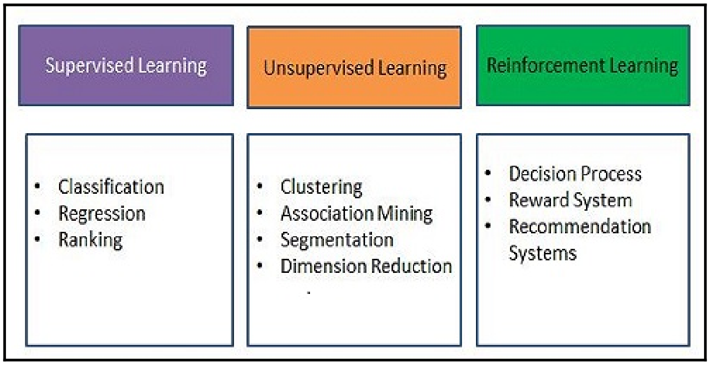
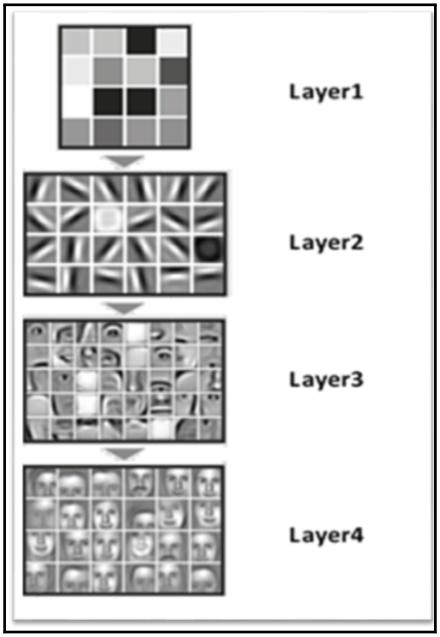
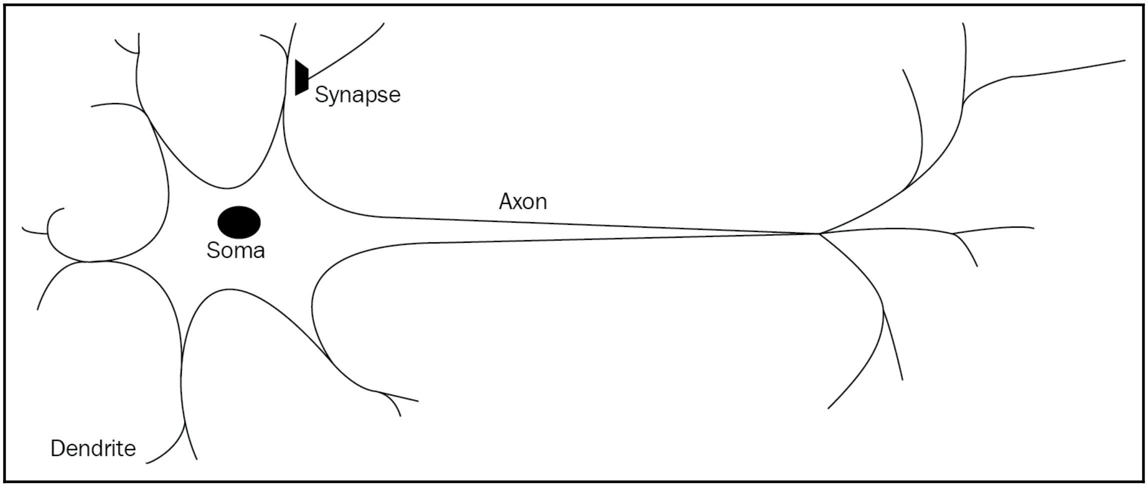
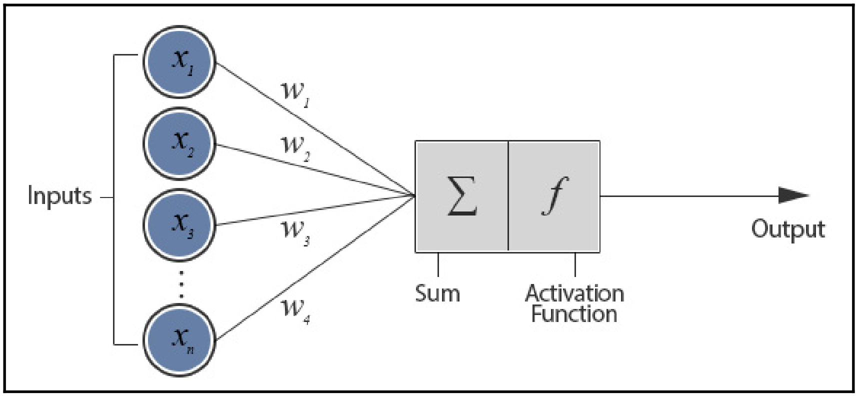
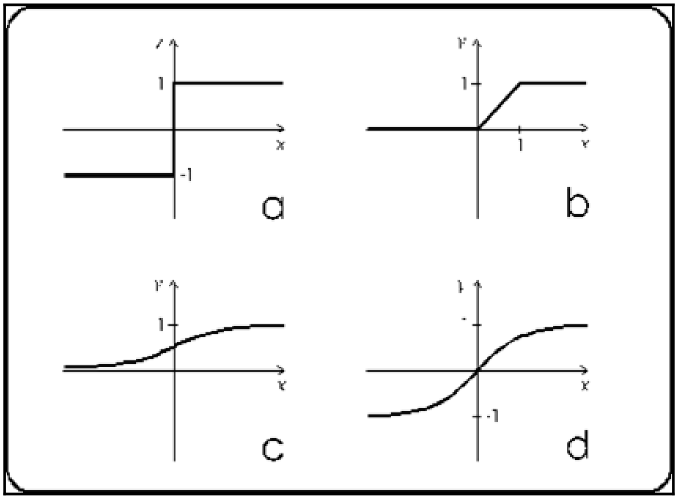
Chapter 1: Getting Started with Deep Learning

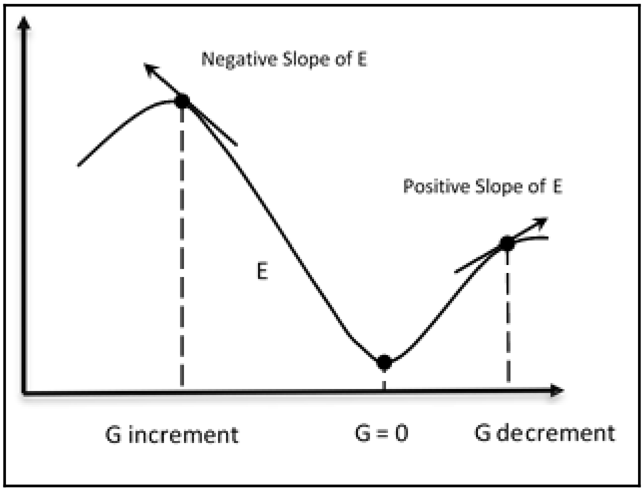


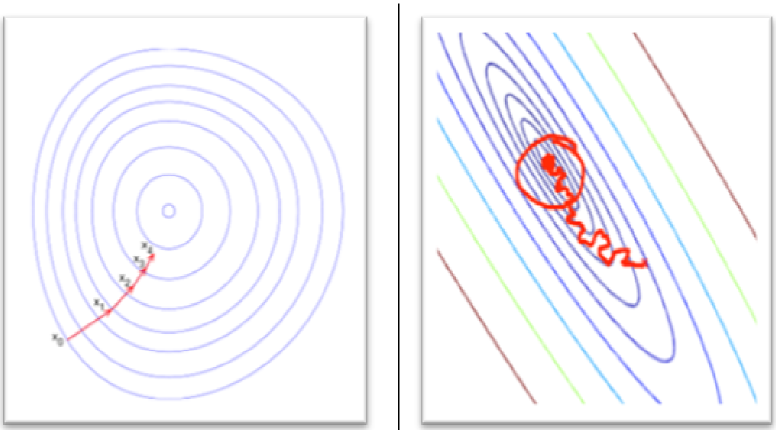



Synapse
Axon
Soma
Dendrite


x_1
x_2
x_3
x_n
w_1
w_2
w_3
w_4
Inputs
Σ
f
Sum
Activation Function
Output

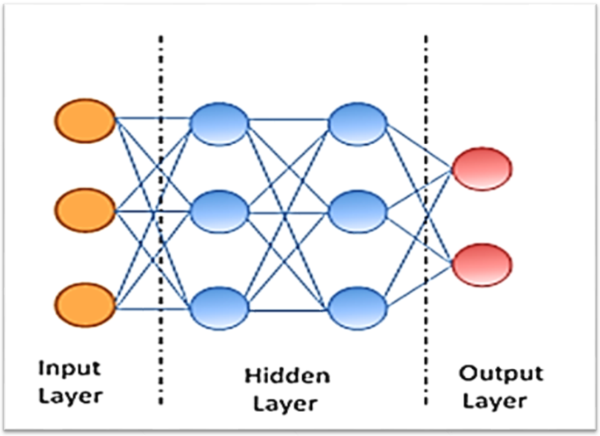

1
-1
x
a
1
1
x
b
1
x
c
-1
x
d

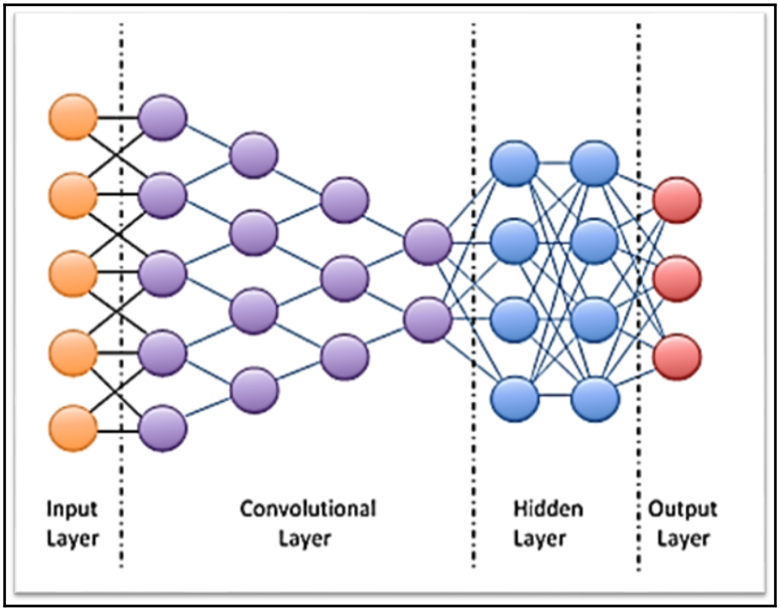

Negative Slope of E
Positive Slope of E
E
G increment
G = 0
G decrement

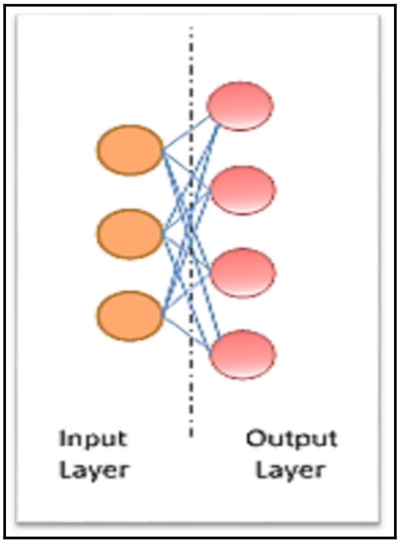


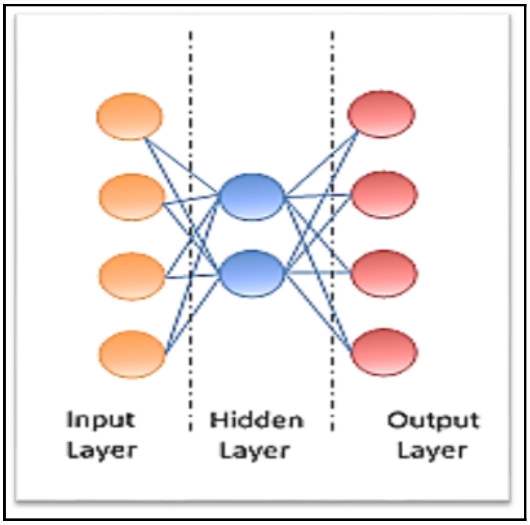
(a) Gradient Descent - GD

(b) Stochastic Gradient Descent - SGD




Input Layer
Convolutional Layer
Hidden Layer
Output Layer

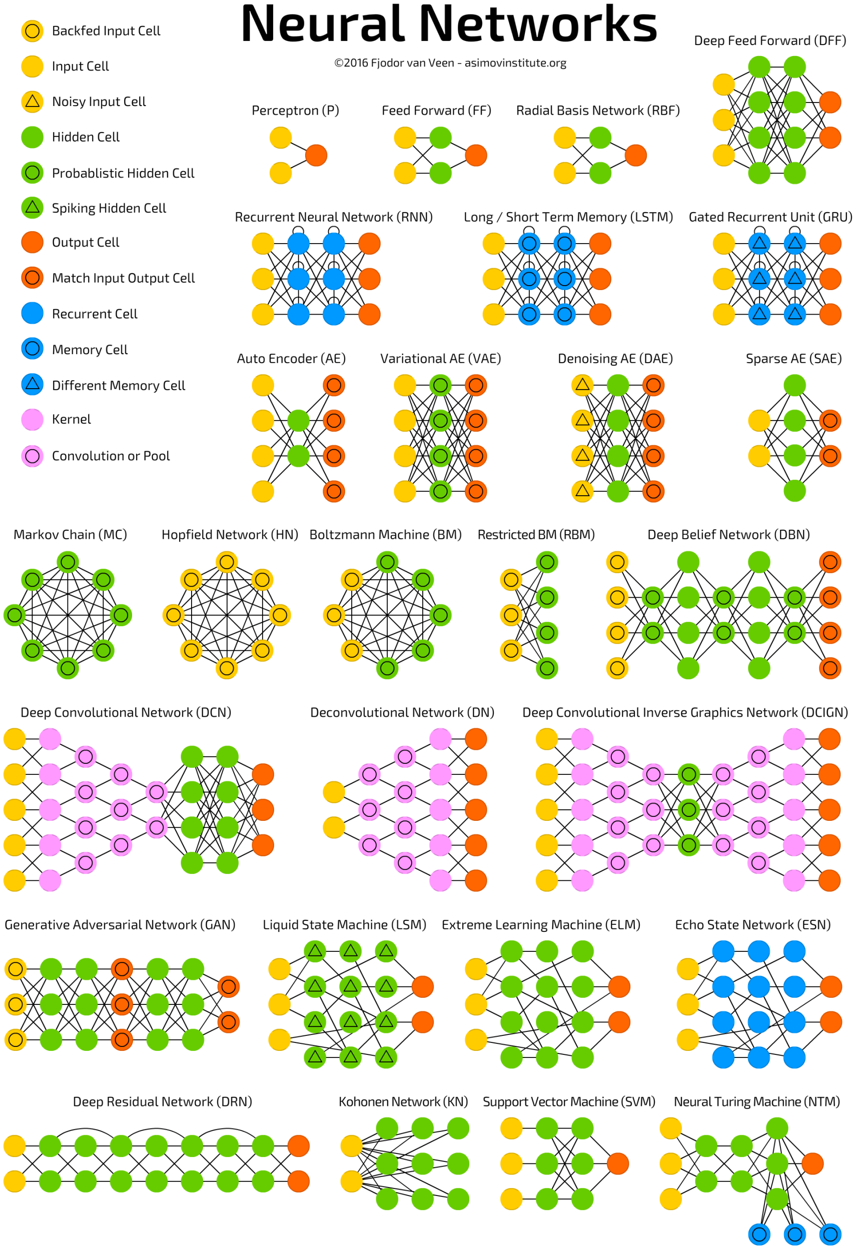

Input Layer
Output Layer


Input Layer
Hidden Layer
Output Layer

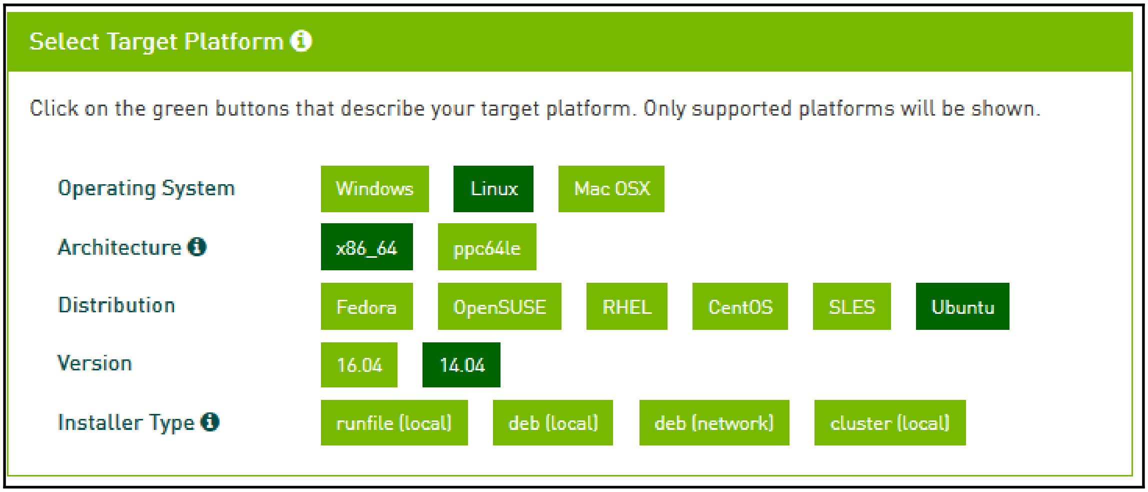
A mostly complete chart of

Neural Networks

©2016 Fjodor van Veen - asimovinstitute.org



Chapter 2: First Look at TensorFlow



Select Target Platform

Click on the green buttons that describe your target platform. Only supported platforms will be shown.

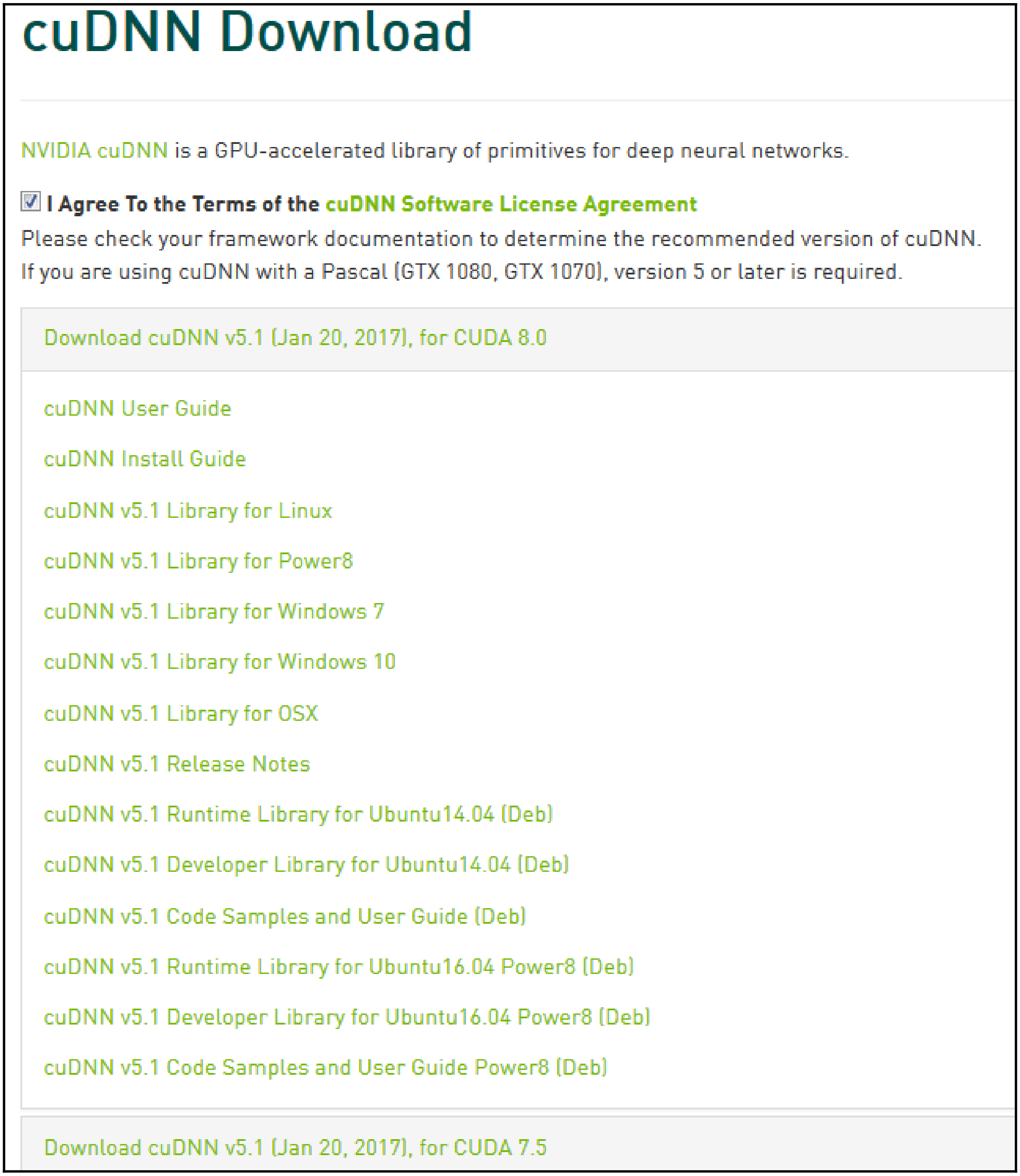
Operating System: Windows [✓ Linux] Mac OSX

Architecture: [✓ x86_64] ppc64le

Distribution: Fedora OpenSUSE RHEL CentOS SLES [✓ Ubuntu]

Version: 16.04 [✓ 14.04]

Installer Type: runfile (local) deb (local) deb (network) cluster (local)



cuDNN Download

NVIDIA cuDNN is a GPU-accelerated library of primitives for deep neural networks.

☑ **I Agree To the Terms of the cuDNN Software License Agreement**

Please check your framework documentation to determine the recommended version of cuDNN.
If you are using cuDNN with a Pascal (GTX 1080, GTX 1070), version 5 or later is required.

Download cuDNN v5.1 (Jan 20, 2017), for CUDA 8.0

cuDNN User Guide

cuDNN Install Guide

cuDNN v5.1 Library for Linux

cuDNN v5.1 Library for Power8

cuDNN v5.1 Library for Windows 7

cuDNN v5.1 Library for Windows 10

cuDNN v5.1 Library for OSX

cuDNN v5.1 Release Notes

cuDNN v5.1 Runtime Library for Ubuntu14.04 (Deb)

cuDNN v5.1 Developer Library for Ubuntu14.04 (Deb)

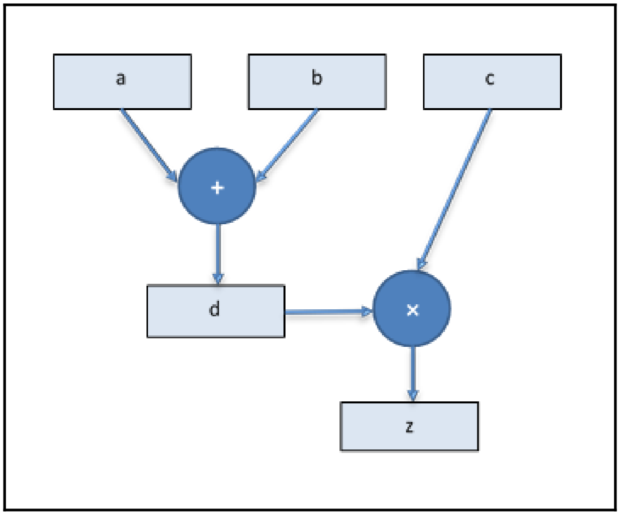
cuDNN v5.1 Code Samples and User Guide (Deb)

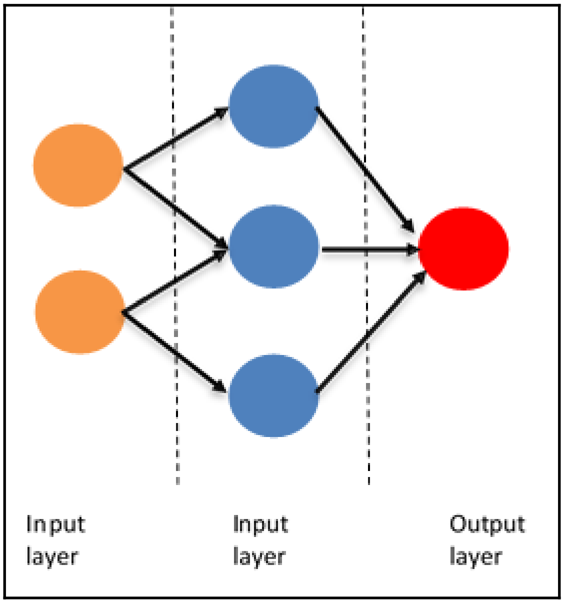
cuDNN v5.1 Runtime Library for Ubuntu16.04 Power8 (Deb)

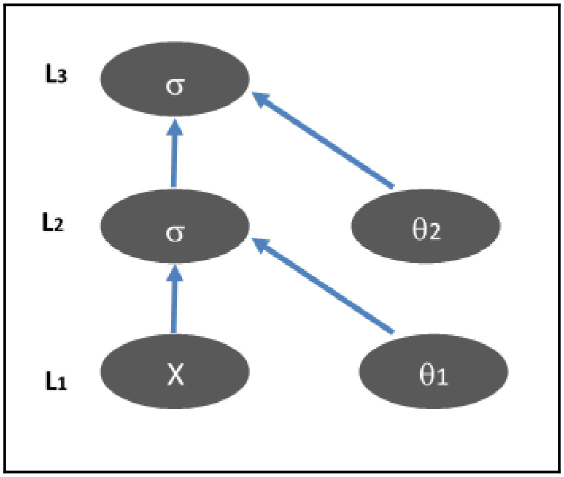
cuDNN v5.1 Developer Library for Ubuntu16.04 Power8 (Deb)

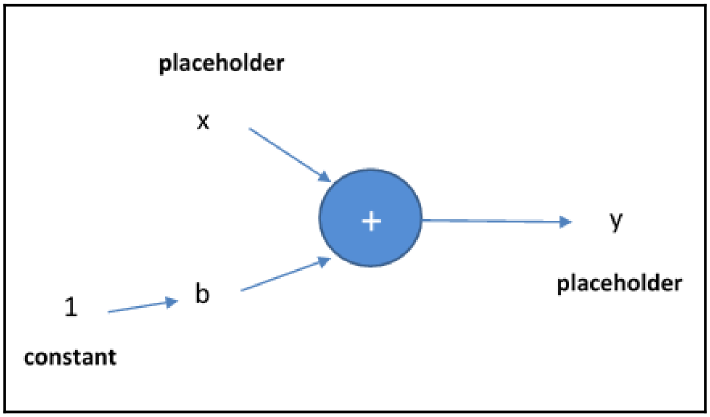
cuDNN v5.1 Code Samples and User Guide Power8 (Deb)

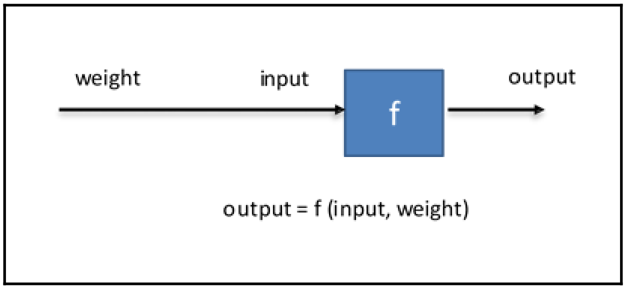
Download cuDNN v5.1 (Jan 20, 2017), for CUDA 7.5

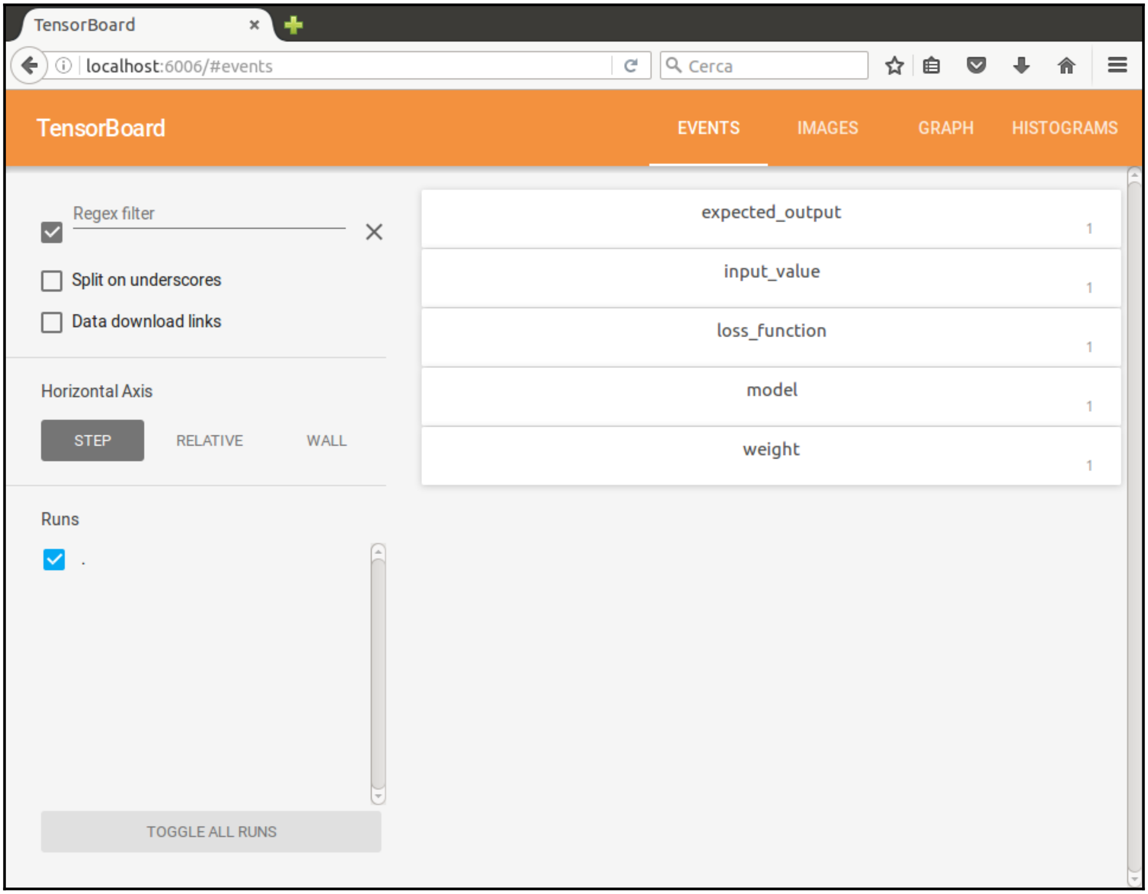

a
b
c
+
d
×
z


Input
layer
Input
layer
Output
layer


L3
σ
L2
σ
θ2
L1
X
θ1


placeholder
x
+
y
placeholder
1
b
constant


weight
input
f
output
output = f (input, weight)



☑ Regex filter

□ Split on underscores

□ Data download links

Horizontal Axis: [✓ STEP] RELATIVE WALL

Runs

☑ .

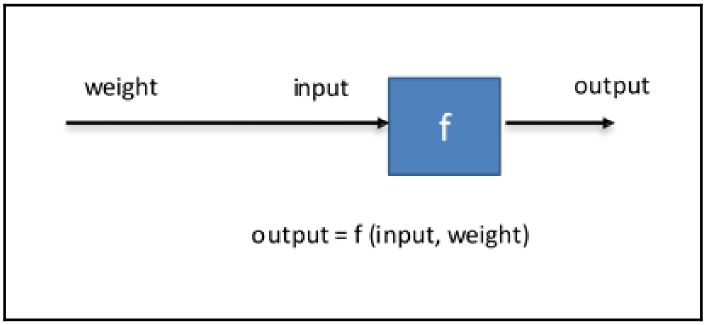
expected_output 1

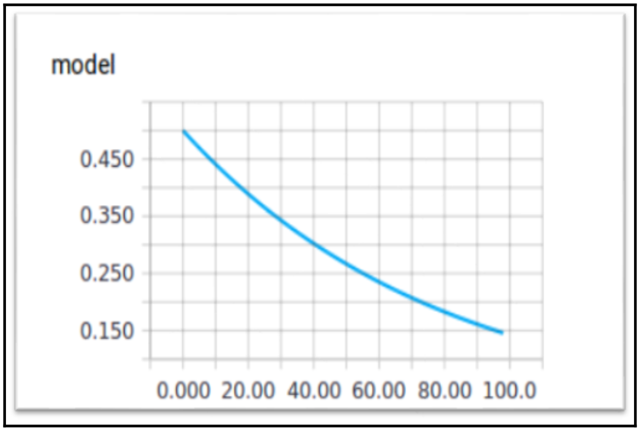
input_value 1

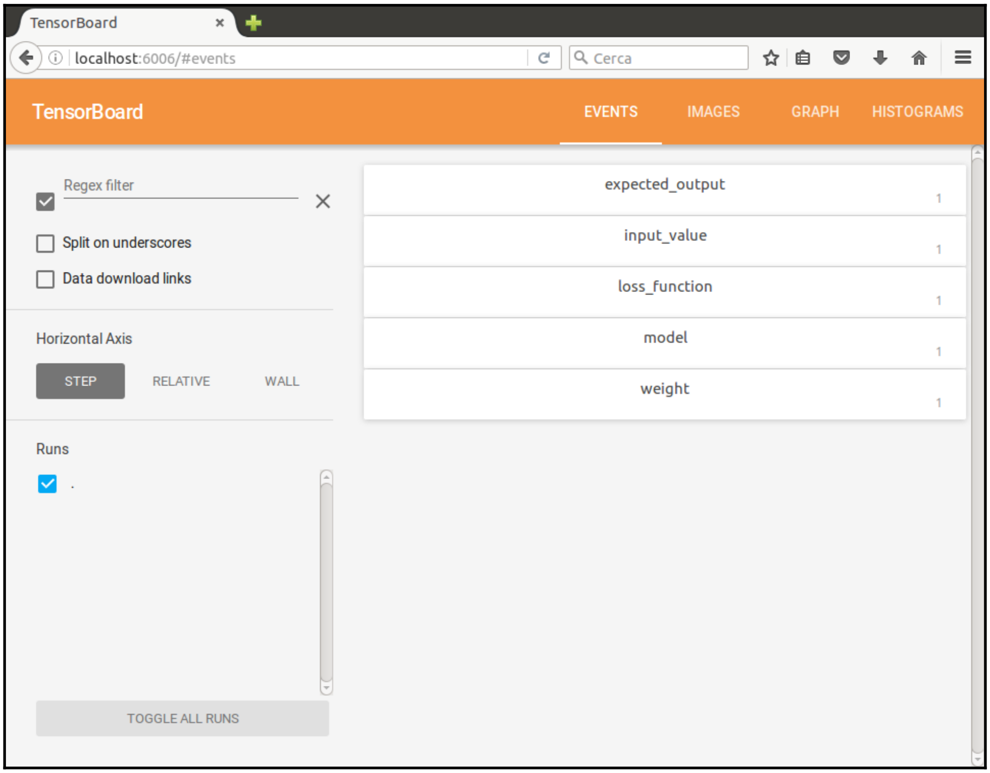
loss_function 1

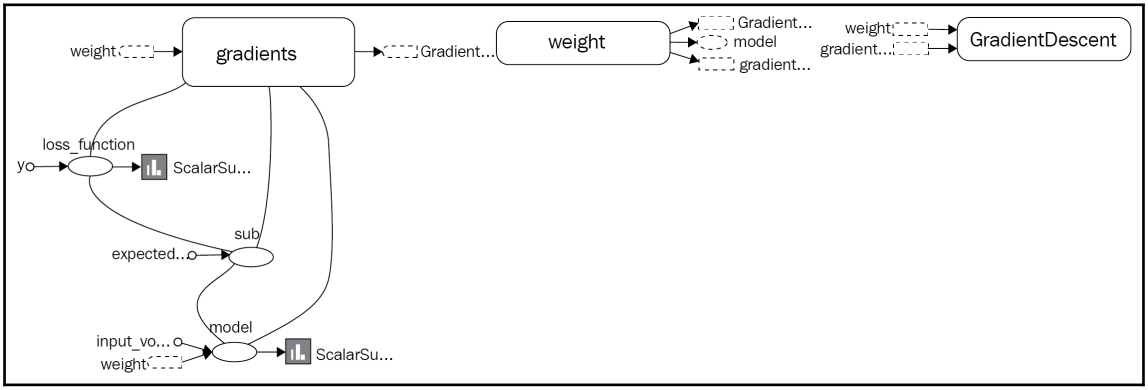
model 1

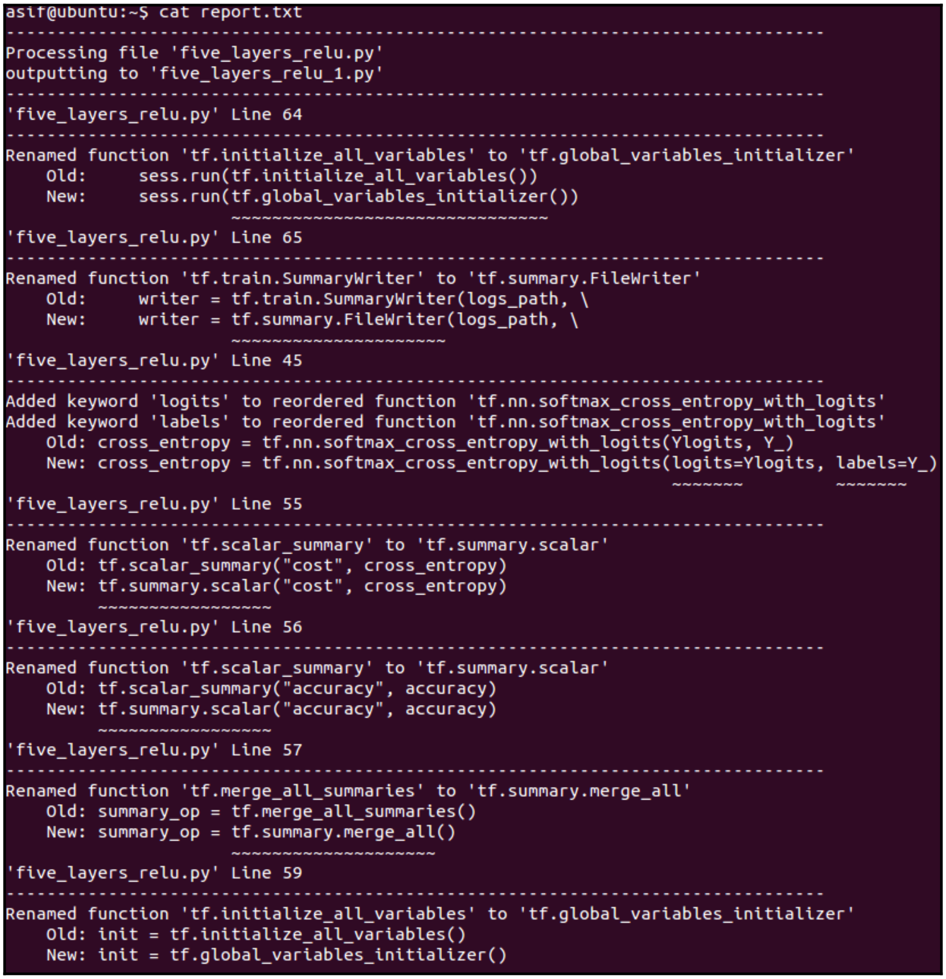
weight 1



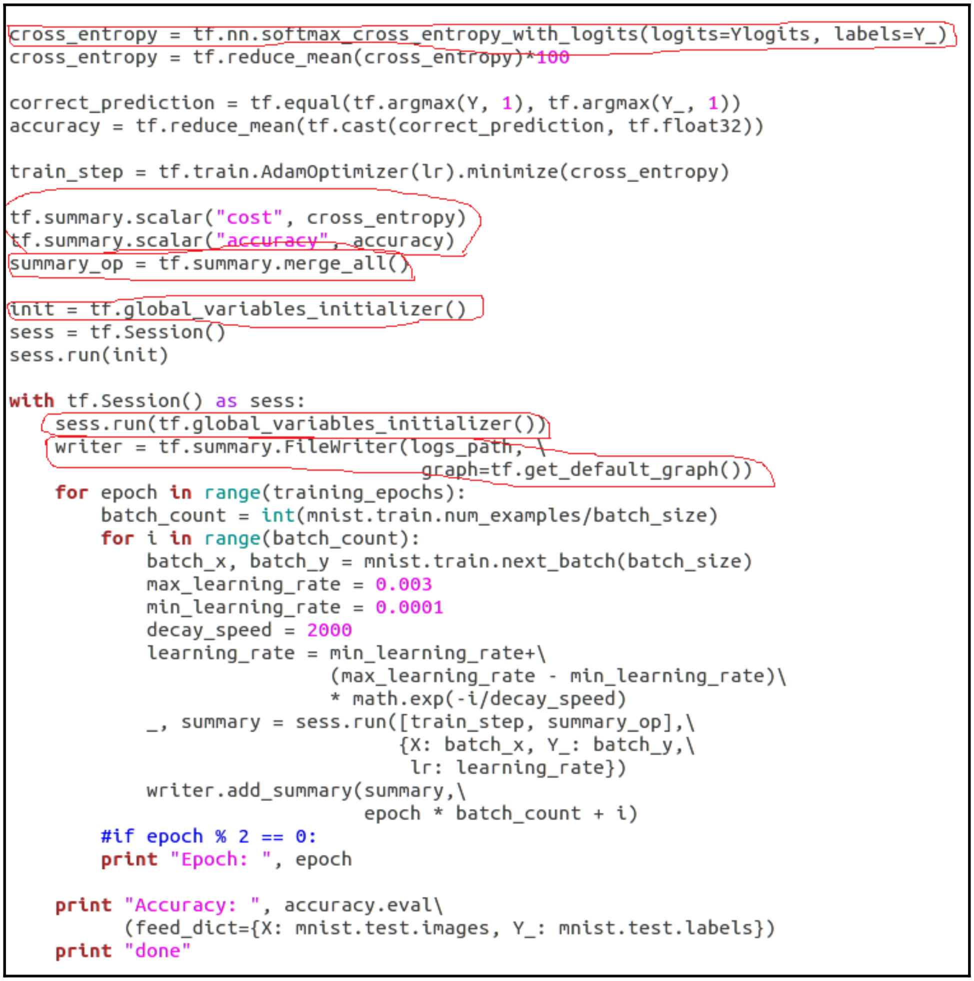

model
0.450
0.350
0.250
0.150
0.000
20.00
40.00
60.00
80.00
100.0


TensorBoard
localhost:6006/#events
Cerca
TensorBoard
EVENTS
IMAGES
GRAPH
HISTOGRAMS
☑ Regex filter
□ Split on underscores
□ Data download links
Horizontal Axis
STEP
RELATIVE
WALL
Runs
☑ .
TOGGLE ALL RUNS
expected_output 1
input_value 1
loss_function 1
model 1
weight 1



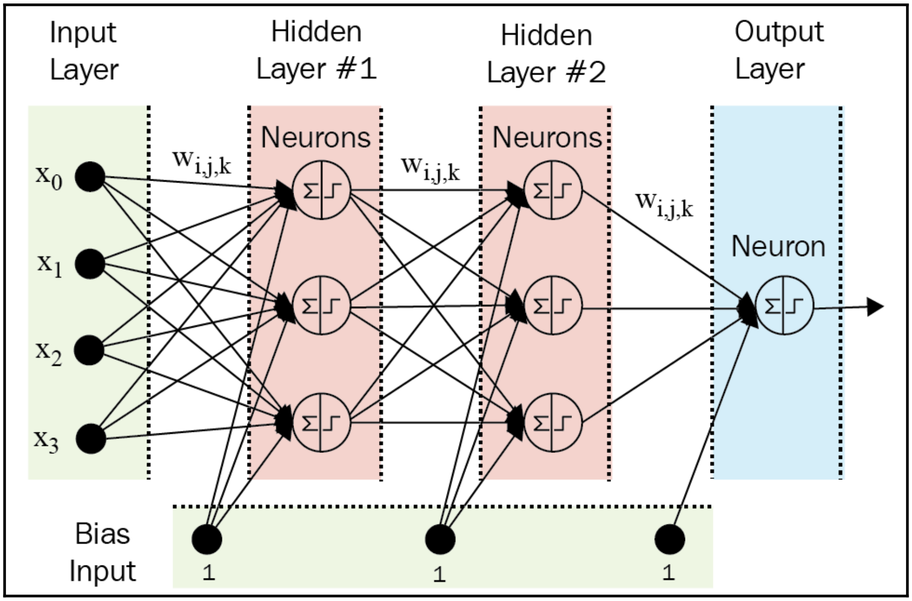


```
asif@ubuntu:~$ cat report.txt
--------------------------------------------------------------------------------
Processing file 'five_layers_relu.py'
 outputting to 'five_layers_relu_1.py'
--------------------------------------------------------------------------------
'five_layers_relu.py' Line 64
--------------------------------------------------------------------------------
Renamed function 'tf.initialize_all_variables' to 'tf.global_variables_initializer'
    Old:     sess.run(tf.initialize_all_variables())
    New:     sess.run(tf.global_variables_initializer())
                      ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
'five_layers_relu.py' Line 65
--------------------------------------------------------------------------------
Renamed function 'tf.train.SummaryWriter' to 'tf.summary.FileWriter'
    Old:     writer = tf.train.SummaryWriter(logs_path, \
    New:     writer = tf.summary.FileWriter(logs_path, \
                      ~~~~~~~~~~~~~~~~~~~~~
'five_layers_relu.py' Line 45
--------------------------------------------------------------------------------
Added keyword 'logits' to reordered function 'tf.nn.softmax_cross_entropy_with_logits'
Added keyword 'labels' to reordered function 'tf.nn.softmax_cross_entropy_with_logits'
    Old: cross_entropy = tf.nn.softmax_cross_entropy_with_logits(Ylogits, Y_)
    New: cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits=Ylogits, labels=Y_)
                                                                 ~~~~~~~         ~~~~~~~
'five_layers_relu.py' Line 55
--------------------------------------------------------------------------------
Renamed function 'tf.scalar_summary' to 'tf.summary.scalar'
    Old: tf.scalar_summary("cost", cross_entropy)
    New: tf.summary.scalar("cost", cross_entropy)
         ~~~~~~~~~~~~~~~~~
'five_layers_relu.py' Line 56
--------------------------------------------------------------------------------
Renamed function 'tf.scalar_summary' to 'tf.summary.scalar'
    Old: tf.scalar_summary("accuracy", accuracy)
    New: tf.summary.scalar("accuracy", accuracy)
         ~~~~~~~~~~~~~~~~~
'five_layers_relu.py' Line 57
--------------------------------------------------------------------------------
Renamed function 'tf.merge_all_summaries' to 'tf.summary.merge_all'
    Old: summary_op = tf.merge_all_summaries()
    New: summary_op = tf.summary.merge_all()
                      ~~~~~~~~~~~~~~~~~~~~
'five_layers_relu.py' Line 59
--------------------------------------------------------------------------------
Renamed function 'tf.initialize_all_variables' to 'tf.global_variables_initializer'
    Old: init = tf.initialize_all_variables()
    New: init = tf.global_variables_initializer()
```



```
cross_entropy = tf.nn.softmax_cross_entropy_with_logits(logits=Ylogits, labels=Y_)
cross_entropy = tf.reduce_mean(cross_entropy)*100

correct_prediction = tf.equal(tf.argmax(Y, 1), tf.argmax(Y_, 1))
accuracy = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))

train_step = tf.train.AdamOptimizer(lr).minimize(cross_entropy)

tf.summary.scalar("cost", cross_entropy)
tf.summary.scalar("accuracy", accuracy)
summary_op = tf.summary.merge_all()

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

with tf.Session() as sess:
    sess.run(tf.global_variables_initializer())
    writer = tf.summary.FileWriter(logs_path, \
                                    graph=tf.get_default_graph())
    for epoch in range(training_epochs):
        batch_count = int(mnist.train.num_examples/batch_size)
        for i in range(batch_count):
            batch_x, batch_y = mnist.train.next_batch(batch_size)
            max_learning_rate = 0.003
            min_learning_rate = 0.0001
            decay_speed = 2000
            learning_rate = min_learning_rate+\
                            (max_learning_rate - min_learning_rate)\
                            * math.exp(-i/decay_speed)
            _, summary = sess.run([train_step, summary_op],\
                                  {X: batch_x, Y_: batch_y,\
                                   lr: learning_rate})
            writer.add_summary(summary,\
                               epoch * batch_count + i)
        #if epoch % 2 == 0:
        print "Epoch: ", epoch

    print "Accuracy: ", accuracy.eval\
          (feed_dict={X: mnist.test.images, Y_: mnist.test.labels})
    print "done"
```

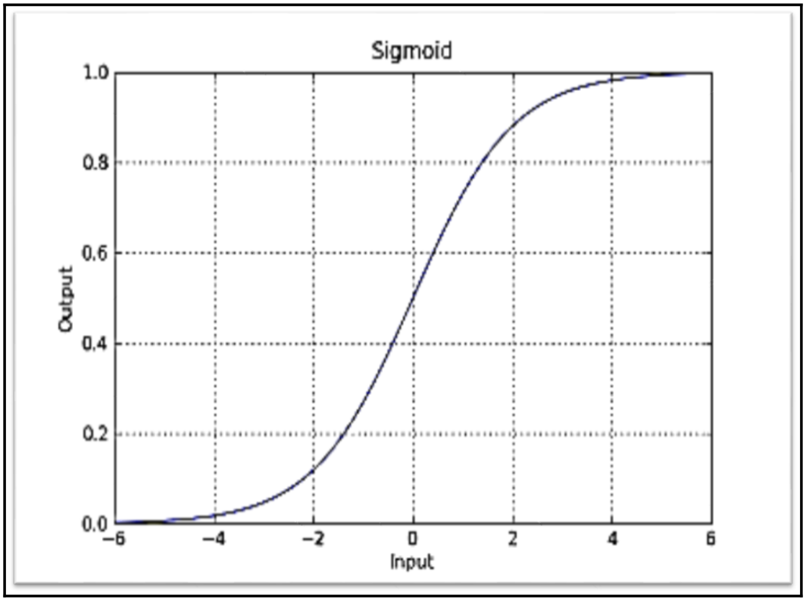
Chapter 3: Using TensorFlow on a Feed-Forward Neural Network



$$\mathrm{net}_i = \sum\nolimits_j w_{ij}\, x_j. \ldots\ldots\ldots\ldots\ldots \ldots (a)$$

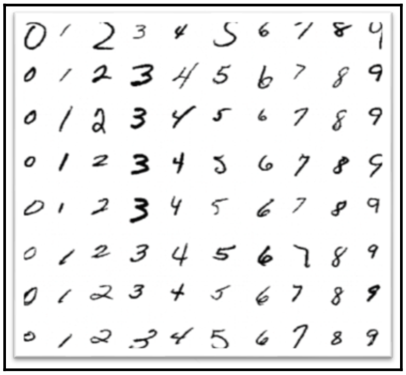
$$\mathrm{net}_i = \sum\nolimits_j w_{ij}\, x_j + b_i \ldots\ldots\ldots\ldots\ldots\ldots (b)$$

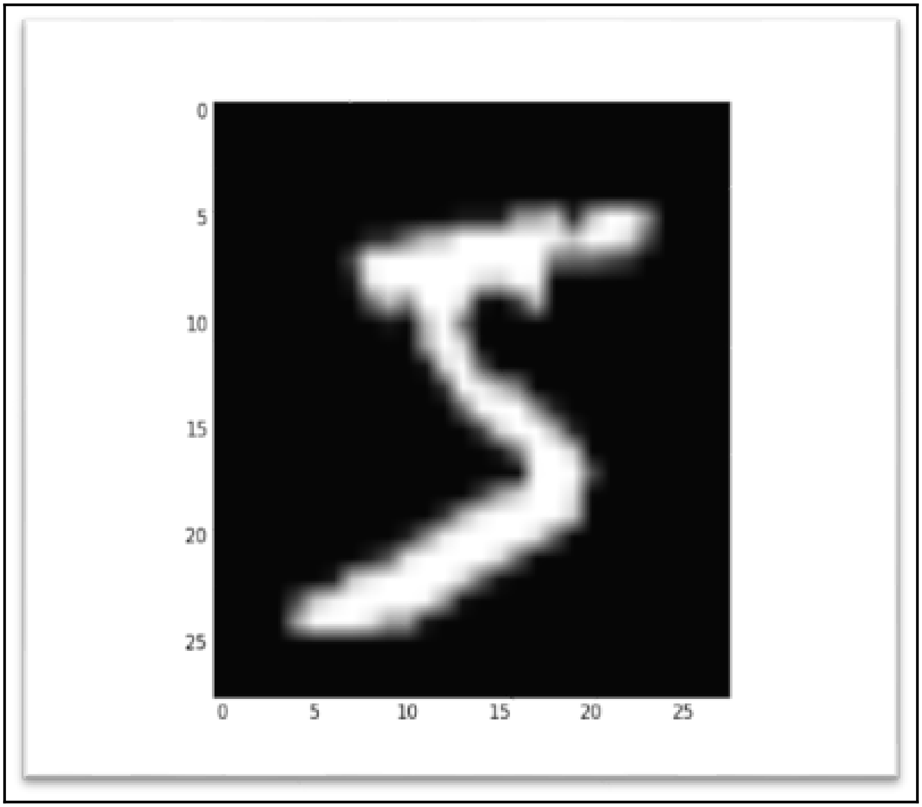
$$out_i = \frac{1}{1 + e^{-net_i}}$$

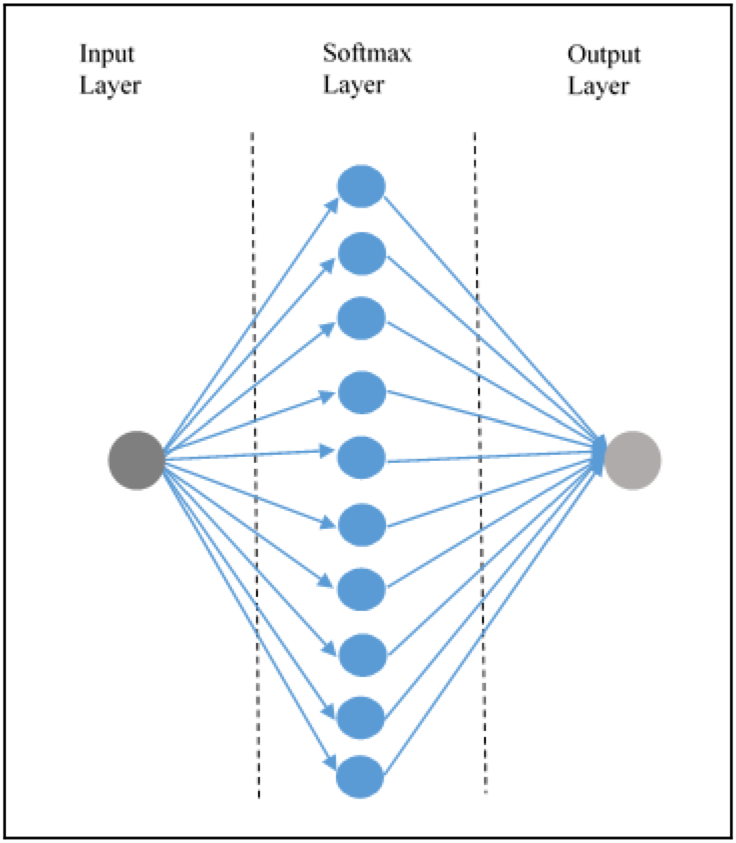


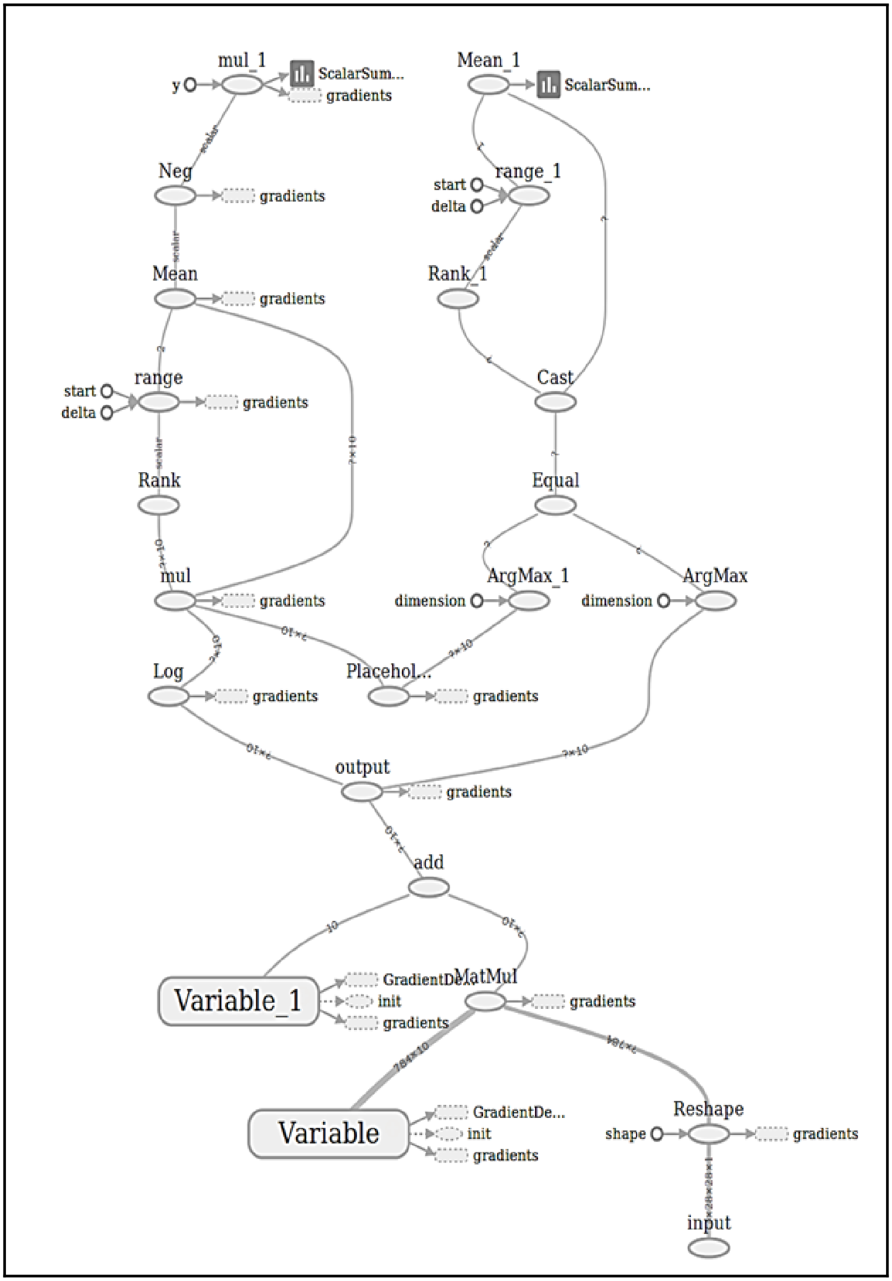
$$out_i = \frac{e^{net_i}}{\sum_{j=1}^{N} e^{net_j}}$$

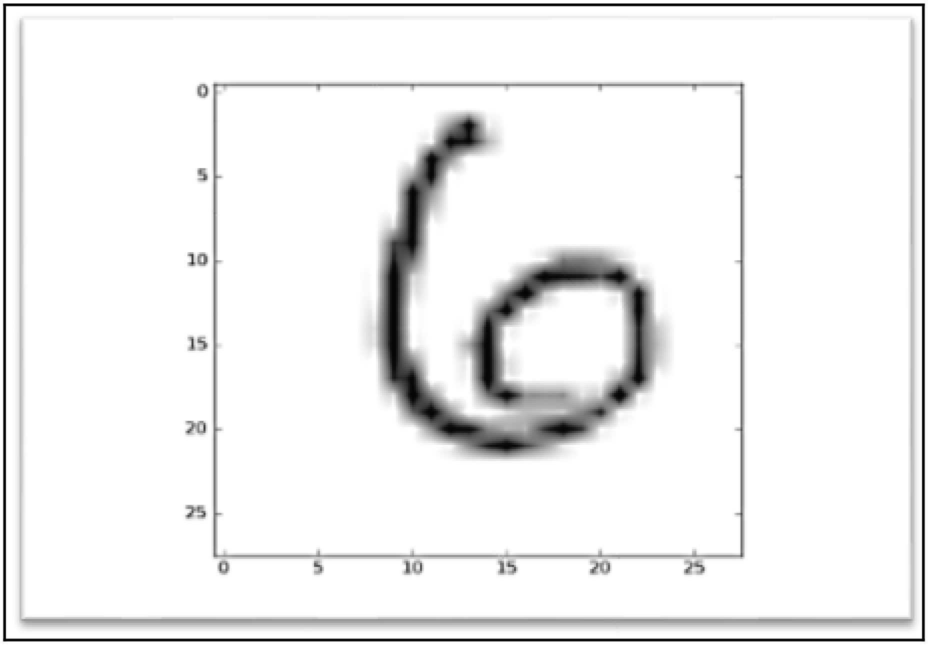
$$0 \leq \mathrm{out}_i \leq 1 \text{ con } \sum_i \mathrm{out}_i = 1$$

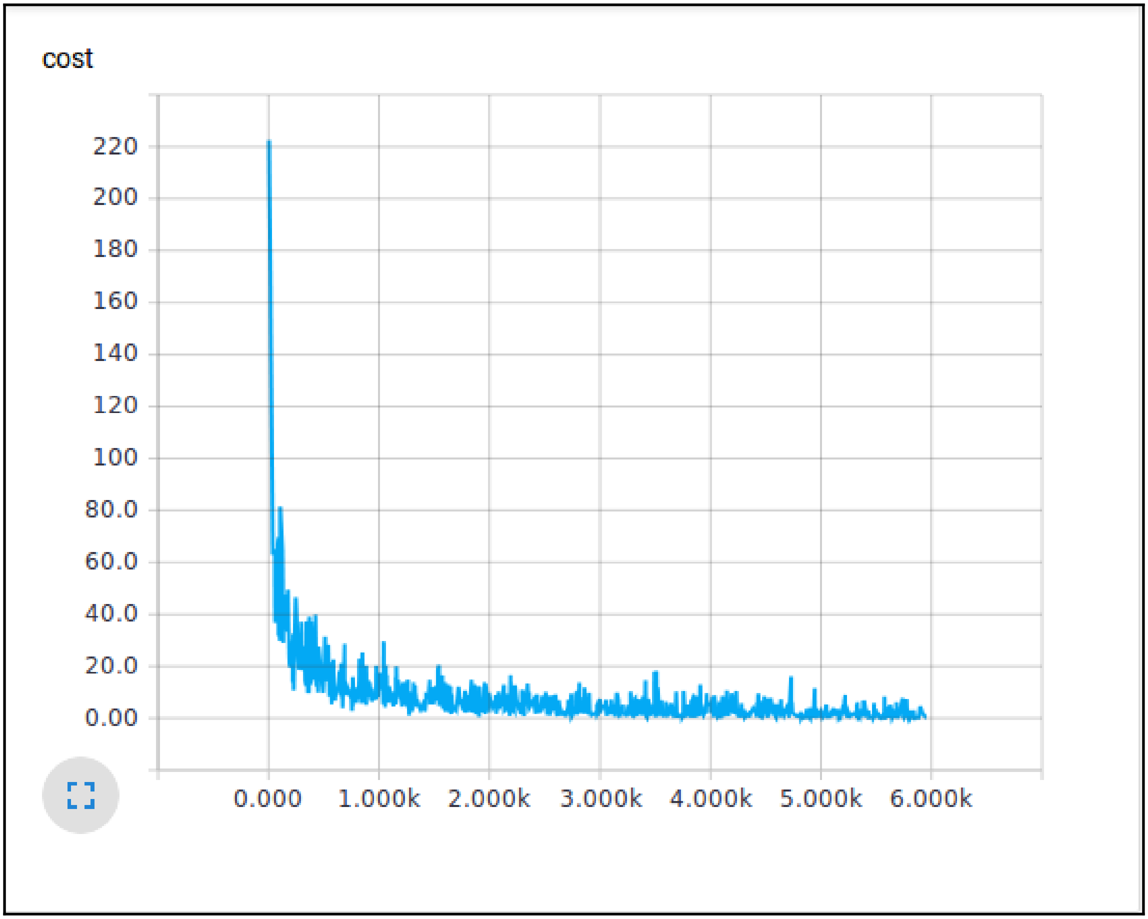


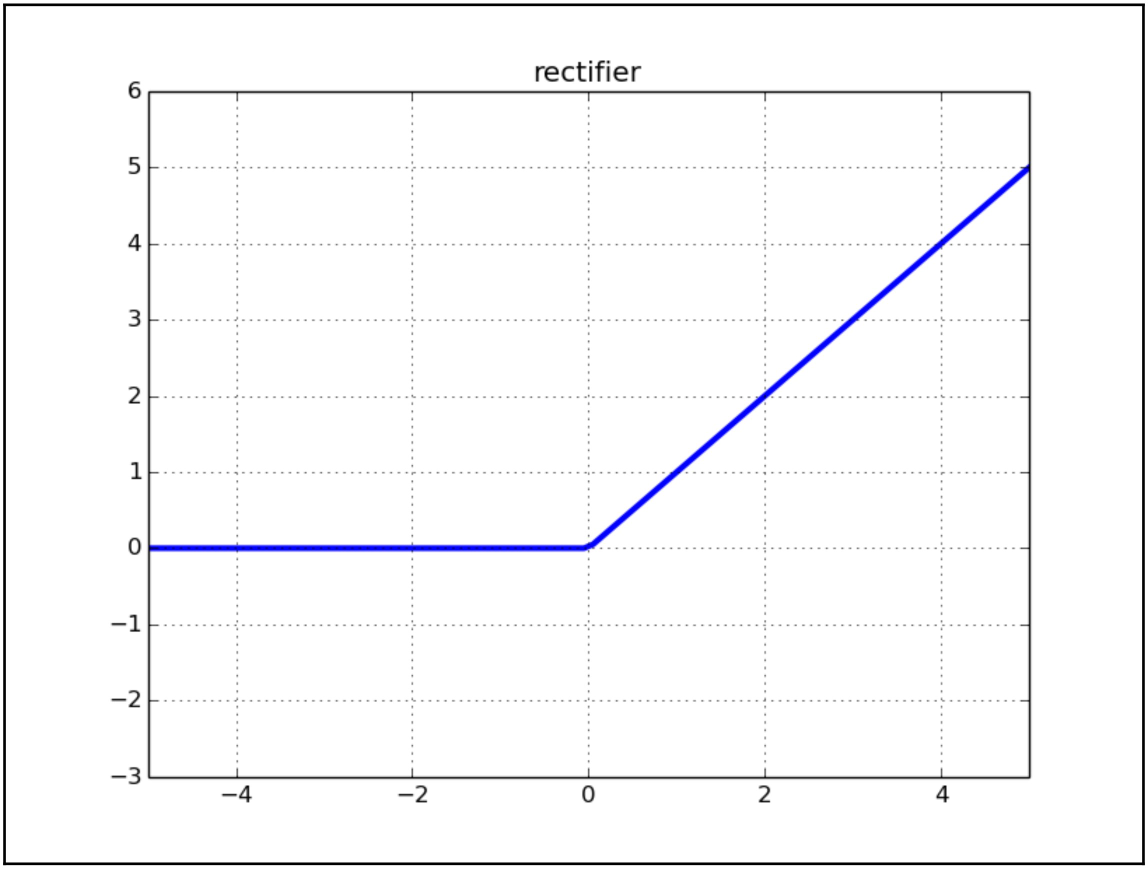

0
5
10
15
20
25
0
5
10
15
20
25

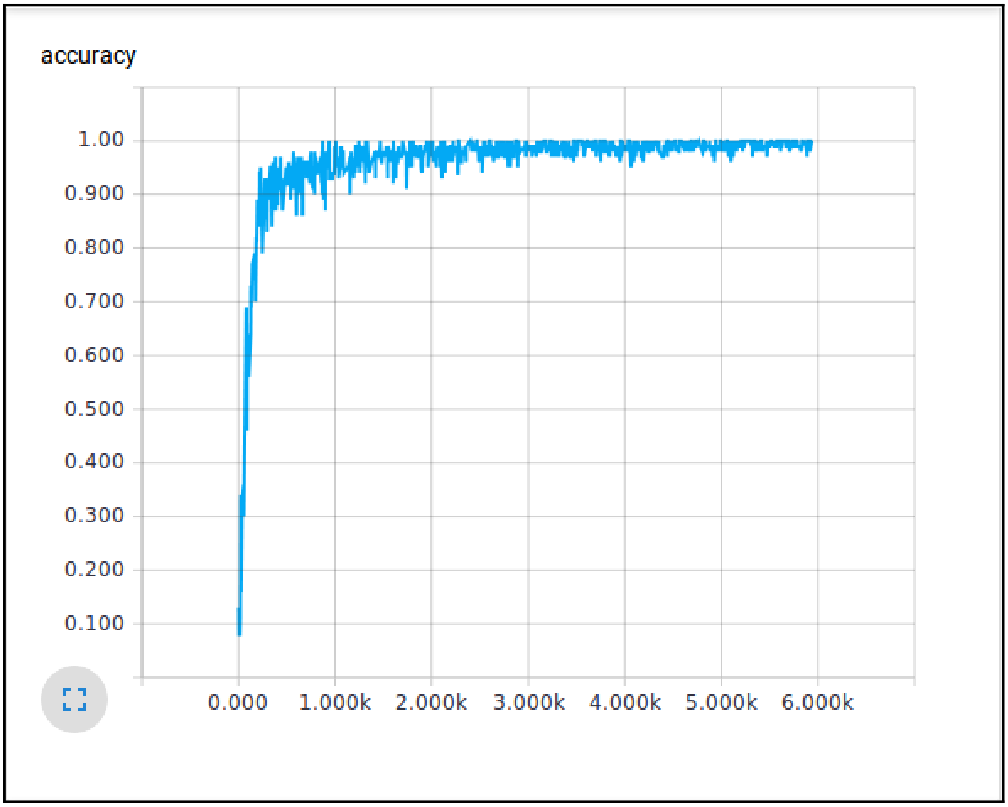

Input
Layer
Softmax
Layer
Output
Layer

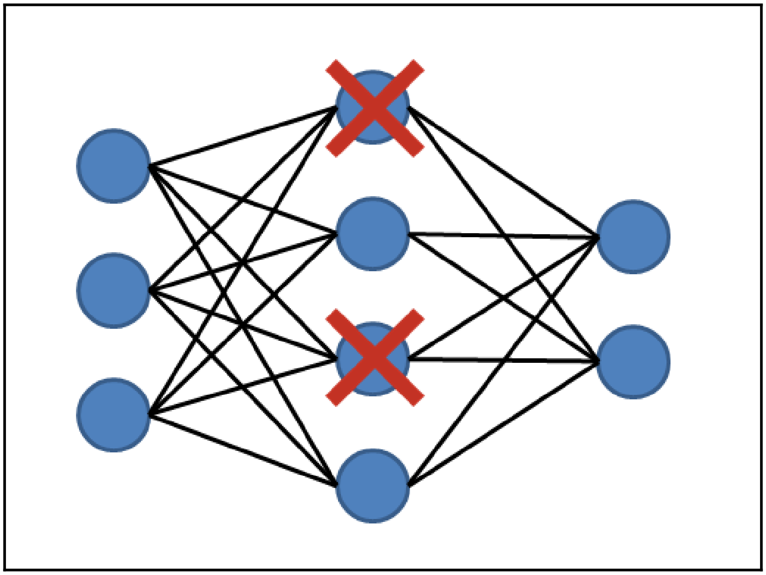

mul_1
y
ScalarSum...
gradients
Mean_1
ScalarSum...
scalar
?
Neg
gradients
range_1
start
delta
?
scalar
Mean
gradients
Rank_1
?
?
range
start
delta
gradients
Cast
?×10
scalar
Rank
?
Equal
?×10
?
?
mul
gradients
ArgMax_1
dimension
ArgMax
dimension
?×10
?×10
?×10
Log
gradients
Placehol...
gradients
?×10
?×10
output
gradients
?×10
add
10
?×10
Variable_1
GradientDe...
init
gradients
MatMul
gradients
784×10
?×784
Variable
GradientDe...
init
gradients
Reshape
shape
gradients
?×28×28×1
input

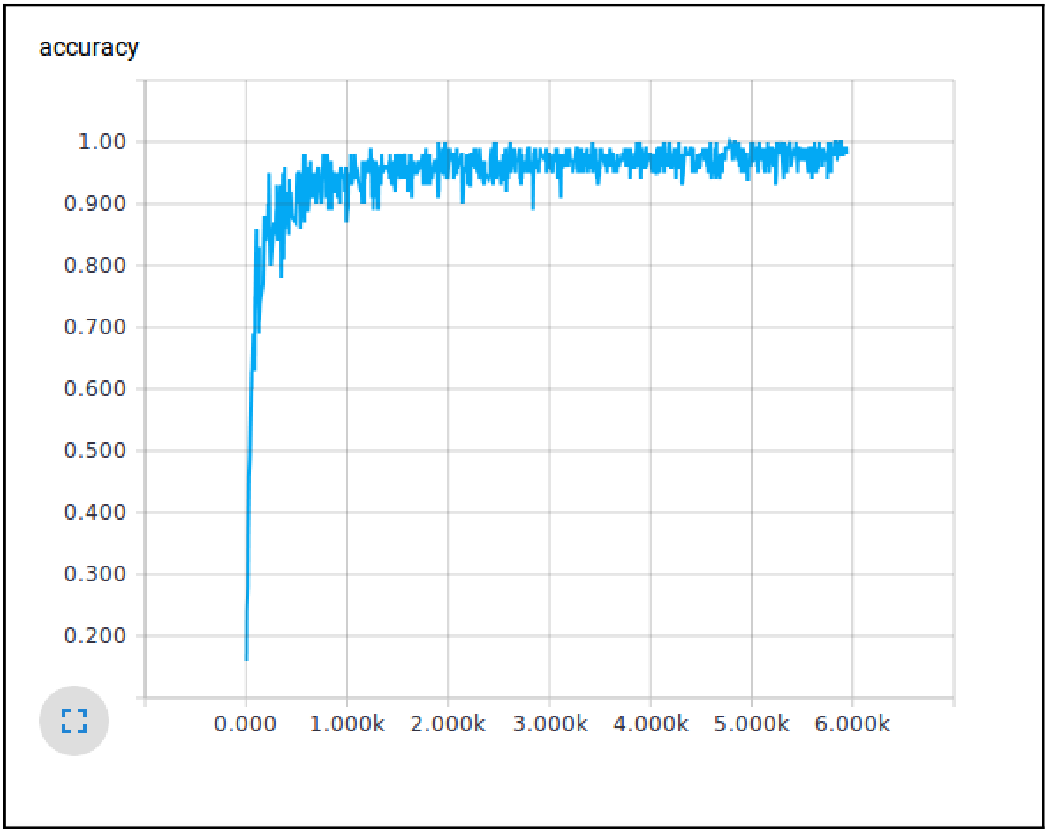

0
5
10
15
20
25
0
5
10
15
20
25

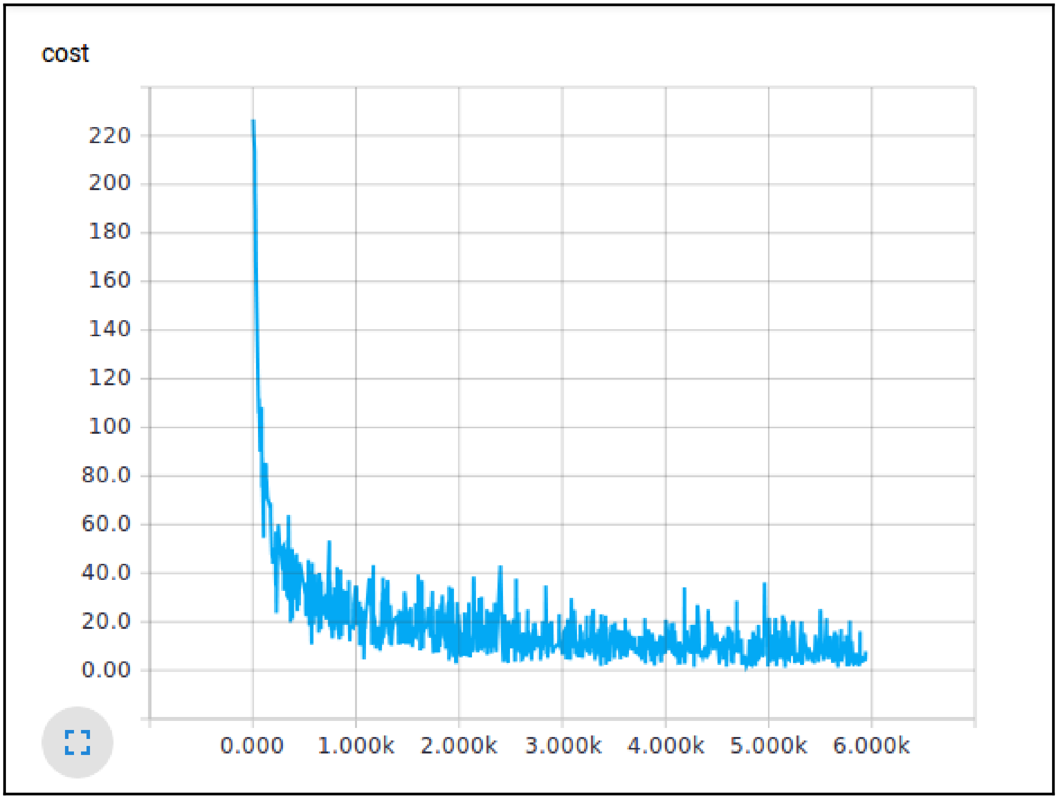

cost
220
200
180
160
140
120
100
80.0
60.0
40.0
20.0
0.00
0.000
1.000k
2.000k
3.000k
4.000k
5.000k
6.000k


rectifier
6
5
4
3
2
1
0
−1
−2
−3
−4
−2
0
2
4

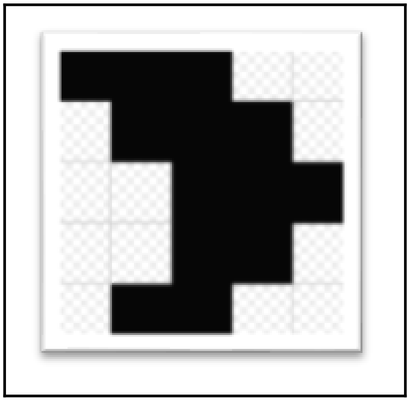

accuracy
1.00
0.900
0.800
0.700
0.600
0.500
0.400
0.300
0.200
0.100
0.000
1.000k
2.000k
3.000k
4.000k
5.000k
6.000k

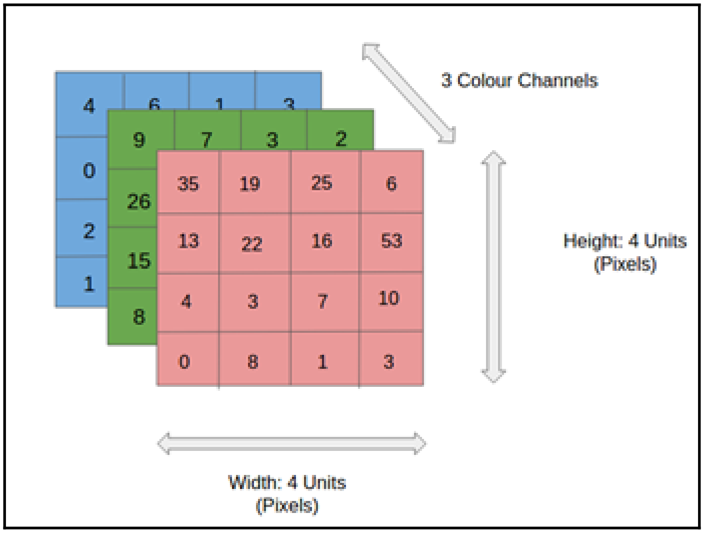



accuracy
1.00
0.900
0.800
0.700
0.600
0.500
0.400
0.300
0.200
0.000
1.000k
2.000k
3.000k
4.000k
5.000k
6.000k

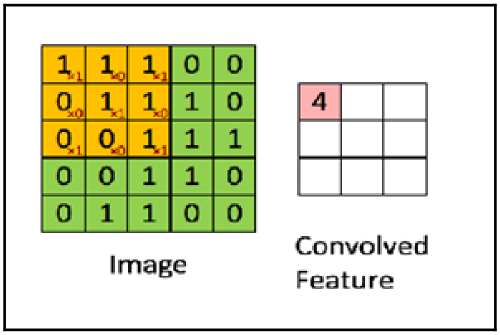


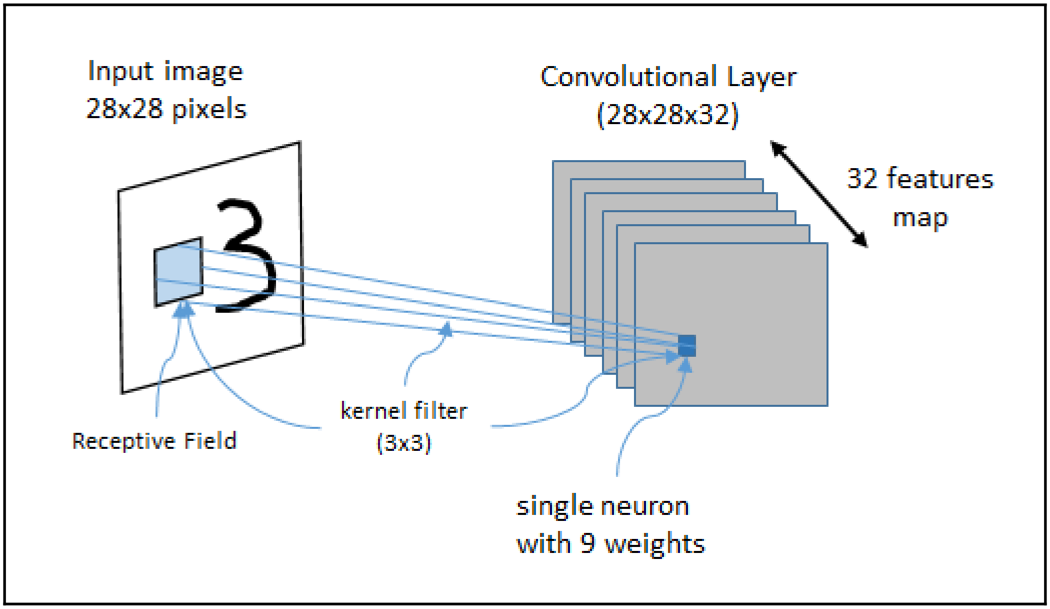
Chapter 4: TensorFlow on a Convolutional Neural Network

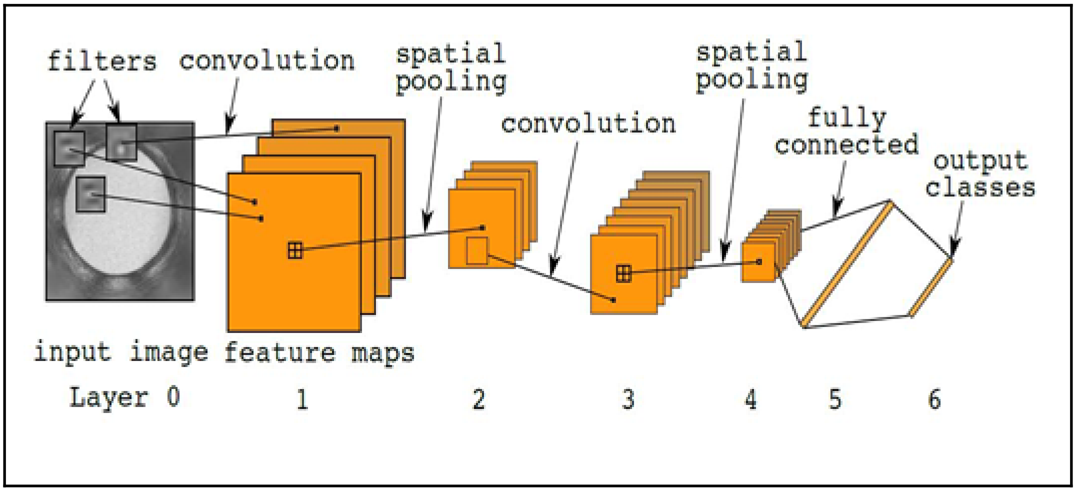


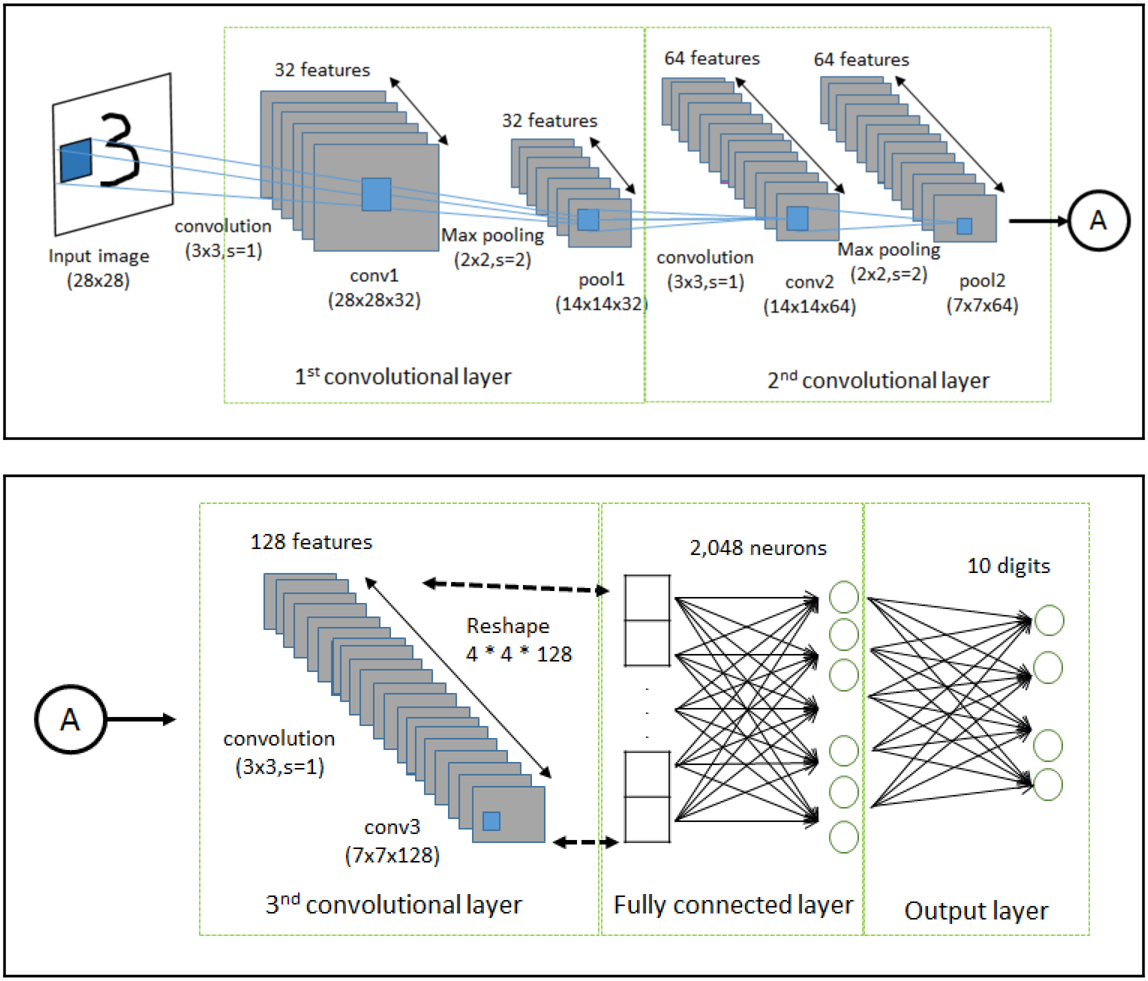


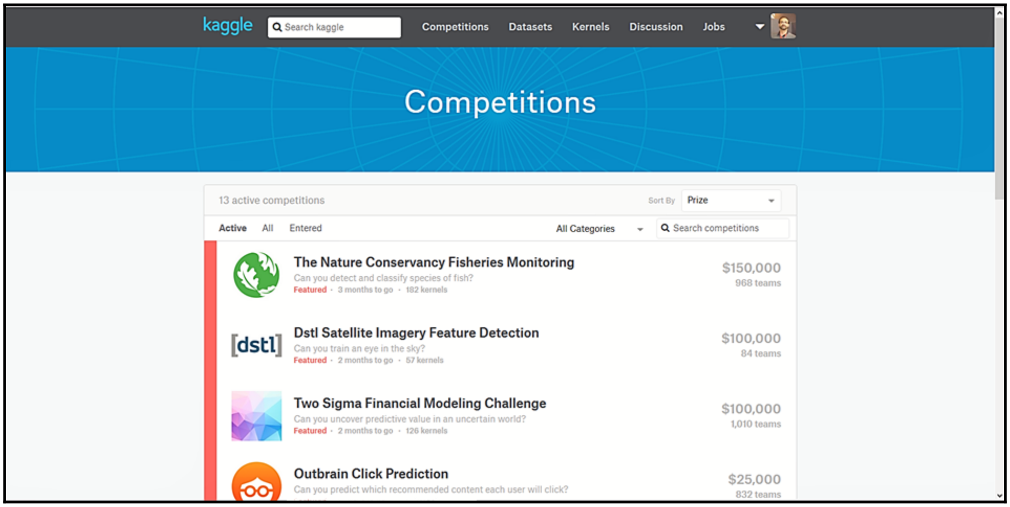
$$\begin{bmatrix}1 & 0 & 1\\ 0 & 1 & 0\\ 1 & 0 & 1\end{bmatrix}$$




Input image
28x28 pixels
Convolutional Layer
(28x28x32)
32 features
map
Receptive Field
kernel filter
(3x3)
single neuron
with 9 weights


filters
convolution
spatial
pooling
convolution
spatial
pooling
fully
connected
output
classes
input image
feature maps
Layer 0
1
2
3
4
5
6


Input image
(28x28)
convolution
(3x3,s=1)
32 features
conv1
(28x28x32)
Max pooling
(2x2,s=2)
32 features
pool1
(14x14x32)
1st convolutional layer
64 features
convolution
(3x3,s=1)
conv2
(14x14x64)
Max pooling
(2x2,s=2)
64 features
pool2
(7x7x64)
2nd convolutional layer
A
A
128 features
convolution
(3x3,s=1)
conv3
(7x7x128)
Reshape
4 * 4 * 128
3nd convolutional layer
2,048 neurons
Fully connected layer
10 digits
Output layer



Competitions

13 active competitions

Sort By Prize

Active All Entered

All Categories

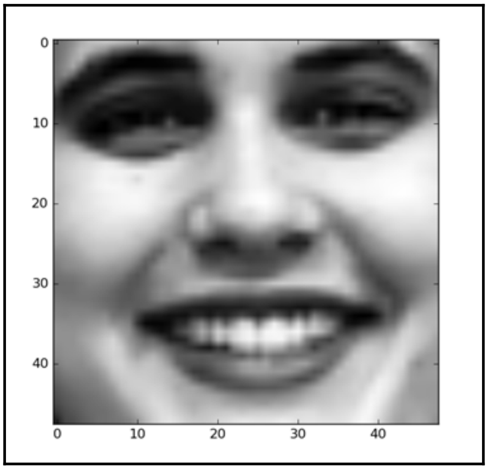
Search competitions

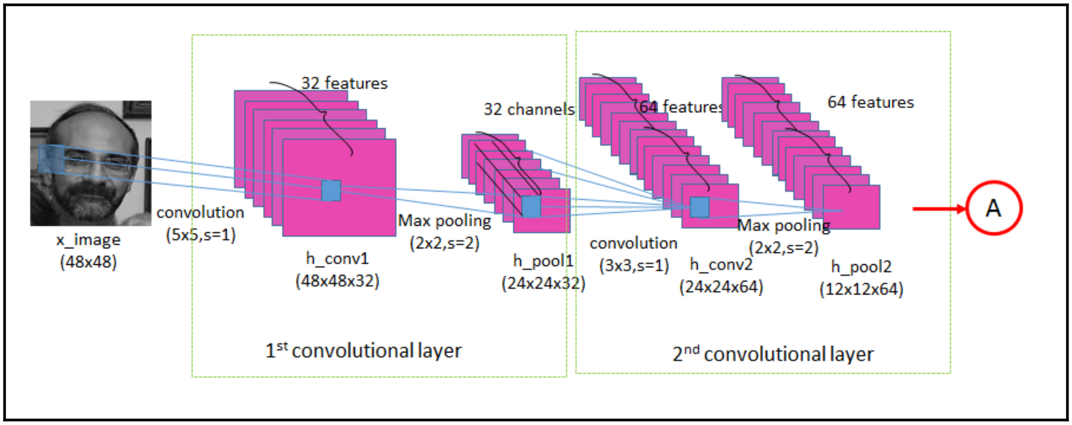
The Nature Conservancy Fisheries Monitoring
Can you detect and classify species of fish?
Featured · 3 months to go · 182 kernels
$150,000
968 teams

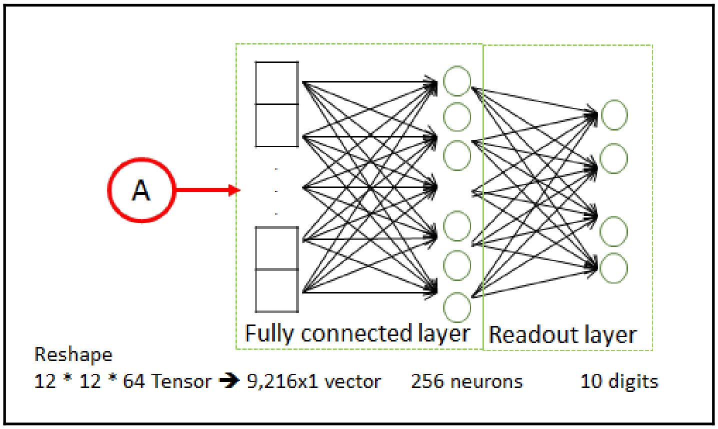
Dstl Satellite Imagery Feature Detection
Can you train an eye in the sky?
Featured · 2 months to go · 57 kernels
$100,000
84 teams

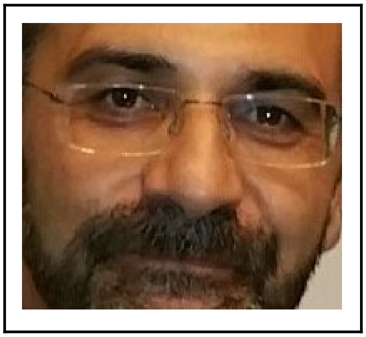
Two Sigma Financial Modeling Challenge
Can you uncover predictive value in an uncertain world?
Featured · 2 months to go · 126 kernels
$100,000
1,010 teams

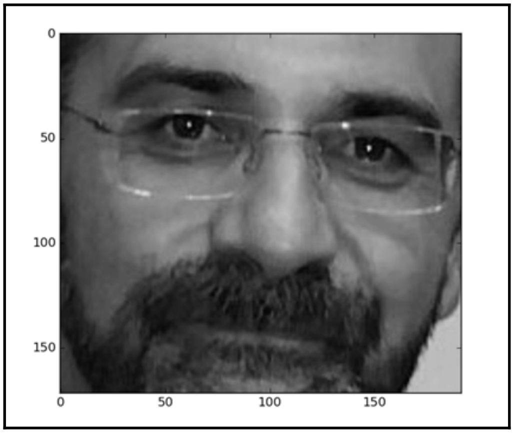
Outbrain Click Prediction
Can you predict which recommended content each user will click?
$25,000
832 teams



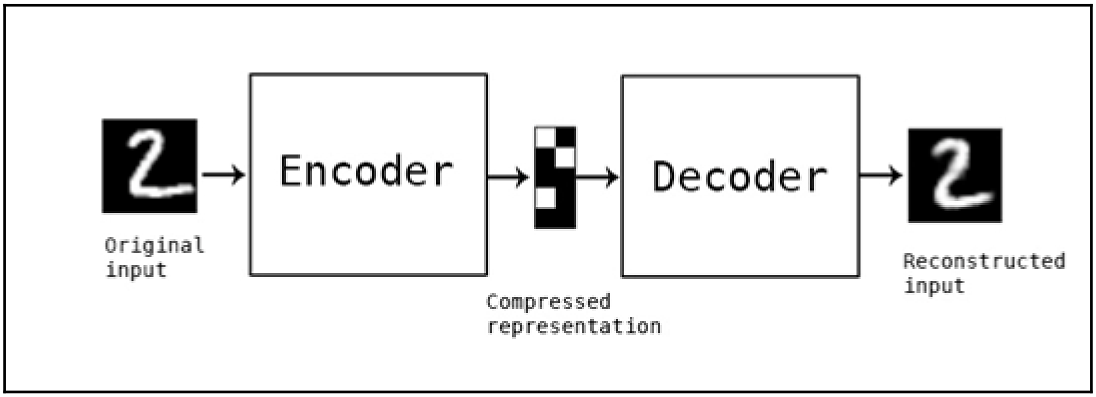

32 features
32 channels
64 features
64 features
x_image
(48x48)
convolution
(5x5,s=1)
h_conv1
(48x48x32)
Max pooling
(2x2,s=2)
h_pool1
(24x24x32)
convolution
(3x3,s=1)
h_conv2
(24x24x64)
Max pooling
(2x2,s=2)
h_pool2
(12x12x64)
1st convolutional layer
2nd convolutional layer
A

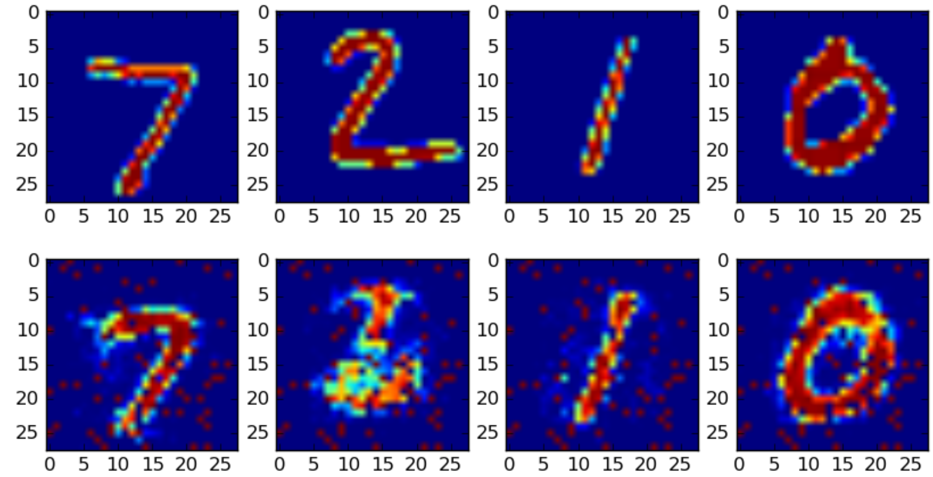

A
Fully connected layer
Readout layer
Reshape
12 * 12 * 64 Tensor → 9,216x1 vector
256 neurons
10 digits

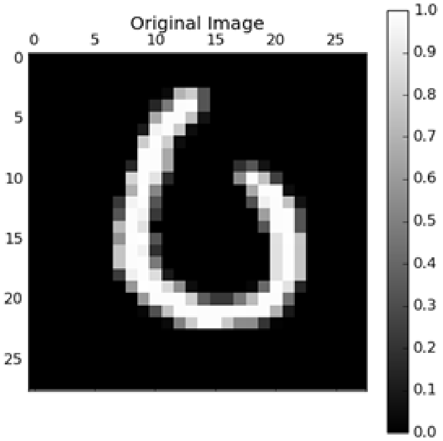
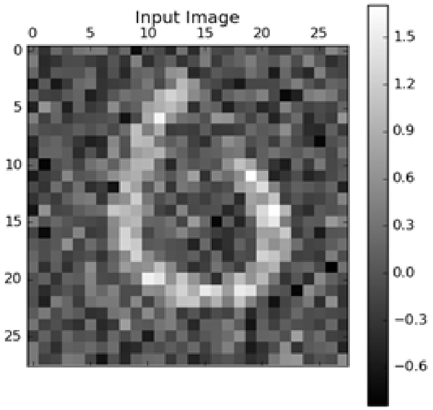


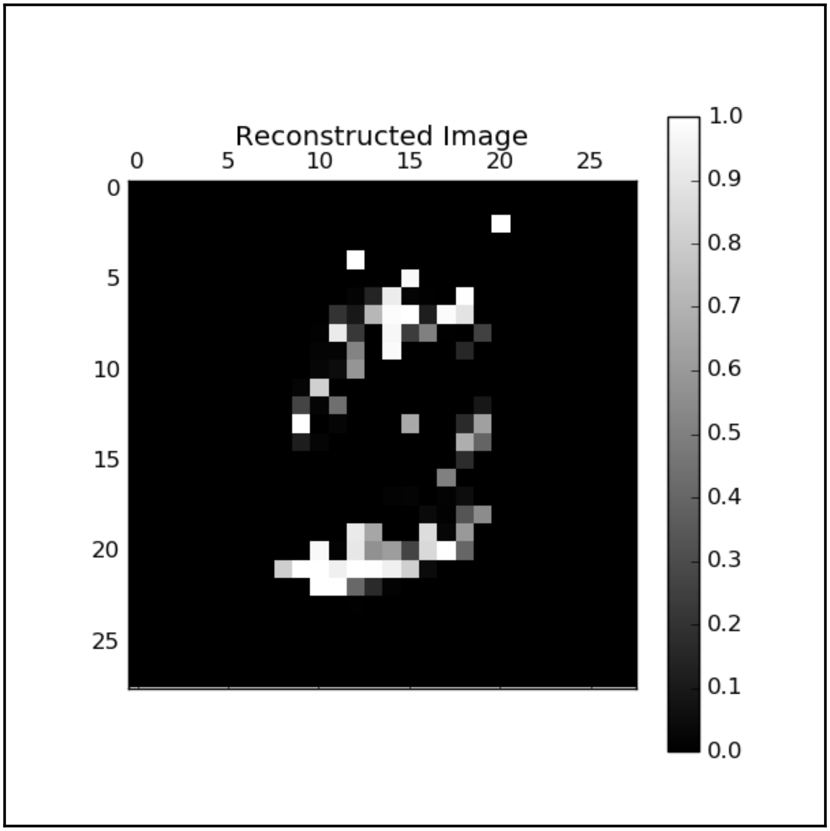


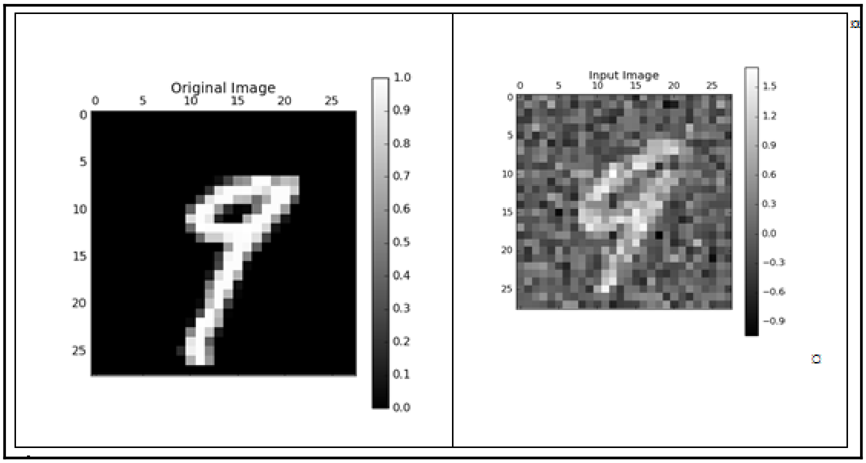
Chapter 5: Optimizing TensorFlow Autoencoders

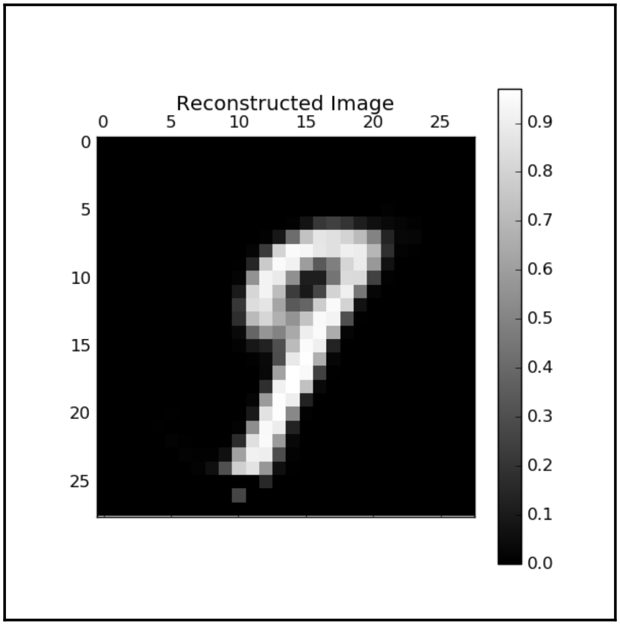



0
5
10
15
20
25
0 5 10 15 20 25

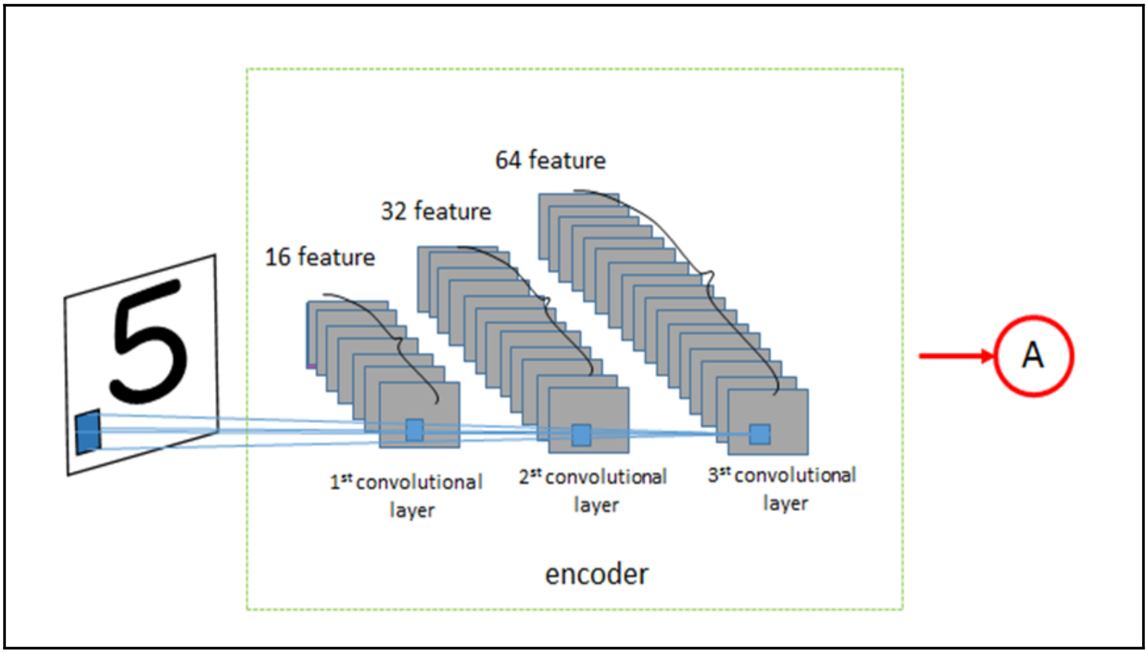
Original Image 0 5 10 15 20 25 0 5 10 15 20 25 1.0 0.9 0.8 0.7 0.6 0.5 0.4 0.3 0.2 0.1 0.0	Input Image 0 5 10 15 20 25 0 5 10 15 20 25 1.5 1.2 0.9 0.6 0.3 0.0 −0.3 −0.6


Reconstructed Image
0
5
10
15
20
25
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0

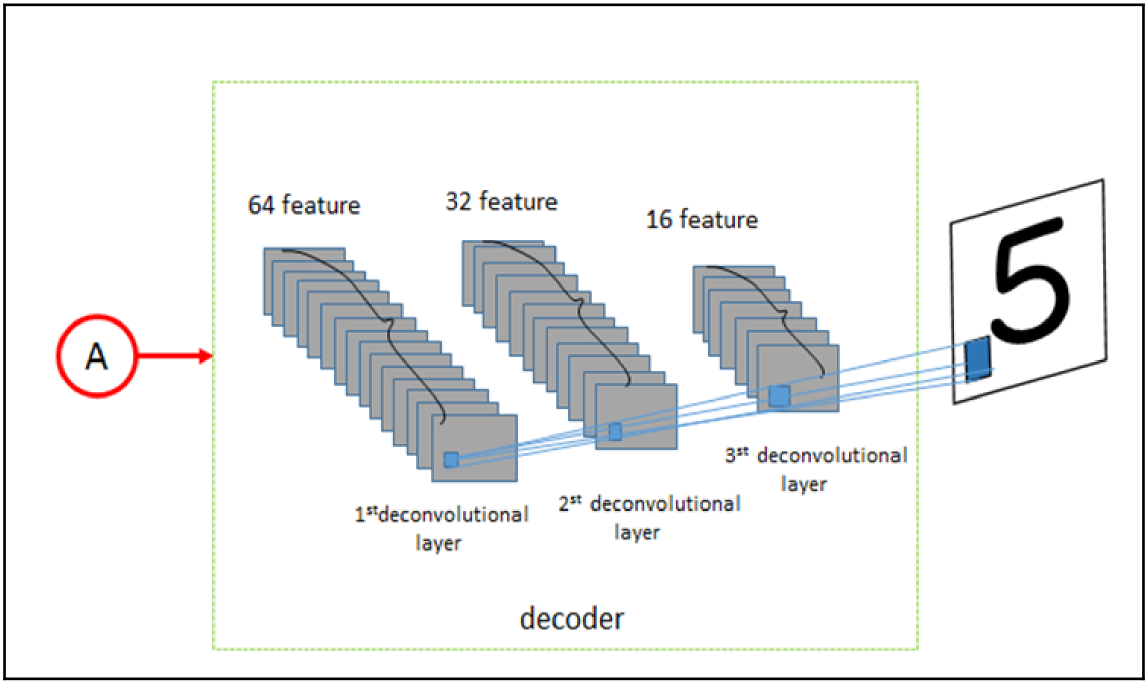

Original Image
0
5
10
15
20
25
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Input Image
0
5
10
15
20
25
1.5
1.2
0.9
0.6
0.3
0.0
−0.3
−0.6
−0.9

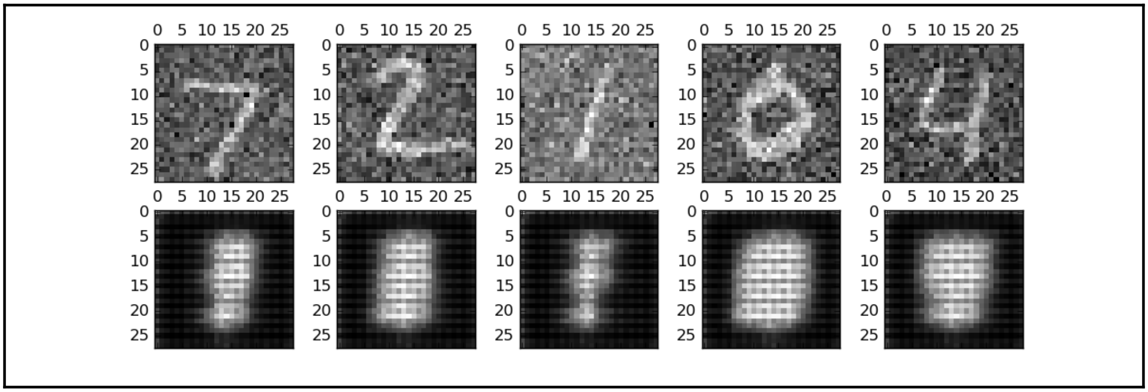


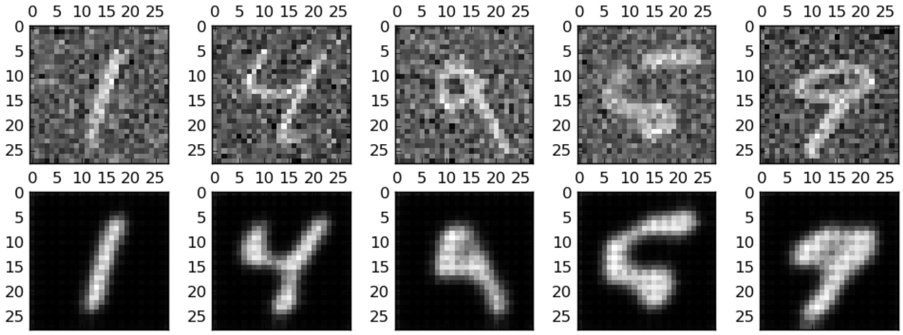
Reconstructed Image
0
5
10
15
20
25
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0

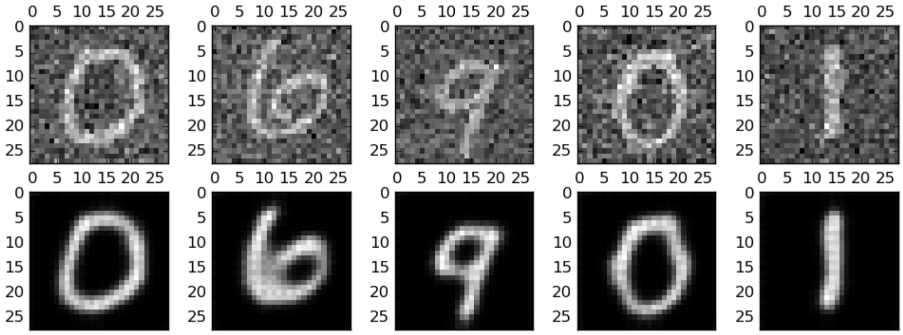


64 feature
32 feature
16 feature
5
1st convolutional layer
2st convolutional layer
3st convolutional layer
encoder
A

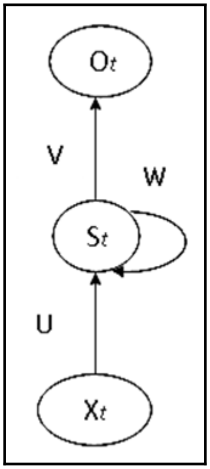

A
64 feature
32 feature
16 feature
5
1stdeconvolutional layer
2st deconvolutional layer
3st deconvolutional layer
decoder


0 5 10 15 20 25
0 5 10 15 20 25



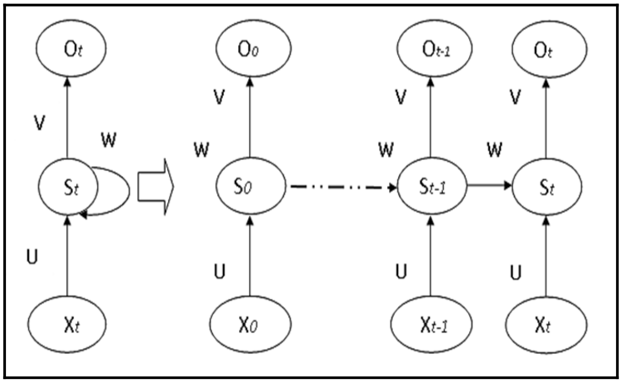


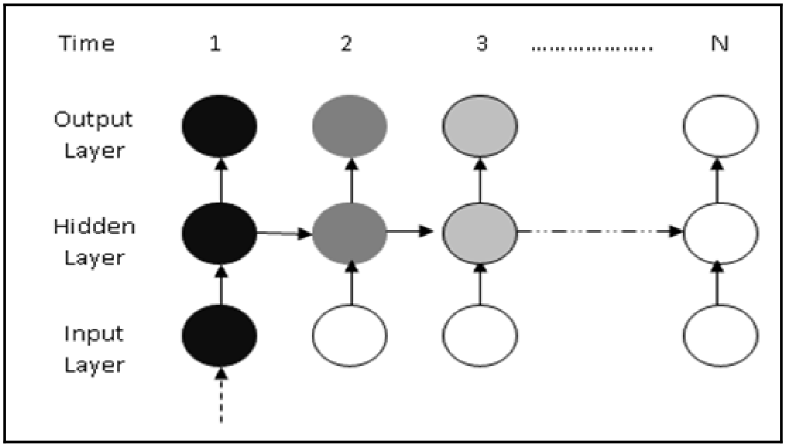
Chapter 6: Recurrent Neural Networks

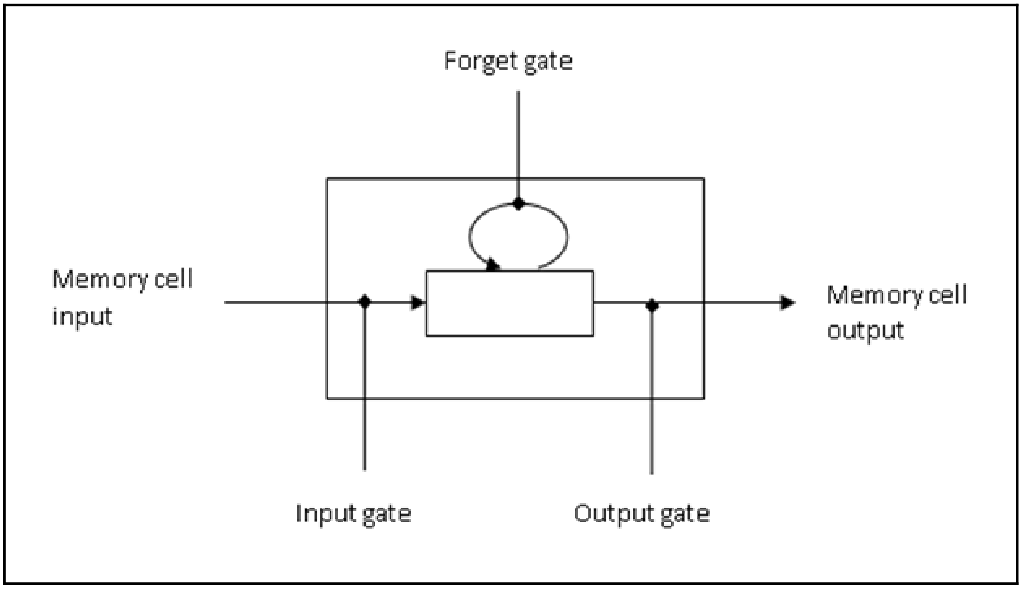


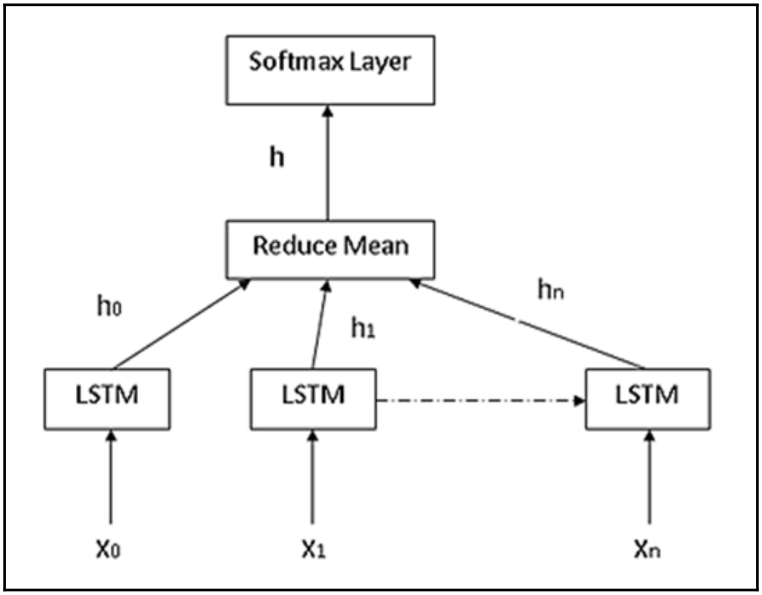
$$\boxed{S_t = f\,(U \cdot X_t + W \cdot S_{t-1})}$$

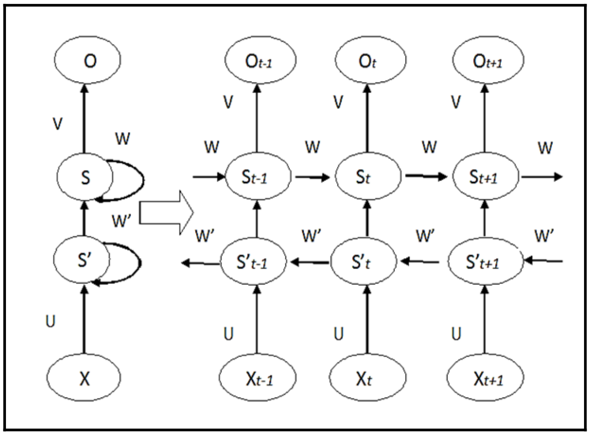
$$\boxed{O_t = V \cdot S_t}$$



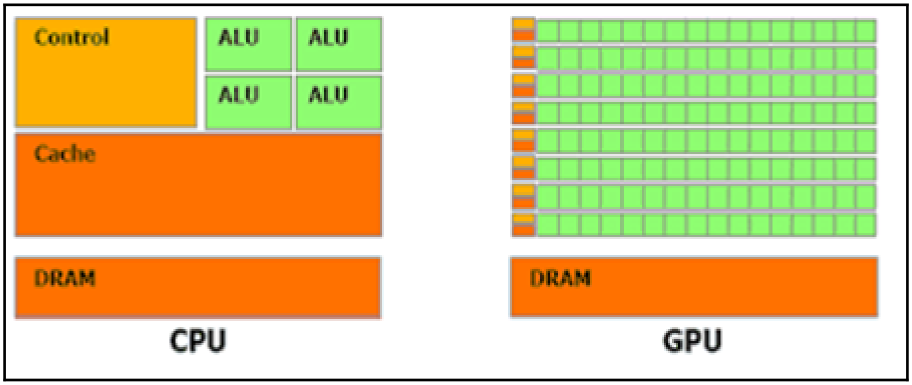

Time
1
2
3
……………….
N
Output Layer
Hidden Layer
Input Layer

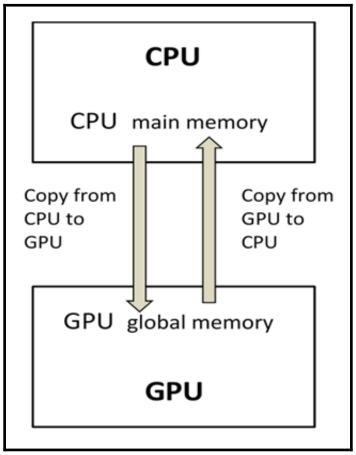

Forget gate
Memory cell input
Memory cell output
Input gate
Output gate

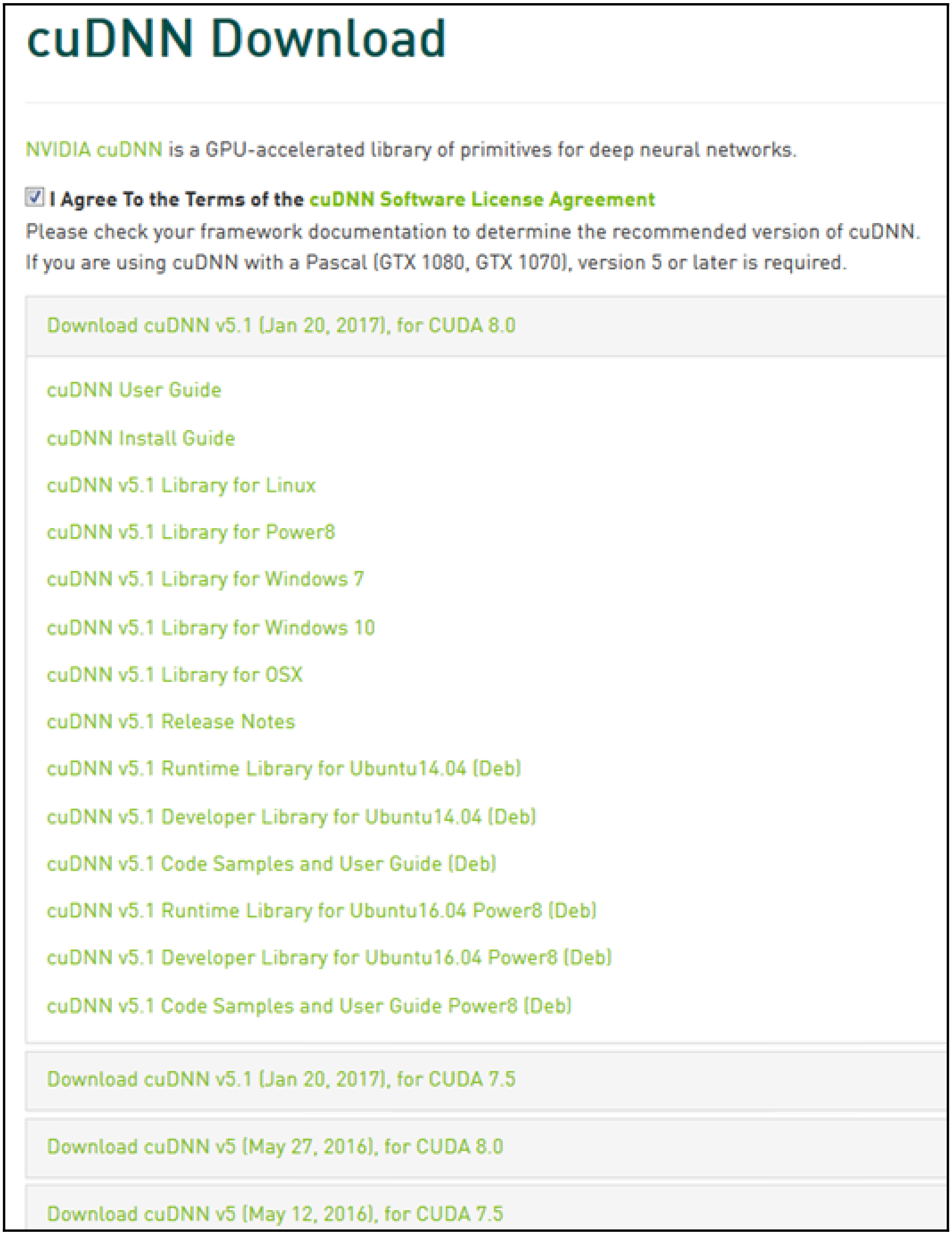




Chapter 7: GPU Computing


Control
ALU
ALU
ALU
ALU
Cache
DRAM
CPU
DRAM
GPU


CPU
CPU main memory
Copy from CPU to GPU
Copy from GPU to CPU
GPU global memory
GPU



cuDNN Download

NVIDIA cuDNN is a GPU-accelerated library of primitives for deep neural networks.

☑ **I Agree To the Terms of the cuDNN Software License Agreement**

Please check your framework documentation to determine the recommended version of cuDNN.
If you are using cuDNN with a Pascal (GTX 1080, GTX 1070), version 5 or later is required.

Download cuDNN v5.1 (Jan 20, 2017), for CUDA 8.0

cuDNN User Guide

cuDNN Install Guide

cuDNN v5.1 Library for Linux

cuDNN v5.1 Library for Power8

cuDNN v5.1 Library for Windows 7

cuDNN v5.1 Library for Windows 10

cuDNN v5.1 Library for OSX

cuDNN v5.1 Release Notes

cuDNN v5.1 Runtime Library for Ubuntu14.04 (Deb)

cuDNN v5.1 Developer Library for Ubuntu14.04 (Deb)

cuDNN v5.1 Code Samples and User Guide (Deb)

cuDNN v5.1 Runtime Library for Ubuntu16.04 Power8 (Deb)

cuDNN v5.1 Developer Library for Ubuntu16.04 Power8 (Deb)

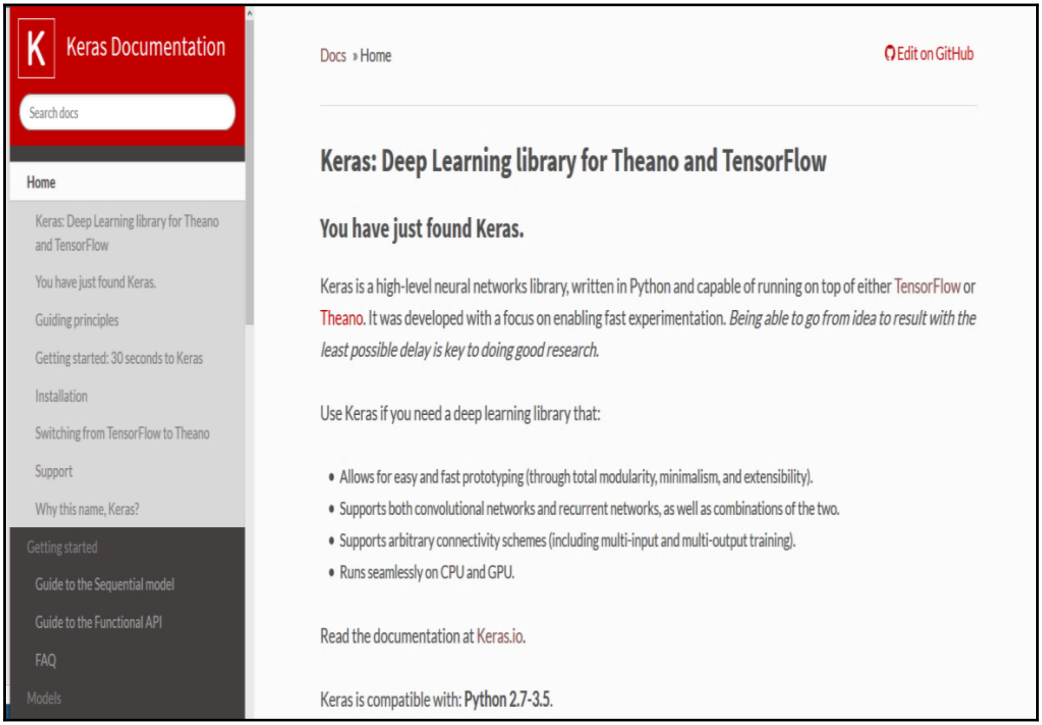
cuDNN v5.1 Code Samples and User Guide Power8 (Deb)

Download cuDNN v5.1 (Jan 20, 2017), for CUDA 7.5

Download cuDNN v5 (May 27, 2016), for CUDA 8.0

Download cuDNN v5 (May 12, 2016), for CUDA 7.5

Chapter 8: Advanced TensorFlow Programming



K
Keras Documentation
Search docs
Home
Keras: Deep Learning library for Theano and TensorFlow
You have just found Keras.
Guiding principles
Getting started: 30 seconds to Keras
Installation
Switching from TensorFlow to Theano
Support
Why this name, Keras?
Getting started
Guide to the Sequential model
Guide to the Functional API
FAQ
Models
Docs » Home
Edit on GitHub

Keras: Deep Learning library for Theano and TensorFlow

You have just found Keras.

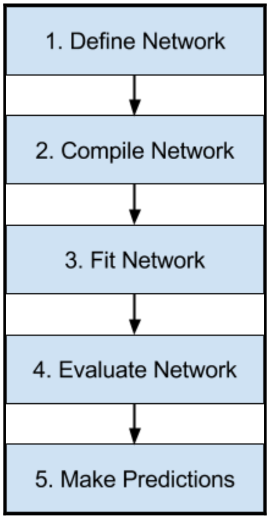
Keras is a high-level neural networks library, written in Python and capable of running on top of either TensorFlow or Theano. It was developed with a focus on enabling fast experimentation. *Being able to go from idea to result with the least possible delay is key to doing good research.*

Use Keras if you need a deep learning library that:

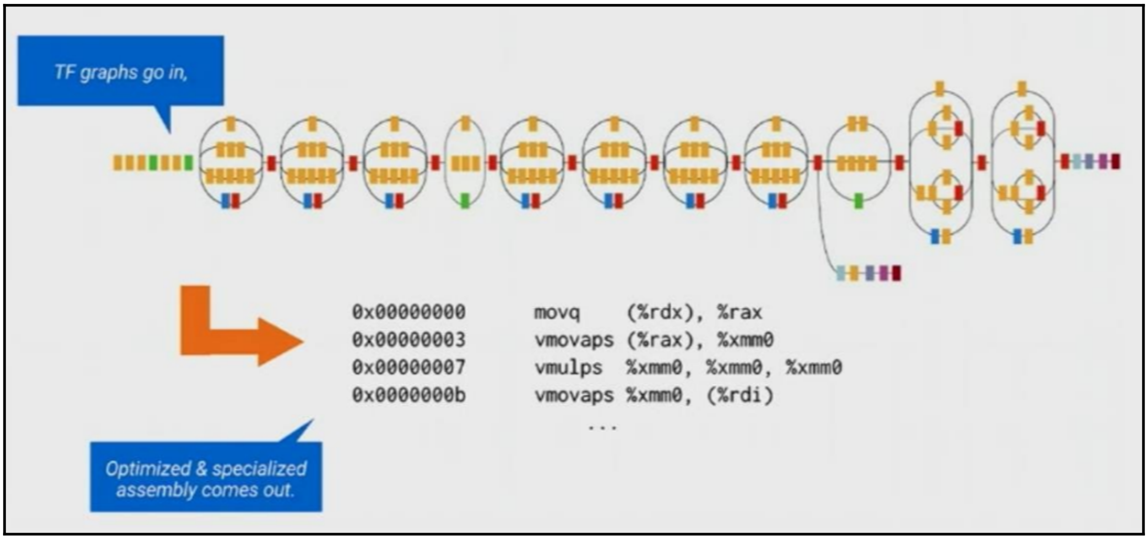
- Allows for easy and fast prototyping (through total modularity, minimalism, and extensibility).
- Supports both convolutional networks and recurrent networks, as well as combinations of the two.
- Supports arbitrary connectivity schemes (including multi-input and multi-output training).
- Runs seamlessly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

Read the documentation at Keras.io.

Keras is compatible with: **Python 2.7-3.5.**



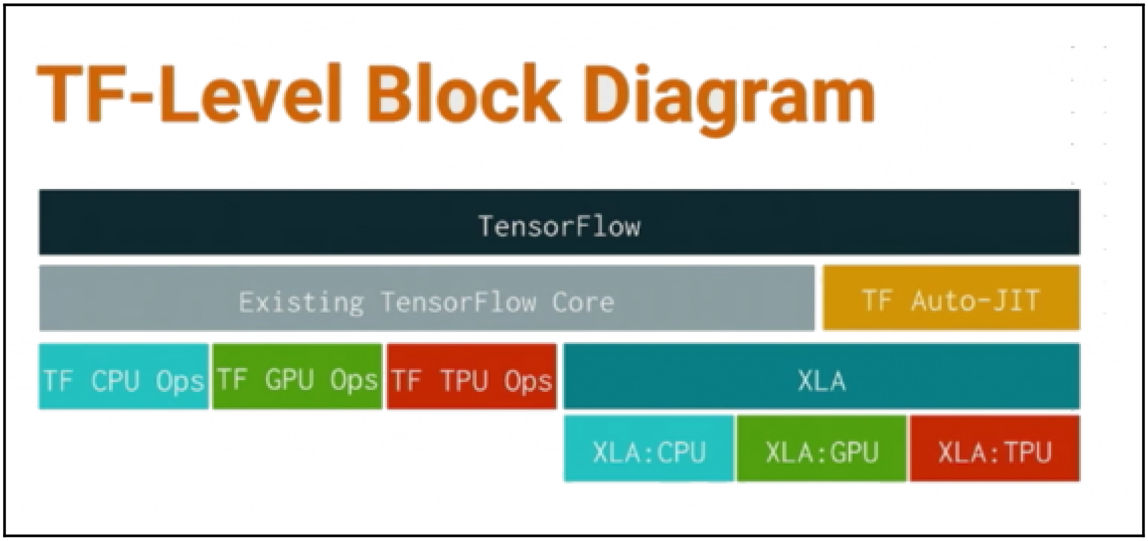
Chapter 9: Advanced Multimedia Programming with TensorFlow



TF graphs go in,

```
0x00000000      movq    (%rdx), %rax
0x00000003      vmovaps (%rax), %xmm0
0x00000007      vmulps  %xmm0, %xmm0, %xmm0
0x0000000b      vmovaps %xmm0, (%rdi)
                    ...
```

Optimized & specialized assembly comes out.



TF-Level Block Diagram

TensorFlow

Existing TensorFlow Core

TF Auto-JIT

TF CPU Ops

TF GPU Ops

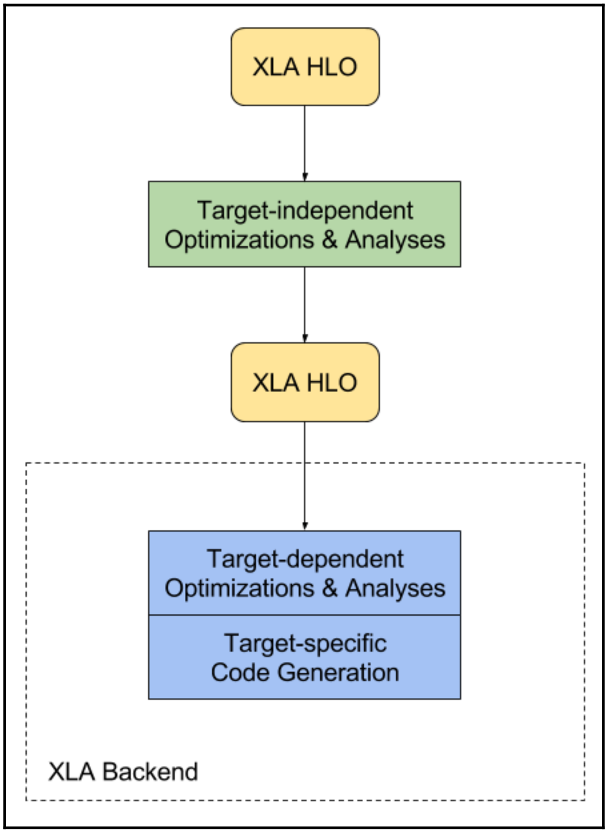
TF TPU Ops

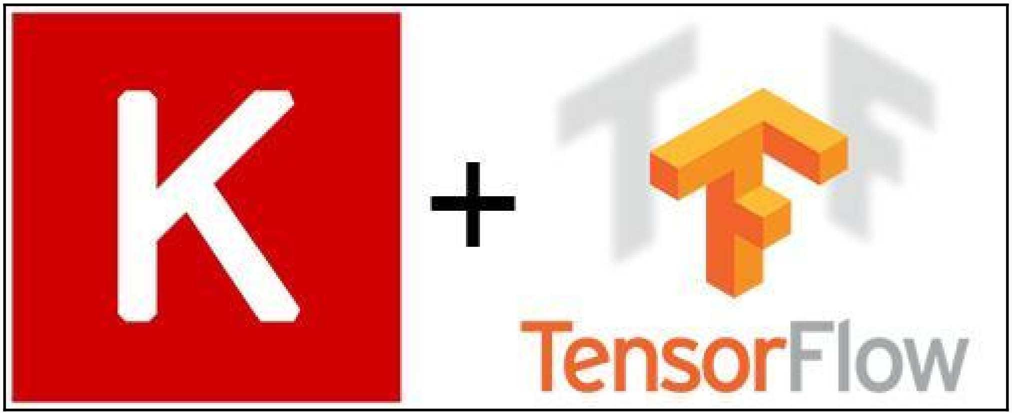
XLA

XLA:CPU

XLA:GPU

XLA:TPU

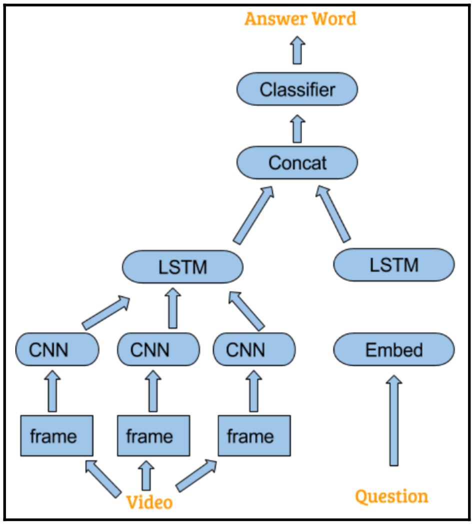

XLA HLO
Target-independent Optimizations & Analyses
XLA HLO
Target-dependent Optimizations & Analyses
Target-specific Code Generation
XLA Backend

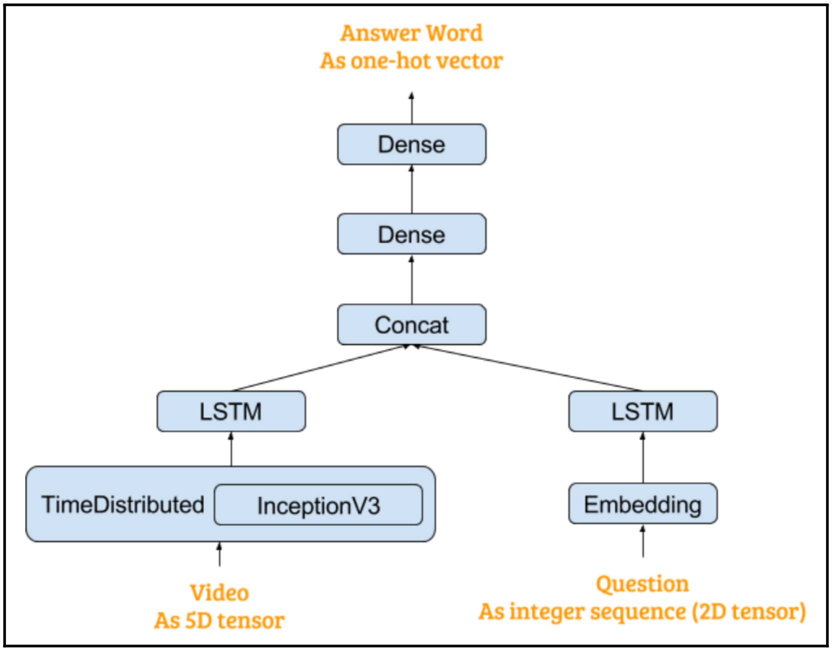


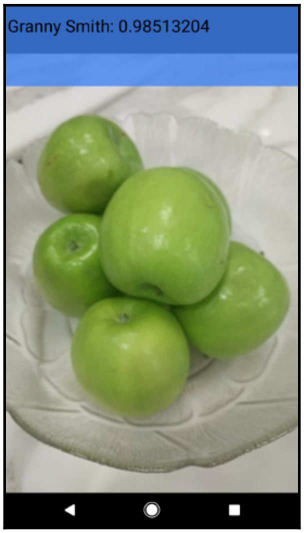
Q: What struggled with its balance?
A: This kitten
Q: Did it fall off?
A: Yes
Q: Does cute kitten jump into man's hands?
A: No

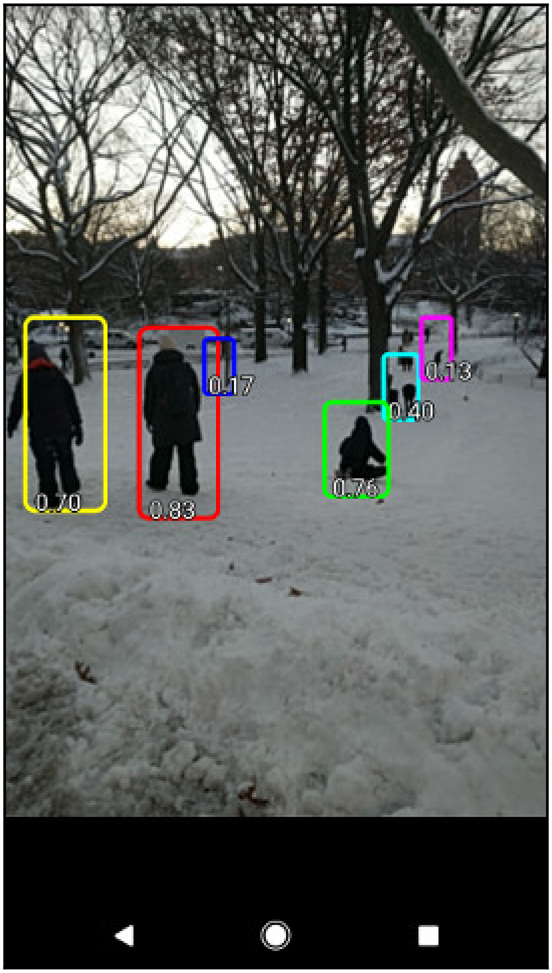
Q: Who suddenly finds herself in a fight for her scarf?
A: The woman
Q: Who starts tug of war with tourists scarf?
A: Baby elephant

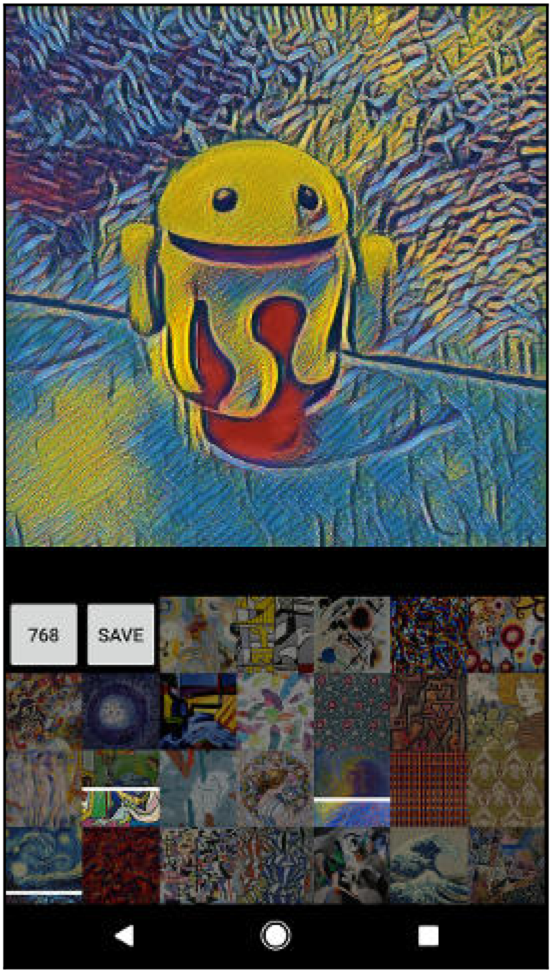
Q: Does the guy do his backflip successfully?
A: No
Q: Where does the guy attempts back flips?
A: Beach
Q: Did the man attempt a backflip off the diving board?
A: No



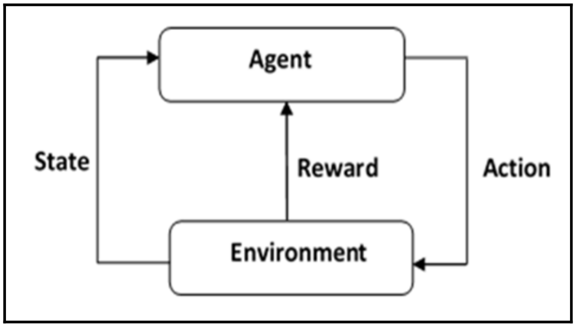


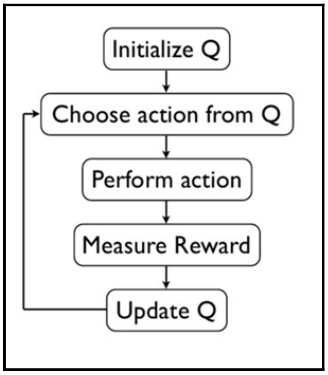

Granny Smith: 0.98513204

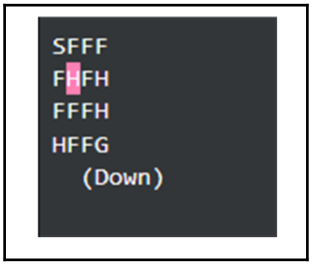

0.70
0.83
0.17
0.13
0.40
0.76



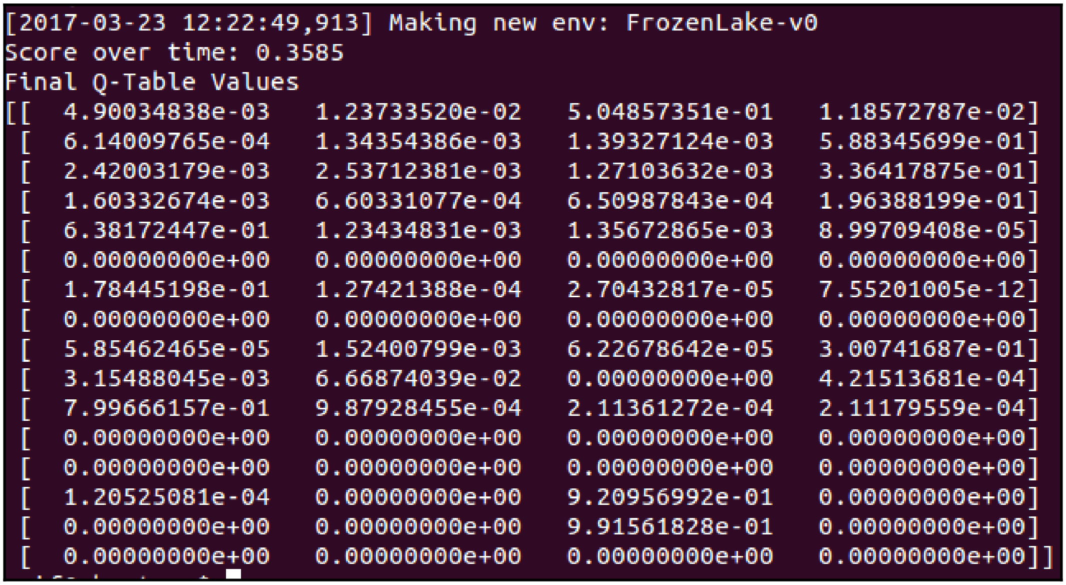
Chapter 10: Reinforcement Learning







```
SFFF
FHFH
FFFH
HFFG
  (Down)
```



```
[2017-03-23 12:22:49,913] Making new env: FrozenLake-v0
Score over time: 0.3585
Final Q-Table Values
[[  4.90034838e-03   1.23733520e-02   5.04857351e-01   1.18572787e-02]
 [  6.14009765e-04   1.34354386e-03   1.39327124e-03   5.88345699e-01]
 [  2.42003179e-03   2.53712381e-03   1.27103632e-03   3.36417875e-01]
 [  1.60332674e-03   6.60331077e-04   6.50987843e-04   1.96388199e-01]
 [  6.38172447e-01   1.23434831e-03   1.35672865e-03   8.99709408e-05]
 [  0.00000000e+00   0.00000000e+00   0.00000000e+00   0.00000000e+00]
 [  1.78445198e-01   1.27421388e-04   2.70432817e-05   7.55201005e-12]
 [  0.00000000e+00   0.00000000e+00   0.00000000e+00   0.00000000e+00]
 [  5.85462465e-05   1.52400799e-03   6.22678642e-05   3.00741687e-01]
 [  3.15488045e-03   6.66874039e-02   0.00000000e+00   4.21513681e-04]
 [  7.99666157e-01   9.87928455e-04   2.11361272e-04   2.11179559e-04]
 [  0.00000000e+00   0.00000000e+00   0.00000000e+00   0.00000000e+00]
 [  0.00000000e+00   0.00000000e+00   0.00000000e+00   0.00000000e+00]
 [  1.20525081e-04   0.00000000e+00   9.20956992e-01   0.00000000e+00]
 [  0.00000000e+00   0.00000000e+00   9.91561828e-01   0.00000000e+00]
 [  0.00000000e+00   0.00000000e+00   0.00000000e+00   0.00000000e+00]]
```

$$\boxed{loss = \sum(Q\text{-}target - Q)^2}$$