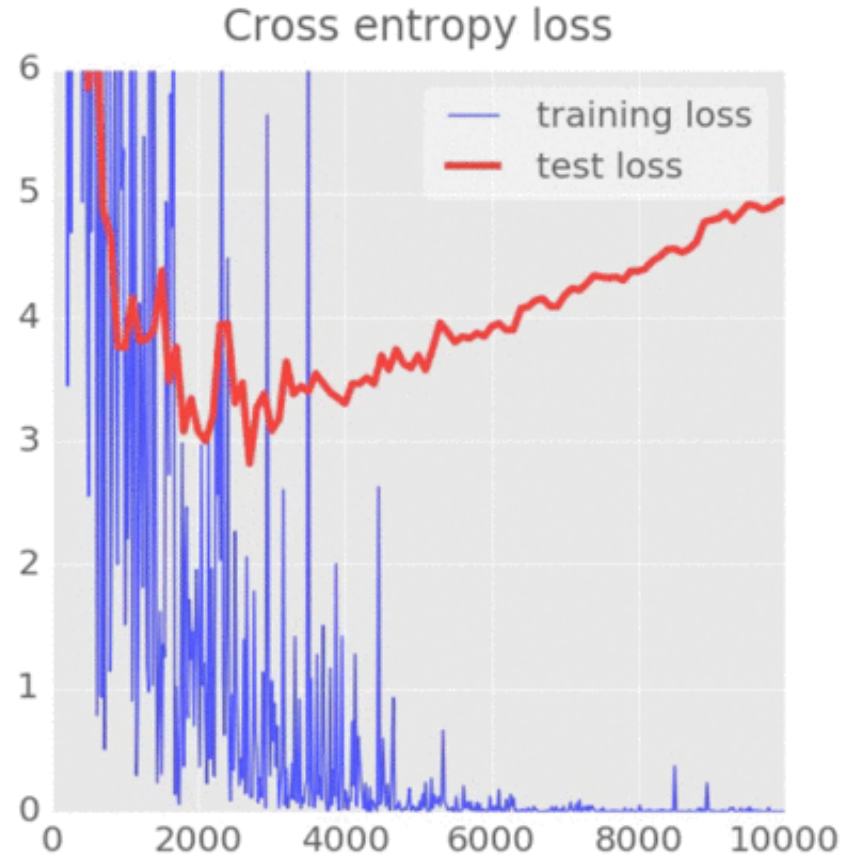
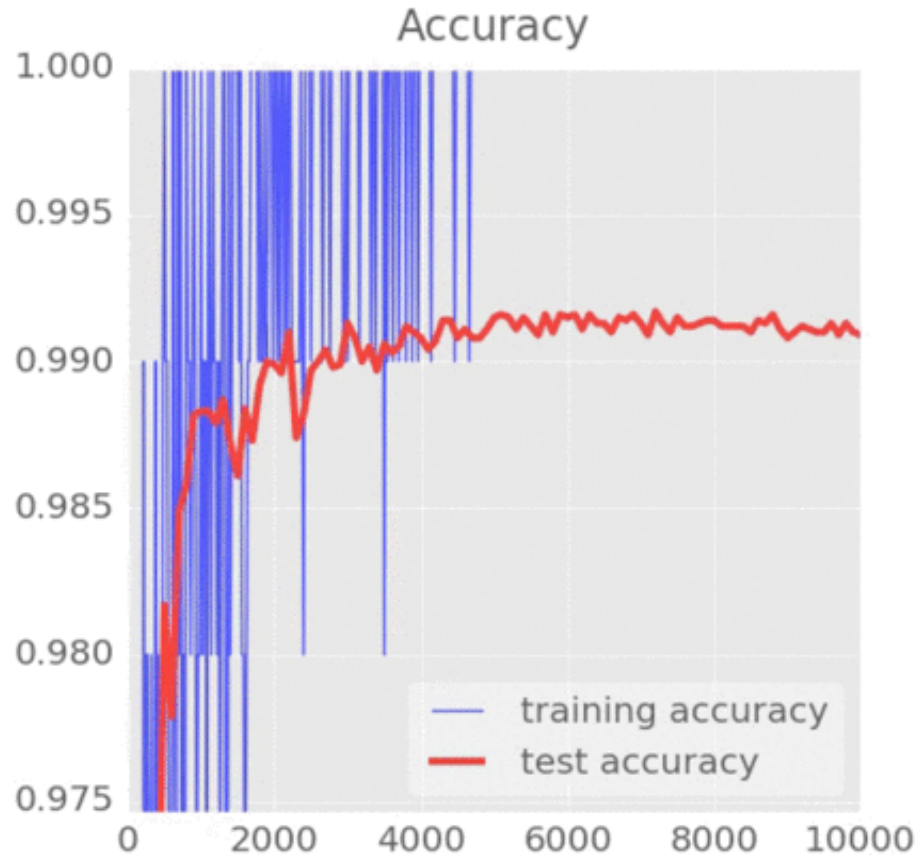
TensorFlow MNIST Tutorial



99.3%



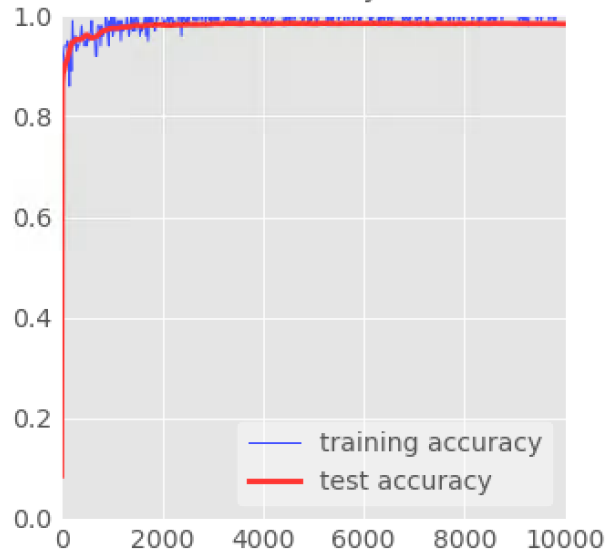
TensorFlow MNIST Tutorial



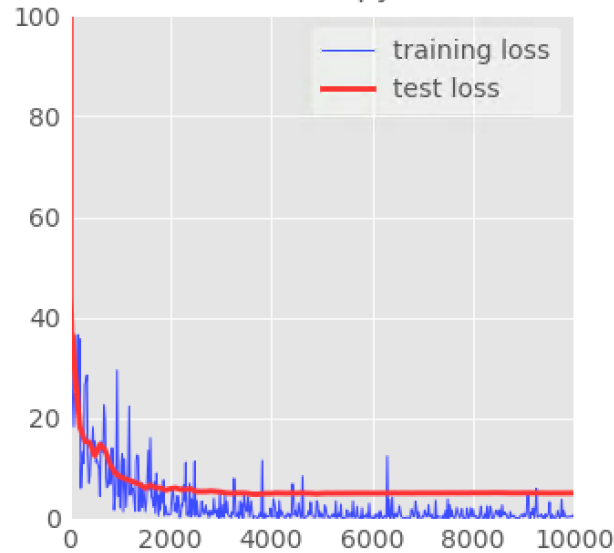
larger convolutional network

TensorFlow MNIST Tutorial

Accuracy



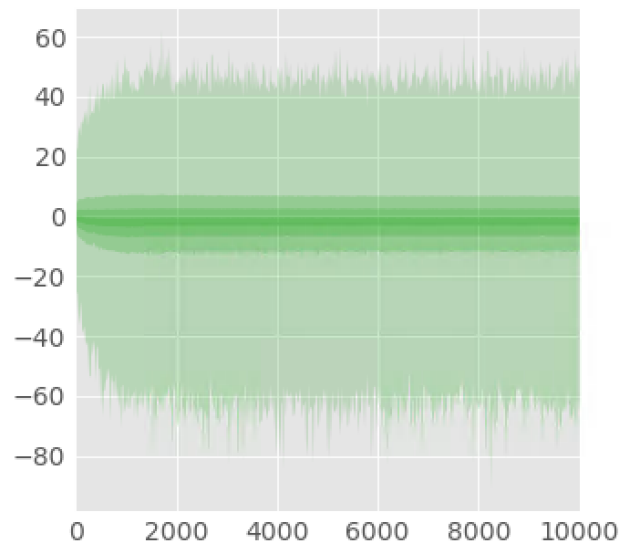
Cross entropy loss



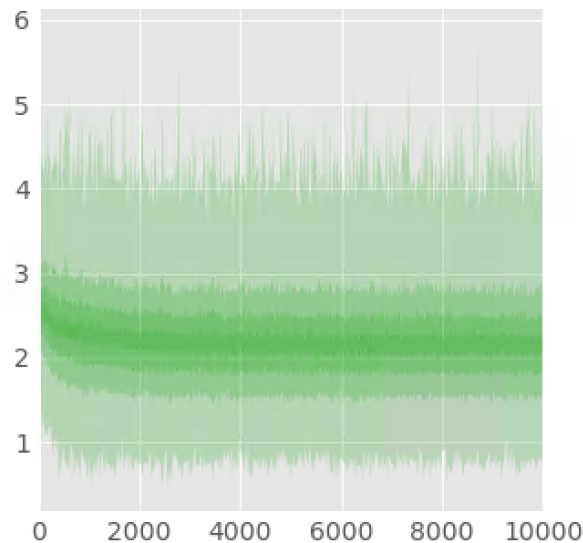
Training digits

6 6 4 3 5 5 0 7 1 0
4 5 4 4 1 2 7 4 3 4
7 4 2 2 4 0 9 2 3 0
9 5 8 0 4 1 4 0 1 0
2 7 5 4 5 7 1 7 5 9
0 9 1 6 4 7 5 5 1 3
7 5 2 9 5 9 2 9 9 4
6 1 8 0 0 7 8 0 2 3
7 1 7 6 0 2 7 1 1 0
5 4 0 6 3 4 4 9 9 3

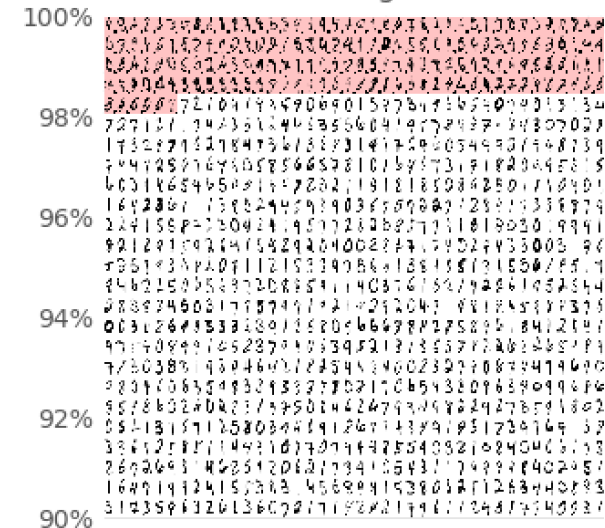
Logits



Max activations across batch



Test digits



TensorFlow MNIST Tutorial



Features Business Explore Pricing

This repository

Search

[Sign in](#) or [Sign up](#)

[martin-gorner](#) / [tensorflow-mnist-tutorial](#)

Watch

50

Star

489

Fork

204

Code

Issues **6**

Pull requests **2**

Projects **0**

Pulse

Graphs

Sample code for "Tensorflow and deep learning, without a PhD" presentation and code lab.

102 commits

1 branch

0 releases

4 contributors

Apache-2.0

Branch: **master** ▾

[New pull request](#)

[Find file](#)

[Clone or download ▾](#)



martin-gorner committed on **GitHub** Update INSTALL.txt ...

Latest commit ed331aa 25 days ago

mlengine	added example using the Tensorflow high level layers API	26 days ago
.gitignore	small bug fix in batch norm	6 months ago
CONTRIBUTING.md	initial commit 2	4 months ago
INSTALL.txt	Update INSTALL.txt	25 days ago
LICENSE	Initial commit	a year ago
README.md	better image URL	3 months ago
mnist_1.0_softmax.py	global_variables_initializer used everywhere instead of initalize_al...	2 months ago
mnist_2.0_five_layers_sigmoid.py	Fix spacing in the network structure comment	a month ago
mnist_2.1_five_layers_relu_lrdecay...	Fix spacing in the network structure comment	a month ago

TensorFlow Playground

Tinker With a **Neural Network** Right Here in Your Browser.
Don't Worry, You Can't Break It. We Promise.



Iterations
000,582

Learning rate

0.03

Activation

Tanh

Regularization

None

Regularization rate

0

Problem type

Classification

DATA

Which dataset do you want to use?



Ratio of training to test data: 50%



Noise: 0



Batch size: 10

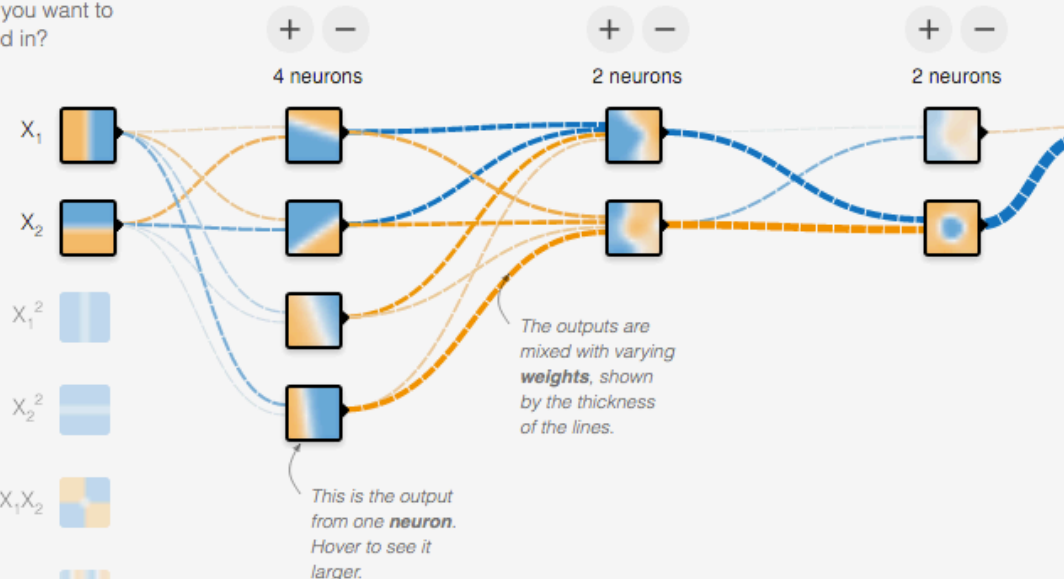


INPUT

Which properties do you want to feed in?

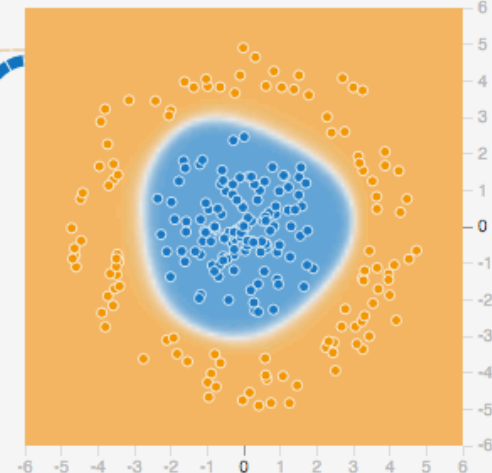


3 HIDDEN LAYERS



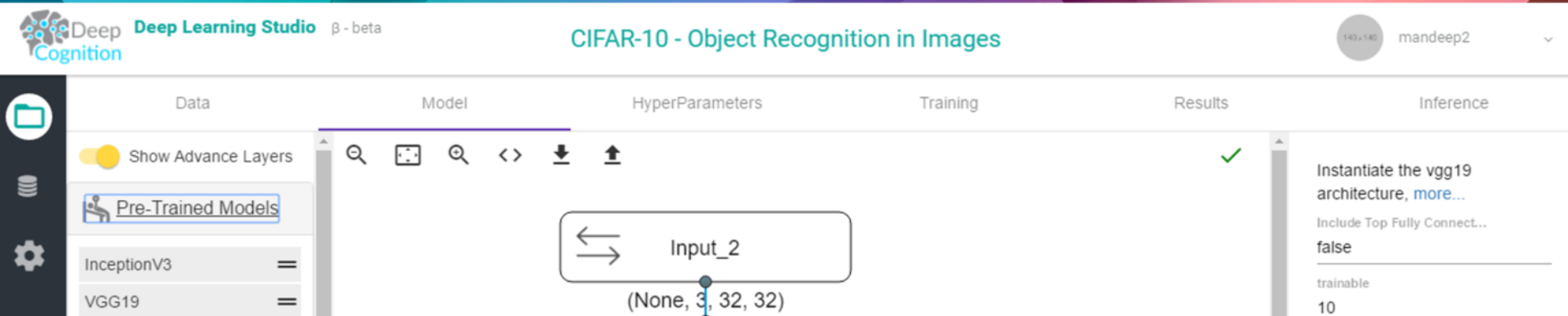
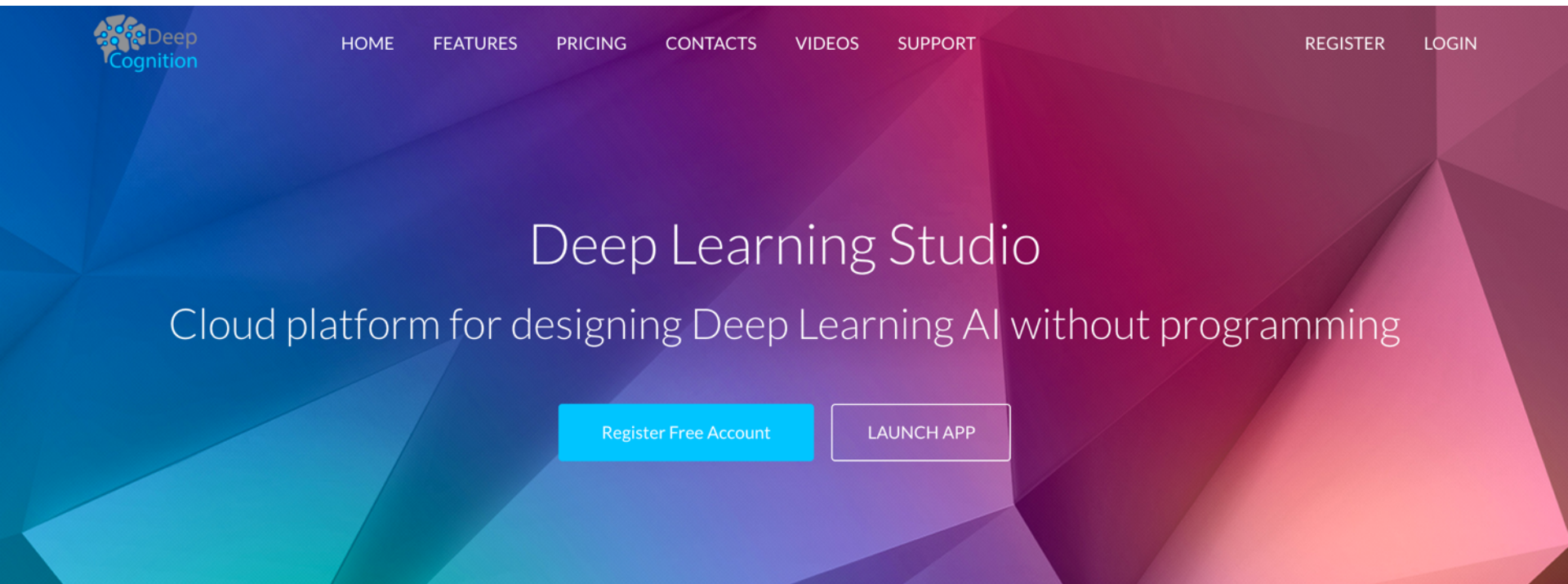
OUTPUT

Test loss 0.000
Training loss 0.000



Deep Learning Studio

Cloud platform for designing Deep Learning AI without programming



Deep Learning Studio

Cloud platform for designing Deep Learning AI without programming

 Deep Learning Studio β - beta

CIFAR-10 - Object Recognition in Images

140 x 140 mandeep2



Show Advance Layers

Pre-Trained Models

InceptionV3

VGG19

VGG16

ResNet50

Special Functions

Convolutional Layers

Core Layers

Pooling Layers

Recurrent Layers

Advanced Activations Layers

Convolutional Recurrent Layers

Noise Layers

Model

HyperParameters

Training

Results

Inference

↔ Input_2

(None, 3, 32, 32)

👤 VGG19_1

(None, 512, 1, 1)

📊 Flatten_2

(None, 512)

📊 Dense_5

(None, 100)

📊 Dropout_1

(None, 100)

📊 Dense_3

(None, 10)

↔ Output_2

(None, 10)

Instantiate the vgg19 architecture, [more...](#)

Include Top Fully Connect...

false

trainable

10

☐ Show Advance Options

<http://deepcognition.ai/>



Deep Learning Studio

Cloud platform for designing Deep Learning AI without programming

eta

MNIST Handwritten Digits Classifier

Model

HyperParameters

Training

Results

Dataset Source: Testing

Training Run: Run0

Start Inference

or

Download Trained Model

Digit Label	Image	predictions
• 9		• 9
• 1		• 1
• 1		• 1
• 5		• 3
• 0		• 0
• 5		• 5
• 1		• 1
• 2		• 6
• 2		• 2
• 3		• 3

« Previous

1

2

3

4

5

...

351

Next »

Download Results

References

- Sunila Gollapudi (2016), Practical Machine Learning, Packt Publishing
- Sebastian Raschka (2015), Python Machine Learning, Packt Publishing
- Martin Gorner (2017), TensorFlow and Deep Learning without a PhD, Part 1 (Google Cloud Next '17), <https://www.youtube.com/watch?v=u4aGIomYP4>
- Martin Gorner (2017), TensorFlow and Deep Learning without a PhD, Part 2 (Google Cloud Next '17), <https://www.youtube.com/watch?v=fTUwdXUffl8>
- Martin Gorner (2017), TensorFlow and Deep Learning without a PhD, <https://goo.gl/pHeXe7>, <https://codelabs.developers.google.com/codelabs/cloud-tensorflow-mnist>
- Deep Learning Basics: Neural Networks Demystified, <https://www.youtube.com/playlist?list=PLiaHhY2iBX9hdHaRr6b7XevZtgZRa1PoU>
- Deep Learning SIMPLIFIED, <https://www.youtube.com/playlist?list=PLjJh1vISEYgvGod9wWiydumYl8hOXixNu>
- TensorFlow: <https://www.tensorflow.org/>
- Theano: <http://deeplearning.net/software/theano/>
- Keras: <http://keras.io/>
- Deep Learning Studio: Cloud platform for designing Deep Learning AI without programming, <http://deepcognition.ai/>
- <http://p.migdal.pl/2017/04/30/teaching-deep-learning.html>
- <https://github.com/leriomaggio/deep-learning-keras-tensorflow>