

Deep Learning / Machine Learning Seminar



- MNIST to CNN-

박성현

학습을 시켰으면 시험을 봐야지?

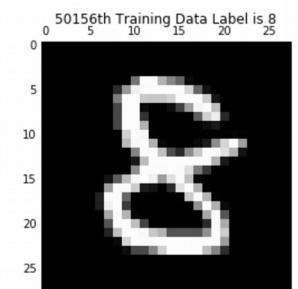
- 학습만 하고 끝? 그럼 왜 함 이걸 ㅎㅎ
- Training Data Set 과 Test Data Set을 분리해서 확인
- 간단하게 MNIST 데이터 셋을 통해서 이것을 해보자!

MNIST

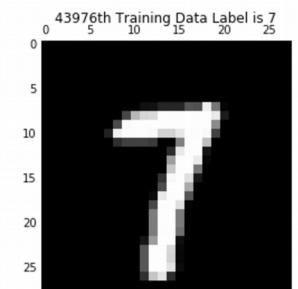
- MNIST ?
- Modified National Institute of Standards and Technology



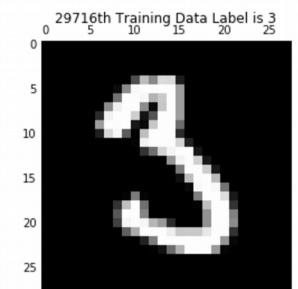
How does the training data look like?



43976th Training Data Label is 7



29716th Training Data Label is 3



```
(python)>>from tensorflow.examples.tutorials.mnist import input_data
```

MNIST Dataset



[train-images-idx3-ubyte.gz](#): training set images (9912422 bytes)

[train-labels-idx1-ubyte.gz](#): training set labels (28881 bytes)

[t10k-images-idx3-ubyte.gz](#): test set images (1648877 bytes)

[t10k-labels-idx1-ubyte.gz](#): test set labels (4542 bytes)

<http://yann.lecun.com/exdb/mnist/>

Before testing MNIST...

복습합니다

1. tensorflow basic / One-hot encoding
2. Normalized Data / softmax
3. linear / logistic regression

이걸 알아봅니다

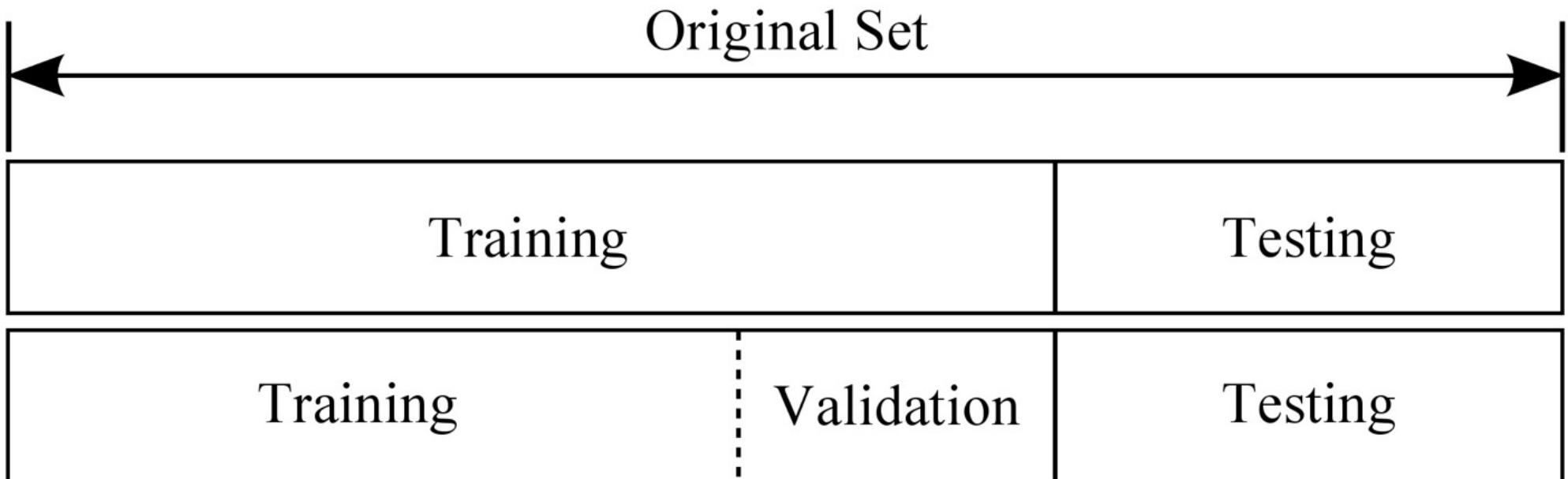
1. epoch? Batch?
2. training / test set
3. CNN
4. optimizer and initializer
5. tensorboard

epoch and batch

- epoch : 모든 훈련 데이터를 전부 학습하는데 걸리는 횟수
- batch : 한번 훈련 데이터를 학습할 때 학습하는 개수들
- 즉, 한번 학습할 때 걸리는 iteration 횟수는 epoch / batch

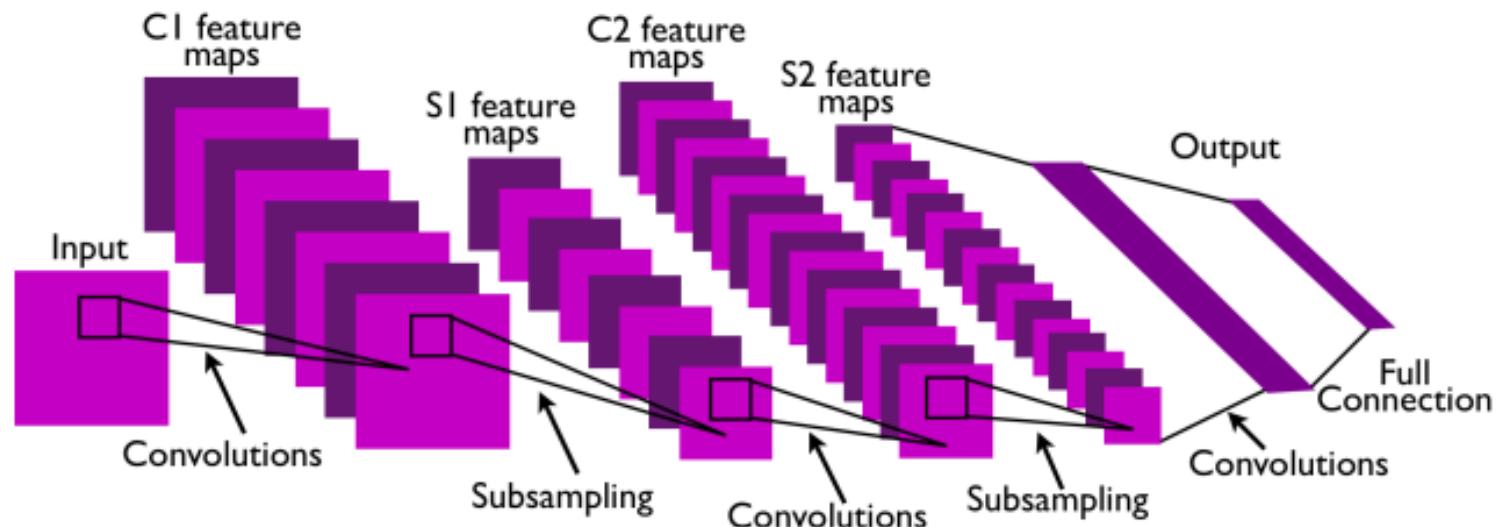
Training and Test Set

- Need to separate test set from training set



CNN

Convolutional Neural Network



This is pretty much **everything** about the convolutional neural network.

Convolution + Subsampling + Full Connection

Convolution

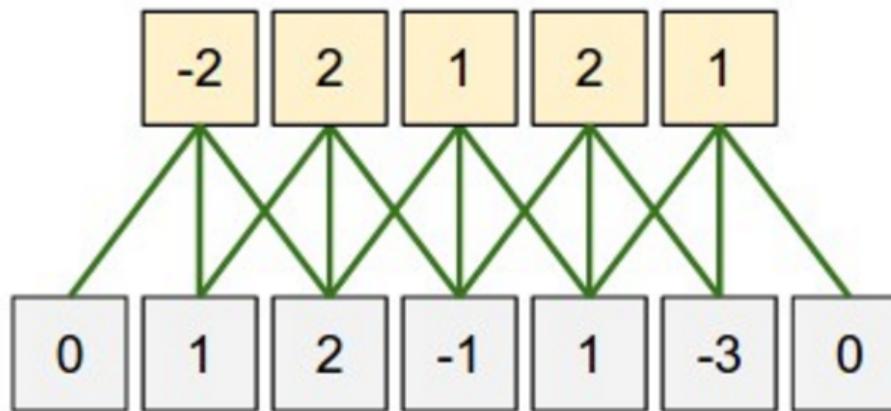
1 <small>x1</small>	1 <small>x0</small>	1 <small>x1</small>	0	0
0 <small>x0</small>	1 <small>x1</small>	1 <small>x0</small>	1	0
0 <small>x1</small>	0 <small>x0</small>	1 <small>x1</small>	1	1
0	0	1	1	0
0	1	1	0	0

Image

4		

Convolved
Feature

Zero-padding



What is the size of the input?

$$n_{in} = 5$$

What is the size of the output?

$$n_{out} = 5$$

What is the size of the filter?

$$n_{filter} = 3$$

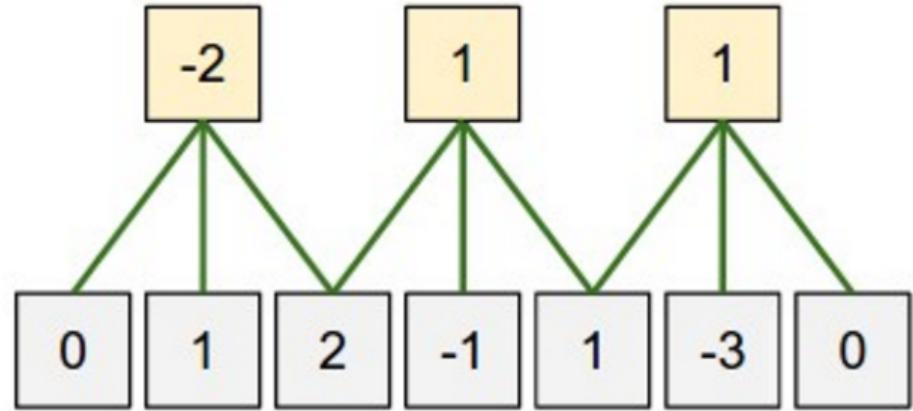
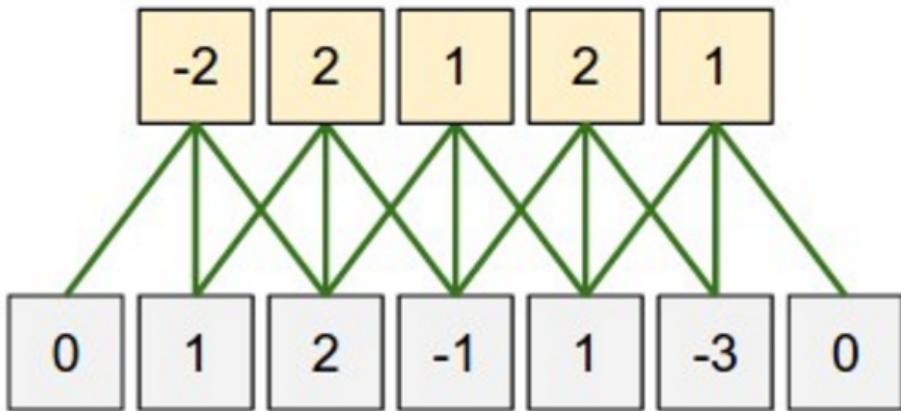
What is the size of the zero-padding?

$$n_{padding} = 1$$

$$n_{out} = (n_{in} + 2 * n_{padding} - n_{filter}) + 1$$

$$5 = (5 + 2 * 1 - 3) + 1$$

Stride



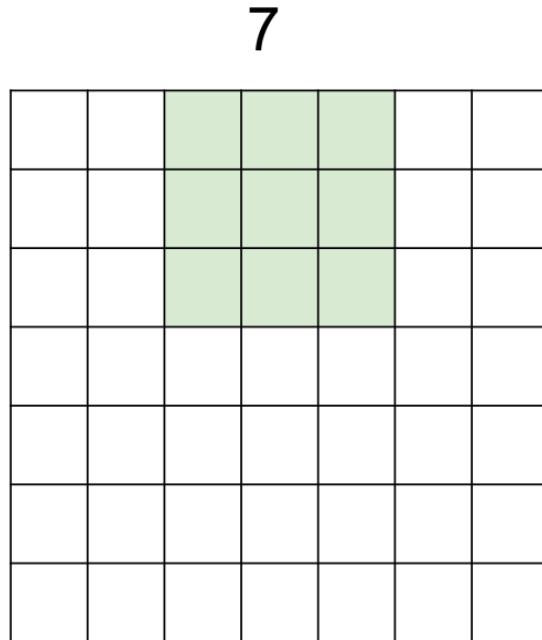
(Left) Stride size: 1

(Right) Stride size: 2

If stride size equals the filter size, there will be **no overlapping**.

Stride and Zero-padding in CNN

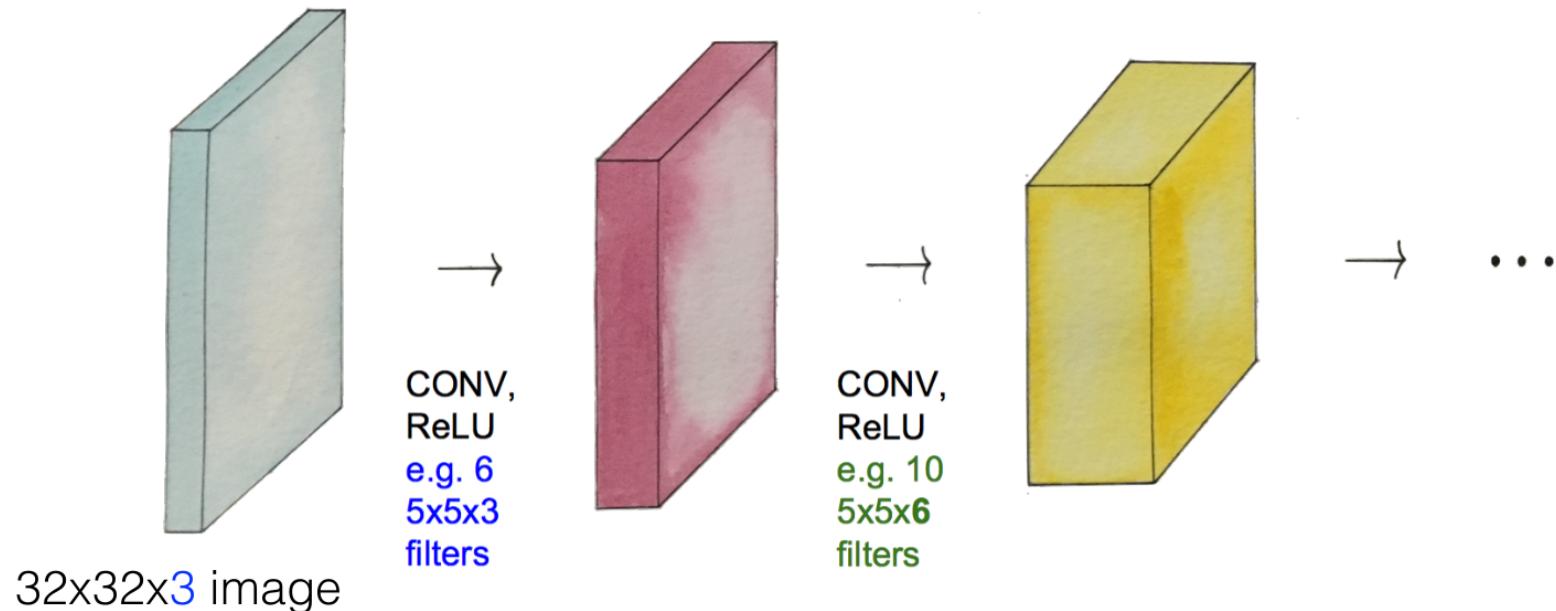
- Output size = $[N-F] / \text{stride} + 1$
A closer look at spatial dimensions:



7x7 input (spatially)
assume 3x3 filter
applied **with stride 2**

CNN in process

- $32 \times 32 \times 3 \rightarrow [5 \times 5 \times 3] * 6 \rightarrow [5 \times 5 \times 6] * 10 \rightarrow ?$



Case Study: AlexNet

[Krizhevsky et al. 2012]

Full (simplified) AlexNet architecture:

[227x227x3] INPUT

[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0

[27x27x96] MAX POOL1: 3x3 filters at stride 2

[27x27x96] NORM1: Normalization layer

[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2

[13x13x256] MAX POOL2: 3x3 filters at stride 2

[13x13x256] NORM2: Normalization layer

[13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1

[13x13x384] CONV4: 384 3x3 filters at stride 1, pad 1

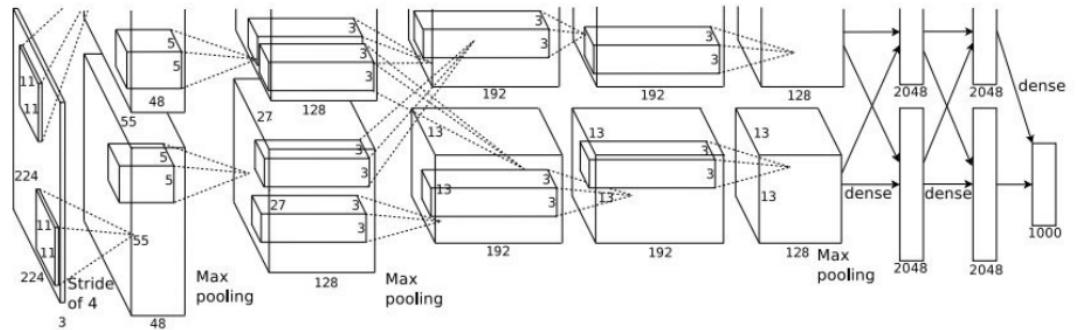
[13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1

[6x6x256] MAX POOL3: 3x3 filters at stride 2

[4096] FC6: 4096 neurons

[4096] FC7: 4096 neurons

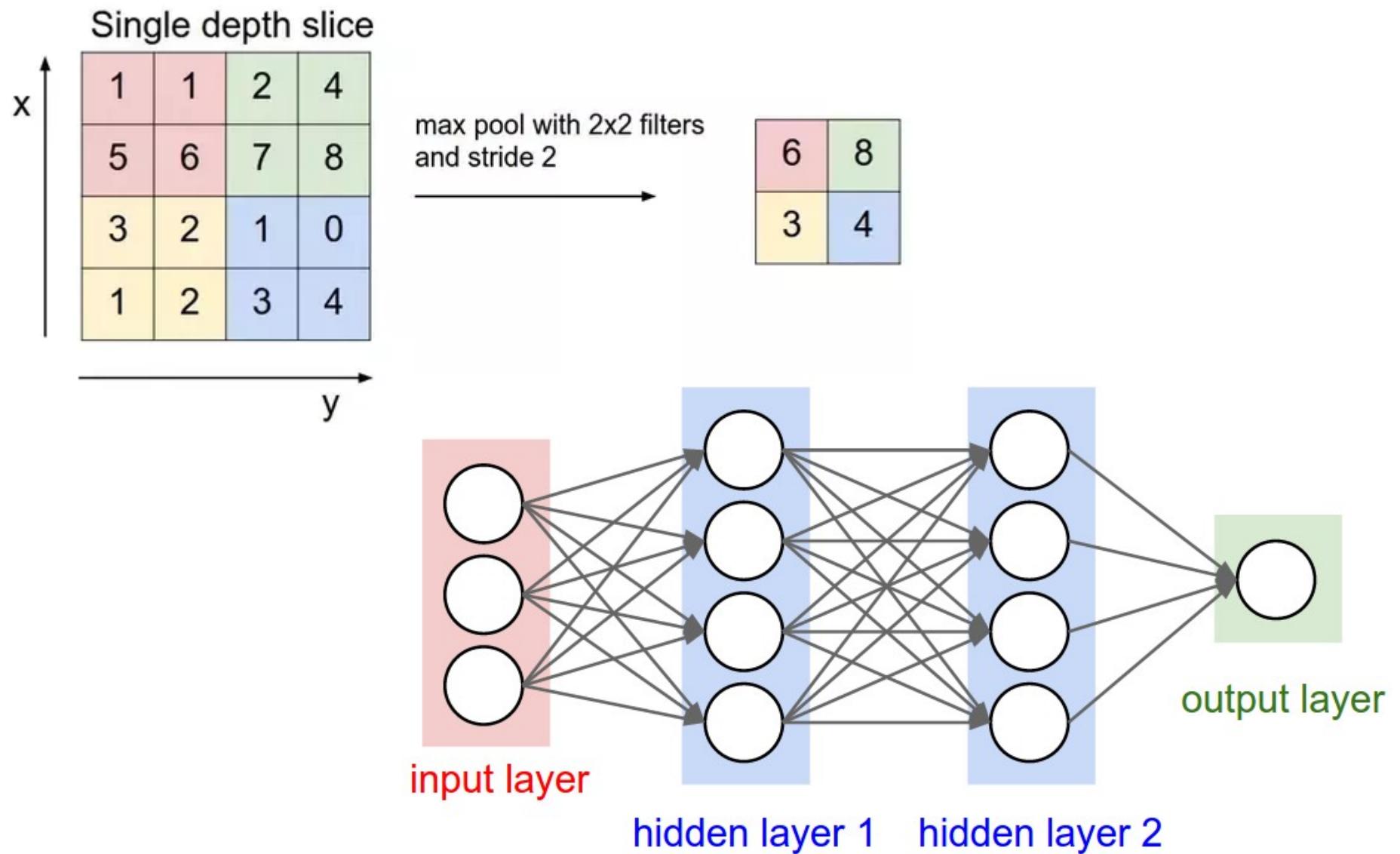
[1000] FC8: 1000 neurons (class scores)



Details/Retrospectives:

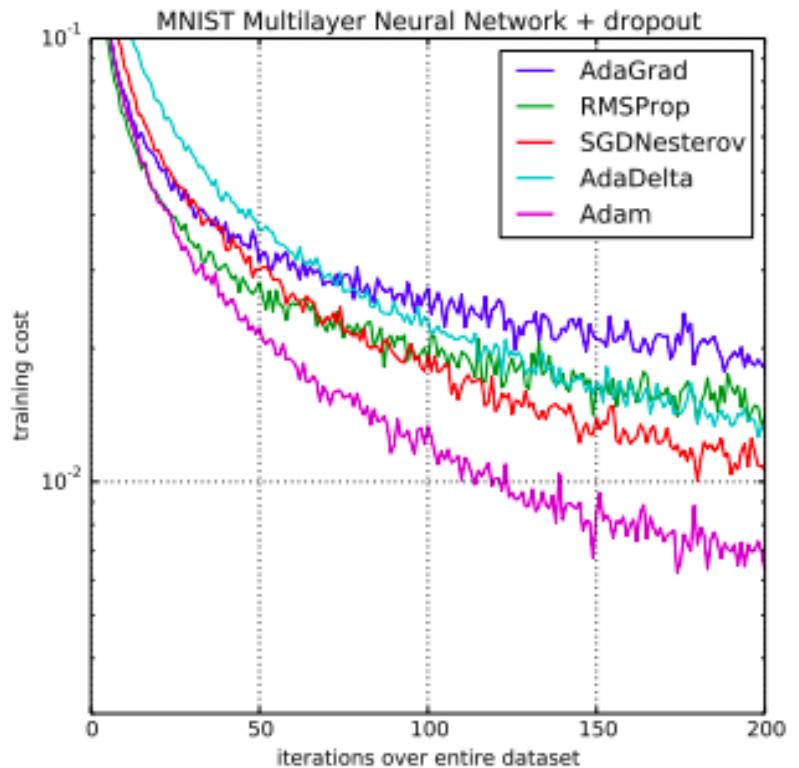
- first use of ReLU
- used Norm layers (not common anymore)
- heavy data augmentation
- dropout 0.5
- batch size 128
- SGD Momentum 0.9
- Learning rate 1e-2, reduced by 10 manually when val accuracy plateaus
- L2 weight decay 5e-4
- 7 CNN ensemble 18.2% -> 15.4%

Max pooling / FC network



Adam Optimizer

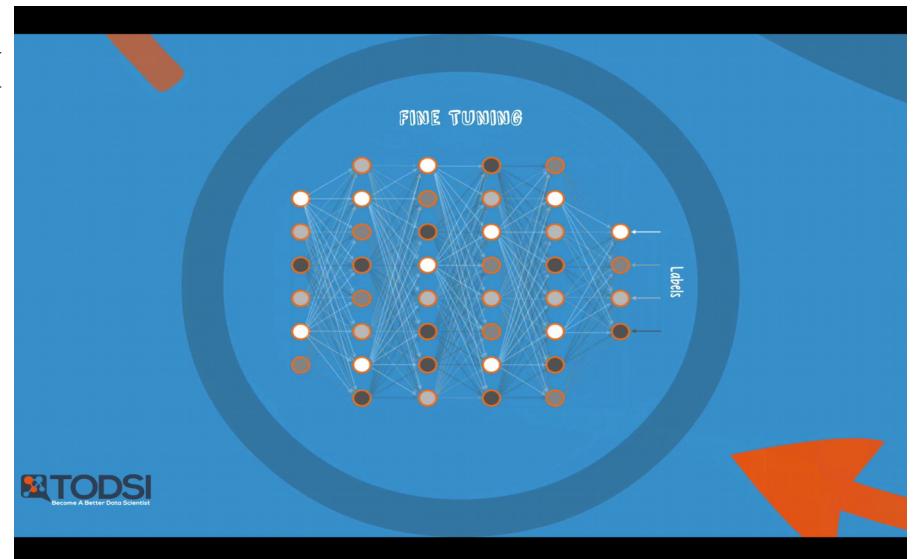
- 압도적인 성능!!



- 어떻게 작동하는가?
- <http://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/>

Xavier Initialization

- 초창기의 대부분의 학습 실패 원인은 weight, bias 초기값 설정 실패에 따른
- 다양한 초기화 방법들이 등장: ex] RBM
RBM?
- 우리는 간단하게 합시다.
(
- Xavier Initialization



Init method	maxout	ReLU	VLReLU	tanh	Sigmoid
LSUV	93.94	92.11	92.97	89.28	n/c
OrthoNorm	93.78	91.74	92.40	89.48	n/c
OrthoNorm-MSRA scaled	–	91.93	93.09	–	n/c
Xavier	91.75	90.63	92.27	89.82	n/c
MSRA	n/c†	90.91	92.43	89.54	n/c

How to use?

- `np.random.rand(fan_in, fan_out) / np.sqrt(fan_in/2)` <<- simple!!

```
86                                     feed_dict={X: x_data, Y: y_data})
87 •     print("\n" + "Hypothesis: ", h, "\n" + "Correct: ", c, "\n" + "Accuracy: ", a)
88
89     # 2010
90 • W = np.random.randn(fan_in, fan_out) / np.sqrt(fan_in)
91     # 2015
92 • W = np.random.randn(fan_in, fan_out) / np.sqrt(fan_in / 2)
93     # fan_in = n_feature_maps_in * receptive_field_height * receptive_field_width
94 • # fan_out = n feature maps out * receptive field height * receptive field width / max_pool_area
95
```

+ x code/practice4-2.py* ①7 ▲5 ①0 69:36

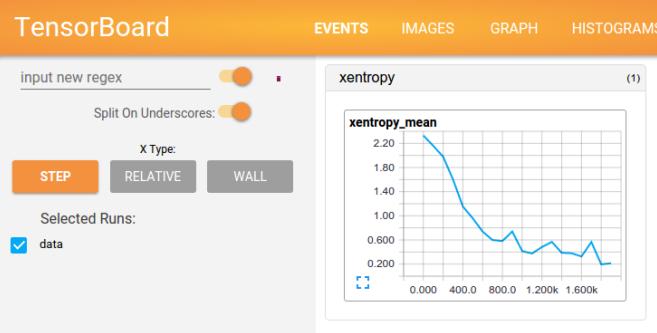
tensorboard

- https://www.tensorflow.org/get_started/summaries_and_tensorboard
- For debugging / accuracy

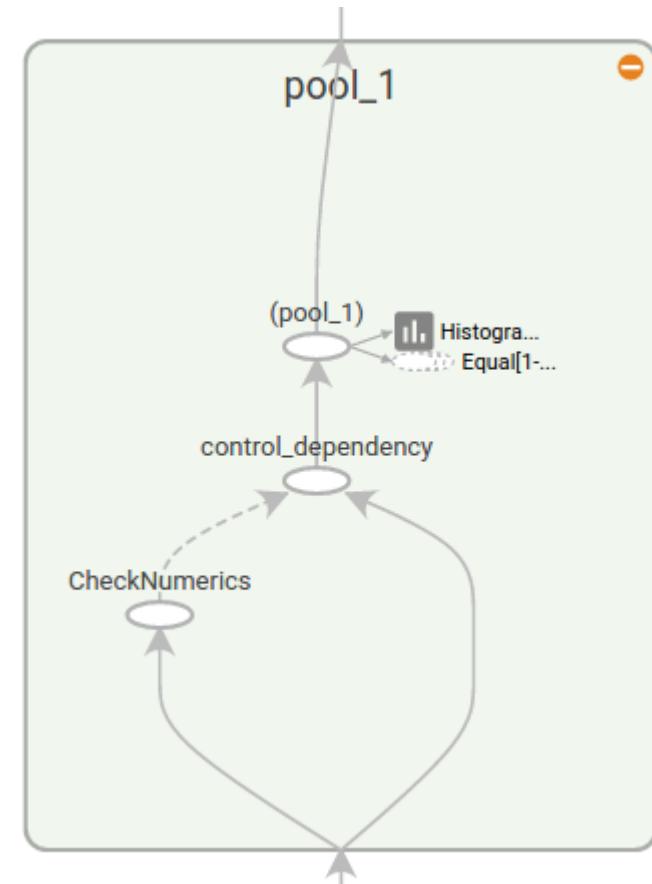
Getting Started
Getting Started With TensorFlow
MNIST For ML Beginners
Deep MNIST for Experts
TensorFlow Mechanics 101
tf.contrib.learn Quickstart
Building Input Functions with tf.contrib.learn
Logging and Monitoring Basics with tf.contrib.learn
[TensorBoard: Visualizing Learning](#)
TensorBoard: Embedding Visualization
TensorBoard: Graph Visualization
TensorBoard Histogram Dashboard
TensorFlow Versions

TensorBoard: Visualizing Learning

The computations you'll use TensorFlow for - like training a massive deep neural network - can be complex and confusing. To make it easier to understand, debug, and optimize TensorFlow programs, we've included a suite of visualization tools called TensorBoard. You can use TensorBoard to visualize your TensorFlow graph, plot quantitative metrics about the execution of your graph, and show additional data like images that pass through it. When TensorBoard is fully configured, it looks like this:



This screenshot shows the TensorBoard interface. The top navigation bar has tabs for EVENTS, IMAGES, GRAPH, and HISTOGRAMS. The EVENTS tab is selected. Below the tabs, there's a search bar labeled 'input new regex' and a switch labeled 'Split On Underscores'. Underneath are buttons for 'STEP' (which is selected), 'RELATIVE', and 'WALL'. A checkbox 'Selected Runs:' is checked with 'data' selected. To the right is a line chart titled 'xentropy_mean' with a single blue line showing values decreasing from approximately 2.2 at step 0 to around 0.2 at step 1,600k. At the bottom left, there's a note about the tutorial and a video thumbnail of a person speaking.



Practice Lab

1. CNN with MNIST
2. xavier initialization / adam optimizer
3. tensorboard