exam 개인학습

Lec 4

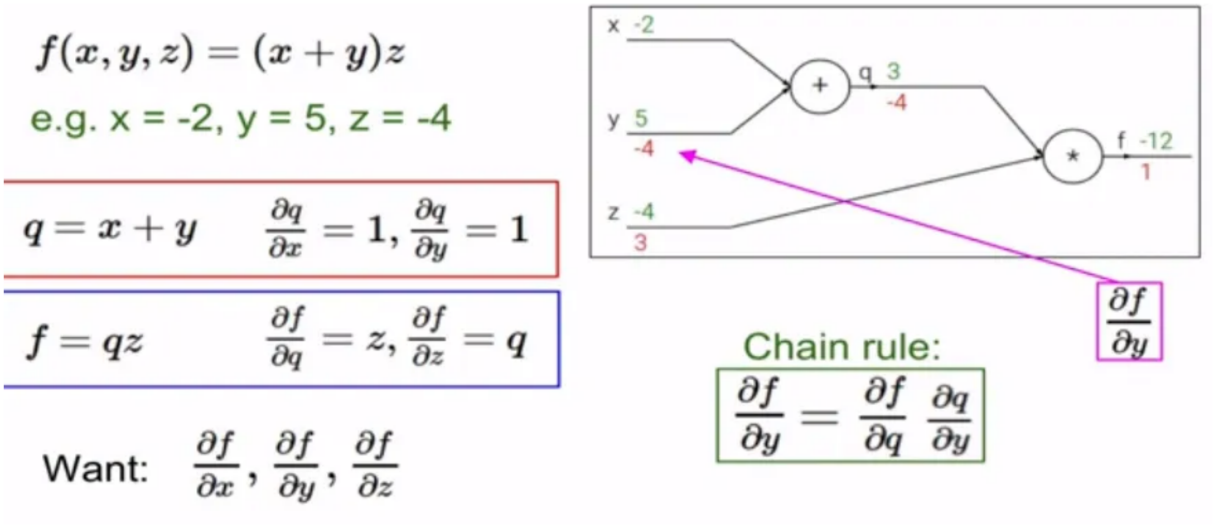
Type something...

01) Introduction to Neural Networks
01_Backpropagation
02_Forward/Backward API
03_Vectorized example
04_Neural Network

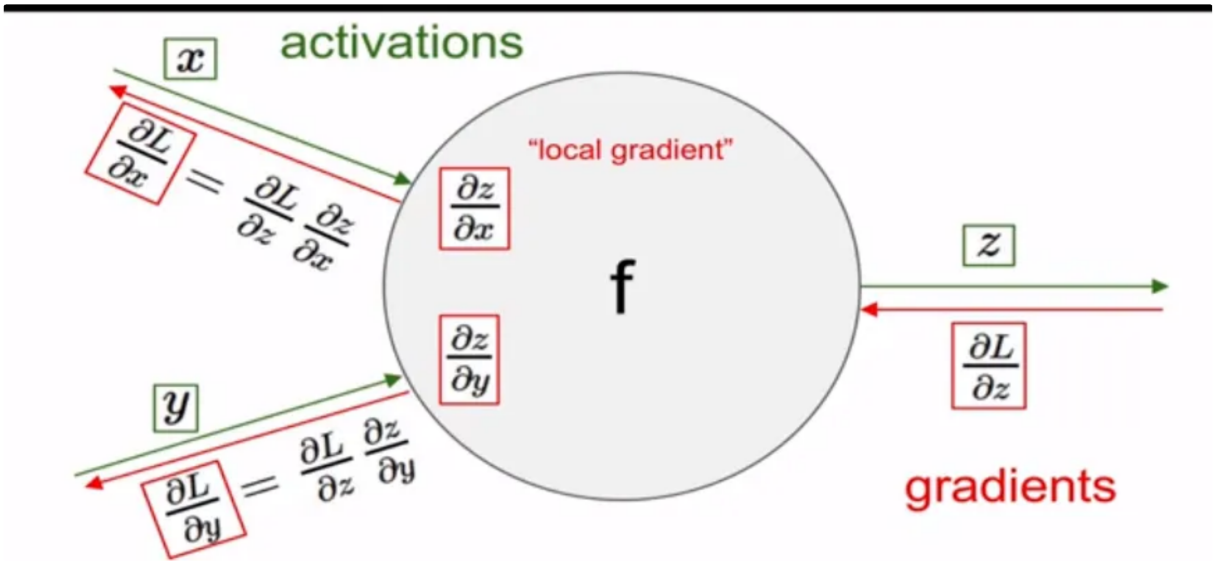
01) Introduction to Neural Networks

01_Backpropagation

- `Backpropagation` 은 chain rule을 활용하여 gradient를 구하는 방식이다. 뒤에서부터 앞으로 오기 때문에 back이 붙었다.



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

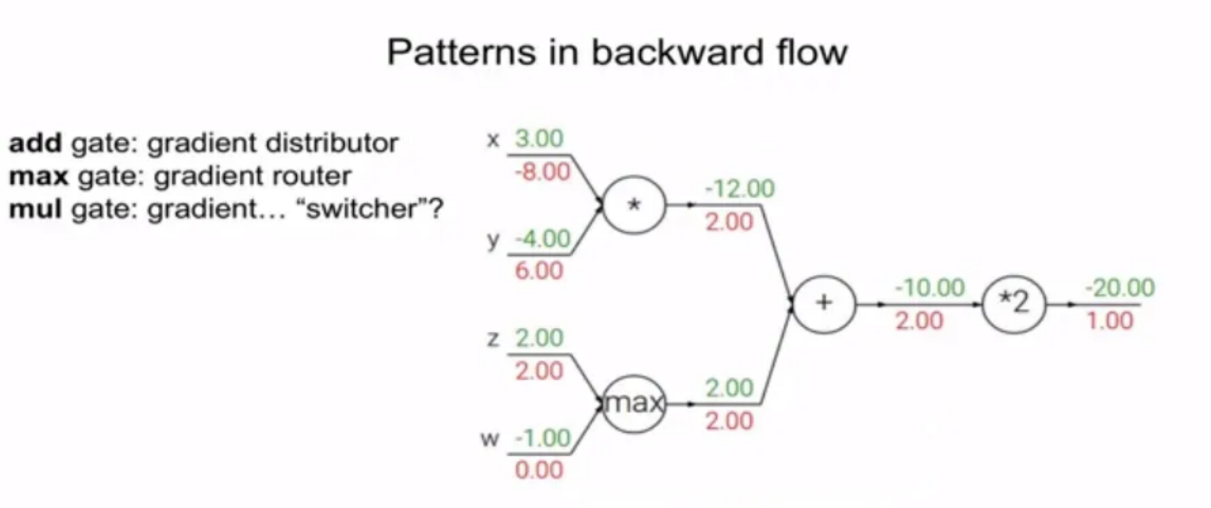


- forward pass시에 cache에 데이터들을 저장해주고 그 값을 backward에 활용하여 `local gradient` 를 구한다.

> 만약 f가 3개의 변수(u,v,w)에 대한 실함수이고, z=f(u,v,w) 이며, u,v,w는 x에 대한 함수 u=g(x), v=h(x), w=k(x)인 경우에 z=f(u,v,w)에서 g(x), h(x), k(x)를 대입하면 z=f(g(x),h(x),k(x))가 됩니다. 이 함수에 대한 연쇄법칙은 다음과 같습니다.

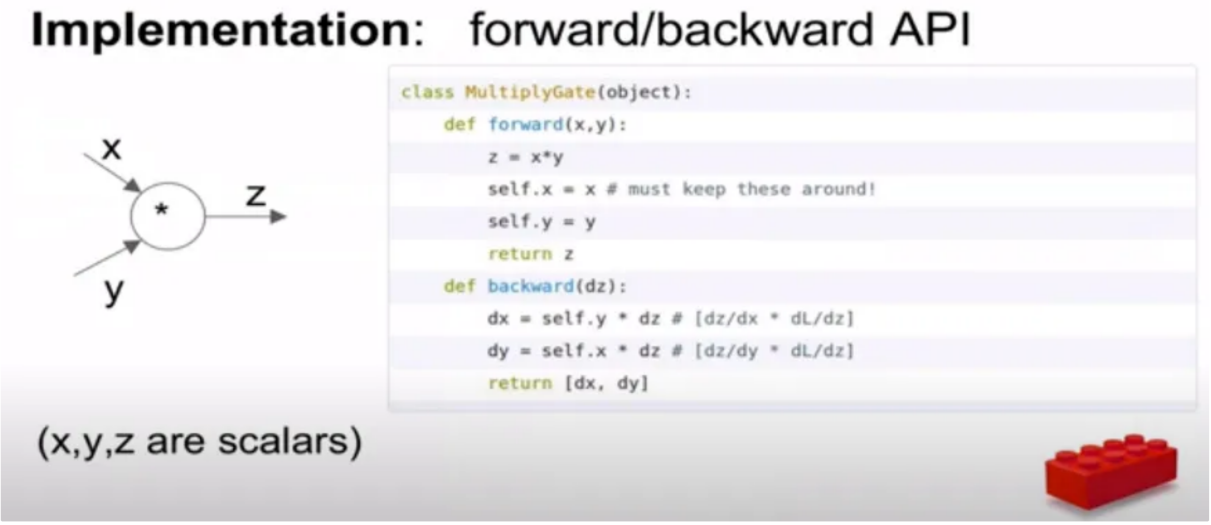
$$\frac{dz}{dx} = \frac{\partial z}{\partial u}\frac{du}{dx} + \frac{\partial z}{\partial v}\frac{dv}{dx} + \frac{\partial z}{\partial w}\frac{dw}{dx}.$$

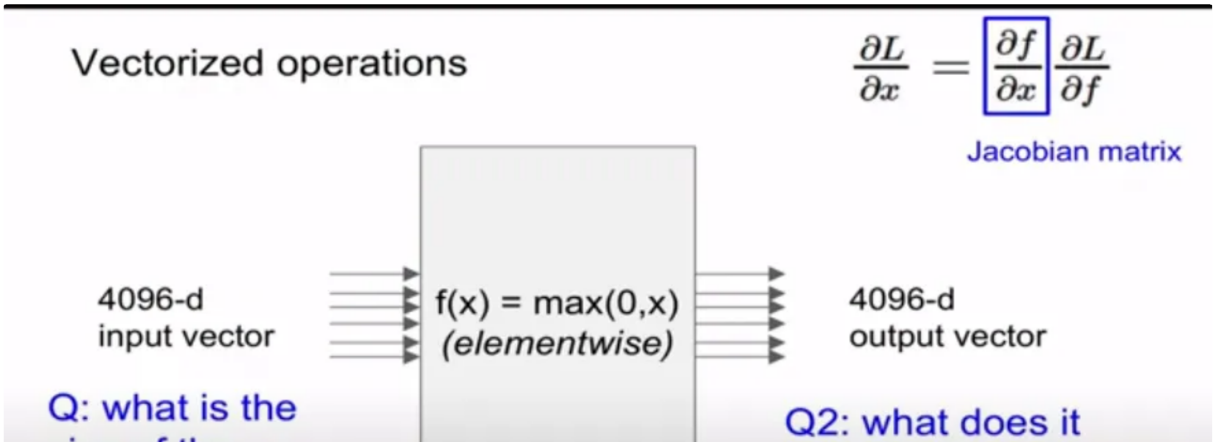
- z가 vector라면 z 각각의 변수들을 미분한 값들을 모두 더해주면 된다.



- Add: local gradient = 1 → distributor
- Max: local gradient 1 or 0 → rotuer
- mul: local gradient swith → switcher

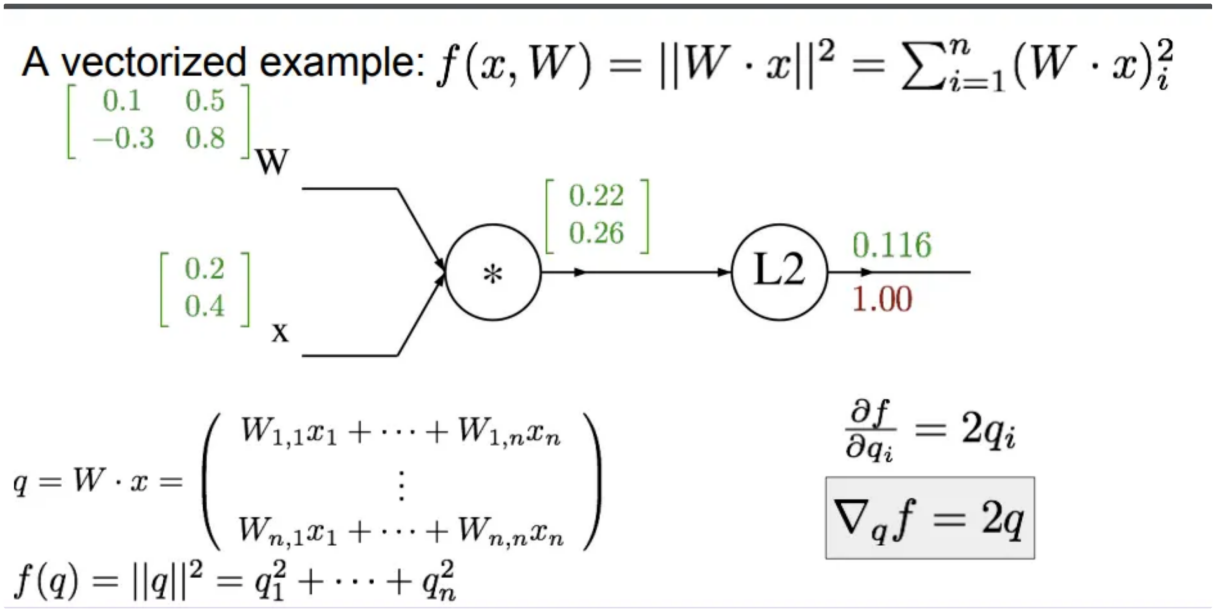
02_Forward/Backward API





- vector일 때는 gradient도 vector로 나온다.
- output 과 input이 사이즈가 4096이라면 자코비안 matrix size = (4096, 4096) 이다.
- 모양은 sparse하게 0과 1이 흩뿌려져 있는 모습이다.

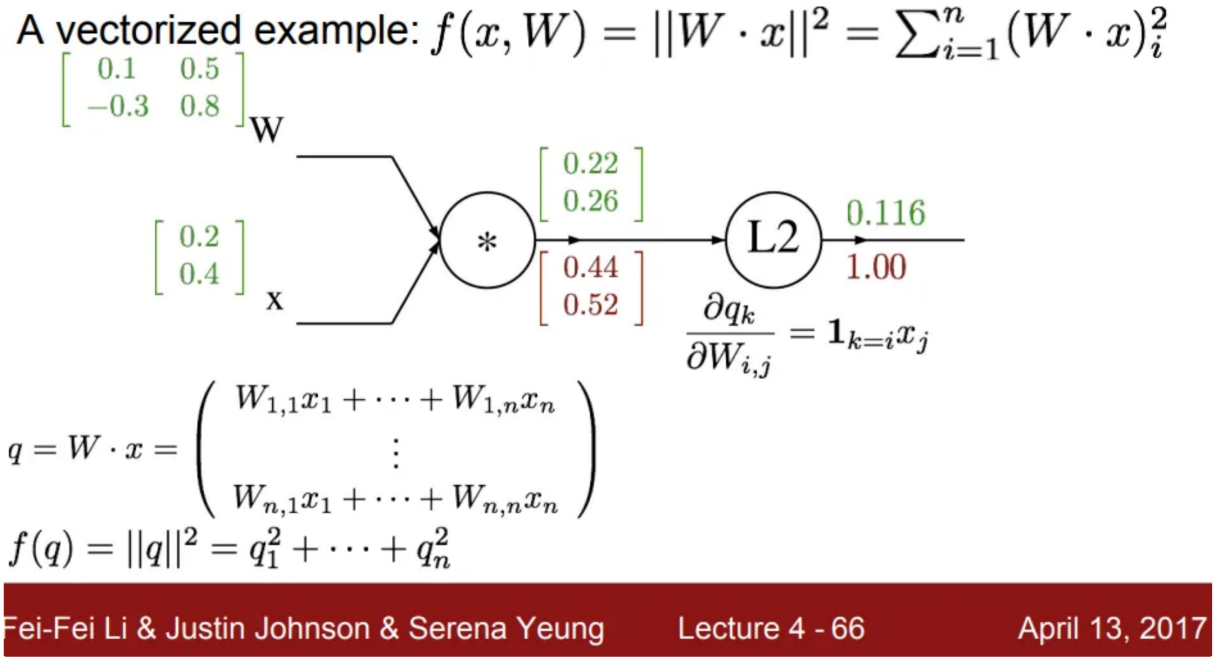
03_Vectorized example



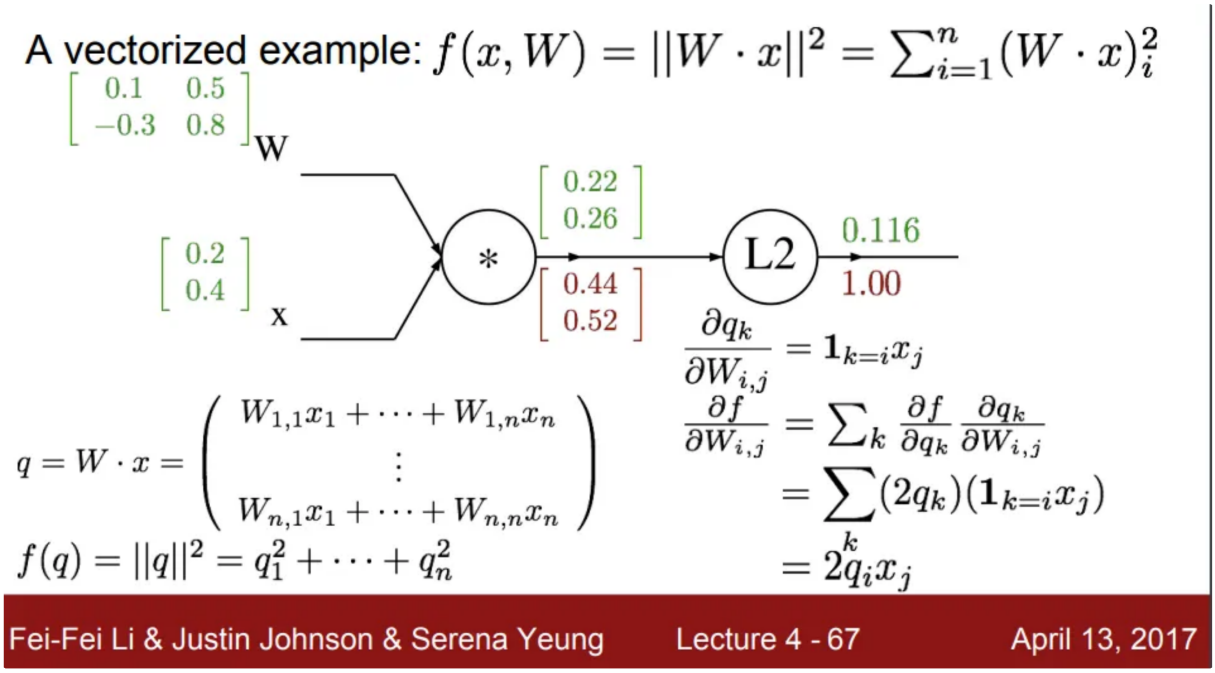
$$f(x, W) = \quad (W * x)^2$$

- q = W * x로 두고 gradient를 천천히 계산해보자.

$$\frac{\partial f}{\partial q} = 2q$$



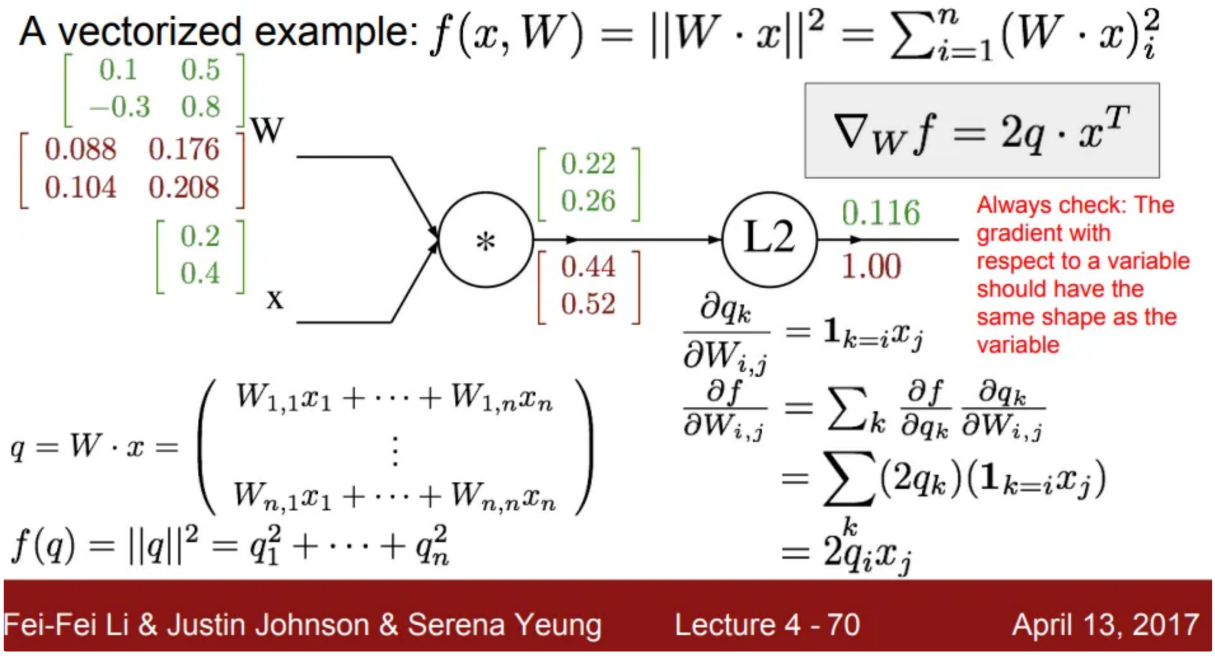
$$\frac{\partial q_k}{\partial W_{i,j}} = 1_{k=i} x_j$$



$$\frac{\partial f}{\partial W_{i,j}} = \sum_k \frac{\partial f}{\partial q_k} \frac{\partial q_k}{\partial W_{i,j}}$$

$$= \sum (2q_k)(1_{k=i} x_j)$$

$$= 2q_i x_j$$



- 이를 벡터로 표현하면,

$$\frac{\partial f}{\partial W} = 2q * x^T$$

- q : n x 1, W: n x n, x = n x 1
- x에 대한 미분은 생략하겠다.

04_Neural Network

Neural Network: without the brain stuff

(**Before**) Linear score function: $f = Wx$

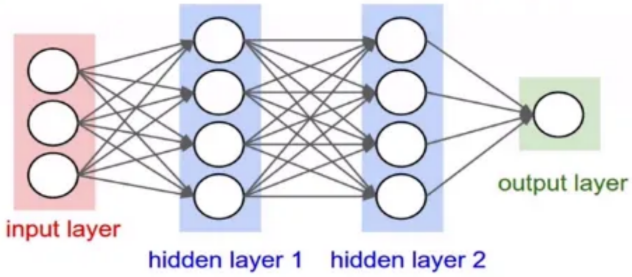
(**Now**) 2-layer Neural Network or 3-layer Neural Network

$$f = W_2 \max(0, W_1 x)$$

$$f = W_3 \max(0, W_2 \max(0, W_1 x))$$

- activation fucntion을 집어넣어서 nonlinearity를 보장해준다.
- 2-layer Neural Network: Linear → activation → Linear
- 3-layer Neural Network: Linear → activation → Linear → actiavtion → Linear
- 2-layer Neural Network: Linear → activation → Linear

Example Feed-forward computation of a Neural Network




```
# forward-pass of a 3-layer neural network:
f = lambda x: 1.0/(1.0 + np.exp(-x)) # activation function (use sigmoid)
x = np.random.randn(3, 1) # random input vector of three numbers (3x1)
h1 = f(np.dot(W1, x) + b1) # calculate first hidden layer activations (4x1)
h2 = f(np.dot(W2, h1) + b2) # calculate second hidden layer activations (4x1)
out = np.dot(W3, h2) + b3 # output neuron (1x1)
```

- sigmoid를 사용한 3-layer neural network의 코드이다.