Deep Learning Basics

Marcel & Nicolas

Introduction to Back Propagation and Optimization

```
model.compile(optimizer='adam',
loss='sparse_categorical_crossentropy', metrics=['accuracy'])
```

Standard Neural Network Training Objective...

- The standard neural network training objective is given by:

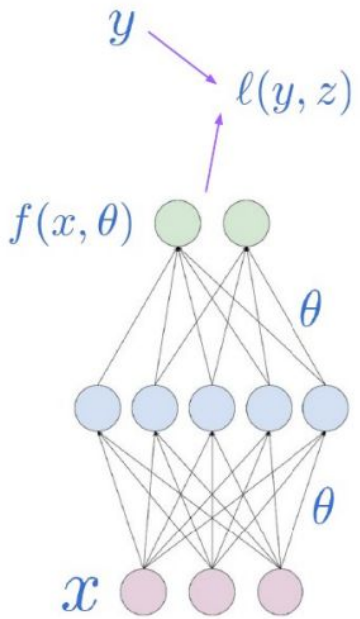
$$h(\theta) = \frac{1}{m} \sum_{i=1}^{m} \ell(y_i, f(x_i, \theta))$$

where:

$\ell(y, z)$ is a loss function measuring disagreement between y and z

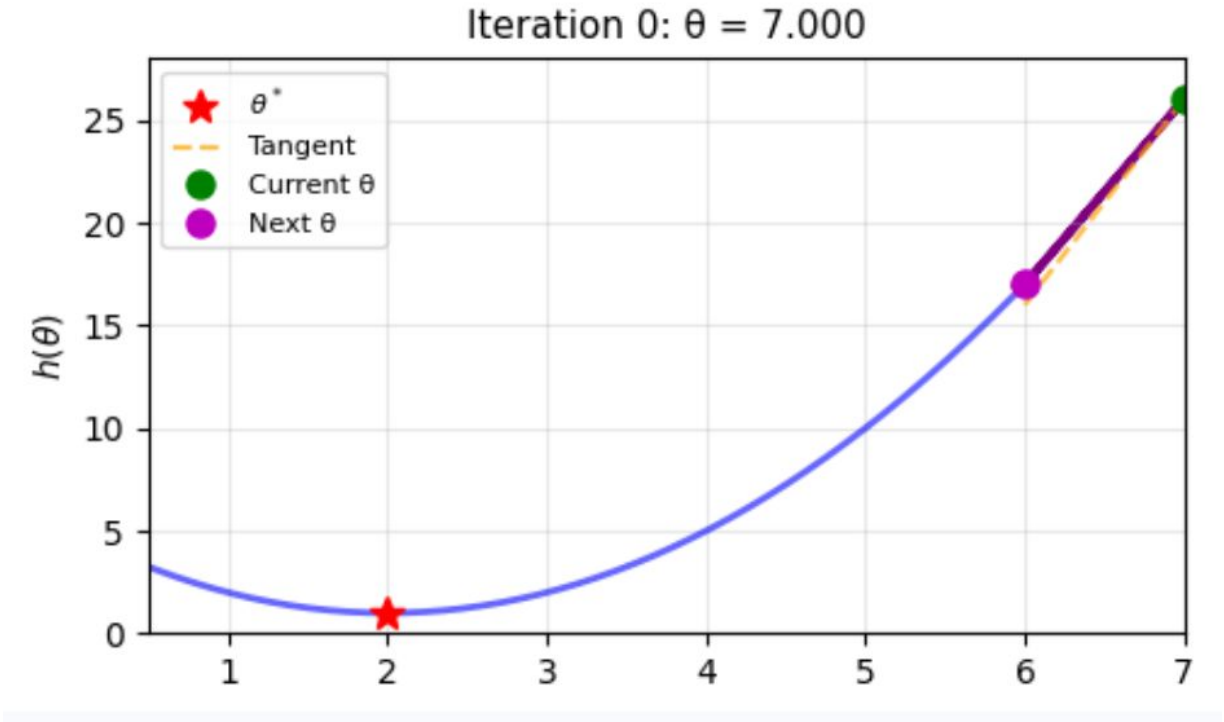
and

$f(x, \theta)$ is a neural network function taking input x and outputing some prediction



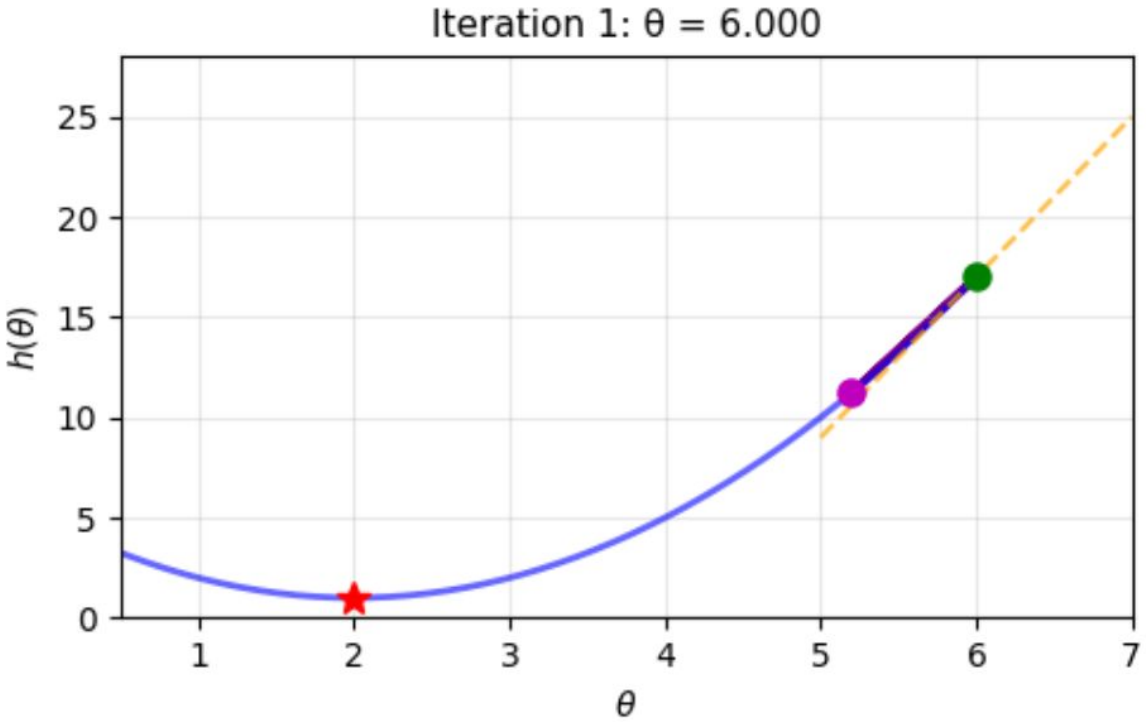
Gradient Descent

$$\theta_{k+1} = \theta_k - \alpha_k \nabla h(\theta_k)$$



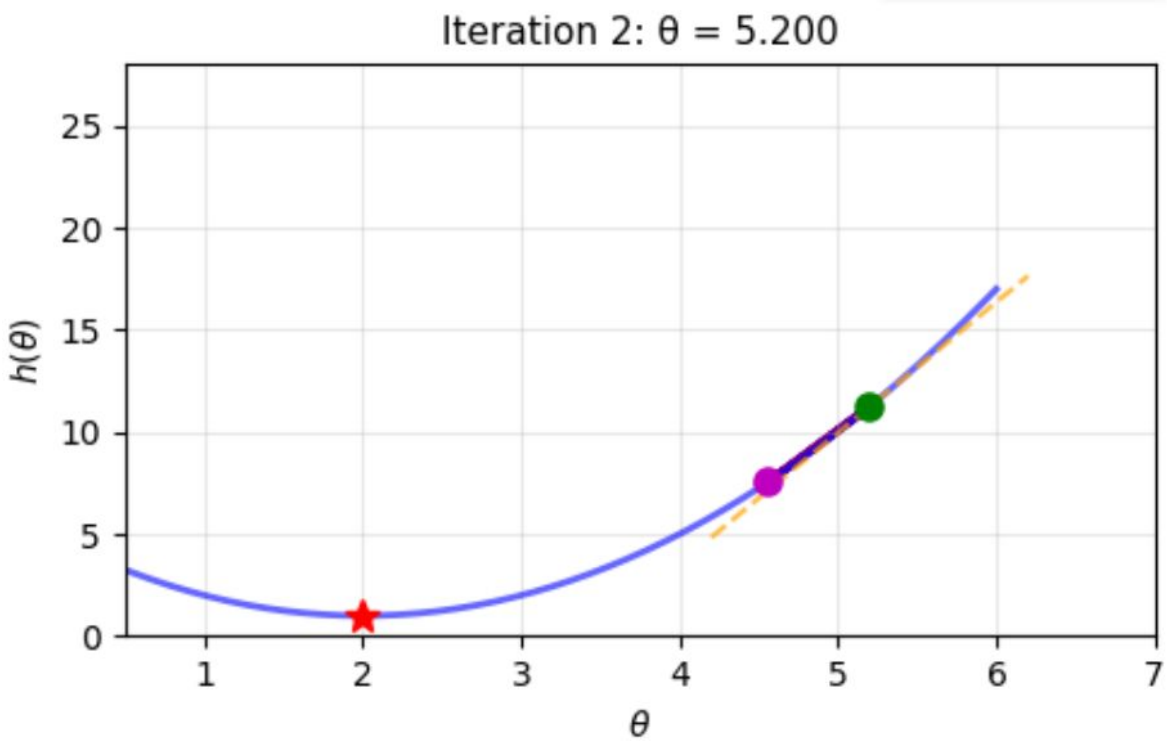
Gradient Descent

$$\theta_{k+1} = \theta_k - \alpha_k \nabla h(\theta_k)$$



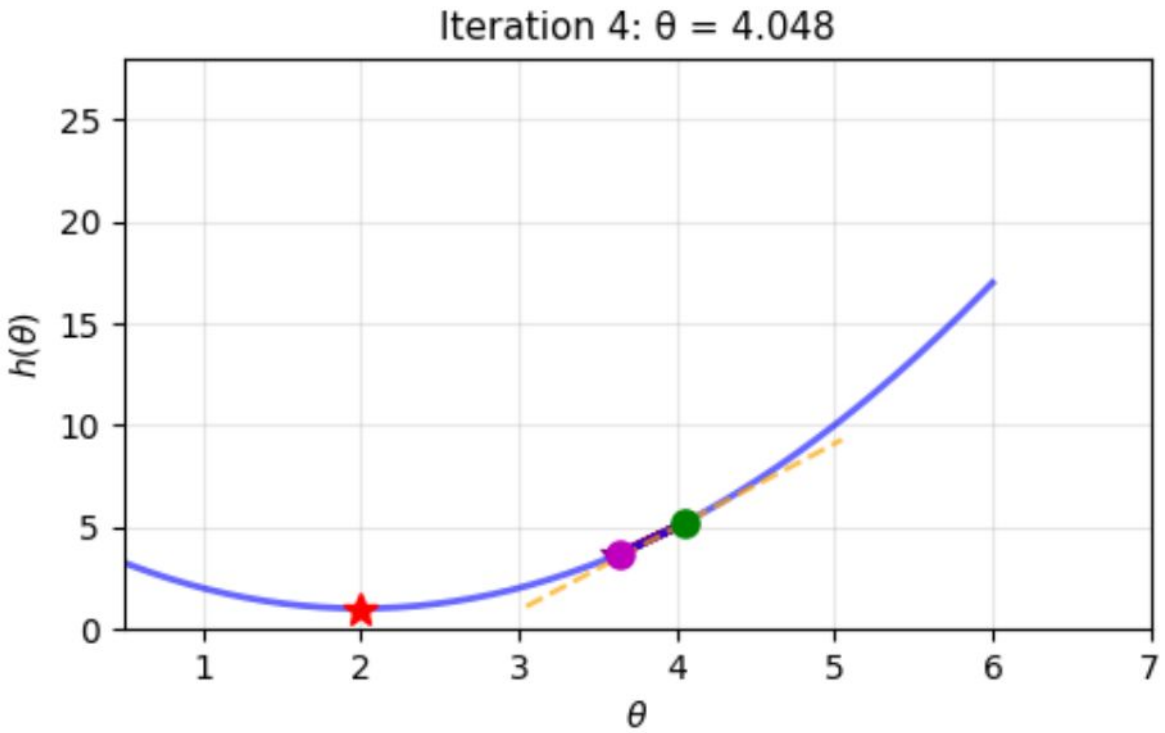
Gradient Descent

$$\theta_{k+1} = \theta_k - \alpha_k \nabla h(\theta_k)$$



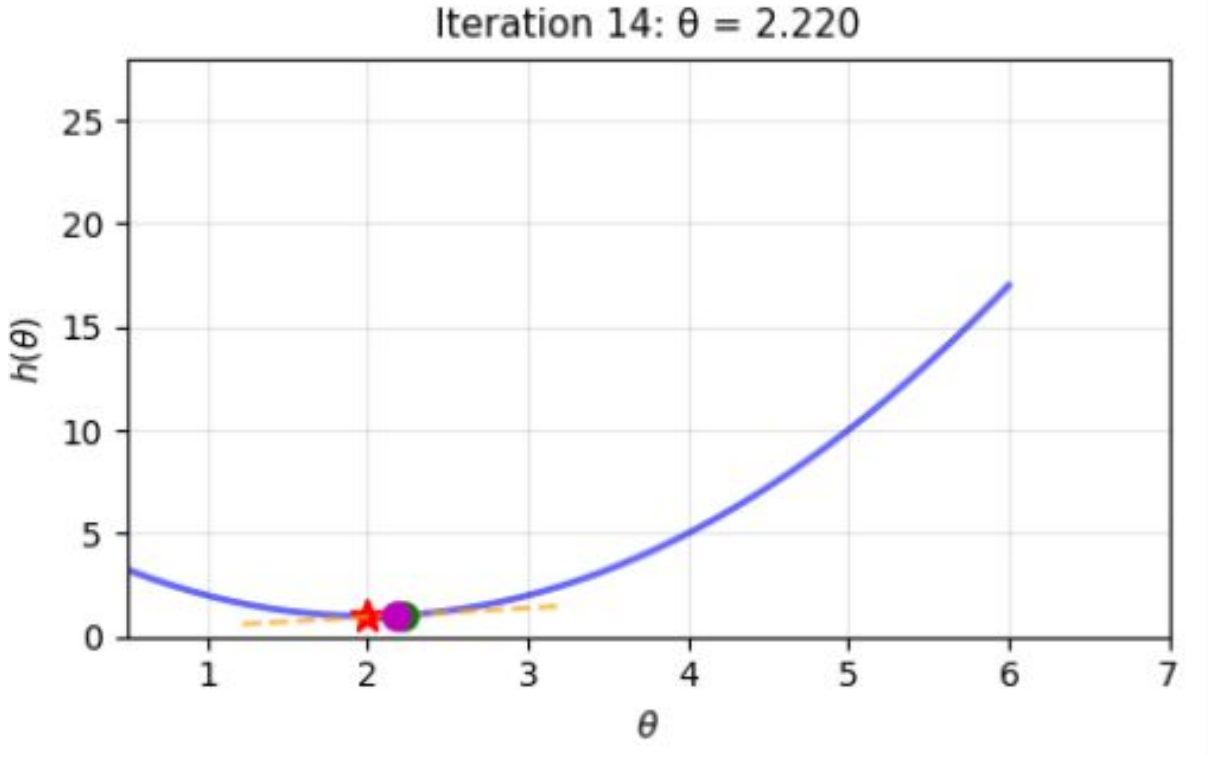
Gradient Descent

$$\theta_{k+1} = \theta_k - \alpha_k \nabla h(\theta_k)$$



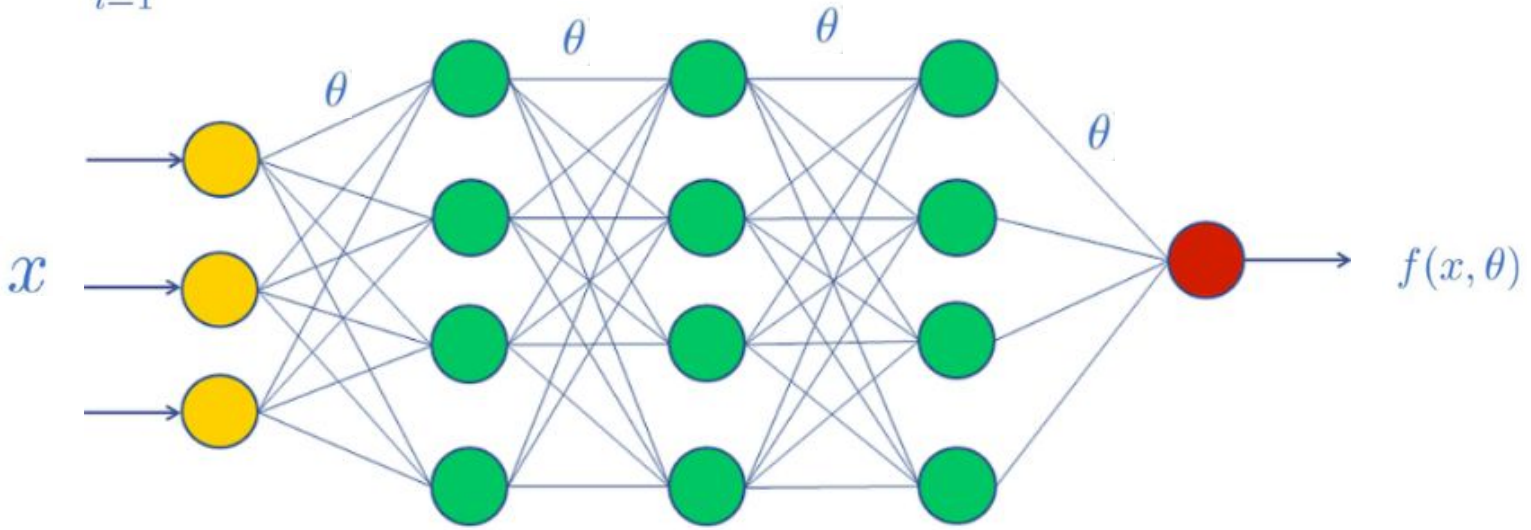
Gradient Descent

$$\theta_{k+1} = \theta_k - \alpha_k \nabla h(\theta_k)$$



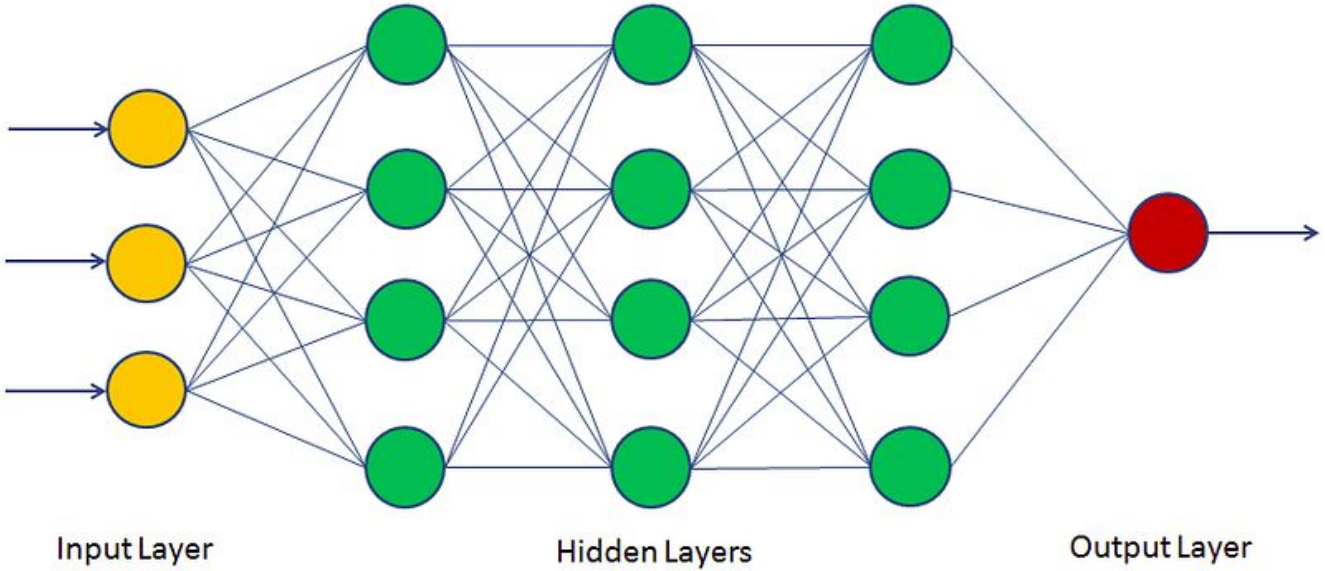
The problem

$$h(\theta) = \frac{1}{m}\sum_{i=1}^{m} \ell(y_i, f(x_i, \theta))$$



The problem

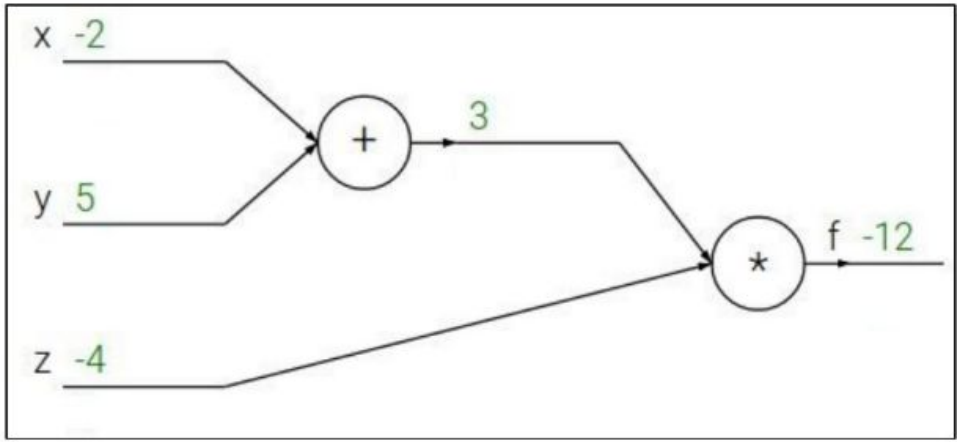
$$h(\theta) = \frac{1}{m}\sum_{i=1}^{m} \ell(y_i, f(x_i, \theta))$$



Backpropagation

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

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



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

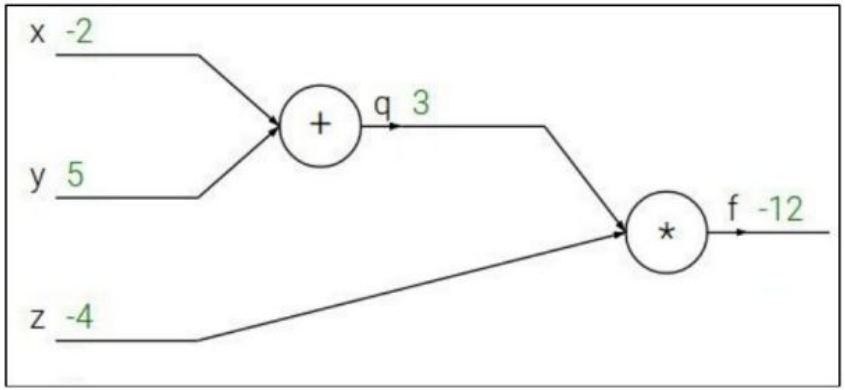
Backpropagation

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

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

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

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



Backpropagation

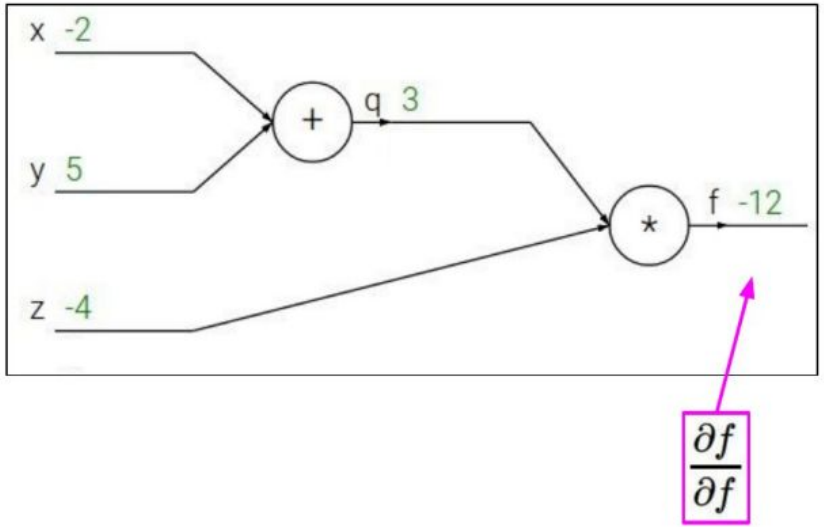
$$f(x, y, z) = (x + y)z$$

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



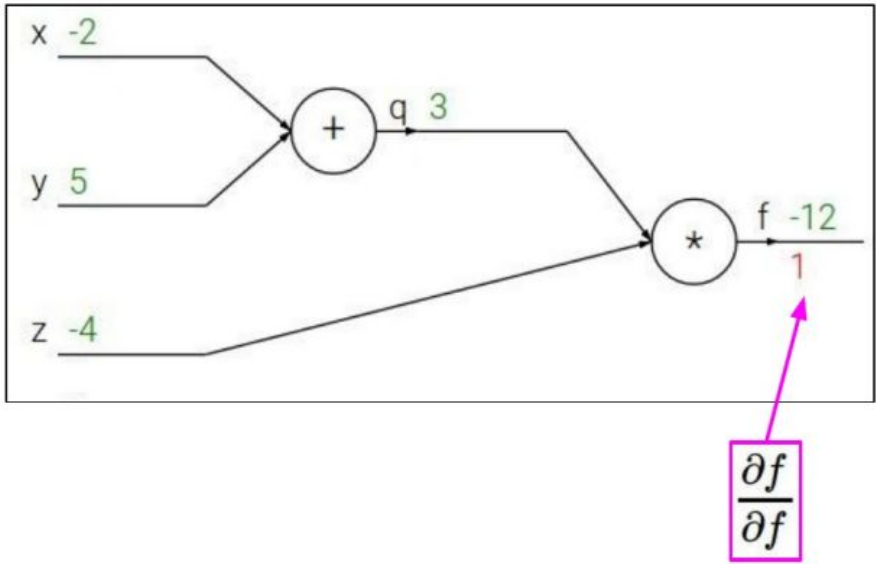
Backpropagation

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

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

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

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



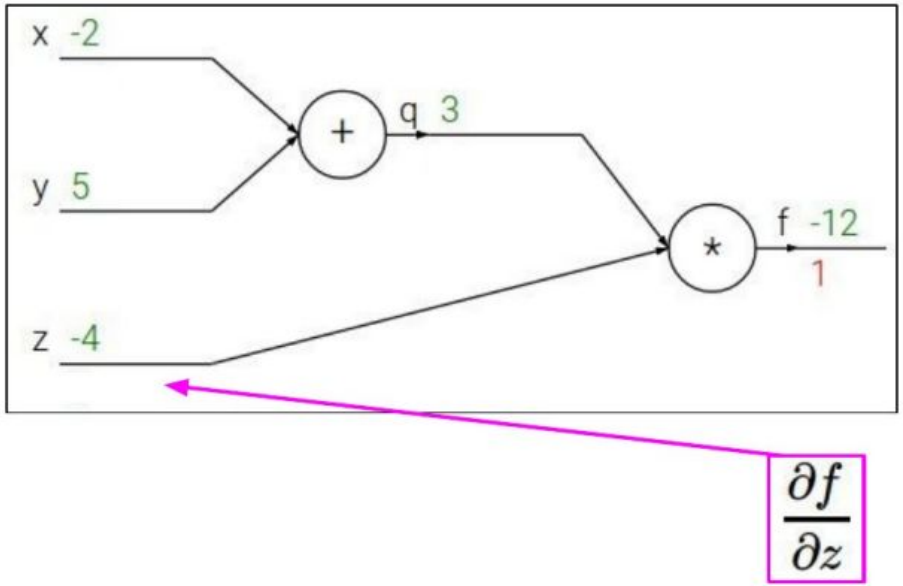
Backpropagation

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

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

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

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



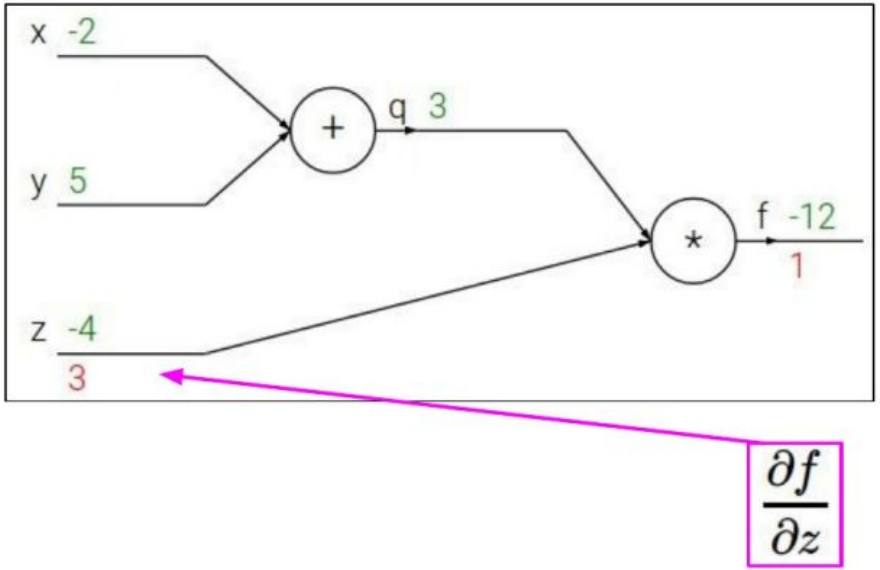
Backpropagation

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

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

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

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



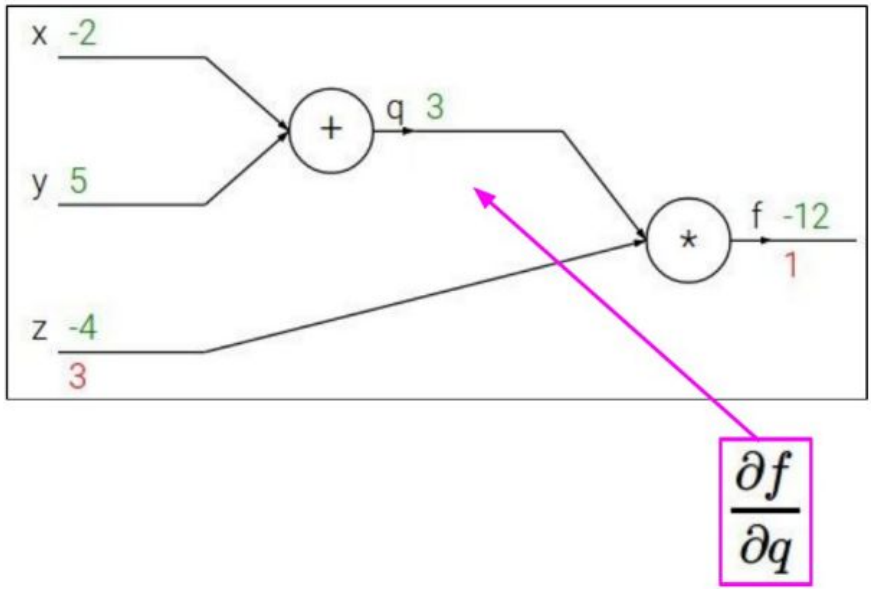
Backpropagation

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

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

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

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



Backpropagation

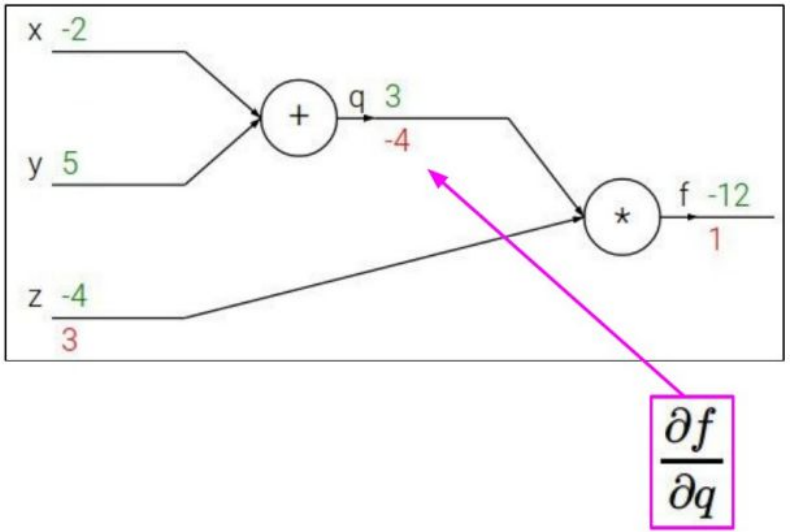
$$f(x, y, z) = (x + y)z$$

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

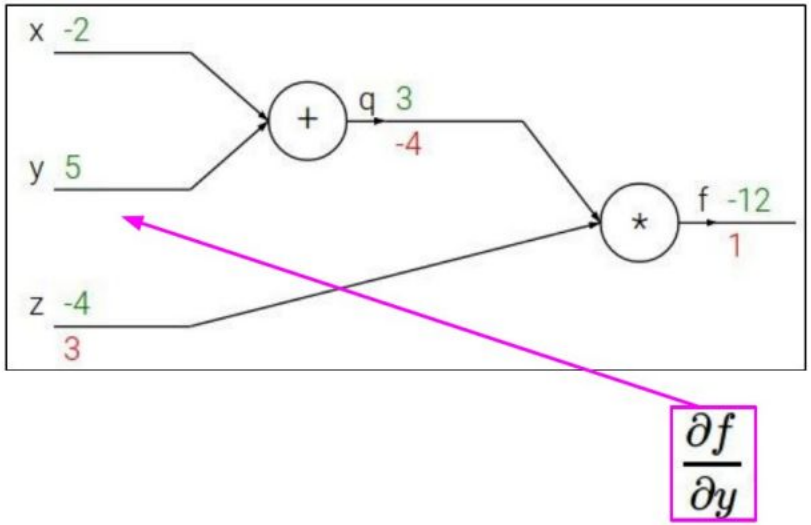
$$f(x, y, z) = (x + y)z$$

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

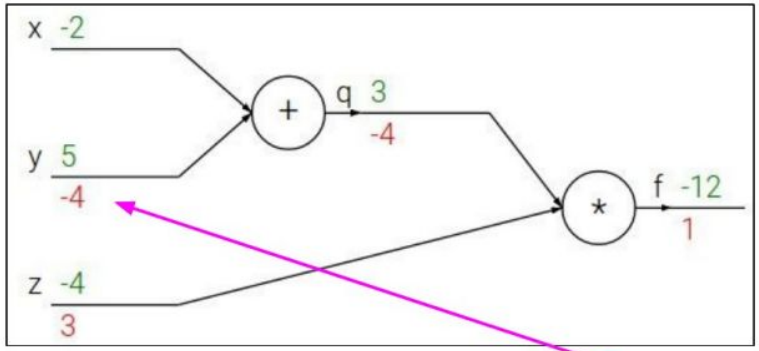
$$f(x, y, z) = (x + y)z$$

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



Chain rule:

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

Backpropagation

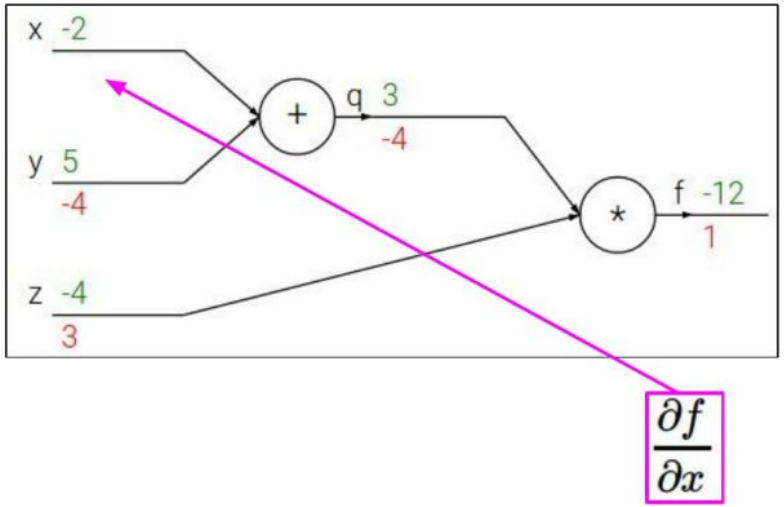
$$f(x, y, z) = (x + y)z$$

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

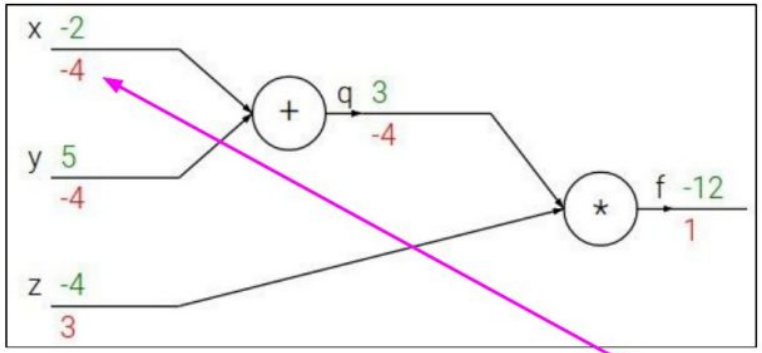
$$f(x, y, z) = (x + y)z$$

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

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

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

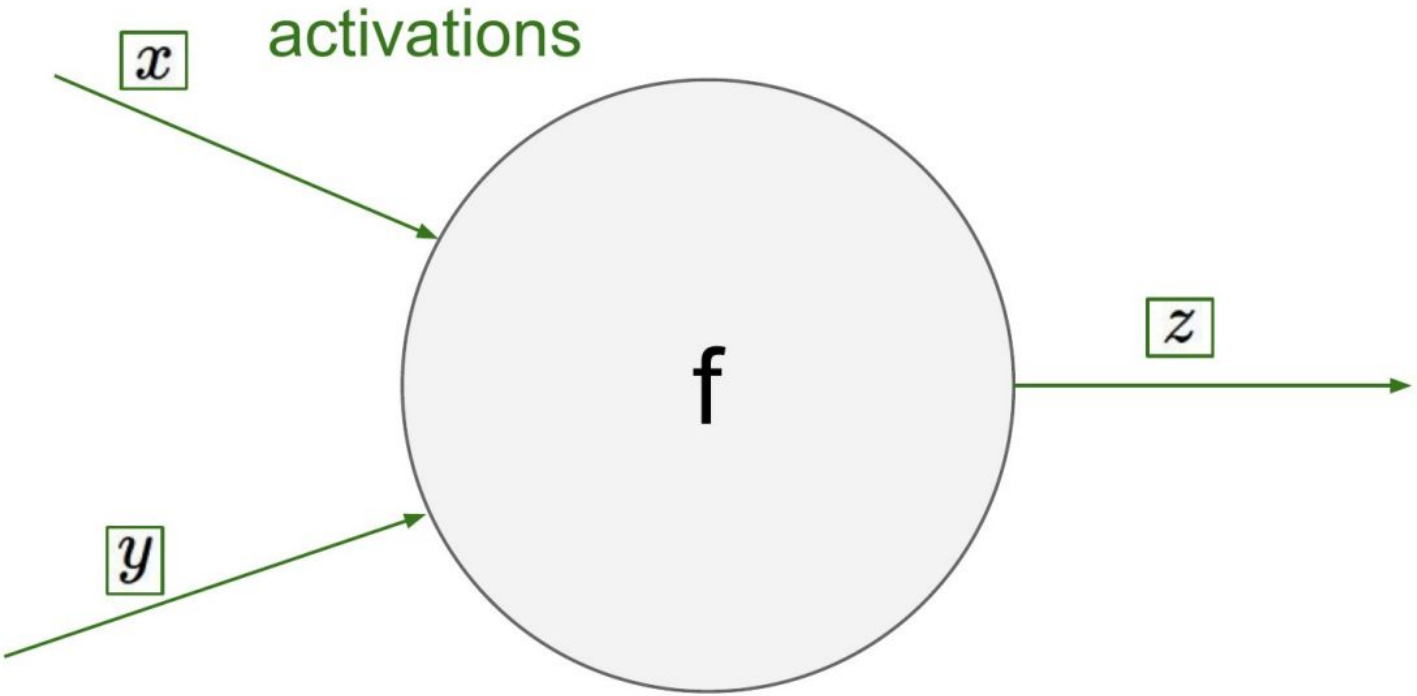
Want: $\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}$

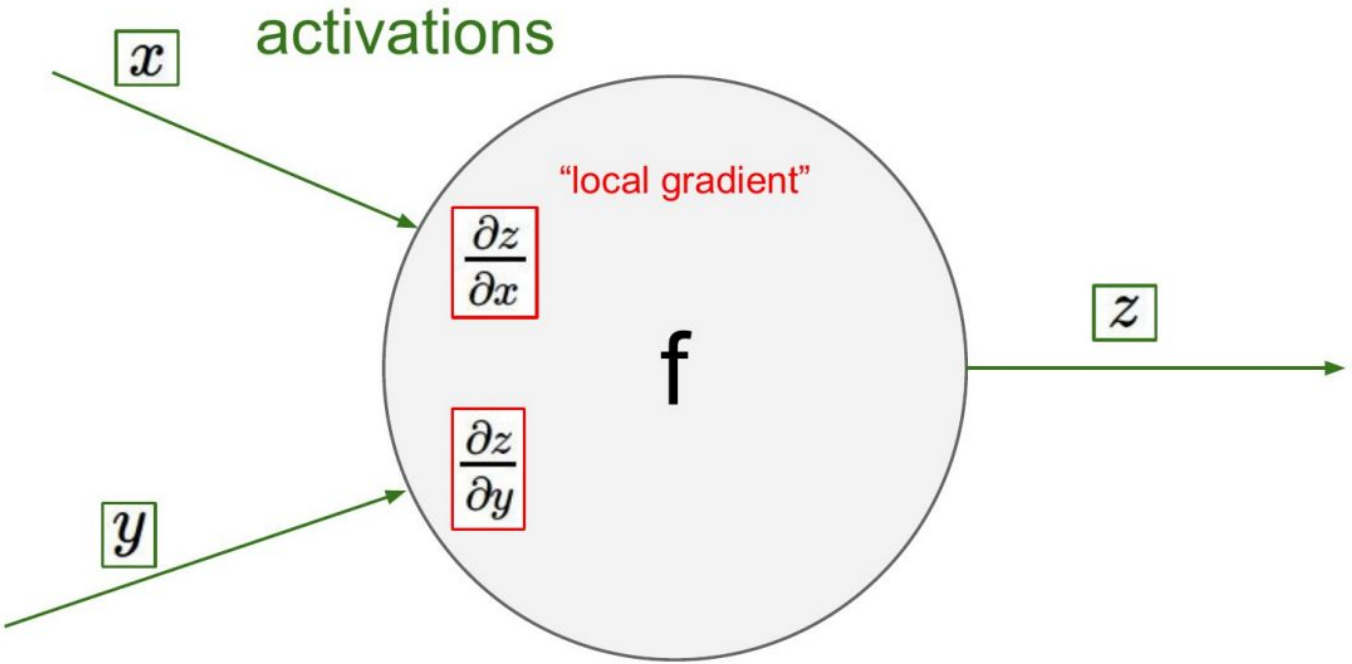


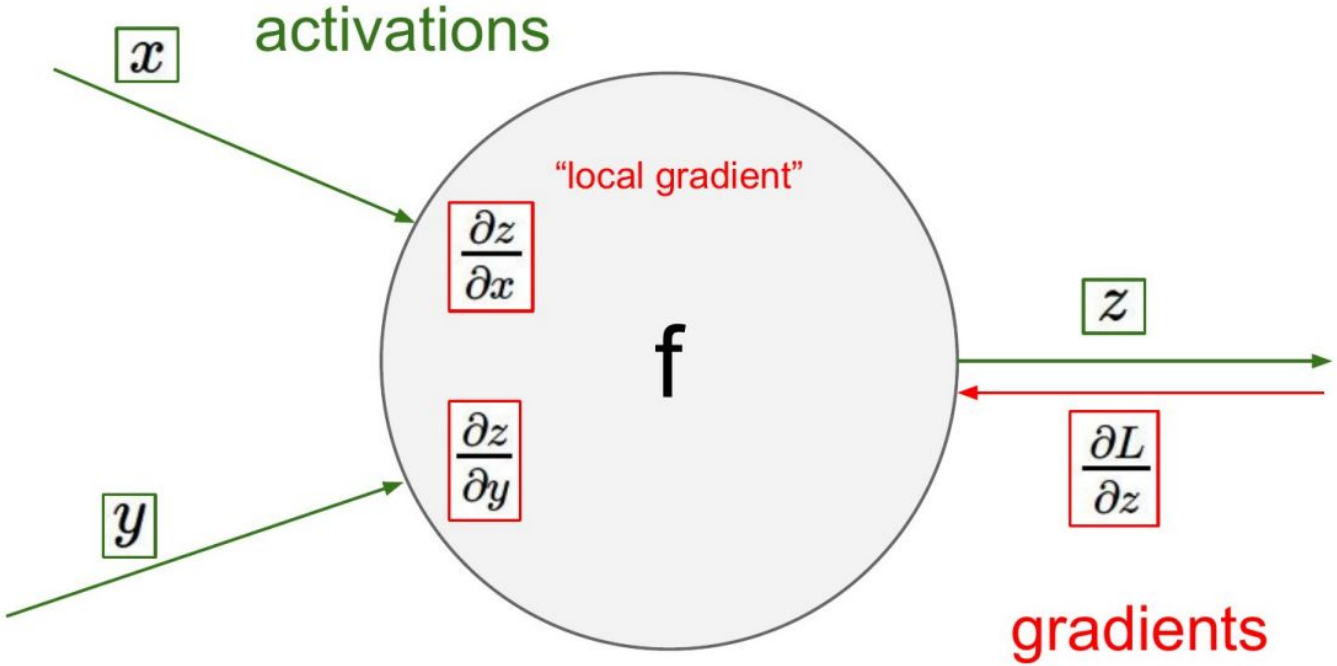
$\frac{\partial f}{\partial x}$

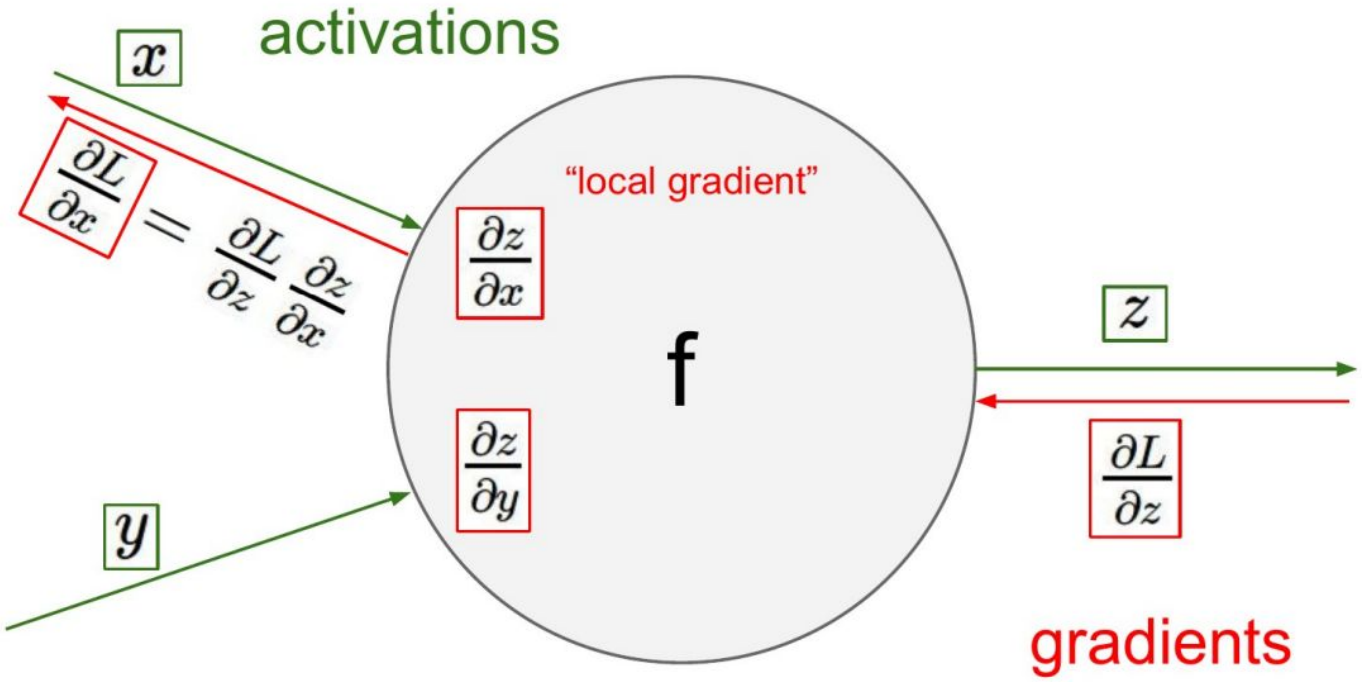
Chain rule:

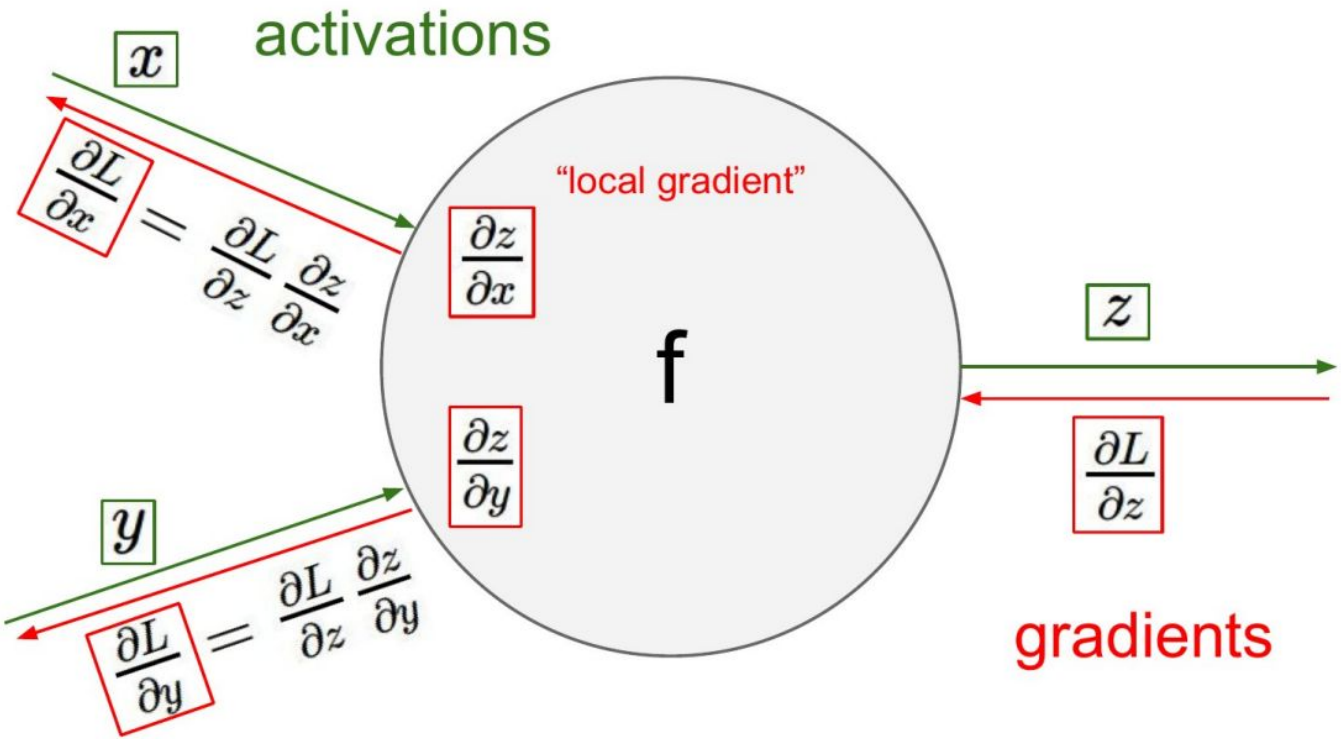
$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial q} \frac{\partial q}{\partial x}$$

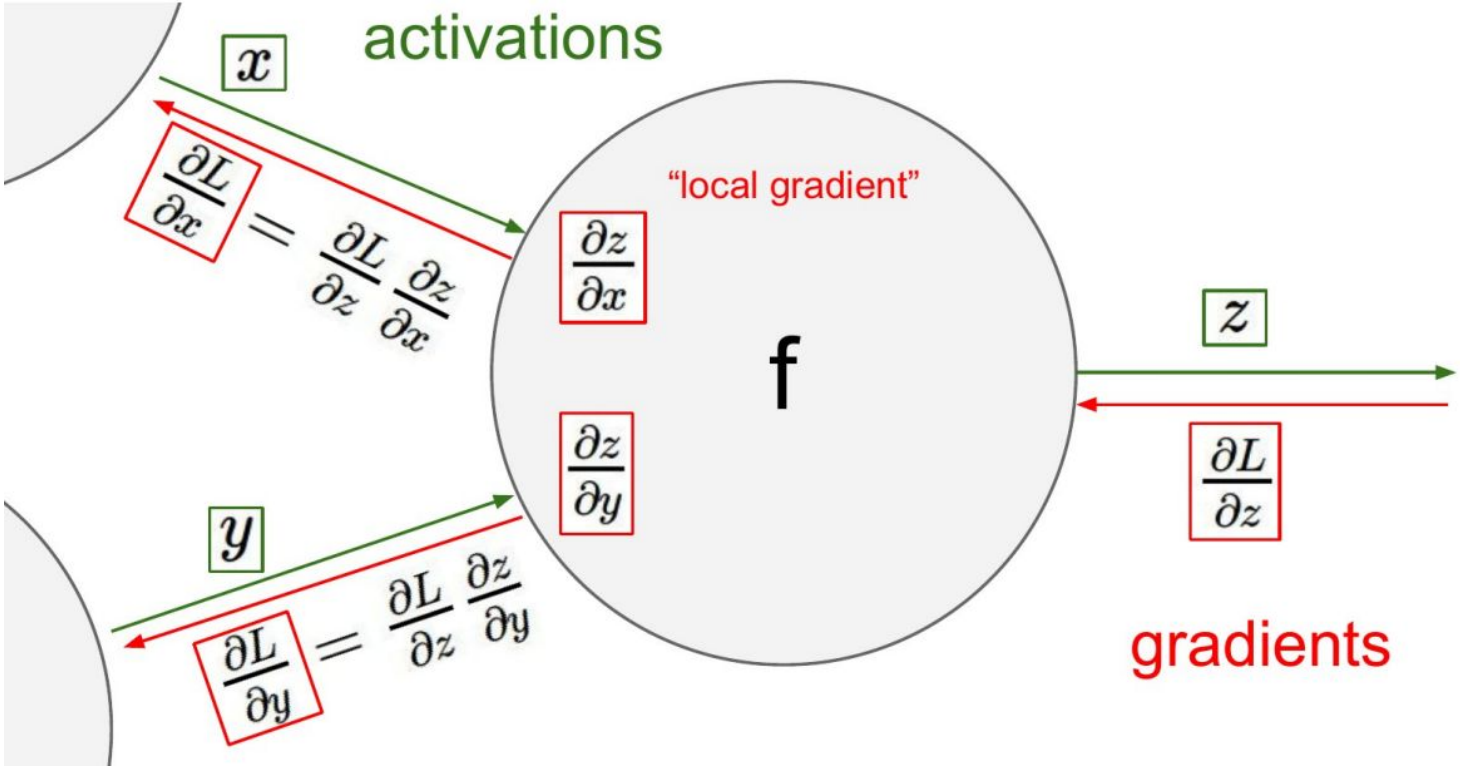

activations
x
y
f
z


activations
x
"local gradient"
$\frac{\partial z}{\partial x}$
f
z
$\frac{\partial z}{\partial y}$
y


activations
x
"local gradient"
$\frac{\partial z}{\partial x}$
f
z
$\frac{\partial z}{\partial y}$
$\frac{\partial L}{\partial z}$
y
gradients

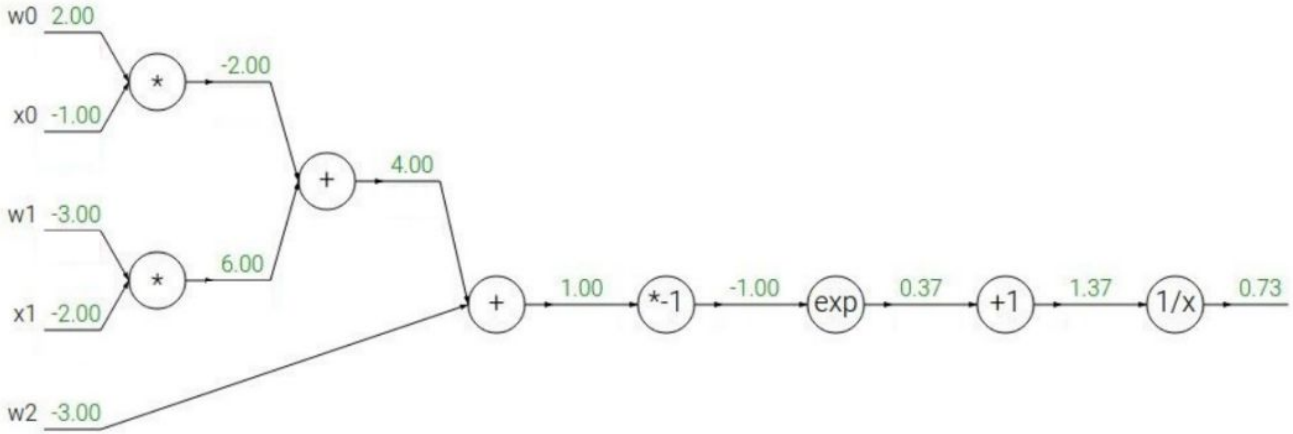

activations
x
$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial x}$
"local gradient"
$\frac{\partial z}{\partial x}$
f
z
$\frac{\partial z}{\partial y}$
$\frac{\partial L}{\partial z}$
y
gradients


activations
x
$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial x}$
"local gradient"
$\frac{\partial z}{\partial x}$
f
z
$\frac{\partial z}{\partial y}$
$\frac{\partial L}{\partial z}$
y
$\frac{\partial L}{\partial y} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial y}$
gradients


activations
x
$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial x}$
"local gradient"
$\frac{\partial z}{\partial x}$
f
z
$\frac{\partial z}{\partial y}$
$\frac{\partial L}{\partial z}$
y
$\frac{\partial L}{\partial y} = \frac{\partial L}{\partial z}\frac{\partial z}{\partial y}$
gradients

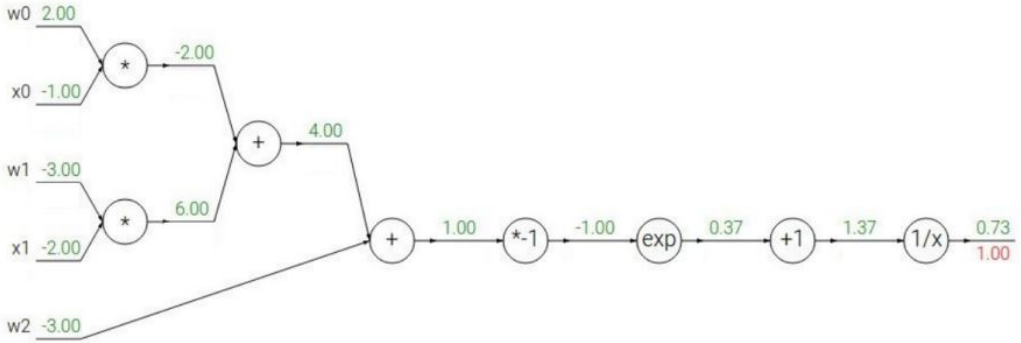
Becoming a back propagation ninja

$$f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$$



Becoming a back propagation ninja

$$f(w,x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}}$$



$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

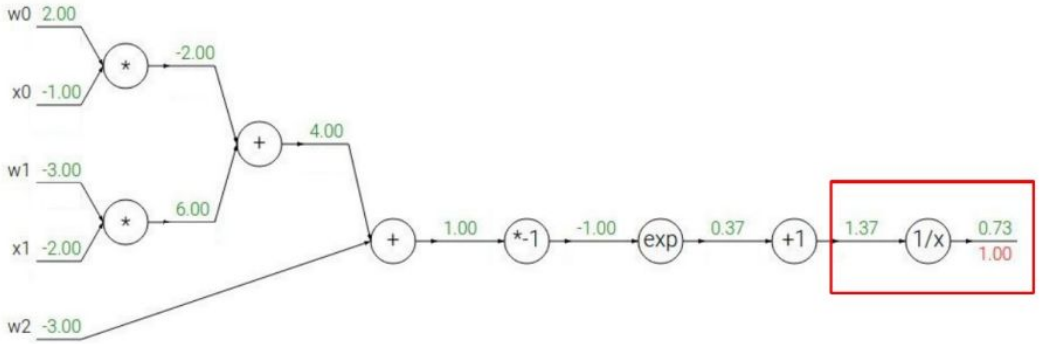
$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Becoming a back propagation ninja

$$f(w,x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}}$$



$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

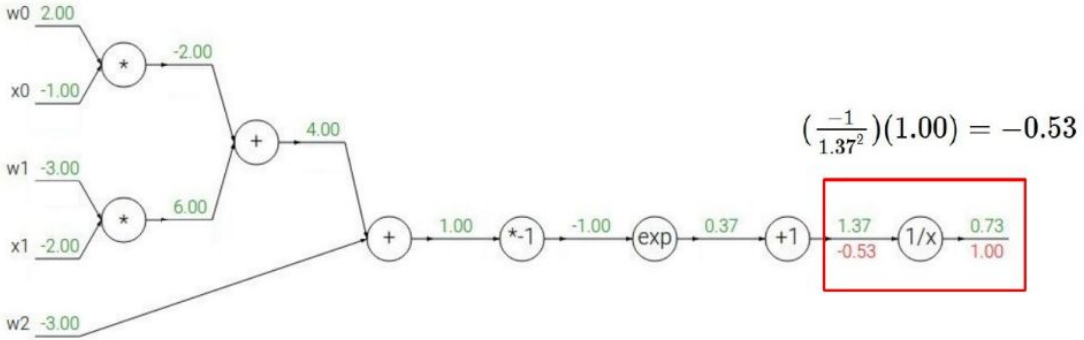
$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Becoming a back propagation ninja

$$f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$$



$$(\tfrac{-1}{1.37^2})(1.00) = -0.53$$

$f(x) = e^x \rightarrow \frac{df}{dx} = e^x$

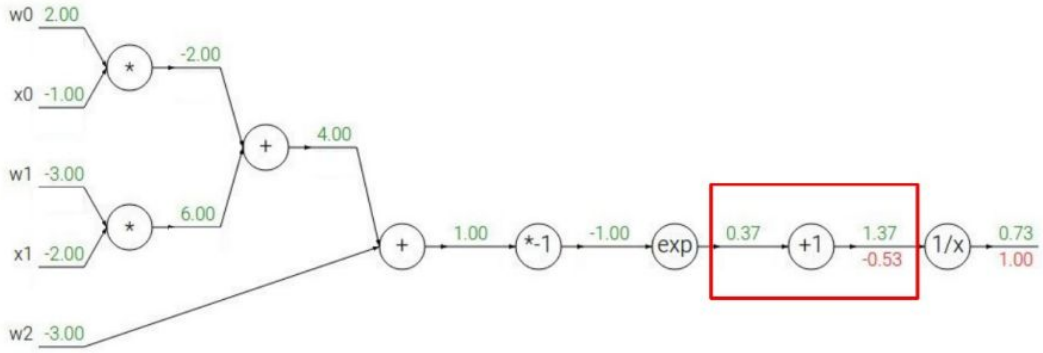
$f_a(x) = ax \rightarrow \frac{df}{dx} = a$

$f(x) = \frac{1}{x} \rightarrow \frac{df}{dx} = -1/x^2$

$f_c(x) = c + x \rightarrow \frac{df}{dx} = 1$

Becoming a back propagation ninja

$$f(w,x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}}$$



$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

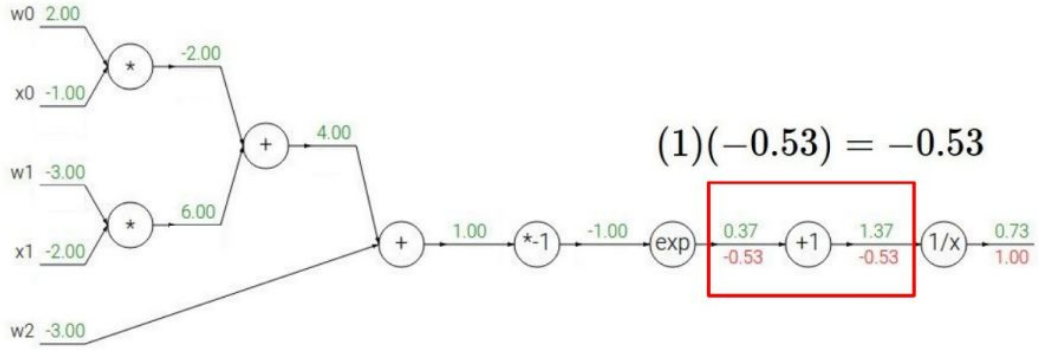
$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Becoming a back propagation ninja

$$f(w,x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}}$$



$$(1)(-0.53) = -0.53$$

$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

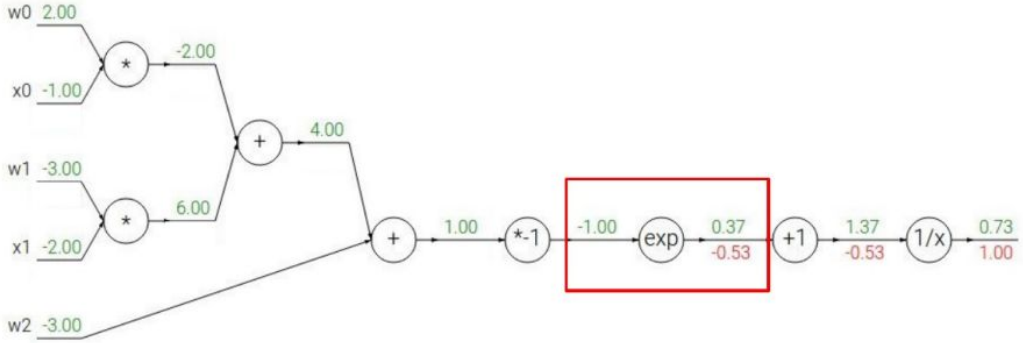
$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Becoming a back propagation ninja

$$f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$$



$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

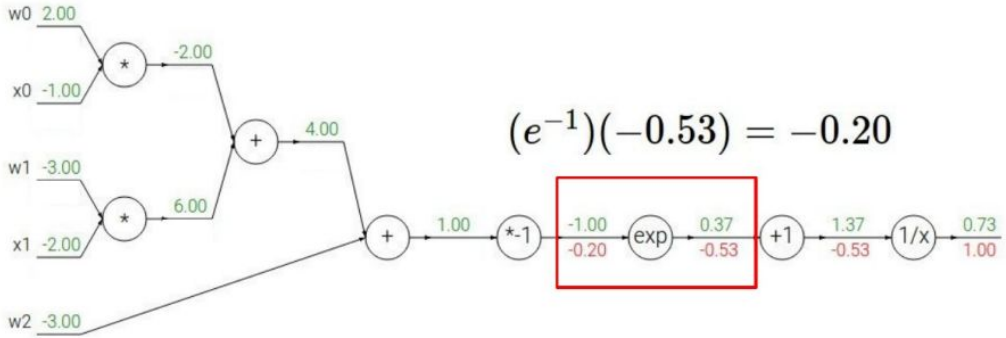
$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Becoming a back propagation ninja

$$f(w,x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}}$$



$$(e^{-1})(-0.53) = -0.20$$

$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

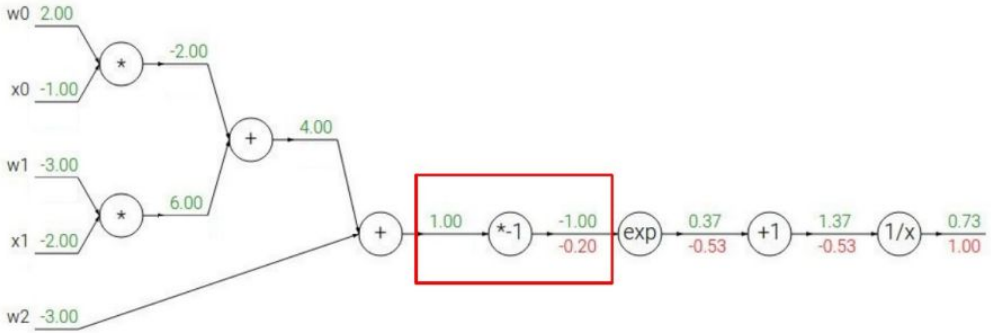
$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Becoming a back propagation ninja

$$f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$$



$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

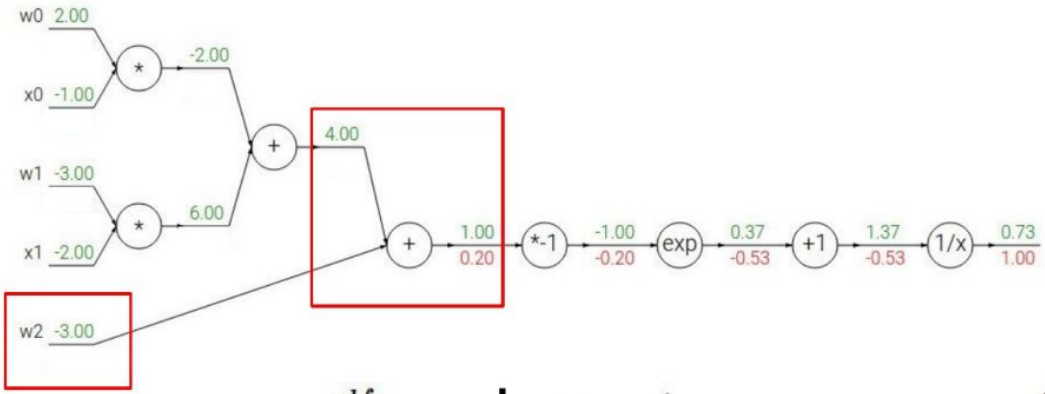
$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Becoming a back propagation ninja

$$f(w,x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}}$$



$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

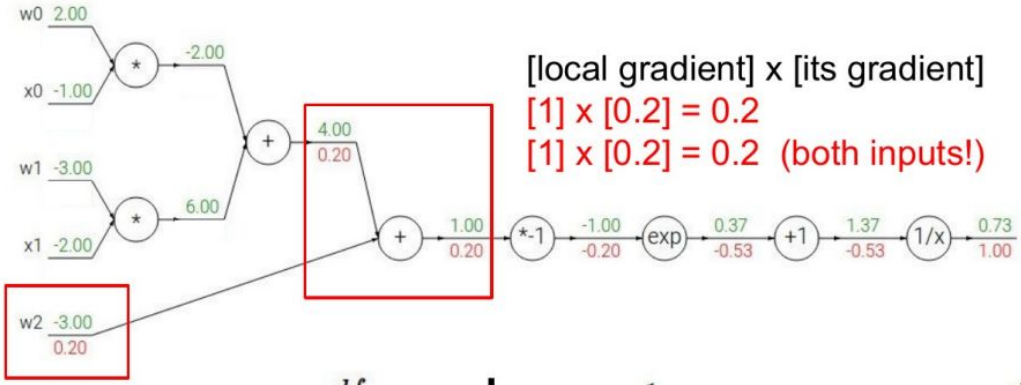
$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Becoming a back propagation ninja

$$f(w,x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}}$$



$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

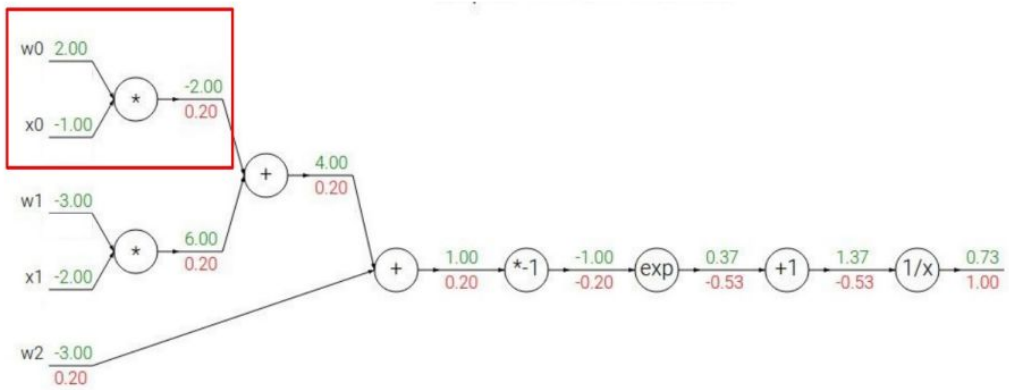
$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Becoming a back propagation ninja

$$f(w,x) = \frac{1}{1 + e^{-(w_0 x_0 + w_1 x_1 + w_2)}}$$



$$f(x) = e^x \quad \rightarrow \quad \frac{df}{dx} = e^x$$

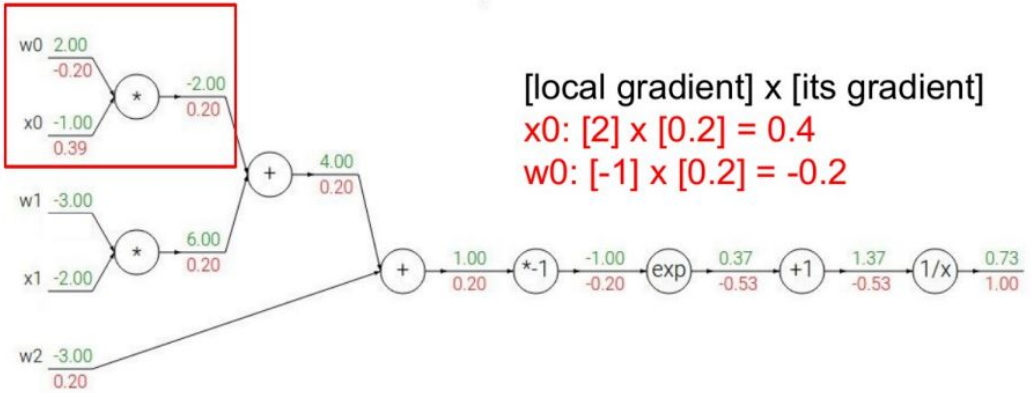
$$f_a(x) = ax \quad \rightarrow \quad \frac{df}{dx} = a$$

$$f(x) = \frac{1}{x} \quad \rightarrow \quad \frac{df}{dx} = -1/x^2$$

$$f_c(x) = c + x \quad \rightarrow \quad \frac{df}{dx} = 1$$

Becoming a back propagation ninja

$$f(w,x) = \frac{1}{1+e^{-(w_0x_0+w_1x_1+w_2)}}$$



[local gradient] x [its gradient]
x0: [2] x [0.2] = 0.4
w0: [-1] x [0.2] = -0.2

$$f(x) = e^x \rightarrow \frac{df}{dx} = e^x \qquad f(x) = \frac{1}{x} \rightarrow \frac{df}{dx} = -1/x$$

$$f_a(x) = ax \rightarrow \frac{df}{dx} = a \qquad f_c(x) = c + x \rightarrow \frac{df}{dx} = 1$$

Becoming a back propagation ninja

Architectures

First things first: the MLP

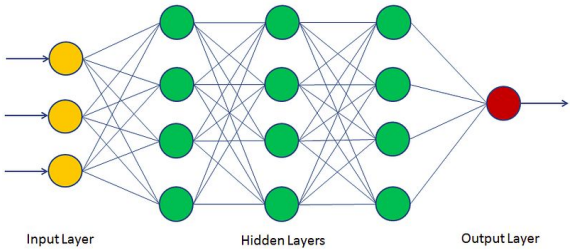
The Universal Approximation Theorem (UAT)

Cybenko 1989: Any multivariate function defined continuously on a compact set can be approximated by a **finite linear combination** of a fixed univariate **sigmoid function** and **affine functionals**.

$$f(x) = \sum_{i=1}^{N} \alpha_i \sigma(y_i^T x + b_i)$$

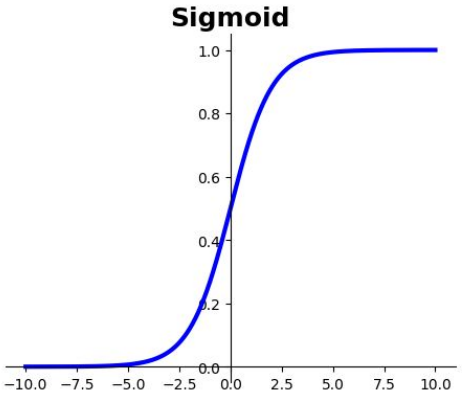
- Provides a way to tackle **high-dimensional**, **complex** and **non-linear** problems
- Catch: only an asymptotic convergence
- Many UAT variants exist

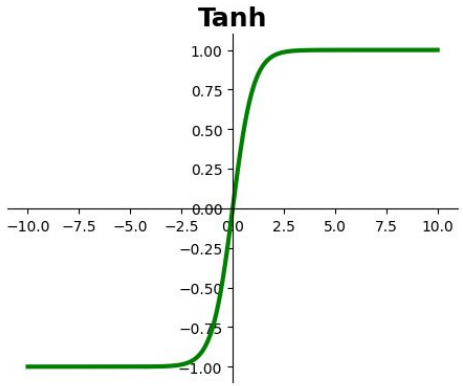
In practice: the Multi-Layer Perceptron (MLP)

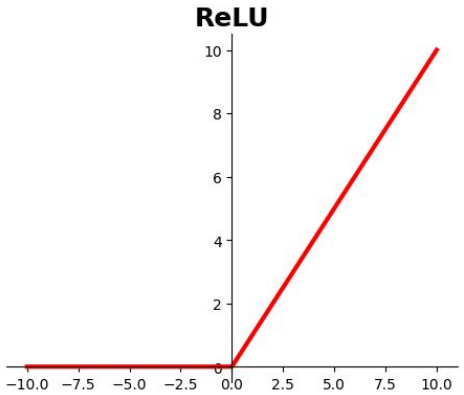


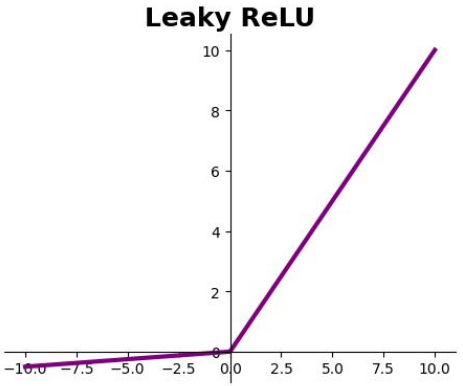
Activation functions

- Without them at each *hidden* layer, a MLP would be only a single linear operation: they allow both complexity and non-linearity.
- Notice the different output ranges

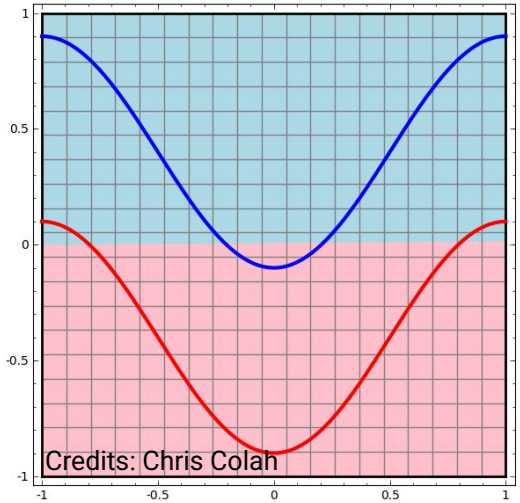


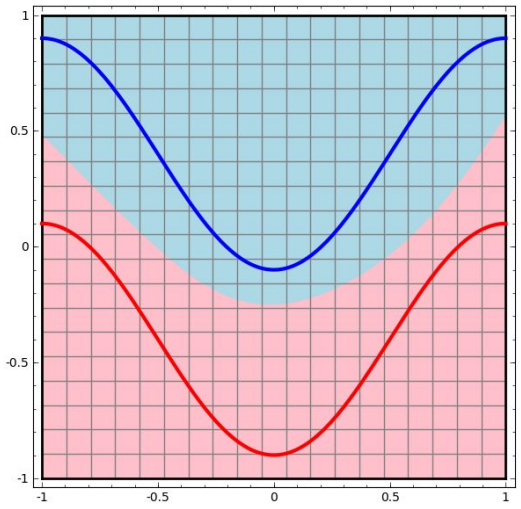


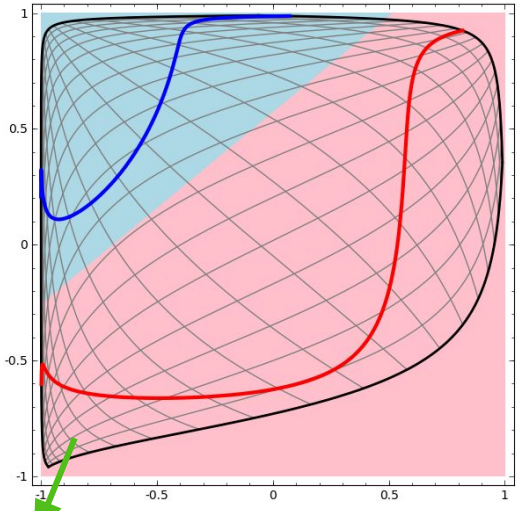


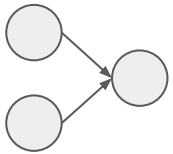


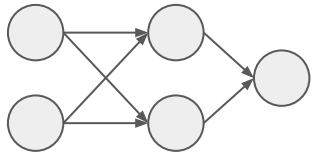
MLPs: Visualization and Topology





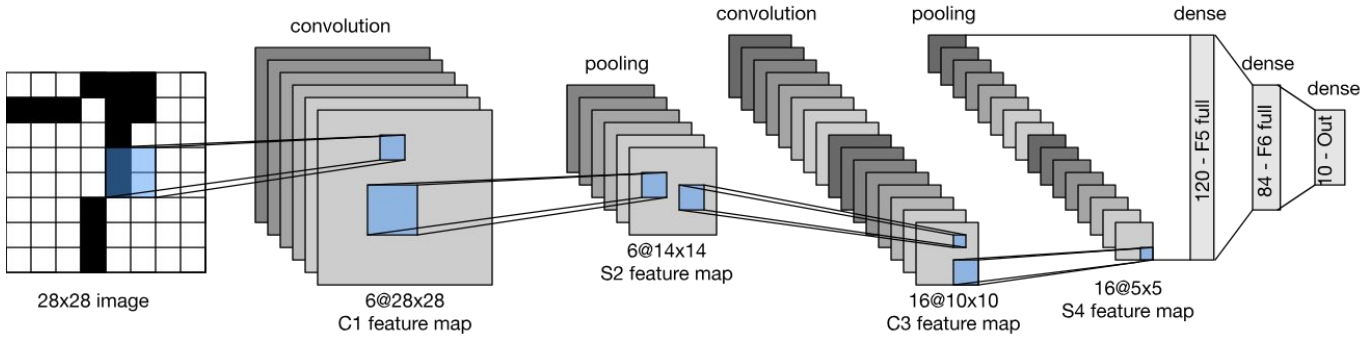






Also check ConvNetJS: https://cs.stanford.edu/people/karpathy/convnetjs//demo/classify2d.html

Convolutional Neural Networks



LeNet

How to efficiently process image data ?

Imagine passing this image through a MLP to detect the zebras:

- Image size:1000x400x3 pixels
- E.g. 256 neurons in the first layer
- → 30 million weights in the first layer only !

Intuitively, we can assume that:

- A zebra is not defined by its location in the image
- A zebra is defined by a composition of local features

→

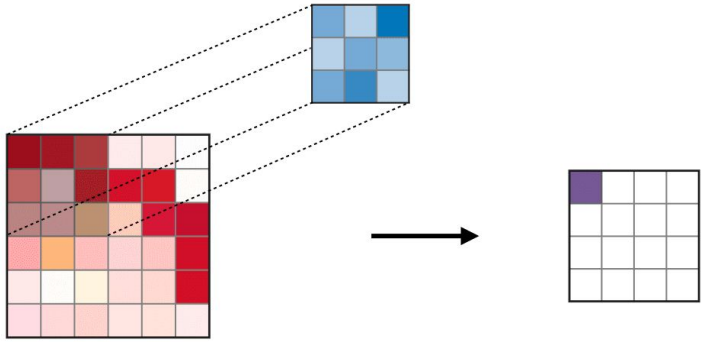
- Translational invariance
- Locality
- Feature hierarchy

Inductive biases

The Convolution operation

- Convolves an image by a filter, outputs an activation map

$$A_{j,k} = (F \star I)_{j,k} = \sum_{l,m} F_{l,m} I_{j-l,k-m}$$

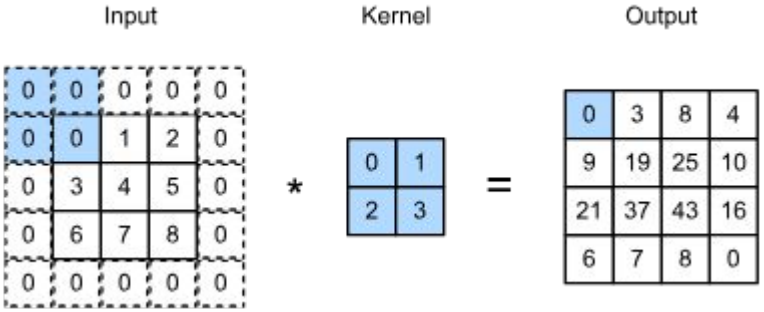


- Works in 1D, 2D, 3D…
- Several filters in a Conv layer

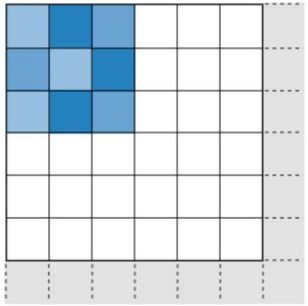
$$I : (.,.,C), F : (h,w,C) \times K$$
$$\Rightarrow A : (.,.,K)$$

- **Translational invariance** comes for free with the convolution
- **Locality**: filters can be of small size

Padding & striding



Padding: add extra numbers on the sides for better **perception of the edges** and control of the **output shape**

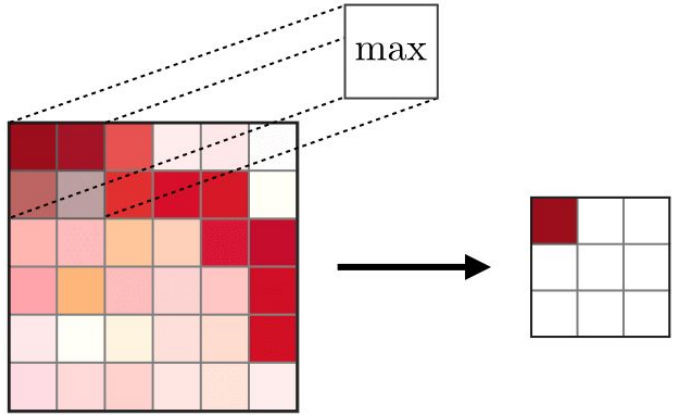


Striding: skip some operations and slide the filter by several pixels at a time for better **efficiency** and **downsampling**

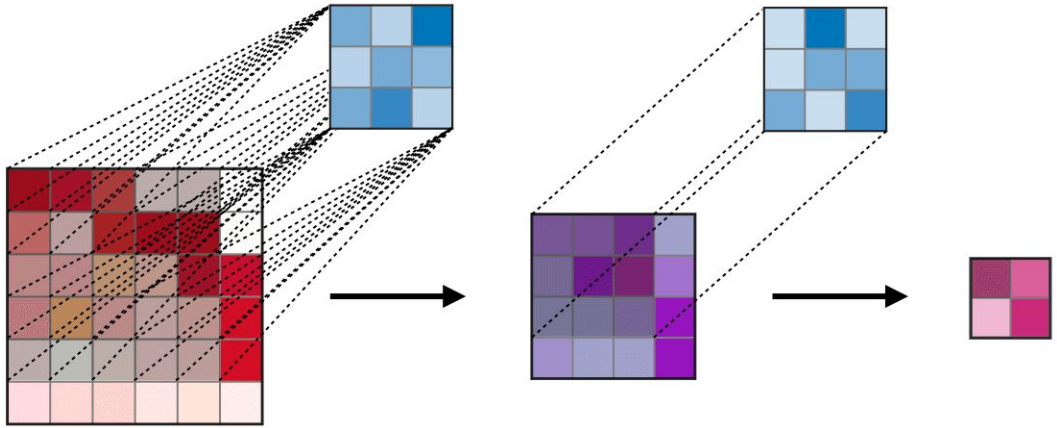
Pooling

Summarizes a group of pixel into one value (maximum or average)

- Increase the **receptive field** of each neuron
- Downsamples data for better efficiency
- Agreggates information: helps to learn a **global representation**



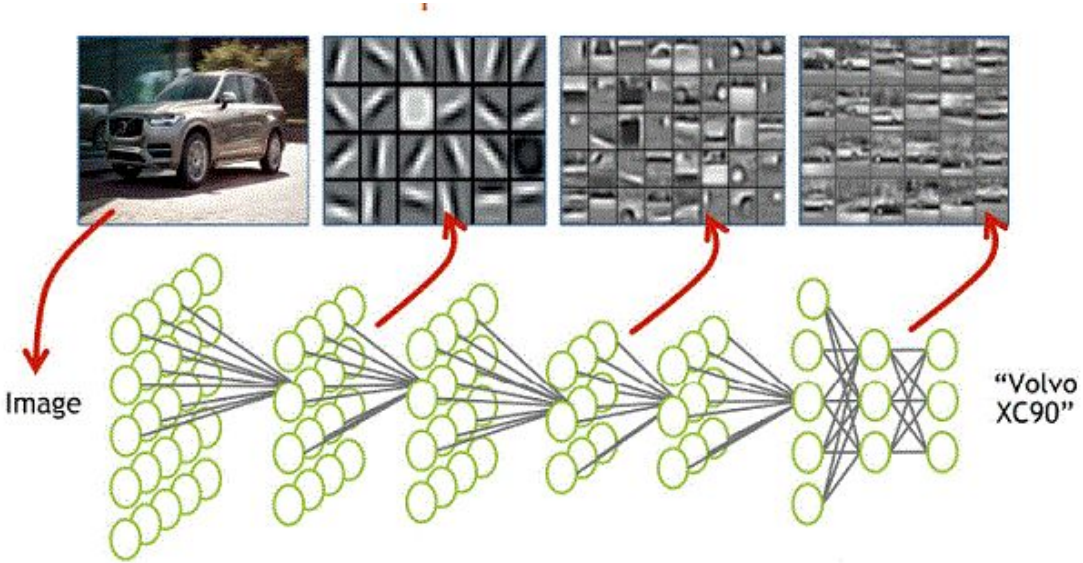
Receptive field



At a given layer, neurons see a larger portion of the image than in the previous layer.

High-level features are gradually extracted ! → **Hierarchy of features**

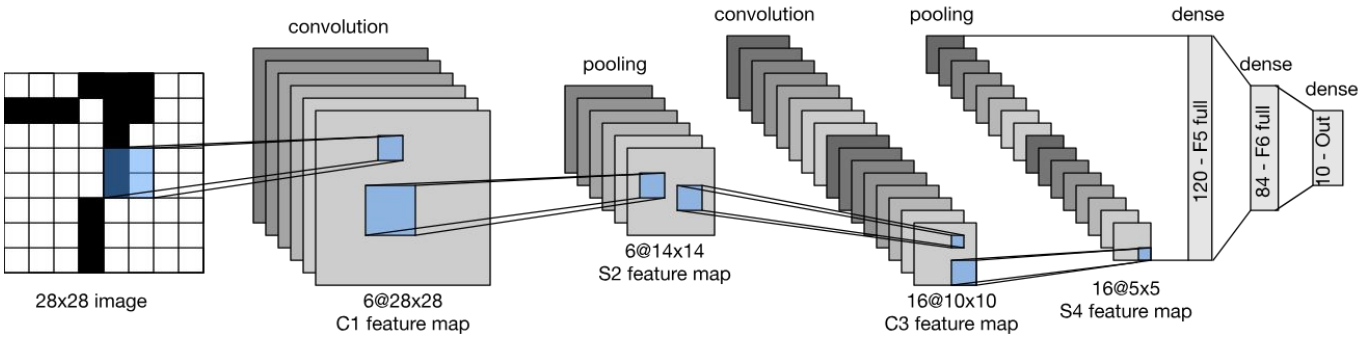
Hierarchy of features



You can even visualize what maximizes the filter activation at each layer

- Initial layers almost always capture very basic features, e.g. edges
- Next layers get more and more specialized towards the specific data and task

We just built a simple ‘prehistoric’ CNN !



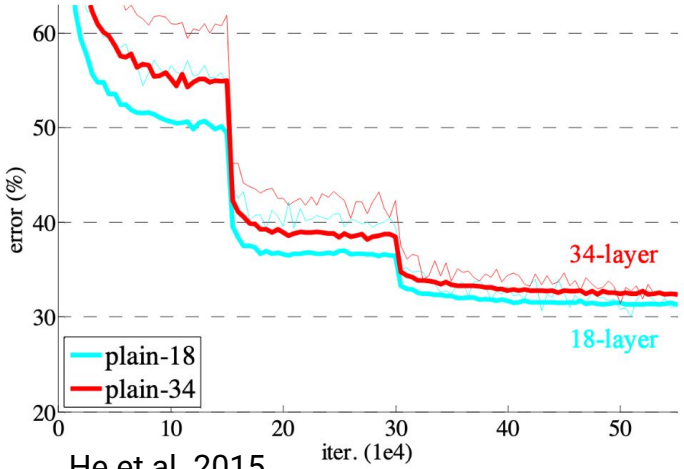
Le Cun et al. 1998

- Finally, the activation maps are flattened to be postprocessed by dense layers

We need to go deeper

Complex features are learnt through a sequence of layers…

→ How many layers can we have ?



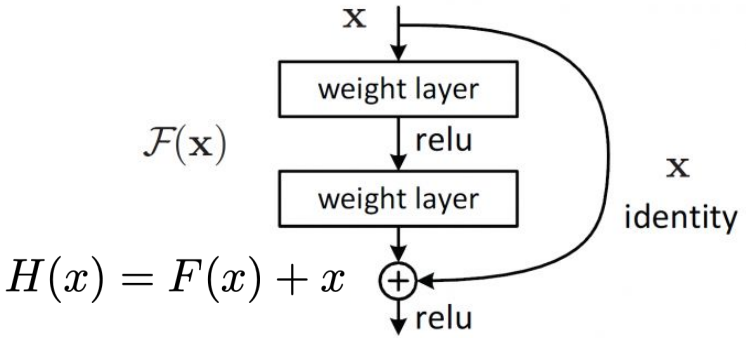
He et al. 2015

Problems with very deep models:

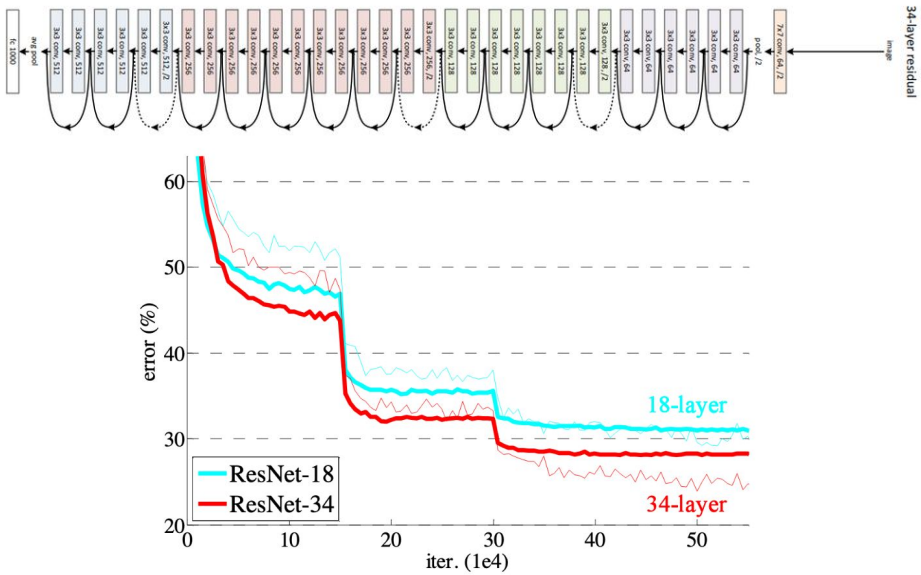
- Exploding/vanishing gradients
- Perform worse than shallow networks

Residual Networks (ResNets)

He et al. 2015 introduced the residuals blocks, featuring a **skip-connection**



- F(x) is now the residual H(x) - x : easier problem.
- Gradients flow through the identity branch



Allows very deep (100+ layers) networks with improved results !

U-Nets

Image-to-image tasks

- Segmentation: which pixel belongs to a zebra, to the grass, to the sky ?
- Inpainting: recover masked regions
- Super-resolution

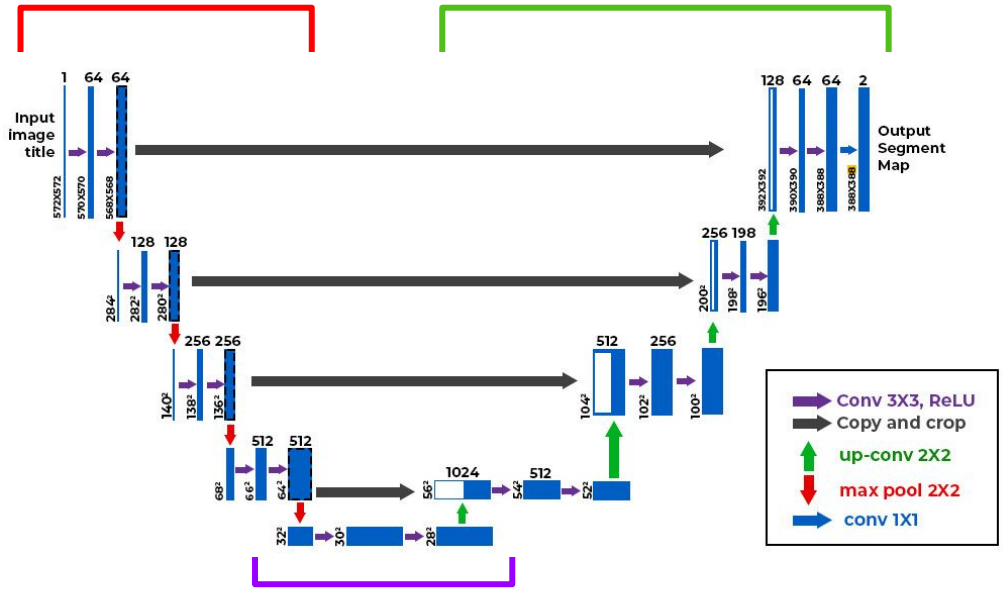
…

U-Nets

Ronneberger et al. 2015

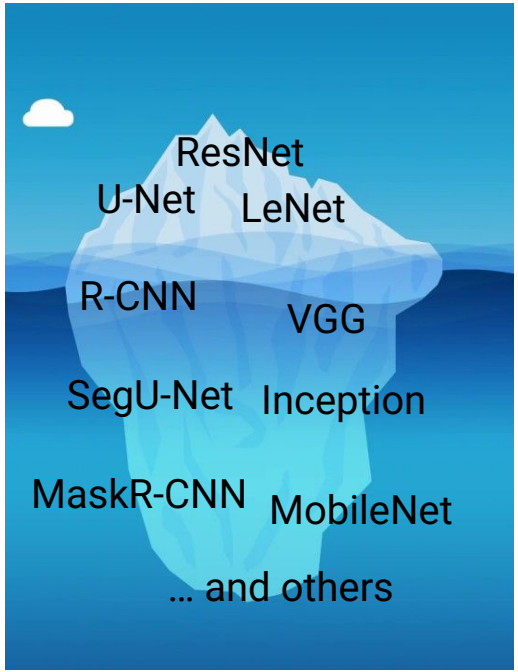
An architecture made to generate an image based on an other input image.

- **Encoder part**: extracts relevant features
- **Bottleneck**: processes high-level features
- **Decoder part**: generates sequentially an image with fine details
- **Skip-connections**: allow to pass small-scale features to the decoder

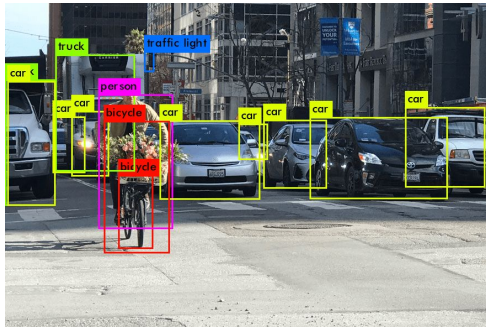


CNN-based architectures

This is only the tip of the iceberg…



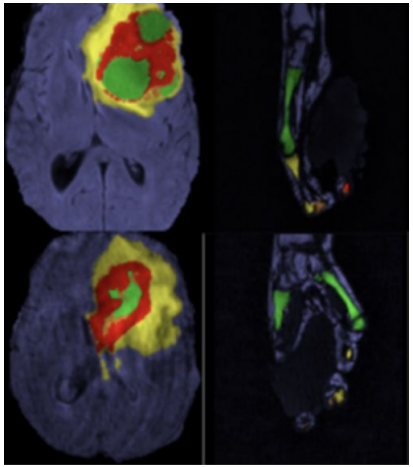
A few examples in various fields…



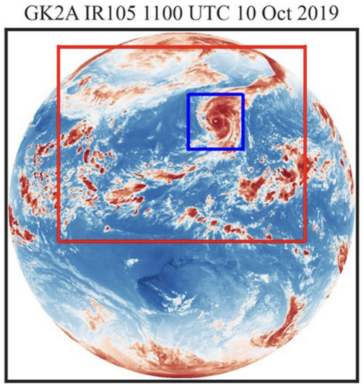
Reis et al. 2023

YOLOv8: Instant object detection, classification and segmentation

3D CNNs for medical MRI imaging segmentation



Kalayibay et al. 2017



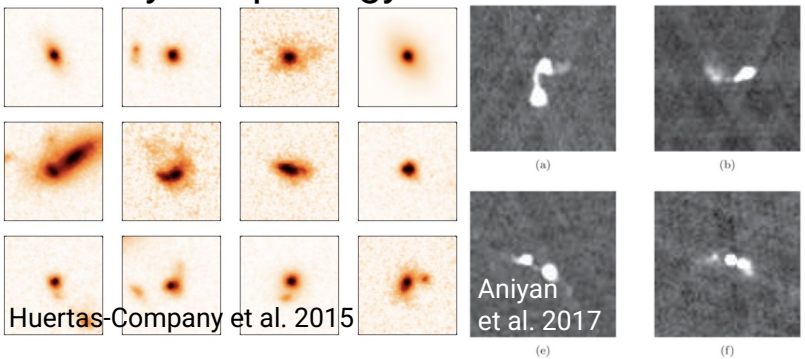
CNNs + Transfer Learning for intensity prediction of tropical cyclones from satellite data

Jung et al. 2024

And what about in Astrophysics ?

CNNs in Astrophysics

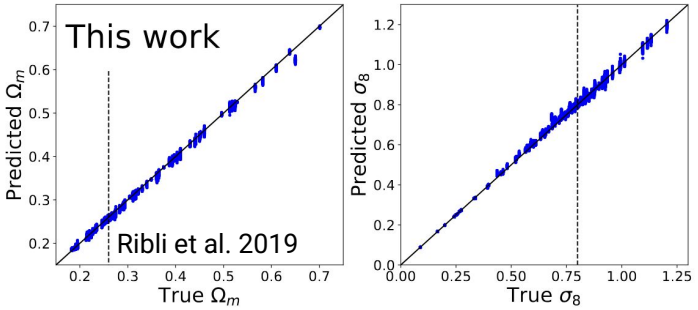
Galaxy Morphology classification



Huertas-Company et al. 2015

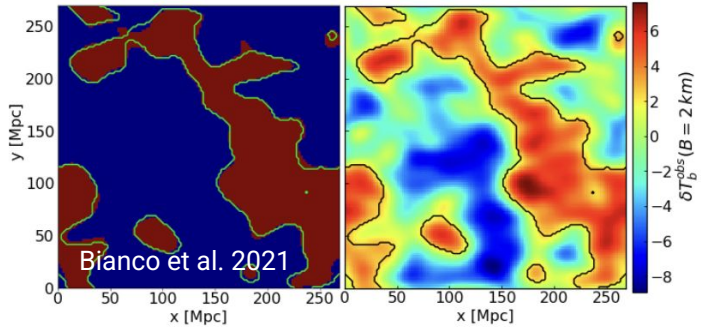
Aniyan et al. 2017

Cosmological parameters regression



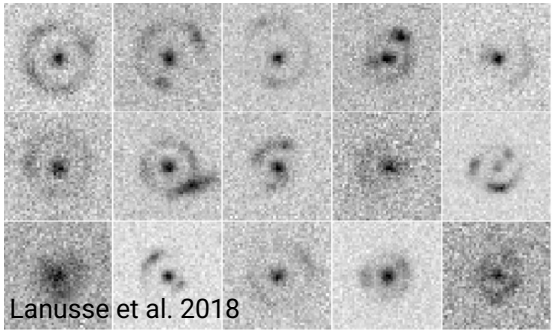
Ribli et al. 2019

Segmentation of 21cm observations



Bianco et al. 2021

Automatic strong lens finding

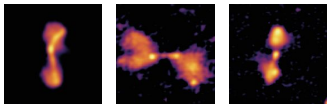


Lanusse et al. 2018

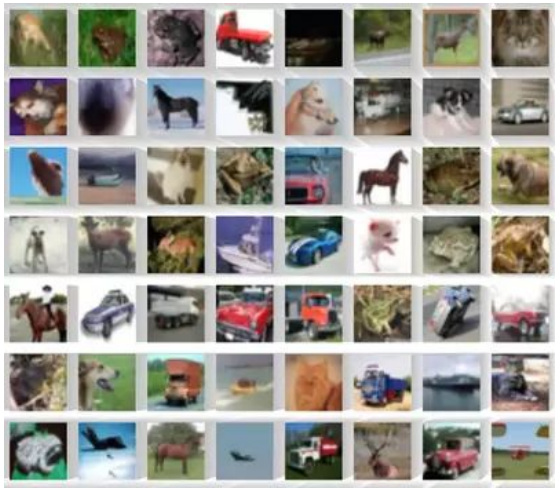
CNNs in Astrophysics

We face specific challenges due to our data:

- Limited **dataset size** and label availability
- **Domain shift** between traditional DL datasets and astronomical images
- Huge **dynamic range**
- High level of **noise**
- **Instrumental artefacts** vary for different telescopes (reusability)



VS

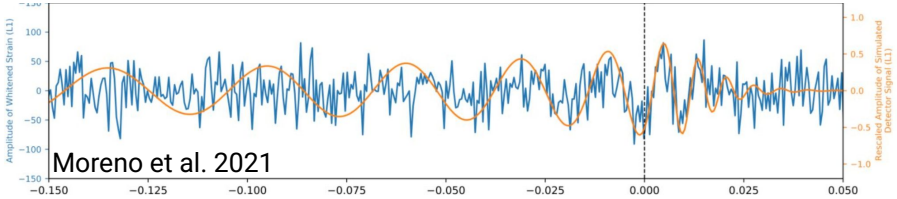


CIFAR10 Dataset

Recurrent Neural Networks

What about data that is sequential in nature ?

- Time series



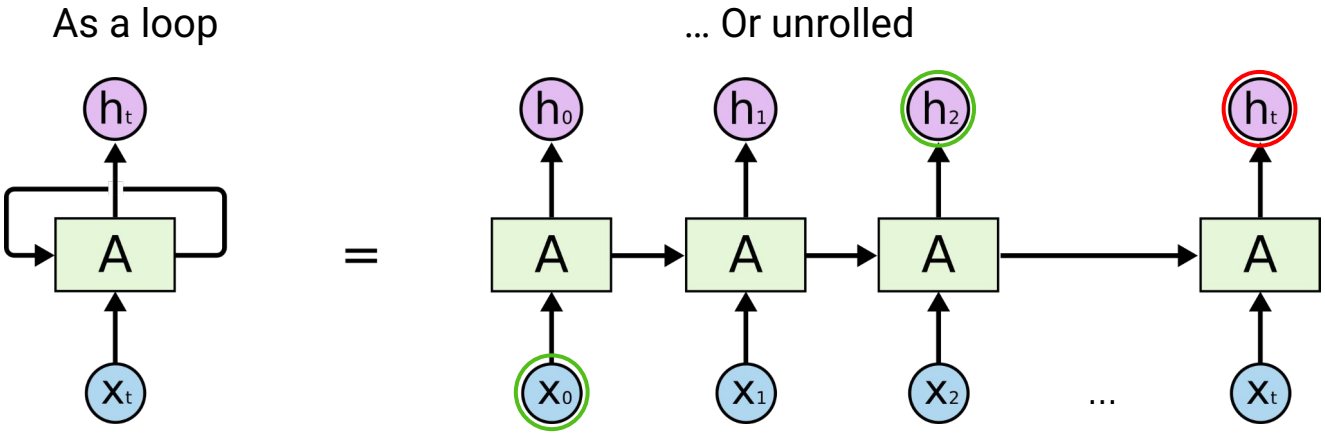
- Text

```
The SKAO confirms aliens from Fornax cluster say hi.
```

- Persistence/Memory
- (Time-)Ordering
- Long and Short-term dependencies
- Flexible sequence length

Recurrent Neural Networks

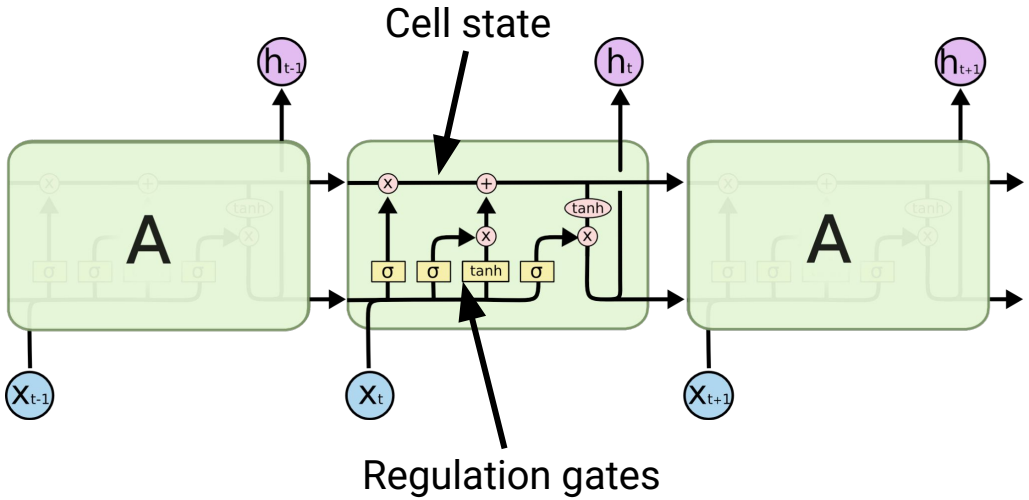
Basic idea: let a “hidden state” to be transmitted from iteration to iteration.



- A can be any kind of neural network (e.g. with dense and/or convolutional layers…)
- In practice, fails to catch long-term dependencies

Long-Short Term Memory (LSTMs)

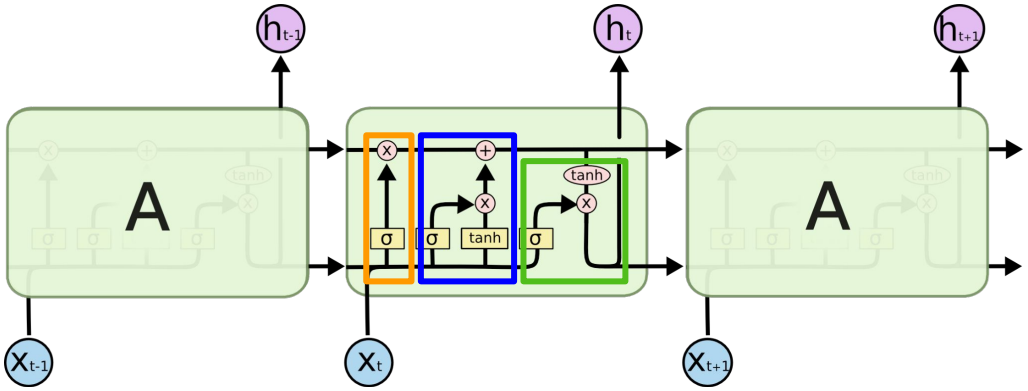
- Designed to address the long-term memory issues from standard RNNs
- The access to the ‘memory’ is carefully regulated by gates



Long-Short Term Memory (LSTMs)

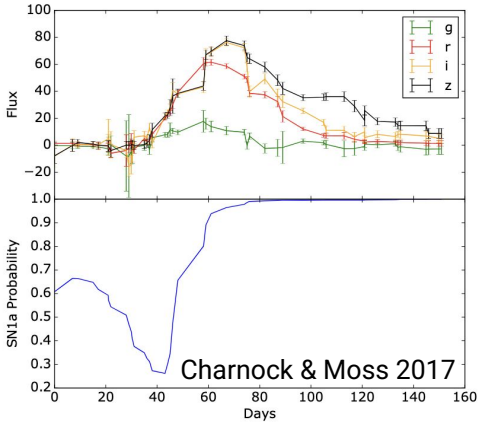
Hochreiter & Schmidhuber 1997

- '**Forget gate**' : deletes information in the cell state
- '**Update gate**' : writes information from the new input to the state cell
- '**Output gate**': combines the cell state and input to generate the cell output



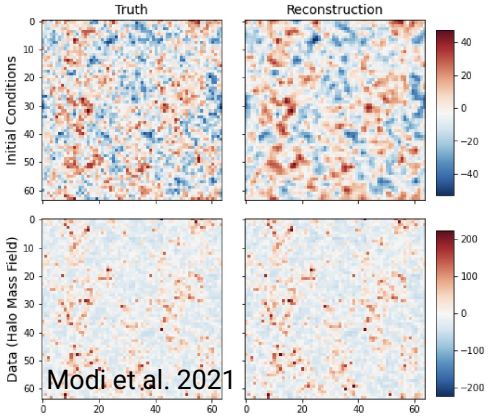
Many other variants exist..

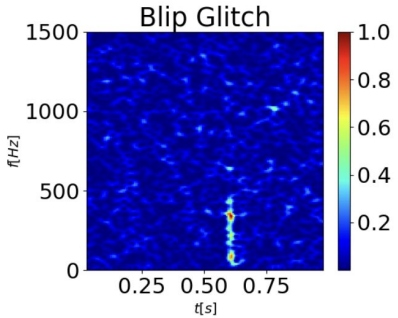
RNNs in Astrophysics

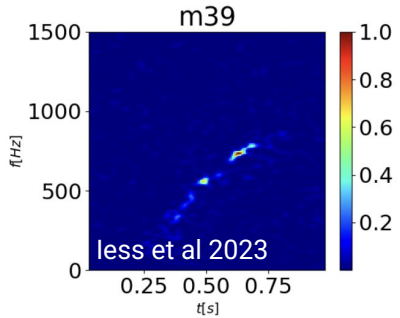


SN classification with LSTMs

Initial conditions reconstruction with U-net LSTM







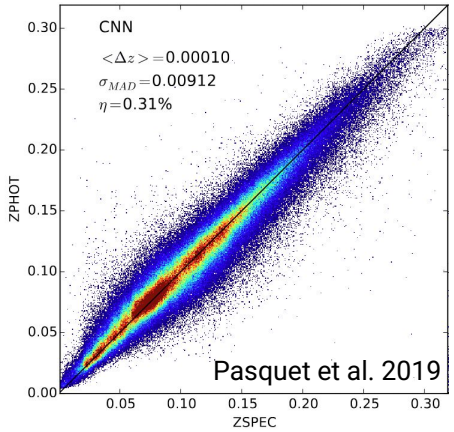
LSTM for GW/noise transients classification and alarm triggering

Tasks and learning objectives

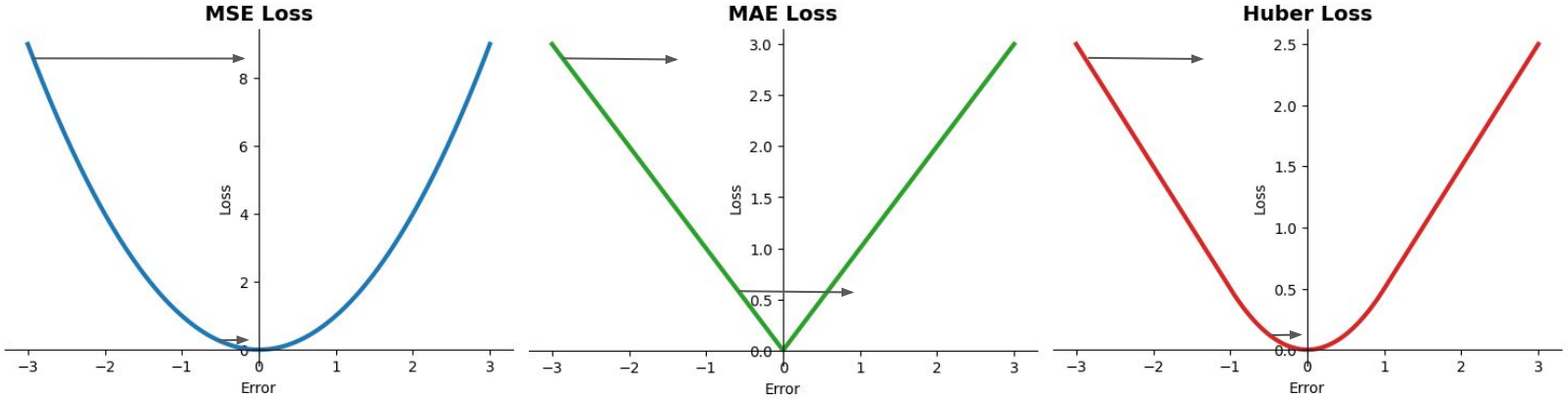
Regression

Infer a value from complex data:

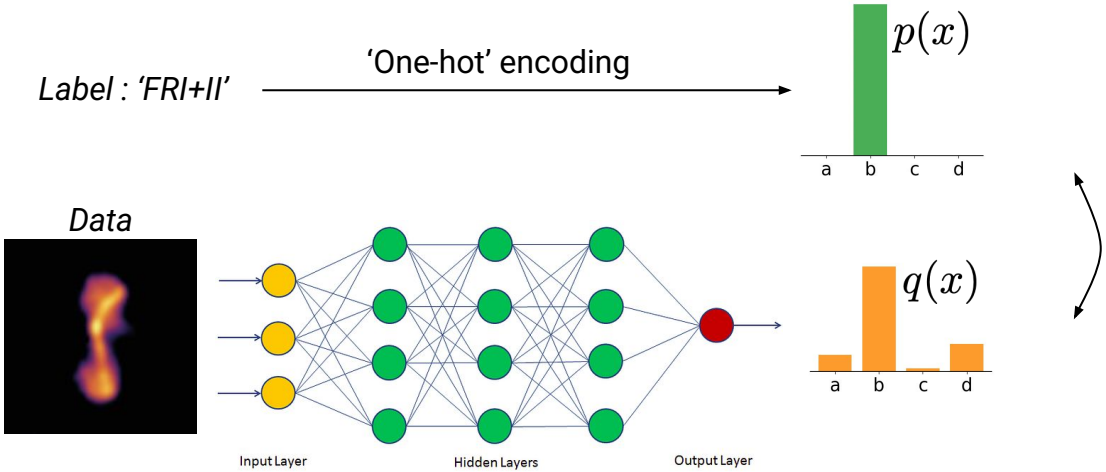
- a redshift from photometric data
- a cosmological parameter from a galaxy survey



Usual losses exploit the error



Classification



Information theory provides us with the **Cross-Entropy:**

$$H(p,q) = \mathbb{E}_{x\sim p(x)}(-\log_2 q(x))$$
$$= -\sum_x p(x)\log_2 q(x)$$

Minimal for: $p(x) = q(x)$

Intuition of CE: How many bits (e.g. information) in average do I need to encode a element from p if I (mistakenly) think it comes from q ?

Generative Modeling

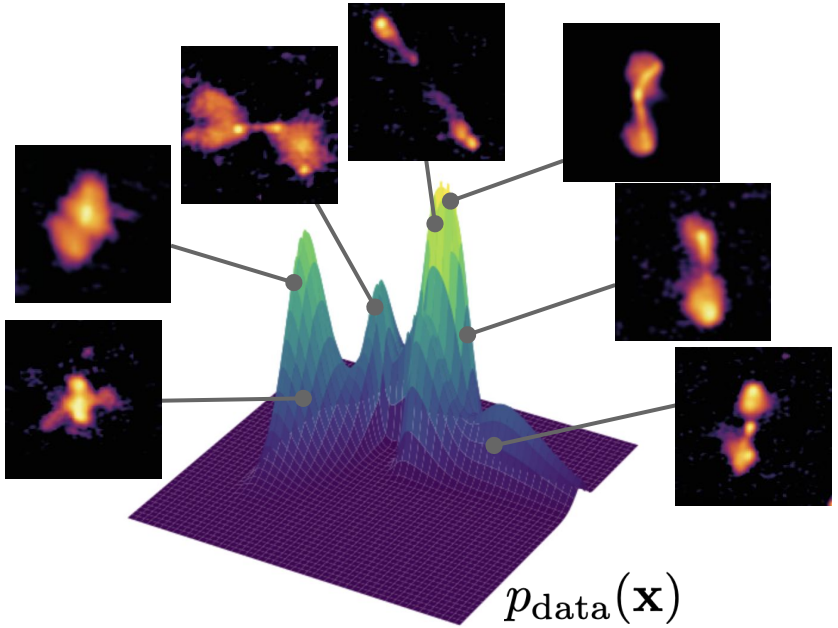
Until now we saw applications to **analyse** existing data (classification / regression)

… But can we generate **new** data ?

A dataset of images (e.g. radio AGNs) stems from a probability distribution on the space of all possible images.

→ Need to learn and sample this probability

$$p_{\text{data}}(\mathbf{x}) \approx p_\theta(\mathbf{x})$$



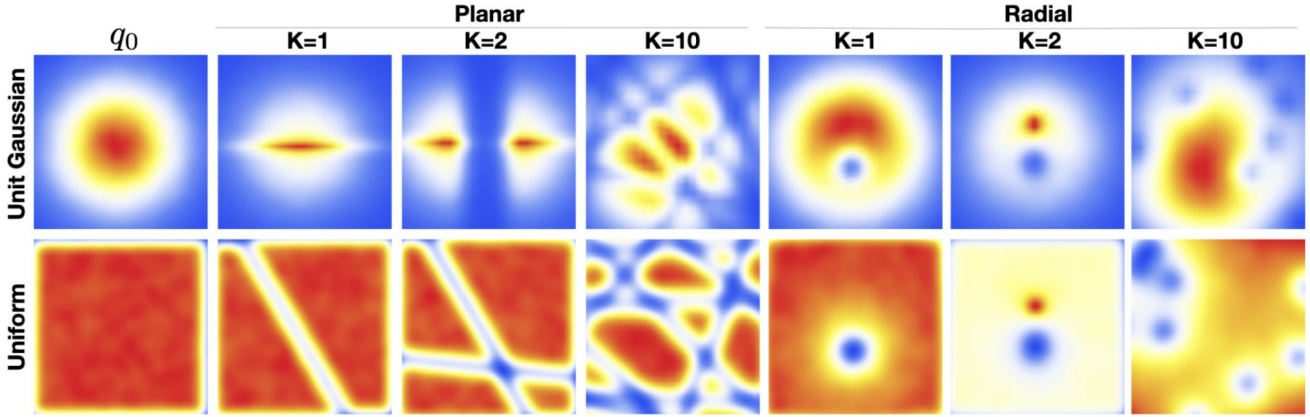
Normalizing Flows

→ Learn a change of variables to map a simple distribution into the target.

$$\mathbf{x} = g_\theta(\mathbf{z})$$

Uses a composition of **non-linear** and **bijective** transformations.

$$g_\theta = g_\theta^1 \circ \ldots \circ g_\theta^K$$



Rezende and Mohamed 2016

Optimization: maximize the **likelihood** of the dataset under the network's output probability.

Generative Adversarial Networks (GANs)

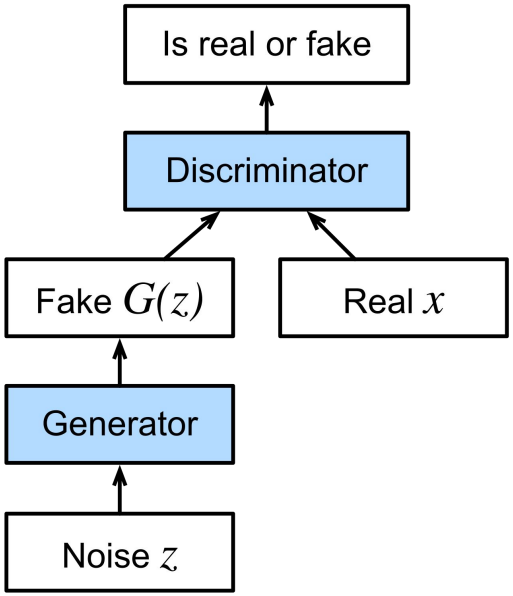
Two networks competing against each other:

- **Generator**: creates fake data from noise
- **Discriminator:** classifies if the input image is fake or real.

It's a classification task → Binary Cross Entropy loss

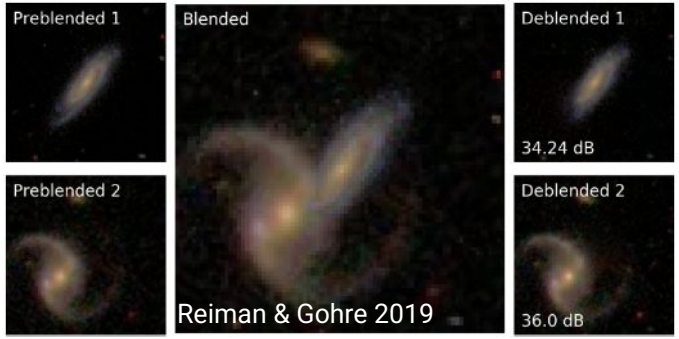
- Minimized by the Discriminator
- Maximized by the Generator

$$\min_D \max_G V(D,G)$$
$$= \mathbb{E}_{x\sim p_{\text{ref}}(x)}[\log D(x)] + \mathbb{E}_{x\sim p_{\text{G}}(x)}[\log(1-D(x))]$$

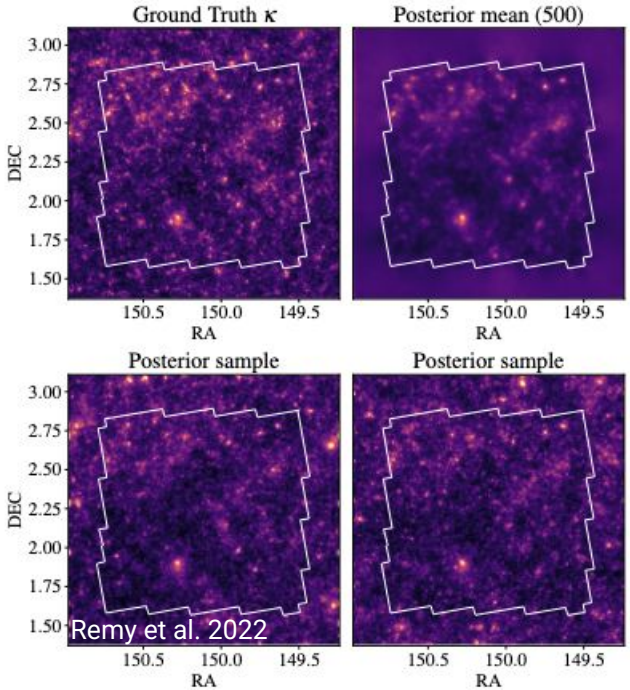


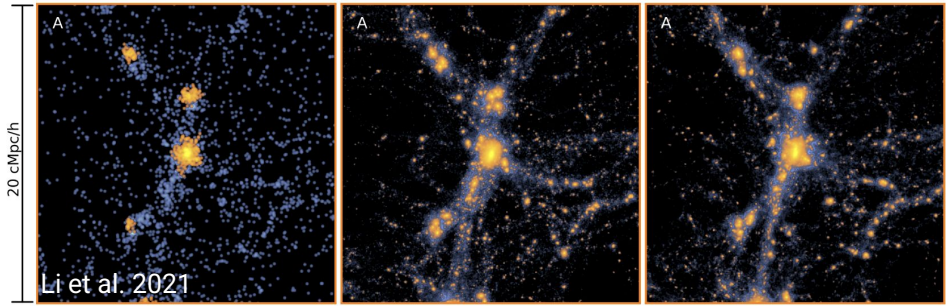
Generative modeling in Astrophysics

Galaxy deblending with GANs



Reconstructing mass maps with Diffusion models (see DL tutorial 2 for DDPMs)





Super-resolution of cosmological simulations with GANs

DL Basics - takeaways

Different data requires different architectures and their underlying inductive biases:

- CNNs are translation-invariant and can learn complex hierarchical features in images.
- RNNs learn long and short term dependencies from flexible-length time series or text.

Are neural networks **black boxes** ?

- Yes: depth and number of params make it impossible to understand exactly the outcome of one particular net.
- No: it relies on *solid theoretical foundations* (statistics, optimization, information theory…).
  + There are interpretability tools (see DL tutorial 1).

On the DL workflow

- There are already many architectures out there, do not **reinvent** one !
- Any architecture and optimization is full of implementation details… Experiment to see their impact on **your** task.

References

- Stanford's Convolutional Neural Networks cheatsheet, https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks
- The DAWES review 10: The impact of deep learning for the analysis of galaxy surveys, Huertas-Company and Lanusse, PASA 2022, https://arxiv.org/abs/2210.01813
- Chris Colah's blog: https://colah.github.io/
- MVA Course on Generative Modelling, Bruno Galerne and Arthur Leclaire, https://generativemodelingmva.github.io/