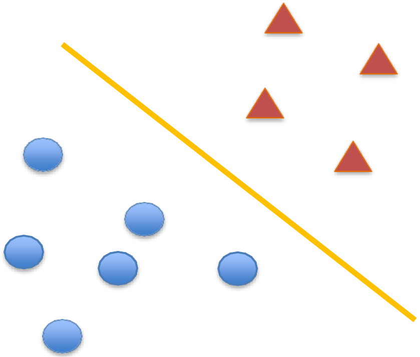
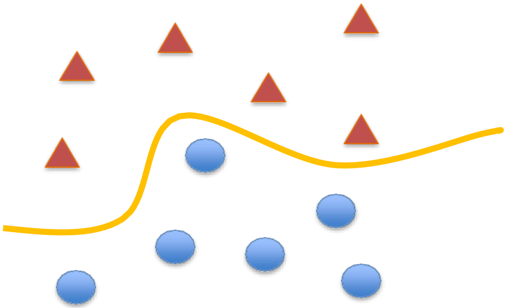
L8 – Neural networks

Non-linear models: Neural networks

Linear Model
(Perceptron)



Need a non-linear model!



Non-linear models: Neural networks

What should we do in case the target is not linearly separable?

- Use **multiple linear functions** and combine them using **non-linear activation functions**

activation function

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

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

$$x \in \mathbb{R}^D, W_1 \in \mathbb{R}^{H \times D}, W_2 \in \mathbb{R}^{C \times H}$$

Non-linear models: Neural networks

- A "neuron" computes a linear function $\boldsymbol{f} = \boldsymbol{W}\boldsymbol{x}$

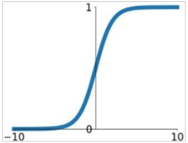
- Neural network is a nesting of 'functions'
  - 2-layers: $\boldsymbol{f} = \boldsymbol{W}_2 \boxed{\max(\mathbf{0}, \boldsymbol{W}_1\boldsymbol{x})}$ ← **activation function**
  - 3-layers: $\boldsymbol{f} = \boldsymbol{W}_3 \max(\mathbf{0}, \boldsymbol{W}_2 \max(\mathbf{0}, \boldsymbol{W}_1\boldsymbol{x}))$
  - 4-layers: $\boldsymbol{f} = \boldsymbol{W}_4 \tanh(\boldsymbol{W}_3, \max(\mathbf{0}, \boldsymbol{W}_2 \max(\mathbf{0}, \boldsymbol{W}_1\boldsymbol{x})))$
  - 5-layers: $\boldsymbol{f} = \boldsymbol{W}_5 \sigma(\boldsymbol{W}_4 \tanh(\boldsymbol{W}_3, \max(\mathbf{0}, \boldsymbol{W}_2 \max(\mathbf{0}, \boldsymbol{W}_1\boldsymbol{x}))))$
  - … up to hundreds of layers

Non-linear models: Neural networks

Activation functions

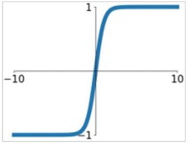
Sigmoid

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



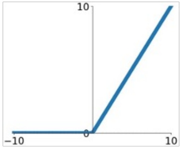
tanh

$\tanh(x)$



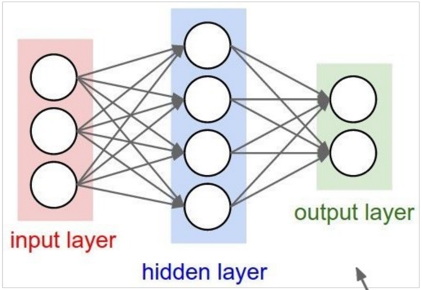
ReLU

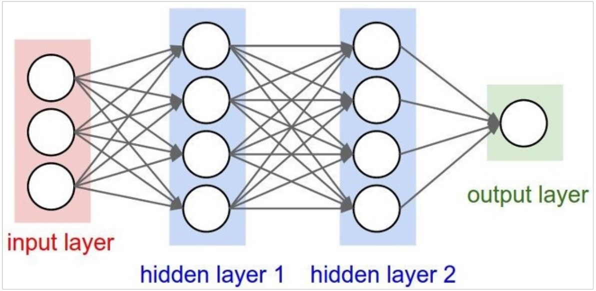
$\max(0, x)$



Non-linear models: Neural networks

Neural networks: Architectures



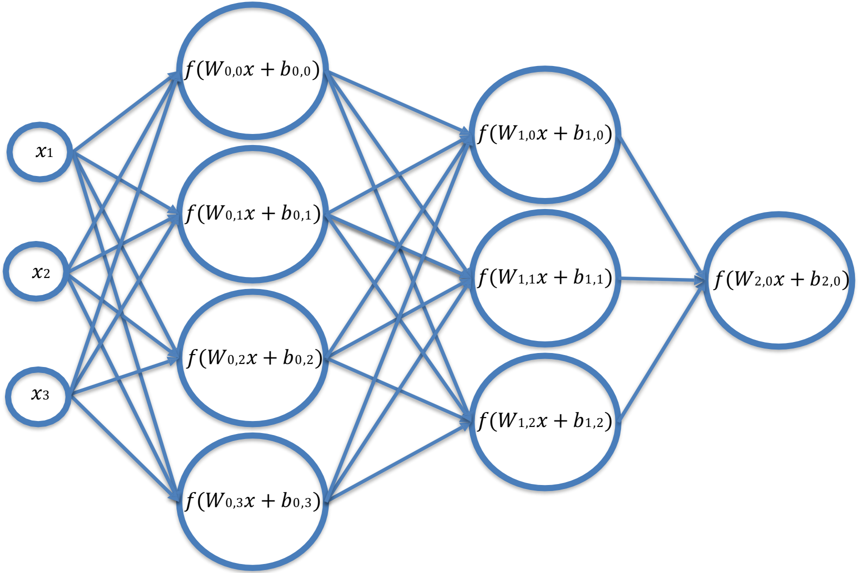


"2-layer Neural Net", or
"1-hidden-layer Neural Net"

"3-layer Neural Net", or
"2-hidden-layer Neural Net"

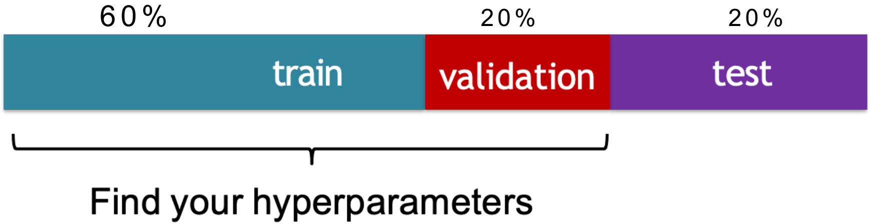
"Fully-connected" layers

Non-linear models: Neural networks



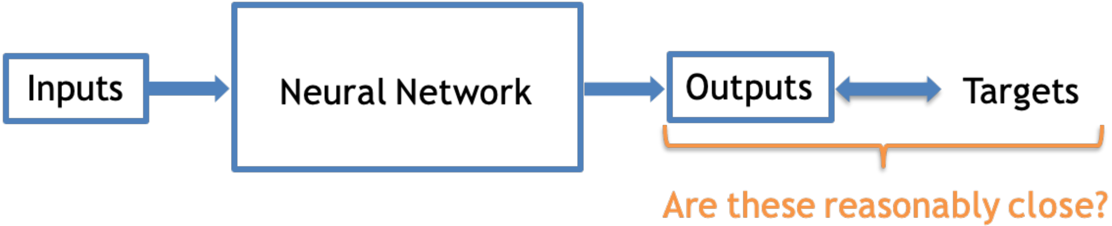
Training recipe for MLP

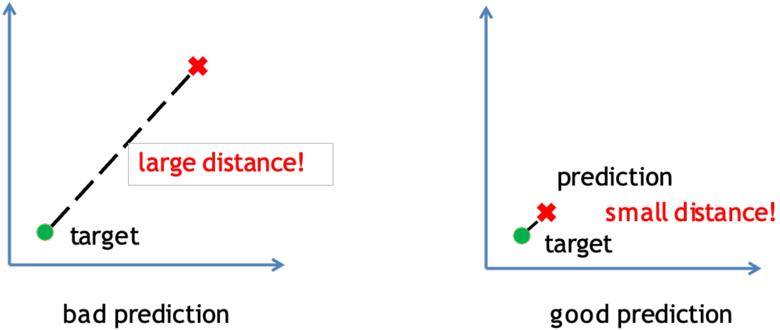
- Split your data



Other splits are also possible (e.g., 80% / 10% / 10%)

Training recipe for MLP





- Select a suitable loss function

Loss functions for classification

Binary cross entropy loss

$$\mathcal{L}_{\mathrm{BCE}} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)\right]$$

- N is the total number of samples
- y_i is the true label for the i-th sample (either 0 or 1).
- $\hat{y}_i$ is the predicted probability for the i-th sample.

Yes! (0.8)

No! (0.2)

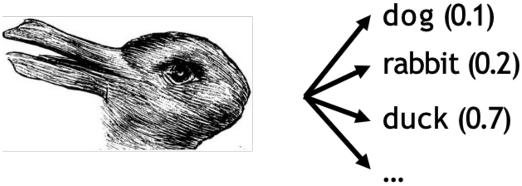
The network predicts the probability of the input belonging to the "yes" class

Loss functions for classification

Cross entropy loss

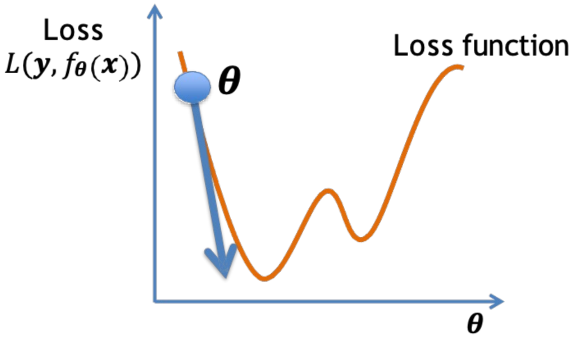
$$\mathcal{L}_{\mathrm{CE}} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log(\hat{y}_{i,c})$$

- N is the total number of samples
- $y_{i,c}$ is a binary indicator (1 if the true class of sample i is c, 0 otherwise
- $\hat{y}_i$ is the predicted probability for the i-th sample.
- C is the number of classes



Training a neural net

- Minimize: $L(\boldsymbol{y}, f_{\boldsymbol{\theta}}(\boldsymbol{x}))$ w.r.t. $\boldsymbol{\theta}$
- We use gradient-based optimization: **gradient descent**



$$\boldsymbol{\theta}^* = \arg\min L\big(\boldsymbol{y}, f_{\boldsymbol{\theta}}(\boldsymbol{x})\big)$$

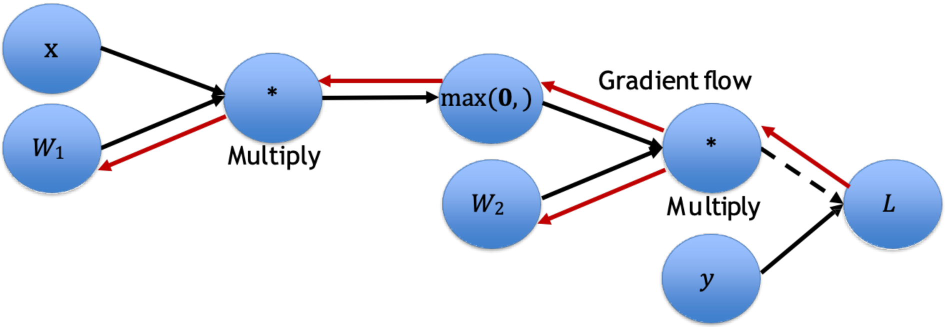
Iterative update:

Learning rate

$$\boldsymbol{\theta} = \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} L\big(\boldsymbol{y}, f_{\boldsymbol{\theta}}(\boldsymbol{x})\big)$$

Training a neural net

- Given inputs $\boldsymbol{x}$ and targets $\boldsymbol{y}$
- Given multi-layer NN
  - Need to propagate gradients from end to first layer ($\boldsymbol{W_1}$)
- Backpropagation: Use chain rule to compute gradients



Non-linear models: Neural networks

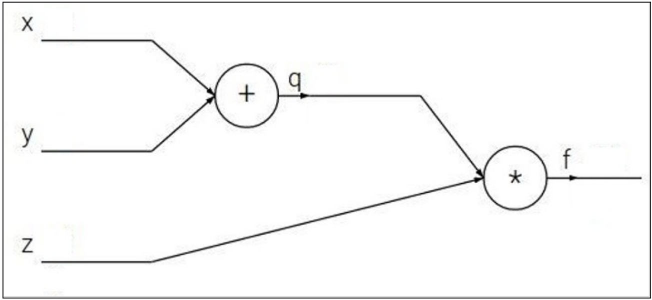
The flow of computations in a neural network goes in two ways:

1. **Left-to-right**: This is referred to as ***forward propagation***, which results in computing the output of the network
2. **Right-to-left**: This is referred to as ***back propagation***, which results in computing the gradients (or derivatives) of the parameters in the network

Backpropagation

Backpropagation: a simple example

$$f(x, y, z) = (x + y)z$$

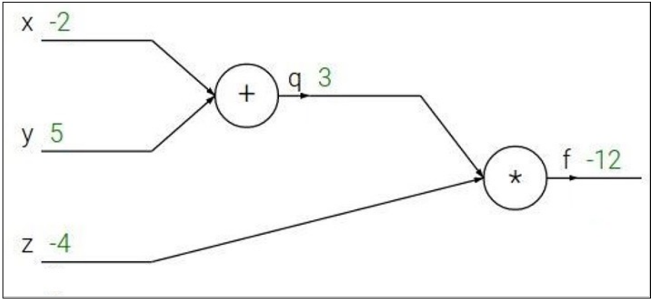


Backpropagation

Backpropagation: a simple example

$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4



Backpropagation

Backpropagation: a simple example

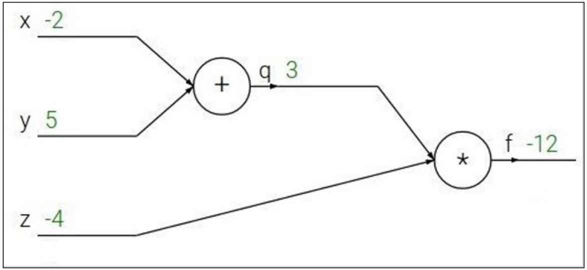
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Backpropagation

Backpropagation: a simple example

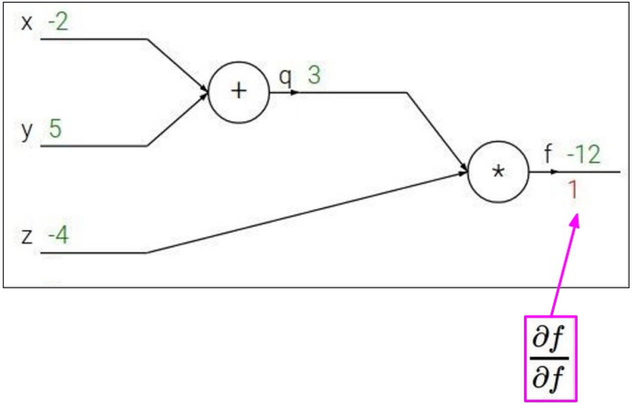
$$f(x,y,z) = (x+y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



Backpropagation

Backpropagation: a simple example

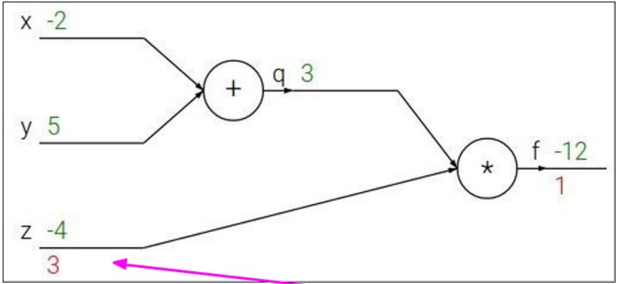
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$$\frac{\partial f}{\partial z}$$

Backpropagation

Backpropagation: a simple example

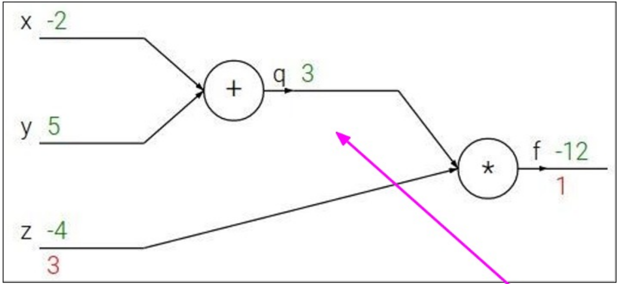
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



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

Backpropagation

Backpropagation: a simple example

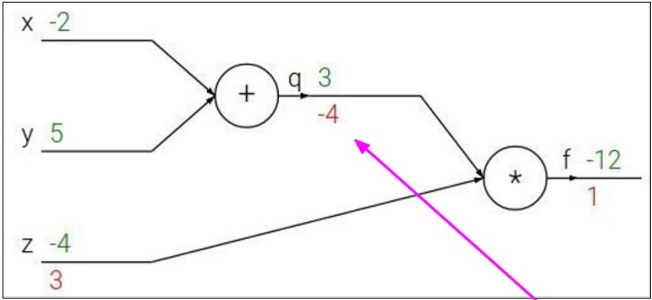
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



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

Backpropagation

Backpropagation: a simple example

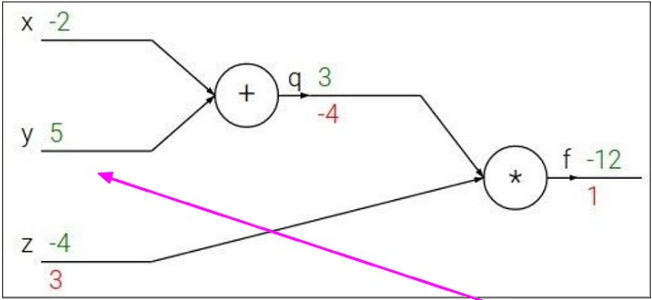
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$$\frac{\partial f}{\partial y}$$

Chain rule:

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

Backpropagation

Backpropagation: a simple example

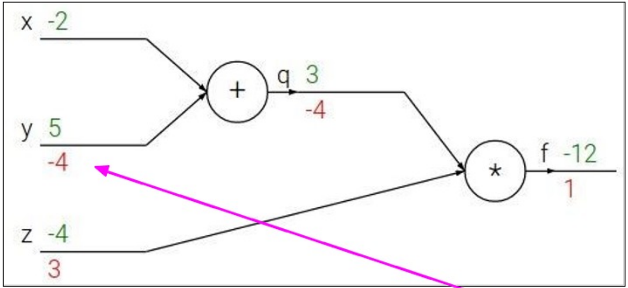
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



$$\boxed{\frac{\partial f}{\partial y}}$$

Chain rule:

$$\boxed{\frac{\partial f}{\partial y} = \frac{\partial f}{\partial q}\frac{\partial q}{\partial y}}$$

Upstream gradient

Local gradient

Backpropagation

Backpropagation: a simple example

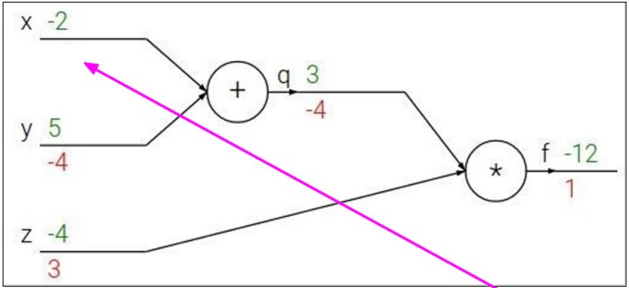
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $$\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$$



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

Chain rule:

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

Upstream gradient

Local gradient

Backpropagation

Backpropagation: a simple example

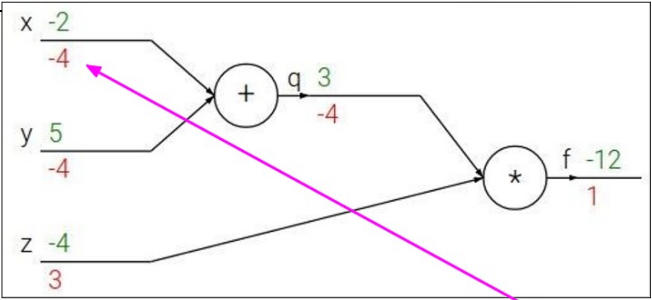
$$f(x, y, z) = (x + y)z$$

e.g. x = -2, y = 5, z = -4

$$q = x + y \qquad \frac{\partial q}{\partial x} = 1, \frac{\partial q}{\partial y} = 1$$

$$f = qz \qquad \frac{\partial f}{\partial q} = z, \frac{\partial f}{\partial z} = q$$

Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$



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

Chain rule:

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

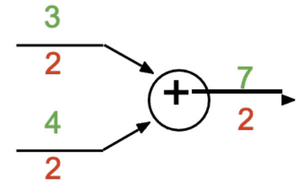
Upstream gradient

Local gradient

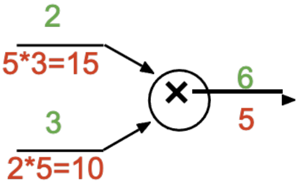
Backpropagation

Patterns in gradient flow

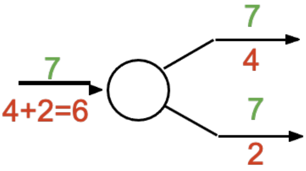
add gate: gradient distributor



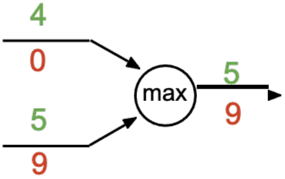
mul gate: “swap multiplier”



copy gate: gradient adder



max gate: gradient router



Backpropagation

If we have a linear combination

$$W \cdot x = q$$

W matrix
x matrix or vector

Using the chain rule we can compute the gradients:

$$\boxed{\begin{aligned} \nabla f_W &= \nabla f_q \cdot x^T \\ \nabla f_x &= W^T \cdot \nabla f_q \end{aligned}}$$

Inference

At test time / validation time

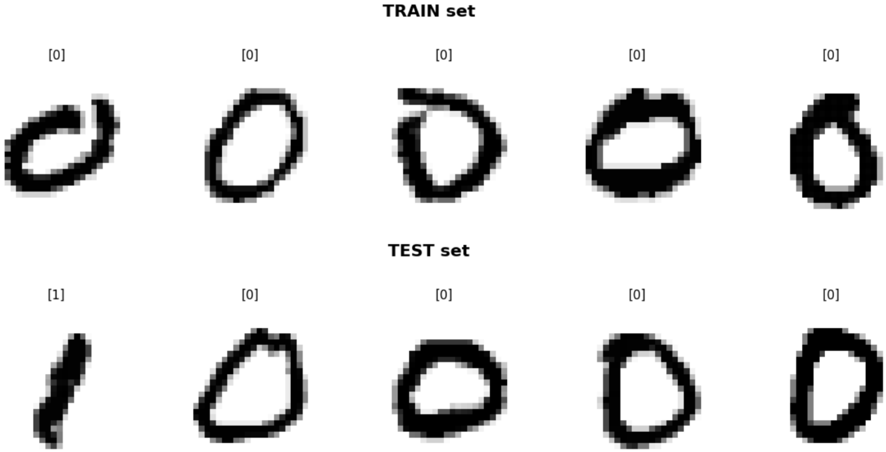
- Perform the forward pass and get the probabilities

- Binary classification: Threshold the probability to find the predicted class

$$\begin{cases} \text{if } p(y=1|x;\theta) < 0.5 \text{ predict class } 0 \\ \text{if } p(y=1|x;\theta) \geq 0.5 \text{ predict class } 1 \end{cases}$$

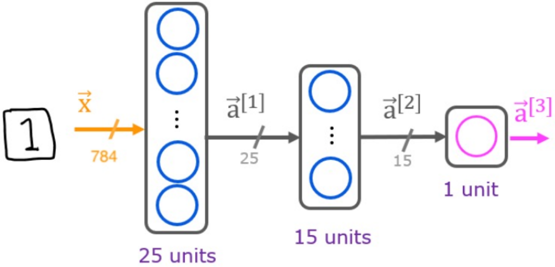
Exercises

Classify digits on the MNIST dataset



Exercises

Implement this architecture and compute using backpropagation the gradients of the parameters



- Train the network on the training set
- Compute the accuracy on the test set