CoE202
Fundamentals of Artificial intelligence
<Big Data Analysis and Machine Learning>

Backpropagation in Neural Network

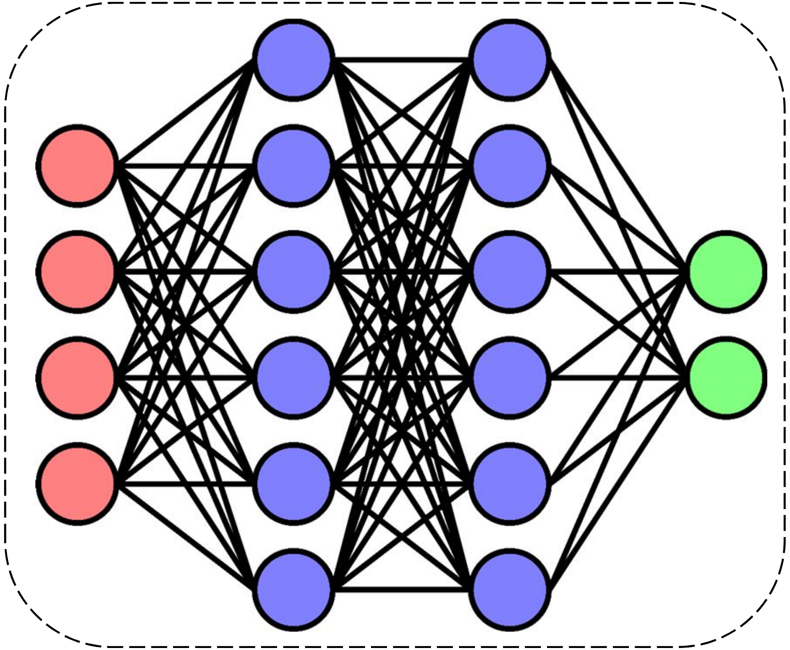
Prof. Young-Gyu Yoon
School of EE, KAIST

Contents

- Recap
  - Neural network
  - Single layer in a neural network
  - Activation function (Sigmoid & ReLU)
  - Universal approximation theorem

- Gradient descent for neural network
  - Forward propagation
  - Backward propagation

Revisit: The "Neural Network"



$f : X \to Y$

For a data set

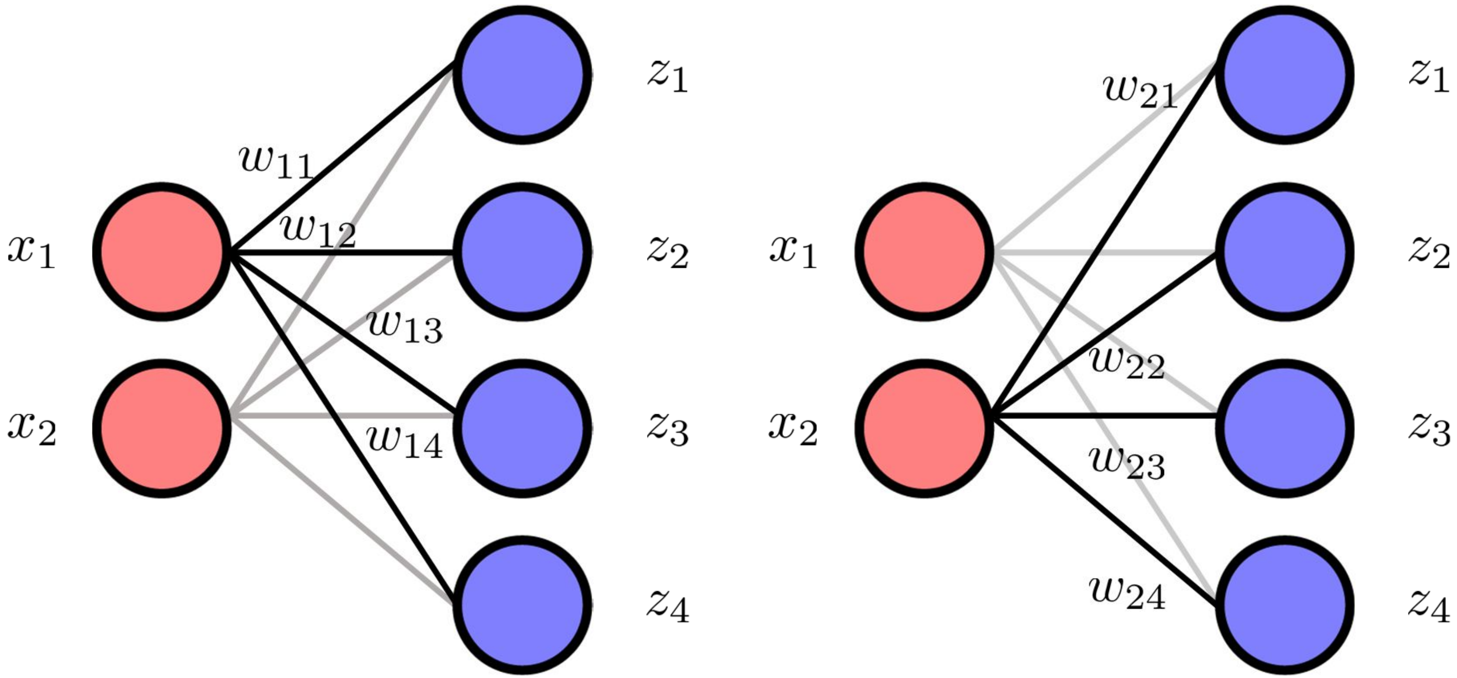
$\mathcal{D} = \{(\vec{x_1}, \vec{y_1}), (\vec{x_2}, \vec{y_2}), \cdots, (\vec{x_N}, \vec{y_N})\}$

Seeks a function $f : X \to Y$

Such that a loss function

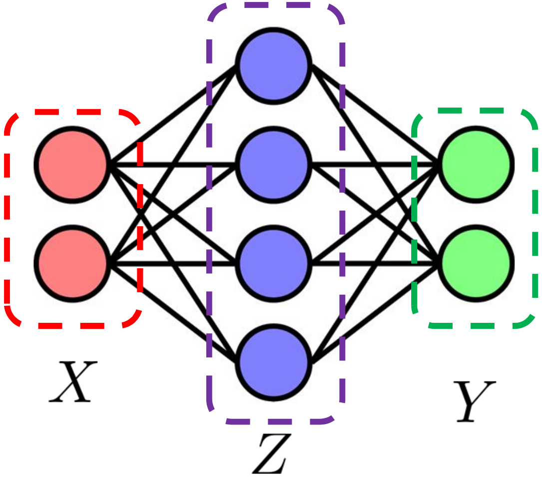
$\mathcal{L} : X \times Y \to \mathcal{R}$ is minimized

Revisit: single layer in a neural network



$$Z = f_1(X)$$

Revisit: Activation function



$$Y = f(X) = f_2(Z) = f_2(f_1(X))$$

$$\boxed{Z = \mathbf{h}(W_{f1}X) = f_1(X)}$$

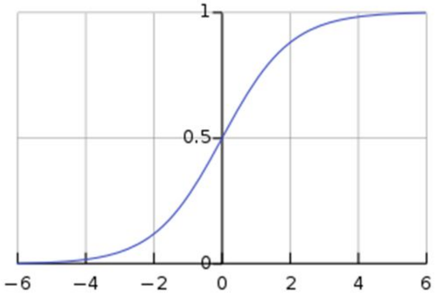
$$Y = \mathbf{h}(W_{f2}Z) = f_2(Z)$$

- *h* is called the **activation function**
- Single layer consists of a **matrix multiplication** & **activation**

Revisit: Activation function

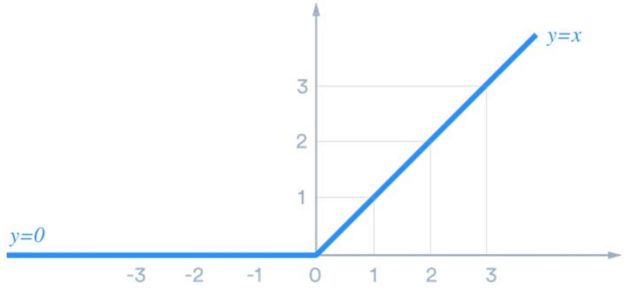
Sigmoid function

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



ReLU (rectified linear unit) function

$$R(z) = max(z, 0)$$



- These two functions are used often as the activation function
  - ReLU is the most popular choice these days
  - There are many other types of activation functions …

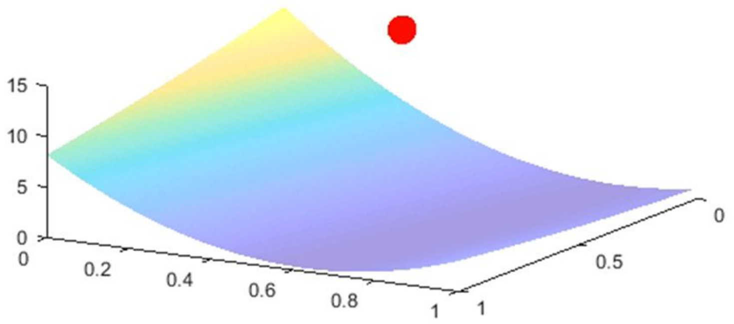
Revisit: Neural network as a function approximator

$$Y = f_W(X)$$

- Conceptually, for an (almost) arbitrary data set, we are using a neural network to model the relationship between the input and the output
- **Universal approximation theorem**, in a nutshell, states that any continuous function can be approximated by a neural network (with a sufficient number of neurons)
- Simply put, neural network is a good model for almost any supervised learning tasks (which is why neural network is so popular)

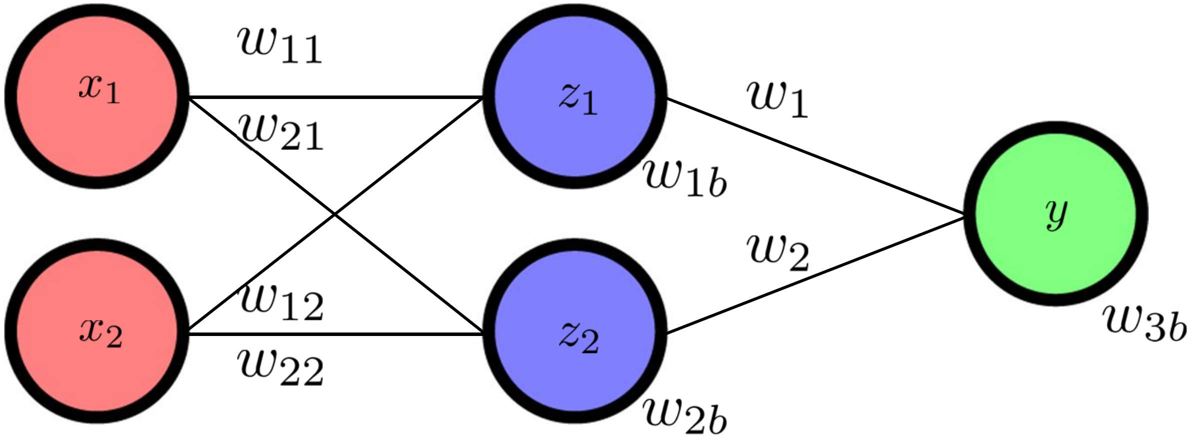
Revisit: Gradient Descent

$$\theta^{(k+1)} = \theta^{(k)} - \gamma \nabla \mathcal{L}(\theta^{(k)})$$



- **Gradient descent** is an iterative algorithm for finding a local minimum of a differentiable function
- It requires only the gradient value at one point at each iteration step (does not require closed-form gradient function)

We need gradient!

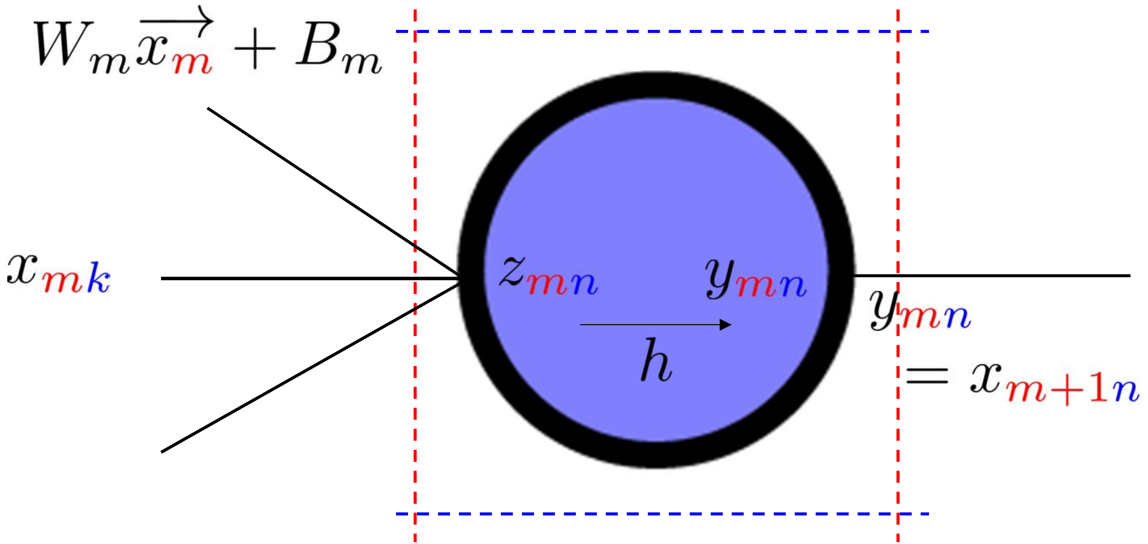


$$\nabla_W \mathcal{L} = \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial w_1} \\ \frac{\partial \mathcal{L}}{\partial w_2} \\ \frac{\partial \mathcal{L}}{\partial w_{11}} \\ \frac{\partial \mathcal{L}}{\partial w_{12}} \\ \frac{\partial \mathcal{L}}{\partial w_{21}} \\ \frac{\partial \mathcal{L}}{\partial w_{22}} \\ \frac{\partial \mathcal{L}}{\partial w_{1b}} \\ \frac{\partial \mathcal{L}}{\partial w_{2b}} \\ \frac{\partial \mathcal{L}}{\partial w_{3b}} \end{bmatrix}$$

$$X = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad Z = \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} \qquad Y = \begin{bmatrix} y \end{bmatrix}$$

This notation is not good for gradient calculation …

Let’s change our notation

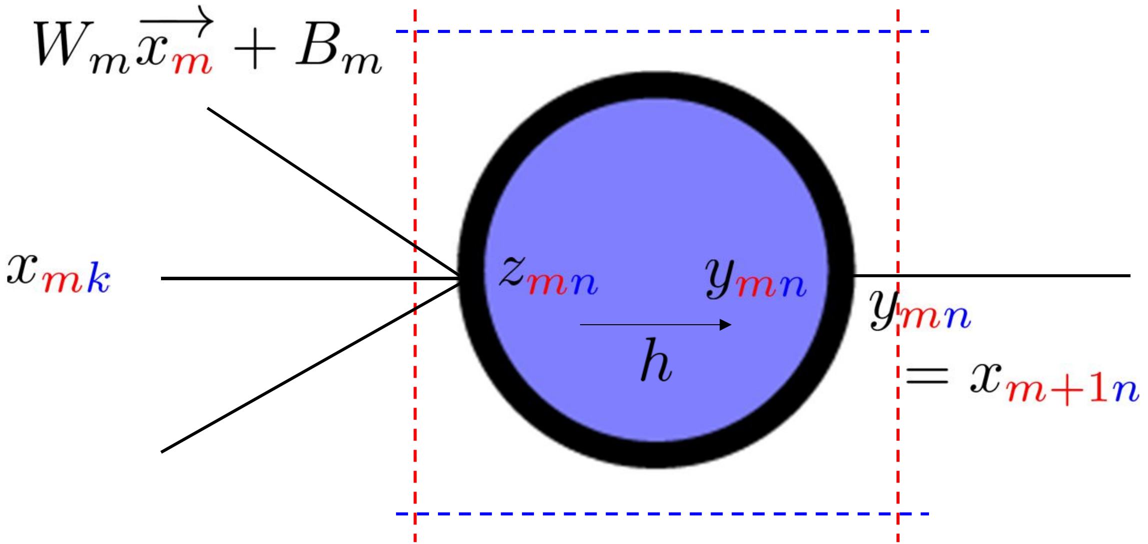


x_{mk} : input from kth neuron of mth layer

$\overrightarrow{x_m}$: input from mth layer as a vector

W_m and B_m : weight parameters

Let's change our notation

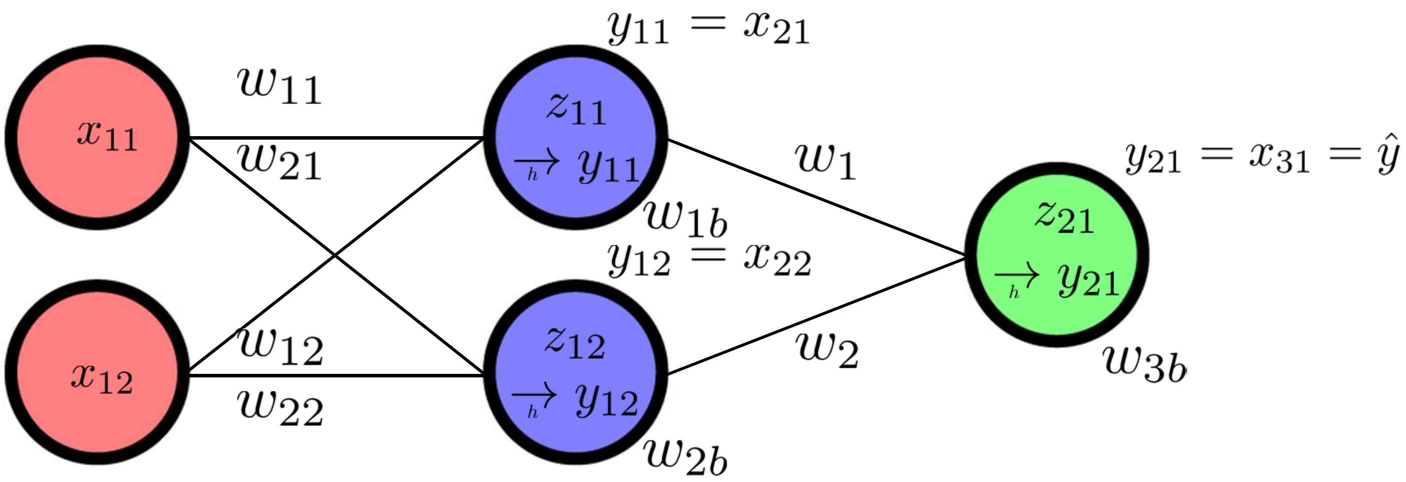


z_{mn} : (summed) input to nth neuron of m+1th layer

y_{mn} : output from nth neuron of m+1th layer

W_m and B_m : weight parameters

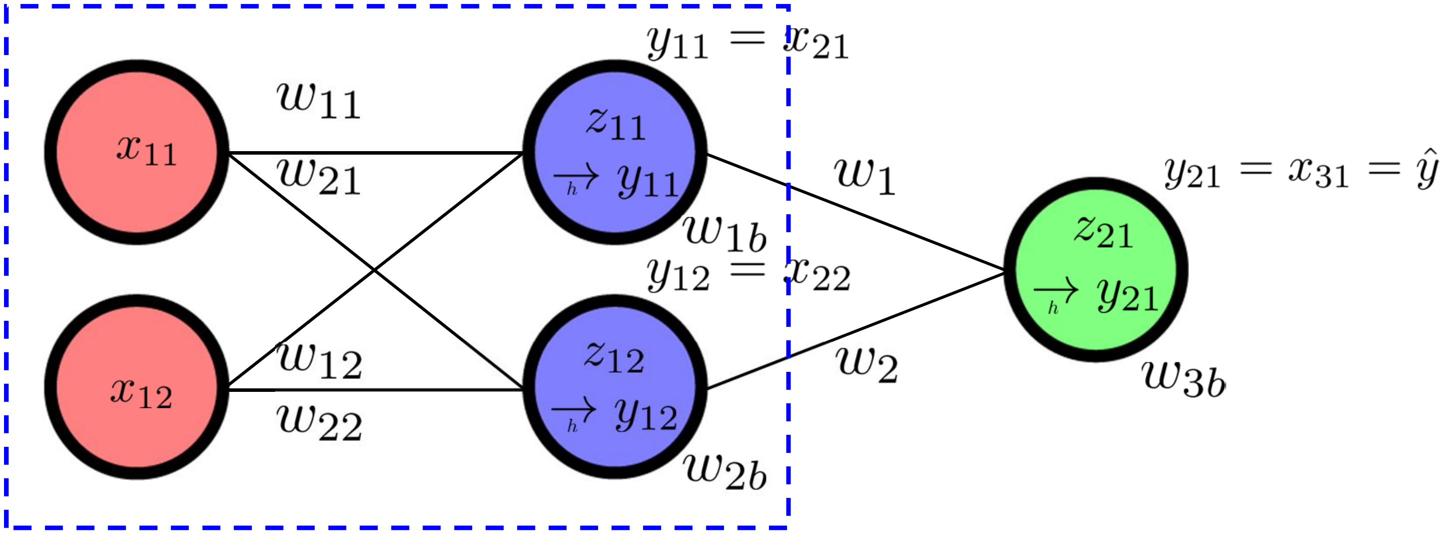
Let's change our notation



$$z_{11} = w_{11}x_{11} + w_{12}x_{12} + w_{1b}$$

$$x_{21} = y_{11} = h(z_{11}) = h(w_{11}x_{11} + w_{12}x_{12} + w_{1b})$$

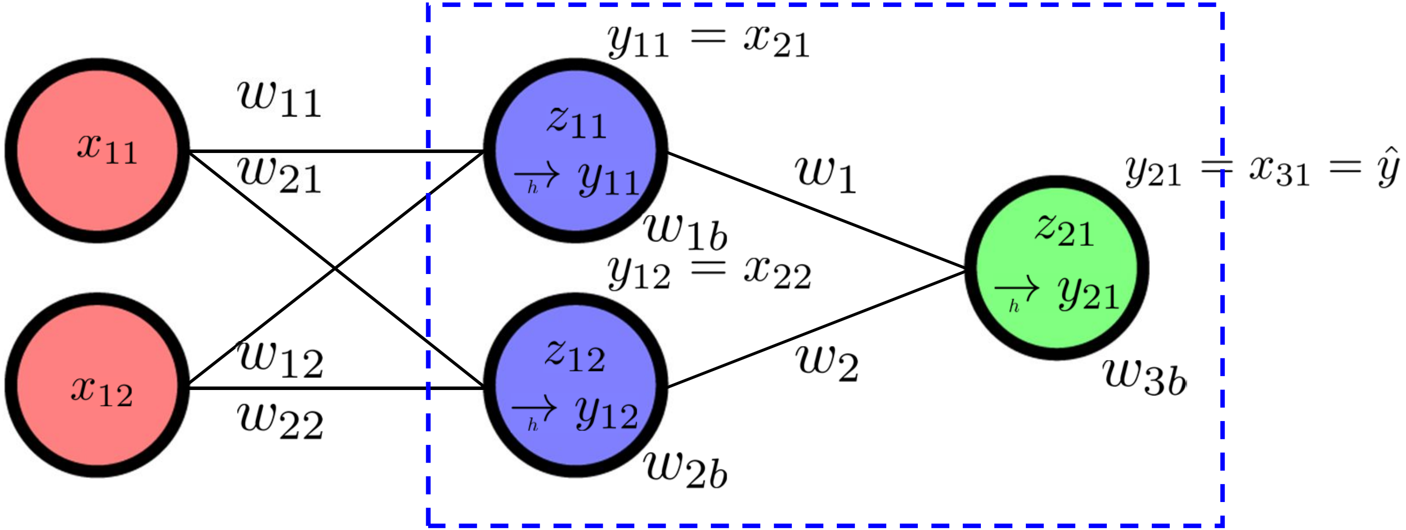
Forward propagation



$$x_{21} = h(w_{11}x_{11} + w_{12}x_{12} + w_{1b})$$
$$x_{22} = h(w_{21}x_{11} + w_{22}x_{12} + w_{2b})$$

$$\begin{bmatrix} x_{21} \\ x_{22} \end{bmatrix} = h(\begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} \begin{bmatrix} x_{11} \\ x_{12} \end{bmatrix} + \begin{bmatrix} w_{1b} \\ w_{2b} \end{bmatrix}) = h(\begin{bmatrix} w_{11} & w_{12} & w_{1b} \\ w_{21} & w_{22} & w_{2b} \end{bmatrix} \begin{bmatrix} x_{11} \\ x_{12} \\ 1 \end{bmatrix})$$

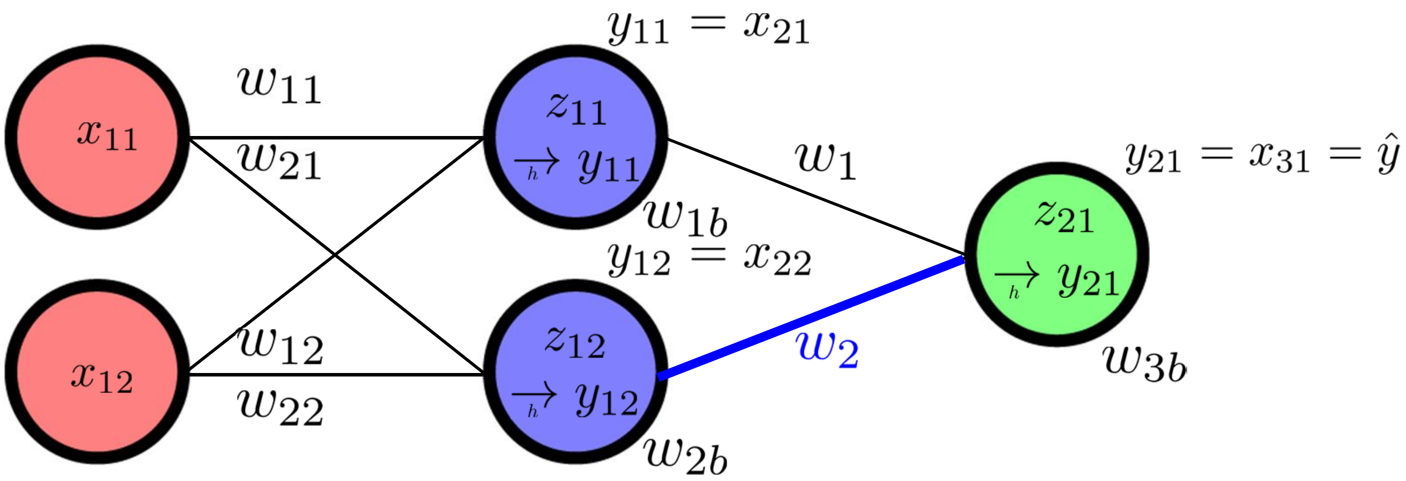
Forward propagation



$$\hat{y} = x_{31} = h(w_1 x_{21} + w_2 x_{22} + w_{3b})$$

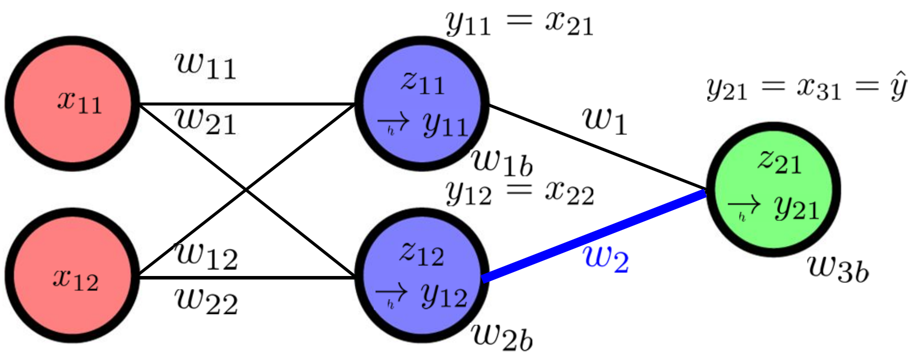
$$\hat{y} = x_{31} = h(\begin{bmatrix} w_1 & w_2 \end{bmatrix} \begin{bmatrix} x_{21} \\ x_{22} \end{bmatrix} + w_{3b}) = h(\begin{bmatrix} w_1 & w_2 & w_{3b} \end{bmatrix} \begin{bmatrix} x_{21} \\ x_{22} \\ 1 \end{bmatrix})$$

Gradient calculation: chain rule



$$\frac{\partial \mathcal{L}}{\partial w_2} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_2} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_{21}} \frac{\partial z_{21}}{\partial w_2} = \frac{\partial \mathcal{L}}{\partial y_{21}} \frac{\partial y_{21}}{\partial z_{21}} \frac{\partial z_{21}}{\partial w_2}$$

Gradient calculation: chain rule



Let's assume that we are using binary cross entropy loss

$$\mathcal{L} = -(y\,log\hat{y} + (1-y)log(1-\hat{y}))$$

and we are using the sigmoid function as our activation function *h*

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

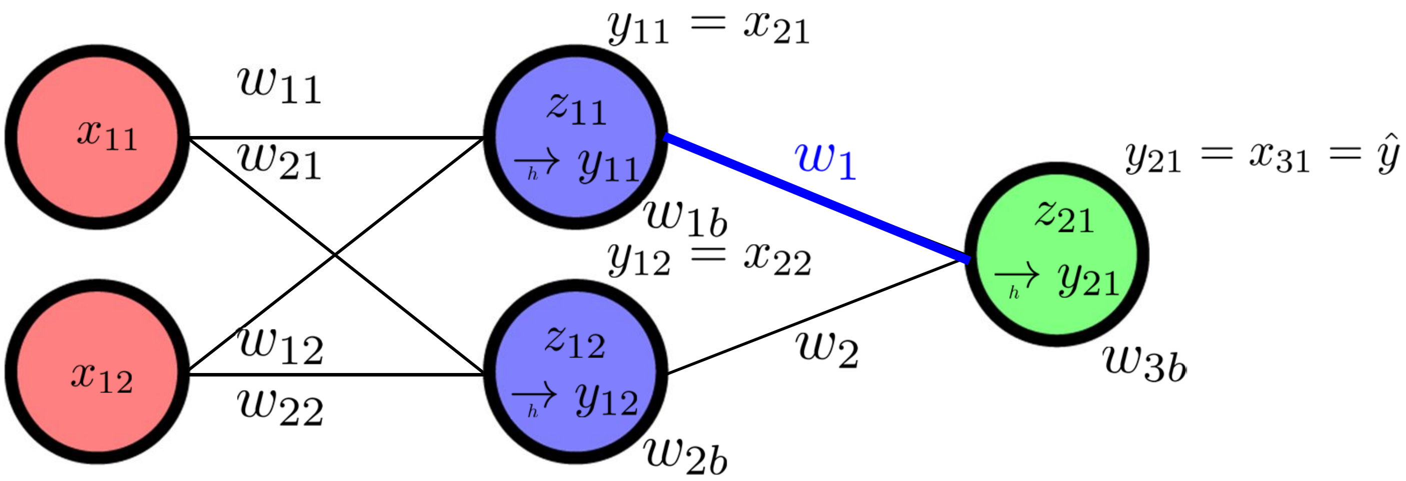
$$\frac{\partial \mathcal{L}}{\partial w_2} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_2} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_{21}} \frac{\partial z_{21}}{\partial w_2}$$

$$\frac{\partial \mathcal{L}}{\partial \hat{y}} = -\frac{\partial}{\partial \hat{y}}(y\,log\hat{y} + (1-y)log(1-\hat{y})) = -(\frac{y}{\hat{y}} - \frac{1-y}{1-\hat{y}}) = -(\frac{y-\hat{y}}{\hat{y}(1-\hat{y})})$$

$$\frac{\partial \hat{y}}{\partial z_{21}} = (\frac{1}{1+e^{-z_{21}}})(\frac{-e^{-z_{21}}}{1+e^{-z_{21}}}) = \hat{y}(1-\hat{y})$$

$$\frac{\partial z_{21}}{\partial w_2} = \frac{\partial}{\partial w_2}(w_1 x_{21} + w_2 x_{22} + w_{3b}) = x_{22}$$

Gradient calculation: chain rule



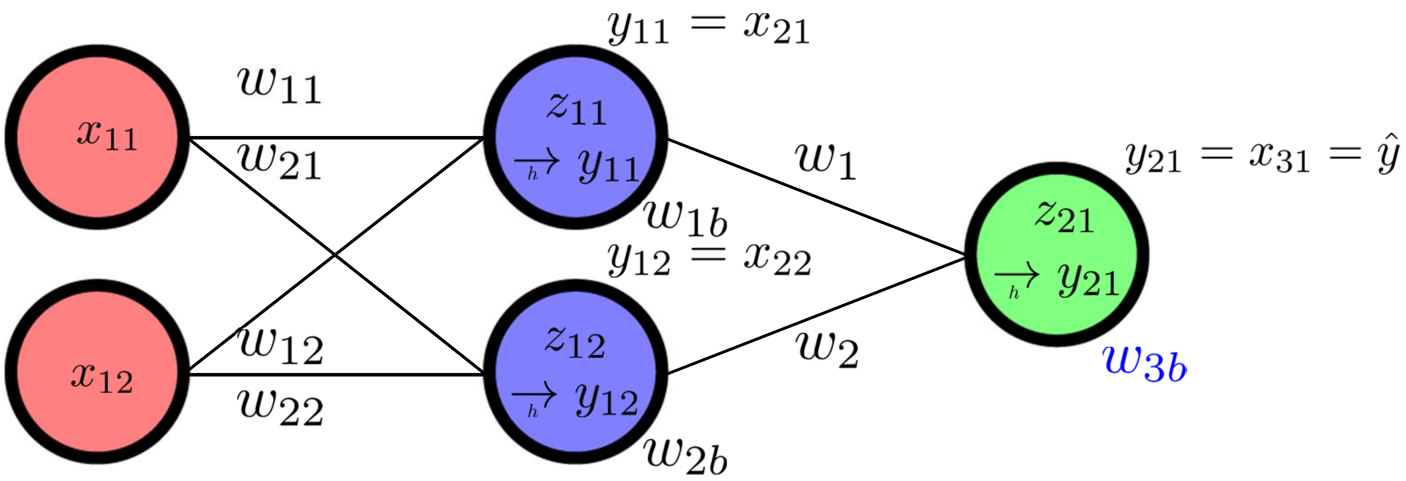
$$\frac{\partial \mathcal{L}}{\partial w_1} = \frac{\partial \mathcal{L}}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial w_1} = \frac{\partial \mathcal{L}}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial z_{21}}\frac{\partial z_{21}}{\partial w_1} = \boxed{\frac{\partial \mathcal{L}}{\partial y_{21}}}\boxed{\frac{\partial y_{21}}{\partial z_{21}}}\boxed{\frac{\partial z_{21}}{\partial w_1}}$$

$$\frac{\partial \mathcal{L}}{\partial \hat{y}} = -\frac{\partial}{\partial \hat{y}}(y\,log\hat{y} + (1-y)log(1-\hat{y})) = -(\frac{y}{\hat{y}} - \frac{1-y}{1-\hat{y}}) = -(\frac{y-\hat{y}}{\hat{y}(1-\hat{y})})$$

$$\frac{\partial \hat{y}}{\partial z_{21}} = (\frac{1}{1+e^{-z_{21}}})(\frac{-e^{-z_{21}}}{1+e^{-z_{21}}}) = \hat{y}(1-\hat{y})$$

$$\frac{\partial z_{21}}{\partial w_1} = \frac{\partial}{\partial w_1}(w_1 x_{21} + w_2 x_{22} + w_{3b}) = x_{21}$$

Gradient calculation: chain rule



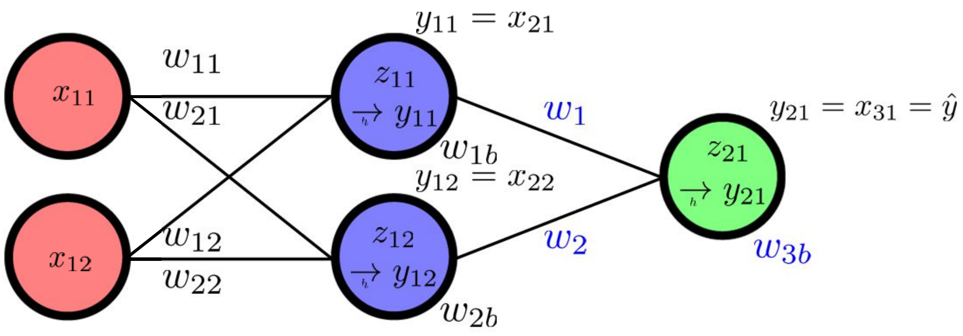
$$\frac{\partial \mathcal{L}}{\partial w_{3b}} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_{3b}} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_{21}} \frac{\partial z_{21}}{\partial w_{3b}} = \boxed{\frac{\partial \mathcal{L}}{\partial y_{21}}}\boxed{\frac{\partial y_{21}}{\partial z_{21}}}\boxed{\frac{\partial z_{21}}{\partial w_{3b}}}$$

$$\frac{\partial \mathcal{L}}{\partial \hat{y}} = -\frac{\partial}{\partial \hat{y}}(y\, log\hat{y} + (1-y)log(1-\hat{y})) = -(\frac{y}{\hat{y}} - \frac{1-y}{1-\hat{y}}) = -(\frac{y-\hat{y}}{\hat{y}(1-\hat{y})})$$

$$\frac{\partial \hat{y}}{\partial z_{21}} = (\frac{1}{1+e^{-z_{21}}})(\frac{-e^{-z_{21}}}{1+e^{-z_{21}}}) = \hat{y}(1-\hat{y})$$

$$\frac{\partial z_{21}}{\partial w_{3b}} = \frac{\partial}{\partial w_{3b}}(w_1 x_{21} + w_2 x_{22} + w_{3b}) = 1$$

Gradient calculation: chain rule



$$\frac{\partial \mathcal{L}}{\partial w_1} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_1} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_{21}} \frac{\partial z_{21}}{\partial w_1}$$

$$\frac{\partial \mathcal{L}}{\partial w_2} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_2} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_{21}} \frac{\partial z_{21}}{\partial w_2}$$

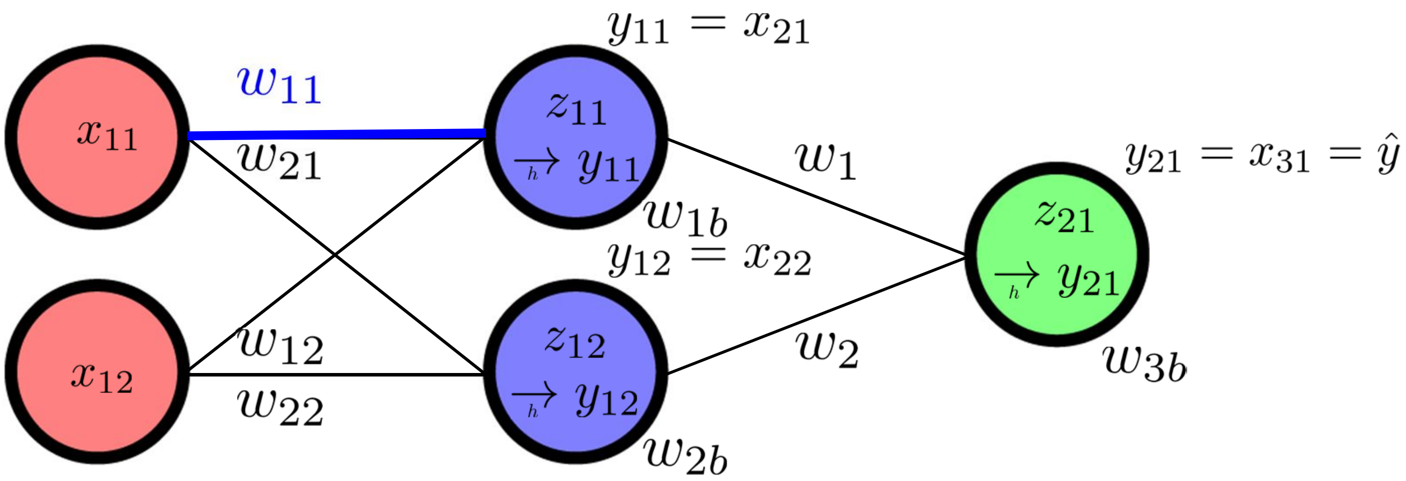
$$\frac{\partial \mathcal{L}}{\partial w_{3b}} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial w_{3b}} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_{21}} \frac{\partial z_{21}}{\partial w_{3b}}$$

$$\frac{\partial z_{21}}{\partial w_1} = \frac{\partial}{\partial w_1}(w_1 x_{21} + w_2 x_{22} + w_{3b}) = x_{21}$$

$$\frac{\partial z_{21}}{\partial w_2} = \frac{\partial}{\partial w_2}(w_1 x_{21} + w_2 x_{22} + w_{3b}) = x_{22}$$

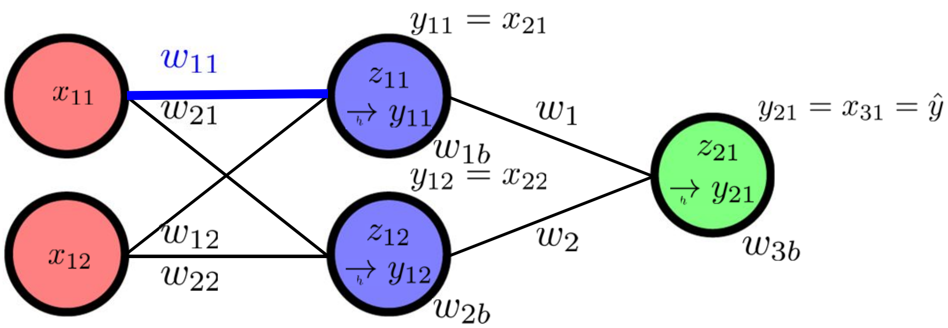
$$\frac{\partial z_{21}}{\partial w_{3b}} = \frac{\partial}{\partial w_{3b}}(w_1 x_{21} + w_2 x_{22} + w_{3b}) = 1$$

Gradient calculation: chain rule



$$\frac{\partial \mathcal{L}}{\partial w_{11}} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_{21}} \frac{\partial z_{21}}{\partial w_{11}} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_{21}} \frac{\partial z_{21}}{\partial y_{11}} \frac{\partial y_{11}}{\partial z_{11}} \frac{\partial z_{11}}{\partial w_{11}}$$

Gradient calculation: chain rule



$$\frac{\partial \mathcal{L}}{\partial w_{11}} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_{21}} \frac{\partial z_{21}}{\partial w_{11}}$$

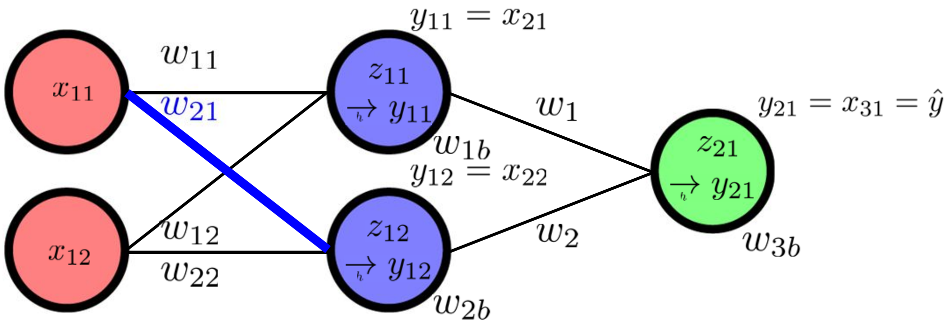
$$= \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_{21}} \frac{\partial z_{21}}{\partial y_{11}} \frac{\partial y_{11}}{\partial z_{11}} \frac{\partial z_{11}}{\partial w_{11}}$$

$$\frac{\partial z_{21}}{\partial y_{11}} = \frac{\partial}{\partial y_{11}}(w_1 x_{21} + w_2 x_{22} + w_{3b}) = w_1$$

$$\frac{\partial y_{11}}{\partial z_{11}} = y_{11}(1 - y_{11})$$

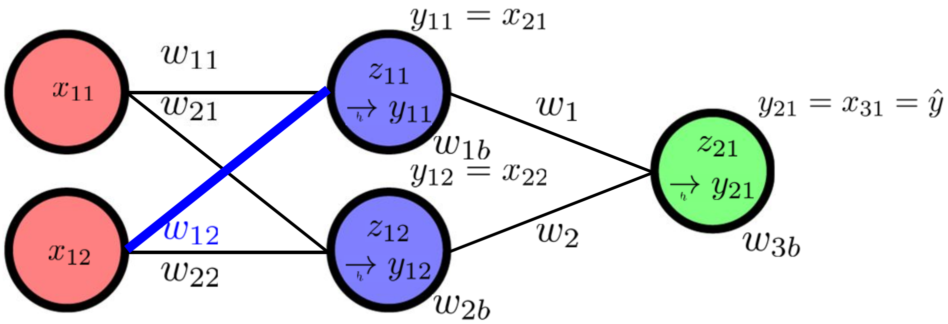
$$\frac{\partial z_{11}}{\partial w_{11}} = \frac{\partial}{\partial w_{11}}(w_{11} x_{11} + w_{12} x_{12} + w_{1b}) = x_{11}$$

Gradient calculation: chain rule



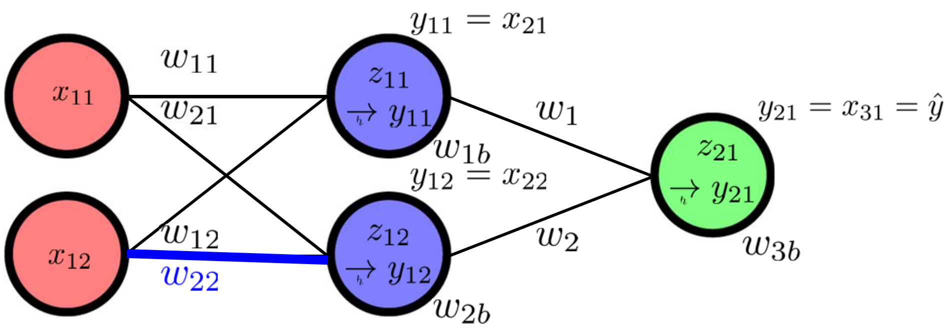
$$\frac{\partial \mathcal{L}}{\partial w_{21}} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_{21}} \frac{\partial z_{21}}{\partial w_{21}}$$

Gradient calculation: chain rule



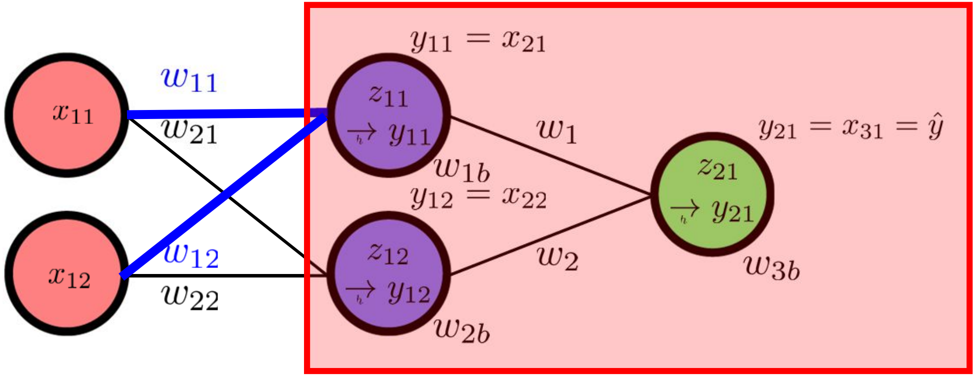
$$\frac{\partial \mathcal{L}}{\partial w_{12}} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_{21}} \frac{\partial z_{21}}{\partial w_{12}}$$

Gradient calculation: chain rule



$$\frac{\partial \mathcal{L}}{\partial w_{22}} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_{21}} \frac{\partial z_{21}}{\partial w_{22}}$$

Gradient calculation: back-propagation

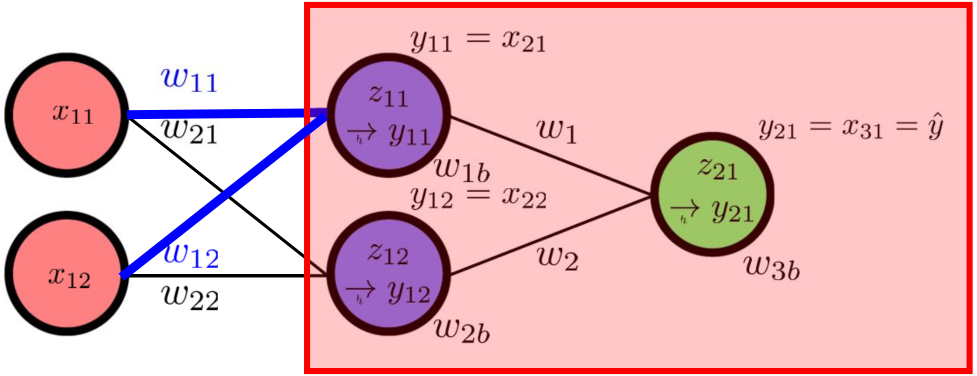


$$\frac{\partial \mathcal{L}}{\partial w_{11}} = \frac{\partial \mathcal{L}}{\partial x_{21}} \frac{\partial x_{21}}{\partial w_{11}}$$

$$\frac{\partial \mathcal{L}}{\partial w_{12}} = \frac{\partial \mathcal{L}}{\partial x_{21}} \frac{\partial x_{21}}{\partial w_{12}}$$

For calculation of $\frac{\partial \mathcal{L}}{\partial w_{11}}$ and $\frac{\partial \mathcal{L}}{\partial w_{12}}$
we just need to know $\frac{\partial \mathcal{L}}{\partial x_{21}}$
about the latter part of the network

Gradient calculation: back-propagation



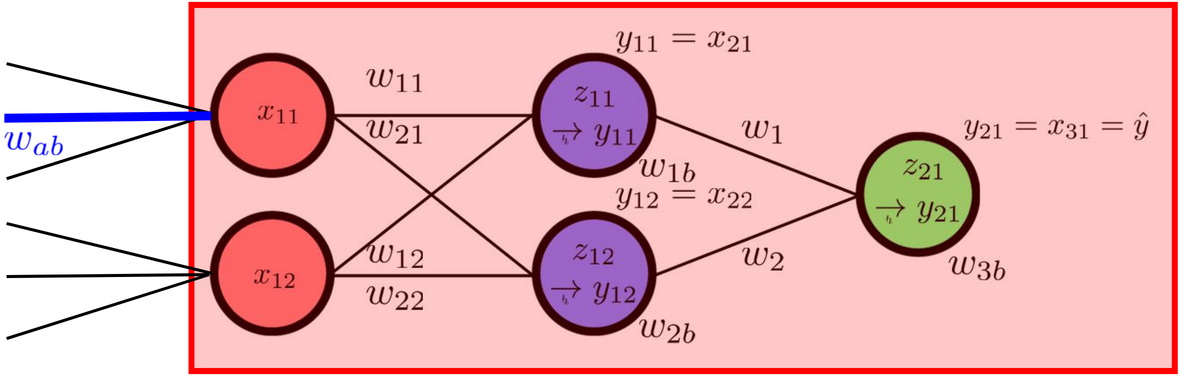
$$\frac{\partial \mathcal{L}}{\partial w_{11}} = \frac{\partial \mathcal{L}}{\partial x_{21}} \frac{\partial x_{21}}{\partial w_{11}} \qquad \frac{\partial \mathcal{L}}{\partial w_{12}} = \frac{\partial \mathcal{L}}{\partial x_{21}} \frac{\partial x_{21}}{\partial w_{12}}$$

For calculation of $\frac{\partial \mathcal{L}}{\partial w_{11}}$ and $\frac{\partial \mathcal{L}}{\partial w_{12}}$
we just need to know $\frac{\partial \mathcal{L}}{\partial x_{21}}$
about the latter part of the network

$$\frac{\partial \mathcal{L}}{\partial x_{21}} = \frac{\partial \mathcal{L}}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_{21}} \frac{\partial z_{21}}{\partial x_{21}}$$

$$\frac{\partial z_{21}}{\partial x_{21}} = \frac{\partial z_{21}}{\partial y_{11}} = \frac{\partial}{\partial y_{11}}(w_1 x_{21} + w_2 x_{22} + w_{3b}) = w_1$$

Gradient calculation: back-propagation

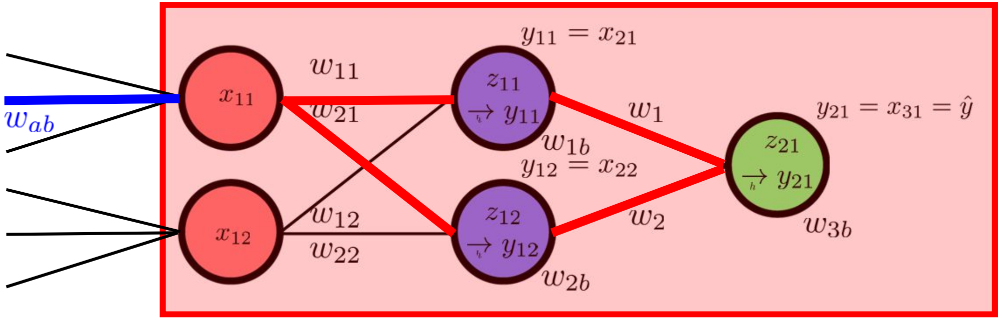


$$\frac{\partial \mathcal{L}}{\partial w_{ab}} = \frac{\partial \mathcal{L}}{\partial x_{11}} \frac{\partial x_{11}}{\partial w_{ab}}$$

For calculation of $\frac{\partial \mathcal{L}}{\partial w_{ab}}$

we just need to know $\frac{\partial \mathcal{L}}{\partial x_{11}}$
about the latter part of the network

Gradient calculation: back-propagation



$$\frac{\partial \mathcal{L}}{\partial w_{ab}} = \frac{\partial \mathcal{L}}{\partial x_{11}} \frac{\partial x_{11}}{\partial w_{ab}}$$

For calculation of $\frac{\partial \mathcal{L}}{\partial w_{ab}}$

we just need to know $\frac{\partial \mathcal{L}}{\partial x_{11}}$

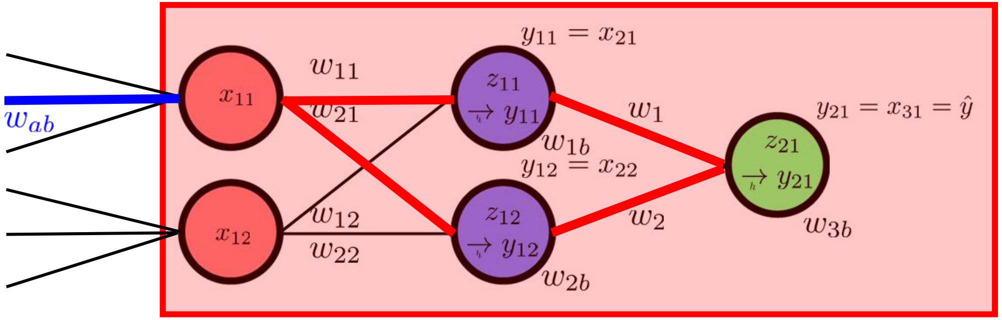
about the latter part of the network

$$\frac{\partial \mathcal{L}}{\partial x_{11}} = \frac{\partial \mathcal{L}}{\partial x_{21}} \frac{\partial x_{21}}{\partial x_{11}} + \frac{\partial \mathcal{L}}{\partial x_{22}} \frac{\partial x_{22}}{\partial x_{11}}$$

$$= \boxed{\frac{\partial \mathcal{L}}{\partial x_{21}}} \frac{\partial x_{21}}{\partial z_{11}} \frac{\partial z_{11}}{\partial x_{11}} + \boxed{\frac{\partial \mathcal{L}}{\partial x_{22}}} \frac{\partial x_{22}}{\partial z_{12}} \frac{\partial z_{12}}{\partial x_{11}}$$

We have these terms from backpropagation (from the latter layer)

Gradient calculation: back-propagation



$$\frac{\partial \mathcal{L}}{\partial w_{ab}} = \frac{\partial \mathcal{L}}{\partial x_{11}} \frac{\partial x_{11}}{\partial w_{ab}}$$

For calculation of $\frac{\partial \mathcal{L}}{\partial w_{ab}}$

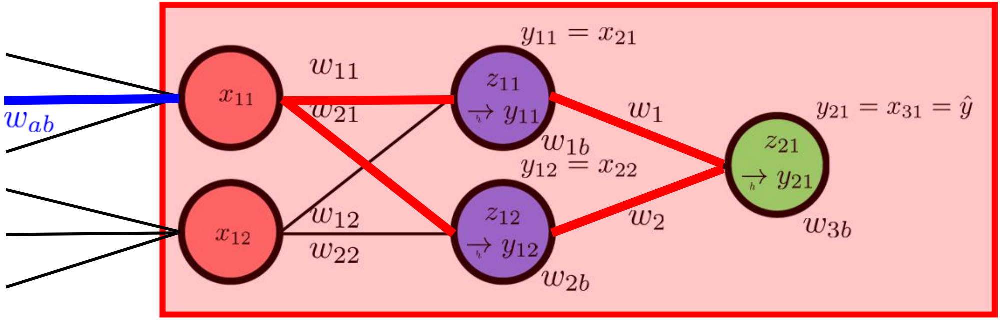
we just need to know $\frac{\partial \mathcal{L}}{\partial x_{11}}$
about the latter part of the network

$$\frac{\partial \mathcal{L}}{\partial x_{11}} = \frac{\partial \mathcal{L}}{\partial x_{21}} \frac{\partial x_{21}}{\partial x_{11}} + \frac{\partial \mathcal{L}}{\partial x_{22}} \frac{\partial x_{22}}{\partial x_{11}}$$

$$= \frac{\partial \mathcal{L}}{\partial x_{21}} \boxed{\frac{\partial x_{21}}{\partial z_{11}}} \frac{\partial z_{11}}{\partial x_{11}} + \frac{\partial \mathcal{L}}{\partial x_{22}} \boxed{\frac{\partial x_{22}}{\partial z_{12}}} \frac{\partial z_{12}}{\partial x_{11}}$$

These are just differentiation of Sigmoid function

Gradient calculation: back-propagation



$$\frac{\partial \mathcal{L}}{\partial w_{ab}} = \frac{\partial \mathcal{L}}{\partial x_{11}} \frac{\partial x_{11}}{\partial w_{ab}}$$

For calculation of $\frac{\partial \mathcal{L}}{\partial w_{ab}}$

we just need to know $\frac{\partial \mathcal{L}}{\partial x_{11}}$

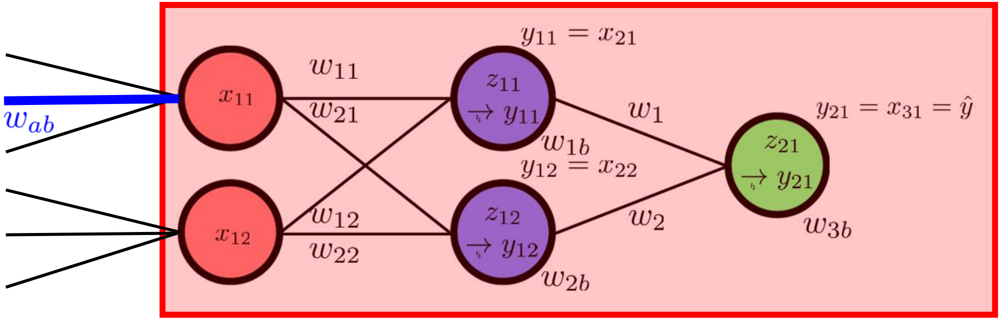
about the latter part of the network

$$\frac{\partial \mathcal{L}}{\partial x_{11}} = \frac{\partial \mathcal{L}}{\partial x_{21}} \frac{\partial x_{21}}{\partial x_{11}} + \frac{\partial \mathcal{L}}{\partial x_{22}} \frac{\partial x_{22}}{\partial x_{11}}$$

$$= \frac{\partial \mathcal{L}}{\partial x_{21}} \frac{\partial x_{21}}{\partial z_{11}} \boxed{\frac{\partial z_{11}}{\partial x_{11}}} + \frac{\partial \mathcal{L}}{\partial x_{22}} \frac{\partial x_{22}}{\partial z_{12}} \boxed{\frac{\partial z_{12}}{\partial x_{11}}}$$

These are just w_{11} and w_{21}

Gradient calculation: back-propagation



$$\frac{\partial \mathcal{L}}{\partial x_{11}} = \frac{\partial \mathcal{L}}{\partial x_{21}}\frac{\partial x_{21}}{\partial x_{11}} + \frac{\partial \mathcal{L}}{\partial x_{22}}\frac{\partial x_{22}}{\partial x_{11}}$$

$$= \frac{\partial \mathcal{L}}{\partial x_{21}}\frac{\partial x_{21}}{\partial z_{11}} w_{11} + \frac{\partial \mathcal{L}}{\partial x_{22}}\frac{\partial x_{22}}{\partial z_{12}} w_{21}$$

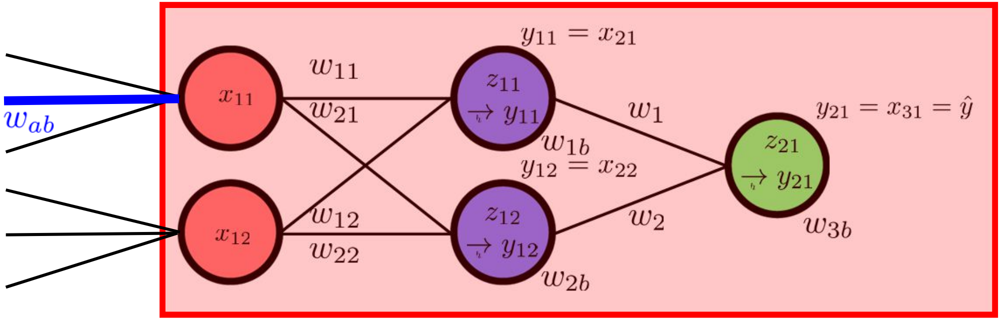
$$\frac{\partial \mathcal{L}}{\partial x_{12}} = \frac{\partial \mathcal{L}}{\partial x_{21}}\frac{\partial x_{21}}{\partial x_{12}} + \frac{\partial \mathcal{L}}{\partial x_{22}}\frac{\partial x_{22}}{\partial x_{12}}$$

$$= \frac{\partial \mathcal{L}}{\partial x_{21}}\frac{\partial x_{21}}{\partial z_{11}} w_{12} + \frac{\partial \mathcal{L}}{\partial x_{22}}\frac{\partial x_{22}}{\partial z_{12}} w_{22}$$

$$\begin{bmatrix} \frac{\partial \mathcal{L}}{\partial x_{11}} \\ \frac{\partial \mathcal{L}}{\partial x_{12}} \end{bmatrix} = \begin{bmatrix} w_{11} & w_{21} \\ w_{12} & w_{22} \end{bmatrix} \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial x_{21}} \boxed{\frac{\partial x_{21}}{\partial z_{11}}} \\ \frac{\partial \mathcal{L}}{\partial x_{22}} \frac{\partial x_{22}}{\partial z_{12}} \end{bmatrix}$$

This would be just $x_{21}(1 - x_{21})$
if we are using Sigmoid as our activation

Forward propagation vs. Backpropagation



Forward propagation

$$\begin{bmatrix} x_{21} \\ x_{22} \end{bmatrix} = h\left(\boxed{\begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix}} \begin{bmatrix} x_{11} \\ x_{12} \end{bmatrix} + \begin{bmatrix} w_{1b} \\ w_{2b} \end{bmatrix}\right)$$

Backpropagation

Transpose!

$$\begin{bmatrix} \frac{\partial \mathcal{L}}{\partial x_{11}} \\ \frac{\partial \mathcal{L}}{\partial x_{12}} \end{bmatrix} = \boxed{\begin{bmatrix} w_{11} & w_{21} \\ w_{12} & w_{22} \end{bmatrix}} \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial x_{21}} \frac{\partial x_{21}}{\partial z_{11}} \\ \frac{\partial \mathcal{L}}{\partial x_{22}} \frac{\partial x_{22}}{\partial z_{12}} \end{bmatrix}$$

Forward propagation vs. Backpropagation

Forward propagation

$$\begin{bmatrix} x_{21} \\ x_{22} \end{bmatrix} = h\left(\boxed{\begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix}} \begin{bmatrix} x_{11} \\ x_{12} \end{bmatrix} + \begin{bmatrix} w_{1b} \\ w_{2b} \end{bmatrix}\right)$$

1. Output (ŷ) can be obtained by **repeating single layer forward propagation**
2. For single layer forward propagation
   1. We take input from the previous layer
   2. **Multiply with weight matrix**
   3. Add bias terms
   4. Apply activation function

Backpropagation

$$\begin{bmatrix} \frac{\partial \mathcal{L}}{\partial x_{11}} \\ \frac{\partial \mathcal{L}}{\partial x_{12}} \end{bmatrix} = \boxed{\begin{bmatrix} w_{11} & w_{21} \\ w_{12} & w_{22} \end{bmatrix}} \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial x_{21}} \frac{\partial x_{21}}{\partial z_{11}} \\ \frac{\partial \mathcal{L}}{\partial x_{22}} \frac{\partial x_{22}}{\partial z_{12}} \end{bmatrix}$$

1. Output (dL/dx, dL/dw) can be obtained by **repeating single layer backpropagation**
2. For single layer backpropagation
   1. We take dL/dx from the latter layer
   2. Element-wise multiplication with the partial derivative of activation function
   3. **Multiply with the transpose of weight matrix**

Summary

- We can use gradient descent to train a neural network
- Gradient descent requires calculation of gradient
- We can calculate gradient (partial derivatives) using chain rule
- Gradient in a neural network can be efficiently computed by "reusing" terms
    - Backpropagation allows us to do this!

References

- Lecture notes
  - MIT 6.036 Intro to Machine Learning (Chapter 8)
    - https://www.mit.edu/~lindrew/6.036.pdf
  - CC229 lecture note
    - http://cs229.stanford.edu/notes-spring2019/cs229-notes-deep_learning.pdf
    - http://cs229.stanford.edu/notes-spring2019/backprop.pdf
- Website
  - CS231n course website: https://cs231n.github.io/