Deep Learning

Representative Power of Multilayer Networks

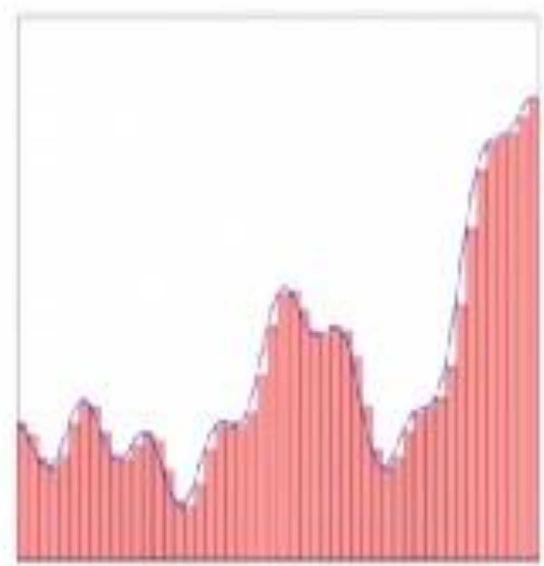
- A multilayer network of perceptrons with a single hidden layer can be used to approximate any Boolean function precisely

- A multilayer network of sigmoid neurons with a single hidden layer can be used to approximate any continuous function to any desired precision

↑ sigmoid

Multilayer Network

- For any function $f(x)$: $R^n \longrightarrow R^m$, we can find a network with enough neurons, whose output $g(x)$ satisfies $|g(x) - f(x)| < \epsilon$
- Such an arbitrary function can be represented by several tower functions



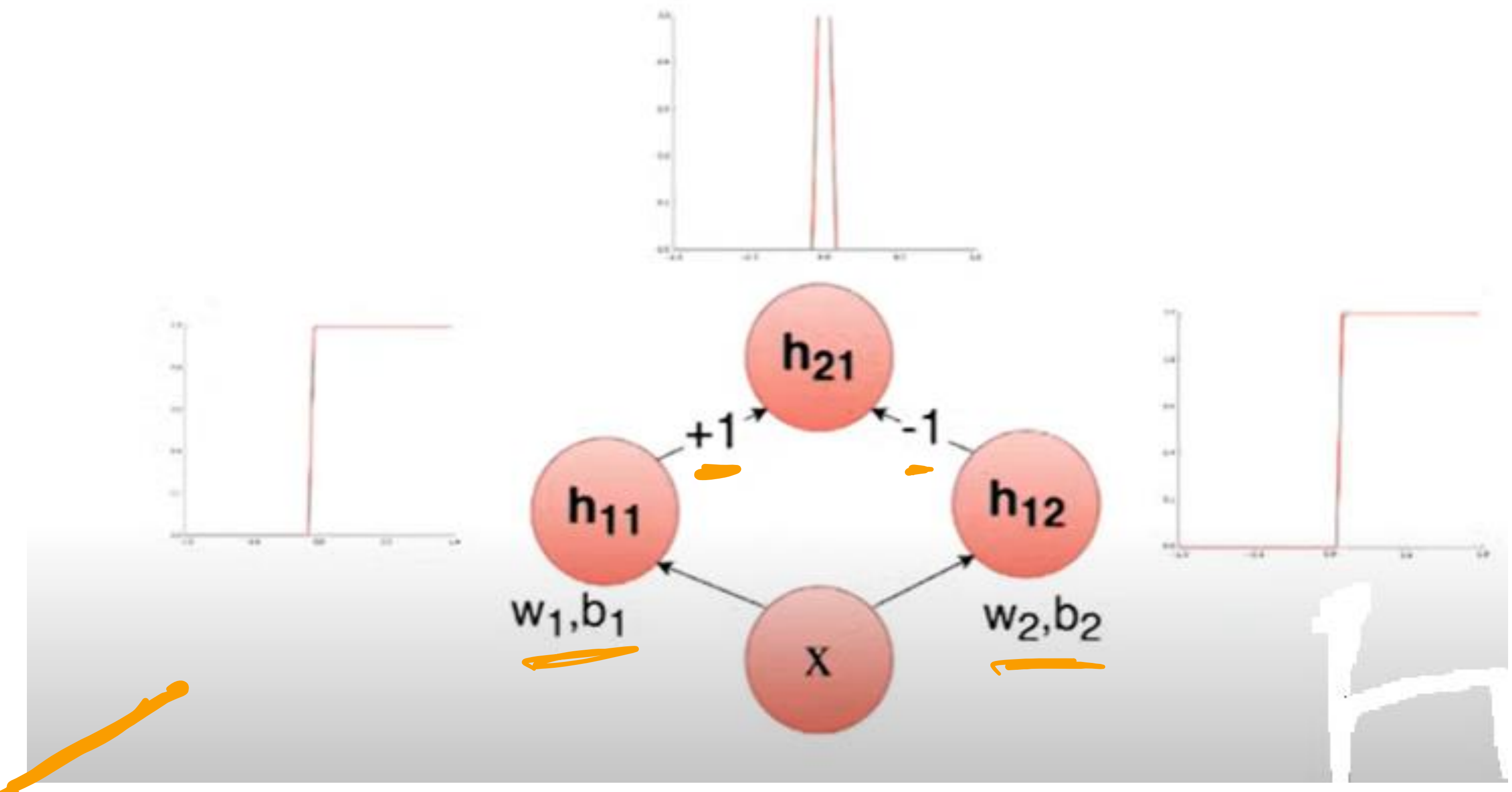
Multilayer Network

- All tower functions are similar and only differ in height and position on x-axis
- A black box takes some input and constructs a tower function
  - A network can add them up to approximate the function
- If we take the logistic function and set *w* to a very high value, we can recover step function
  - *w* controls the slope of the logistic function
- Can also adjust value of *b* to control position on x-axis at which function transitions from 0 to 1

b

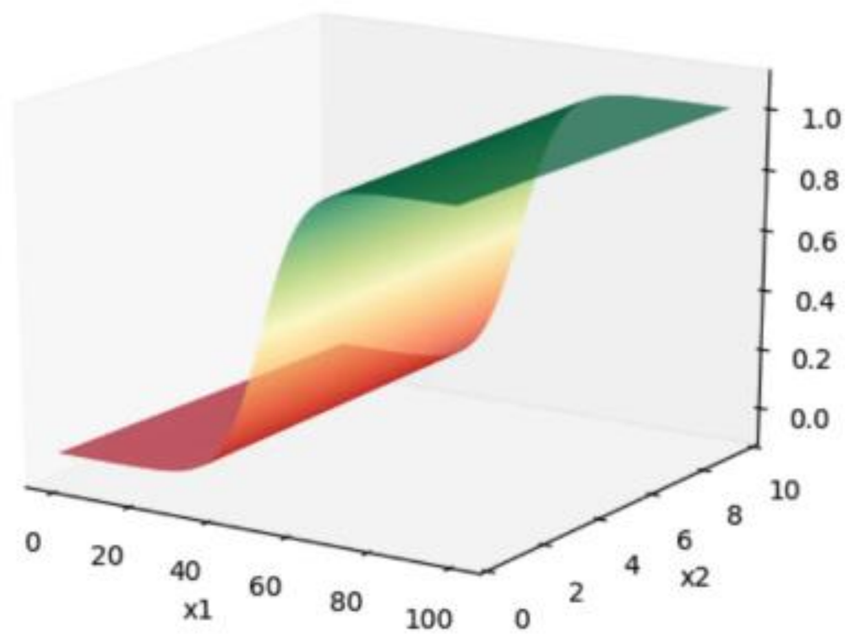
Multilayer Network

Take two such sigmoid functions, with different *b*'s, and subtract them – will get a tower function

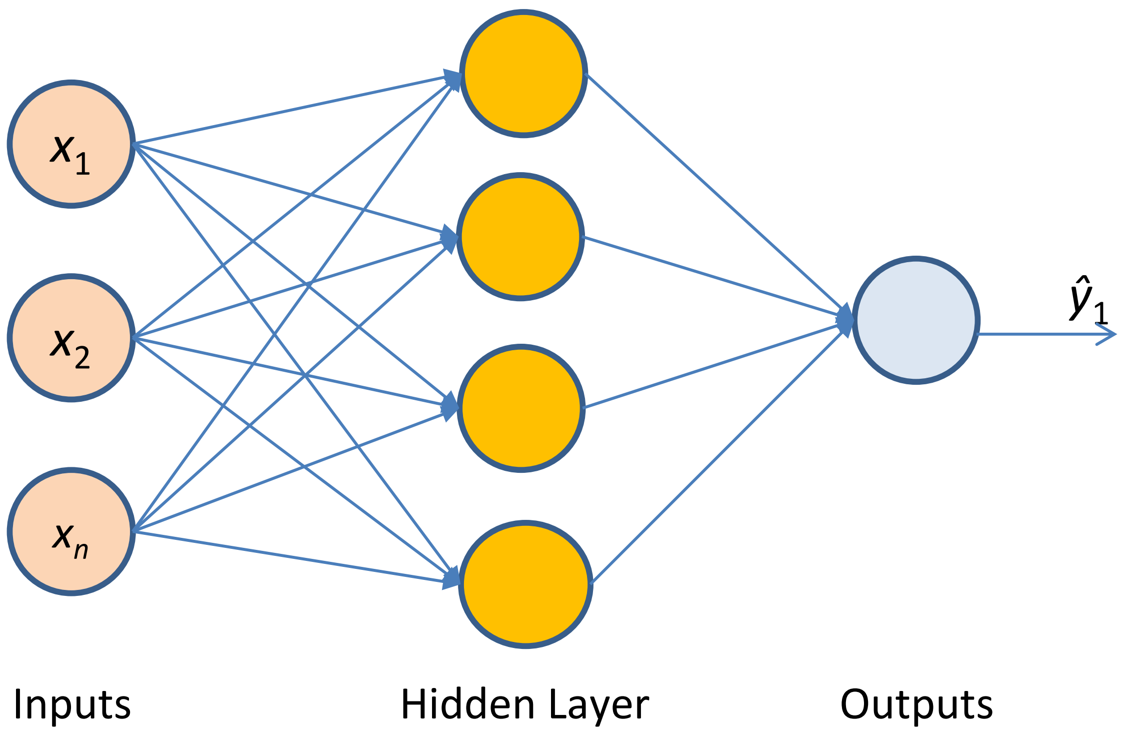


Multilayer Network

- More input parameters??
- Ex. 2 parameters



Single Hidden Layer Neural Network



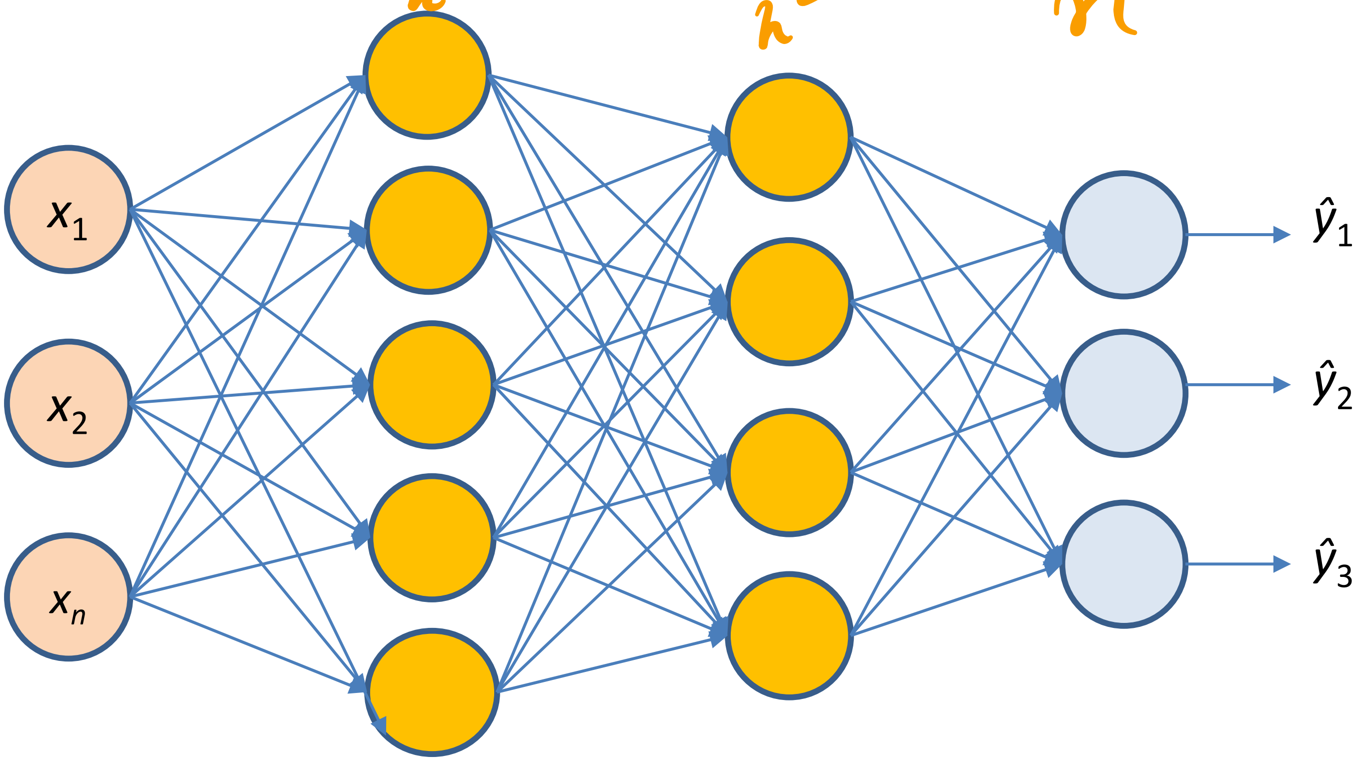
Hidden Layer: States of nodes are unobserved

Inputs are densely connected to perceptrons, hence they are called **Dense** layers or **Fully Connected** layers

Feedforward Neural Network

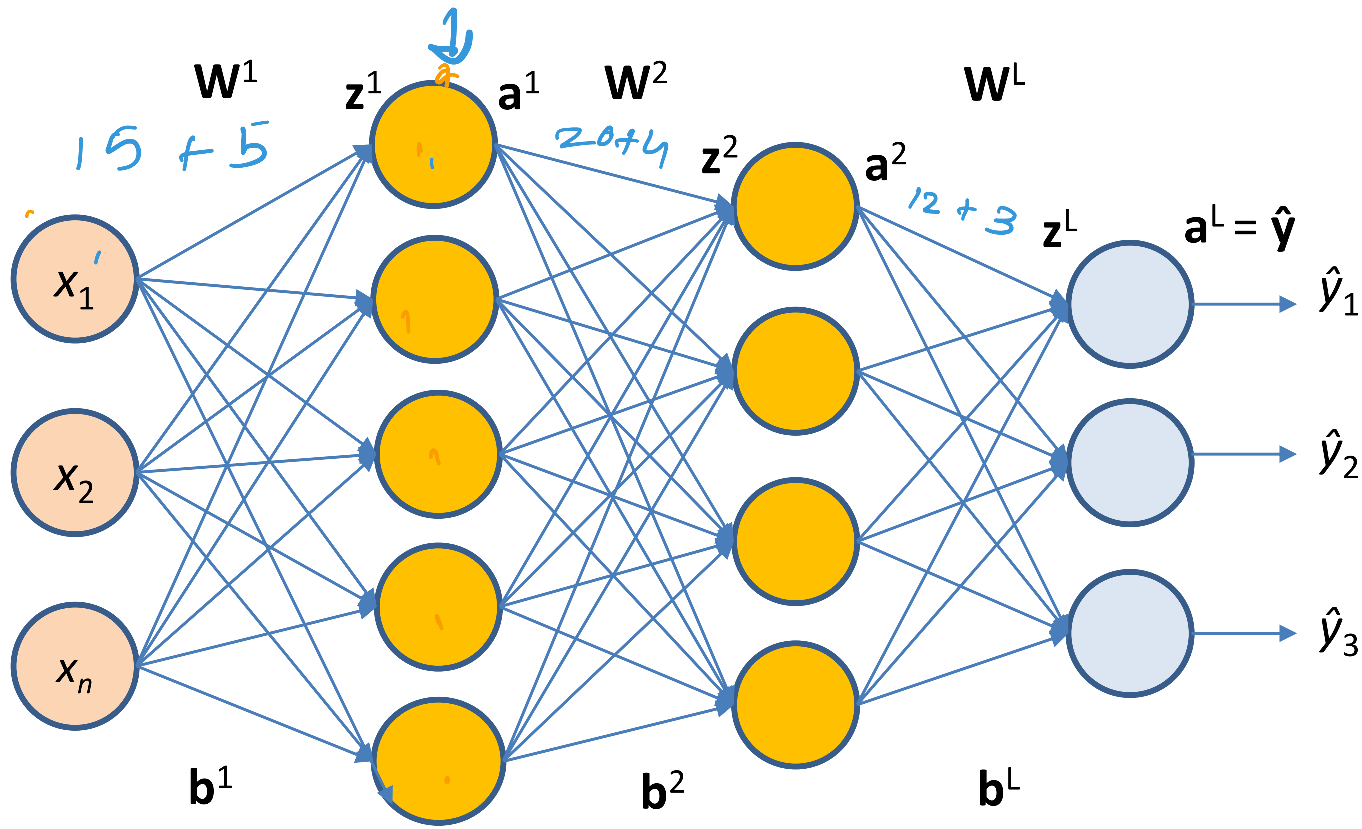
- Input is an n-dimensional vector (0^{th} layer) ∈ R^n
- W^1 Network has L-1 hidden layers
- 1 output layer containing k neurons (ex. for k classes)
- Each neuron – aggregation and activation

h^1 h^2 n



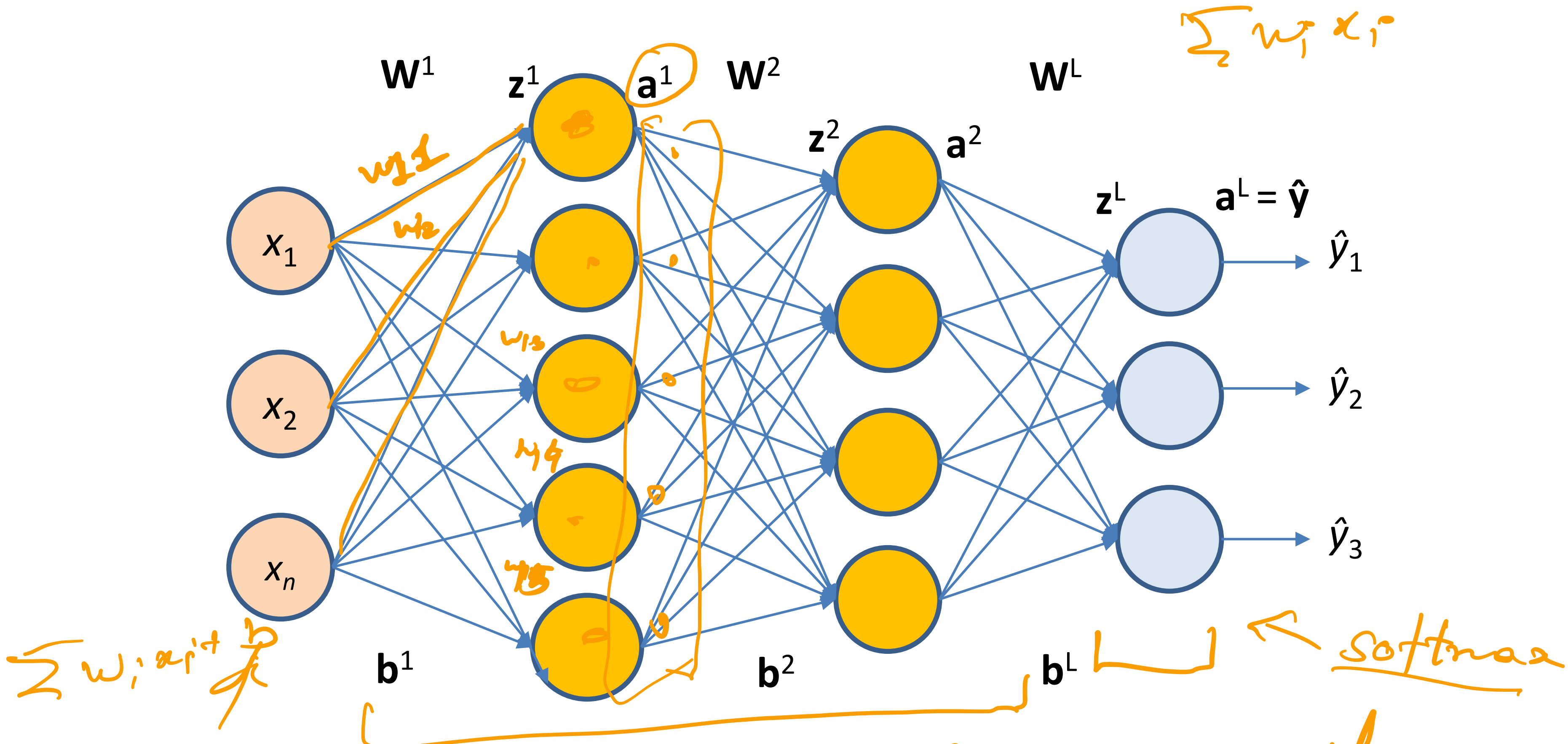
L-1

Feedforward Neural Network



Assuming n^i neurons in hidden layer h^i, $W^i \in R^{n(i-1)*ni}$ and $b^i \in R^{ni}$ between layers i -1 and i for $0<i<L$

$W^L \in R^{ni*k}$ and $b^L \in R^k$ between last hidden layer and output layer

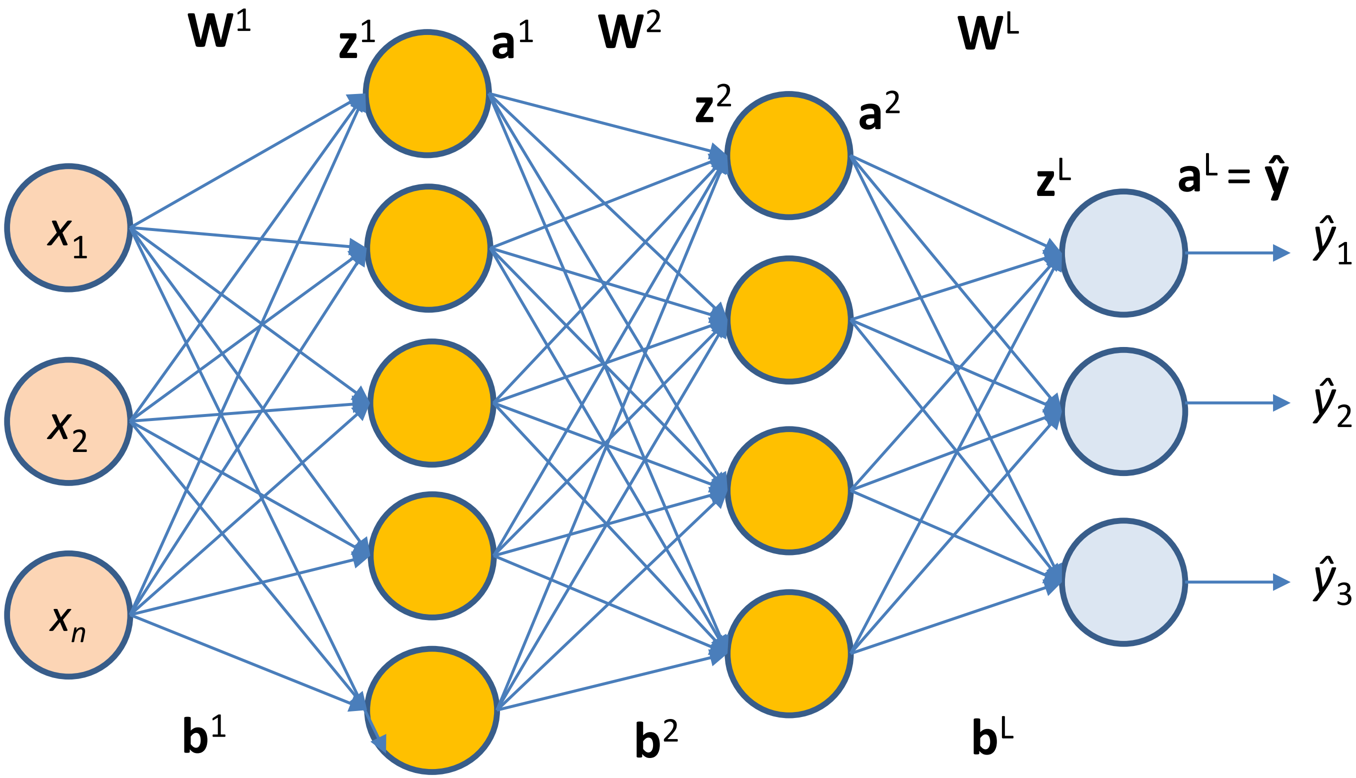


ReLU, Sigmoid

Aggregation at layer i : $\mathbf{z}^i = \mathbf{W}^i \mathbf{a}^{i-1} + \mathbf{b}^i$

For first hidden layer: $\mathbf{z}^1 = \mathbf{W}^1 \mathbf{a}^0 + \mathbf{b}^1$

$$\begin{bmatrix} z_1{}^1 \\ z_2{}^1 \\ z_3{}^1 \end{bmatrix} = \begin{pmatrix} W_{11} & W_{12} & W_{13} \\ W_{21} & W_{22} & W_{23} \\ W_{31} & W_{32} & W_{33} \end{pmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{pmatrix} \sum W_{1i} x_i + b_1 \\ \sum W_{2i} x_i + b_2 \\ \sum W_{3i} \mathrm{x_i} + b_3 \end{pmatrix}$$



Activation at layer i = $g(\mathbf{z}^i) = g(\mathbf{b}^i + \mathbf{W}^i\, \mathbf{a}^{i-1})$

For first hidden layer: $g(\mathbf{z}^1) = g(\mathbf{b}^1 + \mathbf{W}^1\, \mathbf{a}^0)$

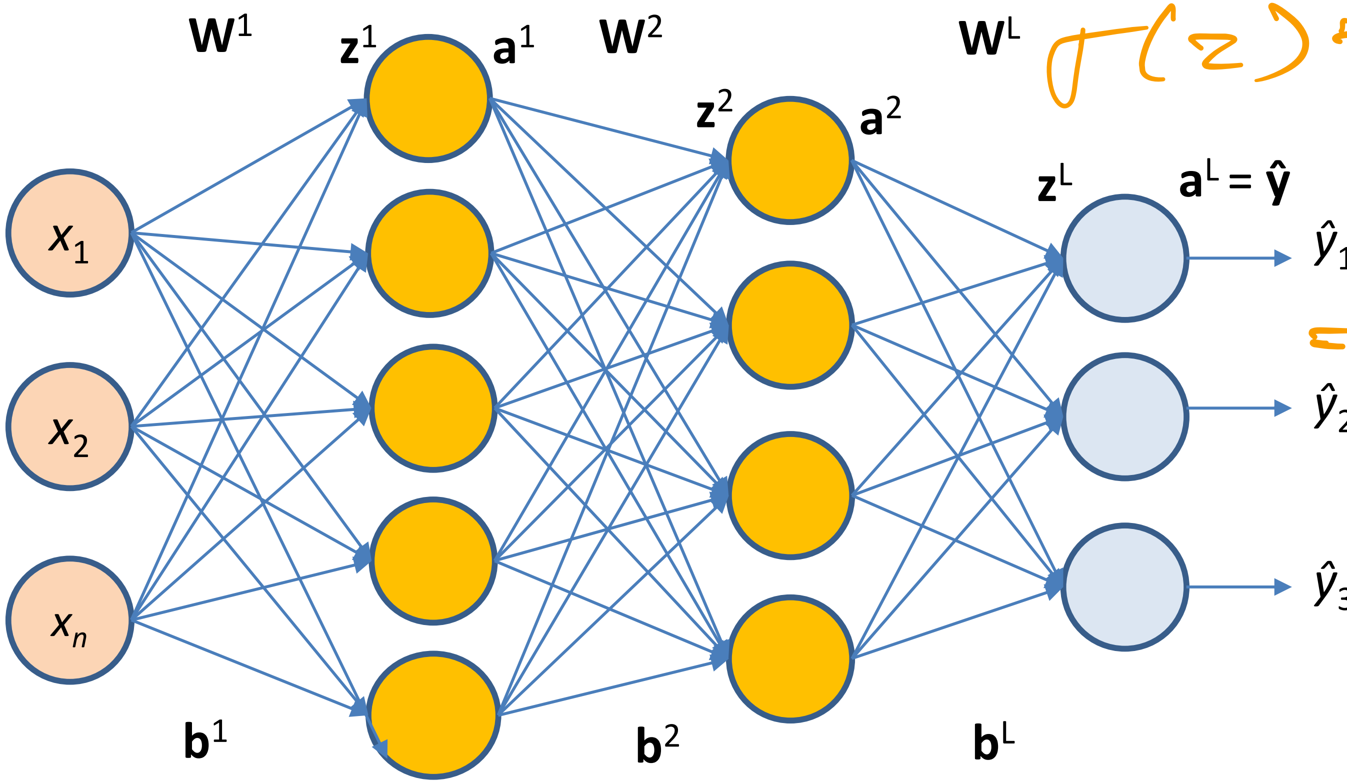
$$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} = \begin{pmatrix} g(z_1) \\ g(z_2) \\ g(z_3) \end{pmatrix}$$

$y = \tanh$

$y' = 1 - y^2$

Eg. $g(z_1) = \sigma(z_1) = 1 / (1 + e^{-z1})$

g: activation function (logistic, tanh, linear etc.)



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

$\sigma'(z) = \sigma(z)(1-\sigma(z))$

Aggregation at output layer L = z^L = $\mathbf{W^L a^{L-1} + b^L}$

$$z_1 = w_{11}x_1 + w_{12}x_2 + w_{13}x_3 + b$$

$$z_2 = w_{21}x_1 + w_{22}x_2 + w_{23}x_3 + b$$

Activation at output layer L = $\mathbf{\hat{y}}$ = $g(z^L)$ = $g(\mathbf{W^L a^{L-1} + b^L})$

$$\begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} g(z_1) \\ g(z_2) \end{pmatrix}$$

Learning parameters

In given example, dimensions of parameters:

- $\mathbf{W^1}$: $n^1 * n$ $\quad\quad$ $\mathbf{b^1}$: n^1
- $\mathbf{W^2}$: $n^2 * n^1$ $\quad\quad$ $\mathbf{b^2}$: n^2
- $\mathbf{W^L}$: $n^2 * k$ $\quad\quad$ $\mathbf{b^L}$: k

- Assuming L layers and n^i neurons in hidden layer h^i and k neurons in output layer, no. of parameters to be learned:
  - Weights: $(L\text{-}1)*(n^{i-1} * n^i) + (n*k)$ $\quad\quad$ for $0<i<L$
  - Bias: $(L\text{-}1)*n^i + k$

Learning parameters

- **Data:** $\{x_i, y_i\}$ $i = 1..m$
- **Model:** $[w, b]$

  $\hat{\mathbf{y}} = f(\mathbf{x}) = g(\mathbf{W}^3 g(\mathbf{W}^2 g(\mathbf{W}^1\mathbf{x} + \mathbf{b}^1) + \mathbf{b}^2) + \mathbf{b}^3)$

  $\hat{\mathbf{y}} = [\hat{y}^1 \quad \hat{y}^2 \ldots \hat{y}^k]$
- **Algorithm:** Gradient Descent with back Propagation
- **Loss/Error function:** Sum of squared error loss

  $min \frac{1}{N} \sum_{i=1}^{m} \sum_{j=1}^{k} (\hat{y}_j^i - y_j^i)$ for i^{th} sample for all classes j

Learning parameters

- Gradient Descent:

  t:=0;

  $max\_iterations$:=1000;

  Initialize $\boldsymbol{\theta_0} := [\mathbf{W^1}_0, \ldots \mathbf{W^L}_0, \mathbf{b^1}_0 \ldots \mathbf{b^L}_0]$;

  while t++ < $max\_iterations$ do

  $$\boldsymbol{\theta_{t+1}} := \boldsymbol{\theta_t} - \eta \nabla \boldsymbol{\theta_t};$$

  end

where, $\nabla\theta_t = [\frac{\partial L(\theta)}{\partial W_t}, \frac{\partial L(\theta)}{\partial b_t}]^T$

$\nabla\theta$ composed of:

- $\nabla W^1, \nabla W^2, \ldots \nabla W^{L-1} \in R^{n(i-1)xni}$, $\nabla W^L \in R^{nxk}$
- $\nabla b^1, \nabla b^2, \ldots \nabla b^{L-1} \in R^{ni}$, $\nabla b^L \in R^k$

Loss function

- Loss function should capture how much $\hat{y}_i$ deviates from y_i
- $y_i \in R^n$ then squared error loss can be used:

  $$L(\theta) = (1/m)*\sum (y_i - \hat{y}_i)^2$$

- Problems with squared error loss:

  $$\frac{\partial L(w,b)}{\partial w} = (\hat{y} - y) * \hat{y} * (1 - \hat{y}) * x$$

  – If $y_i = 1$ and $\hat{y}_i \sim 0$, $\frac{\partial L(w,b)}{\partial w} \sim 0$ Undesirable
  – If $y_i = 0$ and $\hat{y}_i \sim 1$, $\frac{\partial L(w,b)}{\partial w} \sim 0$ Undesirable
  – Weight updation becomes very slow

Loss function

- Information content (IC):
  - Events with high probability have low information content
    - "The sun will rise tomorrow"
  - Events with low probability have high information content
    - "There will be a cyclone tomorrow"
- $IC(\mathrm{A}) = -\log_2(p(\mathrm{A}))$
- Entropy: Expected information content = $\sum p_i * IC(i)$

  $= - \sum p_i \log_2(\mathrm{p_i})$

Loss function

Entropy: y_i = [0 1 0 0] //Team B wins game

$\hat{y}_i$ = [0.2 0.1 0.4 0.3] //Our prediction

10K 5K 8K 1K //Profit for each team win

Expected profit??

- Entropy: Expected information content = $\sum p_i \, IC(i)$

$= - \sum p_i \log_2(p_i)$

Actual predicted.

Loss function

- Cross-entropy: gives a measure on how close a predicted distribution is to a true distribution
  - True distribution p_i , Estimated distribution q_i
  - Estimated information content = $\boxed{- \sum p_i \log_2(q_i)}$
  - Capture difference between two probability distributions
  - If prediction is close to actual, cross entropy will be low

$L(\theta) = - \sum y_c \log_2(\hat{y}_c)$	for all k classes
$y_c = 1$	if $c = t$ (true class)
$= 0$	otherwise

$$\boxed{L(\theta) = - \log_2(\hat{y}_t)}$$

Loss function

- Objective function for classification:
  - Cross-entropy Loss

    minimize: $L(\theta) = -\log_2(\hat{y}_t)$

$$\frac{\partial L}{\partial w} = -\frac{1}{\hat{y}_t}\frac{\partial \hat{y}}{\partial w}$$

$\hat{y}_t$: predicted probability of correct event

$\log_2(\hat{y}_t)$: probability that x belongs to t^{th} class, log-likelihood of data

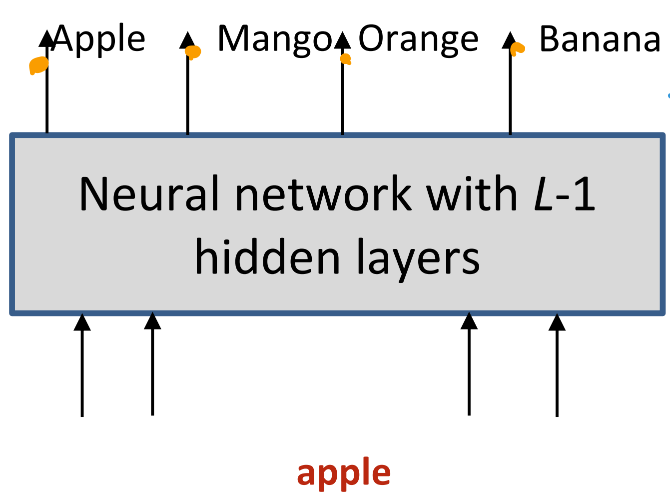
Output Activation Function

- Output activation function:
  - Sum of outputs should be 1
  - $\hat{y}$ should be a probability distribution
  - Sigmoid – probabilities will be $0<p<1$ but sum not equal to 1

$\sigma \rightarrow$ Softmax

$y_i = \{1 \quad 0 \quad 0 \quad 0\}$

$$\frac{e^{Tz_1}}{\sum_{i=1}^{k} e^{Tz_i}}$$



Classification problem

$\frac{e^z}{\Sigma}$

Output Activation Function

- Softmax function

  $z^L = b^L + W^L a^{L-1}$

  $\hat{y} = g(z^L_j) = e^z_j / \sum e^z_j$ for $j = 1..k$

  z^L_j is j^{th} element of z^L

- Example: $z^L = [10 \quad 20 \quad -30]$

  $\hat{y} = [e^{10}/(e^{10} + e^{20} + e^{-30}) \quad e^{20}/(e^{10} + e^{20} + e^{-30}) \quad e^{-30}/(e^{10} + e^{20} + e^{-30})\ ]$

NOTE: Exponent converts –ve values to +ve values

Loss function

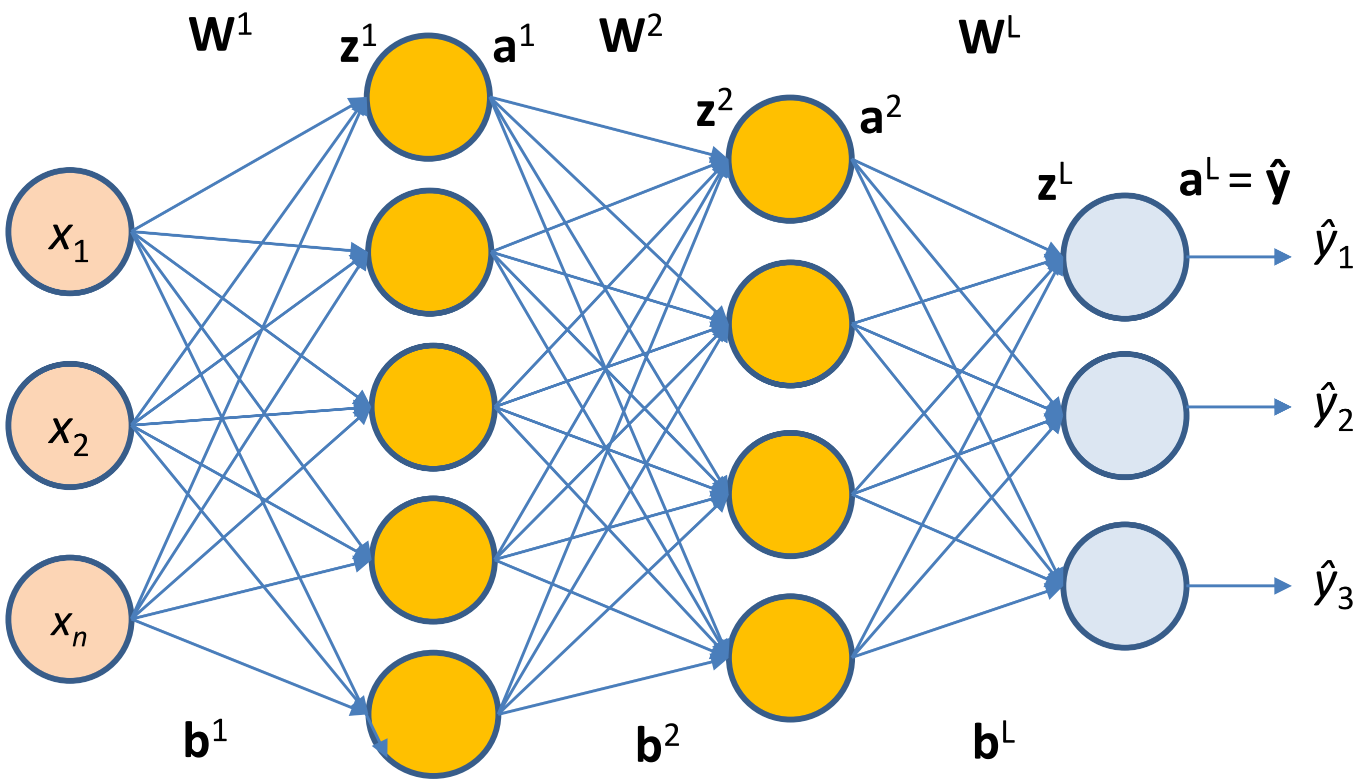
	Outputs	
	Real values	**Probabilities**
Output activation	Linear	Softmax
Loss function	Squared error	Cross-entropy

Backpropagation

How to compute $\nabla\theta$ composed of:

$\nabla W^1, \nabla W^2, \ldots \nabla W^{L-1} \in R^{n\times n}$, $\nabla W^L \in R^{n\times k}$

$\nabla b^1, \nabla b^2, \ldots \nabla b^{L-1} \in R^n$, $\nabla b^L \in R^k$

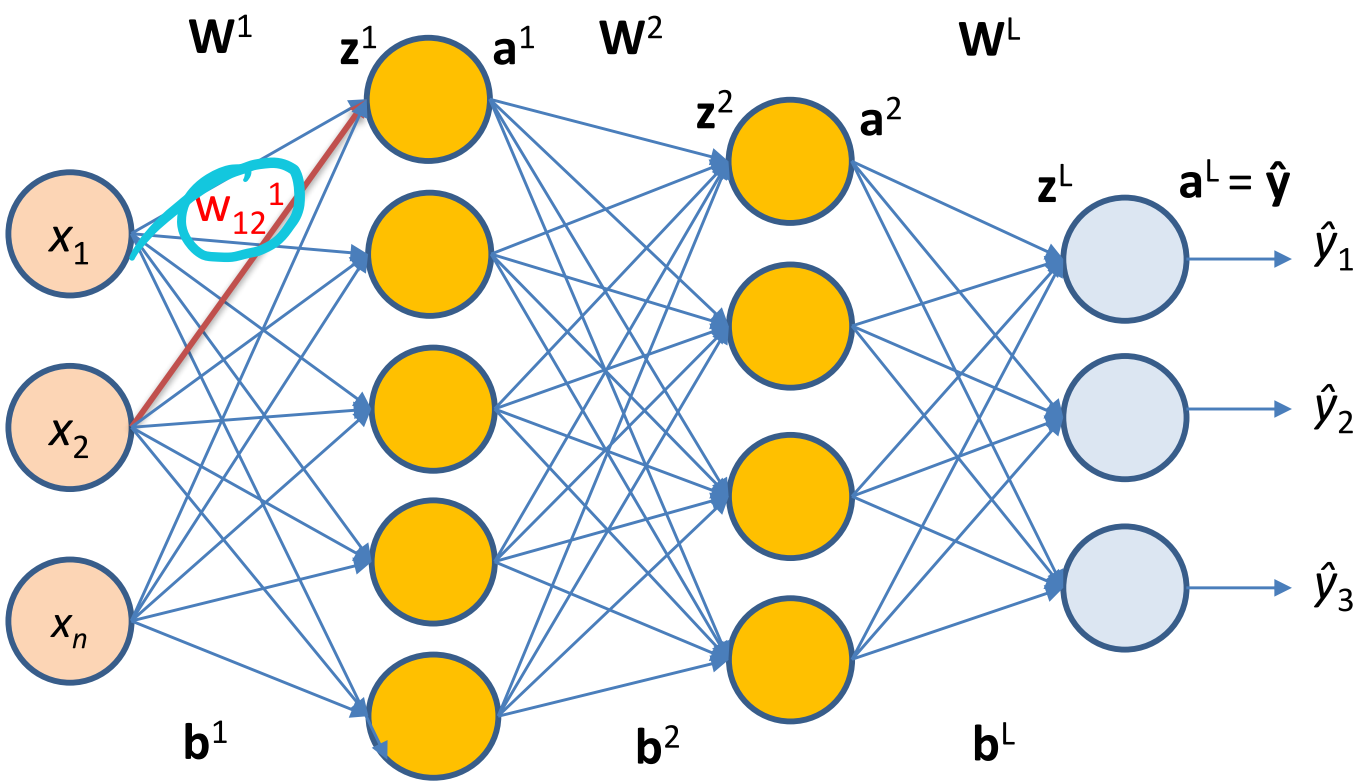


$$\frac{\partial L}{\partial w_{12}} = \frac{\partial L}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial z_L} \times \frac{\partial z_L}{\partial a^2} \times \frac{\partial a^2}{\partial z^2} \times \frac{\partial z^2}{\partial a^1} \times \frac{\partial a^1}{\partial z^1} \times \frac{\partial z_1}{\partial w_{12}}$$

Backpropagation

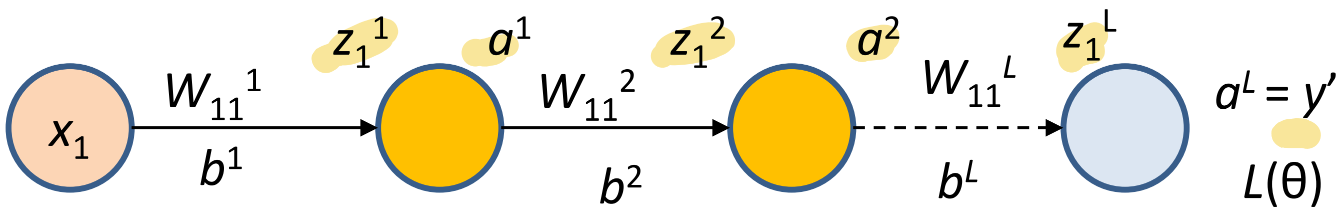
Assuming classification problem, $L(\theta) = -\log_2(\hat{y}_t)$

- To learn weight $w_{12}{}^1$ use SGD and compute $\frac{\partial L(w,b)}{\partial W_{12}}$



Backpropagation

Assume a deep thin network, who is responsible for the loss??



Find derivative by chain rule:

$$\frac{\partial L(\theta)}{\partial W_{11}^1} = \underbrace{\frac{\partial L(\theta)}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z_1^L}}_{\text{Output layer}} * \underbrace{\frac{\partial z_1^L}{\partial a_1^2} * \frac{\partial a_1^2}{\partial z_1^2}}_{\text{Previous hidden layer}} * \underbrace{\frac{\partial z_1^2}{\partial a_1^1} * \frac{\partial a_1^1}{\partial z_1^1}}_{\text{Previous hidden layer}} * \underbrace{\frac{\partial z_1^1}{\partial W_{11}^1}}_{\text{Weights}}$$

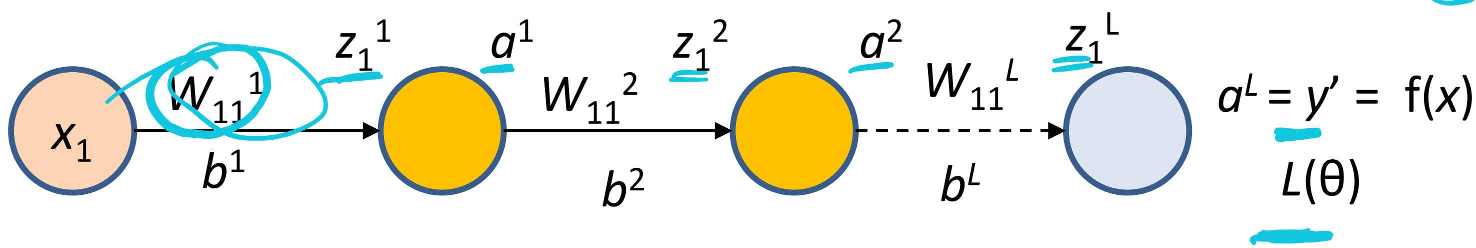
If we change W_{11}, how much does the loss change

Backpropagation

SSE

Cross Entropy $= -\sum y_i \log_2(\hat{y}_i)$

Assume a deep thin network



Find derivative by chain rule:

$$\frac{\partial L(\theta)}{\partial W_{11}^{1}} = \frac{\partial L(\theta)}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z_1^{L}} * \frac{\partial z_1^{L}}{\partial a_1^{2}} * \frac{\partial a_1^{2}}{\partial z_1^{2}} * \frac{\partial z_1^{2}}{\partial a_1^{1}} * \frac{\partial a_1^{1}}{\partial z_1^{1}} * \frac{\partial z_1^{1}}{\partial W_{11}^{1}}$$

$$\frac{\partial L(\theta)}{\partial W_{11}^{2}} = \frac{\partial L(\theta)}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z_1^{L}} * \frac{\partial z_1^{L}}{\partial a_1^{2}} * \frac{\partial a_1^{2}}{\partial z_1^{2}} * \frac{\partial z_1^{2}}{\partial W_{11}^{2}}$$

$$\frac{\partial L(\theta)}{\partial W_{11}^{L}} = \frac{\partial L(\theta)}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z_1^{L}} * \frac{\partial z_1^{L}}{\partial W_{11}^{L}}$$

$$L(w,b) = -y \log \hat{y} - (1-y) \log(1-\hat{y})$$

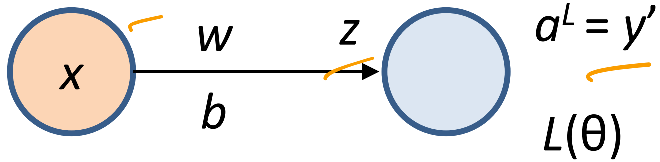
$$\frac{\partial L}{\partial w} = \left[\frac{-y}{\hat{y}} + \frac{(1-y)}{(1-\hat{y})} \right] \hat{y}(1-\hat{y})\, x$$

$$\frac{\partial \hat{y}}{\partial z} \times \partial$$

$$= (\hat{y} - y) \; - - -$$

BACKPROPAGATION WITH SIGMOID OUTPUT ACTIVATION & BINARY CROSS-ENTROPY LOSS

Backpropagation



Assuming binary cross-entropy function

L = -y log ŷ - (1 - y) log (1-$\hat{y}$)

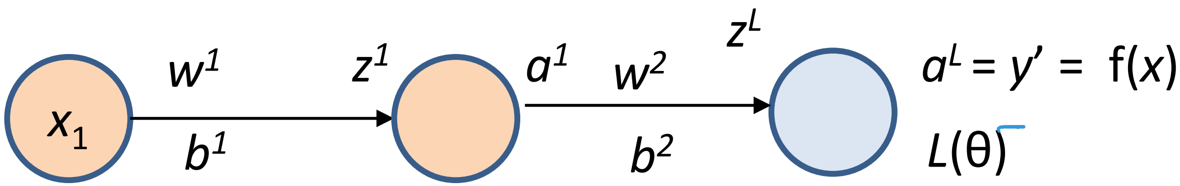
Loss → Cross entropy

Activation → sigmoid

$$\frac{\partial L(\theta)}{\partial \hat{y}} = \frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}$$

$$\frac{\partial L(\theta)}{\partial z} = \frac{\partial L(\theta)}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z} = \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right) * \hat{y}(1-\hat{y}) = \hat{y} - y$$

$$\frac{\partial L(\theta)}{\partial w} = \frac{\partial L(\theta)}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z} * \frac{\partial \hat{z}}{\partial w} = (\hat{y} - y) * x$$



$z^2 = w^2 a_1 + b$ $\quad$ $a^1 = \frac{1}{1+e^{-z^1}}$

$$\frac{\partial L(\theta)}{\partial \hat{y}} = \frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}$$

$$\frac{\partial L(\theta)}{\partial z^2} = \frac{\partial L(\theta)}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z^2} = \left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right) * \hat{y}(1-\hat{y}) = \hat{y} - y \qquad : \delta^L = \frac{\partial L}{\partial \hat{y}} * \sigma'(z^L)$$

$$\frac{\partial L(\theta)}{\partial w^2} = \frac{\partial L(\theta)}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z^2} * \frac{\partial z^2}{\partial w^2} = (\hat{y} - y) * a^1 \qquad : \delta^L * a^1$$

$\hat{y} = \frac{1}{1+e^{-z_2}}$ $\quad$ $z^1 = w_1 a_1 + b_1$

$$\frac{\partial L(\theta)}{\partial a^1} = \frac{\partial L(\theta)}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z^2} * \frac{\partial z^2}{\partial a^1} = (\hat{y} - y) * w^2 \qquad : \delta^L * w^2$$

$$\frac{\partial L(\theta)}{\partial z^1} = \frac{\partial L(\theta)}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z^2} * \frac{\partial z^2}{\partial a^1} * \frac{\partial a^1}{\partial z^1} = (\hat{y} - y) * w^2 * a^1(1-a^1) \qquad : \delta^1 = \delta^L w^2 * \sigma'(z^1)$$

$$\frac{\partial L(\theta)}{\partial w^1} = \frac{\partial L(\theta)}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z^2} * \frac{\partial z^2}{\partial a^1} * \frac{\partial a^1}{\partial z^1} * \frac{\partial z^1}{\partial w^1} = (\hat{y} - y) * w^2 * a^1(1-a^1) * x \qquad : \delta^1 * x$$

$w2$

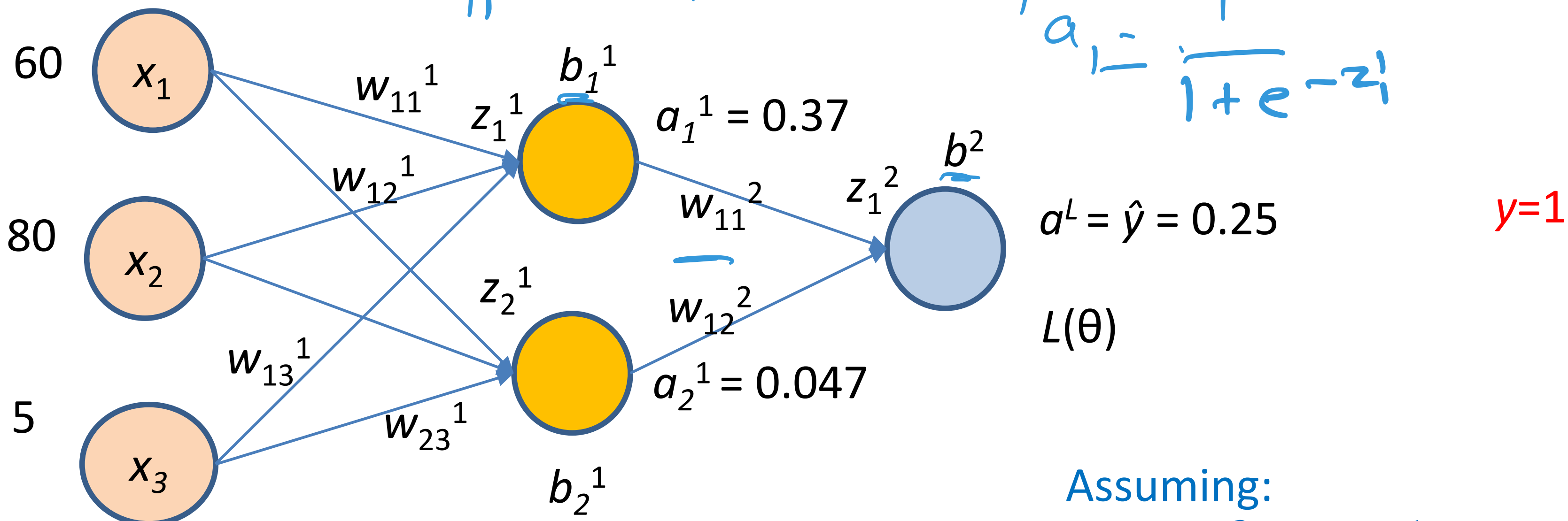
$z^2 = w^2 a_1 + b^2$

Backpropagation Equations

- $\delta^L = \frac{\partial L}{\partial \hat{y}} \odot \sigma'(z^L)$
- $\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot \sigma'(z^l)$
- $\frac{\partial L}{\partial b_j^l} = \delta_j^l$
- $\frac{\partial L}{\partial w_{jk}^l} = a_k^{l-1} \delta_j^l$

$z_1^1 = w_{11}^1 x_1 + w_{12}^1 x_2 + b_1^1$

$a_1^1 = \frac{1}{1+e^{-z_1^1}}$



Assuming:
Initial w_{11}^2=12, w_{11}^1=0.1
η = 0.01

During forward propagation:
a_1^1 = 0.37, z_1^1 = 0.5

$$\frac{\partial L}{\partial w_{11}^2} = \frac{\partial L}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z_1^2} * \frac{\partial z_1^2}{\partial w_{11}^2}$$

$$\frac{\partial L}{\partial \hat{y}} \times \frac{\partial \hat{y}}{\partial z_1^2} \times \frac{\partial z_1^2}{\partial w_{11}^2}$$

$$\frac{\partial L(\theta)}{\partial \hat{y}} = \frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}} = \frac{-1}{0.25} = -4$$

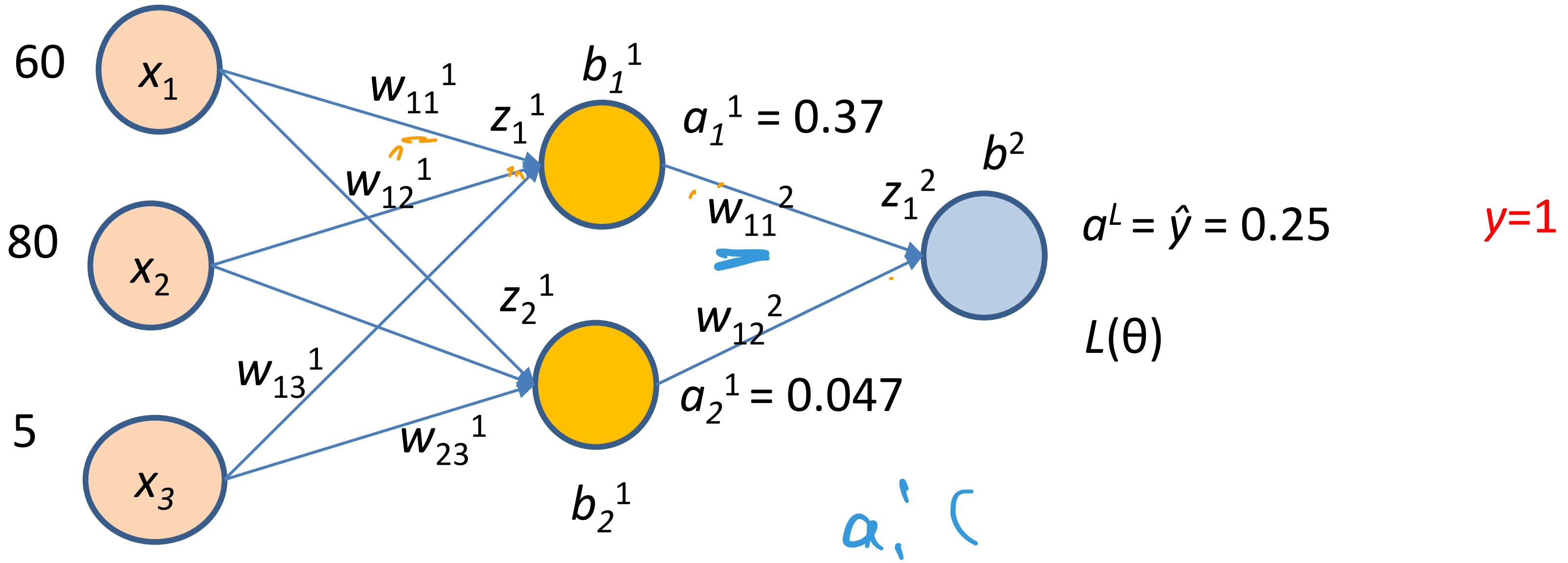
$$\left(\frac{-y}{\hat{y}} + \frac{1-y}{1-\hat{y}}\right) \hat{y}(1-\hat{y})$$

$$\times a_1^1$$

$$\frac{\partial L(\theta)}{\partial z_1^2} = \frac{\partial L(\theta)}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z_1^2} = \hat{y} - y = 0.25 - 1 = -0.75$$

$$\frac{\partial L(\theta)}{\partial w_{11}^2} = \frac{\partial L(\theta)}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z_1^2} * \frac{\partial z}{\partial w_{11}^2} = -0.75 * a_1^1 = -0.75 * 0.37 = -0.2775$$

$$w_{11}^{2*} = w_{11}^2 - \eta * \frac{\partial L}{\partial w_{11}^2} = 12 - 0.01 * (-0.2775) = 12.0028$$



$$\frac{\partial L}{\partial w_{11}^{1}} = \frac{\partial L}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z_1^2} * \frac{\partial z_1^2}{\partial a_1^1} * \frac{\partial a_1^1}{\partial z_1^1} * \frac{\partial z_1^1}{\partial w_{11}^{1}}$$

$$= -0.75 * w_{11}^{2} * a_1^1(1 - a_1^1) * x_1$$

= -0.75 * 12 * 0.37 (1- 0.37)*60

= -125.874

$$w_{11}^{1*} = w_{11}^{1} - \eta * \frac{\partial L}{\partial w_{11}^{1}} = 0.1 - 0.01 * (-125.874) = 1.35$$

Assignment: Compute $\frac{\partial L}{\partial w_{13}^{1}}$

BACKPROPAGATION WITH SOFTMAX OUTPUT ACTIVATION & CROSS-ENTROPY LOSS

Backpropagation

- Computing gradients
  - Gradients w.r.t. output units
  - Gradients w.r.t. hidden units
  - Gradients w.r.t. weights and biases

$$\frac{\partial L(\theta)}{\partial W_{11}^1} = \frac{\partial L(\theta)}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial z_1^L} * \frac{\partial z_1^L}{\partial a_1^2} * \frac{\partial a_1^2}{\partial z_1^2} * \frac{\partial z_1^2}{\partial a_1^1} * \frac{\partial a_1^1}{\partial z_1^1} * \frac{\partial z_1^1}{\partial W_{11}^1}$$

Gradients w.r.t. output units

- Assuming softmax activation and cross entropy loss at output layer for *k* classes:

$L(\theta) = -\log_2(\hat{y}_t)$ t: true class label

$$\frac{\partial L(\theta)}{\partial \hat{y}_i} = \frac{\partial(-log\hat{y}_t)}{\partial \hat{y}_i} \quad \text{for } i = 1..k$$

$$= -\frac{1}{\hat{y}_t} \quad \text{if } i = t$$

$$= 0 \quad \text{otherwise}$$

Gradients w.r.t. output units

$$\frac{\partial L(\theta)}{\partial \hat{y}_i} = -\frac{1}{\hat{y}_t} \quad \text{if } i = t$$

$$= 0 \quad \text{otherwise}$$

$$\frac{\partial L(\theta)}{\partial \hat{y}} = \begin{pmatrix} \frac{\partial L(\theta)}{\partial \hat{y}_1} \\ \frac{\partial L(\theta)}{\partial \hat{y}_2} \\ \cdots\cdot \\ \frac{\partial L(\theta)}{\partial \hat{y}_k} \end{pmatrix} = -\frac{1}{\hat{y}_t} * \begin{pmatrix} 1 \text{ if } t=1 \\ 1 \text{ if } t=2 \\ \cdots \\ 1 \text{ if } t=k \end{pmatrix}$$

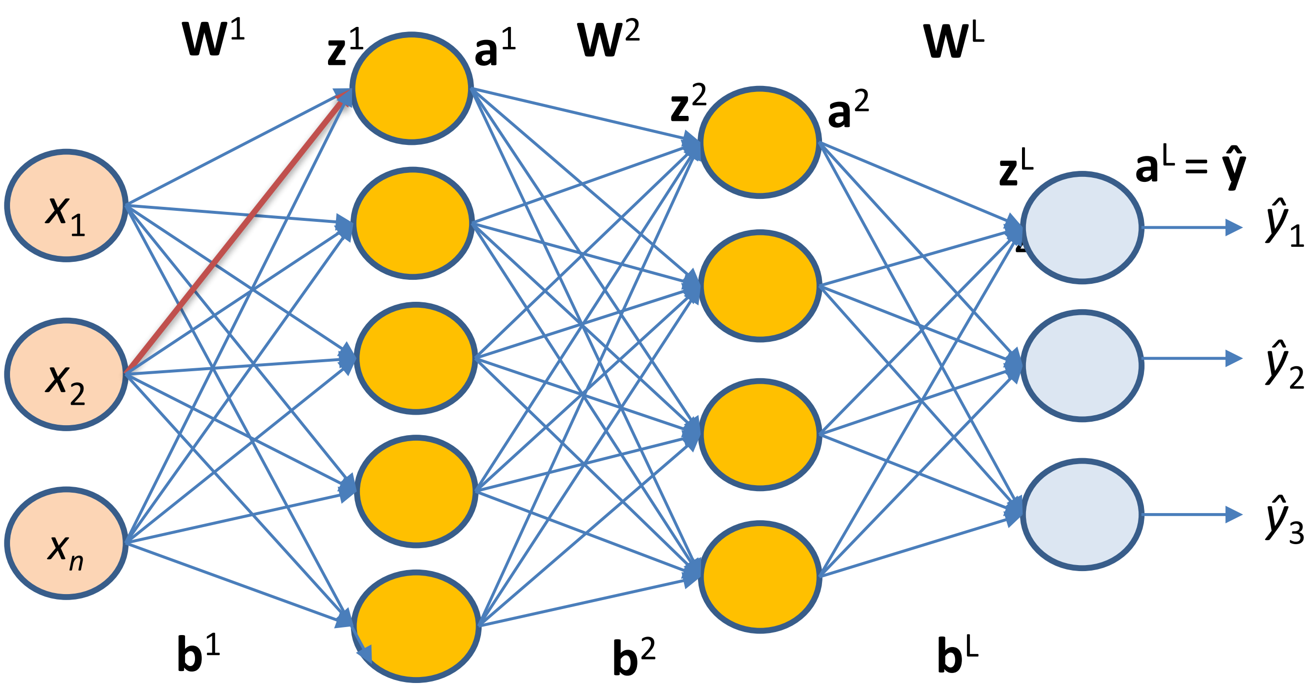
$$= -\frac{1}{\hat{y}_t} * e(t)$$

$e(t)$: One hot k-dimensional vector whose t^{th} entry is 1, others are 0

Gradients w.r.t. output units

$$\frac{\partial L(\theta)}{\partial z_i^L} = \frac{\partial(-\log \hat{y}_t)}{\partial z_i^L} = -\frac{\partial(-\log \hat{y}_t)}{\partial \hat{y}_t} * \frac{\partial \hat{y}_t}{\partial z_i^L} = -\frac{1}{\hat{y}_t} * \frac{\partial \hat{y}_t}{\partial z_i^L}$$

$$\hat{y}_t = \frac{\exp(z_t^L)}{\sum \exp(z_i^L)} \longrightarrow$$ $\hat{y}_t$ depends on $z^L{}_i$



Gradients w.r.t. output units

$$\frac{\partial(-\log \hat{y}_t)}{\partial z_i^L} = -\frac{1}{\hat{y}_t} * \frac{\partial \hat{y}_t}{\partial z_i^L}$$

$z^L = [z_1{}^L \quad z_2{}^L \;\; .... \;\; z_k{}^L]$

$\hat{y}$ = softmax(z^L) = $[\hat{y}_1{}^L \quad \hat{y}_2{}^L \;\; .... \; \hat{y}_t{}^L \;\; ... \; \hat{y}_k{}^L]$

$\hat{y}_t$: t^{th} entry of $\hat{y}$ = softmax($z_t{}^L$) = $\frac{\exp(z_t^L)}{\sum \exp(z_i^L)}$

$$\frac{\partial(-\log \hat{y}_t)}{\partial z_i^L} = -\frac{1}{\hat{y}_t} * \frac{\partial \hat{y}_t}{\partial z_i^L}$$

$$= -\frac{1}{\hat{y}_t} * \frac{\partial(softmax(z_t^L)}{\partial z_i^L}$$

Gradients w.r.t. output units

$$\frac{\partial(-\log \hat{y}_t)}{\partial z_i^L} = -\frac{1}{\hat{y}_t} * \frac{\partial(softmax(z_t^L))}{\partial z_i^L} = -\frac{1}{\hat{y}_t} * \frac{\partial}{\partial z_i^L}\left(\frac{\exp(z_t^L)}{\sum \exp(z_i^L)}\right)$$

$$\frac{\partial(-\log \hat{y}_t)}{\partial z_i^L} = -\frac{1}{\hat{y}_t} * \frac{\frac{\partial\left(\exp(z_t^L)\right)}{\partial z_i^L}}{\sum \exp(z_i^L)} - \frac{\exp(z_t^L) * \frac{\partial\left(\sum \exp(z_i^L)\right)}{\partial z_i^L}}{(\sum \exp(z_i^L))^2}$$

$$= -\frac{1}{\hat{y}_t} * \left(\frac{1_{(t=i)} \exp\left(z_t^L\right)}{\sum \exp\left(z_i^L\right)} - \frac{\exp\left(z_t^L\right) * \exp(z_i^L)}{(\sum \exp(z_i^L))^2}\right)$$

$$= -\frac{1}{\hat{y}_t} * \left(1_{(t=i)} \text{softmax}\,(z_t^L) - \text{softmax}\,(z_t^L) * \text{softmax}\,(z_i^L)\right)$$

$$= -\frac{1}{\hat{y}_t} * \left(1_{(t=i)} \hat{y}_t - \hat{y}_t\, \hat{y}_i\right) = -\left(1_{(t=i)} - \hat{y}_i\right)$$

$$\frac{\partial}{\partial x}\frac{f(x)}{g(x)} = \frac{g(x) * \frac{\partial(f(x))}{\partial x} - f(x) * \frac{\partial(g(x))}{\partial x}}{(g(x))^2} = \frac{\frac{\partial(f(x))}{\partial x}}{g(x)} - \frac{f(x) * \frac{\partial(g(x))}{\partial x}}{(g(x))^2}$$

Gradients w.r.t. output units

$$\frac{\partial L(\theta)}{\partial z_i^L} = -\left(1_{(t=i)} - \hat{y}_i\right)$$

Gradient w.r.t. vector $\mathbf{z}^L$:

$$\frac{\partial L(\theta)}{\partial z^L} = \begin{pmatrix} \frac{\partial L(\theta)}{\partial z_1^L} \\ \frac{\partial L(\theta)}{\partial z_2^L} \\ \cdots \\ \frac{\partial L(\theta)}{\partial z_k^L} \end{pmatrix} = \begin{pmatrix} -\left(1_{(t=1)} - \hat{y}_1\right) \\ -\left(1_{(t=2)} - \hat{y}_2\right) \\ \cdots \\ -\left(1_{(t=k)} - \hat{y}_k\right) \end{pmatrix} = -(e(t) - \hat{y})$$

$e(t)$: One hot k-dimensional vector whose t^{th} entry is 1, others are 0

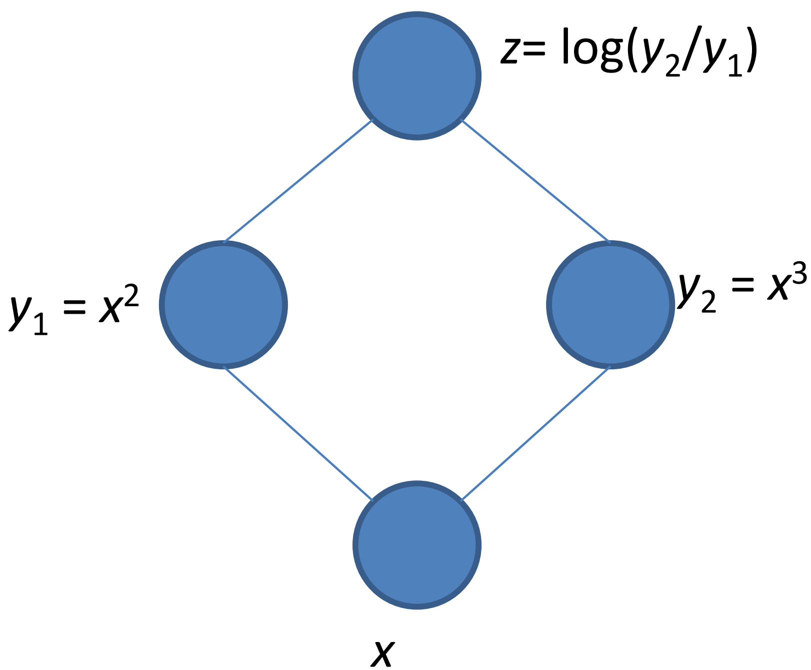
Gradients w.r.t. hidden units

Example

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y_1} * \frac{\partial y_1}{\partial x} + \frac{\partial z}{\partial y_2} * \frac{\partial y_2}{\partial x}$$

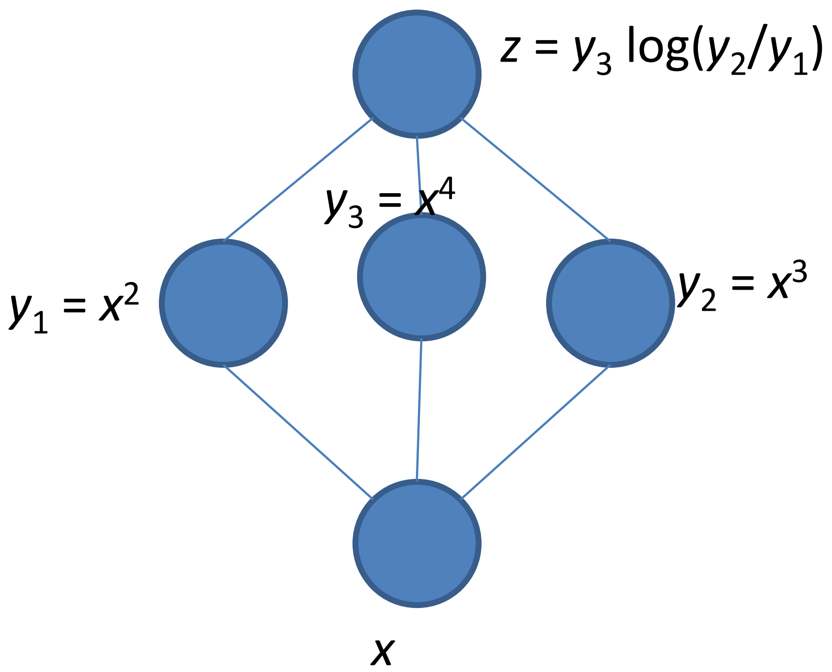
$$= \sum \frac{\partial z}{\partial y_i} * \frac{\partial y_i}{\partial x}$$

for $i = 1,2$



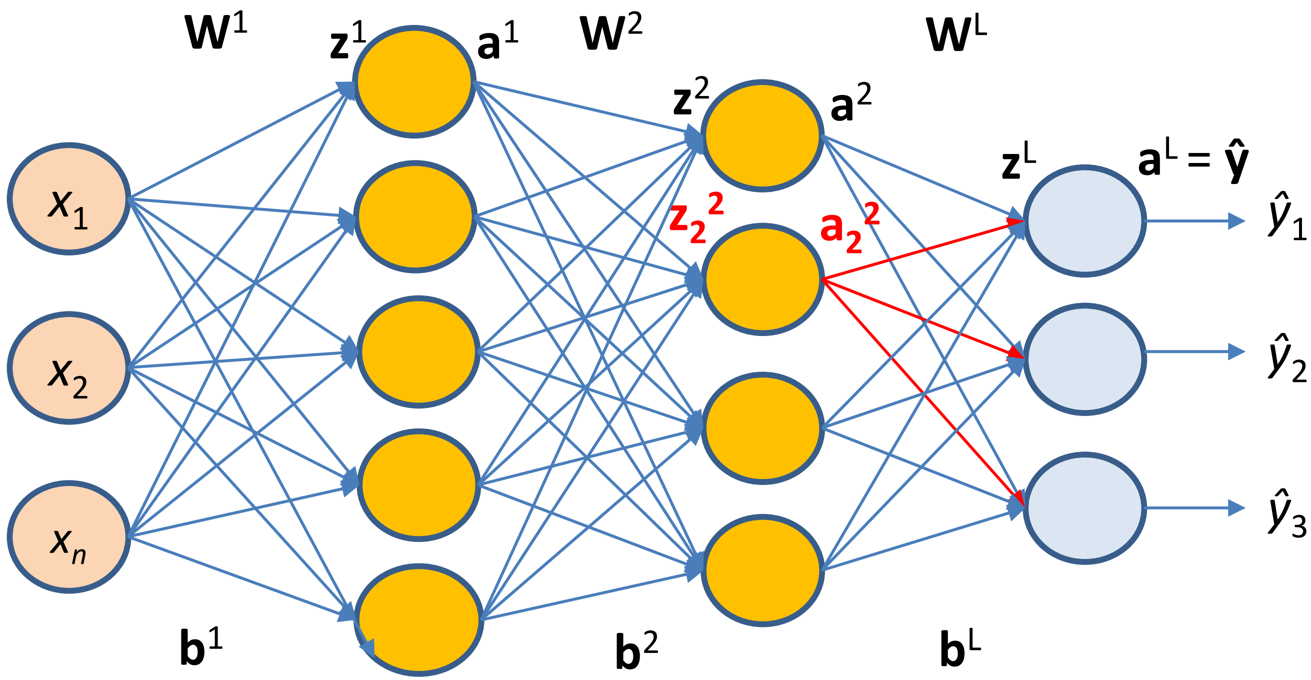
Gradients w.r.t. hidden units

Example $$\frac{\partial z}{\partial x} = \sum \frac{\partial z}{\partial y_i} * \frac{\partial y_i}{\partial x}$$ for i = 1,2,3



Chain rule across multiple paths

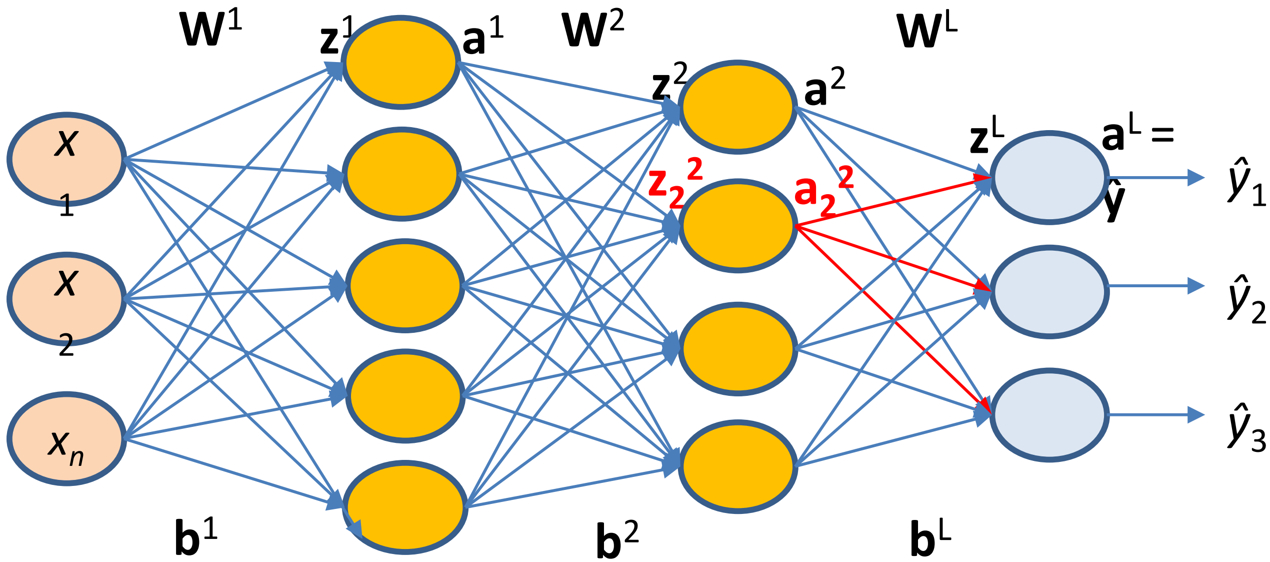
Gradients w.r.t. hidden units



$$z^{l+1} = W^{l+1} a_j^l + b^{l+1}$$

$$\frac{\partial L(\theta)}{\partial a_j^l} = \sum_{p=1}^{k} \frac{\partial L(\theta)}{\partial z_p^{l+1}} * \frac{\partial z_p^{l+1}}{\partial a_j^l}$$

k possible paths from output to hidden unit



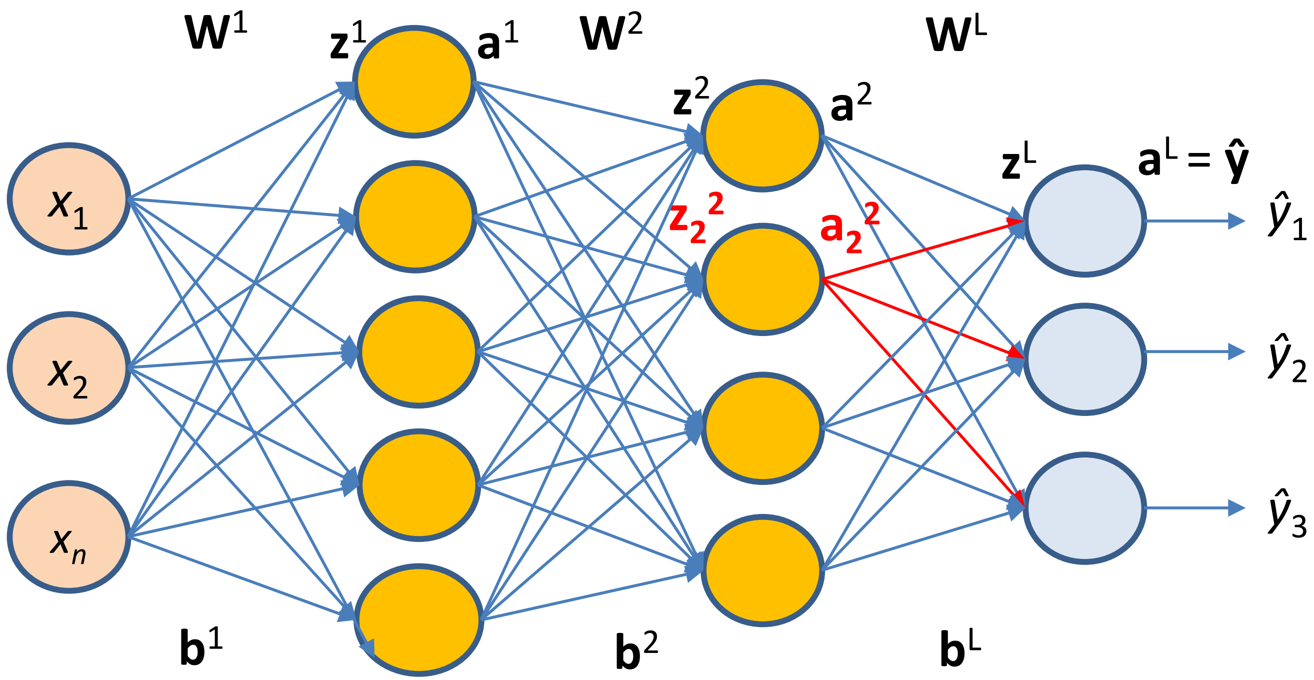
z^{l+1}=$W^{l+1}a_j^l + b^{l+1}$

$$\begin{pmatrix} z^L{}_1 \\ z^L{}_2 \\ z^L{}_3 \end{pmatrix} \begin{matrix} = \\ = \\ = \end{matrix} \begin{pmatrix} w^L{}_{11} & w^L{}_{12} & w^L{}_{13} & w^L{}_{14} \\ w^L{}_{21} & w^L{}_{22} & w^L{}_{23} & w^L{}_{24} \\ w^L{}_{31} & w^L{}_{32} & w^L{}_{33} & w^L{}_{34} \end{pmatrix} \begin{pmatrix} a^2{}_1 \\ a^2{}_2 \\ a^2{}_3 \\ a^2{}_4 \end{pmatrix} + \begin{pmatrix} b^L{}_1 \\ b^L{}_2 \\ b^L{}_3 \end{pmatrix}$$

$$z_1^L = W_{11}^L a_1^2 + W_{12}^L a_2^2 + W_{13}^L a_3^2 + W_{14}^L a_4^2 + b_1^L$$

$$\frac{\partial z_1^L}{\partial a_2^2} = w_{12}^L$$

Gradients w.r.t. hidden units



$z^{l+1} = W^{l+1} a_j^l + b^{l+1}$

$$\frac{\partial L(\theta)}{\partial a_j^l} = \sum_{p=1}^{k} \frac{\partial L(\theta)}{\partial z_p^{l+1}} * \frac{\partial z_p^{l+1}}{\partial a_j^l}$$

$$\frac{\partial L(\theta)}{\partial a_j^l} = \sum_{p=1}^{k} \frac{\partial L(\theta)}{\partial z_p^{l+1}} * W_{p,j}^{l+1}$$

Gradients w.r.t. hidden units

$$\frac{\partial L(\theta)}{\partial a_j^l} = \sum_{p=1}^{k} \frac{\partial L(\theta)}{\partial z_p^{l+1}} * \frac{\partial z_p^{l+1}}{\partial a_j^l}$$

$$\frac{\partial L(\theta)}{\partial a_j^l} = \sum_{p=1}^{k} \frac{\partial L(\theta)}{\partial z_p^{l+1}} * W_{p,j}^{l+1}$$

1 Vision

2 Unsuper

3 Adv. Analytic

$$\frac{\partial L(\theta)}{\partial z^{l+1}} = \begin{pmatrix} \frac{\partial L(\theta)}{\partial z_1^{l+1}} \\ \frac{\partial L(\theta)}{\partial z_2^{l+1}} \\ \dots \\ \frac{\partial L(\theta)}{\partial z_k^{l+1}} \end{pmatrix} \qquad W_{.,j}^{l+1} = \begin{pmatrix} W_{1,j}^{l+1} \\ W_{2,j}^{l+1} \\ \dots \\ W_{k,j}^{l+1} \end{pmatrix}$$

$W_{.,j}^{l+1}$ is the j^{th} column of W^{l+1}

$$(W_{.,j}^{l+1})^{\mathsf{T}} \frac{\partial L(\theta)}{\partial z^{l+1}} = \sum_{p=1}^{k} \frac{\partial L(\theta)}{\partial z_p^{l+1}} * W_{p,j}^{l+1}$$

Gradients w.r.t. hidden units

$$\frac{\partial L(\theta)}{\partial a_j^l} = (W_{\cdot,j}^{l+1})^\top \frac{\partial L(\theta)}{\partial z^{l+1}}$$

$$\frac{\partial L(\theta)}{\partial a_j^l} = \begin{pmatrix} \frac{\partial L(\theta)}{\partial a_1^l} \\ \frac{\partial L(\theta)}{\partial a_2^l} \\ \cdots \\ \frac{\partial L(\theta)}{\partial a_n^l} \end{pmatrix} = \begin{pmatrix} (W_{\cdot,1}^{l+1})^\top \frac{\partial L(\theta)}{\partial z^{l+1}} \\ (W_{\cdot,2}^{l+1})^\top \frac{\partial L(\theta)}{\partial z^{l+1}} \\ \cdots \\ (W_{\cdot,n}^{l+1})^\top \frac{\partial L(\theta)}{\partial z^{l+1}} \end{pmatrix}$$

$$\frac{\partial L(\theta)}{\partial a_j^l} = (W^{l+1})^\top \frac{\partial L(\theta)}{\partial z^{l+1}}$$

How to compute $\frac{\partial L(\theta)}{\partial z^{l+1}}$ for $l \leq L$-1

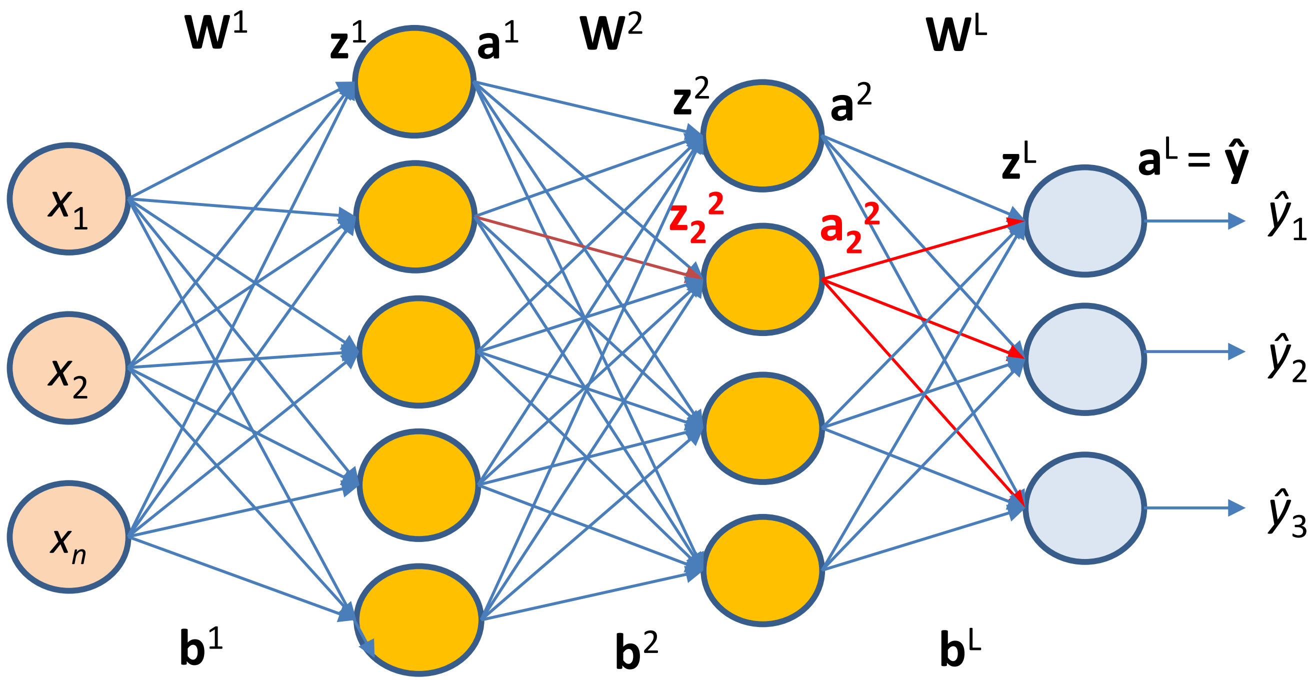
Gradients w.r.t. hidden units

$$\frac{\partial L(\theta)}{\partial z^l} = \begin{pmatrix} \frac{\partial L(\theta)}{\partial z_1^l} \\ \frac{\partial L(\theta)}{\partial z_2^l} \\ \cdots \\ \frac{\partial L(\theta)}{\partial z_n^l} \end{pmatrix}$$

$$\frac{\partial L(\theta)}{\partial z_j^l} = \frac{\partial L(\theta)}{\partial a_j^l} * \frac{\partial a_j^l}{\partial z_j^l} = \frac{\partial L(\theta)}{\partial a_j^l} * \frac{\partial g(z_j^l)}{\partial z_j^l}$$

$$\frac{\partial L(\theta)}{\partial z^l} = \begin{pmatrix} \frac{\partial L(\theta)}{\partial a_1^l} * \frac{\partial g(z_1^l)}{\partial z_1^l} \\ \frac{\partial L(\theta)}{\partial a_2^l} * \frac{\partial g(z_2^l)}{\partial z_2^l} \\ \cdots \\ \frac{\partial L(\theta)}{\partial a_n^l} * \frac{\partial g(z_n^l)}{\partial z_n^l} \end{pmatrix} = \frac{\partial L(\theta)}{\partial a_j^l} \odot [\ldots., \ldots.., \frac{\partial g(z_j^l)}{\partial z_j^l}, \ldots.]$$

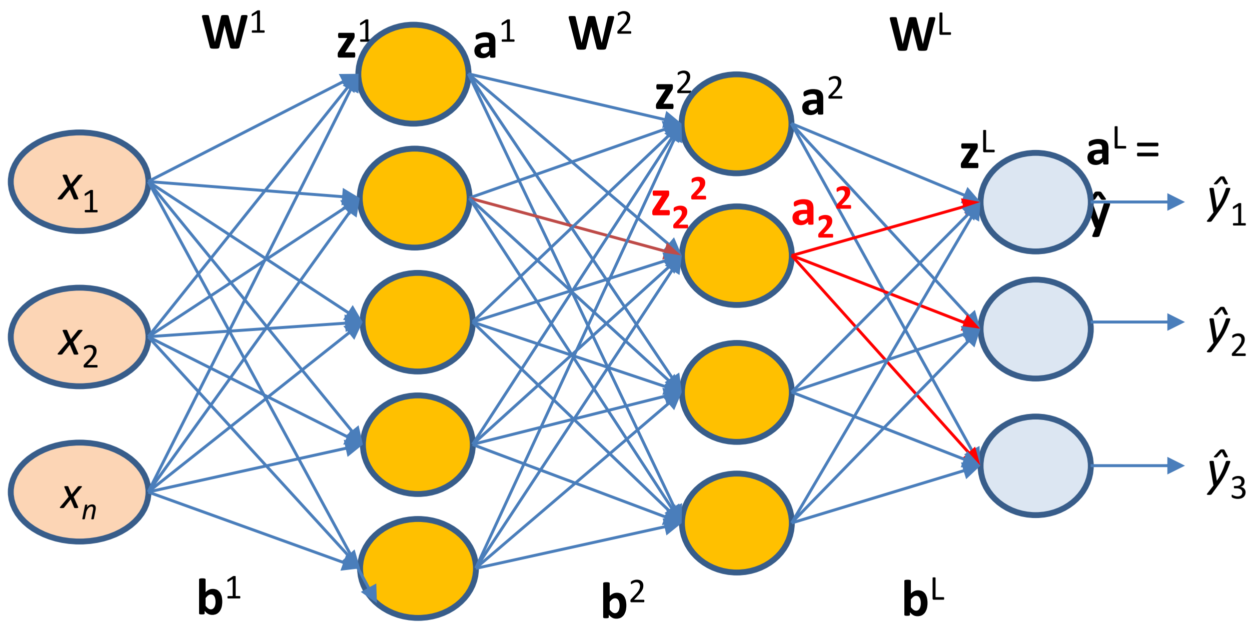
Gradients w.r.t. Parameters



$$\frac{\partial L(\theta)}{\partial w^l} = \frac{\partial L(\theta)}{\partial z^l} * \frac{\partial z^l}{\partial w^l}$$

$$z^l = W^l \, a^{l-1} + b^l$$

Gradients w.r.t. Parameters



$$\frac{\partial L(\theta)}{\partial w^l} = \frac{\partial L(\theta)}{\partial z^l} * \frac{\partial z^l}{\partial w^l}$$

$$z^l = W^l\, a^{l-1} + b^l$$

$$\begin{pmatrix} z_1^l \\ z_2^l \\ \dots \\ z_n^l \end{pmatrix} = \begin{pmatrix} w_{11}^l & w_{12}^l & \dots & w_{1n}^l \\ w_{21}^l & w_{22}^l & \dots & w_{2n}^l \\ & \dots & & \\ w_{n1}^l & w_{n2}^l & \dots & w_{nn}^l \end{pmatrix} \begin{pmatrix} a_1^{l-1} \\ a_2^{l-1} \\ \\ a_n^{l-1} \end{pmatrix} + \begin{pmatrix} b_1^{l-1} \\ b_2^{l-1} \\ \\ b_n^{l-1} \end{pmatrix}$$

Gradients w.r.t. Parameters

$$\frac{\partial L(\theta)}{\partial w^l} = \frac{\partial L(\theta)}{\partial z^l} * \frac{\partial z^l}{\partial w^l}$$

$$\begin{pmatrix} z_1^l \\ z_2^l \\ \cdots\cdot \\ z_n^l \end{pmatrix} = \begin{pmatrix} w_{11}^l & w_{12}^l & \dots & w_{1n}^l \\ w_{21}^l & w_{22}^l & \dots & w_{2n}^l \\ & \dots & & \\ w_{n1}^l & w_{n2}^l & \dots & w_{nn}^l \end{pmatrix} \begin{pmatrix} a_1^{l-1} \\ a_2^{l-1} \\ \\ a_n^{l-1} \end{pmatrix} + \begin{pmatrix} b_1^{l-1} \\ b_2^{l-1} \\ \\ b_n^{l-1} \end{pmatrix}$$

$$\frac{\partial z^l}{\partial w_{12}^l} = a_2^{l-1} \qquad \frac{\partial z^l}{\partial w_{pq}^l} = a_q^{l-1}$$

$$\frac{\partial L(\theta)}{\partial w_{pq}^l} = \frac{\partial L(\theta)}{\partial z^l} * a_q^{l-1}$$

Gradients w.r.t. Parameters

$$\frac{\partial L(\theta)}{\partial w_{pq}^{l}} = \frac{\partial L(\theta)}{\partial z^{l}} * a_q^{l-1}$$

$$\frac{\partial L(\theta)}{\partial w^{l}} = \begin{pmatrix} \frac{\partial L(\theta)}{\partial w_{00}^{l}} & \frac{\partial L(\theta)}{\partial w_{01}^{l}} & \cdots & \frac{\partial L(\theta)}{\partial w_{0n-1}^{l}} \\ \cdots & \cdots & \cdots & .. \\ \cdots & \cdots & & \frac{\partial L(\theta)}{\partial w_{n-1\,n-1}^{l}} \end{pmatrix}$$

Backpropagation Equations

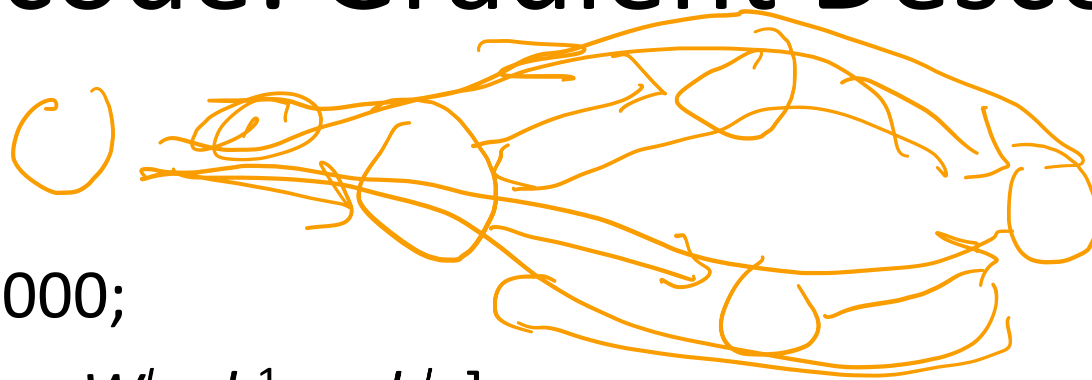
$$\frac{\partial L}{\partial z^l} = (w^{l+1})^T . \frac{\partial L^l}{\partial z^{l+1}} * \sigma'(z^l)$$

$$\frac{\partial L}{\partial w^l} = \frac{\partial L^l}{\partial z^l} . \left(a^{l-1}\right)^T$$

$$\frac{\partial L}{\partial b^l} = \frac{\partial L^l}{\partial z^l}$$

* - element-wise multiplication
 . - dot product
l – layer number
(assuming sigmoid activations)

Pseudo code: Gradient Descent



t:=0;

max_iterations:=1000;

Initialize $\theta_0 := [W^1{}_0, \ldots W^L{}_0, b^1{}_0 \ldots b^L{}_0]$;

while t++ < *max_iterations* **do**

$a^1, a^2 \ldots a^{L-1}, z^1, z^2 \ldots z^L, \hat{y} = forward\_propagation(\theta_t)$;

$\nabla\theta_t$ = back_propagation($a^1, a^2 \ldots a^{L-1}, z^1, z^2 \ldots z^L, y, \hat{y}$);

$\theta_{t+1} := \theta_t - \eta\nabla\theta_t$;

end

Pseudo code: Forward Propagation

for v = 0 to L-1 **do**

$z^v = b^v + W^v a^{v-1}$;

$a^v = g(z^v)$;

end;

$z^L = b^L + W^L a^{L-1}$;

$\hat{y} = O(z^L)$;

Do a forward propagation and compute all a^i's, z^i's, and $\hat{y}$

Pseudo code: Back Propagation

//Compute output gradient:

$\nabla_{z^L} L(\theta) = -(e(y) - \hat{y})$;

for $v = L$ to 1 **do**

//Compute gradients w.r.t. parameters

$\nabla_{W^v} L(\theta) = \nabla_{z^v} L(\theta)\, a^{v-1}$;

$\nabla_{b^v} L(\theta) = \nabla_{z^v} L(\theta)$;

//Compute gradients w.r.t. layer below

$\nabla_{a^{v-1}} L(\theta) = W^v \nabla_{z^v} L(\theta)$;

//Compute gradients w.r.t. layer below (pre-activation)

$\nabla_{z^{v-1}} L(\theta) = \nabla_{a^{v-1}} L(\theta) \odot [\ldots\ldots g'(z^{v-1,j}) \ldots\ldots]$;

end