AI School 6기 3주차

파이썬 기초3

딥러닝 학습 방법론

다양한 학습 방법론을 적용한 필기체 인식기 개발

AI School 6기 3주차

파이썬 기초3

파이썬의 제어문 – for

- 리스트 내포

```
numbers = [1, 2, 3, 4, 5]
result = []
for n in numbers:
    if n % 2 == 1:
        result.append(n*2)
print(result)
```

```
result = [n*2 for n in numbers if n % 2 == 1]
print(result)
```

파이썬의 자료형 – tuple

- 불변한 순서가 있는 객체의 집합
- List형과 비슷하지만 한 번 생성되면 값을 변경할 수 없음

```
t1 = (1, 2, 3)
print(t1)
print(len(t1))
print(t1[0])
print(t1[:2])
del t1[0]
t1[0] = 4
t2 = (4,)
print(t2)
print(t1*3)
print(t1 + t2)
```

파이썬의 제어문 – while

- while 반복문

```
count = 0
while count < 10:
    count += 1
    print(count)
```

```
prompt = """
1. Add
2. Del
3. Quit"""
number = 0
while number != 3:
    print(prompt)
    number = int(input("Enter number:"))
```

파이썬의 제어문 – while

- break

```
coffee = 3
while True:
    money = int(input("돈을 넣어 주세요: "))
    if money == 300:
        print("맛있게 드세요.")
        coffee = coffee -1
    elif money > 300:
        print("거스름돈은 %d원입니다." % (money -300))
        print("맛있게 드세요.")
        coffee = coffee -1
    else:
        print("%d 더 넣어주세요." % (300 - money))
    if coffee == 0:
        print("커피가 다 떨어졌습니다. 판매를 중지 합니다.")
        break
```

파이썬의 제어문 – while

- continue

```
coffee = 3
while coffee > 0:
    print(f'남은 커피: {coffee}')
    money = int(input("돈을 넣어 주세요: "))
    if money < 300:
        continue
    coffee -= 1
    print("맛있게 드세요.")
```

숙제1

- while문을 사용해 1부터 1000까지의 자연수 중 3의 배수의 합을 구하세요.

- While 문을 사용해 다음과 같이 *들을 출력해보세요.

```
*
**
***
****
*****
```

- numbers = [1, 2, 3, 4, 5]

```
numbers = [1, 2, 3, 4, 5]
result = []
for n in numbers:
    if n % 2 == 0:
        result.append(n+2)
```

위 코드를 리스트 내포를 이용해 한줄로 구현해보세요.

파이썬의 파일 입출력

- w: 쓰기모드 – 파일에 내용을 쓸 때 사용
- a: 추가모드 – 파일의 마지막에 새로운 내용을 추가할 때 사용

```
f = open("./write.txt", 'w', encoding='utf-8')
f.write("file write")
f.close()
```

```
f = open("./write.txt", 'w', encoding='utf-8')
for i in range(1, 10):
    data = f'line {i}₩n'
    f.write(data)
f.close()
```

```
with open("./write.txt", 'w', encoding='utf-8') as f:
    for i in range(1, 10):
        data = f'line {i}₩n'
        f.write(data)
```

```
f = open("./write.txt", 'a', encoding='utf-8')
for i in range(10, 20):
    data = f'line {i}₩n'
    f.write(data)
f.close()
```

파이썬의 파일 입출력

- r: 읽기모드 – 파일을 읽기만 할 때 사용

```
f = open("./write.txt", 'r',
encoding='utf-8')
line = f.readline()
print(line)
f.close()
```

```
f = open("./write.txt", 'r',
encoding='utf-8')
line = f.readline()
while line:
    print(line)
    line = f.readline()
f.close()
```

```
f = open("./write.txt", 'r',
encoding='utf-8')
lines = f.readlines()
for line in lines:
    print(line)
f.close()
```

```
f = open("./write.txt", 'r',
encoding='utf-8')
content = f.read()
print(content)
f.close()
```

```
f = open("./write.txt", 'r',
encoding='utf-8')
content = f.read(6)
print(content)
content = f.read(14)
print(content)
f.seek(0)
content = f.read(14)
print(content)
f.close()
```

숙제2

- 주어진 fileIO.txt 파일을 읽어 Key는 성이고 Value는 나이인 딕셔너리 name_age에 정보들을 할당한 후 출력하세요.

입력 파일 예시:
Kim 32
Lee 34
Park 39
Choi 28
Cho 25

결과: {‘Kim’:32, ‘Lee’:34…}

AI School 6기 3주차

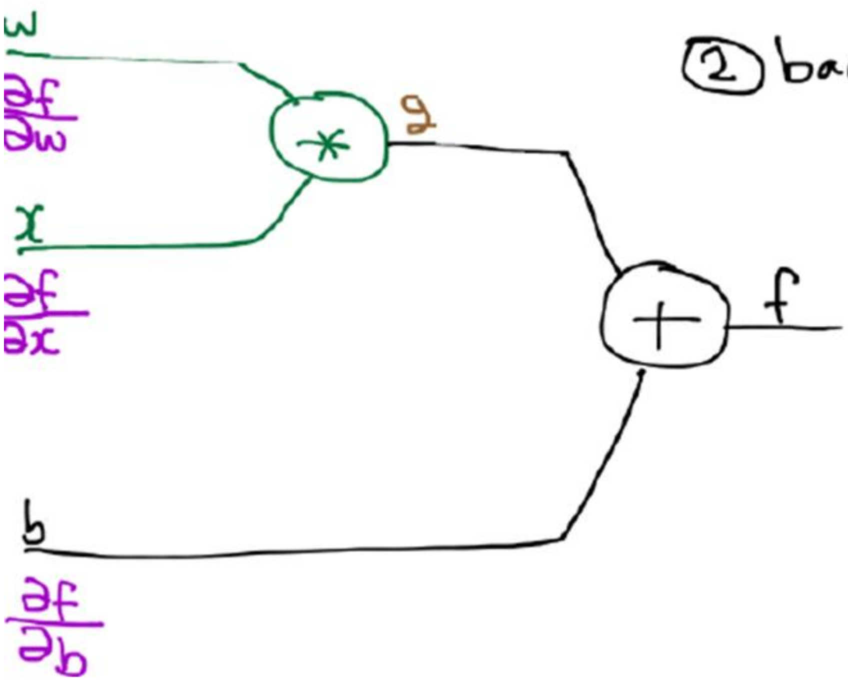
딥러닝 학습 방법론

Back-Propagation

$$f = wx + b, \quad g = wx, \quad f = g + b$$

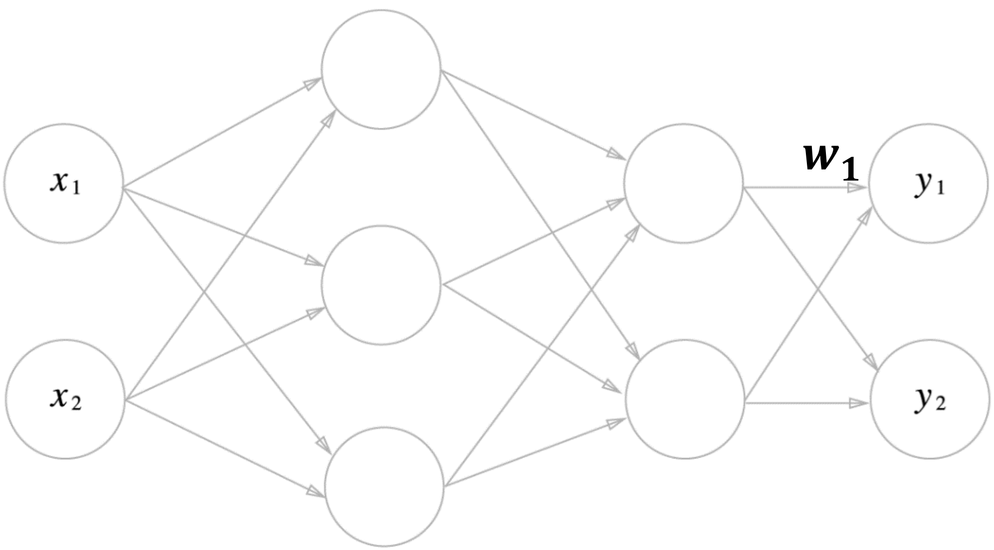
① forward ($w = -2$, $x = 5$, $b = 3$)

② backward



AdaGrad[1]

- 개별 가중치에 적응적으로 (adaptive) 학습률을 조정하면서 학습을 진행
- 현재까지 따라서 많이 갱신된 가중치는 학습률을 낮아짐
- 즉, 학습률 감소가 개별 가중치 마다 다르게 적용



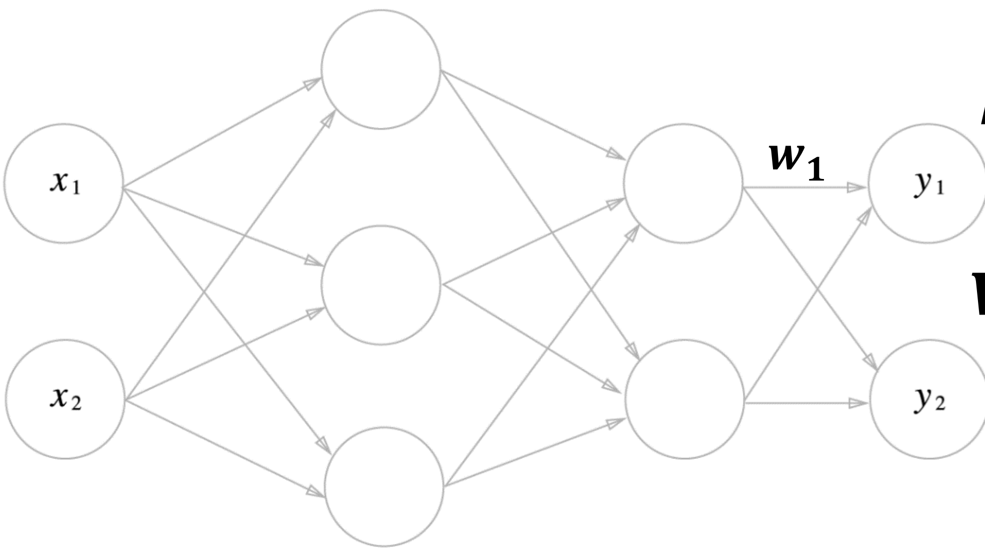
$$h = h + \left(\frac{\partial E}{\partial w_1}\right)^2$$

$$w_1^{t+1} = w_1^t - \varepsilon \frac{1}{\sqrt{h}} \left(\frac{\partial E}{\partial w_1}\right)$$

[1] John Duchi, Elad hazan, and Yoram Singer, "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization, JMLR 2011

RMSProp

- AdaGrad의 단점을 해결하기 위한 방법
- AdaGrad의 식에서 gradient의 제곱값을 더하는 방식이 아니라 지수평균으로 대체
- Gradient가 무한정 커지는 것을 방지



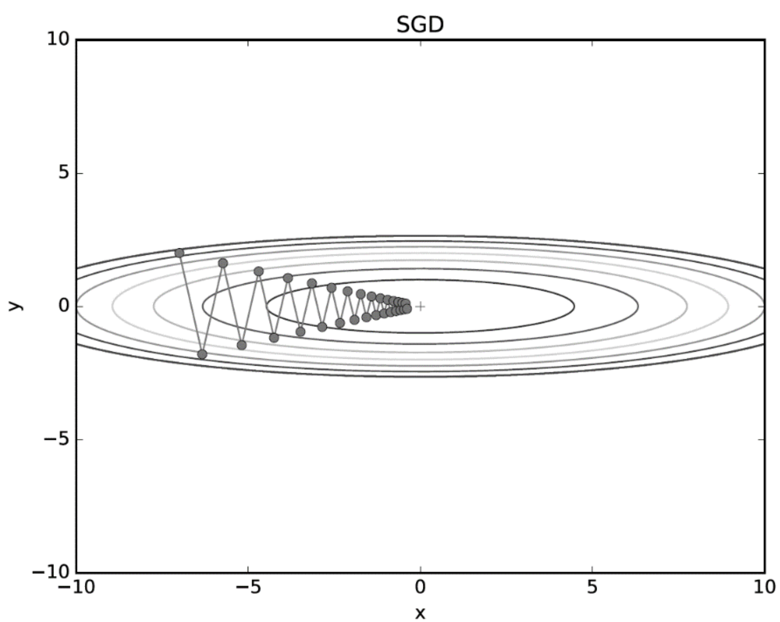
$$h = rh + (1 - r)\left(\frac{\partial E}{\partial w_1}\right)^2$$

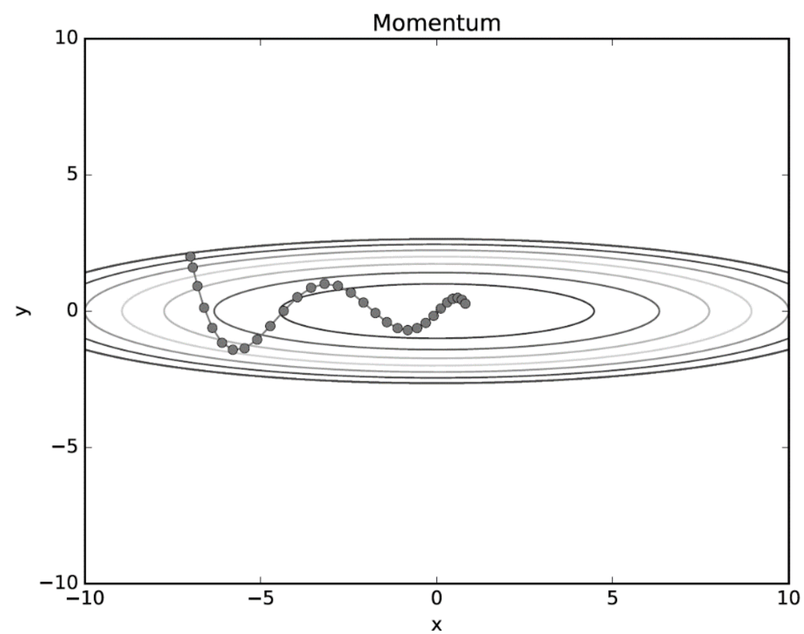
$$w_1^{t+1} = w_1^{t} - \varepsilon \frac{1}{\sqrt{h}}\left(\frac{\partial E}{\partial w_1}\right)$$

Momentum

- 가중치의 업데이트 값에 **이전 업데이트 값의 일정 비율을 더해줌**
- 즉, Gradient decent를 통해 이동하는 과정에 관성을 주는 것
- Adam[1]: AdaGrad (RMSProp) 와 Momentum을 융합한 기법

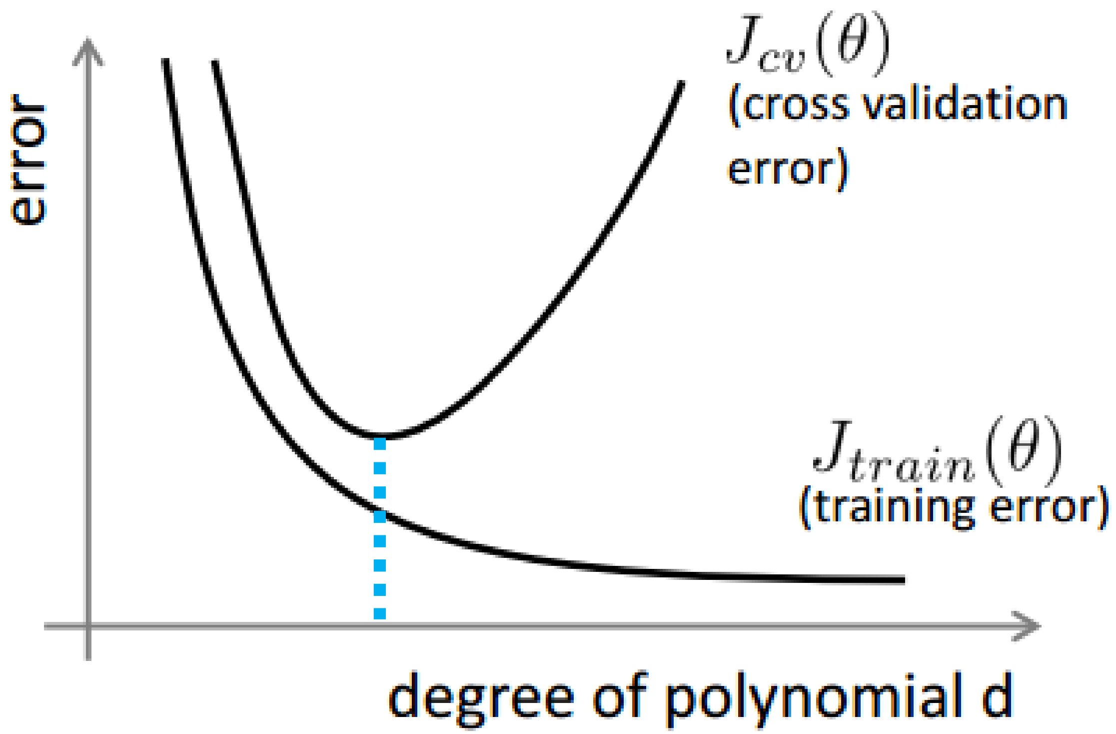
$$w^{t+1} = w^{t} - \varepsilon \nabla E_t + \mu \triangle w^{t-1}$$





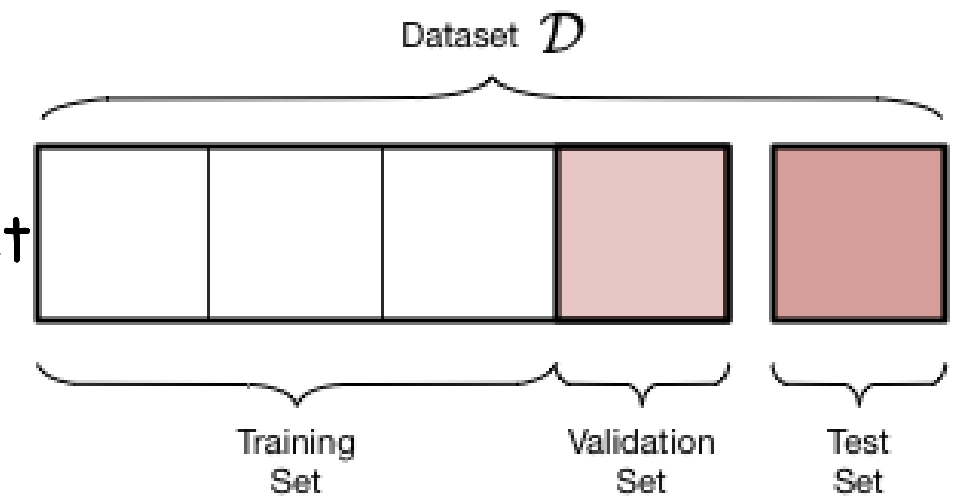
[1] Diederik Kingma and Jimmy Ba, Adam: A Method for Stochastic Optimazation, ICLR 2015

Early Stopping

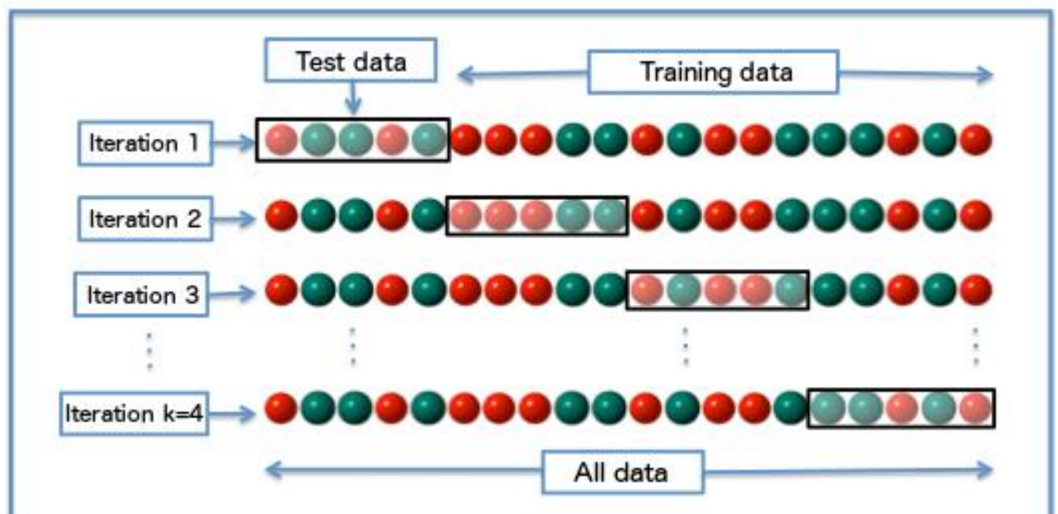


Training, Test, Validation (development) set

Training, Test, Validation set

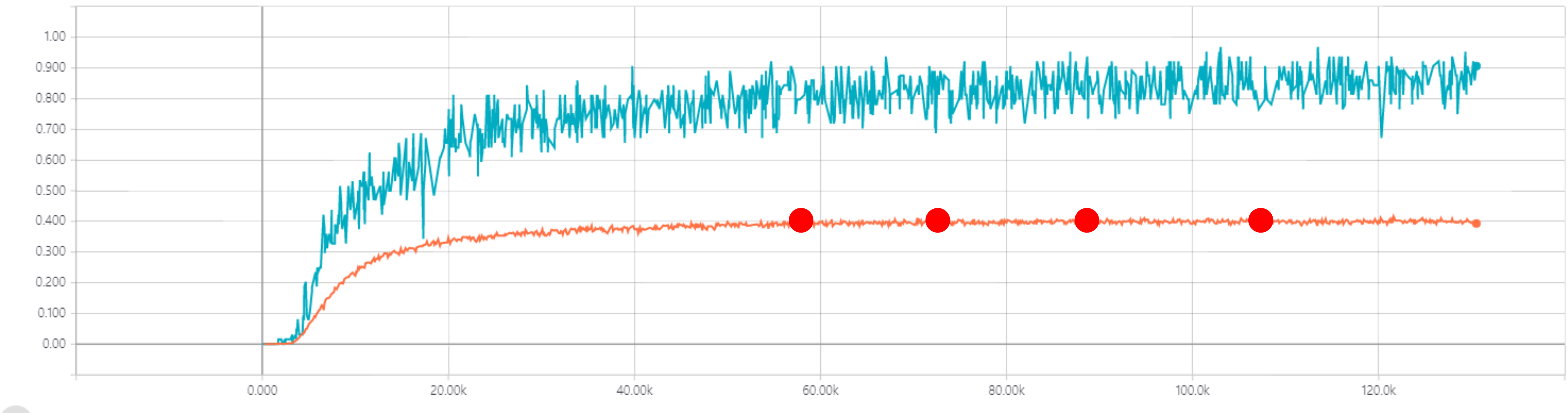


cross validation



Early Stopping

accuracy_1



loss_1

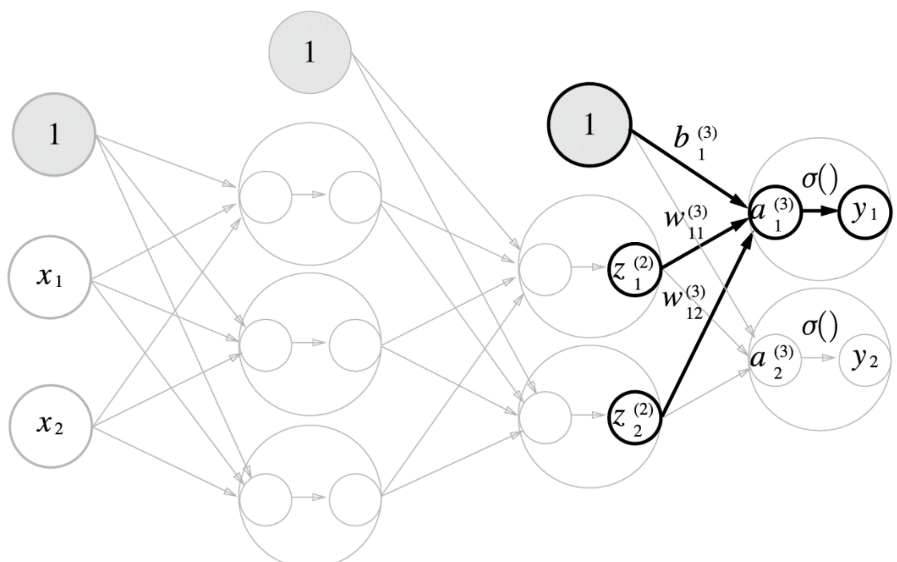
Weight Initialization

- 기존에는 가중치의 초깃값을 정규분포를 따르는 임의 값으로 정함 (예 - 평균: 0 , 표준편차 0.1)
  (가중치의 초깃값을 모두 0으로 할 경우 backpropagation 시 모든 가중치의 값이 똑같이 갱신되기 때문에 학습이 제대로 이뤄지지 않음)

- Xavier[1] 초깃값 (activation function이 sigmoid일 때), He 초기값 (activation function이 ReLU일 때)

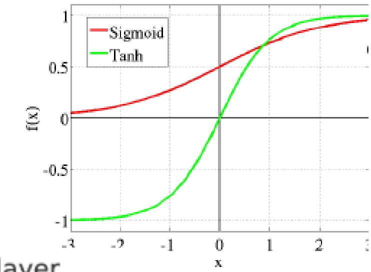
Xavier : 표준편차가 $\frac{1}{\sqrt{n}}$ 인 정규분포로 초기화 (n은 앞 층의 노드 수)

He : 표준편차가 $\sqrt{\frac{2}{n}}$ 인 정규분포로 초기화 (n은 앞 층의 노드 수)



[1] Xavier Glorot and Yoshua Bengio, "Understanding the difficulty of training deep feedforward neural networks, AISTATS 2010

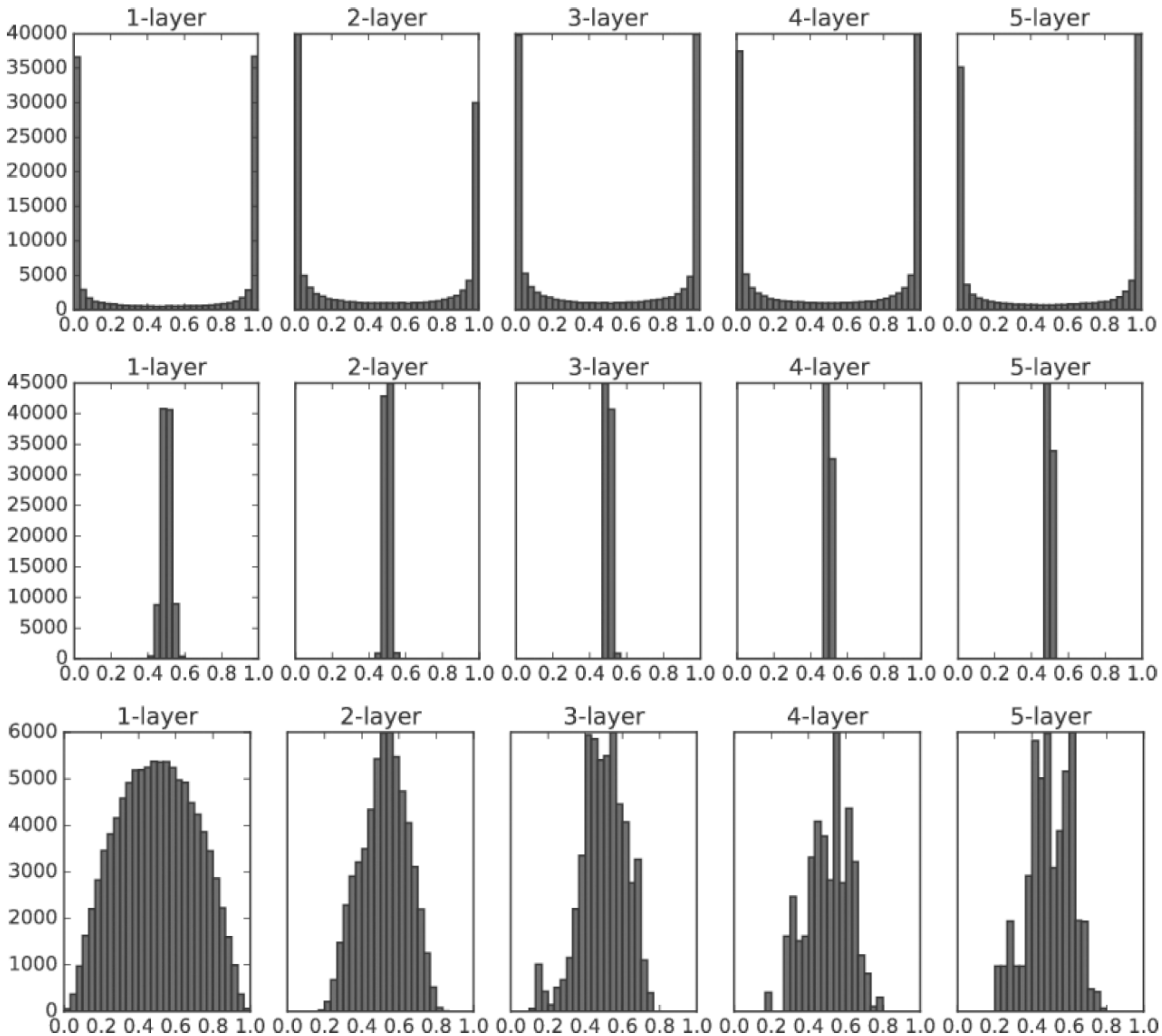
Weight Initialization (sigmoid)



표준편차: 1

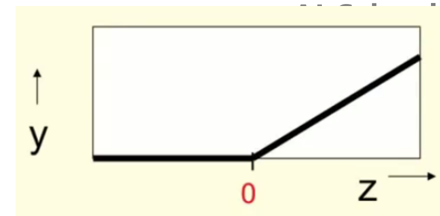
표준편차: 0.01

Xavier

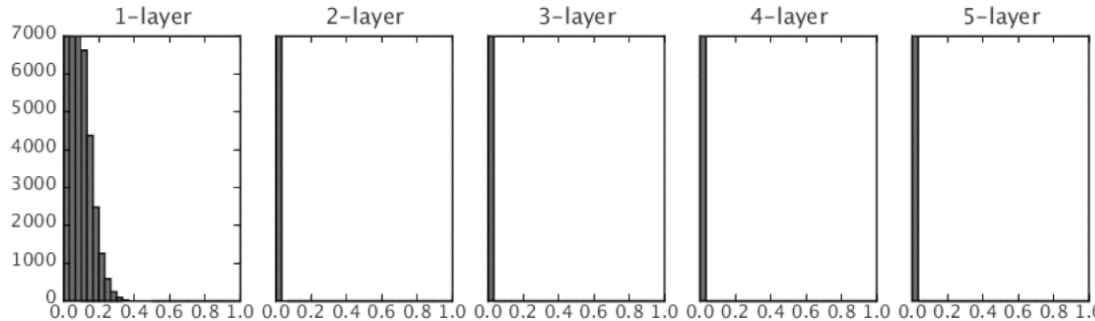


x 축: 활성화값

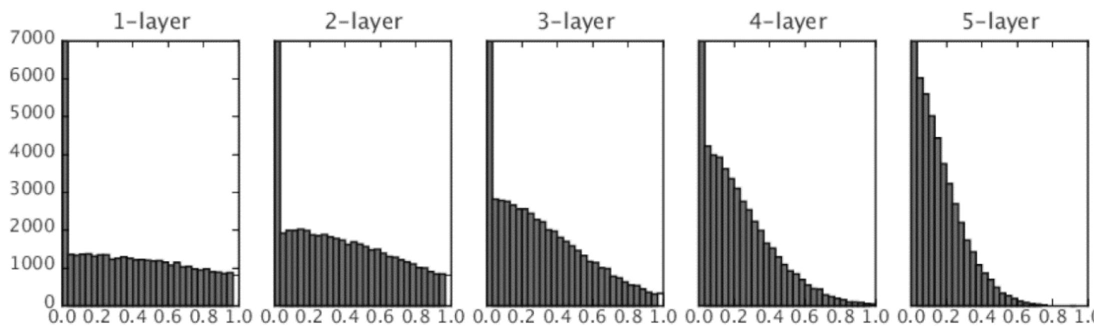
Weight Initialization (ReLU)



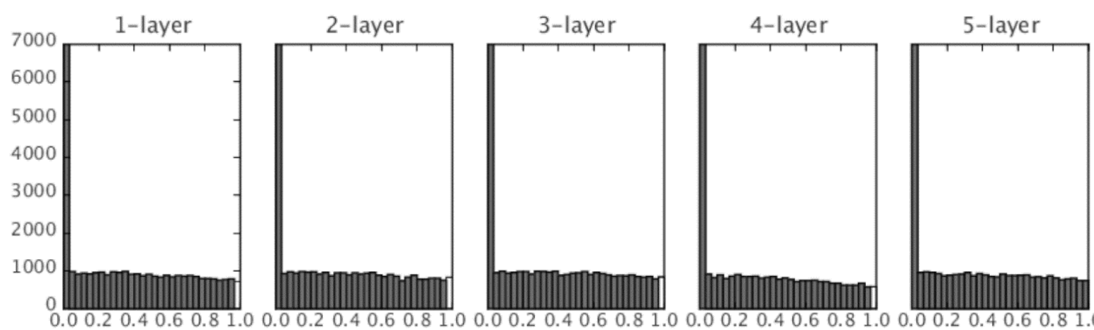
표준편차: 0.01



Xavier



He



He 초깃값을 사용한 경우

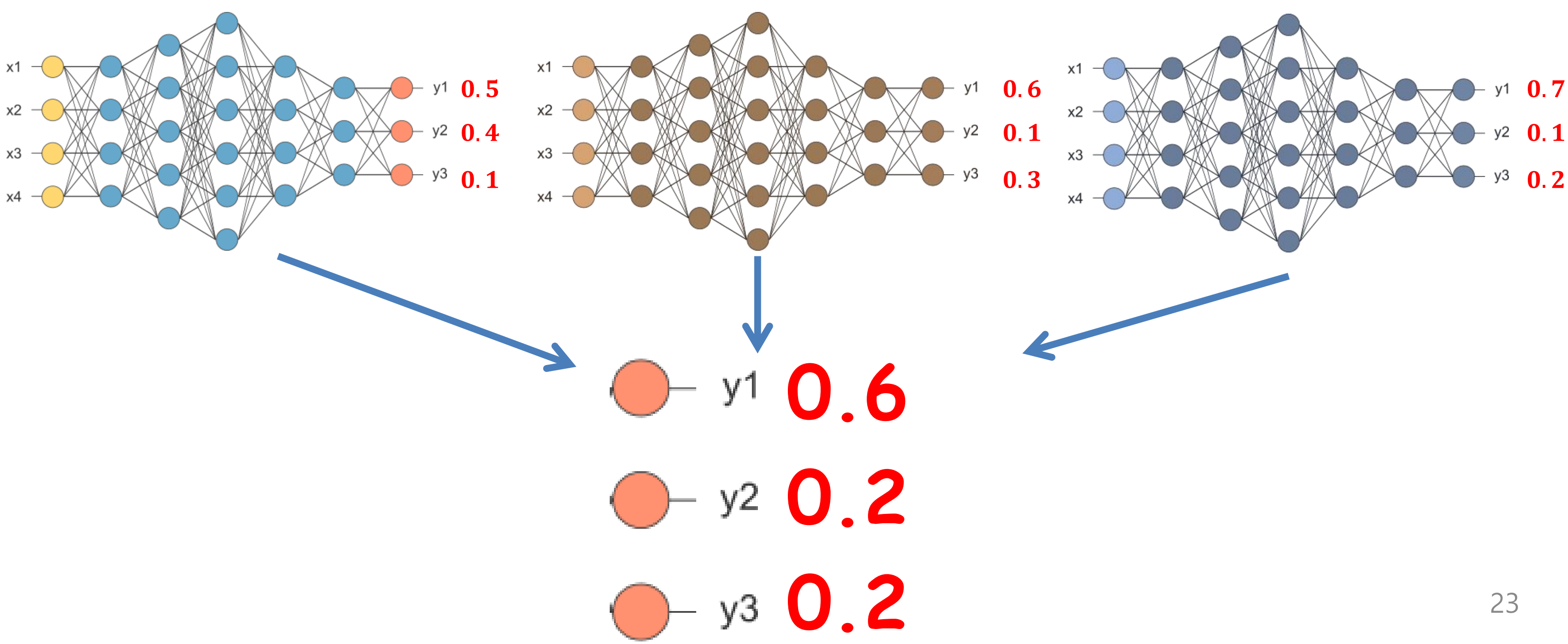
Dropout

- Background
  - When the network has enough hidden units to model it accurately
  - When there is only a limited amount of labeled training data

➡ Overfitted

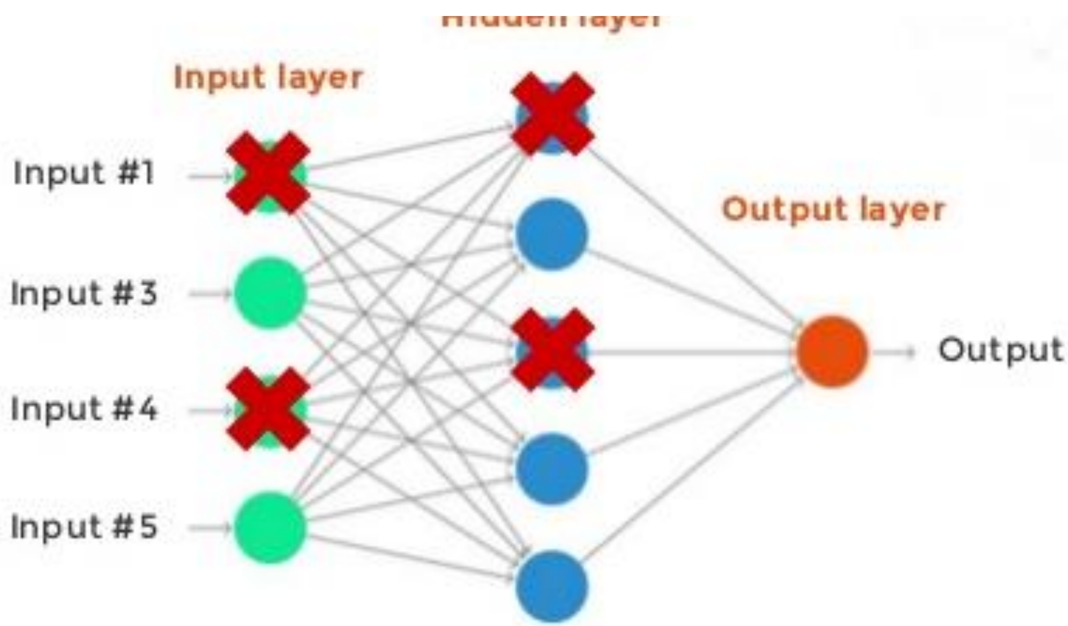
- Main Idea : averaging many models is always good

➔ How a single model can learn as if it's averaging many models

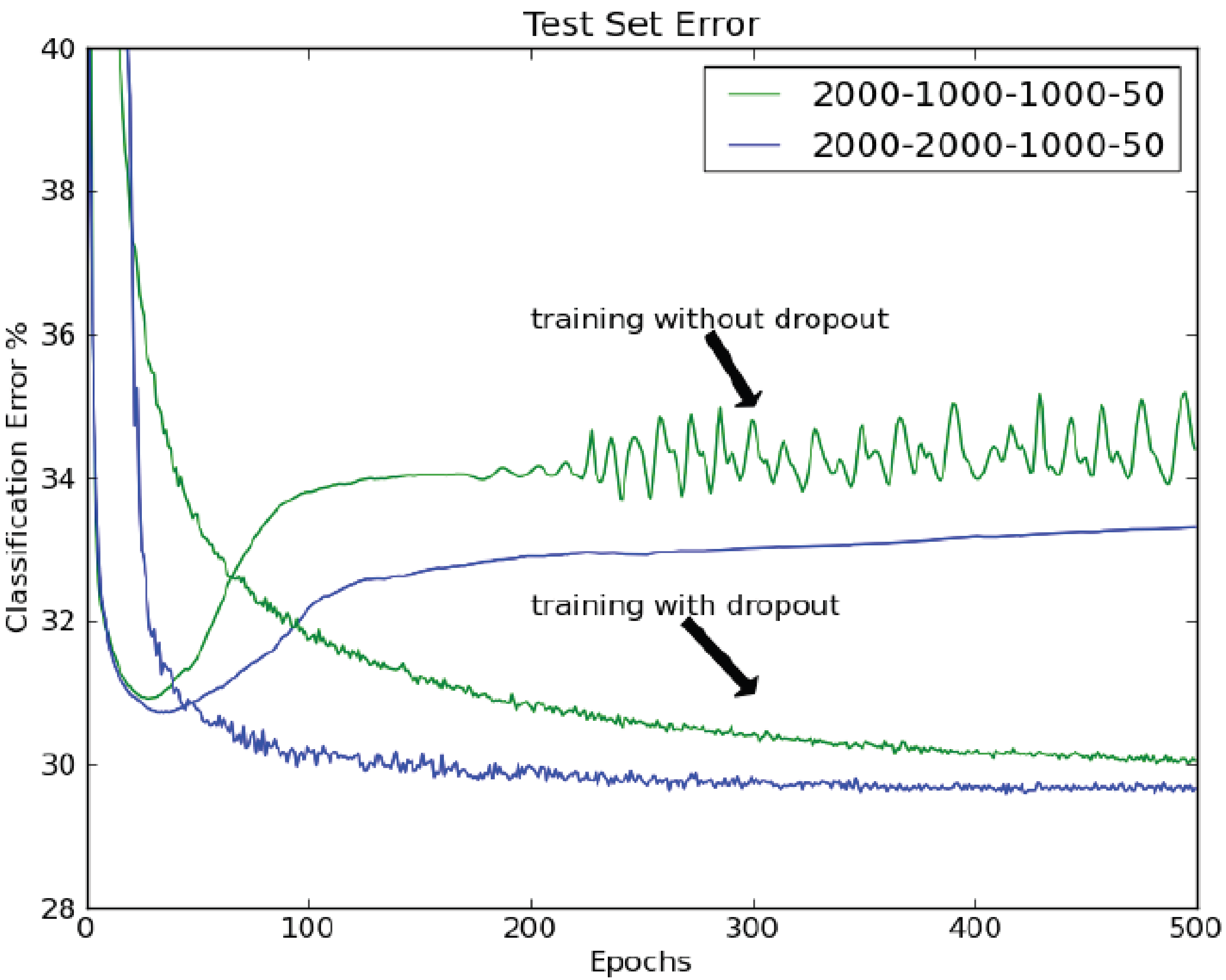


Dropout

- How it works
  - Each time we present a training example, we randomly omit each hidden unit with a probability of 0.5
  - So we are randomly sampling from 2^n different architectures
  - Efficient way of performing model averaging with neural networks



Dropout


Test Set Error
2000-1000-1000-50
2000-2000-1000-50
training without dropout
training with dropout
Classification Error %
Epochs
40
38
36
34
32
30
28
0
100
200
300
400
500

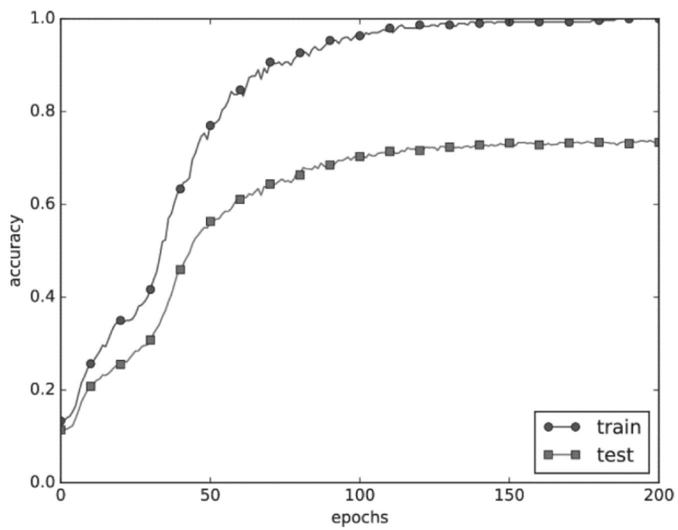
Weight decay, Weight restriction (Parameter Norm Penalties)[1]

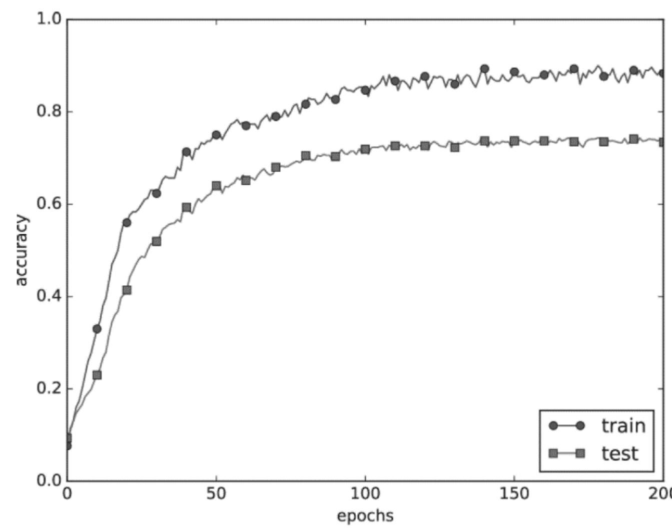
- 자유도가 높을수록 오버피팅 될 가능성이 높음

Weight decay: $$E_t = \frac{1}{N_t}\sum_{n \in D_t} E_n + \frac{\lambda}{2}\underbrace{||w||^2}_{\text{L2-norm}}$$

$$w^{t+1} = w^t - \varepsilon\left(\frac{1}{N_t}\sum \nabla E_n + \lambda w^t\right)$$

Weight restriction: $||w||^2 < c$

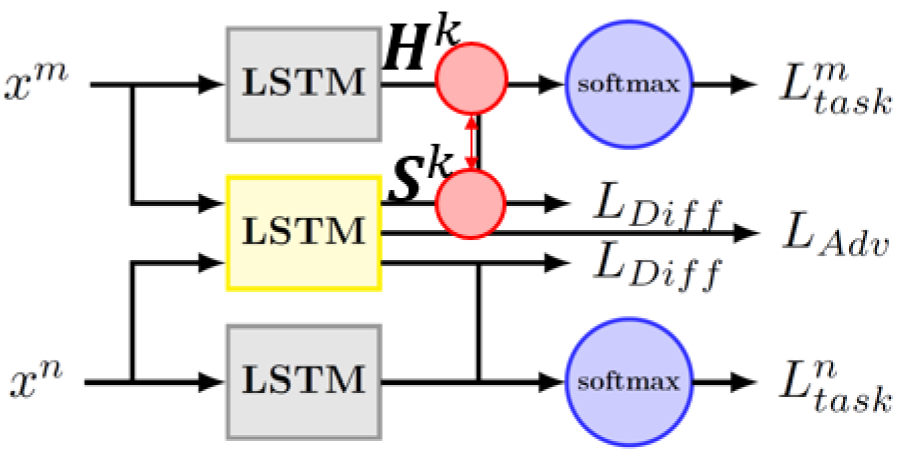




[1] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, DEEP LEARNING, Chapter 7

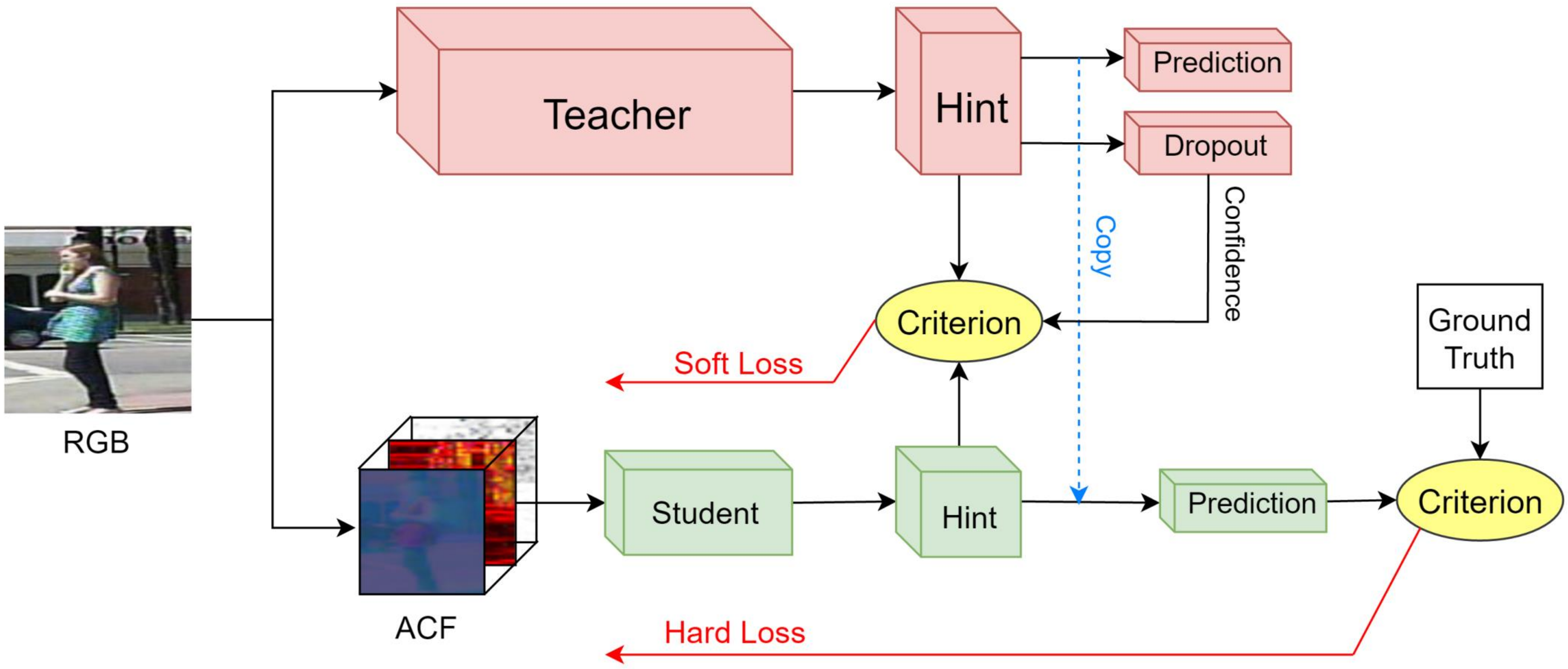
Orthogonality

- They introduce orthogonality constraints, which penalize redundant latent representations.



$$L_{\text{diff}} = \sum_{k=1}^{K} \left\| {\mathbf{S}^k}^\top \mathbf{H}^k \right\|_F^2$$

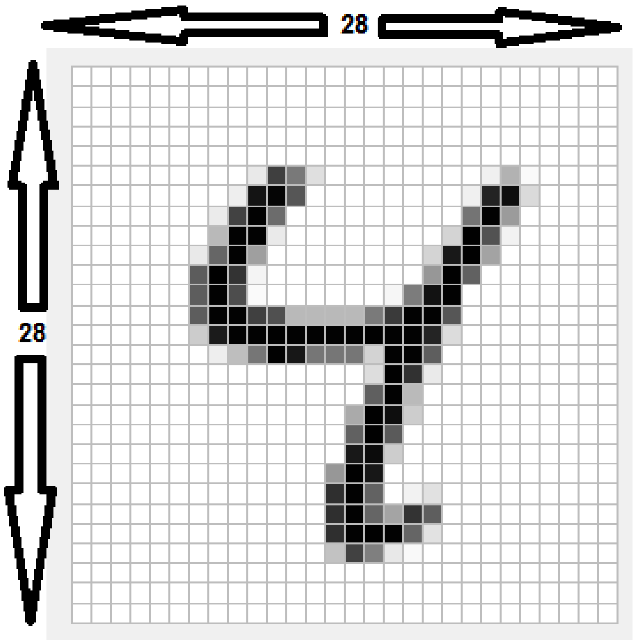
Knowledge distillation



AI School 6기 3주차

다양한 학습 방법론을 적용한 필기체 인식기 개발

MNIST data

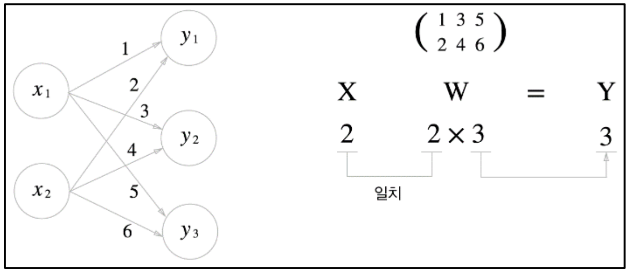



```
# MNIST data image of shape 28 * 28 = 784
X = tf.placeholder(tf.float32, [None, 784])
# 0 - 9 digits recognition = 10 classes
Y = tf.placeholder(tf.float32, [None, nb_classes])
```

MNIST data

```
import matplotlib.pyplot as plt
import numpy as np

from tensorflow.examples.tutorials.mnist import input_data

mnist = input_data.read_data_sets("MNIST_data/", one_hot=True,
validation_size=5000)

print(np.shape(mnist.validation.images))
print(np.shape(mnist.validation.labels))
print(np.shape(mnist.train.images))
print(np.shape(mnist.train.labels))
print(np.shape(mnist.test.images))
print(np.shape(mnist.test.labels))

plt.imshow(
        mnist.train.images[1].reshape(28, 28),
        cmap="Greys",
        interpolation="nearest",
    )
plt.show()
```

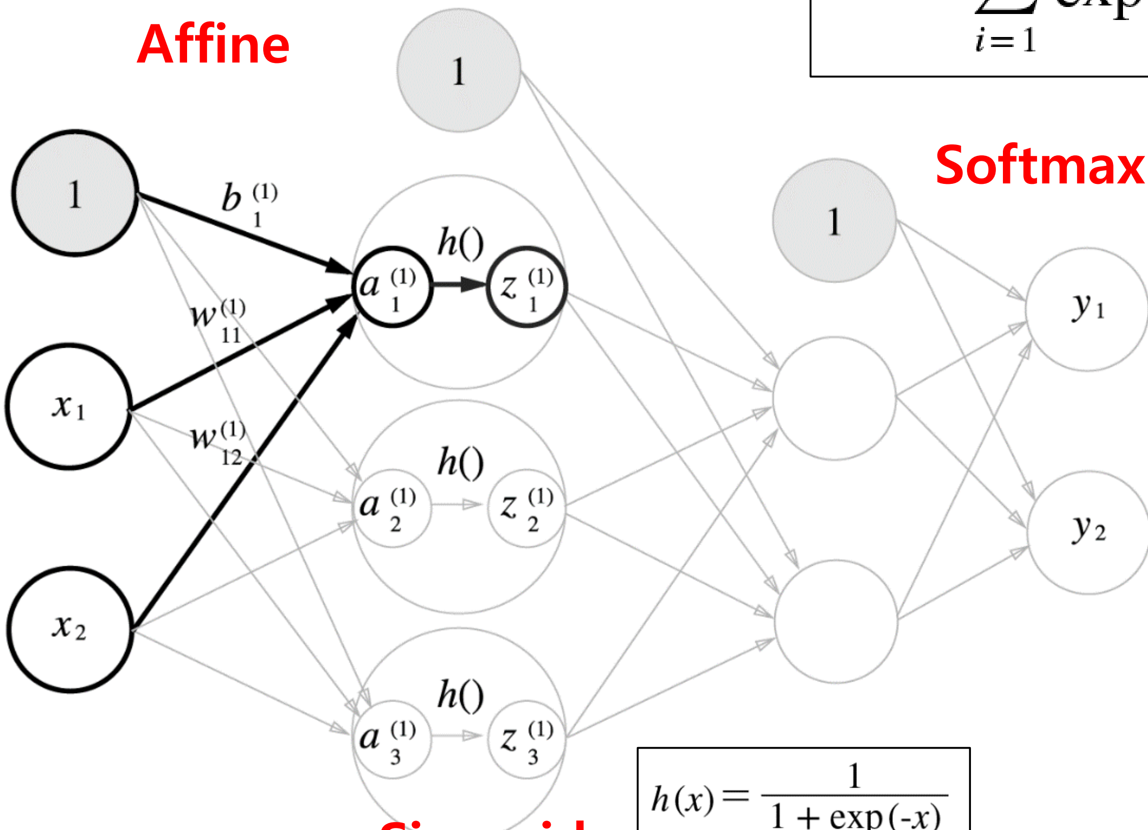
Feedforward



$$y_k = \frac{\exp(a_k)}{\sum_{i=1}^{n} \exp(a_i)}$$

Affine

Softmax



Loss function

$$E = -\sum_k t_k \log y_k$$

$$h(x) = \frac{1}{1 + \exp(-x)}$$

Sigmoid

ReLU

$$h(x) = \begin{cases} x & (x > 0) \\ 0 & (x \leq 0) \end{cases}$$

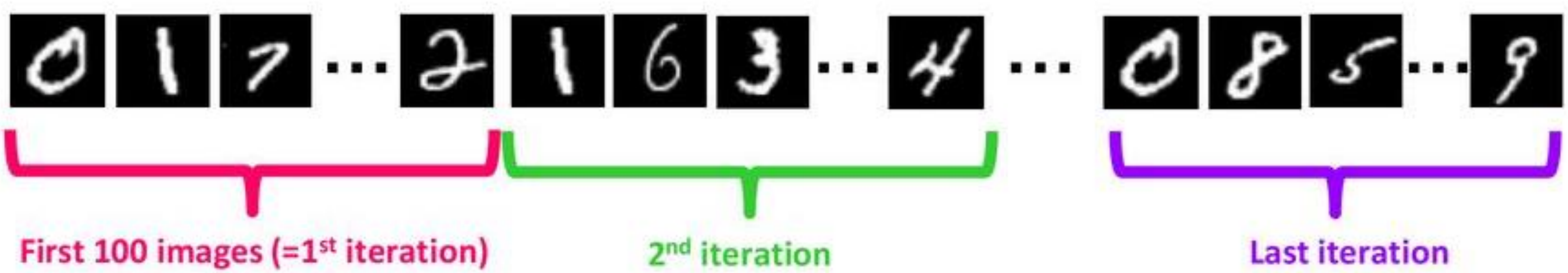
Affine, Activation

```
import tensorflow as tf
from tensorflow.examples.tutorials.mnist import input_data

mnist = input_data.read_data_sets("MNIST_data/", one_hot=True,
validation_size=5000)

X = tf.placeholder(tf.float32, [None, 784], name="X")
Y = tf.placeholder(tf.float32, [None, 10], name=“Y")
W1 = tf.Variable(tf.random_normal([784, 256]))
b1 = tf.Variable(tf.random_normal([256]))
L1 = tf.nn.relu(tf.matmul(X, W1) + b1)
W2 = tf.Variable(tf.random_normal([256, 256]))
b2 = tf.Variable(tf.random_normal([256]))
L2 = tf.nn.relu(tf.matmul(L1, W2) + b2)
W3 = tf.Variable(tf.random_normal([256, 10]))
b3 = tf.Variable(tf.random_normal([10]))
hypothesis = tf.nn.xw_plus_b(L2, W3, b3, name="hypothesis")
correct_prediction = tf.equal(tf.argmax(hypothesis, 1), tf.argmax(Y, 1))
accuracy = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))
```

Epoch, Batch size, Iterations

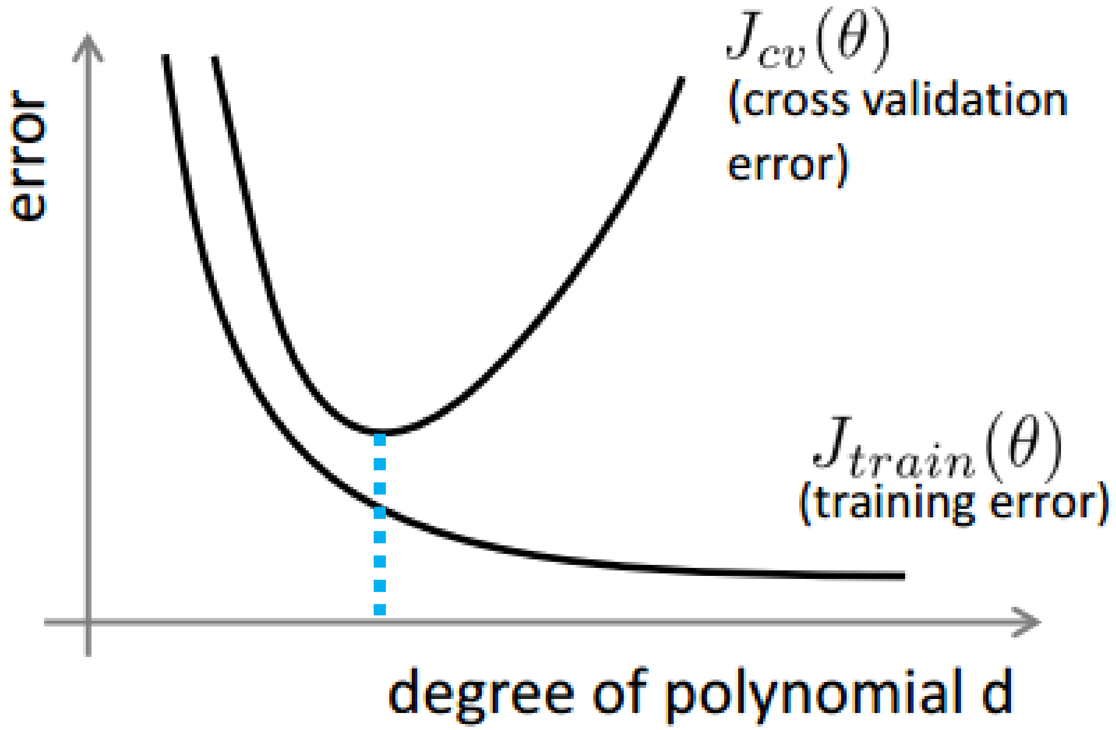
Example: MNIST data

- number of training data: **N=55,000**
- Let's take batch size of **B=100**



- How many iteration in each epoch? **55000/100 = 550** **1 epoch = 550 iteration**

Early Stopping



Early Stopping

```
training_epochs = 100
batch_size = 100

timestamp = str(int(time.time()))
out_dir = os.path.abspath(os.path.join(os.path.curdir, "runs", timestamp))
checkpoint_dir = os.path.abspath(os.path.join(out_dir, "checkpoints"))
checkpoint_prefix = os.path.join(checkpoint_dir, "model")
if not os.path.exists(checkpoint_dir):
    os.makedirs(checkpoint_dir)
saver = tf.train.Saver(tf.global_variables(), max_to_keep=3)
```

```
print('Epoch:', '%04d' % (epoch + 1), 'training cost =', '{:.9f}'.format(avg_cost))
val_accuracy= sess.run(accuracy, feed_dict={X: mnist.validation.images, Y:
mnist.validation.labels})
print('Validation Accuracy:', val_accuracy)
if val_accuracy > max:
    max = val_accuracy
    early_stopped = epoch + 1
    saver.save(sess, checkpoint_prefix, global_step=early_stopped)

print('Learning Finished!')
print('Validation Max Accuracy:', max)
print('Early stopped time:', early_stopped)
```

MNIST_eval.py

```
import tensorflow as tf
from tensorflow.examples.tutorials.mnist import input_data

mnist = input_data.read_data_sets("MNIST_data/", one_hot=True,validation_size=5000)

tf.flags.DEFINE_string("checkpoint_dir", "./runs/1570920722/checkpoints", "Checkpoint directory from training run")
FLAGS = tf.flags.FLAGS
checkpoint_file = tf.train.latest_checkpoint(FLAGS.checkpoint_dir)
graph = tf.Graph()
with graph.as_default():
    sess = tf.Session()
    with sess.as_default():
        saver = tf.train.import_meta_graph("{}.meta".format(checkpoint_file))
        saver.restore(sess, checkpoint_file)
```

MNIST_eval.py

```
.
.
.
        X = graph.get_operation_by_name("X").outputs[0]
        Y = graph.get_operation_by_name("Y").outputs[0]
        keep_prob = graph.get_operation_by_name("keep_prob").outputs[0]
        hypothesis = graph.get_operation_by_name("hypothesis").outputs[0]
        correct_prediction = tf.equal(tf.argmax(hypothesis, 1), tf.argmax(Y, 1))
        accuracy = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))
        test_accuracy = sess.run(accuracy, feed_dict={X: mnist.test.images, Y:
mnist.test.labels, keep_prob: 1.0})
        print('Test Max Accuracy:', test_accuracy)
```

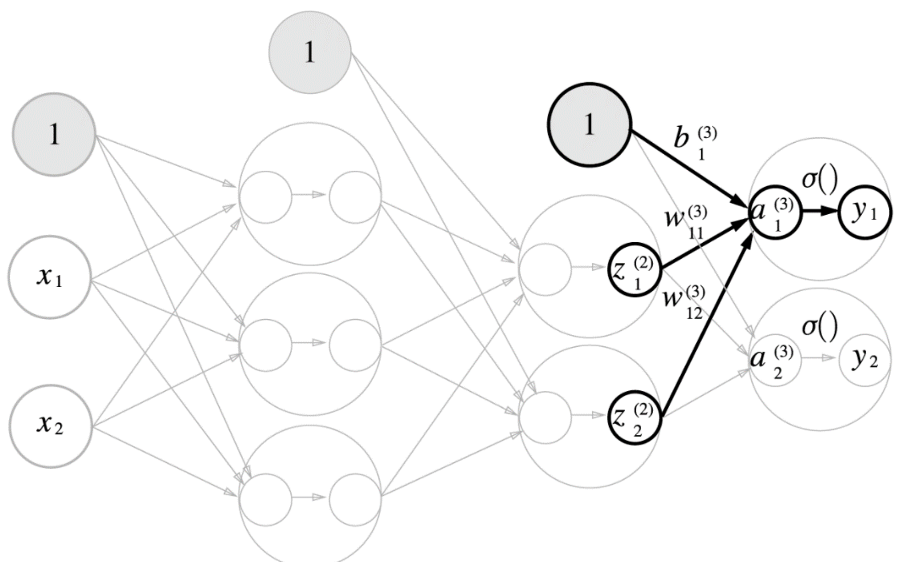
Weight Initialization

- 기존에는 가중치의 초깃값을 정규분포를 따르는 임의 값으로 정함 (예 - 평균: 0 , 표준편차 0.1)
  (가중치의 초깃값을 모두 0으로 할 경우 backpropagation 시 모든 가중치의 값이 똑같이 갱신되기 때문에 학습이 제대로 이뤄지지 않음)

- Xavier[1] 초깃값 (activation function이 sigmoid일 때), He 초기값 (activation function이 ReLU일 때)

Xavier : 표준편차가 $\frac{1}{\sqrt{n}}$ 인 정규분포로 초기화 (n은 앞 층의 노드 수)

He : 표준편차가 $\sqrt{\frac{2}{n}}$ 인 정규분포로 초기화 (n은 앞 층의 노드 수)



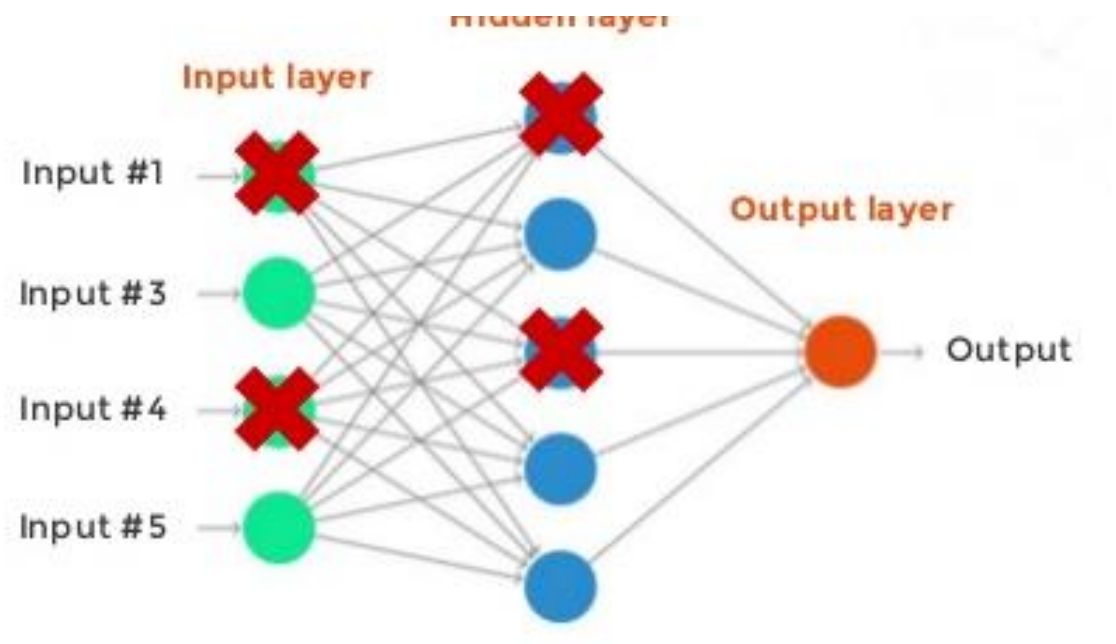
[1] Xavier Glorot and Yoshua Bengio, "Understanding the difficulty of training deep feedforward neural networks, AISTATS 2010

Weight Initialization

```
W1 = tf.get_variable("W1", shape=[784, 256], initializer=tf.contrib.layers.xavier_initializer())



b1 = tf.Variable(tf.random_normal([256]))
L1 = tf.nn.relu(tf.matmul(X, W1) + b1)

                                                                                He?
W2 = tf.get_variable("W2", shape=[256, 256],initializer=tf.contrib.layers.xavier_initializer())
b2 = tf.Variable(tf.random_normal([256]))
L2 = tf.nn.relu(tf.matmul(L1, W2) + b2)

W3 = tf.get_variable("W3", shape=[256, 10],initializer=tf.contrib.layers.xavier_initializer())
b3 = tf.Variable(tf.random_normal([10]))
hypothesis = tf.matmul(L2, W3) + b3
```

Dropout

- How it works
  - Each time we present a training example, we randomly omit each hidden unit with a probability of 0.5
  - So we are randomly sampling from 2^n different architectures
  - Efficient way of performing model averaging with neural networks



Dropout

```
keep_prob = tf.placeholder(tf.float32, name="keep_prob")

W1 = tf.get_variable("W1", shape=[784, 256], initializer=tf.initializers.he_normal())
b1 = tf.Variable(tf.random_normal([256]))
L1 = tf.nn.relu(tf.matmul(X, W1) + b1)
L1 = tf.nn.dropout(L1, keep_prob=keep_prob)

W2 = tf.get_variable("W2", shape=[256, 256],initializer=tf.initializers.he_normal())
b2 = tf.Variable(tf.random_normal([256]))
L2 = tf.nn.relu(tf.matmul(L1, W2) + b2)
L2 = tf.nn.dropout(L2, keep_prob=keep_prob)
.
.
.
feed_dict = {X: batch_xs, Y: batch_ys, keep_prob: 0.8}
.
.
.
print('Accuracy:', sess.run(accuracy, feed_dict={
    X: mnist.test.images, Y: mnist.test.labels, keep_prob:1.0}))
```

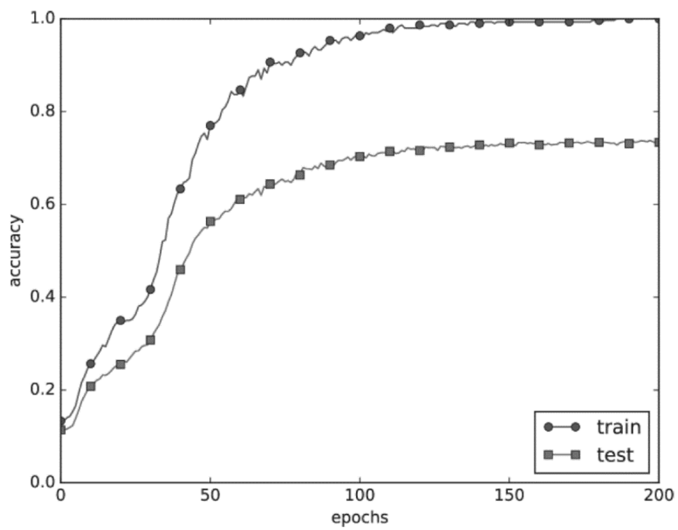
Weight decay, Weight restriction (Parameter Norm Penalties)[1]

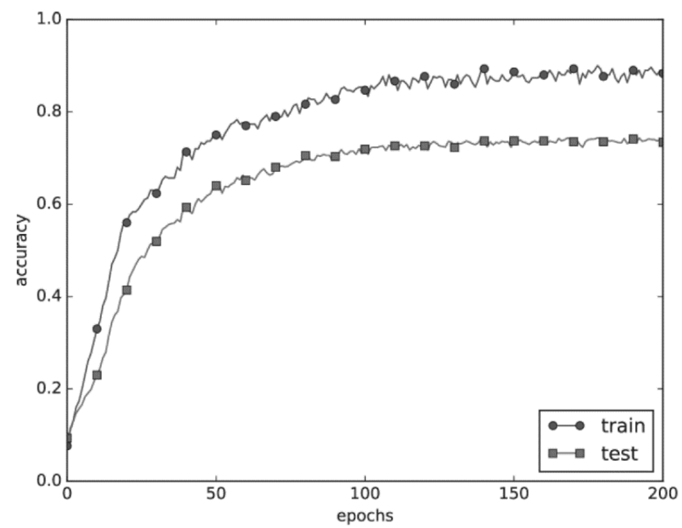
- 자유도가 높을수록 오버피팅 될 가능성이 높음

Weight decay: $$E_t = \frac{1}{N_t}\sum_{n \in D_t} E_n + \frac{\lambda}{2}\underbrace{||w||^2}_{\text{L2-norm}}$$

$$w^{t+1} = w^t - \varepsilon\left(\frac{1}{N_t}\sum \nabla E_n + \lambda w^t\right)$$

Weight restriction: $$||w||^2 < c$$





[1] Ian Goodfellow, Yoshua Bengio, and Aaron Courville, DEEP LEARNING, Chapter 7

Weight decay, Weight restriction (Parameter Norm Penalties)[1]

```
l2_loss = 0.0
W1 = tf.Variable(tf.random_normal([784, 256]))
b1 = tf.Variable(tf.random_normal([256]))
L1 = tf.nn.relu(tf.matmul(X, W1) + b1)
l2_loss +=tf.nn.l2_loss(W1)
l2_loss +=tf.nn.l2_loss(b1)
W2 = tf.Variable(tf.random_normal([256, 256]))
b2 = tf.Variable(tf.random_normal([256]))
L2 = tf.nn.relu(tf.matmul(L1, W2) + b2)
l2_loss +=tf.nn.l2_loss(W2)
l2_loss +=tf.nn.l2_loss(b2)
W3 = tf.Variable(tf.random_normal([256, 10]))
b3 = tf.Variable(tf.random_normal([10]))
hypothesis = tf.matmul(L2, W3) + b3
l2_loss +=tf.nn.l2_loss(W3)
l2_loss +=tf.nn.l2_loss(b3)

l2_loss_lambda = 0.001
cost = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(
    logits=hypothesis, labels=Y)) + l2_loss_lambda * l2_loss
```

Tensorboard

http://localhost:6006

```
accuracy = tf.reduce_mean(tf.cast(correct_prediction, tf.float32))
summary_op = tf.summary.scalar("accuracy", accuracy)
.
.
.
timestamp = str(int(time.time()))
out_dir = os.path.abspath(os.path.join(os.path.curdir, "runs", timestamp))
train_summary_dir = os.path.join(out_dir, "summaries", "train")
train_summary_writer = tf.summary.FileWriter(train_summary_dir, sess.graph)
val_summary_dir = os.path.join(out_dir, "summaries", "dev")
val_summary_writer = tf.summary.FileWriter(val_summary_dir, sess.graph)
checkpoint_dir = os.path.abspath(os.path.join(out_dir, "checkpoints"))
checkpoint_prefix = os.path.join(checkpoint_dir, "model")
if not os.path.exists(checkpoint_dir):
    os.makedirs(checkpoint_dir)
saver = tf.train.Saver(tf.global_variables(), max_to_keep=10)
max = 0
```

Tensorboard

http://localhost:6006

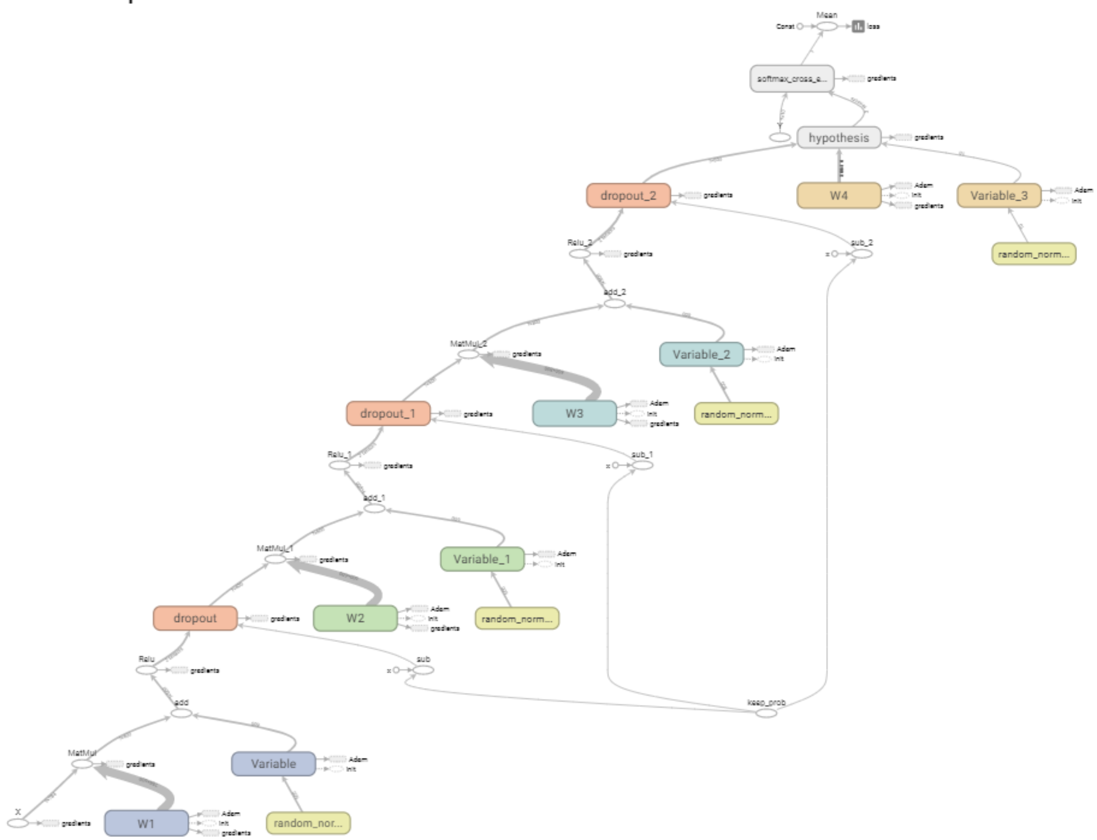
```
for i in range(total_batch):
        batch_xs, batch_ys = mnist.train.next_batch(batch_size)
        feed_dict = {X: batch_xs, Y: batch_ys, keep_prob: 0.8}
        c, _, a = sess.run([cost, optimizer, summary_op], feed_dict=feed_dict)
        avg_cost += c / total_batch

    print('Epoch:', '%04d' % (epoch + 1), 'training cost =', '{:.9f}'.format(avg_cost))
    train_summary_writer.add_summary(a, early_stopped)
    val_accuracy, summaries = sess.run([accuracy, summary_op], feed_dict={X:
mnist.validation.images, Y: mnist.validation.labels, keep_prob: 1.0})
    val_summary_writer.add_summary(summaries, early_stopped)
print('Validation Accuracy:', val_accuracy)
    if val_accuracy > max:
        max = val_accuracy
        early_stopped = epoch + 1
        saver.save(sess, checkpoint_prefix, global_step=early_stopped)
```

Tensorboard

http://localhost:6006

```
(aischool) C:\Users\82102\Anaconda3\envs\aischool>tensorboard --logdir=C:\Users\82102\PycharmProjects\aischool\MNIST\runs\1570923483
```

Main Graph



숙제3

오늘 만든 코드에서

1) He 초기화 적용해보기 (모든 weight)

2) Dropout 적용해보기 (keep_prob:0.8)

3) Weight decay 반영해보기 (모든 weight)

Q&A

과제 제출 : dha8102@naver.com