1 0 주 차   교 육

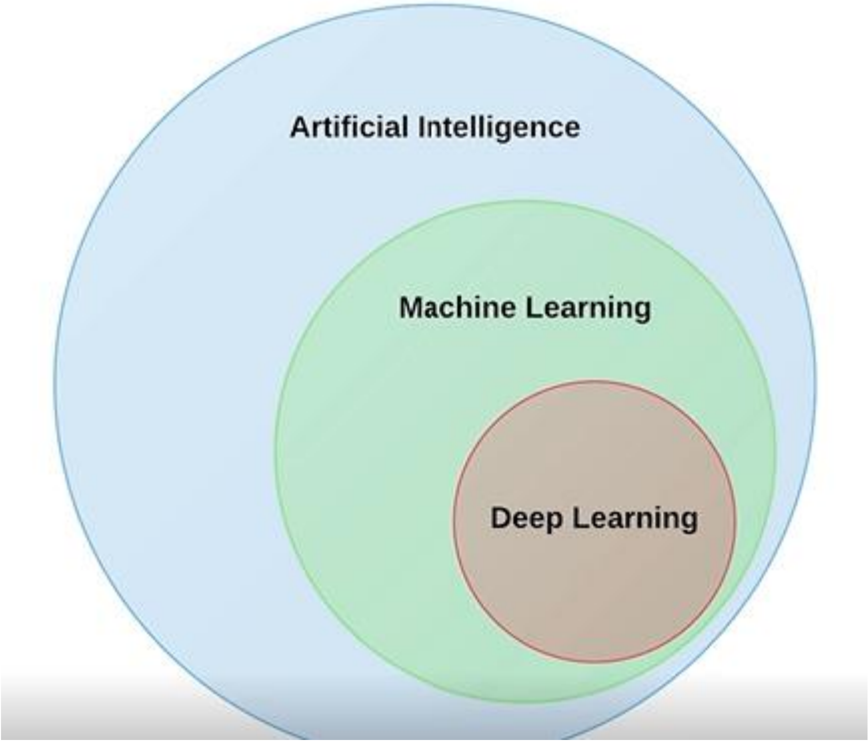
ToBig's 9기 임소정

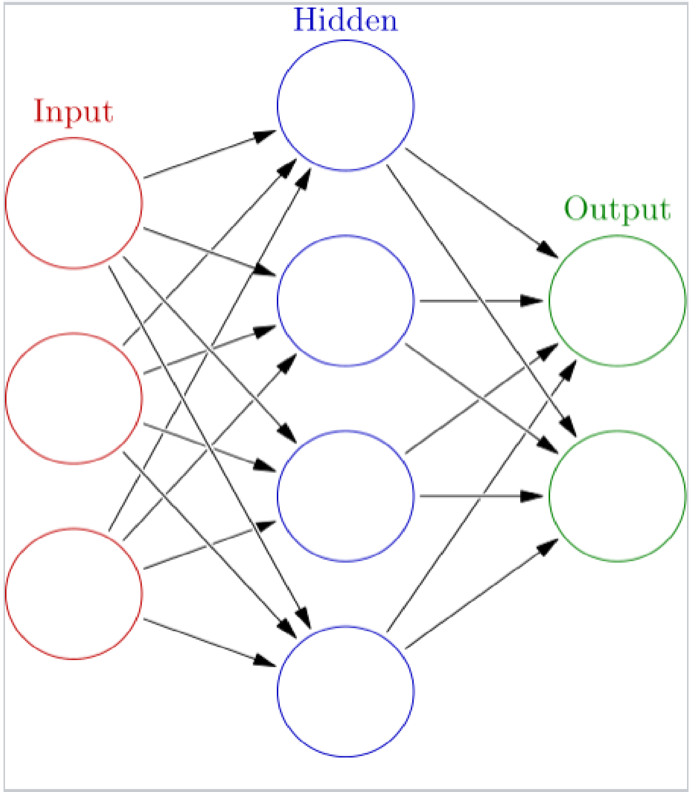
Deep Learning

Neural Network - 1

Contents

Unit 01 | 딥러닝이란?

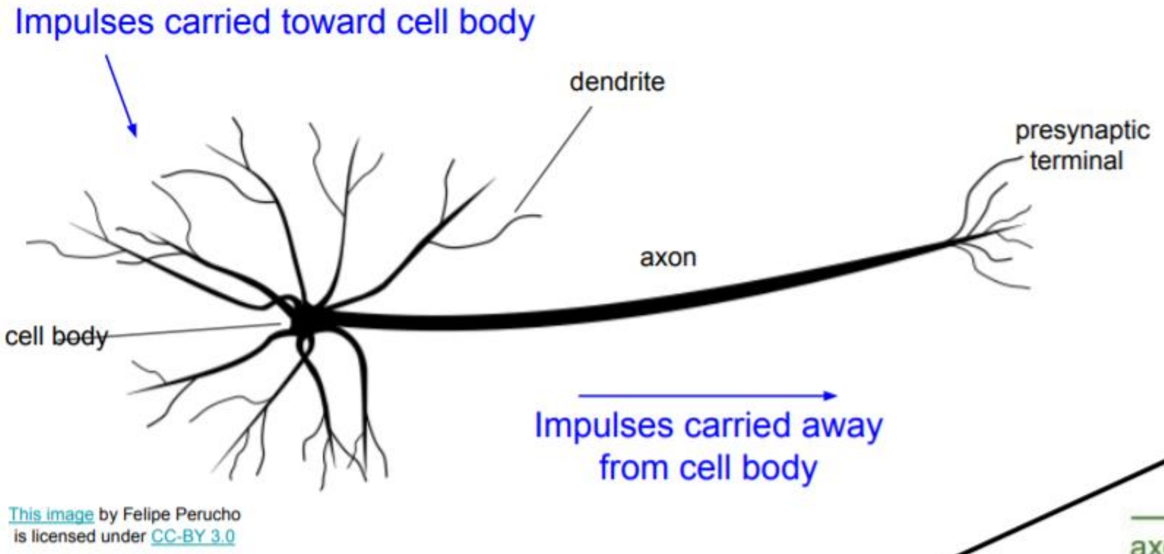


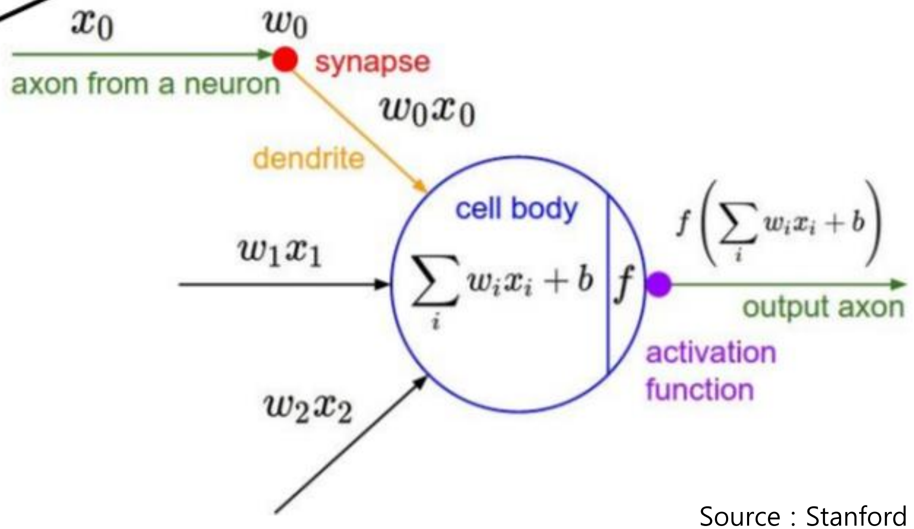


Artificial Neural Network

Source : https://www.youtube.com/watch?v=aF03asAmQbY&feature=youtu.be
https://en.wikipedia.org/wiki/Artificial_neural_network

Unit 01 | 딥러닝이란?





Source : Stanford cs231n Lecture 4

Unit 01 | 딥러닝이란?

Be very careful with your brain analogies!

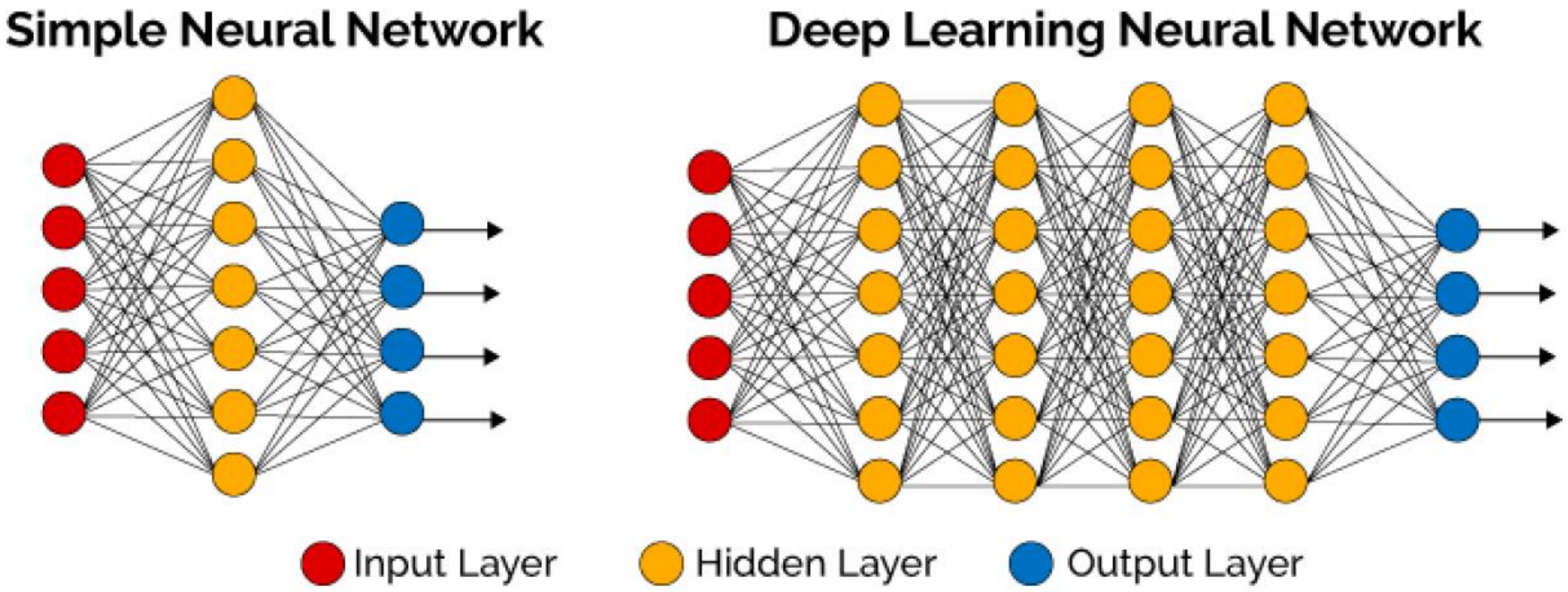
Biological Neurons:

- Many different types
- Dendrites can perform complex non-linear computations
- Synapses are not a single weight but a complex non-linear dynamical system
- Rate code may not be adequate

[Dendritic Computation. London and Hausser]

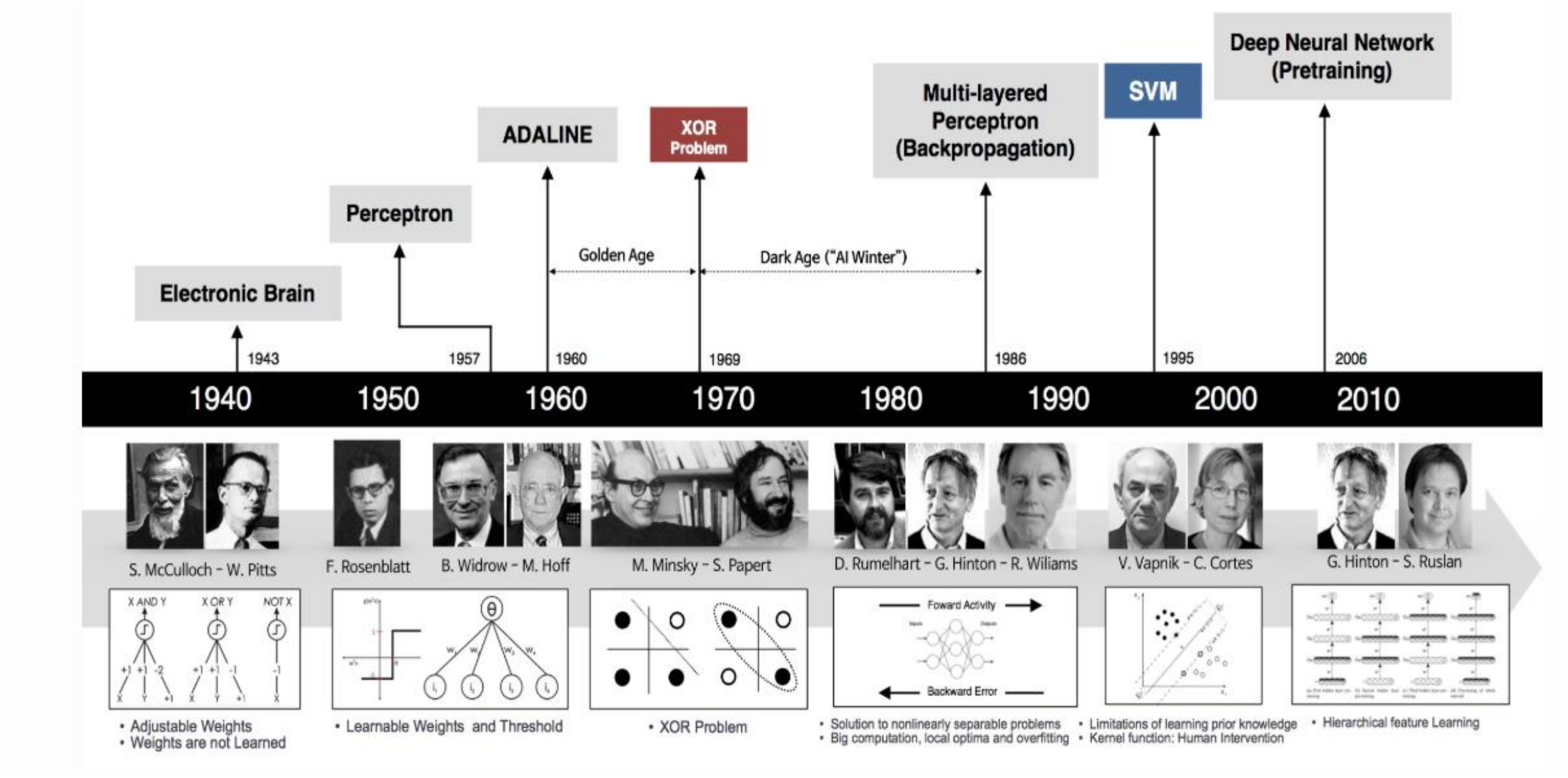
Source : Stanford cs231n Lecture 4

Unit 01 | 딥러닝이란?



Source : https://beamandrew.github.io/deeplearning/2017/02/23/deep_learning_101_part1.html

Unit 01 | 딥러닝이란?



Source : https://beamandrew.github.io/deeplearning/2017/02/23/deep_learning_101_part1.html

Why is Deep Learning Hot Now?

1. Big Data

- Larger Datasets
- Easier Collection & Storage

2. Hardware

- Graphics Processing Units (GPUs)
- Massively Parallelizable

3. Software

- Improved Techniques
- New Models
- Toolboxes

Source : MIT 6.S191: Introduction to Deep Learning

Unit 01 | 딥러닝이란?

Deep Learning Success: Vision

Image Recognition

Source : MIT 6.S191: Introduction to Deep Learning

Unit 01 | 딥러닝이란?

Deep Learning Success: Vision

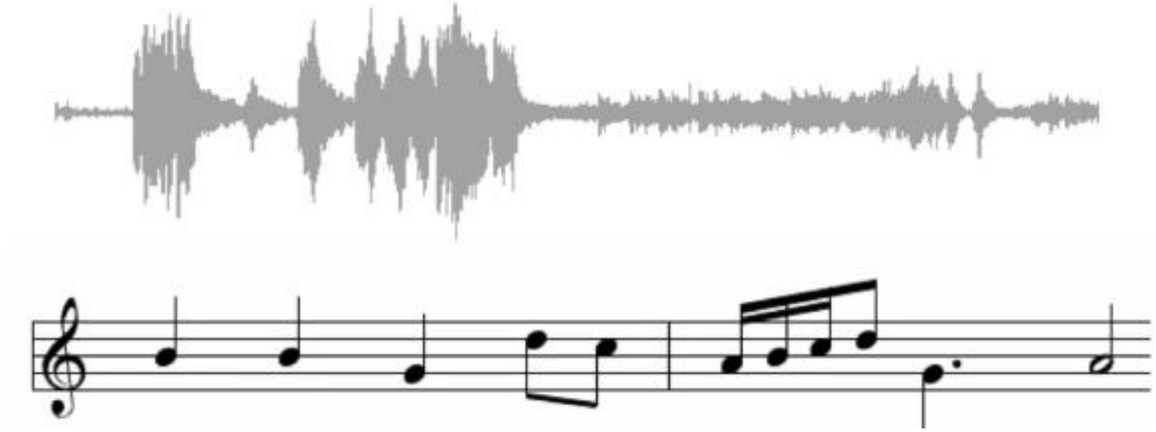
Detect pneumothorax in real X-Ray scans

Source : MIT 6.S191: Introduction to Deep Learning

Unit 01 | 딥러닝이란?

Deep Learning Success: Audio

Music Generation



Source : MIT 6.S191: Introduction to Deep Learning

Unit 01 | 딥러닝이란?

Deep Learning Success

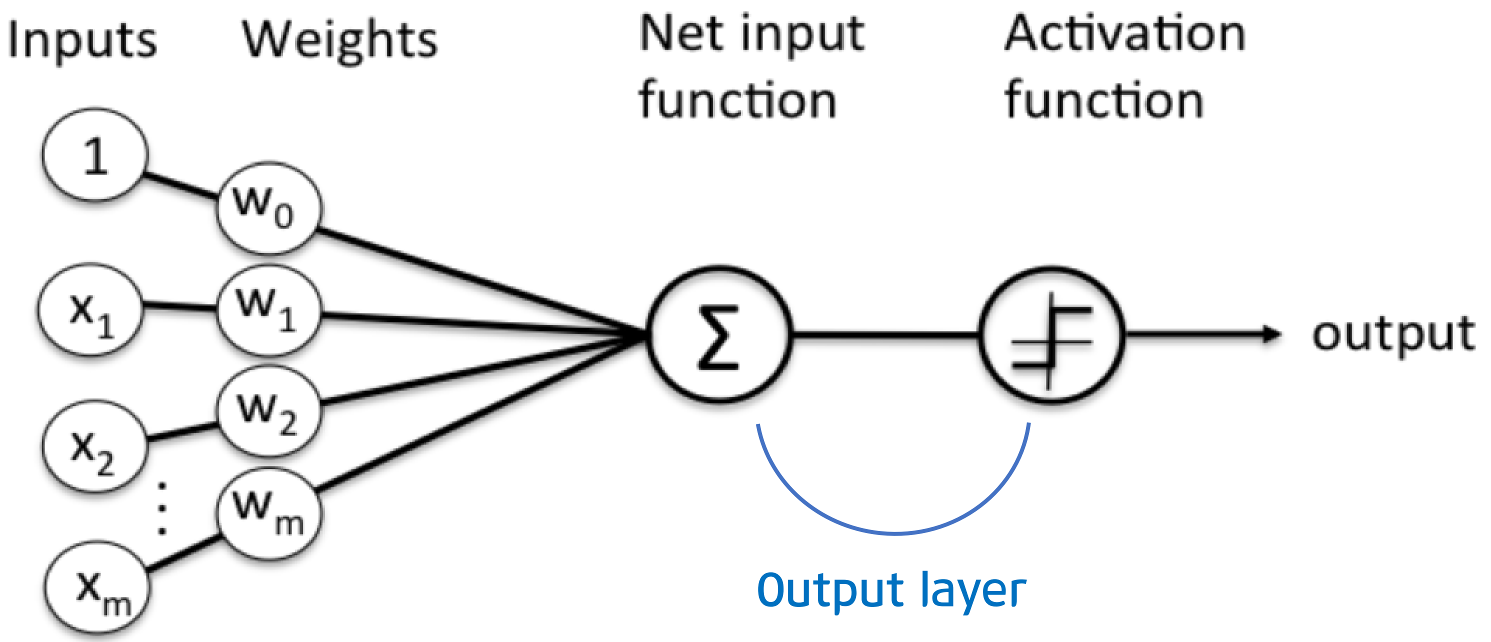
And so many more…

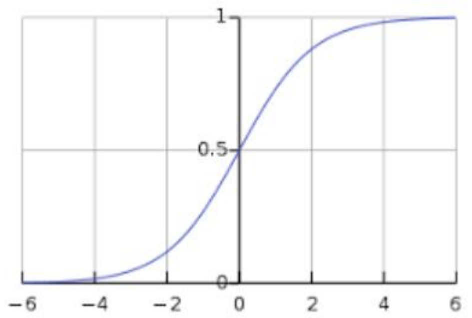
Source : MIT 6.S191: Introduction to Deep Learning

Unit 02 | Perceptron

Perceptron (단층신경망)

- 신경망의 기원이 되는 단층 신경망 알고리즘
- 선형 분류를 수행할 수 있는 피드포워드 뉴럴 네트워크





Activation Function

Output layer

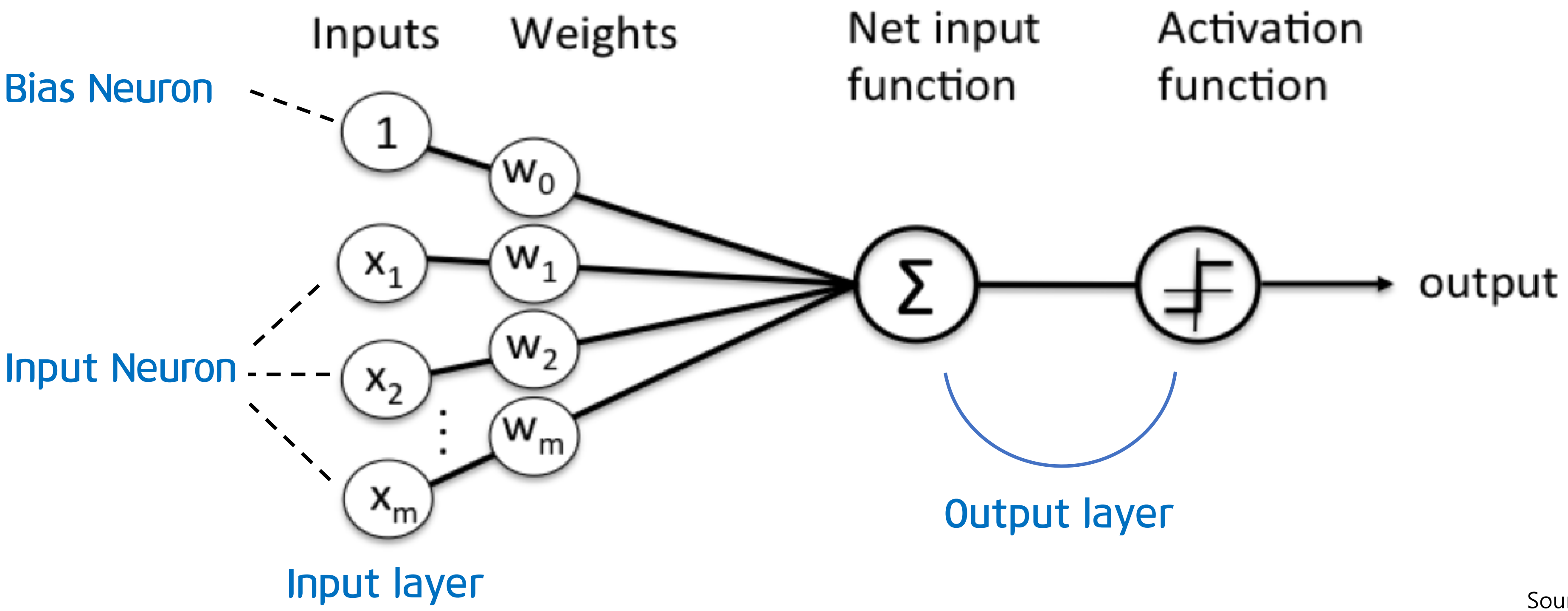
Input layer

Source : http://solarisailab.com/archives/1206
https://github.com/awantik/data-science/wiki/Artificial-Neural-Networks

Unit 02 | Perceptron

Perceptron (단층신경망)

- 신경망의 기원이 되는 단층 신경망 알고리즘
- 선형 분류를 수행할 수 있는 피드포워드 뉴럴 네트워크



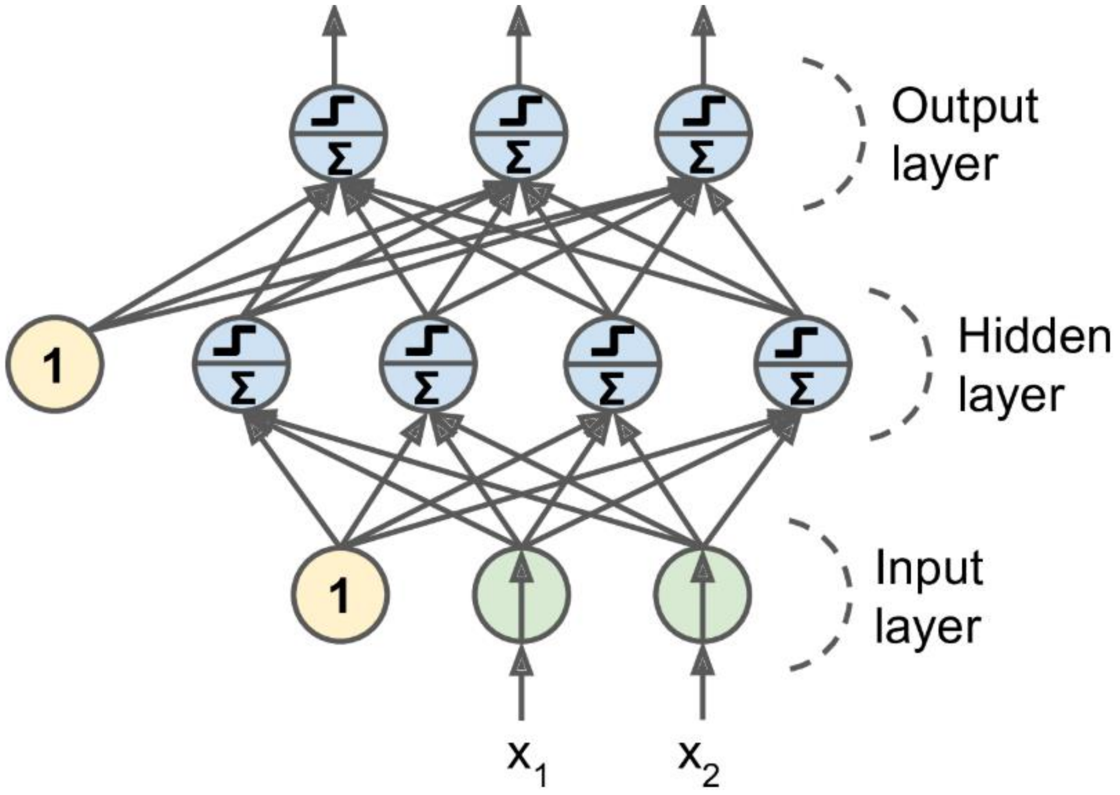
Input Neuron : m
Output Neuron : 1
Bias Neuron : 1
Weights : m+1

Source : http://solarisailab.com/archives/1206

Unit 03 | Multi Layer Perceptron

Multi Layer Perceptron

Input layer, Output layer 외에도 한 개 이상의 hidden layer 로 구성된 신경망



Unit 03 | Multi Layer Perceptron

Softmax Classifier (Multinomial Logistic Regression)

scores = unnormalized log probabilities of the classes.

$$\boxed{P(Y=k|X=x_i)=\frac{e^{s_k}}{\sum_j e^{s_j}}} \quad \text{where} \quad \boxed{s=f(x_i;W)}$$

→ Softmax function

cat	**3.2**
car	5.1
frog	-1.7

Source : Stanford cs231n Lecture 3

Softmax Classifier (Multinomial Logistic Regression)

scores = unnormalized log probabilities of the classes.

$$\boxed{P(Y=k|X=x_i)=\frac{e^{s_k}}{\sum_j e^{s_j}}}$$ where $$s=f(x_i;W)$$

Softmax function

cat	**3.2**
car	5.1
frog	-1.7

Want to maximize the log likelihood, or (for a loss function)
to minimize the negative log likelihood of the correct class:

$$\boxed{L_i=-\log P(Y=y_i|X=x_i)=-\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)}$$

Loss function

Source : Stanford cs231n Lecture 3

Softmax Classifier (Multinomial Logistic Regression)

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

unnormalized probabilities

	unnormalized log probabilities	exp →	unnormalized probabilities	normalize →	probabilities
cat	**3.2**		**24.5**		**0.13**
car	5.1		164.0		0.87
frog	-1.7		0.18		0.00

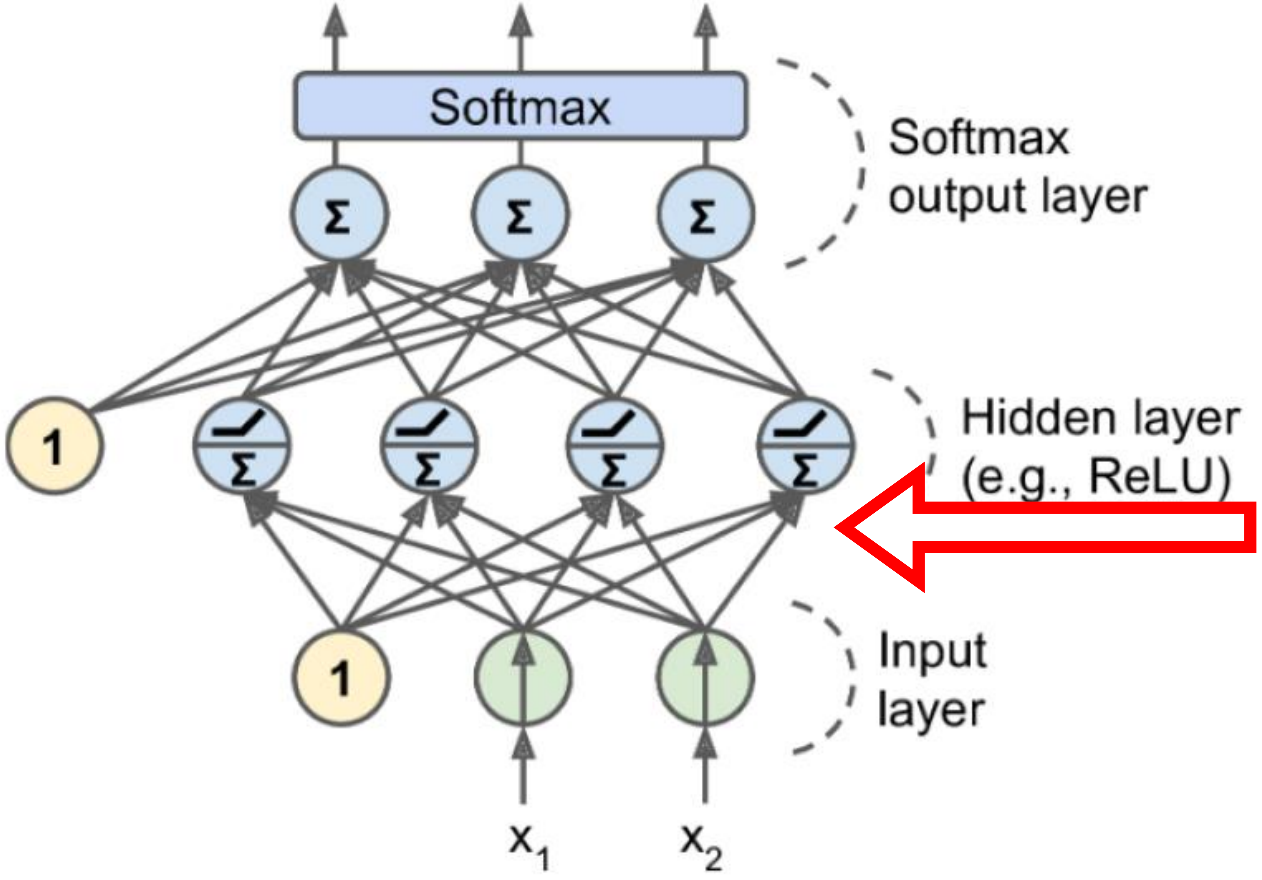
→ L_i = -log(0.13)
= 0.89

unnormalized log probabilities

probabilities

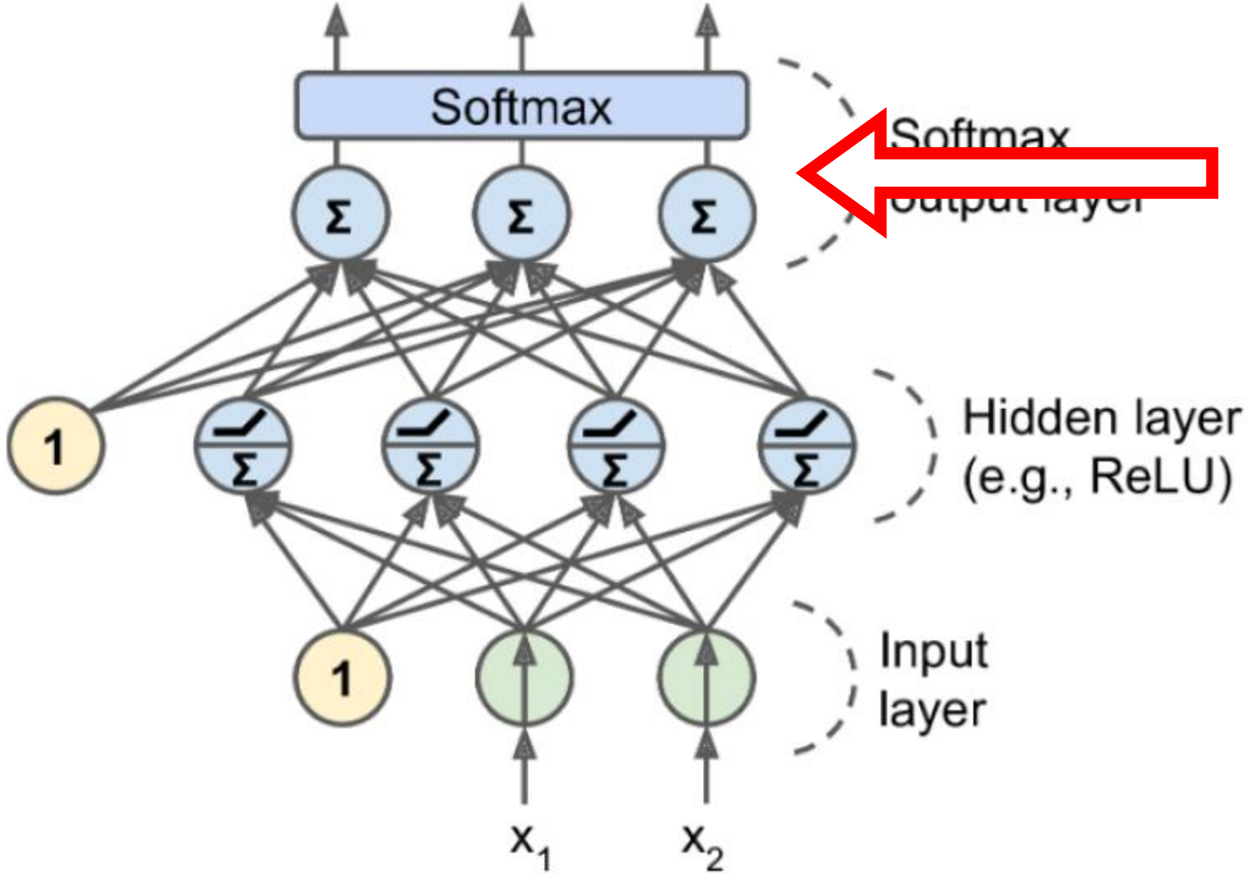
Source : Stanford cs231n Lecture 3

Unit 03 | Multi Layer Perceptron



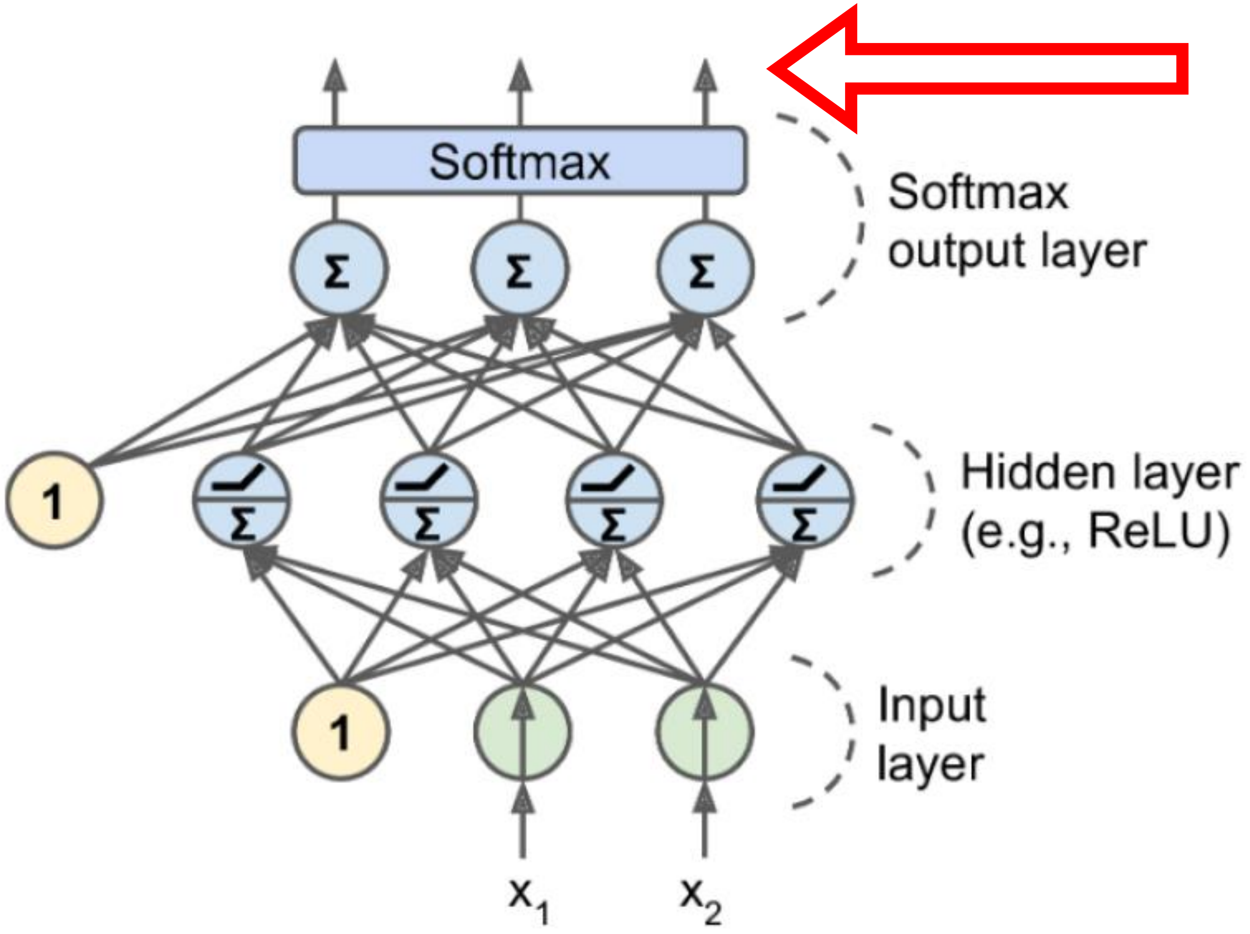

```
W1 = tf.Variable(tf.random_normal([2, 4]))
b1 = tf.Variable(tf.random_normal([4]))
Layer1 = tf.nn.relu(tf.matmul(X, W1) + b1)
```

Unit 03 | Multi Layer Perceptron



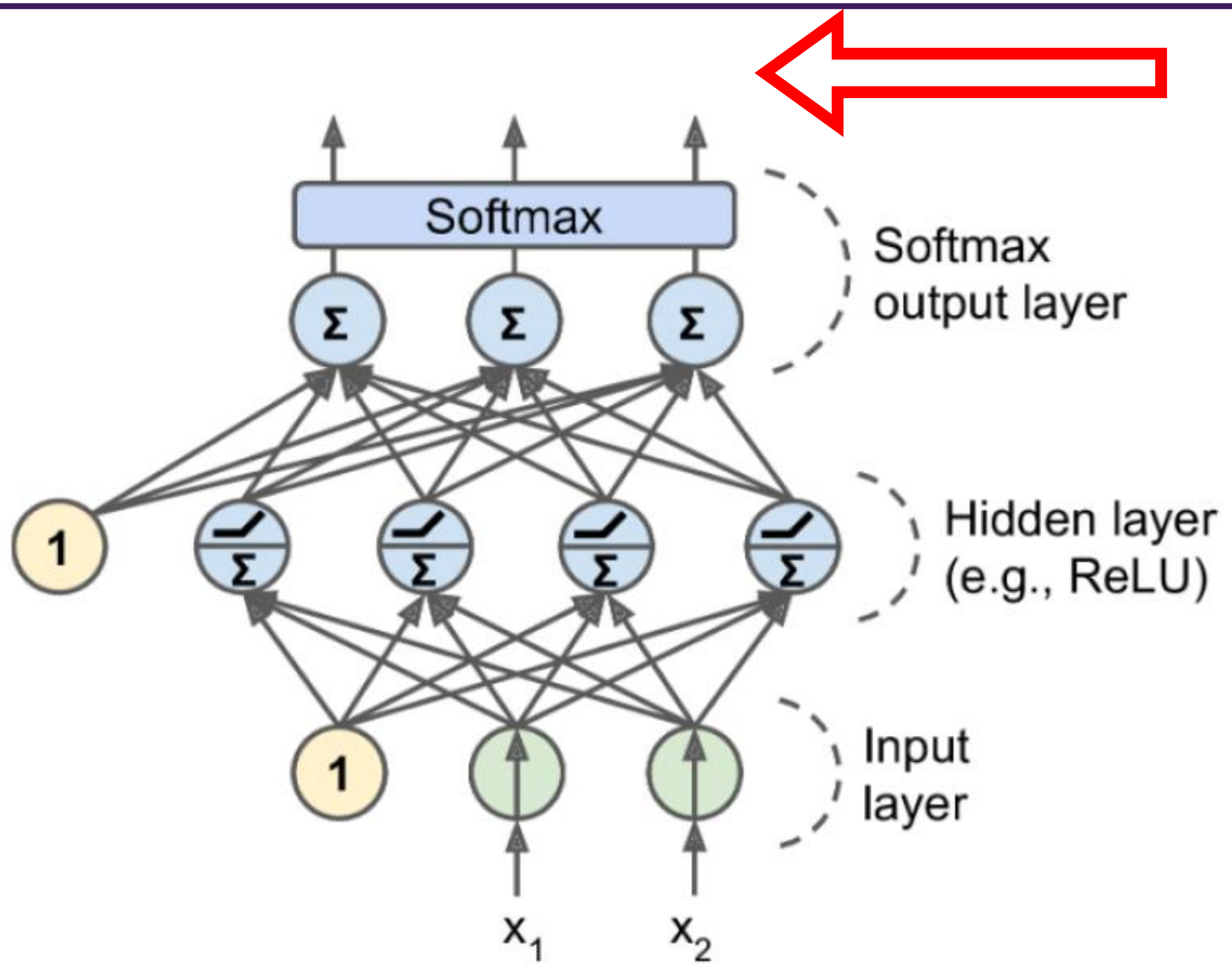
Score (e.g. cat, car, frog)

Unit 03 | Multi Layer Perceptron



Prob. (e.g. cat, car, frog)

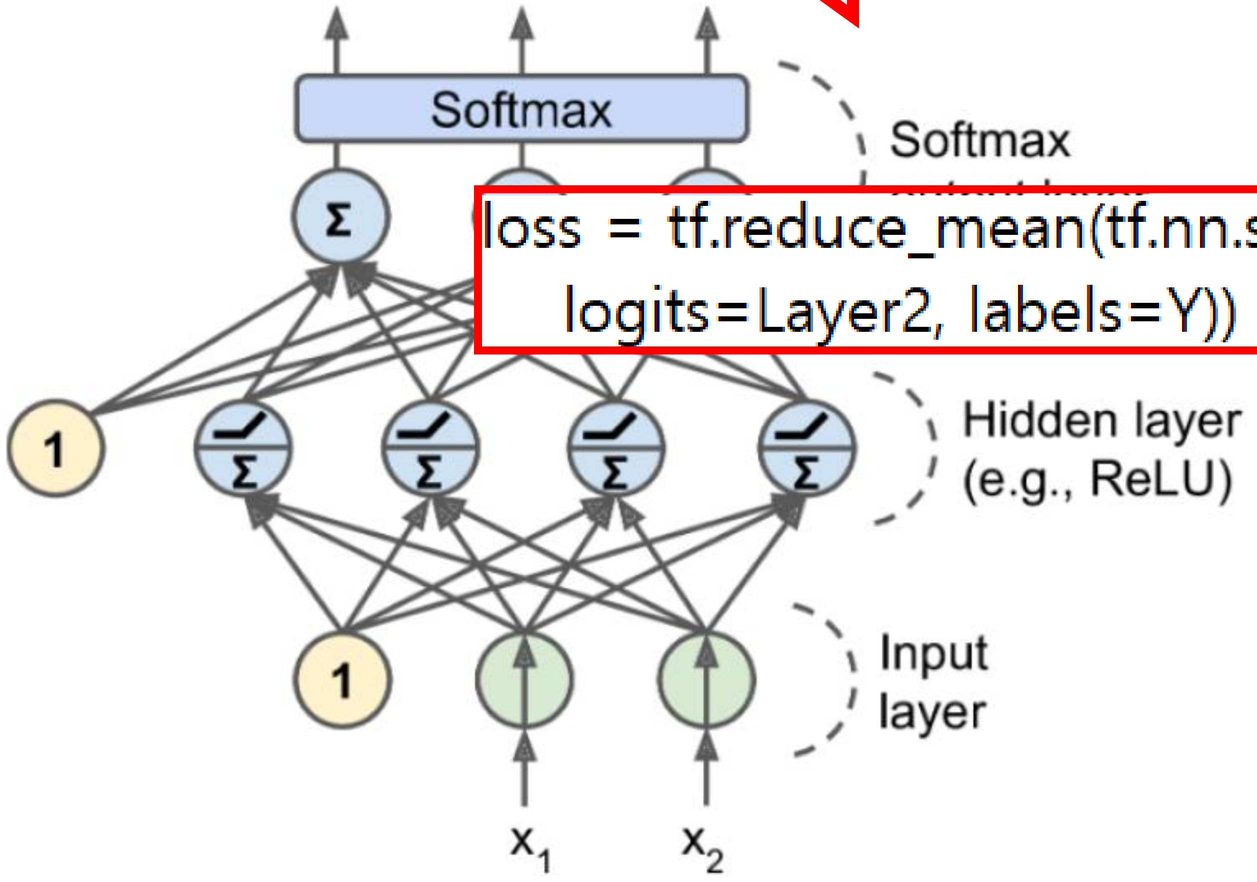
Unit 03 | Multi Layer Perceptron



Loss (per example)
+ called 'cross-entropy loss'

Unit 03 | Multi Layer Perceptron

Loss (per example)
+ called 'cross-entropy loss'

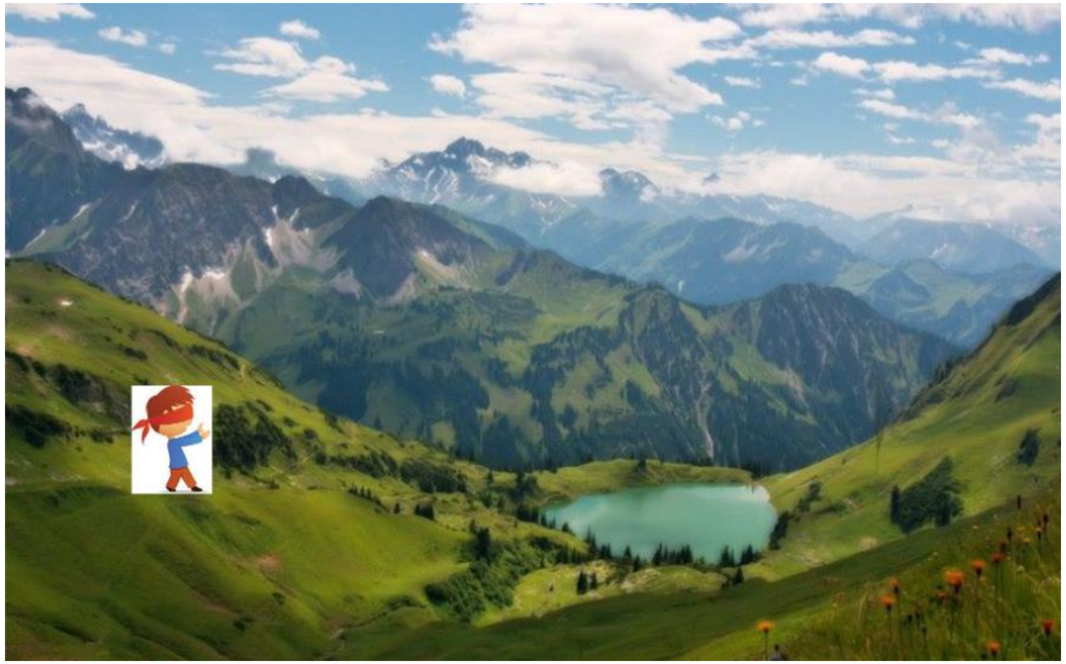



```
loss = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(
    logits=Layer2, labels=Y))
```

Unit 04 | Backpropagation

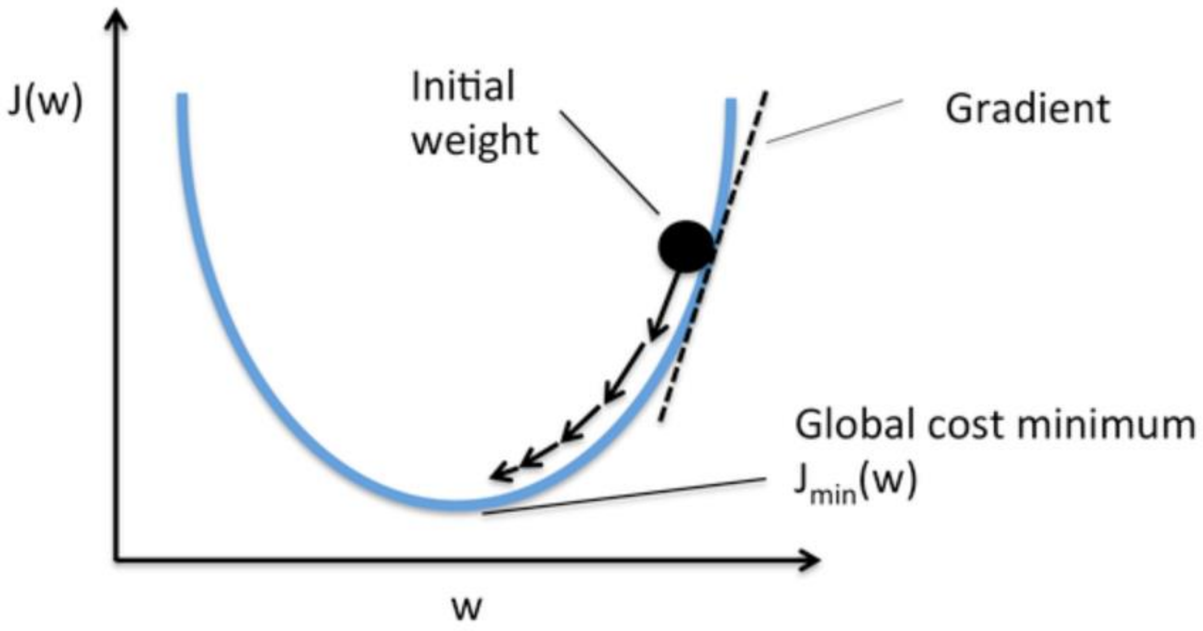
그럼 어떻게 MLP, NN를 잘 학습시킬까???
-> Back Propagation & Gradient Descent

Gradient Descent : loss function을 최소화 하기 위해 사용하는 알고리즘



Gradient Descent : loss function을 최소화 하기 위해 사용하는 알고리즘

1) Weight에 random 한 숫자로 초기값 부여
2) Loss를 minimize 하는 방향으로 w를 업데이트 !

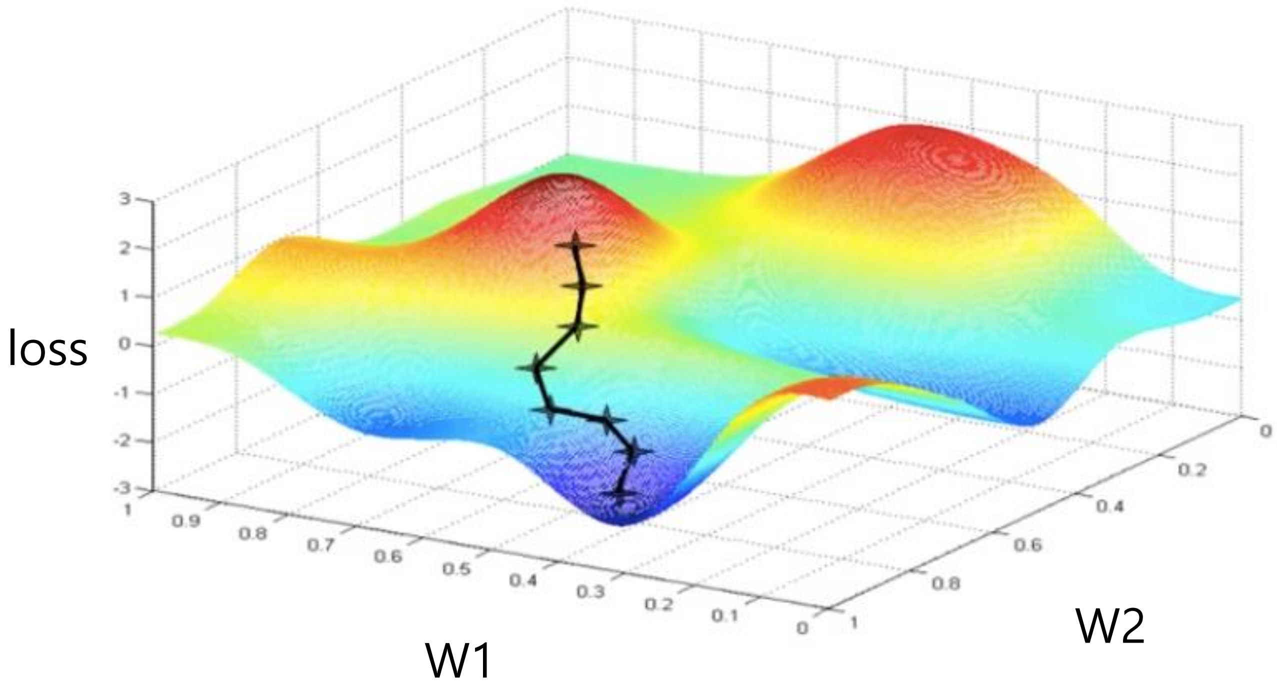


$$\boxed{W := W - \alpha \frac{d}{dw} loss(W)}$$

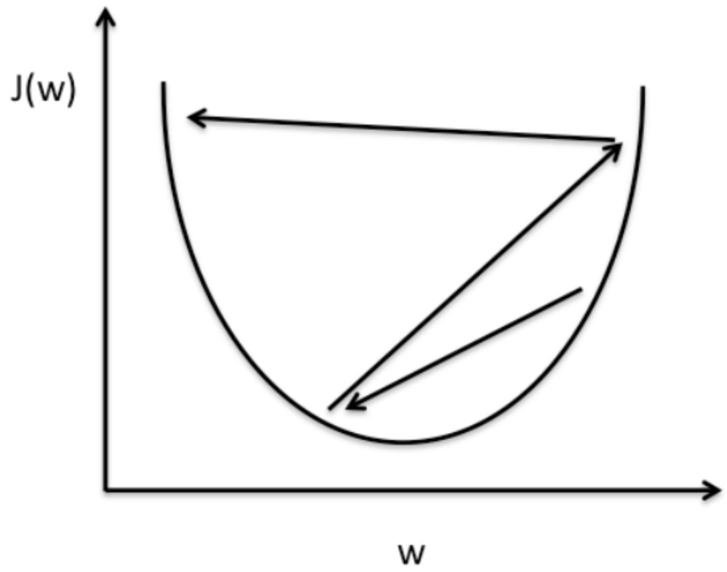
learning rate

Unit 04 | Backpropagation

Gradient Descent : loss function을 최소화 하기 위해 사용하는 알고리즘

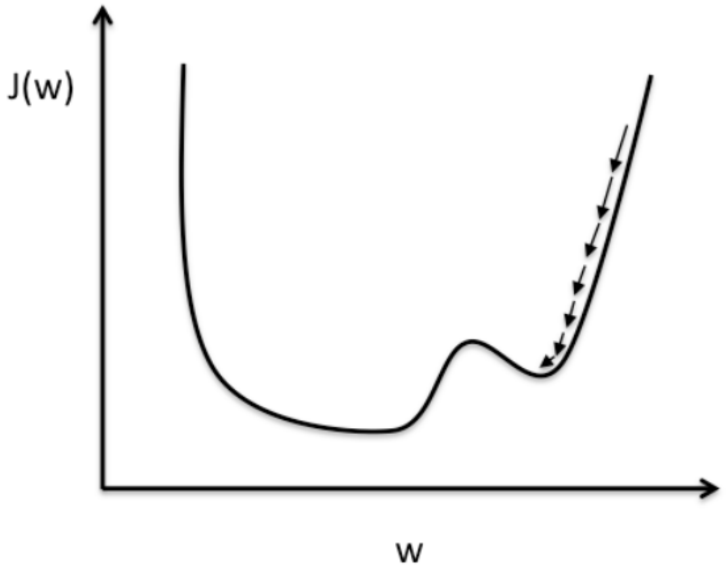


Learning rate (α)



Learning rate 가 큰 경우

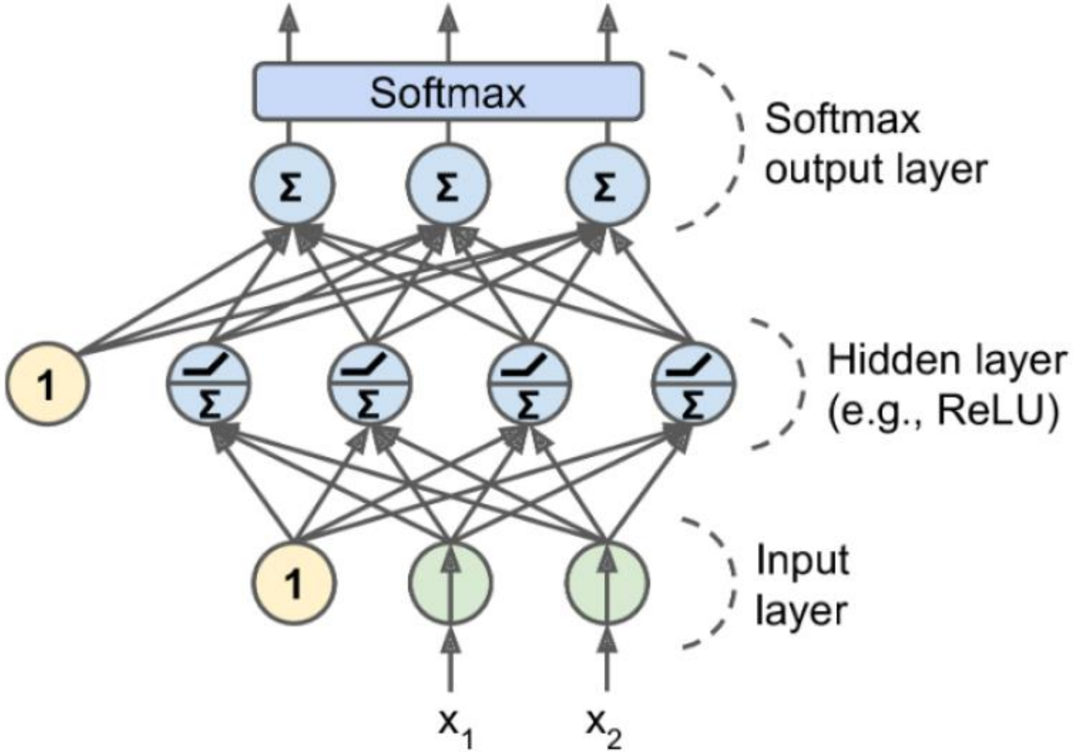
-> overshooting



Learning rate 가 작은 경우

-> 학습하는데 시간이 오래 걸린다
최저점이 아닌 데서 멈추거나,
local minimum에 빠진다.

Unit 04 | Backpropagation



이때 y를 이용하여 loss를 구합니다.

우리는 weights를 update해야합니다.

즉 loss를 최소화하는 방향(gradient)으로

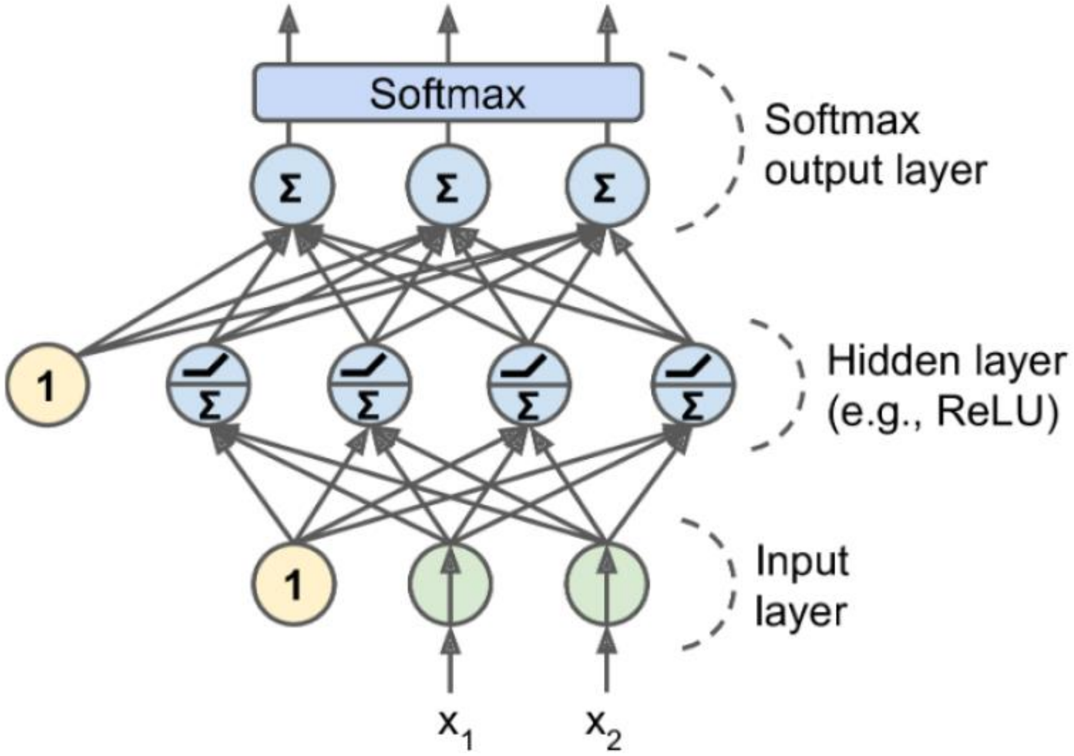
Weights를 update 시켜줘야 합니다.

(-> Gradient Descent, 미분값필요)

하지만 loss는 y에 대한 함수입니다. -log(softmax(y))

즉, loss를 Weights로 직접적인 미분이 불가능하죠!

Unit 04 | Backpropagation



loss는 y의 함수이고, y는 weights의 함수이다.
결국 합성함수
합성 함수의 미분 -> Chain Rule!

The Chain Rule

$$f = f(g)\,;\quad g = g(x)$$

$$\frac{df}{dx} = \frac{df}{dg}\,\frac{dg}{dx}$$

결국 합성함수
합성함수의 미분 -> Chain Rule!

Unit 04 | Backpropagation

Backpropagation: a simple example

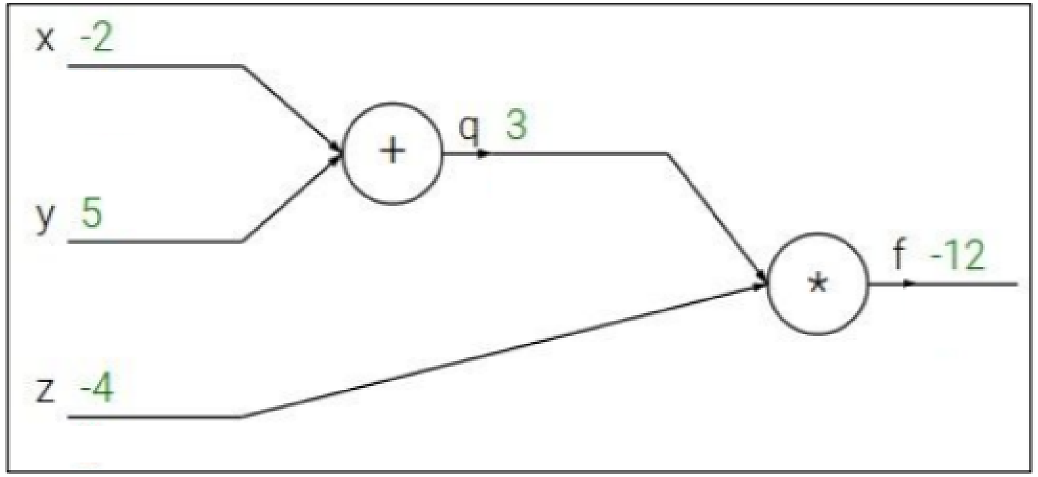
$$f(x,y,z) = (x+y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Source : Stanford cs231n Lecture 4

Unit 04 | Backpropagation

Backpropagation: a simple example

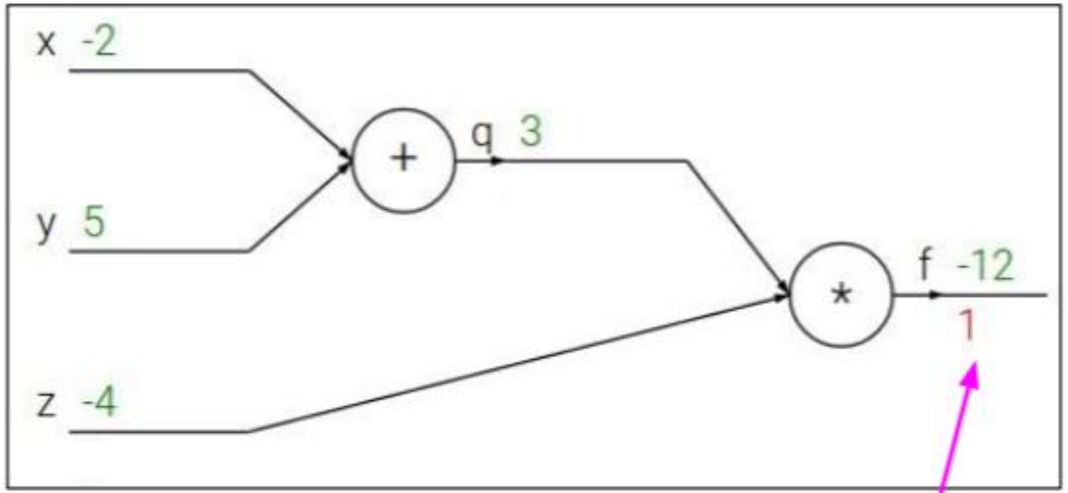
$$f(x,y,z) = (x+y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$$\frac{\partial f}{\partial f}$$

Unit 04 | Backpropagation

Backpropagation: a simple example

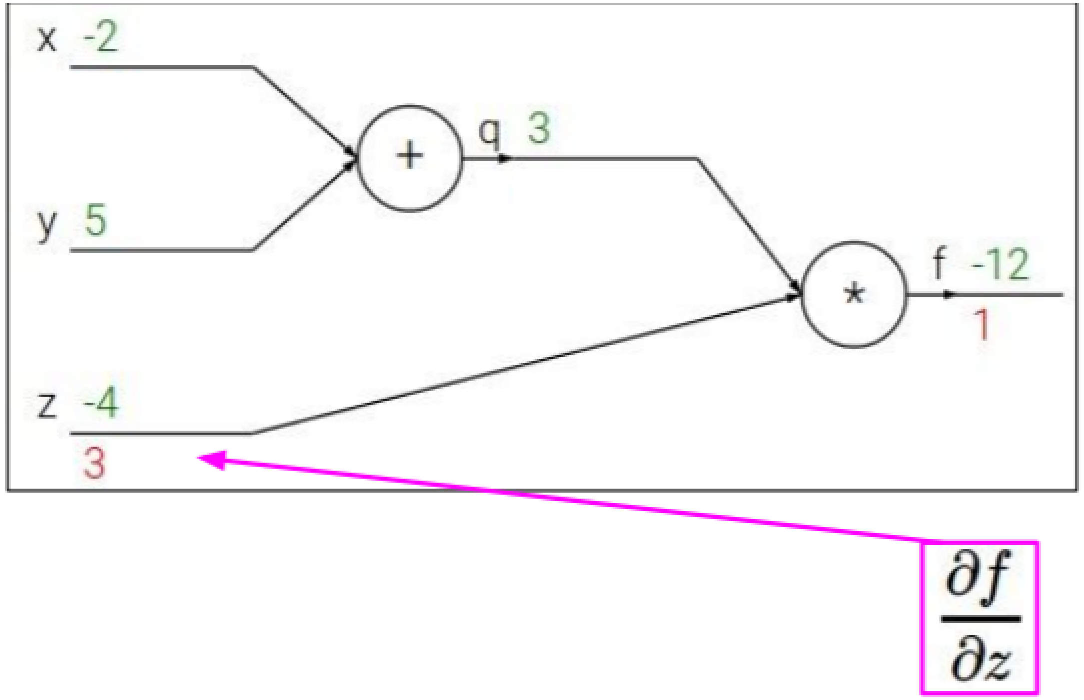
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Source : Stanford cs231n Lecture 4

Unit 04 | Backpropagation

Backpropagation: a simple example

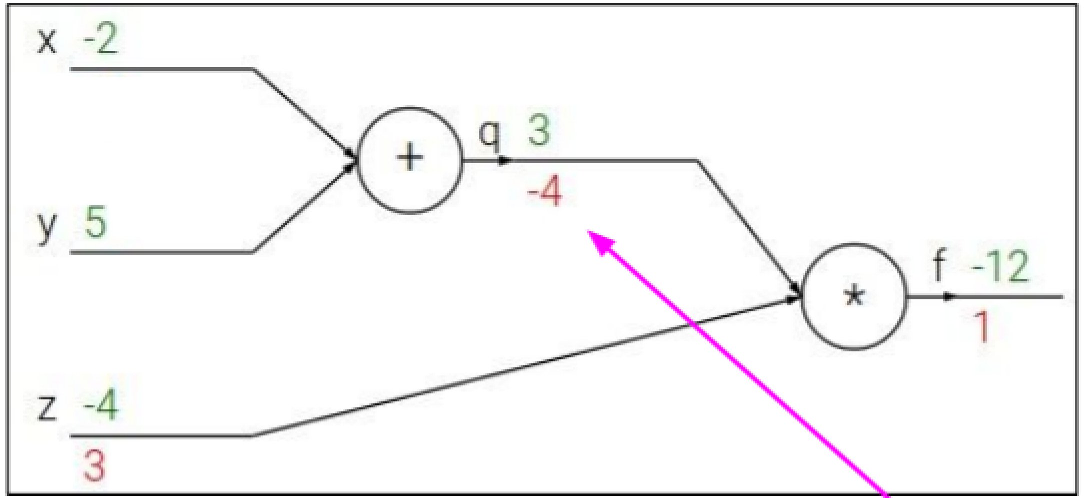
$$f(x,y,z) = (x+y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$$\frac{\partial f}{\partial q}$$

Unit 04 | Backpropagation

Backpropagation: a simple example

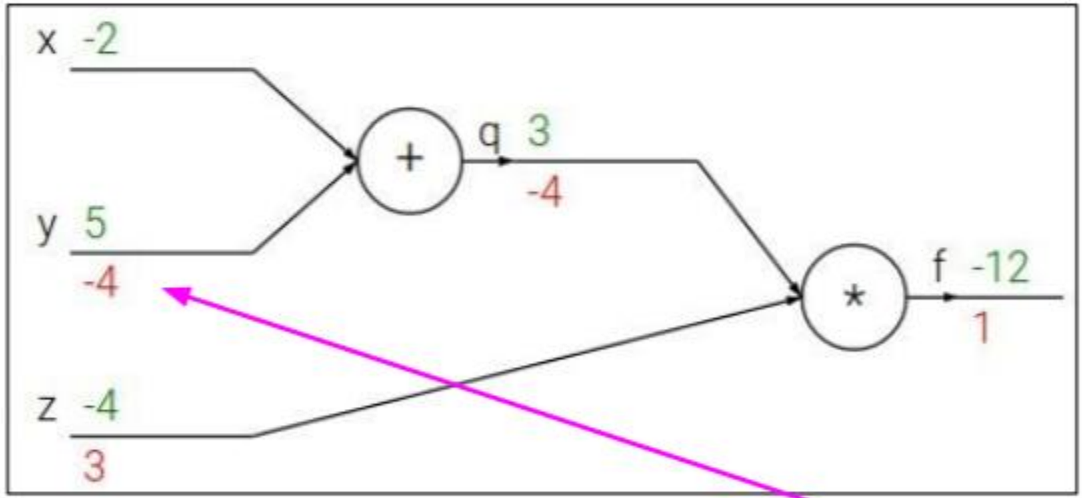
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$$\frac{\partial f}{\partial y}$$

Chain rule:

$$\boxed{\frac{\partial f}{\partial y} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial y}}$$

Source : Stanford cs231n Lecture 4

Unit 04 | Backpropagation

Backpropagation: a simple example

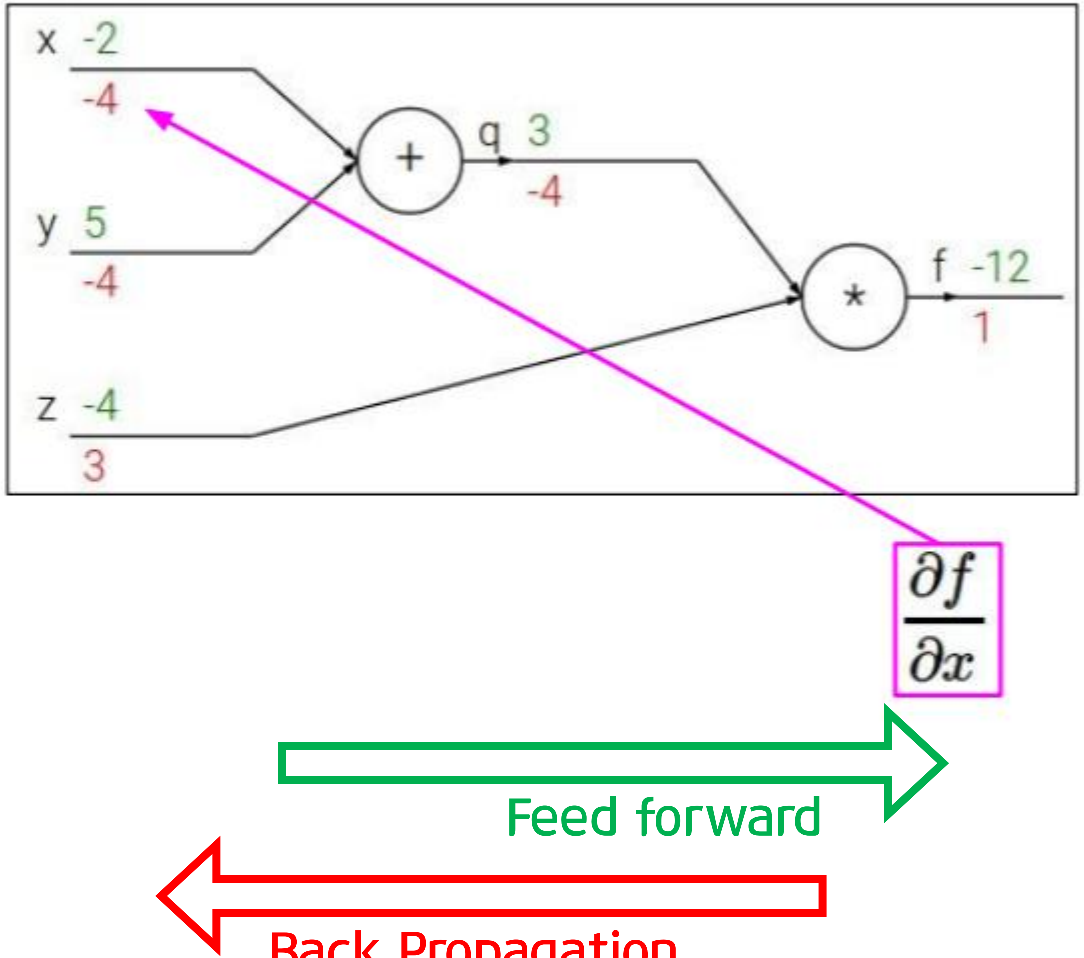
$$f(x,y,z) = (x+y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

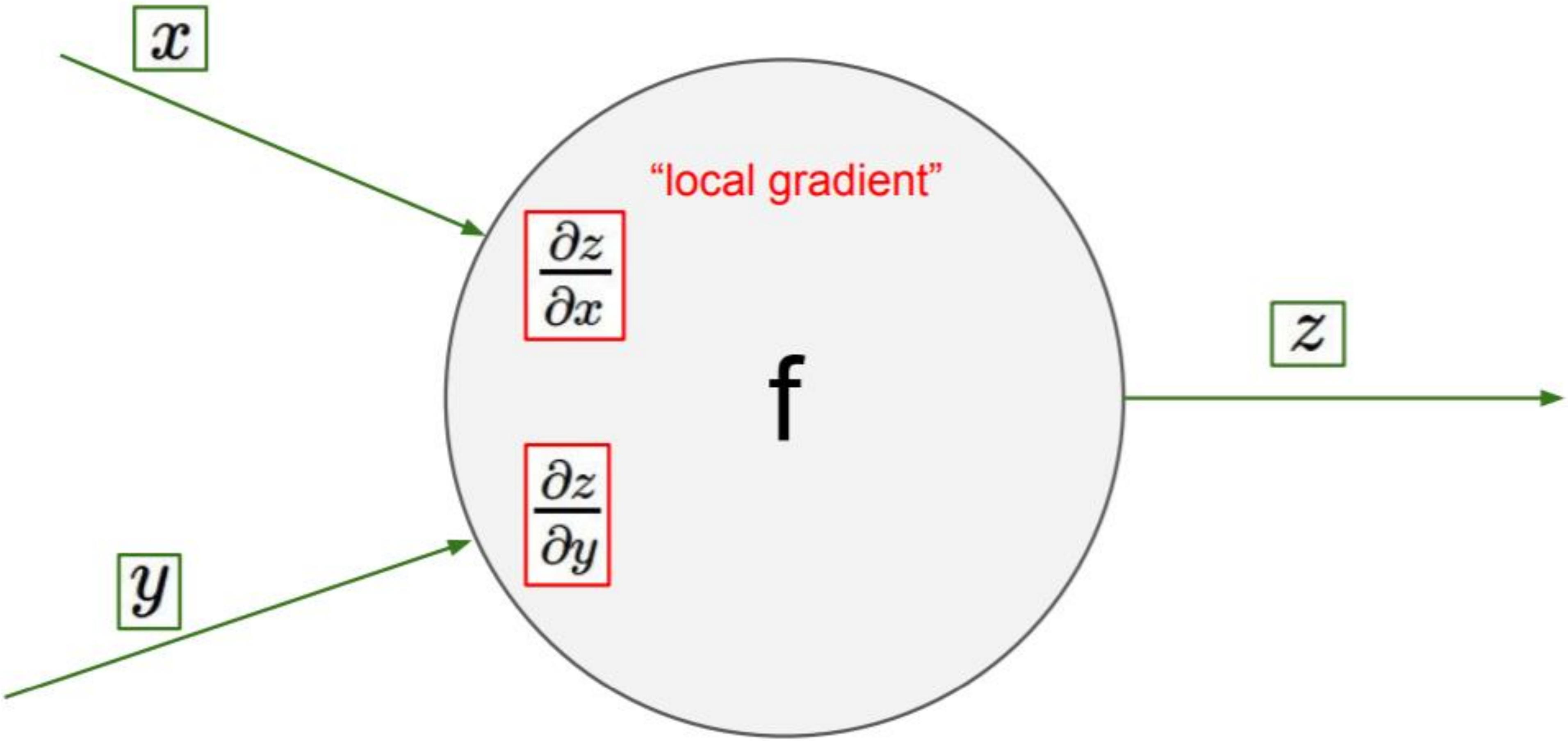
$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



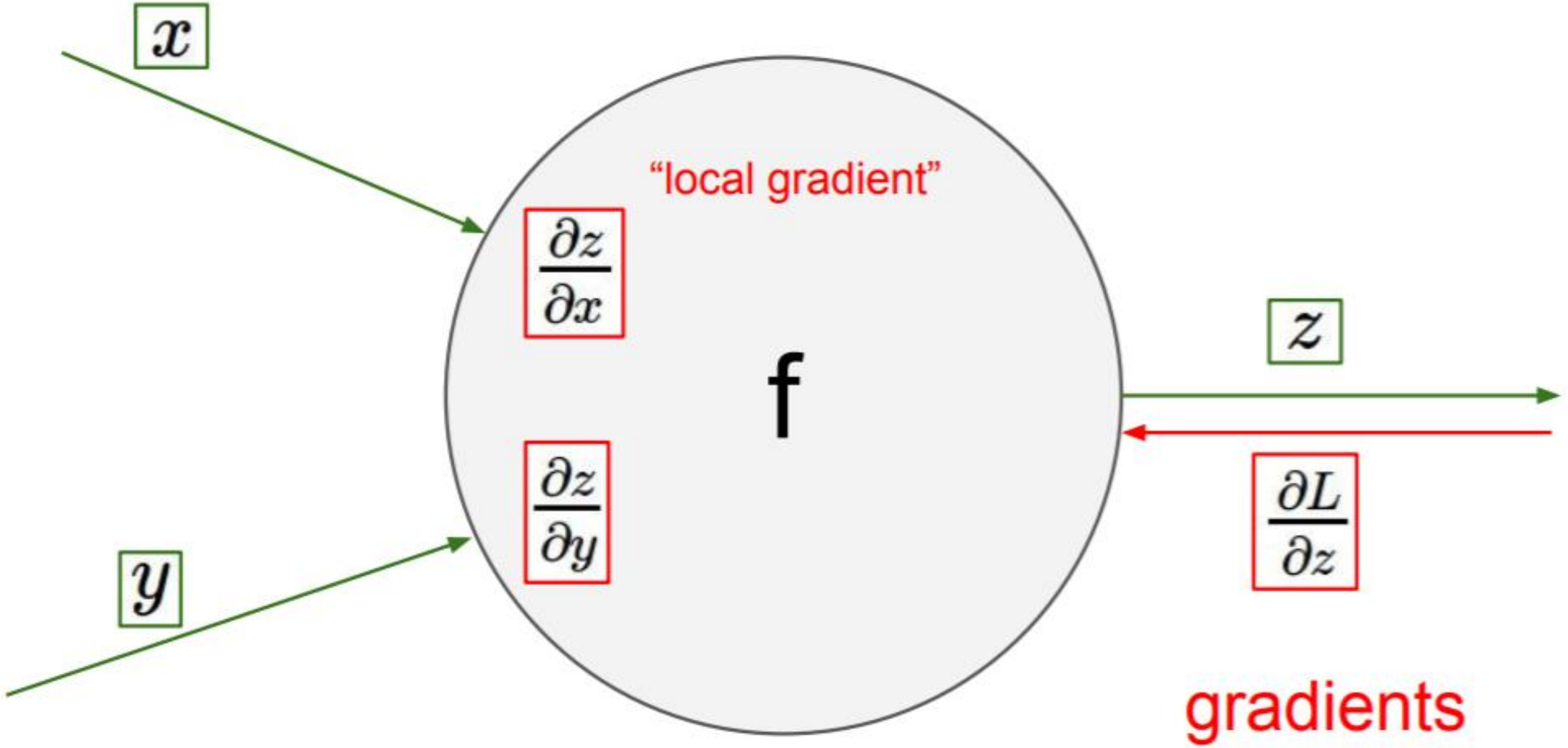
Source : Stanford cs231n Lecture 4

Unit 04 | Backpropagation



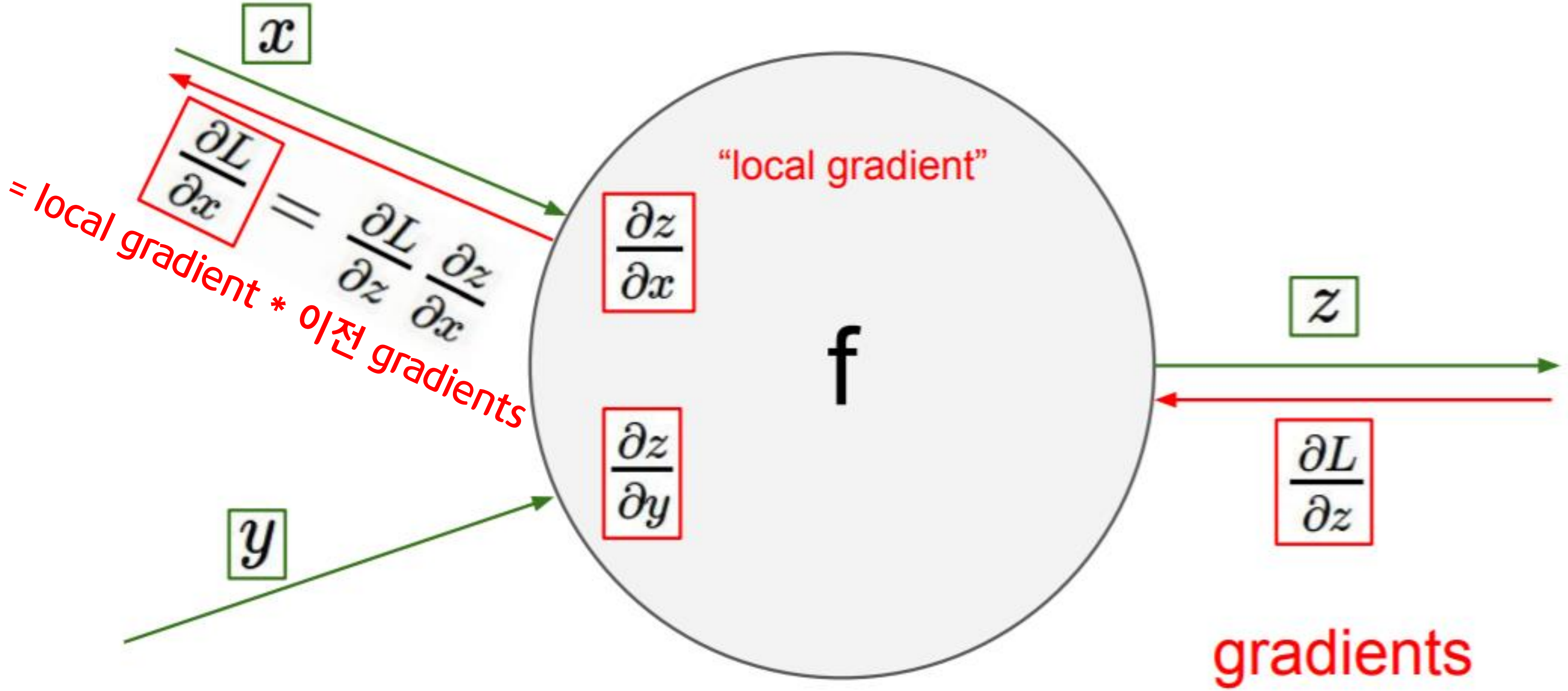
Source : Stanford cs231n Lecture 4

Unit 04 | Backpropagation



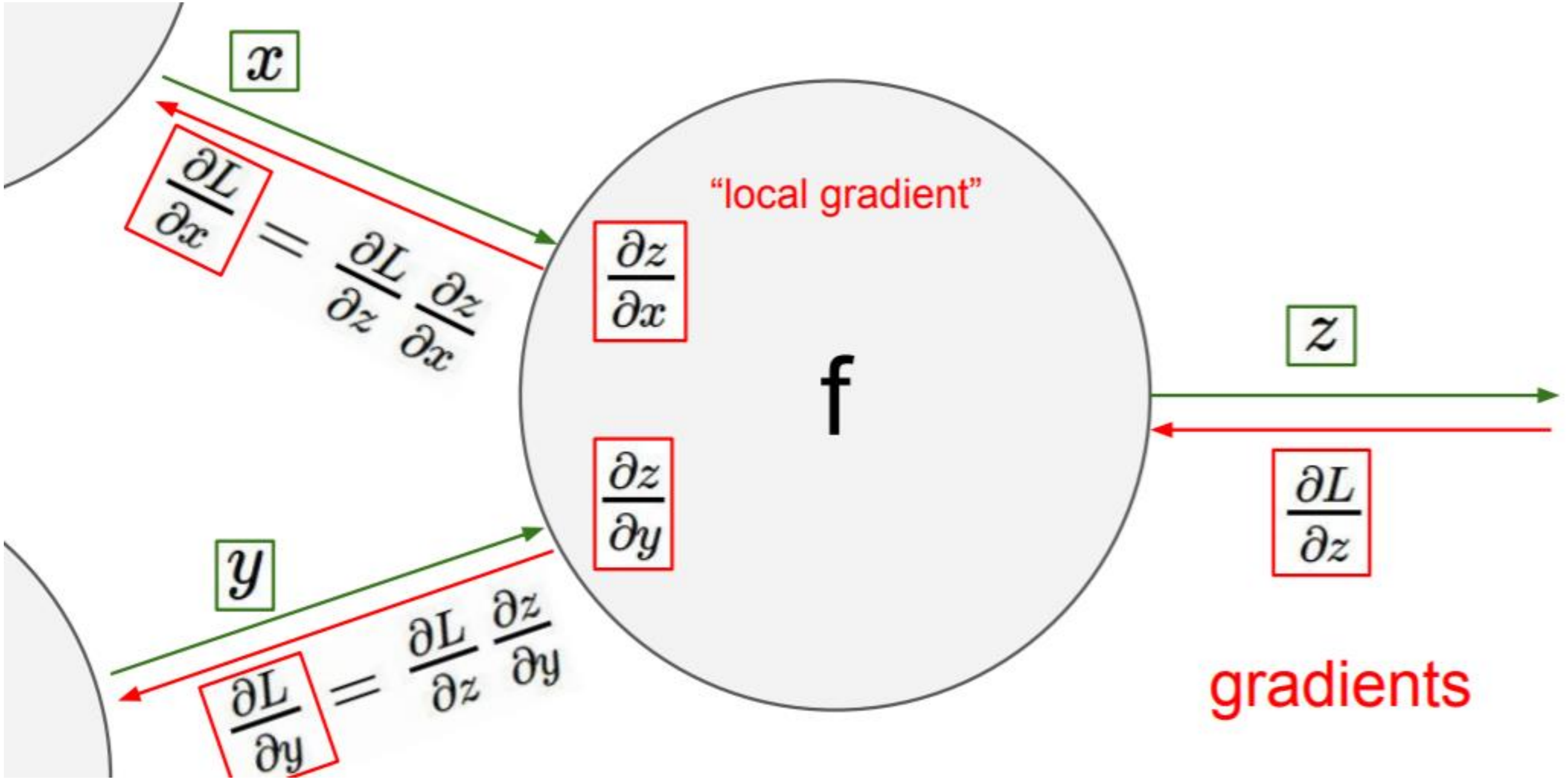
Source : Stanford cs231n Lecture 4

Unit 04 | Backpropagation



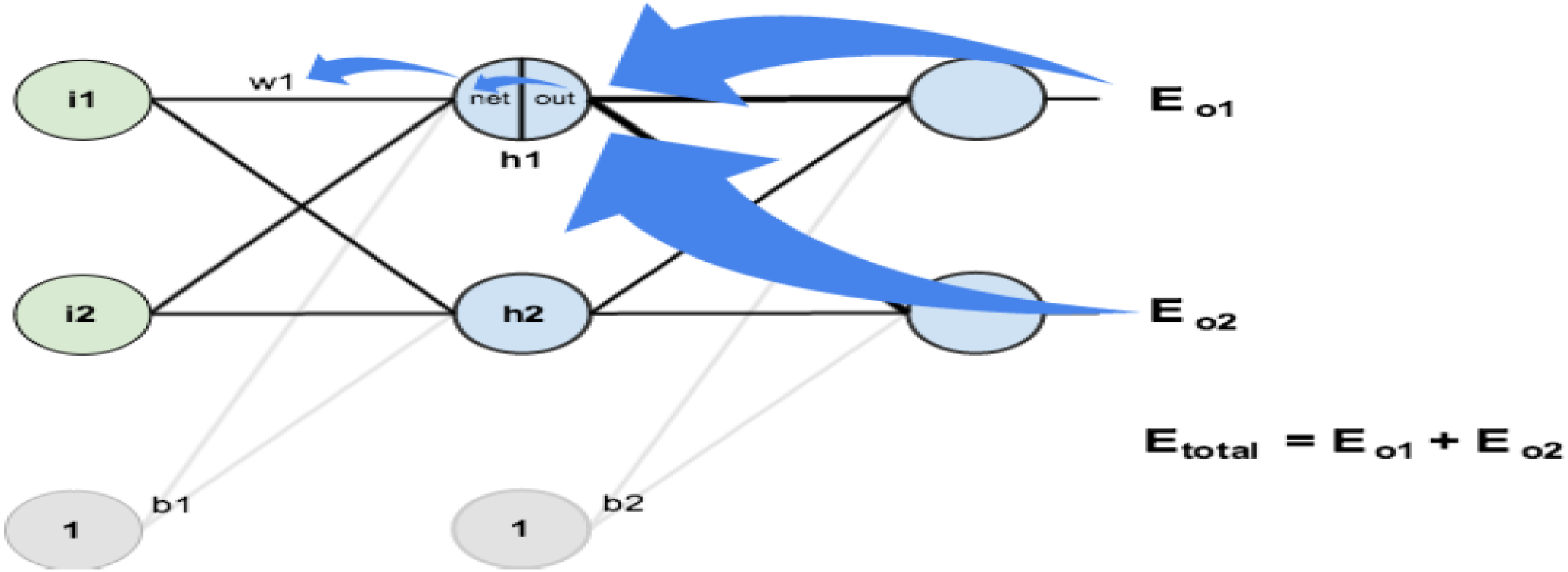
Source : Stanford cs231n Lecture 4

Unit 04 | Backpropagation



Source : Stanford cs231n Lecture 4

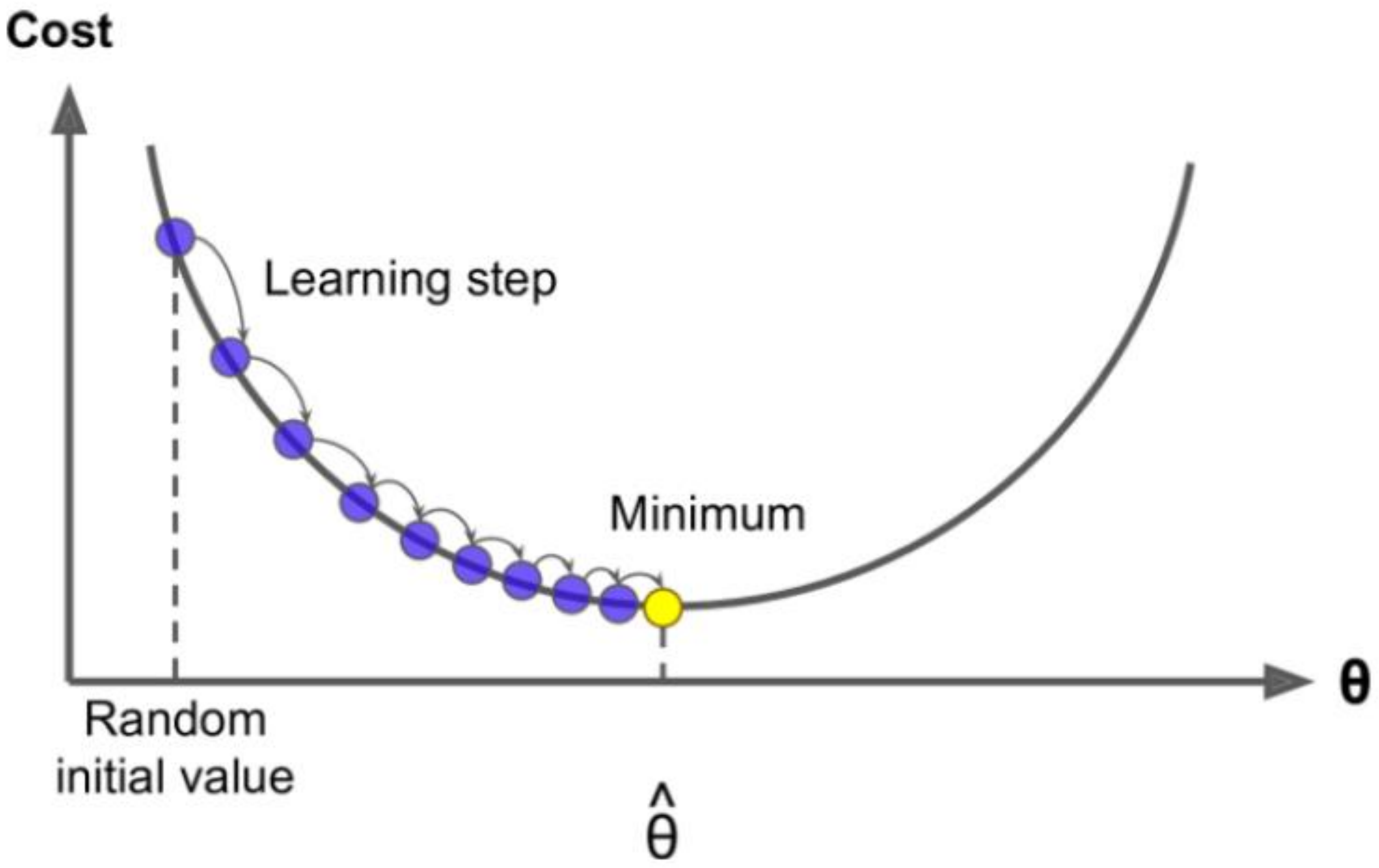
Unit 04 | Backpropagation



Loss와 멀리 떨어져있는 w1에 대한 미분 계산이 가능!

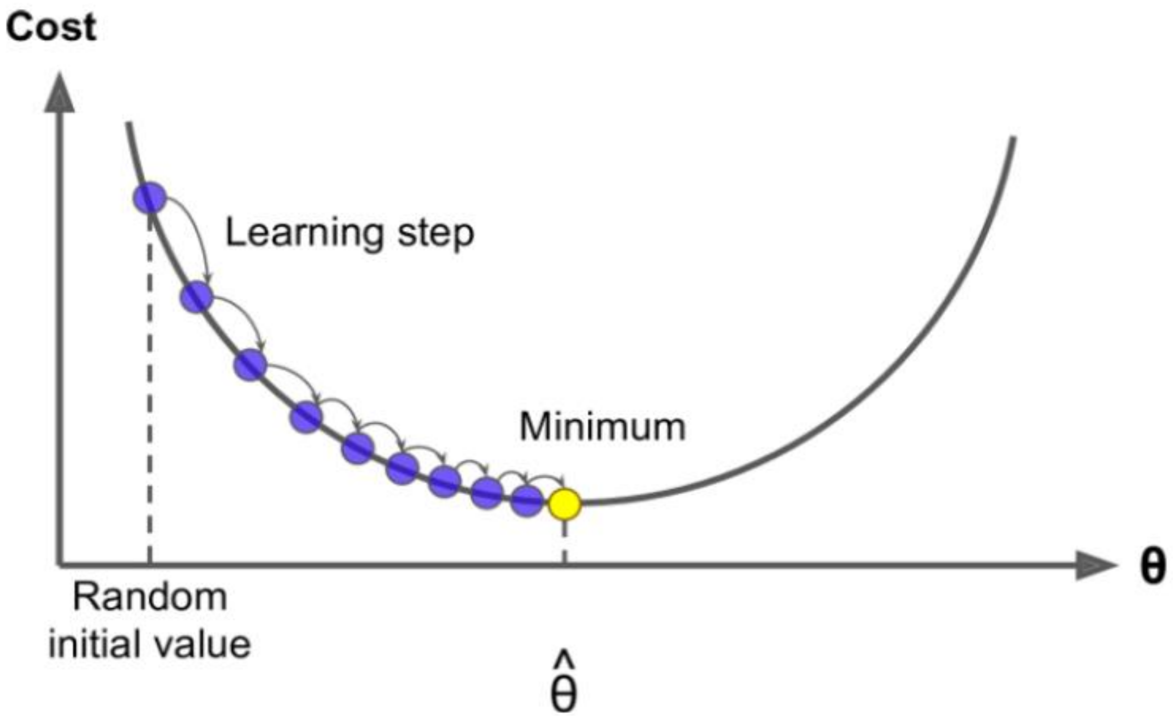
Source : https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/

Unit 04 | Backpropagation



이제 모든 weights에 Gradient Descent를 적용시켜서 Update!

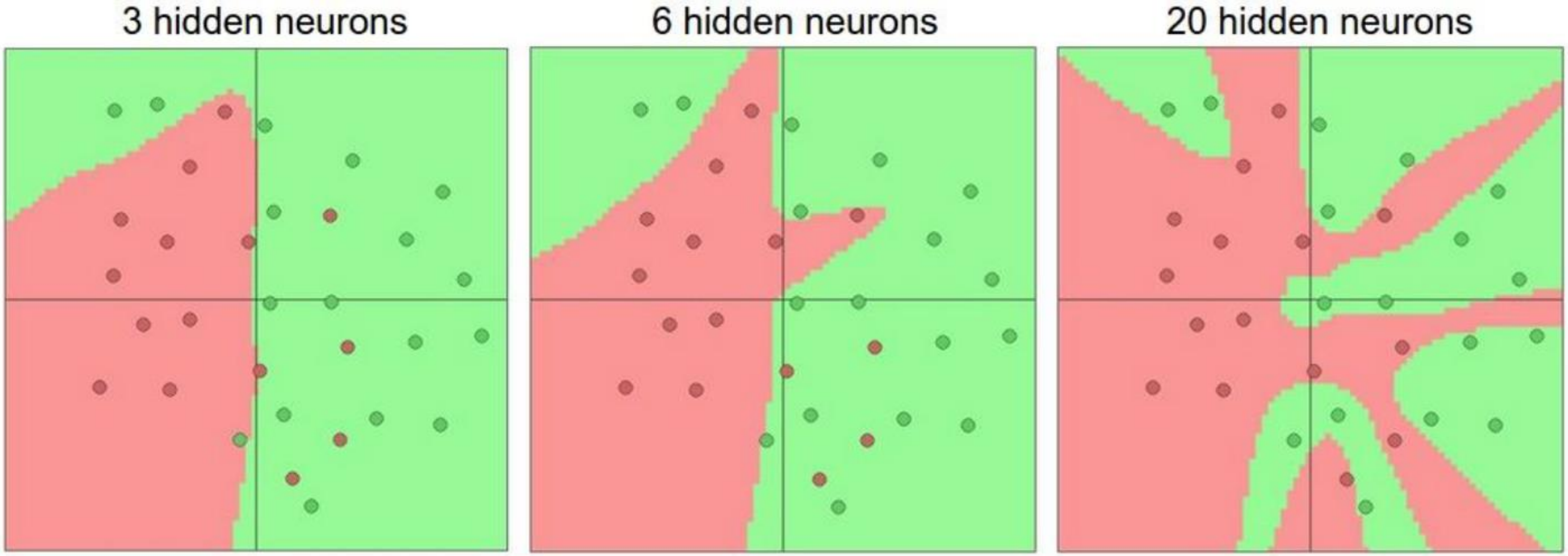
Unit 04 | Backpropagation



이제 모든 weights에 Gradient Descent를 적용시켜서 Update!

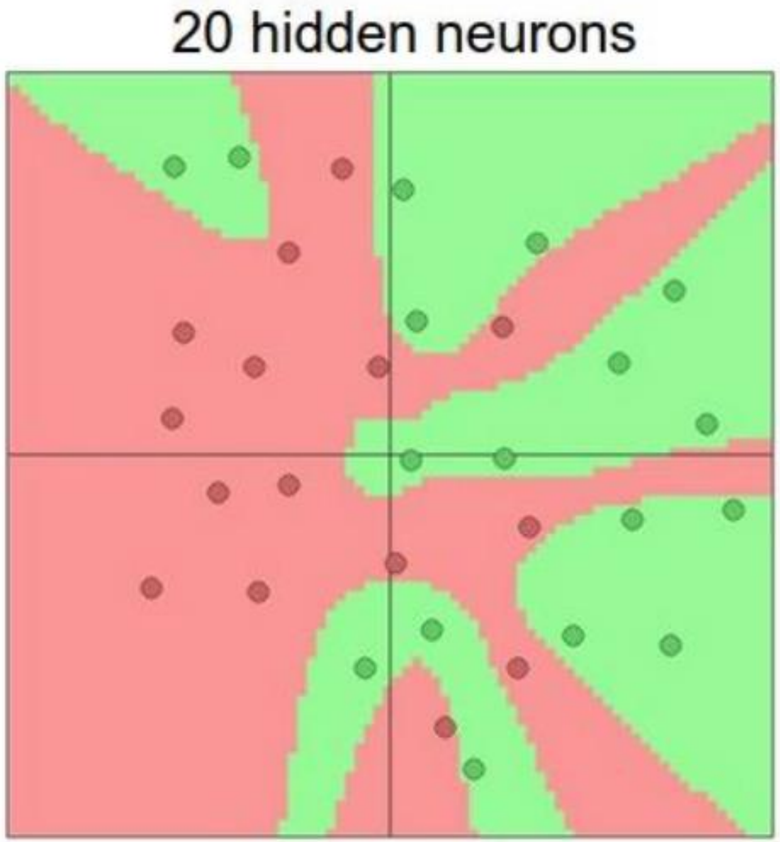
```
optimizer = tf.train.GradientDescentOptimizer(learning_rate=learning_rate)
train_op=optimizer.minimize(loss)
```

Unit 05 | Regularization



Source : Stanford cs231n Lecture 3

Unit 05 | Regularization



Overfit!!

Layer / Node줄여? No!!

Regularization!

Source : Stanford cs231n Lecture 3

Unit 05 | Regularization

Regularization : weight 클수록 패널티 부여하는 것

$$\underbrace{L(W) = \frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i)}_{}$$

Data loss: Model predictions
should match training data

Source : Stanford cs231n Lecture 3

Unit 05 | Regularization

Regularization : weight 클수록 패널티 부여하는 것

$$\underbrace{L(W) = \frac{1}{N}\sum_{i=1}^{N} L_i(f(x_i, W), y_i)}_{\text{Data loss: Model predictions should match training data}} + \underbrace{\lambda R(W)}_{\text{Regularization: Model should be "simple", so it works on test data}}$$

Data loss: Model predictions should match training data

Regularization: Model should be "simple", so it works on test data

* Regularization strength : 얼마나 테스트셋에 일반화 할 것인지
(0과1사이의 값)

Source : Stanford cs231n Lecture 3

Unit 05 | Regularization

In common use:

L2 regularization $R(W) = \sum_k \sum_l W_{k,l}^2$

L1 regularization $R(W) = \sum_k \sum_l |W_{k,l}|$

Elastic net (L1 + L2) $R(W) = \sum_k \sum_l \beta W_{k,l}^2 + |W_{k,l}|$

Max norm regularization (might see later)

Dropout (will see later)

Fancier: Batch normalization, stochastic depth

Source : Stanford cs231n Lecture 3

Unit 05 | Regularization

L2 Regularization (Weight Decay)

$$x = [1, 1, 1, 1]$$

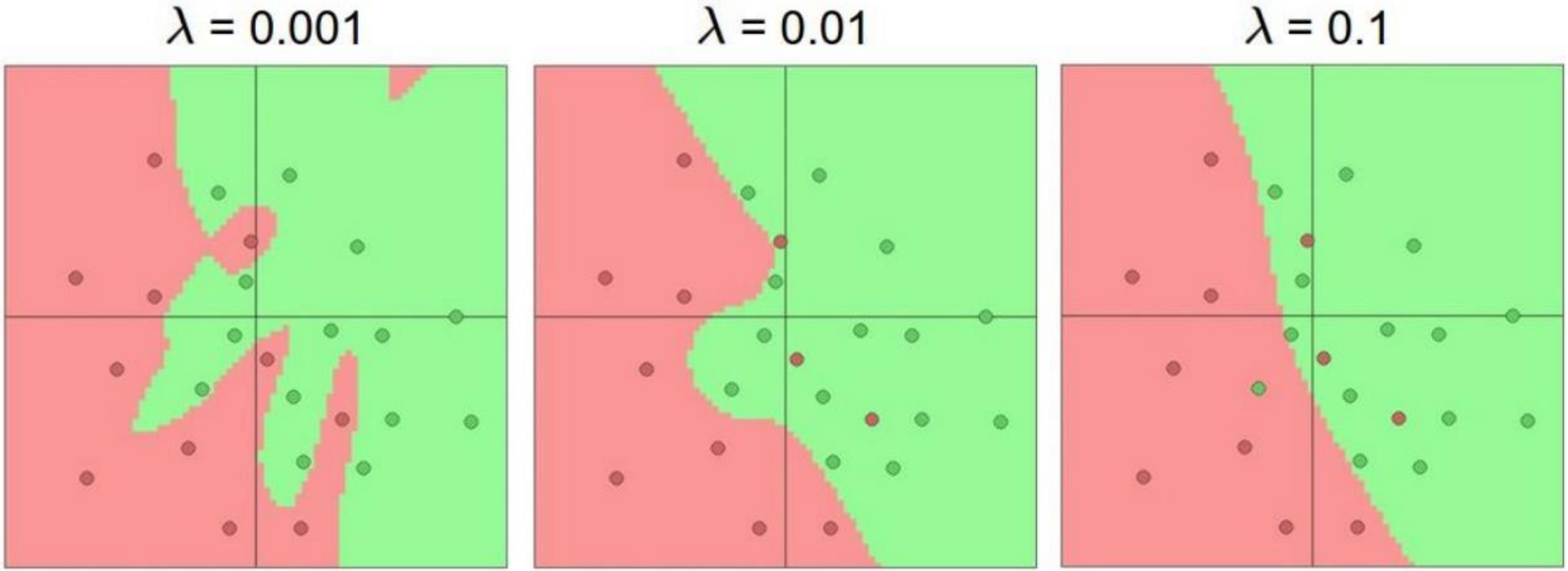
$$R(W) = \sum_k \sum_l W_{k,l}^2$$

$$w_1 = [1, 0, 0, 0]$$
$$w_2 = [0.25, 0.25, 0.25, 0.25]$$

$$w_1^T x = w_2^T x = 1$$

Source : Stanford cs231n Lecture 3

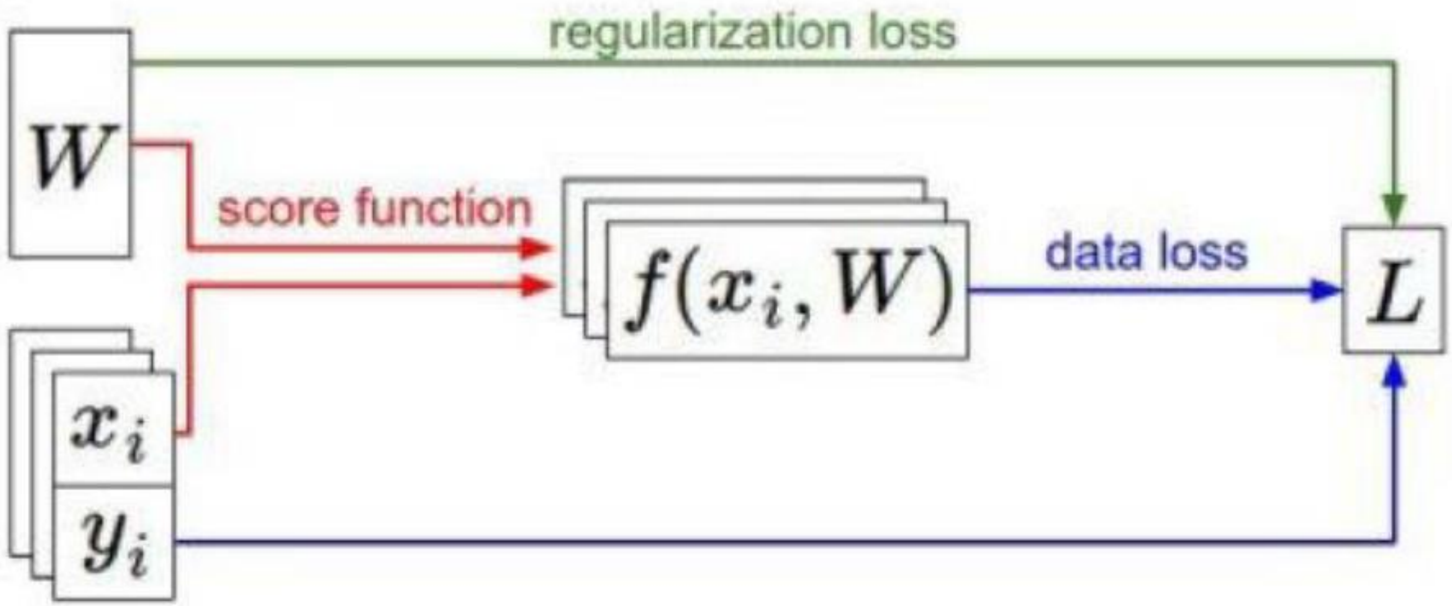
Unit 05 | Regularization



L2 -> Weight를 작게 & 균등하게

Source : Stanford cs231n Lecture 3

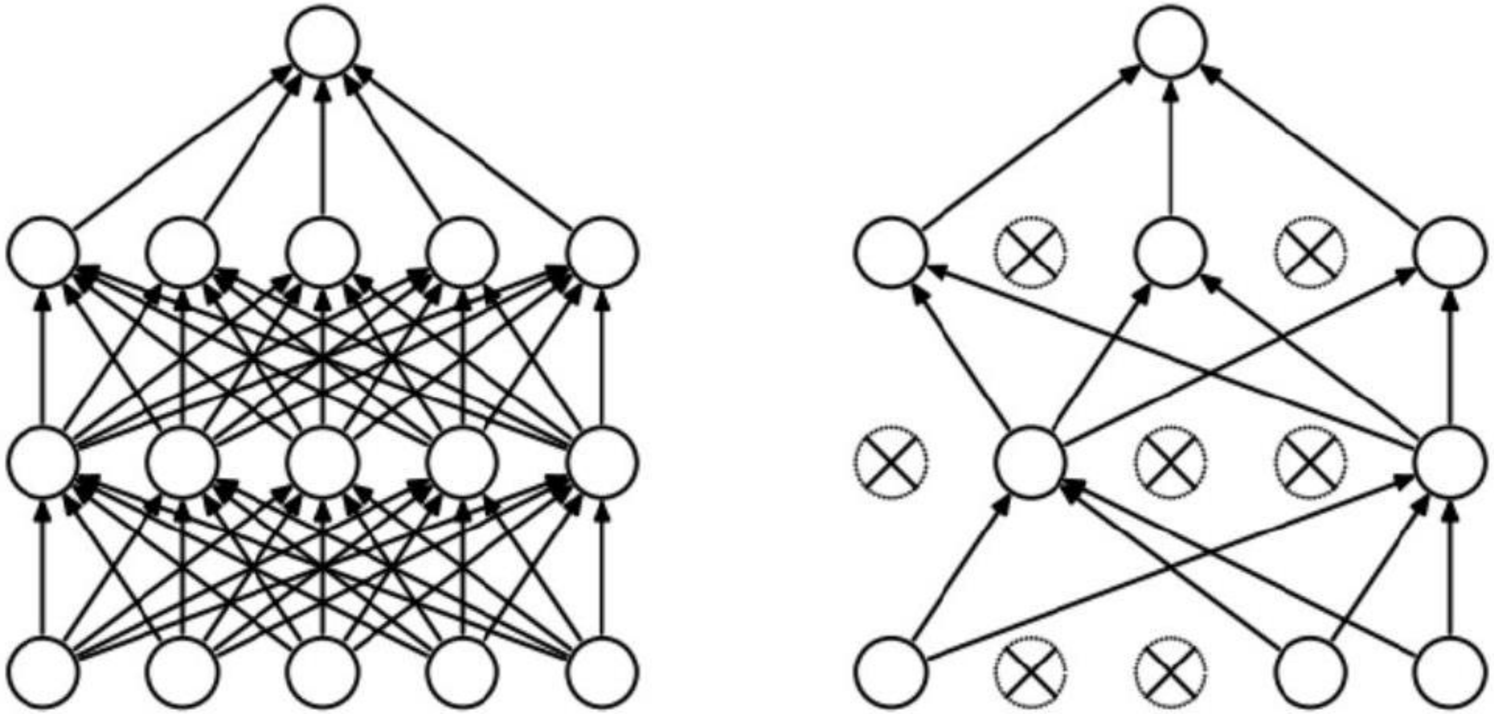
Unit 05 | Regularization



Source : Stanford cs231n Lecture 3

Unit 05 | Regularization

Dropout : layer에 포함된 뉴런 중에서 일부만 참여 시키는 것



(a) Standard Neural Net (b) After applying dropout.

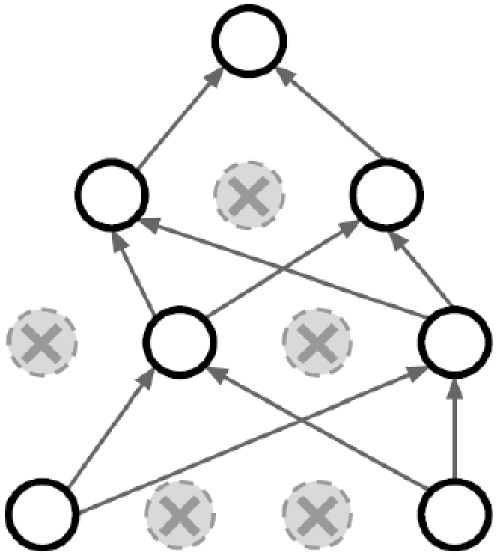
Dropout: A Simple Way to Prevent Neural Networks from Overfitting , N Srivastava et al. 2014

Dropout : layer에 포함된 뉴런 중에서 일부만 참여 시키는 것

Regularization: Dropout

How can this possibly be a good idea?

Forces the network to have a redundant representation;
Prevents co-adaptation of features

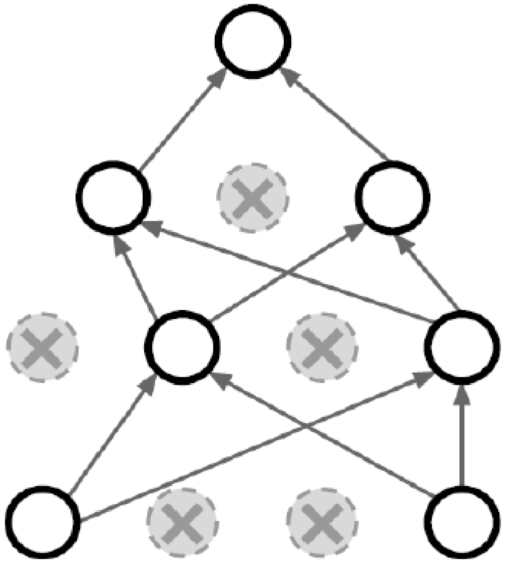


Source : Stanford cs231n Lecture 7

Dropout : layer에 포함된 뉴런 중에서 일부만 참여 시키는 것

Regularization: Dropout

How can this possibly be a good idea?



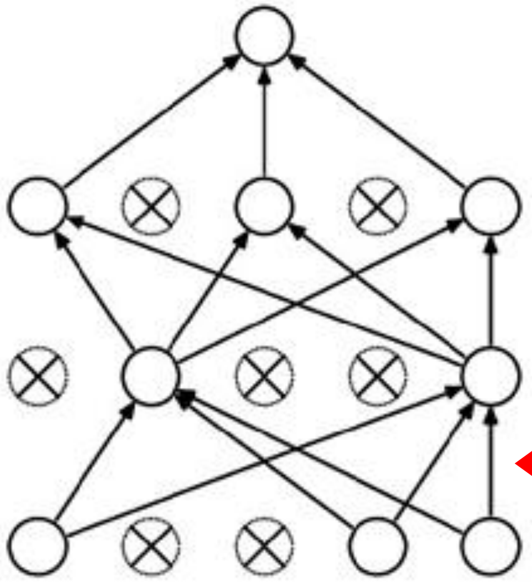
Another interpretation:

Dropout is training a large **ensemble** of models (that share parameters).

Source : Stanford cs231n Lecture 7

Unit 05 | Regularization

Dropout : layer에 포함된 뉴런 중에서 일부만 참여 시키는 것



(b) After applying dropout.

```
Layer1 = tf.nn.dropout(Layer1, keep_prob=keep_prob)
```

Unit 05 | 과제

< 과제 >

Mnist 데이터셋을 사용 .

layer 3개 이상 쌓아서 학습 시키기.

각 epoch 마다 줄어드는 loss를 프린트 통해 확인하기.

오늘 배운 것들을 사용해 정확도 올려 보기!

Q & A

들어주셔서 감사합니다.