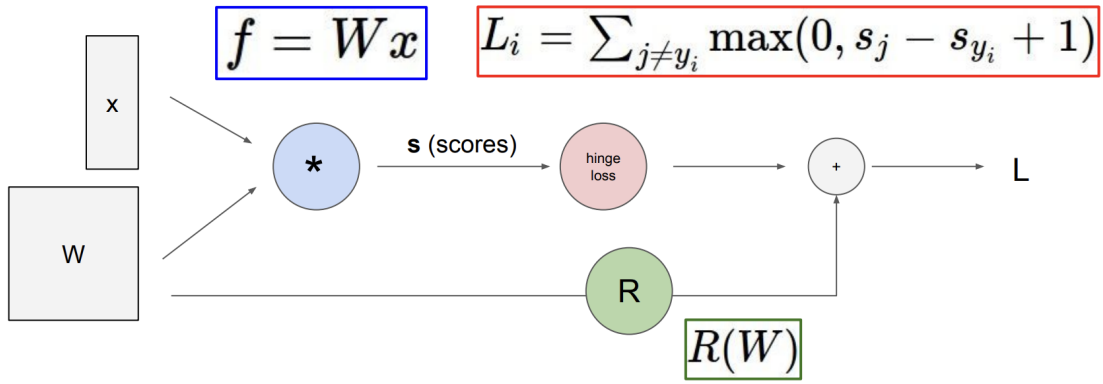
Lecture 4) Backpropagation and Neural Networks

- Lecture3 Review

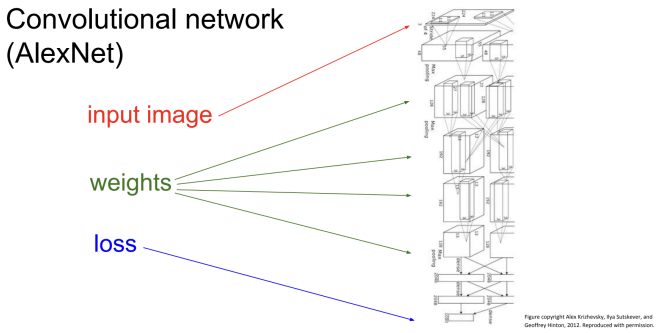
$$s = f(x; W) = Wx \quad \text{scores function}$$
$$L_i = \sum_{j \neq y_i} \max(0, s_j - s_{y_i} + 1) \quad \text{SVM loss}$$
$$L = \frac{1}{N} \sum_{i=1}^{N} L_i + \sum_k W_k^2 \quad \text{data loss + regularization}$$

want $\nabla_W L$

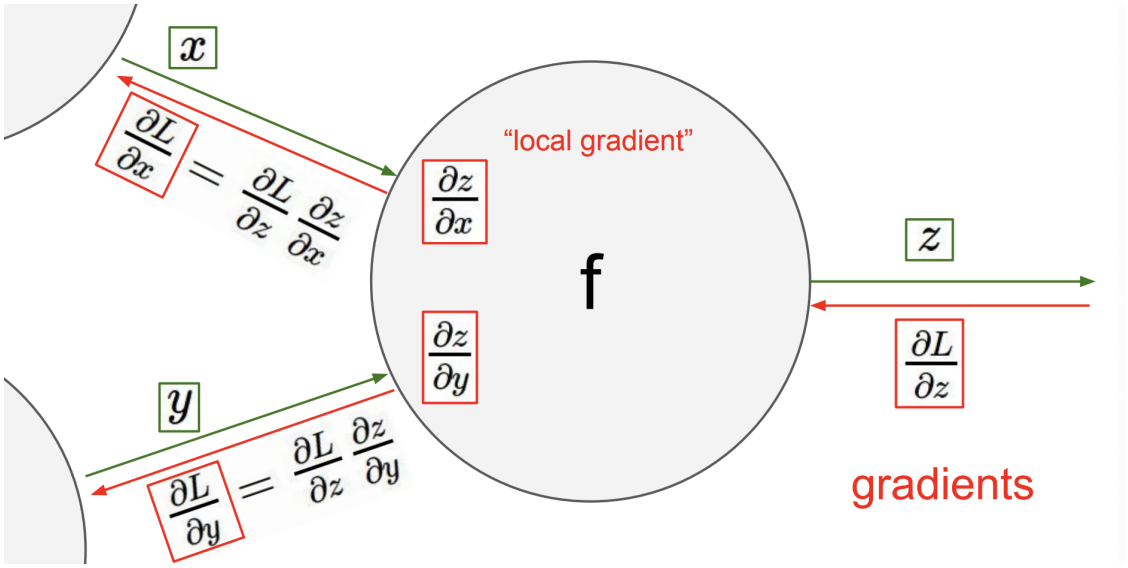
Backpropagation: Graph 계산하기



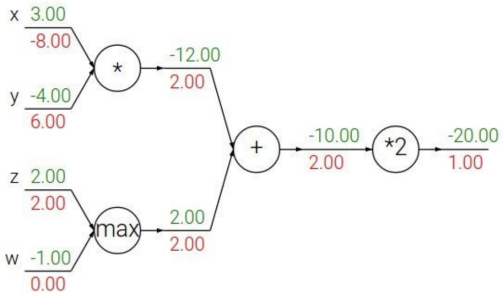
- 설명
  - 파랑: Matrix multiplication
  - 빨강: Data (loss) term
  - 녹색: Regularization term
- 네트워크 구조



- Backpropagation 계산
  - Computational graph로 해석
  - Chain rule을 사용



  - gradient = local gradient * upstream gradient
    - local gradient: 현재 노드에서 계산한 gradient
    - upstream gradient: 이전 단계까지의 gradient
  - sigmoid gate 처럼, 원하는 연산을 모아놓을 수 있다.
- Q. Gates?



  - Max gate: Input 중에 큰 값을 내보내는 게이트. gradient router.
    큰 값이 최종 연산에 영향을 주었기 때문에, back prop에도 큰 값을 되돌려 보내는게 맞다.
  - Add gate: gradient distributer
  - Mul gate: gradient switcher, scaler
- Jacobian matrix → diagonal matrix

Backpropagation 정리

- backprop는 gradient를 얻기 위한 주요한 방법
- forward: 결과를 얻기 위해 forward pass로 연산
- backward: chain rule을 사용하여 loss의 gradient를 계산한다.

Neural Network

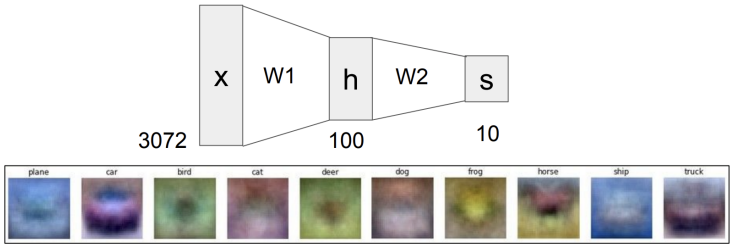
- 식

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

(**Now**) 2-layer Neural Network $f = W_2 \max(0, W_1 x)$

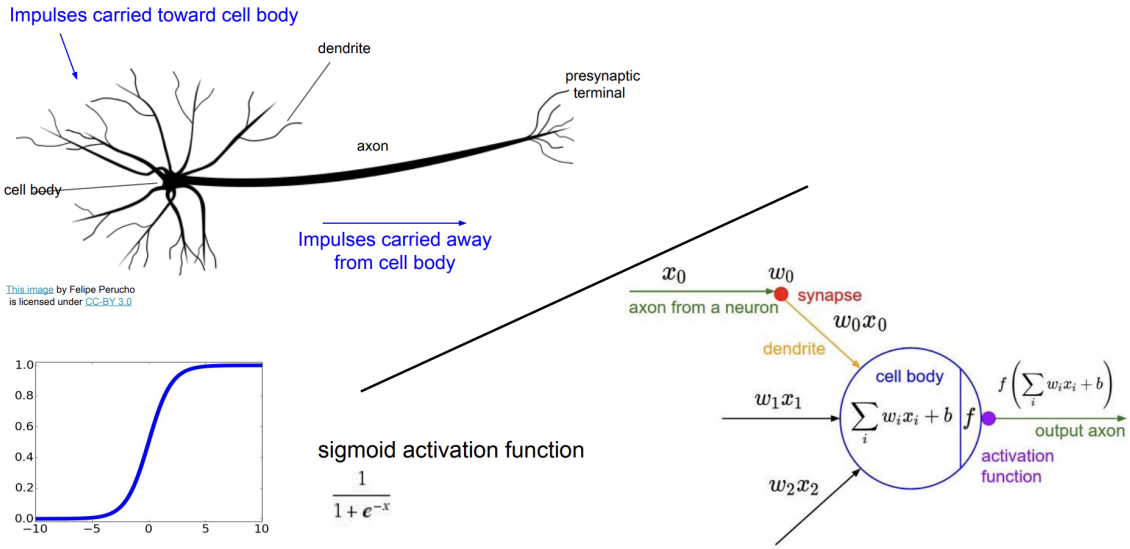
or 3-layer Neural Network $f = W_3 \max(0, W_2 \max(0, W_1 x))$

- 레이어를 쌓는 형태를 보인다.

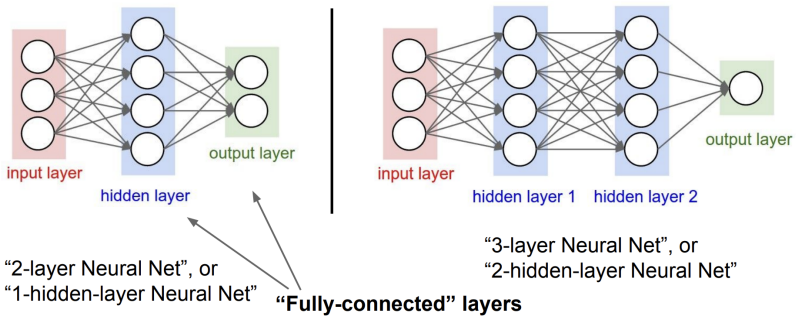


  - 계층적인 방식으로 쌓게 되고, 더 복잡하고 비선형적인 모델을 만들기 위함

- 비유



- 아키텍처

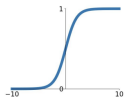


Activation function

- 비선형성을 부여. 뒤의 노드에 얼마나 큰 영향을 줄 것인지 결정하는 역할
- examples

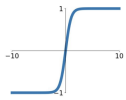
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



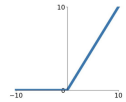
tanh

$$\tanh(x)$$



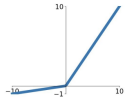
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$



Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$