Department of Computer Science
UNIVERSITY OF COLORADO **BOULDER**

Machine Learning: Yoshinari Fujinuma

University of Colorado Boulder

LECTURE 15

Slides adapted from Chenhao Tan, Jordan Boyd-Graber, Justin Johnson, Andrej Karpathy, Chris Ketelsen, Fei-Fei Li, Mike Mozer, Michael Nielson

Logistics

- Quiz 1 grades are available
- Homework 3 will be available today
- Project team formulation deadline is today

Overview

Forward propagation recap

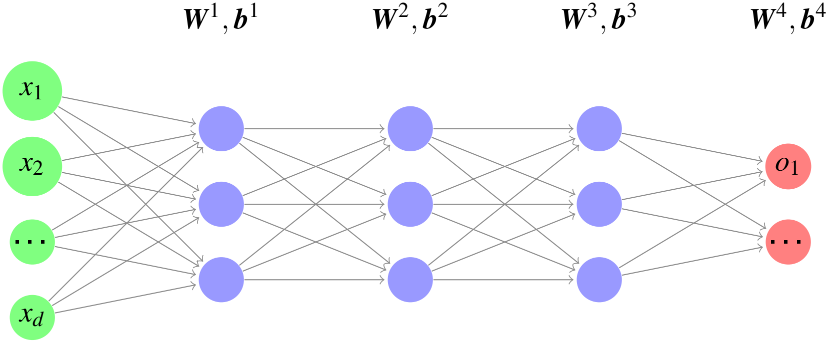
Back propagation

- Chain rule
- Back propagation
- Full algorithm

Outline

Forward propagation algorithm

How do we make predictions based on a multi-layer neural network?
Store the biases for layer l in $\boldsymbol{b}^l$, weight matrix in $\boldsymbol{W}^l$



Forward propagation algorithm

Suppose your network has L layers
Make prediction for an instance $\boldsymbol{x}$

1: Initialize $\boldsymbol{a}^0 = \boldsymbol{x}$
2: **for** $l = 1$ to L **do**
3: $\quad \boldsymbol{z}^l = \boldsymbol{W}^l \boldsymbol{a}^{l-1} + \boldsymbol{b}^l$
4: $\quad \boldsymbol{a}^l = g(\boldsymbol{z}^l)$ // g represents the nonlinear activation
5: **end for**
6: The prediction $\hat{y}$ is simply $\boldsymbol{a}^L$

Neural networks in a nutshell

- Training data $S_{\text{train}} = \{(\boldsymbol{x}, y)\}$
- Network architecture (model)

$$\hat{y} = f_w(\boldsymbol{x})$$

$$\boldsymbol{W}^l, \boldsymbol{b}^l, l = 1, \ldots, L$$

- Loss function (objective function)

$$\mathscr{L}(y, \hat{y})$$

- How do we learn the parameters?

Neural networks in a nutshell

- Training data $S_{\text{train}} = \{(\boldsymbol{x}, y)\}$
- Network architecture (model)

$$\hat{y} = f_w(\boldsymbol{x})$$

$$\boldsymbol{W}^l, \boldsymbol{b}^l, l = 1, \ldots, L$$

- Loss function (objective function)

$$\mathscr{L}(y, \hat{y})$$

- How do we learn the parameters?
  Stochastic gradient descent,

$$\boldsymbol{W}^l \leftarrow \boldsymbol{W}^l - \eta \frac{\partial \mathscr{L}(y, \hat{y})}{\partial \boldsymbol{W}^l}$$

Challenge

- **Challenge**: How do we compute derivatives of the loss function with respect to weights and biases?
- **Solution**: Back propagation

Outline

The Chain Rule

The chain rule allows us to take derivatives of nested functions.

The Chain Rule

The chain rule allows us to take derivatives of nested functions.
Univariate chain rule:

$$\frac{d}{dx} f(g(x)) = f'(g(x))\, g'(x) = \frac{df}{dg}\frac{dg}{dx}$$

The Chain Rule

The chain rule allows us to take derivatives of nested functions.
Univariate chain rule:

$$\frac{d}{dx} f(g(x)) = f'(g(x))\, g'(x) = \frac{df}{dg}\frac{dg}{dx}$$

Example:

$$\frac{d}{dx} \frac{1}{1+\exp(-x)}$$

The Chain Rule

The chain rule allows us to take derivatives of nested functions.
Univariate chain rule:

$$\frac{d}{dx} f(g(x)) = f'(g(x))\, g'(x) = \frac{df}{dg}\frac{dg}{dx}$$

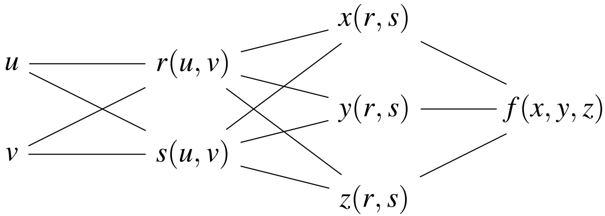
Example:

$$\frac{d}{dx}\, \frac{1}{1+\exp(-x)} = -\frac{1}{(1+\exp(-x))^2} \cdot \exp(-x) \cdot -1$$

$$\frac{d\sigma(x)}{dx} = \sigma(x)(1 - \sigma(x))$$

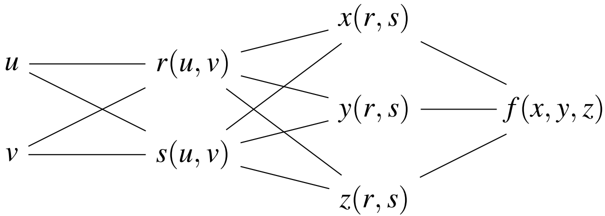
The Chain Rule

Multivariate chain rule:



The Chain Rule

Multivariate chain rule:

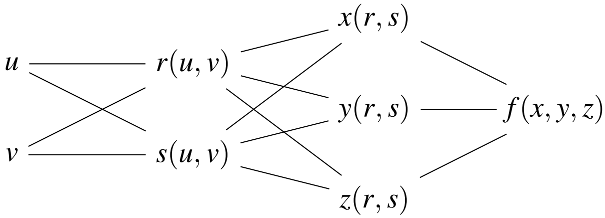


Derivative of $\mathscr{L}$ with respect to x is straightforward. e.g., $f = xyz$, then:

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

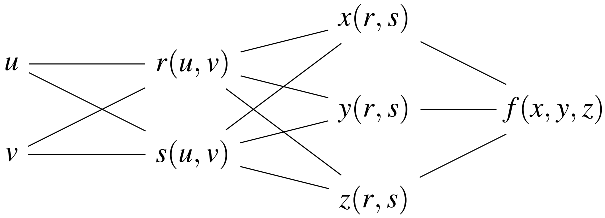
The Chain Rule

What is the derivative of f with respect to r?



The Chain Rule

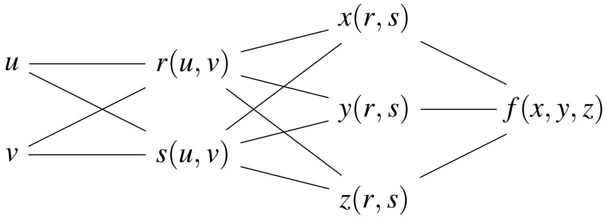
What is the derivative of f with respect to r?



$$\frac{\partial f}{\partial r} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial r} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial r} + \frac{\partial f}{\partial z}\frac{\partial z}{\partial r}$$

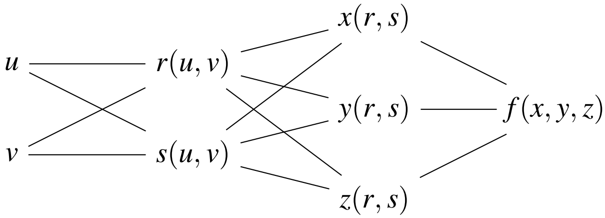
The Chain Rule

What is the derivative of f with respect to s?



The Chain Rule

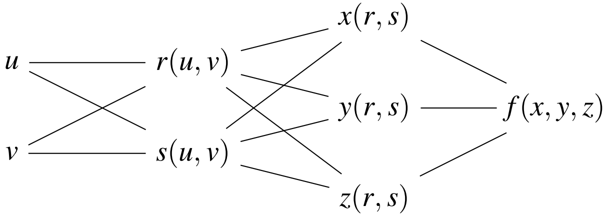
What is the derivative of f with respect to s?



$$\frac{\partial f}{\partial s} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial s} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial s} + \frac{\partial f}{\partial z}\frac{\partial z}{\partial s}$$

The Chain Rule

What is the derivative of f with respect to s?

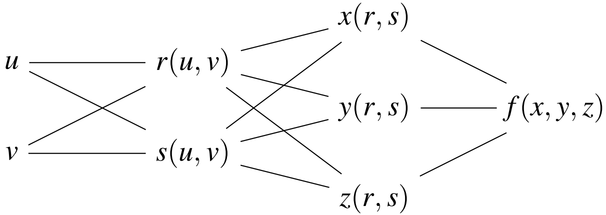


Example: Let $f = xyz$, $x = r$, $y = rs$, and $z = s$. Find $\partial f / \partial s$

$$\frac{\partial f}{\partial s} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial s} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial s} + \frac{\partial f}{\partial z}\frac{\partial z}{\partial s}$$

The Chain Rule

What is the derivative of f with respect to s?

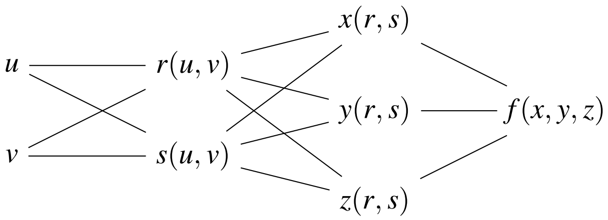


Example: Let $f = xyz$, $x = r$, $y = rs$, and $z = s$. Find $\partial f / \partial s$

$$\frac{\partial f}{\partial s} = yz \cdot 0 + xz \cdot r + xy \cdot 1$$

The Chain Rule

What is the derivative of f with respect to s?

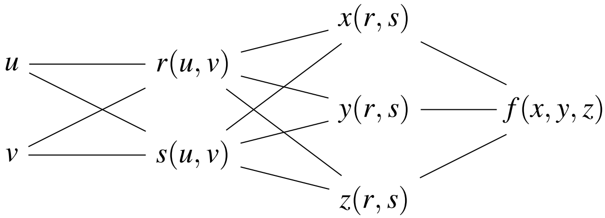


Example: Let $f = xyz$, $x = r$, $y = rs$, and $z = s$. Find $\partial f / \partial s$

$$\frac{\partial f}{\partial s} = rs^2 \cdot 0 + rs \cdot r + r^2 s \cdot 1$$

The Chain Rule

What is the derivative of f with respect to s?

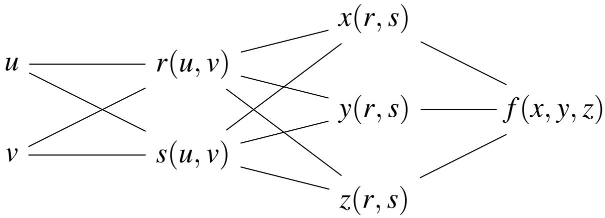


Example: Let $f = xyz$, $x = r$, $y = rs$, and $z = s$. Find $\partial f/\partial s$

$$\frac{\partial f}{\partial s} = 2r^2 s$$

The Chain Rule

What is the derivative of f with respect to s?

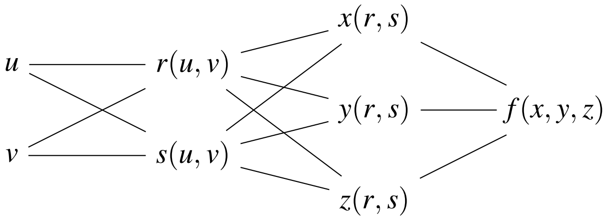


Example: Let $f = xyz$, $x = r$, $y = rs$, and $z = s$. Find $\partial f / \partial s$

$$f(r, s) = r \cdot rs \cdot s = r^2 s^2 \quad \Rightarrow \quad \frac{\partial f}{\partial s} = 2r^2 s \;\checkmark$$

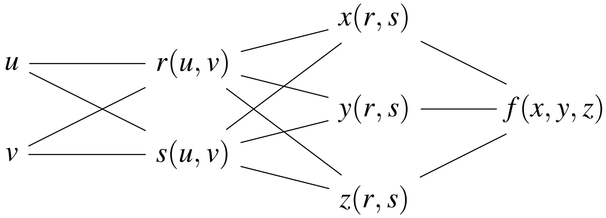
The Chain Rule

What is the derivative of f with respect to u?



The Chain Rule

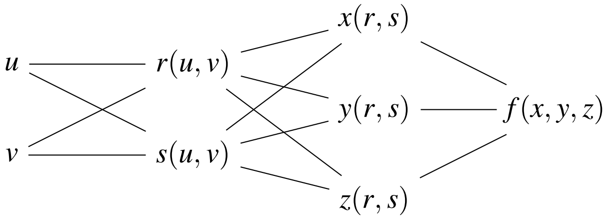
What is the derivative of f with respect to u?



$$\frac{\partial f}{\partial u} = \frac{\partial f}{\partial r}\frac{\partial r}{\partial u} + \frac{\partial f}{\partial s}\frac{\partial s}{\partial u}$$

The Chain Rule

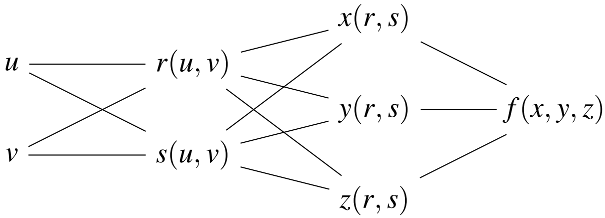
What is the derivative of f with respect to u?



If you know the derivative of objective w.r.t. intermediate value in the chain, you don't need to now the following values in the chain.

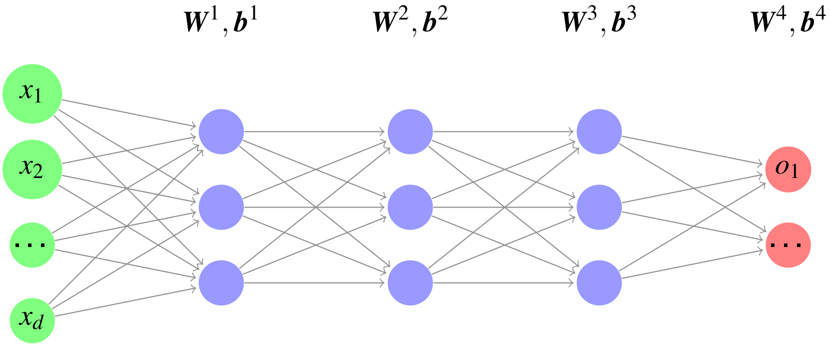
The Chain Rule

What is the derivative of f with respect to u?



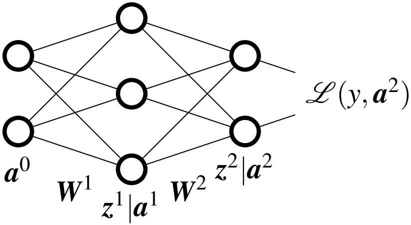
If you know the derivative of objective w.r.t. intermediate value in the chain, you don't need to now the following values in the chain.
This is the cornerstone of the *back propagation* algorithm.

Back Propagation



Back Propagation

For the derivation, we'll consider a simplified network



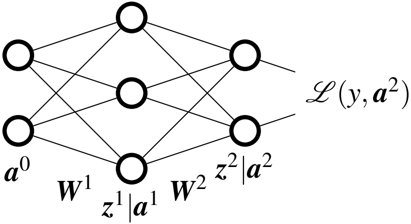
Remind that $a^L = g(z^L)$ where g is the activation function.
Use back propagation to compute partial derivative of $\mathscr{L}$ w.r.t. the weights

$$\frac{\partial \mathscr{L}}{\partial w_{ij}^2}, \text{for } l = 1, 2$$

w_{ij}^l is the weight from node j in layer $l-1$ to node i in layer l.

Back Propagation

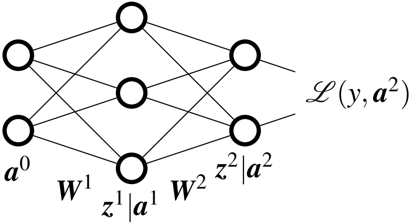
For the derivation, we'll consider a simplified network



Choose an intermediate term so that we can easily compute derivatives.

Back Propagation

For the derivation, we'll consider a simplified network



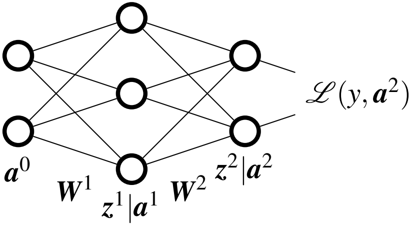
Define the derivative w.r.t. the z's by δ:

$$\delta_j^l = \frac{\partial \mathscr{L}}{\partial z_j^l}$$

Note that $\boldsymbol{\delta}^l$ has the same size as $\boldsymbol{z}^l$ and $\boldsymbol{a}^l$.

Back Propagation

For the derivation, we'll consider a simplified network



Let's compute δ^L for output layer L:

$$\delta_j^L = \frac{\partial \mathscr{L}}{\partial z_j^L} = \frac{\partial \mathscr{L}}{\partial a_j^L} \frac{d a_j^L}{d z_j^L}$$

Back Propagation

$$\delta_j^L = \frac{\partial \mathscr{L}}{\partial z_j^L} = \frac{\partial \mathscr{L}}{\partial a_j^L} \frac{da_j^L}{dz_j^L}$$

We know that $a_j^L = g(z_j^L)$, so $\dfrac{da_j^L}{dz_j^L} = g'(z_j^L)$

$$\delta_j^L = \frac{\partial \mathscr{L}}{\partial a_j^L} g'(z_j^L)$$

Note: The first term is j^{th} entry of gradient of $\mathscr{L}$.

Back Propagation

$$\delta_j^L = \frac{\partial \mathscr{L}}{\partial a_j^L} g'(z_j^L)$$

We can combine all of these into a vector operation

$$\boldsymbol{\delta}^L = \frac{\partial \mathscr{L}}{\partial \boldsymbol{a}^L} \odot g'(\boldsymbol{z}^L)$$

Where $g'(\boldsymbol{z}^L)$ is the activation function applied elementwise to $\boldsymbol{z}^L$.
The symbol $\odot$ indicates element-wise multiplication of vectors.

Back Propagation

$$\delta_j^L = \frac{\partial \mathscr{L}}{\partial a_j^L} g'(z_j^L)$$

We can combine all of these into a vector operation

$$\boldsymbol{\delta}^L = \frac{\partial \mathscr{L}}{\partial \boldsymbol{a}^L} \odot g'(\boldsymbol{z}^L)$$

Where $g'(\boldsymbol{z}^L)$ is the activation function applied elementwise to $\boldsymbol{z}^L$.
The symbol $\odot$ indicates element-wise multiplication of vectors.

Notice that computing $\boldsymbol{\delta}^L$ requires knowing activations.
This means that before we can compute derivatives for SGD through back propagation, we first run forward propagation through the network.

Back Propagation

Example: Suppose we're in regression setting and choose a sigmoid activation function:

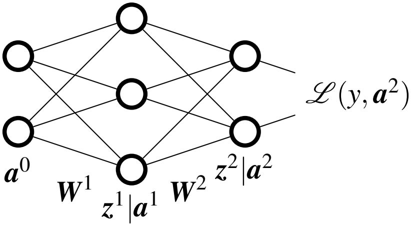
$$\mathscr{L} = \frac{1}{2}\sum_j (y_j - a_j^L)^2 \quad \text{and} \quad a_j^L = \sigma(z)$$

$$\frac{\partial \mathscr{L}}{\partial a_j^L} = (a_j^L - y_j), \quad \frac{da_j^L}{dz_j^L} = \sigma'(z_j^L) = \sigma(z_j^L)(1 - \sigma(z_j^L))$$

So $\boldsymbol{\delta}^L = (\boldsymbol{a}^L - y) \odot \sigma(\boldsymbol{z}^L) \odot (1 - \sigma(\boldsymbol{z}^L))$

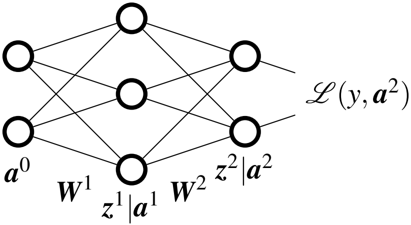
Back Propagation

Now we can easily-ish compute the δ's for the output layer.
But really we're after partials w.r.t. to weights and biases.



Back Propagation

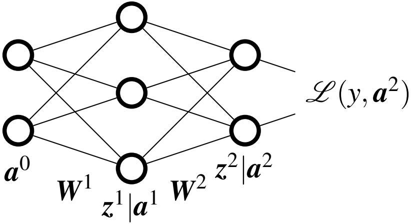
Now we can easily-ish compute the δ's for the output layer.
But really we're after partials w.r.t. to weights and biases.



Question: What do you notice?

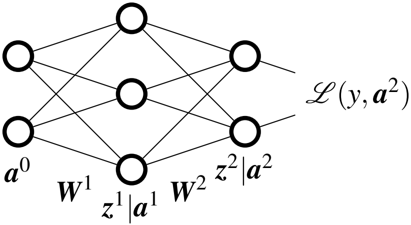
Back Propagation

We want to find derivative $\mathscr{L}$ w.r.t. to weights and biases



Every weight connected to a node in layer L depends on a single δ_j^L

Back Propagation

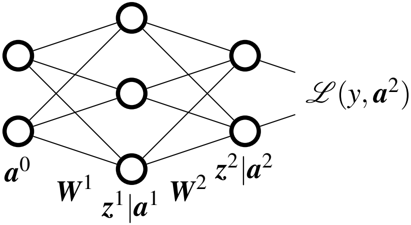


So we have $$\frac{\partial \mathscr{L}}{\partial w_{jk}^L} = \frac{\partial \mathscr{L}}{\partial z_j^L}\frac{\partial z_j^L}{\partial w_{jk}^L} = \delta_j^L \frac{\partial z_j^L}{\partial w_{jk}^L}$$

Need to compute $\frac{\partial z_j^L}{\partial w_{jk}^L}$. Recall $\mathbf{z}^L = W^L \mathbf{a}^{L-1} + \mathbf{b}^L$

$$j^{\text{th}} \text{ entry in vector} \Rightarrow \quad z_j^L = \sum_i w_{ji}^L a_i^{L-1} + b_j^L$$

Back Propagation

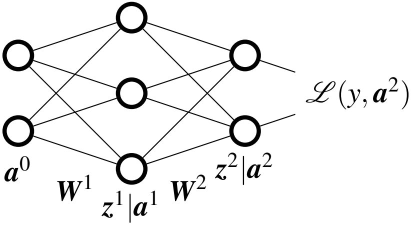


So we have
$$\frac{\partial \mathcal{L}}{\partial w_{jk}^L} = \frac{\partial \mathcal{L}}{\partial z_j^L} \frac{\partial z_j^L}{\partial w_{jk}^L} = \delta_j^L \frac{\partial z_j^L}{\partial w_{jk}^L}$$

Taking derivative w.r.t. w_{jk}^L gives

$$\Rightarrow \quad \frac{\partial z_j^L}{\partial w_{jk}^L} = a_k^{L-1} \quad \Rightarrow \quad \frac{\partial \mathcal{L}}{\partial w_{jk}^L} = a_k^{L-1} \delta_j^L$$

Back Propagation



So we have $$\frac{\partial \mathscr{L}}{\partial w_{jk}^L} = a_k^{L-1}\delta_j^L$$

Easy expression for derivative w.r.t. every weight leading into layer L.

Back Propagation

Let's make the notation a little more practical.

$$\boldsymbol{W}^2 = \begin{bmatrix} w_{11}^2 & w_{12}^2 & w_{13}^2 \\ w_{21}^2 & w_{22}^2 & w_{23}^2 \end{bmatrix}$$

Back Propagation

Let's make the notation a little more practical.

$$\boldsymbol{W}^2 = \begin{bmatrix} w_{11}^2 & w_{12}^2 & w_{13}^2 \\ w_{21}^2 & w_{22}^2 & w_{23}^2 \end{bmatrix}$$

$$\frac{\partial \mathscr{L}}{\partial \boldsymbol{W}^2} = \begin{bmatrix} \frac{\partial \mathscr{L}}{\partial w_{11}^2} & \frac{\partial \mathscr{L}}{\partial w_{12}^2} & \frac{\partial \mathscr{L}}{\partial w_{13}^2} \\ \frac{\partial \mathscr{L}}{\partial w_{21}^2} & \frac{\partial \mathscr{L}}{\partial w_{22}^2} & \frac{\partial \mathscr{L}}{\partial w_{23}^2} \end{bmatrix} = \begin{bmatrix} \delta_1^2 a_1^1 & \delta_1^2 a_2^1 & \delta_1^2 a_3^1 \\ \delta_2^2 a_1^1 & \delta_2^2 a_2^1 & \delta_2^2 a_3^1 \end{bmatrix}$$

Now we can write this as an outer-product of $\boldsymbol{\delta}^2$ and $\boldsymbol{a}^1$,

$$\frac{\partial \mathscr{L}}{\partial \boldsymbol{W}^2} = \boldsymbol{\delta}^2 (\boldsymbol{a}^1)^T$$

Intermediate summary

For a giving training example $\boldsymbol{x}$, perform forward propagation to get $\boldsymbol{z}^l$ and $\boldsymbol{a}^l$ on each layer.
Then to get the partial derivatives for $\boldsymbol{W}^2$ or $\boldsymbol{W}^L$:

1. Compute $\boldsymbol{\delta}^L = \frac{\partial \mathscr{L}}{\partial a_j^L} \odot g'(\boldsymbol{z}^L)$
2. Compute $\frac{\partial \mathscr{L}}{\partial \boldsymbol{W}^L} = \boldsymbol{\delta}^L (\boldsymbol{a}^{L-1})^T$ and $\frac{\partial \mathscr{L}}{\partial \boldsymbol{b}^L} = \boldsymbol{\delta}^L$

We found very simple expressions for the derivatives with respect to the weights in the last hidden layer!

Intermediate summary

For a giving training example $\boldsymbol{x}$, perform forward propagation to get $\boldsymbol{z}^l$ and $\boldsymbol{a}^l$ on each layer.
Then to get the partial derivatives for $\boldsymbol{W}^2$ or $\boldsymbol{W}^L$:

1. Compute $\boldsymbol{\delta}^L = \frac{\partial \mathcal{L}}{\partial a_j^L} \odot g'(\boldsymbol{z}^L)$
2. Compute $\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}^L} = \boldsymbol{\delta}^L(\boldsymbol{a}^{L-1})^T$ and $\frac{\partial \mathcal{L}}{\partial \boldsymbol{b}^L} = \boldsymbol{\delta}^L$

We found very simple expressions for the derivatives with respect to the weights in the last hidden layer!
Problem: How do we do the other layers?

Intermediate summary

For a giving training example $\boldsymbol{x}$, perform forward propagation to get $\boldsymbol{z}^l$ and $\boldsymbol{a}^l$ on each layer.

Then to get the partial derivatives for $\boldsymbol{W}^2$ or $\boldsymbol{W}^L$:

1. Compute $\boldsymbol{\delta}^L = \frac{\partial \mathscr{L}}{\partial a_j^L} \odot g'(\boldsymbol{z}^L)$
2. Compute $\frac{\partial \mathscr{L}}{\partial \boldsymbol{W}^L} = \boldsymbol{\delta}^L (\boldsymbol{a}^{L-1})^T$ and $\frac{\partial \mathscr{L}}{\partial \boldsymbol{b}^L} = \boldsymbol{\delta}^L$

We found very simple expressions for the derivatives with respect to the weights in the last hidden layer!

Problem: How do we do the other layers?

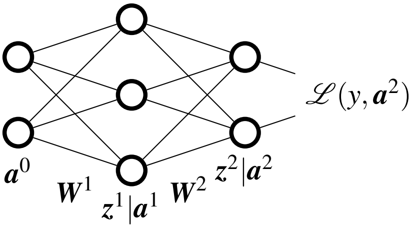
Once we knew the adjacent $\boldsymbol{\delta}^l$, we can easily compute the $\boldsymbol{\delta}^l$'s on earlier layers.

Back Propagation

But the relationship between $\mathscr{L}$ and z^1 is really complicated because of multiple passes through the activation functions.

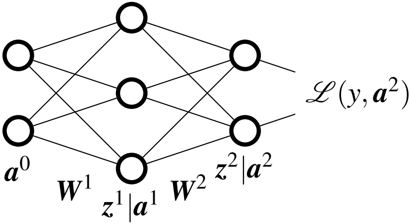
Back Propagation

But the relationship between $\mathscr{L}$ and z^1 is really complicated because of multiple passes through the activation functions.
Notice that δ^1 depends on δ^2.



Back Propagation

Notice that δ^1 depends on δ^2.

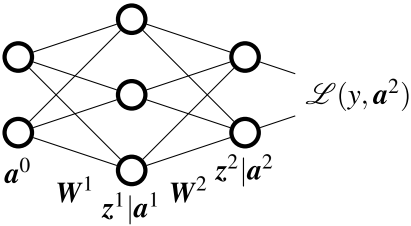


By multivariate chain rule,

$$\frac{\partial \mathcal{L}}{\partial z_k^{l-1}} = \sum_j \frac{\partial \mathcal{L}}{\partial z_j^l} \frac{\partial z_j^l}{\partial z_k^{l-1}}$$

Back Propagation

Notice that δ^1 depends on δ^2.

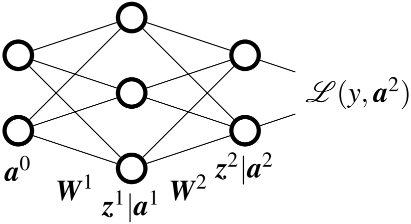


By multivariate chain rule,

$$\delta_k^{l-1} = \frac{\partial \mathscr{L}}{\partial z_k^{l-1}} = \sum_j \frac{\partial \mathscr{L}}{\partial z_j^l} \frac{\partial z_j^l}{\partial z_k^{l-1}} = \sum_j \delta_j^l \frac{\partial z_j^l}{\partial z_k^{l-1}}$$

Back Propagation

Notice that δ^1 depends on δ^2.



By multivariate chain rule,

$$\delta_2^1 = \frac{\partial \mathscr{L}}{\partial z_2^1} = \delta_1^2 \frac{\partial z_1^2}{\partial z_2^1} + \delta_2^2 \frac{\partial z_2^2}{\partial z_2^1}$$

Back Propagation

$$\delta_2^1 = \frac{\partial \mathcal{L}}{\partial z_2^1} = \delta_1^2 \frac{\partial z_1^2}{\partial z_2^1} + \delta_2^2 \frac{\partial z_2^2}{\partial z_2^1}$$

Back Propagation

$$\delta_2^1 = \frac{\partial \mathscr{L}}{\partial z_2^1} = \delta_1^2 \frac{\partial z_1^2}{\partial z_2^1} + \delta_2^2 \frac{\partial z_2^2}{\partial z_2^1}$$

Recall that $\boldsymbol{z}^2 = \boldsymbol{W}^2 a^1 + \boldsymbol{b}^2$, it follows that

$$z_i^2 = w_{i1}^2 a_1^1 + w_{i2}^2 a_2^1 + w_{i3}^2 a_3^1 + b_i^2$$

Taking the derivative $\frac{\partial z_i^2}{\partial z_2^1} = w_{i2}^2 g'(z_2^1)$, and plugging in gives

$$\delta_2^1 = \frac{\partial \mathscr{L}}{\partial z_2^1} = \delta_1^2 w_{12}^2 g'(z_2^1) + \delta_2^2 w_{22}^2 g'(z_2^1)$$

Back Propagation

If we do this for each of the 3 δ_i^1's, something nice happens:

$$
\begin{aligned}
\delta_1^1 &= \delta_1^2 w_{11}^2 g'(z_1^1) + \delta_2^2 w_{21}^2 g'(z_1^1) \\
\delta_2^1 &= \delta_1^2 w_{12}^2 g'(z_2^1) + \delta_2^2 w_{22}^2 g'(z_2^1) \\
\delta_3^1 &= \delta_1^2 w_{13}^2 g'(z_3^1) + \delta_2^2 w_{23}^2 g'(z_3^1)
\end{aligned}
$$

Back Propagation

If we do this for each of the 3 δ_i^1's, something nice happens:

$$
\begin{aligned}
\delta_1^1 &= \delta_1^2 w_{11}^2 g'(z_1^1) + \delta_2^2 w_{21}^2 g'(z_1^1) \\
\delta_2^1 &= \delta_1^2 w_{12}^2 g'(z_2^1) + \delta_2^2 w_{22}^2 g'(z_2^1) \\
\delta_3^1 &= \delta_1^2 w_{13}^2 g'(z_3^1) + \delta_2^2 w_{23}^2 g'(z_3^1)
\end{aligned}
$$

Notice that each row of the system gets multiplied by $g'(z_i^1)$, so let's factor those out.

Back Propagation

If we do this for each of the 3 δ_i^2's, something nice happens:

$$
\begin{aligned}
\delta_1^1 &= (\delta_1^2 w_{11}^2 + \delta_2^2 w_{21}^2) \cdot g'(z_1^1) \\
\delta_2^1 &= (\delta_1^2 w_{12}^2 + \delta_2^2 w_{22}^2) \cdot g'(z_2^1) \\
\delta_3^1 &= (\delta_1^2 w_{13}^2 + \delta_2^2 w_{23}^2) \cdot g'(z_3^1)
\end{aligned}
$$

Back Propagation

If we do this for each of the 3 δ_i^2's, something nice happens:

$$\begin{aligned}
\delta_1^1 &= (\delta_1^2 w_{11}^2 + \delta_2^2 w_{21}^2) \cdot g'(z_1^1) \\
\delta_2^1 &= (\delta_1^2 w_{12}^2 + \delta_2^2 w_{22}^2) \cdot g'(z_2^1) \\
\delta_3^1 &= (\delta_1^2 w_{13}^2 + \delta_2^2 w_{23}^2) \cdot g'(z_3^1)
\end{aligned}$$

Remember $\boldsymbol{\delta}^2 = \begin{bmatrix} \delta_1^2 \\ \delta_2^2 \end{bmatrix}$, $\boldsymbol{W}^2 = \begin{bmatrix} w_{11}^2 & w_{12}^2 & w_{13}^2 \\ w_{21}^2 & w_{22}^2 & w_{23}^2 \end{bmatrix}$

Back Propagation

If we do this for each of the 3 δ_i^2's, something nice happens:

$$\begin{aligned}
\delta_1^1 &= (\delta_1^2 w_{11}^2 + \delta_2^2 w_{21}^2) \cdot g'(z_1^1) \\
\delta_2^2 &= (\delta_1^2 w_{12}^2 + \delta_2^2 w_{22}^2) \cdot g'(z_2^1) \\
\delta_3^2 &= (\delta_1^2 w_{13}^2 + \delta_2^2 w_{23}^2) \cdot g'(z_3^1)
\end{aligned}$$

$$(\boldsymbol{W}^2)^T = \begin{bmatrix} w_{11}^2 & w_{21}^2 \\ w_{12}^2 & w_{22}^2 \\ w_{13}^2 & w_{23}^2 \end{bmatrix}, \boldsymbol{\delta}^2 = \begin{bmatrix} \delta_1^2 \\ \delta_2^2 \end{bmatrix}.$$

Back Propagation

If we do this for each of the 3 δ_i^2's, something nice happens:

$$
\begin{aligned}
\delta_1^1 &= (\delta_1^2 w_{11}^2 + \delta_2^2 w_{21}^2) \cdot g'(z_1^1) \\
\delta_2^1 &= (\delta_1^2 w_{12}^2 + \delta_2^2 w_{22}^2) \cdot g'(z_2^1) \\
\delta_3^1 &= (\delta_1^2 w_{13}^2 + \delta_2^2 w_{23}^2) \cdot g'(z_3^1)
\end{aligned}
$$

$$\boldsymbol{\delta}^1 = (\boldsymbol{W}^2)^T \boldsymbol{\delta}^2 \odot g'(\boldsymbol{z}^1)$$

Back Propagation

We can easily compute $\boldsymbol{\delta}^1$ from $\boldsymbol{\delta}^2$
Then we can compute derivatives of $\mathscr{L}$ w.r.t. weights $\boldsymbol{W}^1$ and biases $\boldsymbol{b}^1$ exactly the way we did for $\boldsymbol{W}^2$ and biases $\boldsymbol{b}^2$

1. Compute $\boldsymbol{\delta}^1 = (\boldsymbol{W}^2)^T \boldsymbol{\delta}^2 \odot g'(\boldsymbol{z}^1)$
2. Compute $\frac{\partial \mathscr{L}}{\partial \boldsymbol{W}^1} = \boldsymbol{\delta}^1 (\boldsymbol{a}^0)^T$ and $\frac{\partial \mathscr{L}}{\partial \boldsymbol{b}^1} = \boldsymbol{\delta}^1$

Back Propagation

We can easily compute $\boldsymbol{\delta}^1$ from $\boldsymbol{\delta}^2$

Then we can compute derivatives of $\mathscr{L}$ w.r.t. weights $\boldsymbol{W}^1$ and biases $\boldsymbol{b}^1$ exactly the way we did for $\boldsymbol{W}^2$ and biases $\boldsymbol{b}^2$

1. Compute $\boldsymbol{\delta}^1 = (\boldsymbol{W}^2)^T \boldsymbol{\delta}^2 \odot g'(\boldsymbol{z}^1)$
2. Compute $\frac{\partial \mathscr{L}}{\partial \boldsymbol{W}^1} = \boldsymbol{\delta}^1 (\boldsymbol{a}^0)^T$ and $\frac{\partial \mathscr{L}}{\partial \boldsymbol{b}^1} = \boldsymbol{\delta}^1$

We've worked this out for a simple network with one hidden layer.

Nothing we've done assumed anything about the number of layers, so we can apply the same procedure recursively with any number of layers.

Back Propagation

$\delta^L = \frac{\partial \mathscr{L}}{\partial a_j^L} \odot g'(\mathbf{z}^L)$ # Compute δ's on output layer

For $\ell = L, \ldots, 1$

$\frac{\partial \mathscr{L}}{\partial \mathbf{W}^\ell} = \boldsymbol{\delta}^\ell (\boldsymbol{a}^{l-1})^T$ # Compute weight derivatives

$\frac{\partial \mathscr{L}}{\partial \boldsymbol{b}^\ell} = \boldsymbol{\delta}^\ell$ # Compute bias derivatives

$\boldsymbol{\delta}^{\ell-1} = \left(W^\ell\right)^T \boldsymbol{\delta}^\ell \odot g'(\mathbf{z}^{\ell-1})$ # Back prop $\boldsymbol{\delta}$'s to previous layer

(After this, ready to do a SGD update on weights/biases)

Training a Feed-Forward Neural Network

Given initial guess for weights and biases.
Loop over each training example in random order:

1. Forward propagate to get activations on each layer
2. Back propagate to get derivatives
3. Update weights and biases via stochastic gradient descent
4. Repeat