Deep learning: algorithms and deployment iSense Big Data Summit

Rob Romijnders

robromijnders.github.io

RomijndersRob@gmail.com

October 24, 2017

Overview

October 24, 2017

Rob Romijnders

Why

How

What

Deploy

Wrap up

Questions

1 Why

2 How

3 What

4 Deploy

5 Wrap up

6 Questions

Object detection

Faster R-CNN

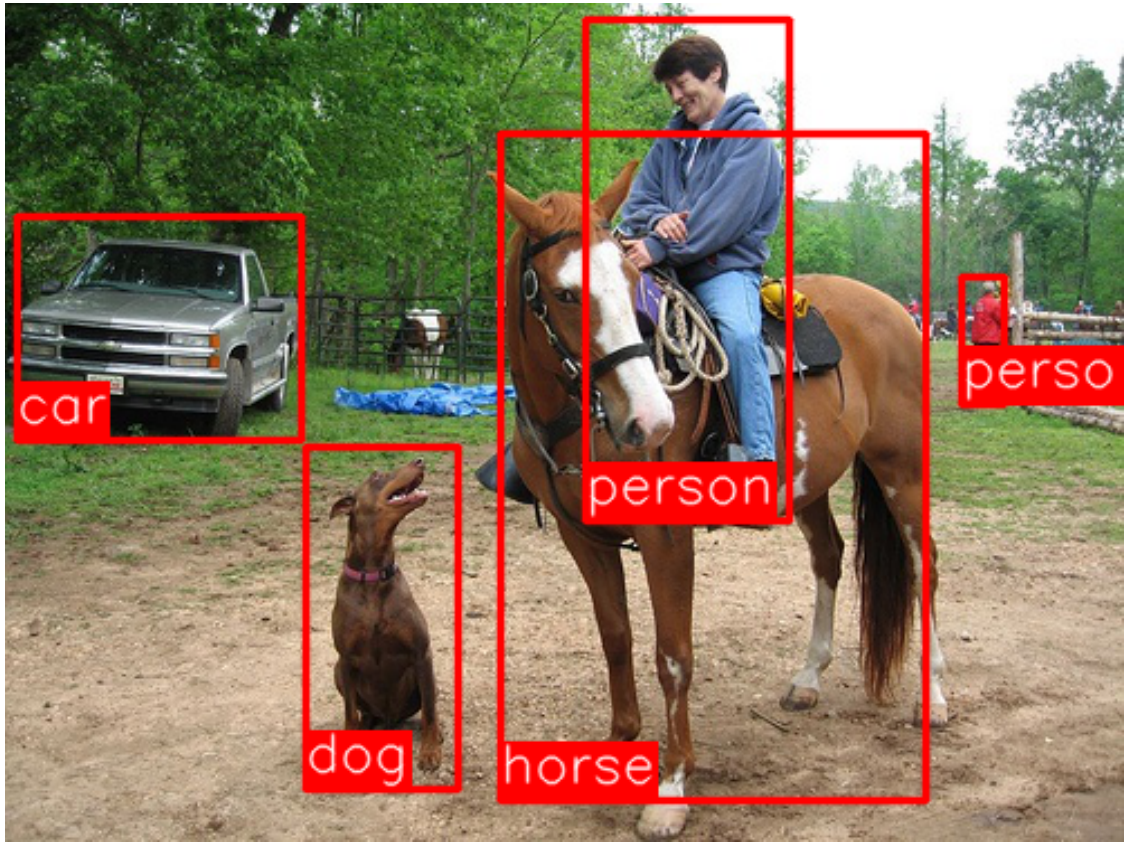


Figure: faster rcnn (image from Github *mitmul*)

Apple siri

Figure: Photo: cultofmac.com

GMail reply

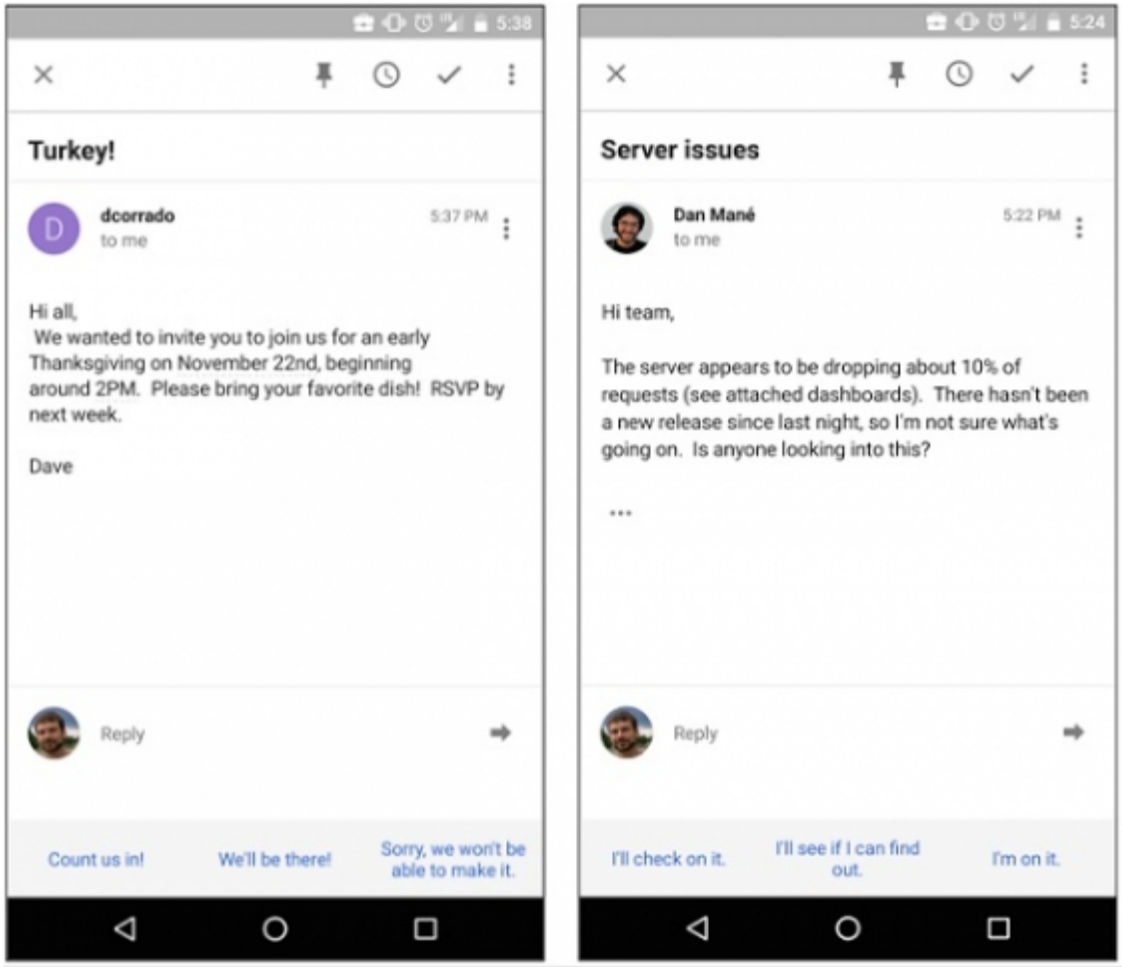


Figure: Photo: Greg Corrado, Google Research Blog

Figure: credits: Google research blog

Segmentation

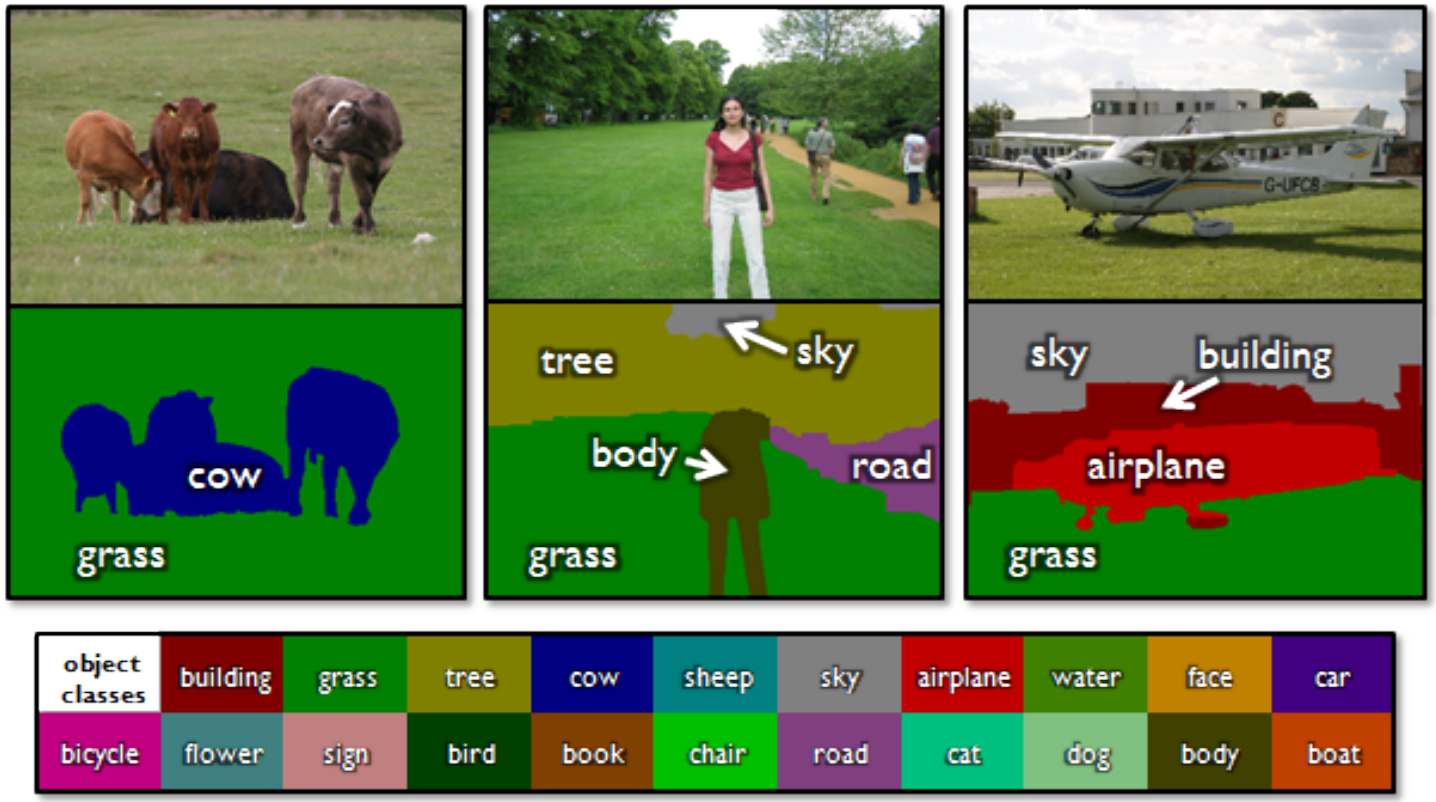


Figure: Semantic Segmentation with CNN (image from jamie.shotton.org) or see youtube.com/watch?v=Nok6Xludc˙Q

Pose estimation

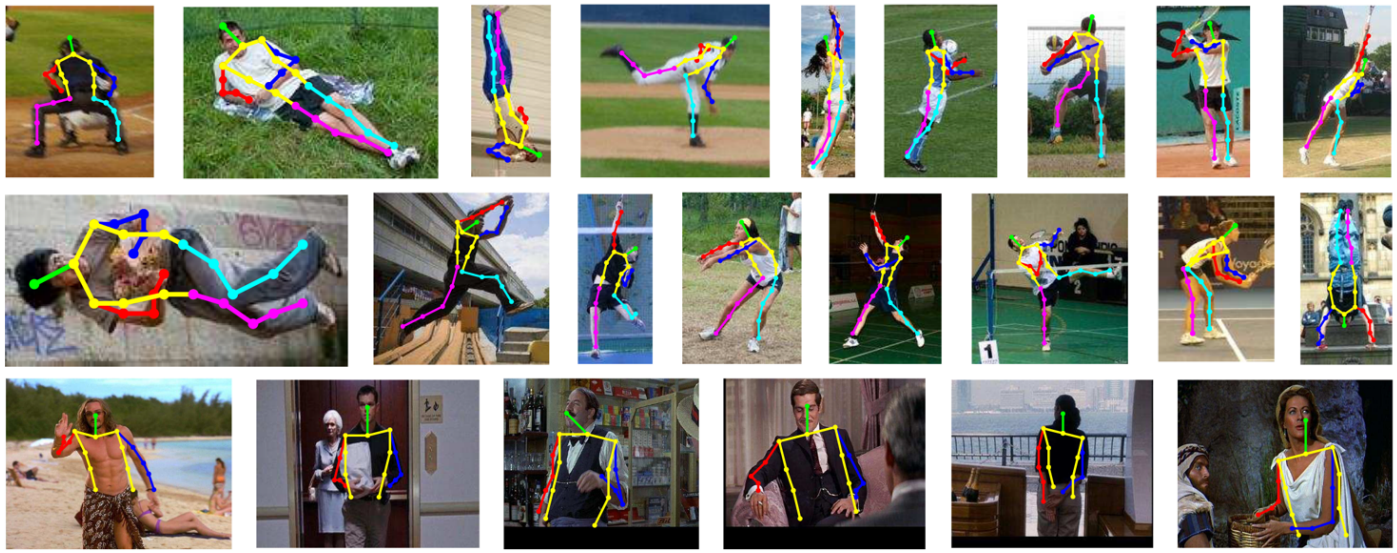


Figure: Source

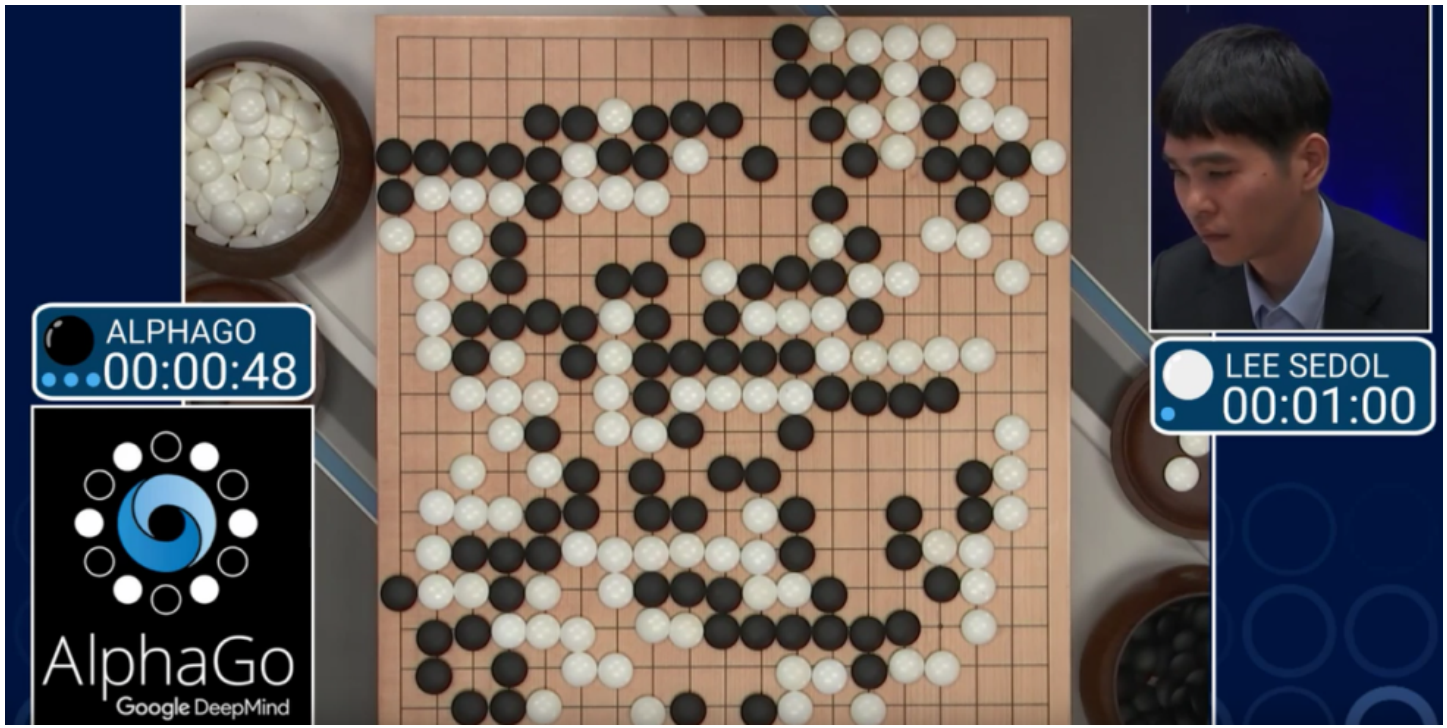


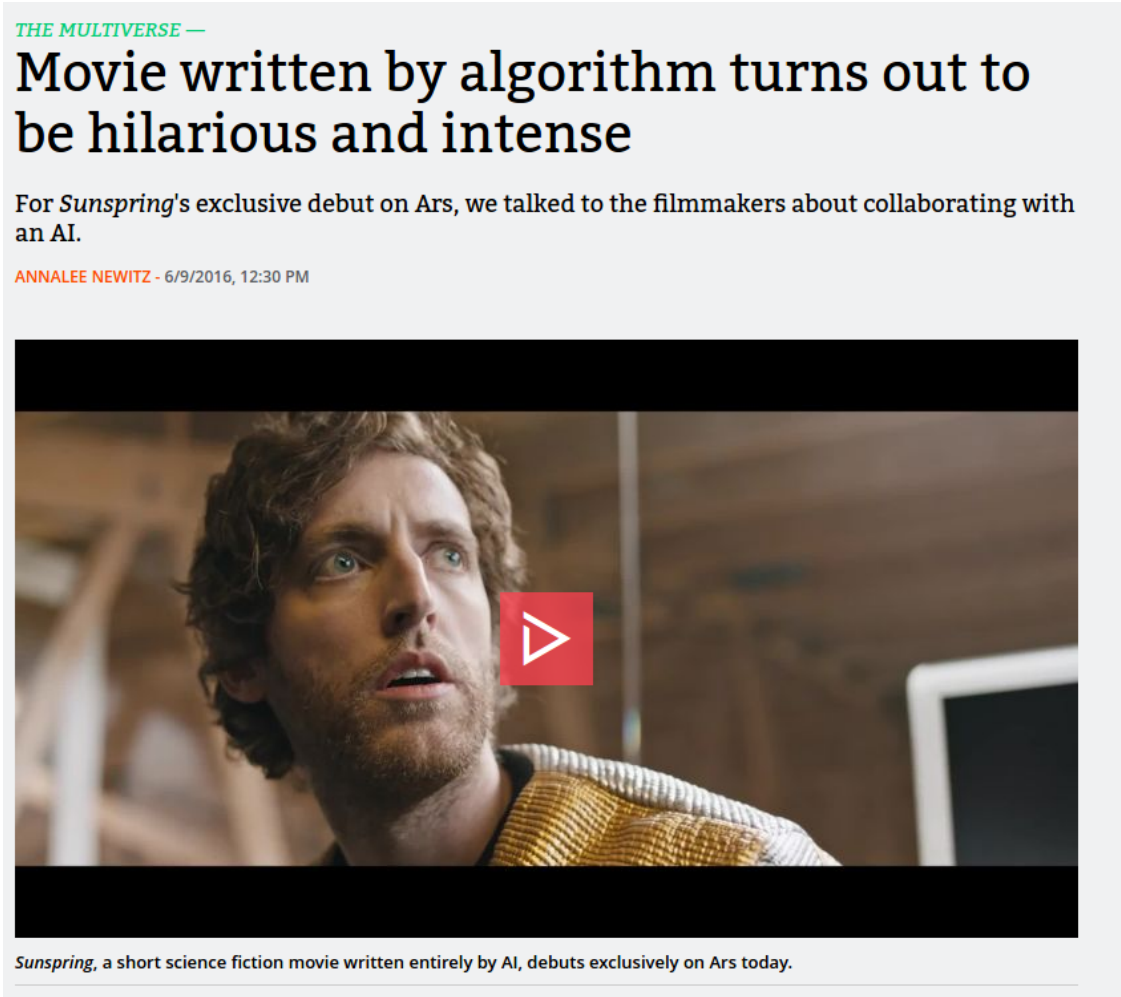
Figure: Credits: popularmechanics.com/technology/a19863/googles-alphago-ai-wins-second-game-go/

Generative modelling

Speech generation

Figure: Source: deepmind.com/blog/wavenet-launches-google-assistant/

Generation



THE MULTIVERSE —

Movie written by algorithm turns out to be hilarious and intense

For *Sunspring*'s exclusive debut on Ars, we talked to the filmmakers about collaborating with an AI.

ANNALEE NEWITZ - 6/9/2016, 12:30 PM

Sunspring, a short science fiction movie written entirely by AI, debuts exclusively on Ars today.

Figure: http://arstechnica.com/the-multiverse/2016/06/an-ai-wrote-this-movie-and-its-strangely-moving/

Why not

Which data not

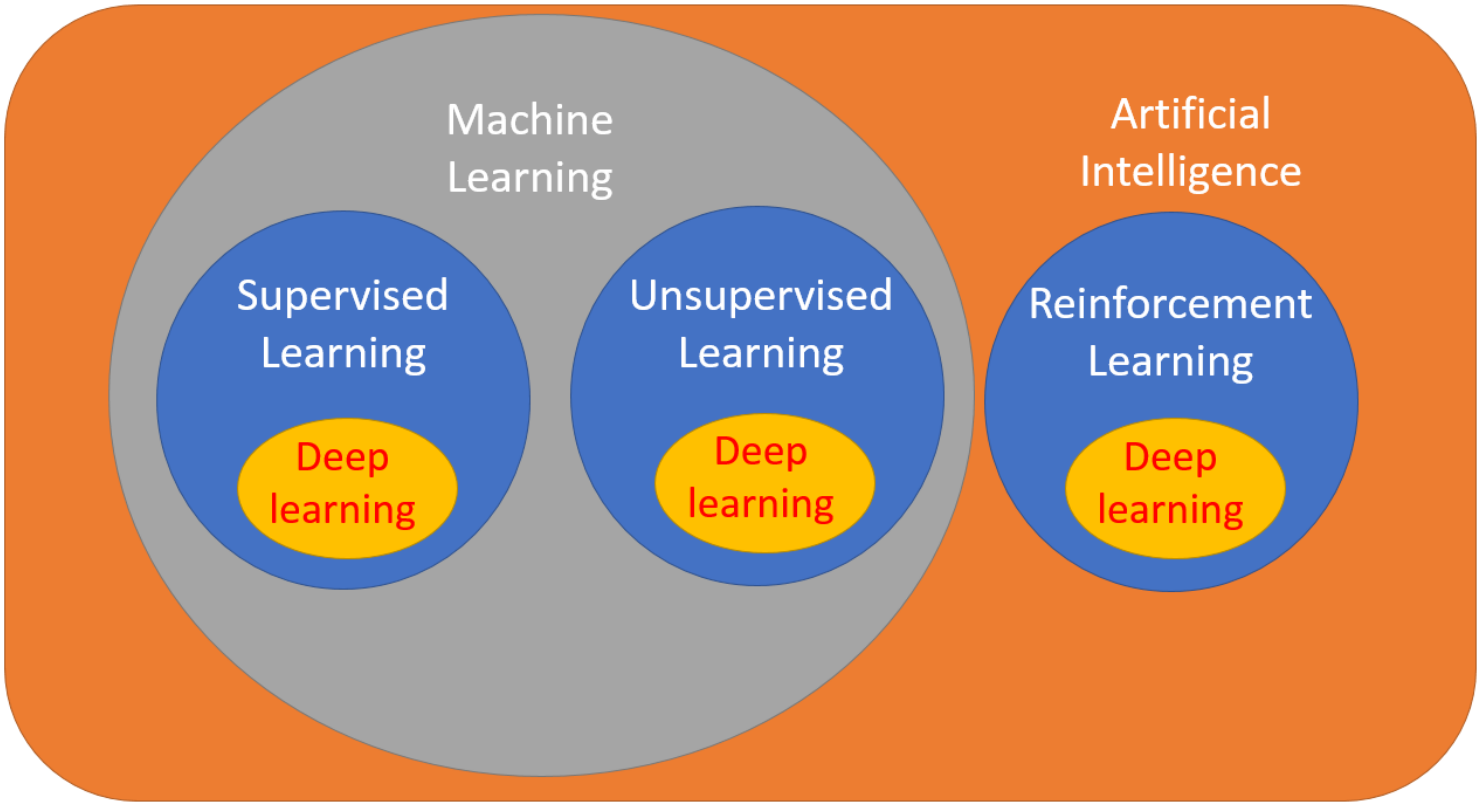
YES: media type data

1. Text, language, speech
2. Images, video, maps
3. Time-series, stocks, valuta
4. Games, robots

NO: categorical data

1. Properties
2. Features
3. Categories

Overview of the field



Neural Networks

Neural nets

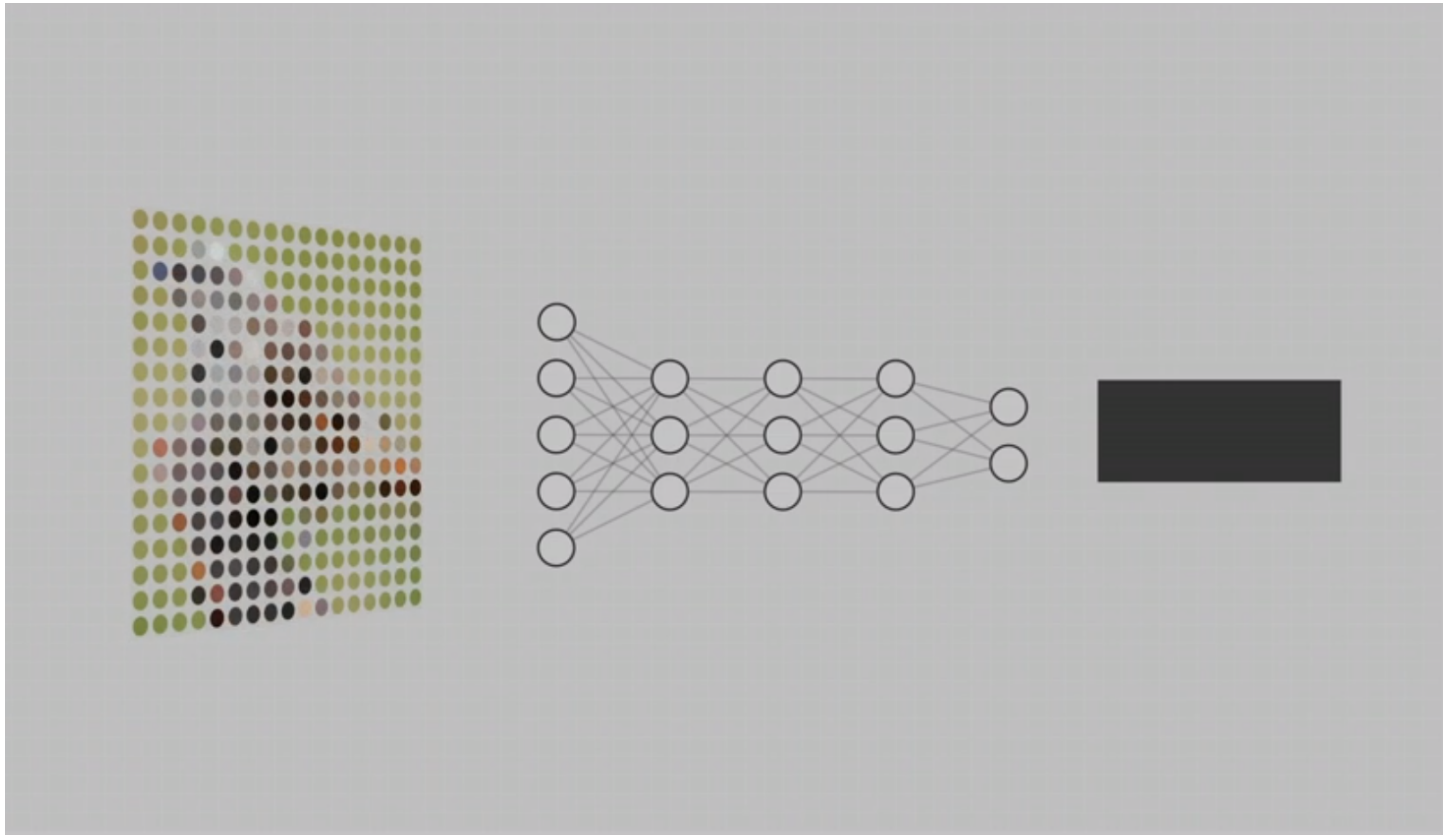
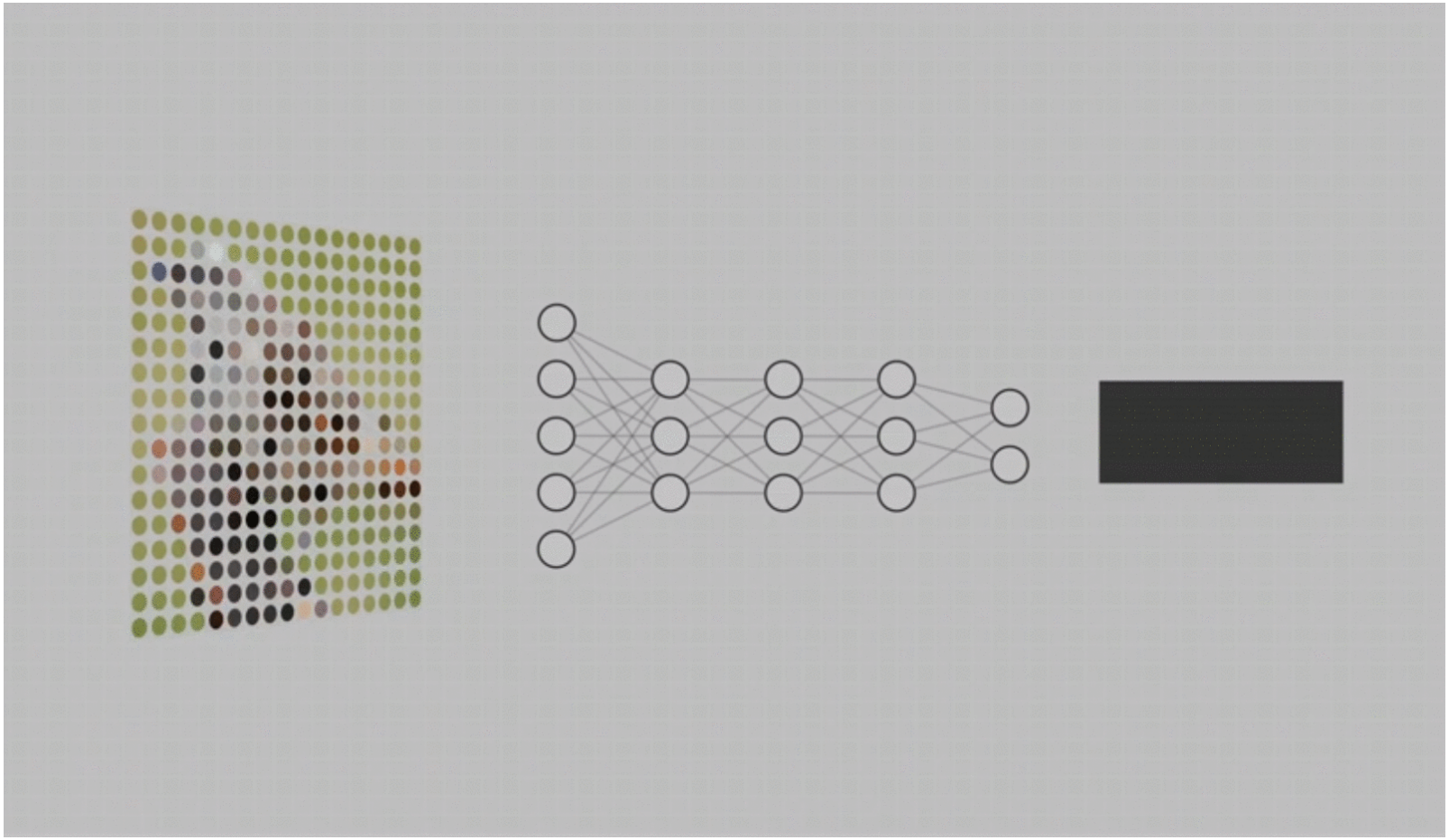
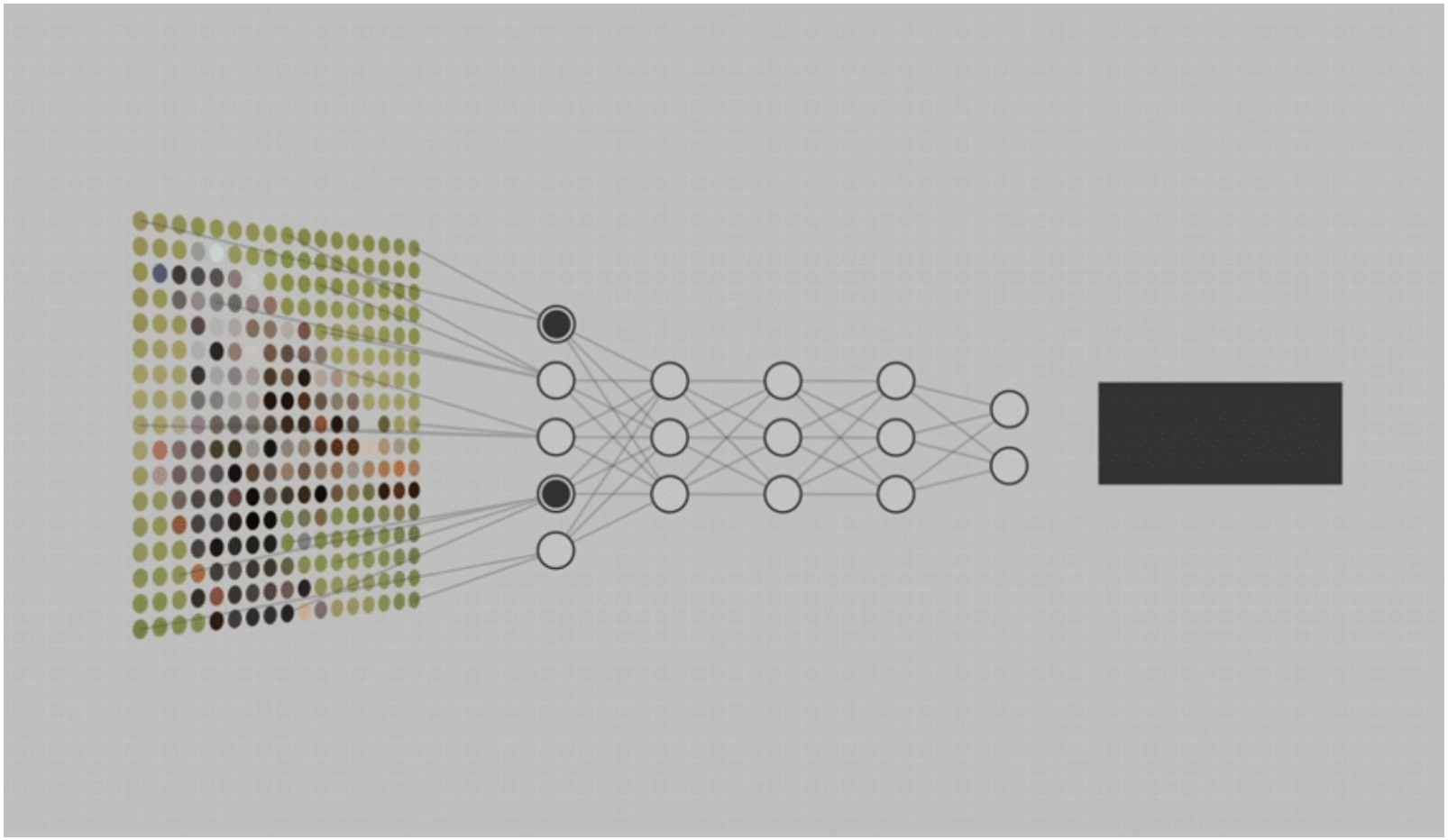


Figure: Neural network explained (credits: Blaise Aguera y Arcas)

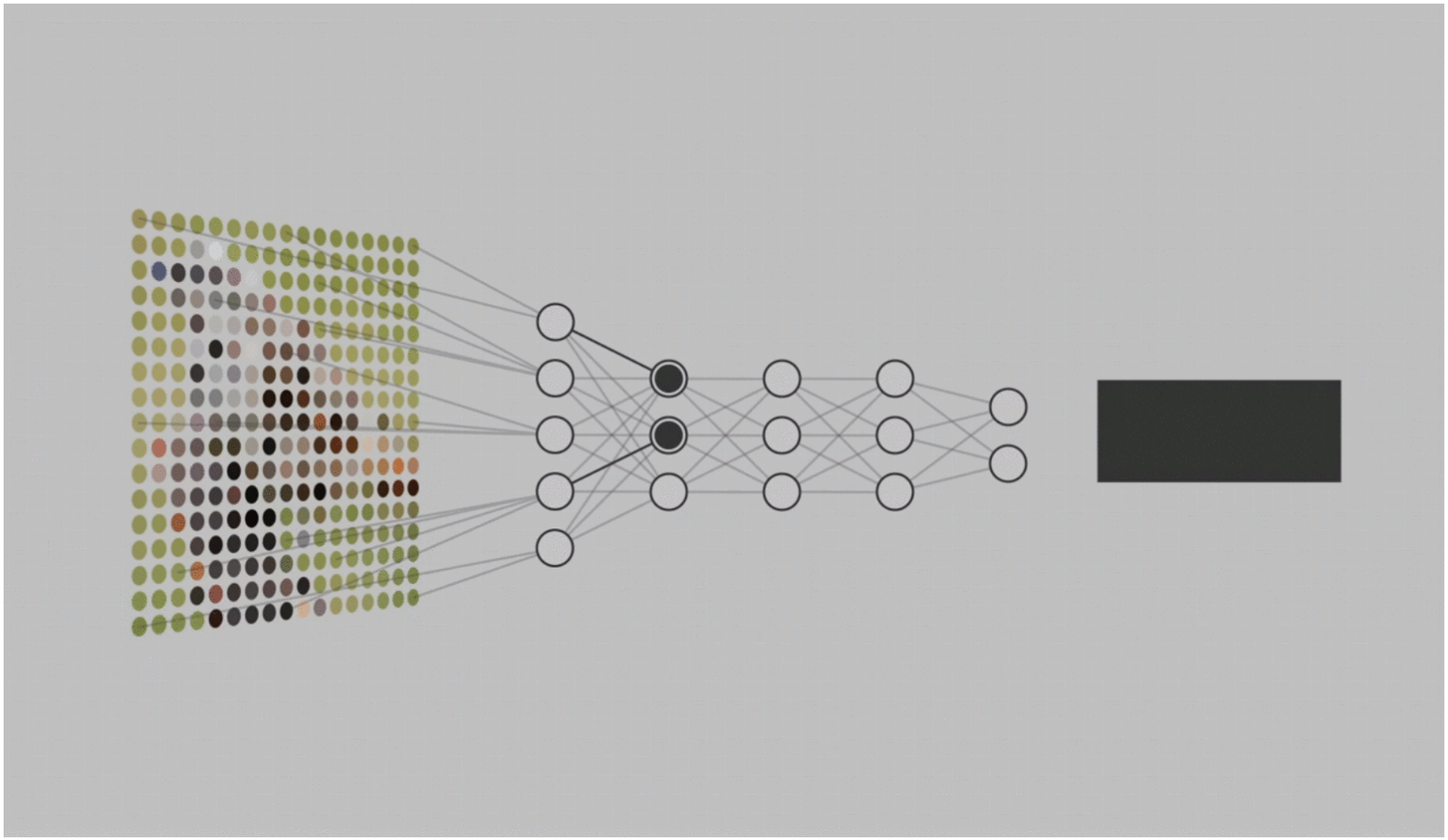
Neural network



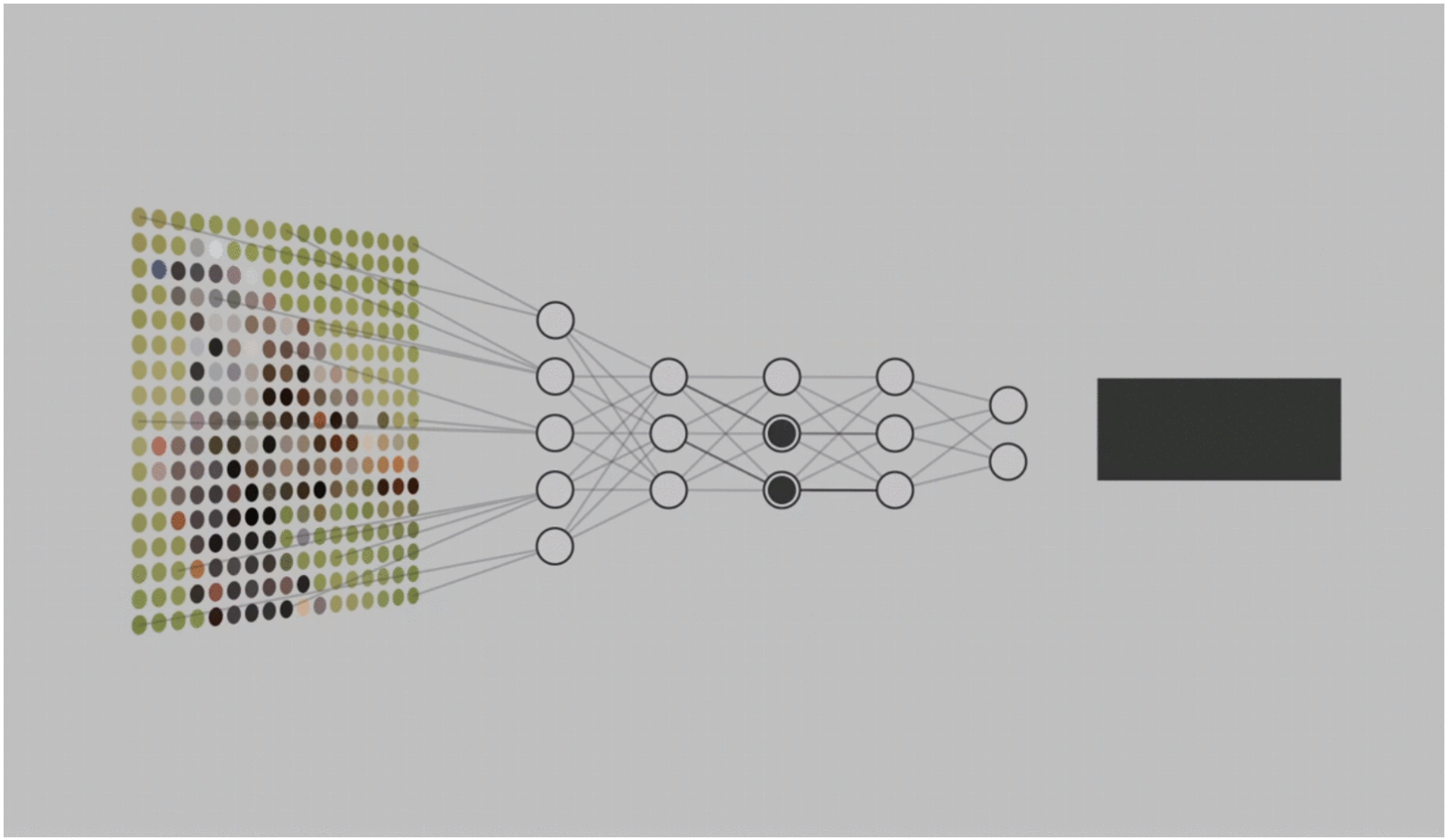
Neural network



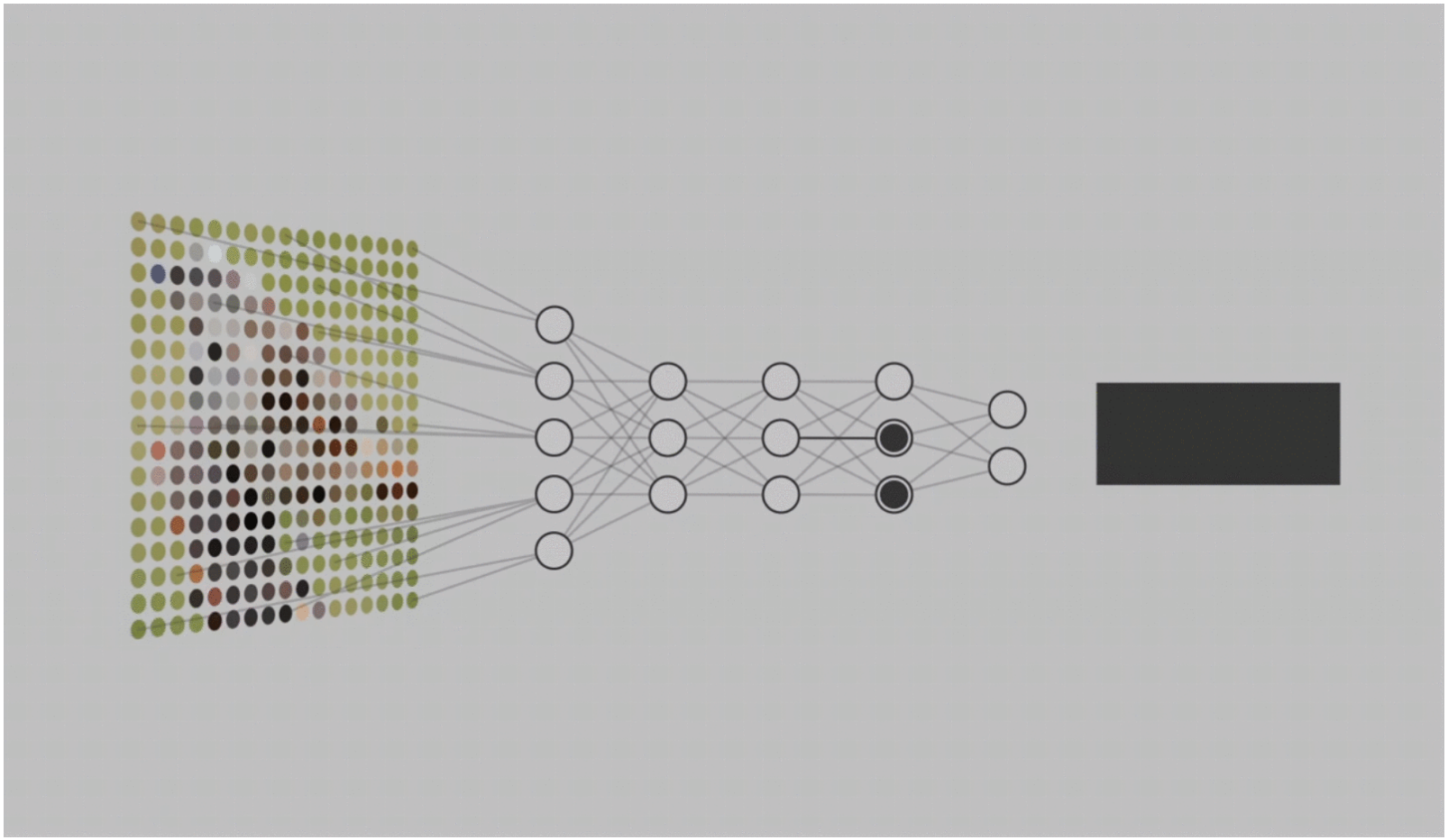
Neural network



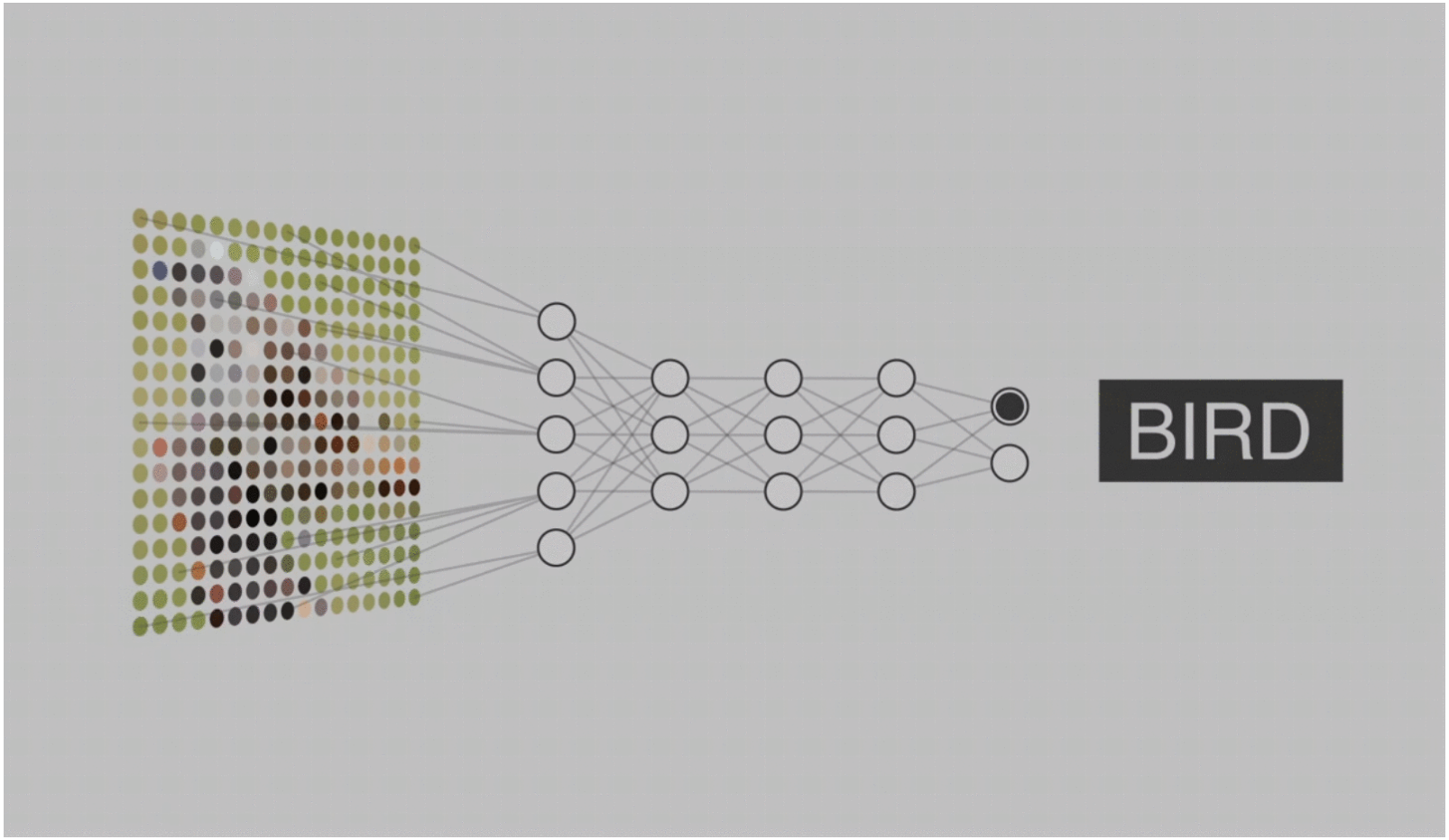
Neural network



Neural network



Neural network



Basic equation

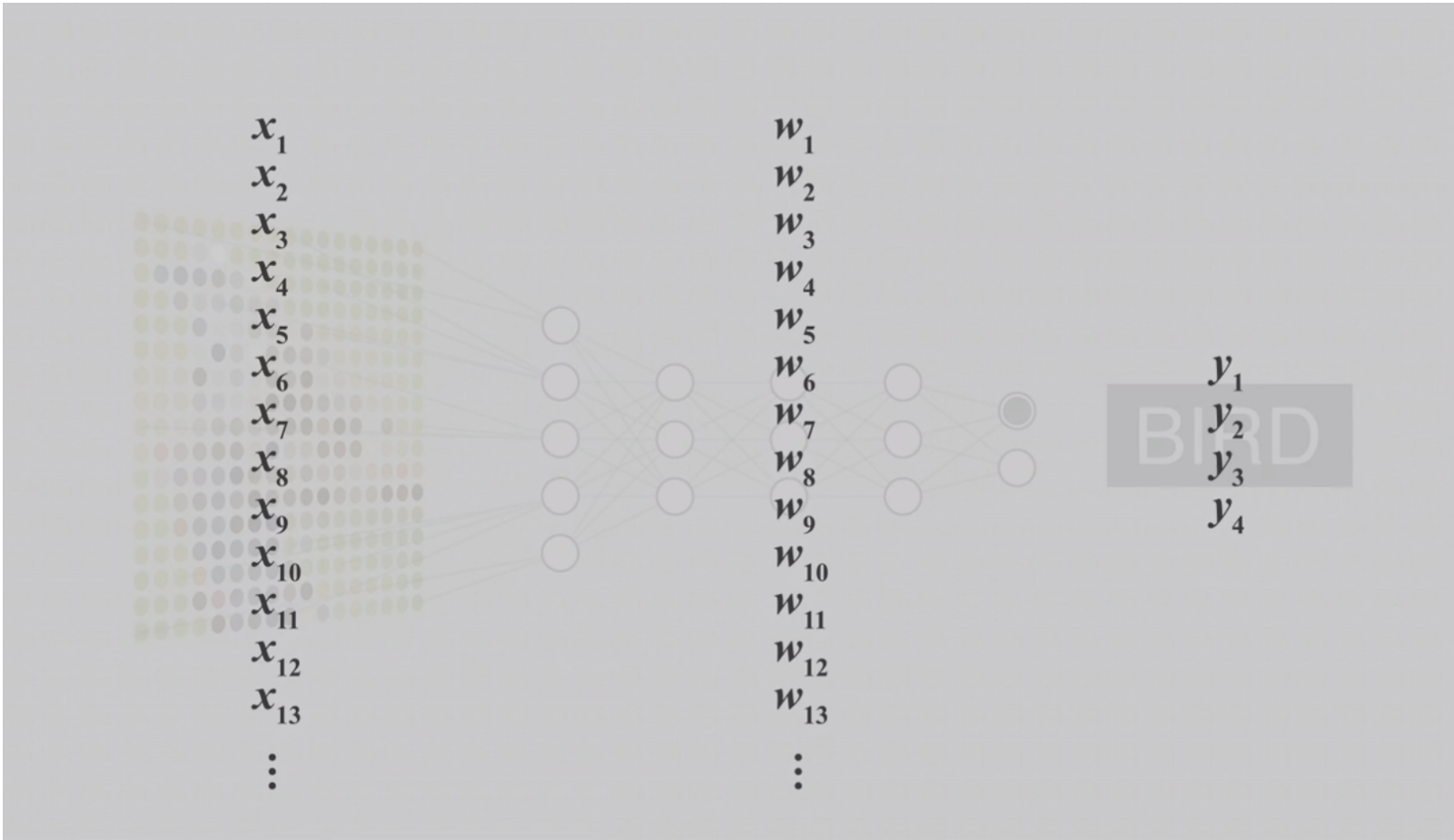


Figure: Neural net in algebraic form (credits: Blaise Aguera y Arcas)

Template equation

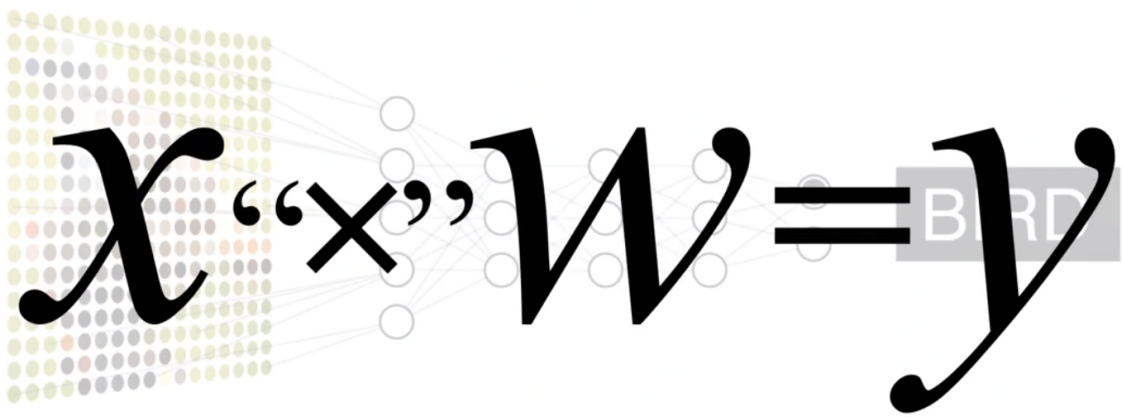


Figure: Template equation neural net (credits: Blaise Aguera y Arcas)

How to use

How to use them

$$w \cdot x = y$$

$$2 \cdot 3 = y$$

Forward inference

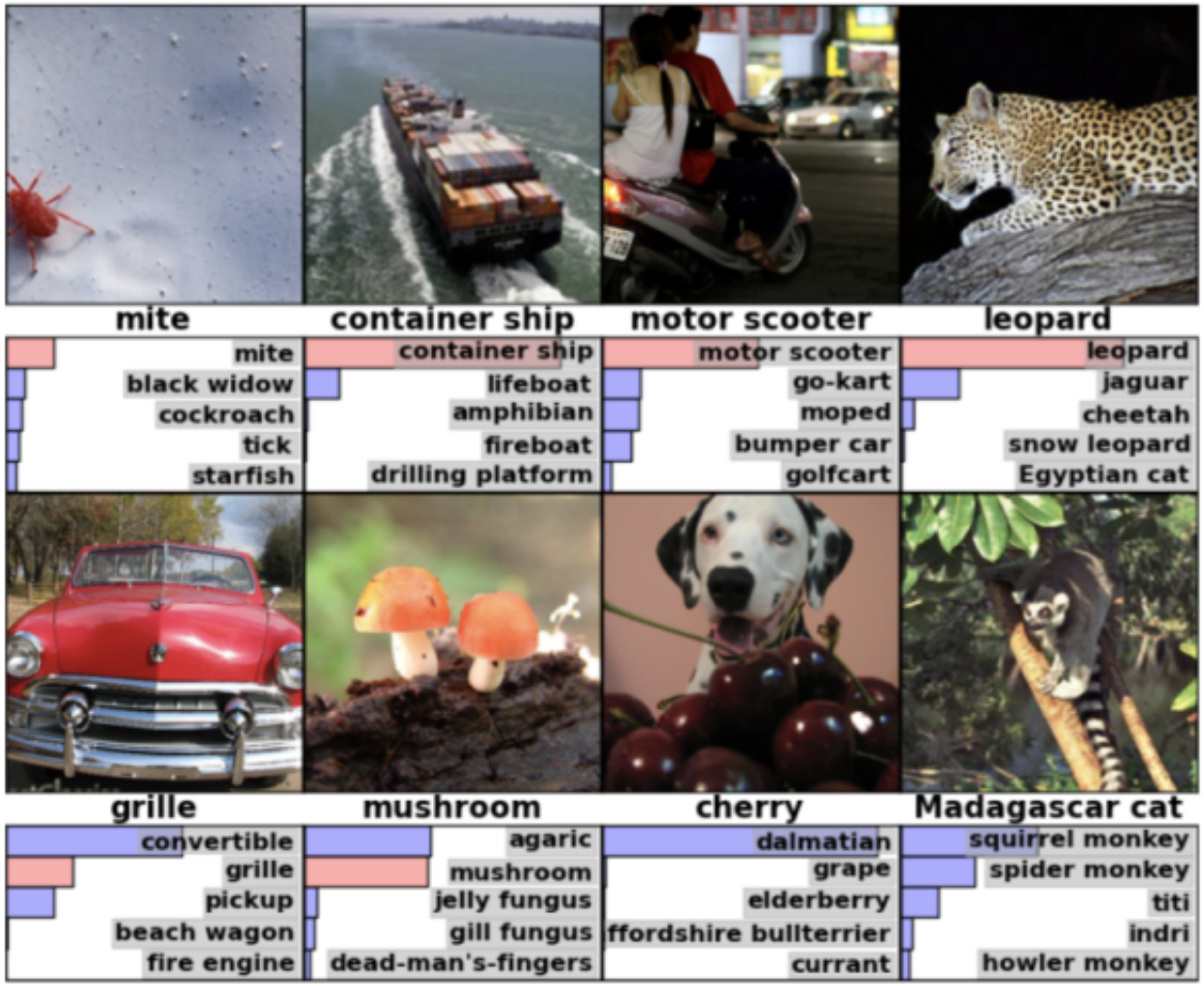


Figure: Forward inference CNN (Krizhevsky et al. 2012)

How to train

How to train them

$$(x, y) = (3, 6)$$

$$w \cdot x = y$$

$$w \cdot 3 = 6$$

$$w \cdot x \approx y$$

$$y - w \cdot x \approx 0$$

$$\text{Loss} = (y - w \cdot x)^2 \Longrightarrow 0$$

Gradient Descent

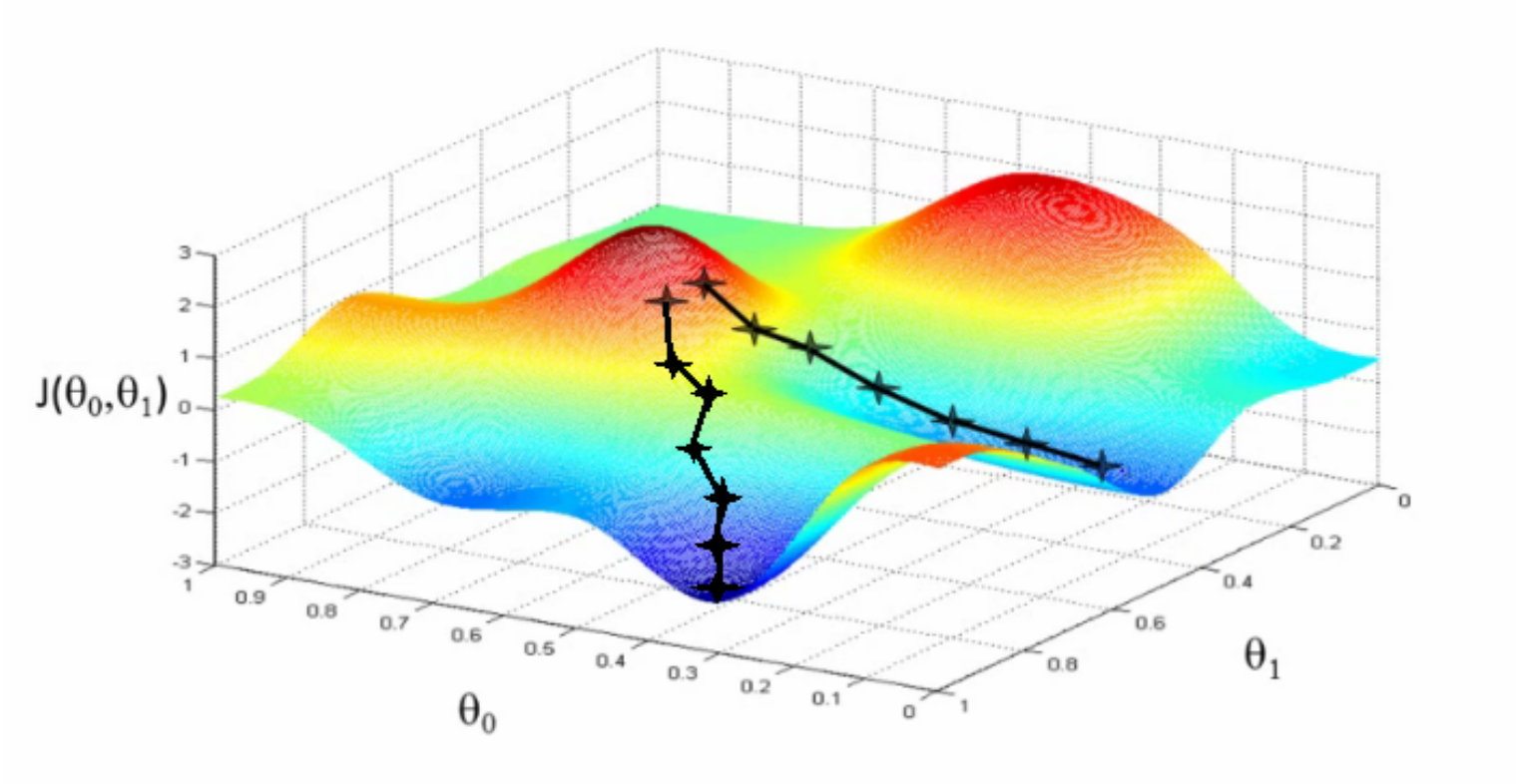


Figure: Credits: ML blog Vasilis Vryniotis

Loss function

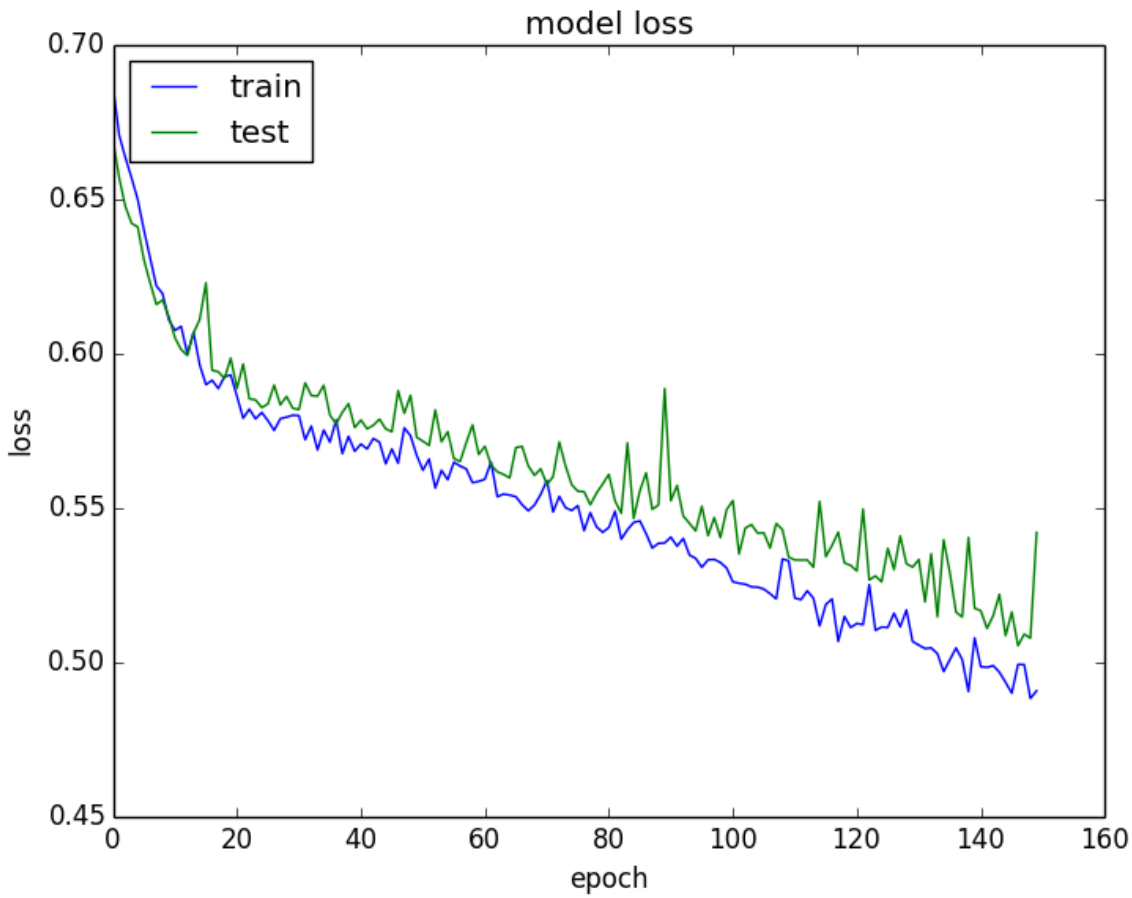


Figure: (http://machinelearningmastery.com/)

Training a neural network

1. Define a neural network
2. Define the loss function

1. Define a neural networkl

One hidden layer

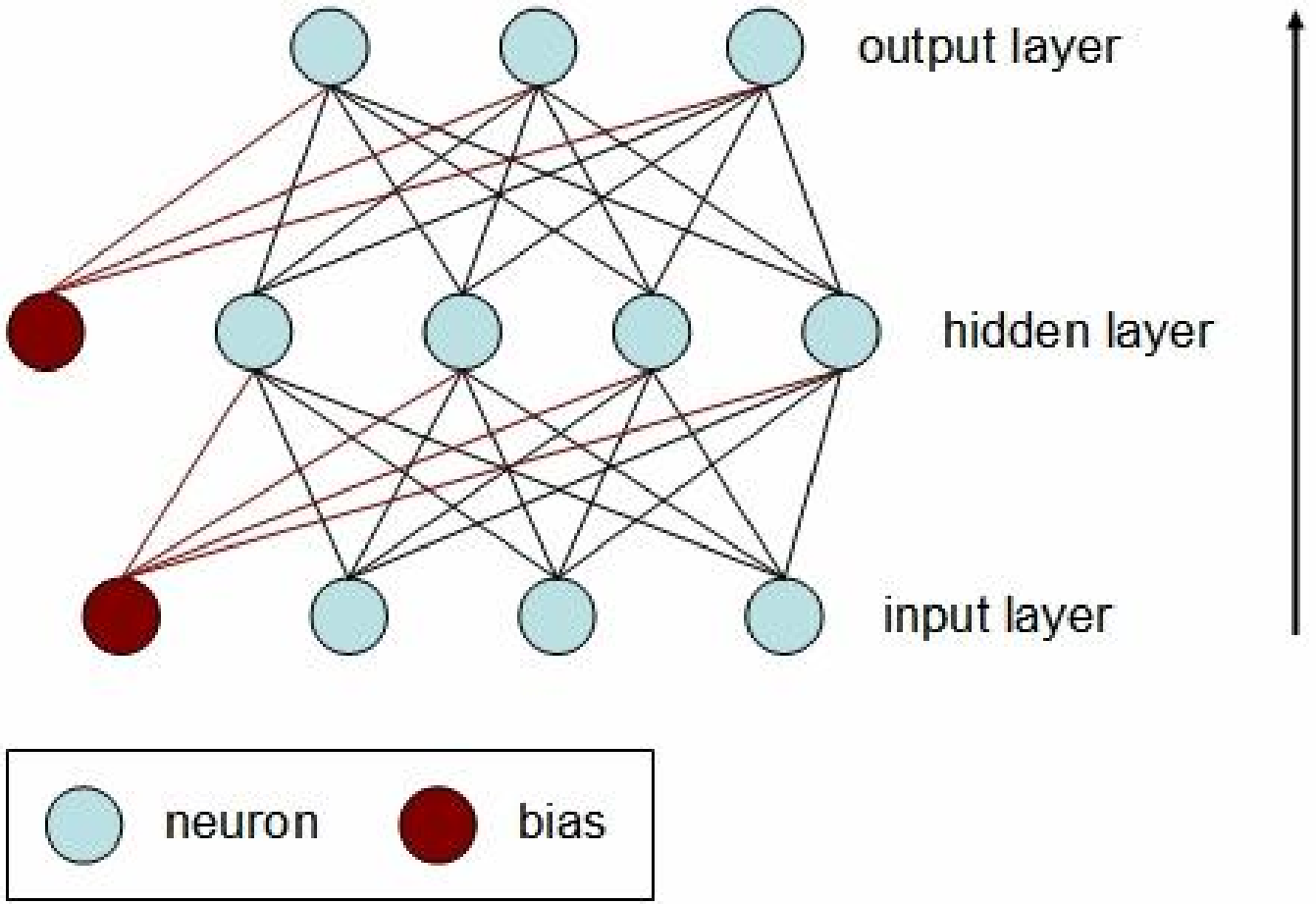


Figure: Neural network with one hidden layer

Activation function

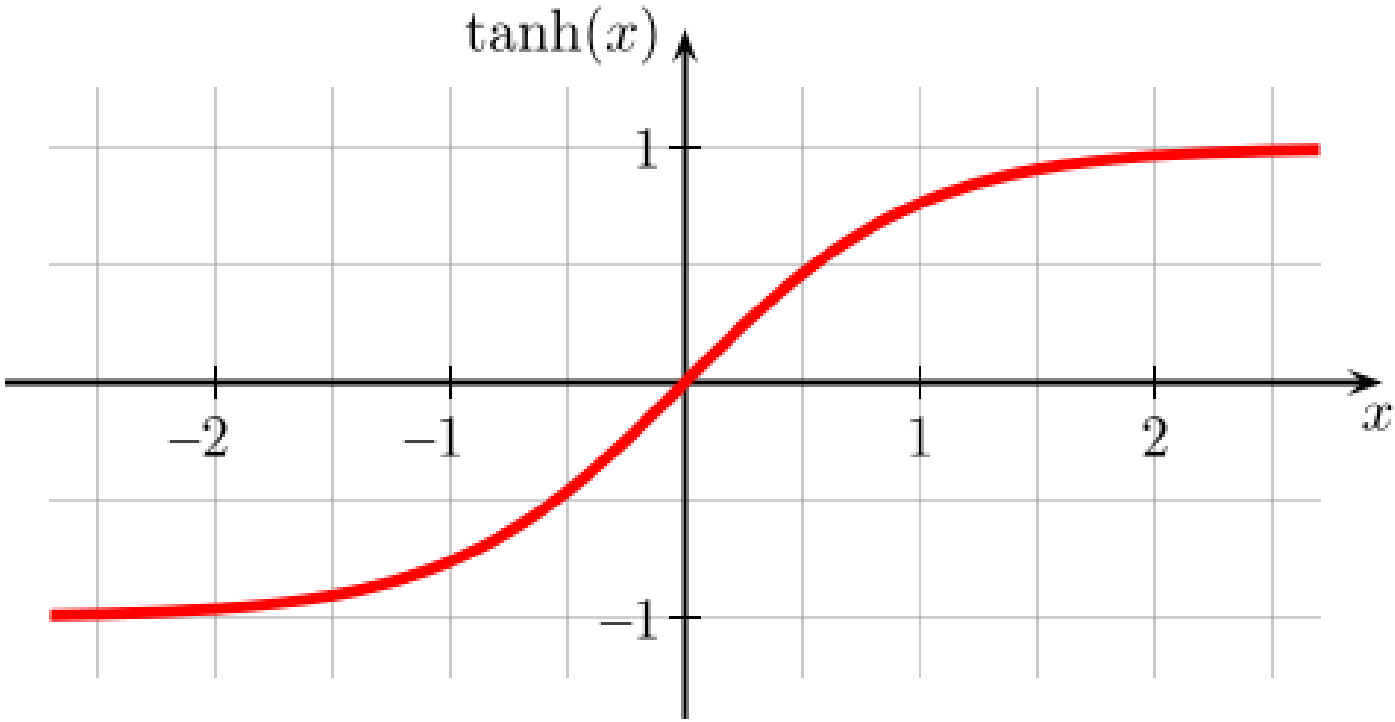


Figure: Hyperbolic tangent as activation function

Python code


```
#FF
def step(x):
    h = np.tanh(np.dot(x, W1) + b1)
    y = np.dot(h, W2) + b2
    return y

y = step(x)
```

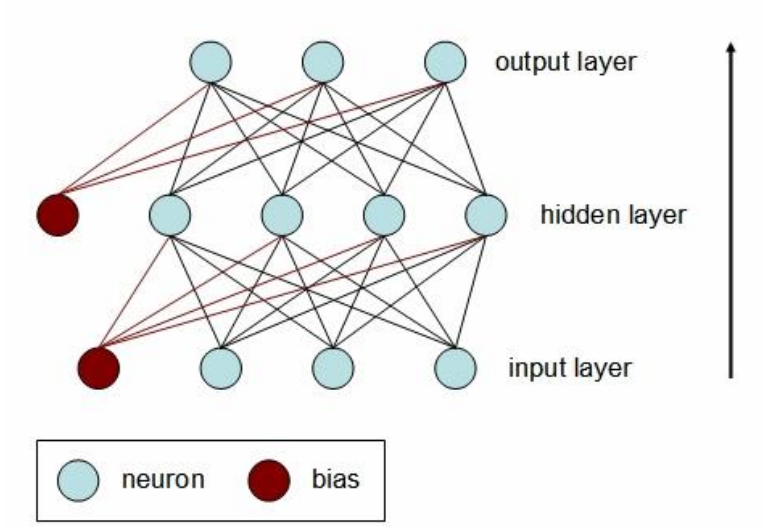


Figure: Neural network with one hidden layer

```
W1.shape => (3, 4)
b1.shape => (4,)
W2.shape => (4, 3)
b2.shape => (3,)
```

2. Define the loss function

Regression

$$Loss = (y_{correct} - \hat{y})^2$$

Classification

$$Loss = -\sum_i y_{correct} log(\hat{y})$$

Softmax

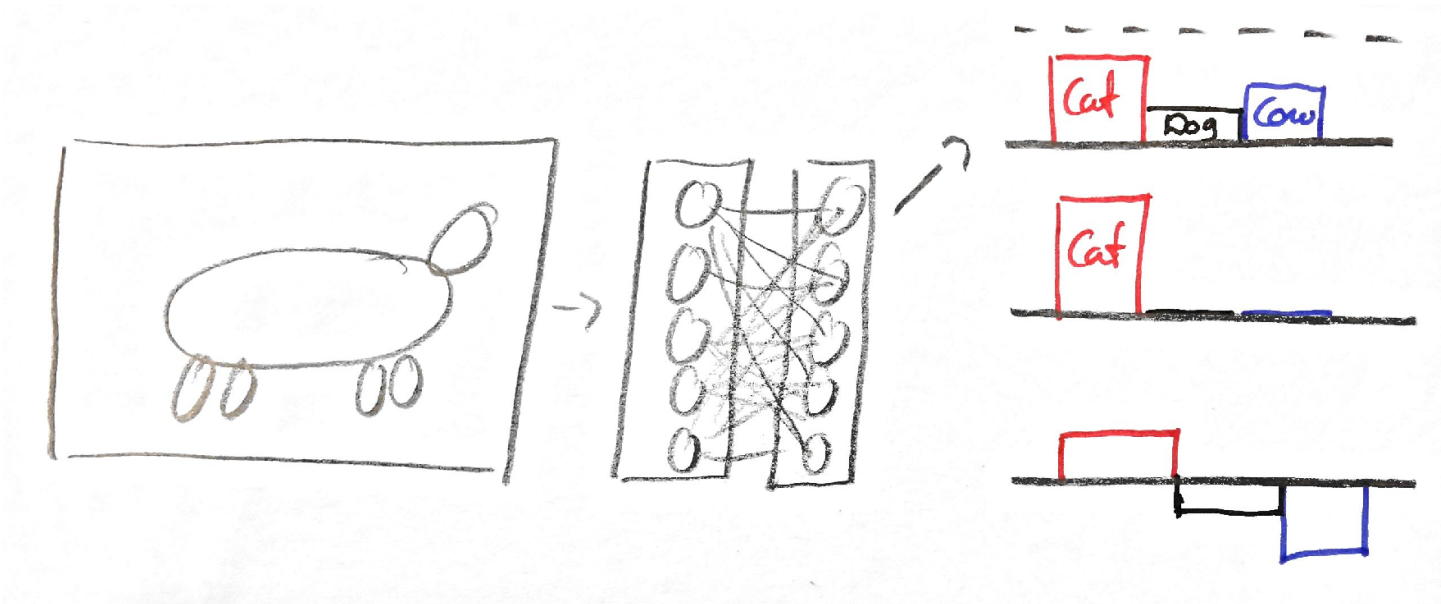


Figure: Gradient from softmax

Equations

1. Softmax

$$softmax(a)_i = \frac{e^{a_i}}{\sum_j e^{a_j}}$$

2. Cross entropy

$$loss = cross(p, q) = -\sum_j p_j log(q_j) = -log(q_{y_{correct}})$$

3. Gradient

$$\frac{\delta Loss}{\delta a_i} = q_i - p_i$$

3. Do stochastic gradient descent

Gradient Descent

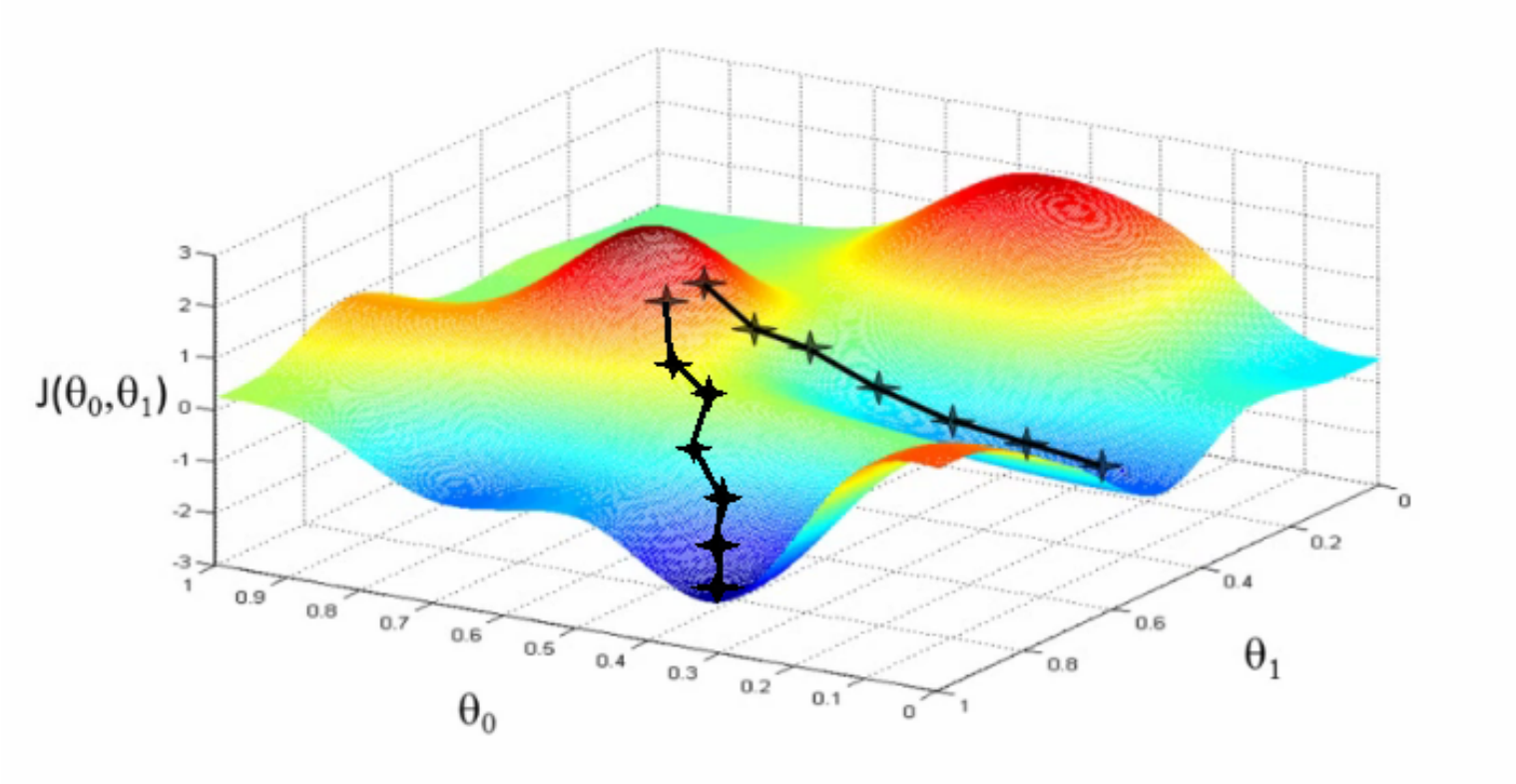


Figure: Credits: ML blog Vasilis Vryniotis

Minimize loss

$$w = w - \eta \frac{\delta Loss}{\delta w}$$

$$\frac{\delta Loss}{\delta w} = \frac{\delta Loss}{\delta a} \frac{\delta a}{\delta h_l} \frac{\delta h_l}{\delta h_{l-1}} \frac{\delta h_{l-1}}{\delta h_{l-2}} \cdots \frac{\delta h_1}{\delta h_0} \frac{\delta h_0}{\delta w}$$

Stochastic gradient descent

Use small batches of size e.g. 32

Two problems

1. Too many weights

$3 \times 1024 \times 100 + 100 \times 3 = 307500$

2. No domain knowledge

Two architectures

1 Convolution

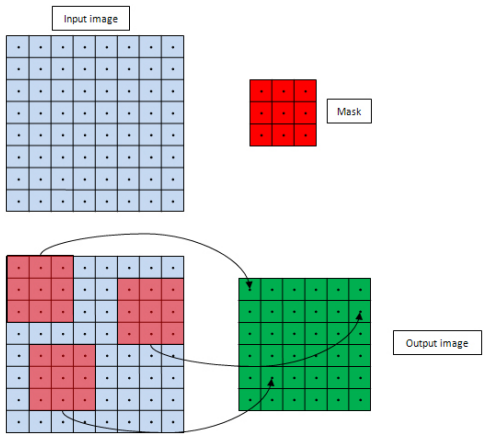


Figure: developer.amd.com

2 Recursion

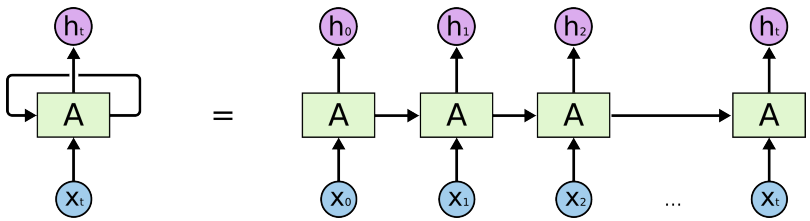


Figure: credits: colah.github.io

Detect patterns

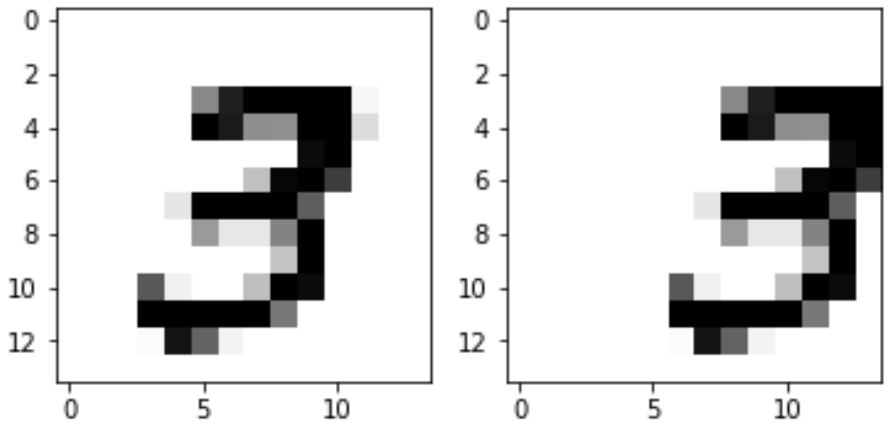


Figure: Translation of image does not change class

Convolution

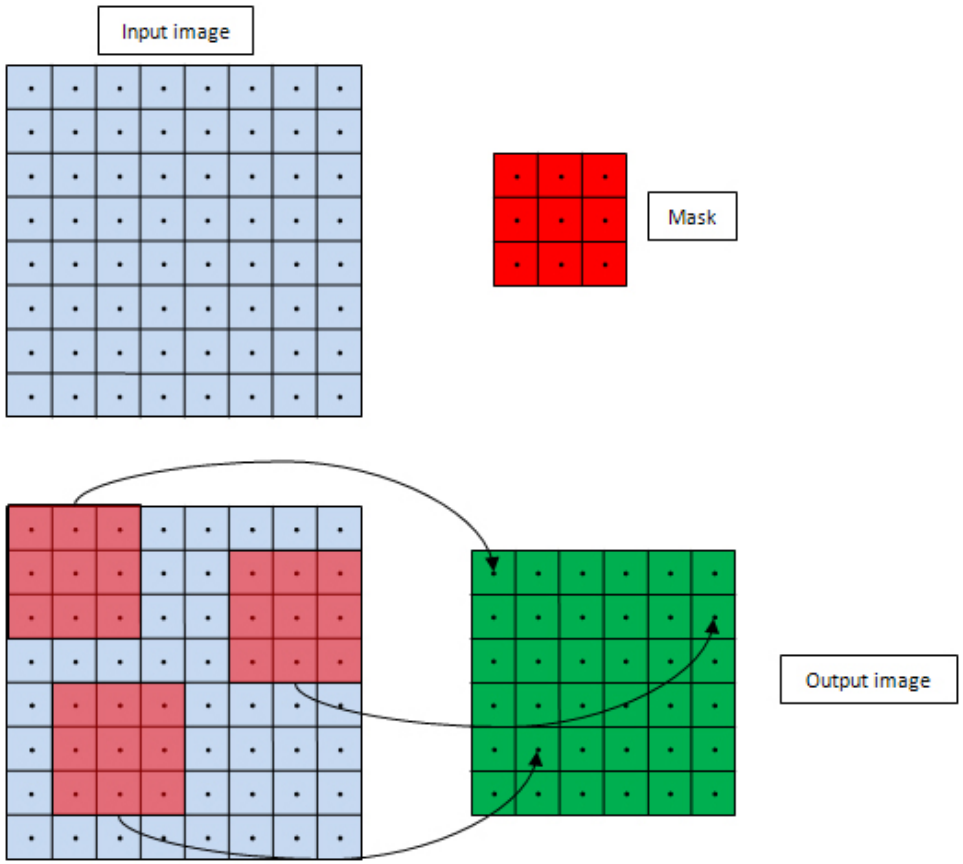


Figure: developer.amd.com

Python code


```
#CNN
def step(x):
  h = np.tanh(convolve(x, W1) + b1)
  y = np.dot(h.flatten(), W2) + b2
  return y

y = step(x)
```

CNN

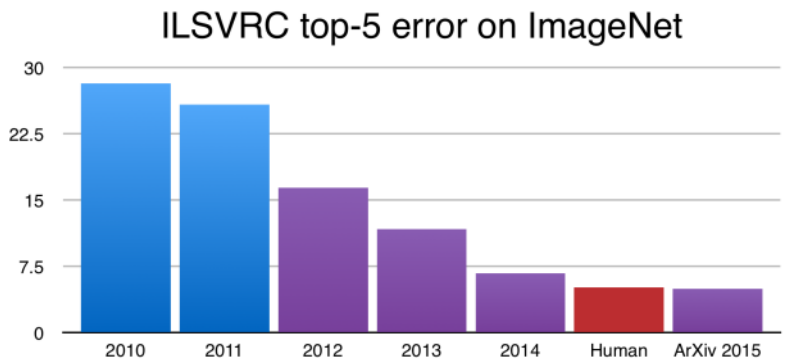


Figure: ImageNet results (credits jackkelly.github.io/)

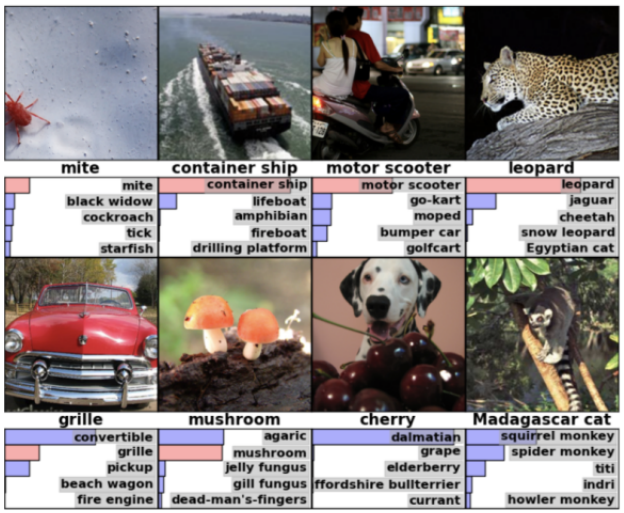


Figure: Forward inference CNN (Krizhevsky et al. 2012)

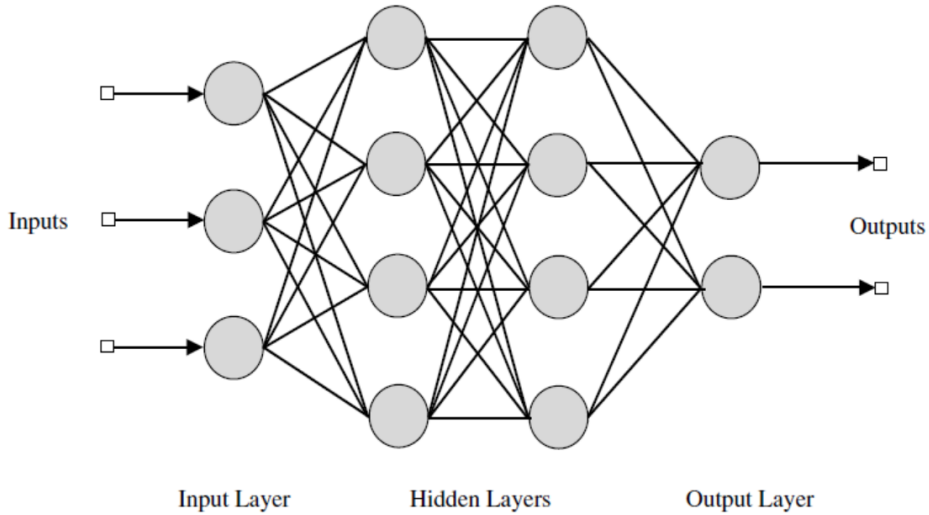


Figure: Feedforward neural netword

$$3 \times 1024 \times 100 + 100 \times 3 = 307500$$

$$3 \cdot 5 \cdot 5 \cdot 100 + 100 \times 3 = 7800$$

Recursion

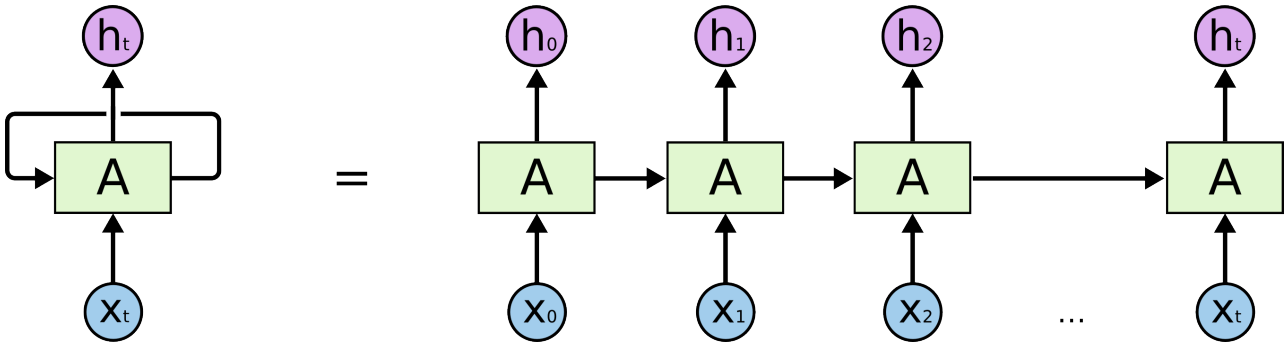
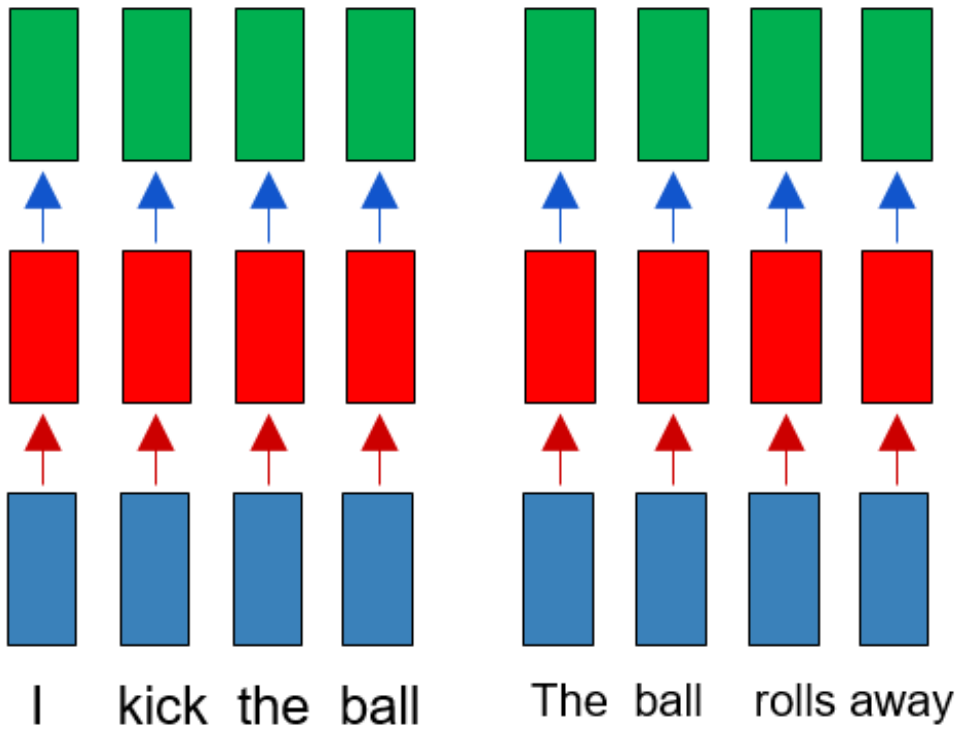


Figure: credits: colah.github.io

Word tagging



Connect over time

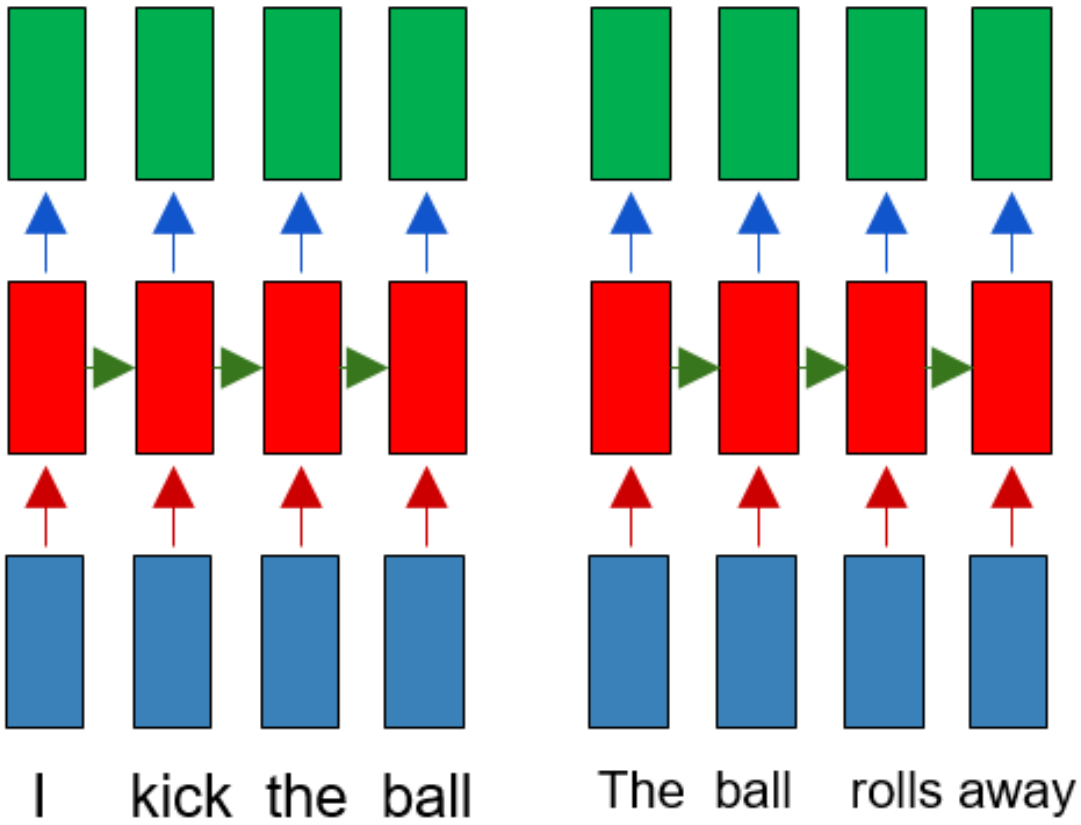


Figure: RNN in 112 lines

Python code


```
#FF
def step(x):
  h = np.tanh(np.dot(x, W1) + b1)
  y = np.dot(h, W2) + b2
  return y

y = step(x)
```

Python code


```
# RNN
rnn = RNN()

class RNN:
  def step(self, x):
    self.h = np.tanh(np.dot(self.h, W1)
          + np.dot(x, W2))
    y = np.dot(self.h, W3)
    return y

y = rnn.step(x)
```

Python code


```
#FF
def step(x):
  h = np.tanh(np.dot(W1, x) + b1)
  y = np.dot(W2, h) + b2
  return y
#RNN
class RNN:
  def step(self, x):
    self.h = np.tanh(np.dot(W1, self.h)
          + np.dot(W2,x))
    y = np.dot(W3, self.h)
    return y
```

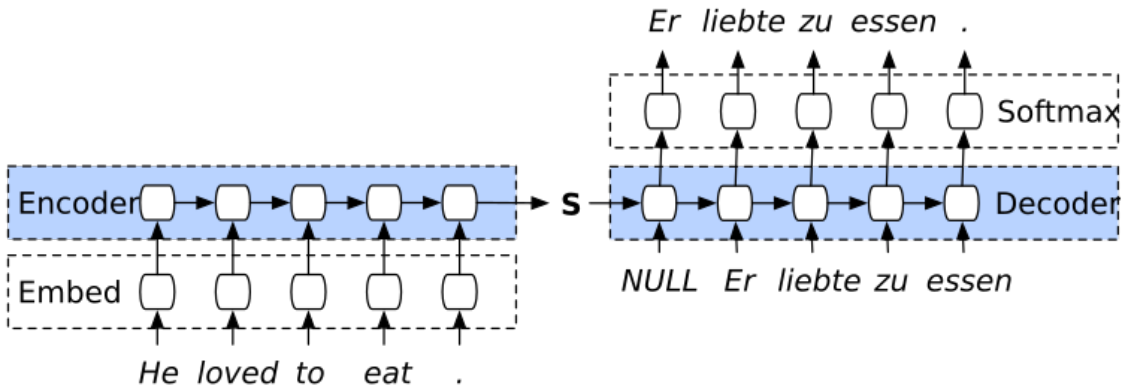


Figure: RNN for Machine Translation (credits: smerity.com)

Long Short-term memory

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i)$$
$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f)$$
$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o)$$

$$c_t = f_t c_{t-1} + i_t tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c)$$
$$h_t = o_t \ tanh(c_t)$$

LSTM

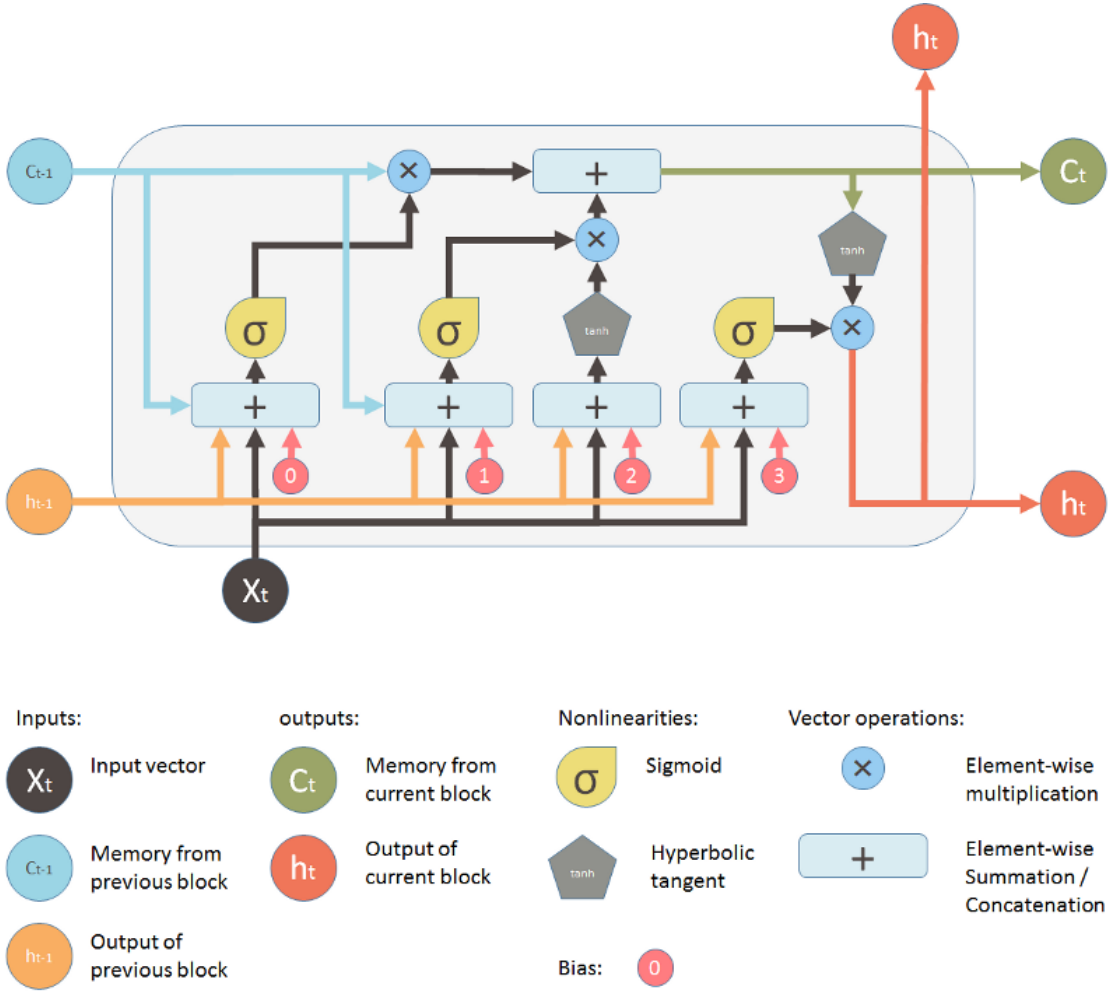


Figure: Diagram depicting LSTM block
Author: Shi Yan. Source: medium.com/@shiyan/

Span

1 CNN

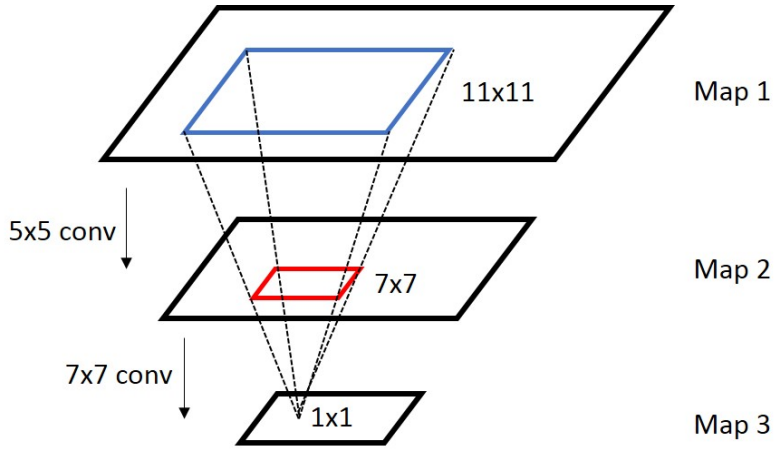


Figure: credits: cvmarcher.com/

2 RNN

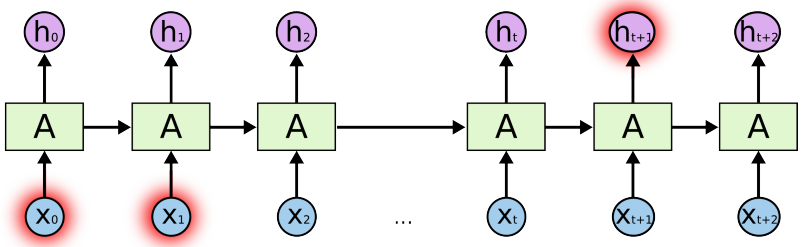


Figure: credits colah.github.io

Overview			
	FF	RNN	CNN
- Basic computation	Matix multiplies	Recurrent multiplications	Convolutions
- Typical data	Feature vectors	Sequences	Images
- Weights ties	No	Sequential	Spatial
- Effective span	Across input	Long term	Local receptive field
- How to train	Backprop	Backprop	Backprop
- Performance	No guarantee	No guarantee	No guarantee
- Typical examples	None	Machine translation, speech tagging, text generation	segmentation, image processing, object detection

- Data
- Libraries
- Start simple
- Cloud training

Gather data

1. SMALL:
    1. MNIST: 60.000 samples
    2. TIMIT: 630 speakers, 10 sentences
2. NORMAL:
    1. Imagenet: 3.2 million images
    2. FLICKR: 1 million images
    3. Word vectors on all of Wikipedia

Semi supervised learning

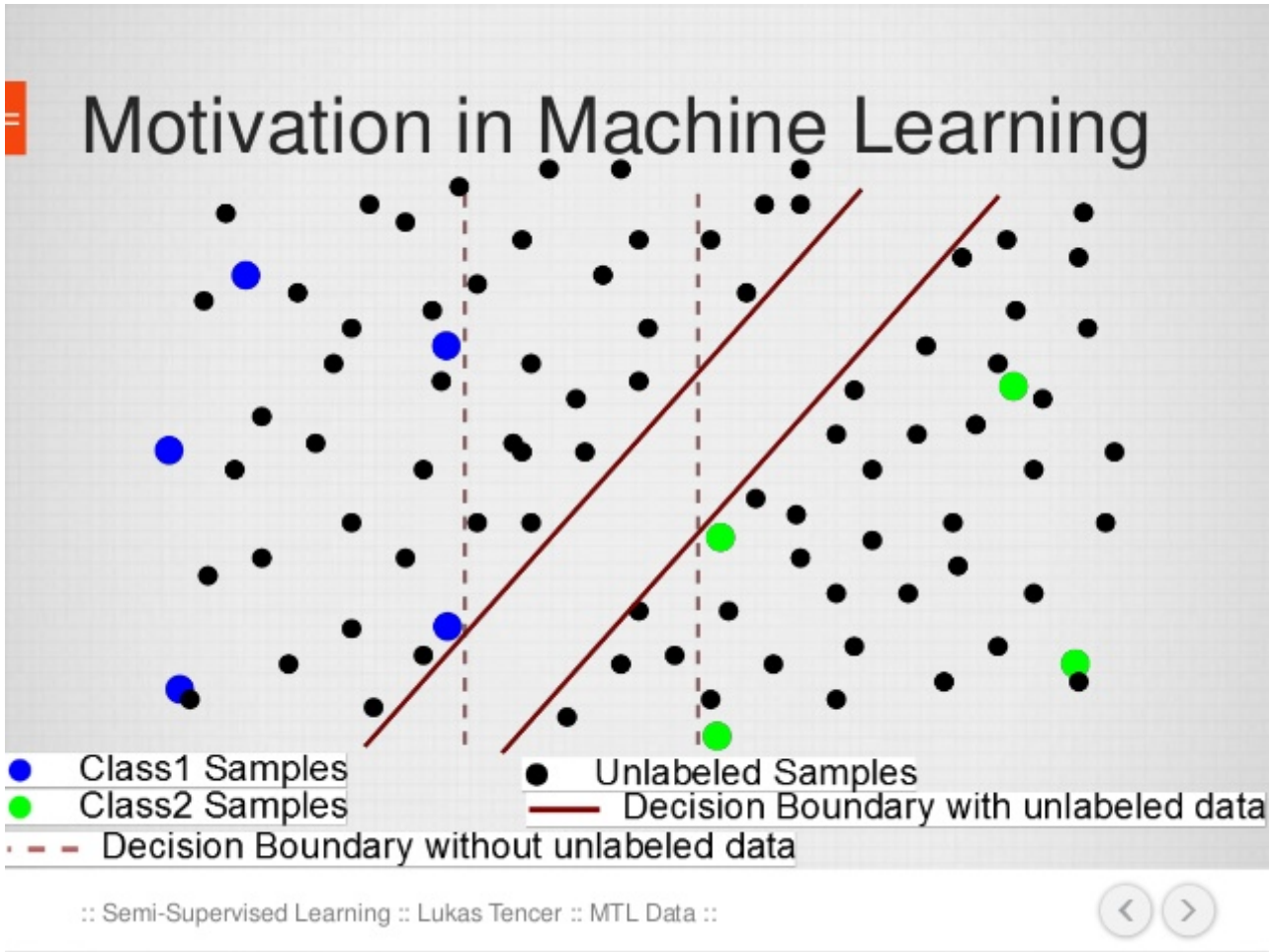


Figure: Credits: Lukas Tencer

MNIST

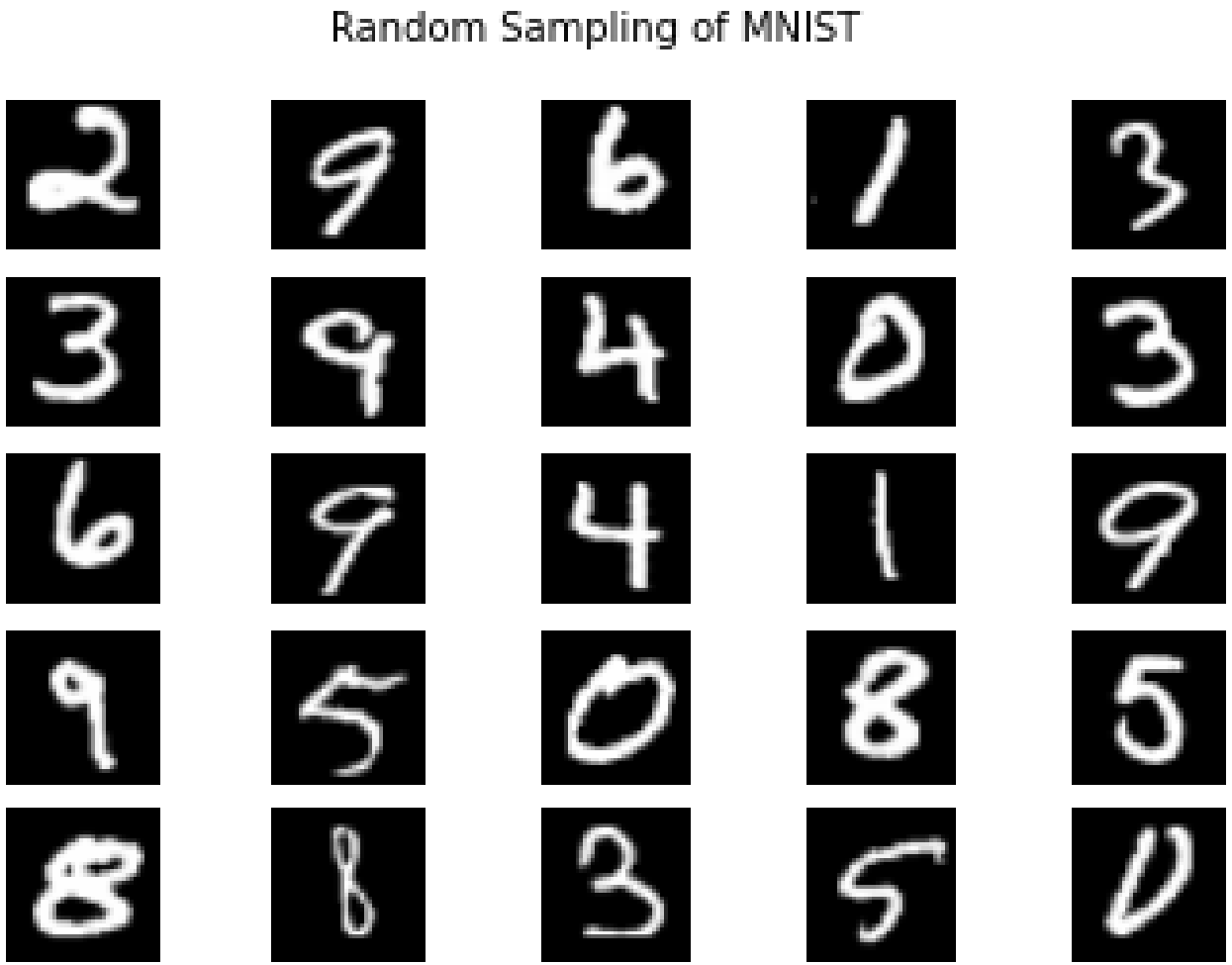


Figure: MNIST images

Code

1. Low level
    1. Tensorflow
    2. Torch, pyTorch
2. High level
    1. Keras
    2. Scikit flow, pretty tensor, tfslim, ...

Train

1. No bells and whistles
2. Small sub dataset
3. See if you can overfit

Choose algorithm

1 Convolution

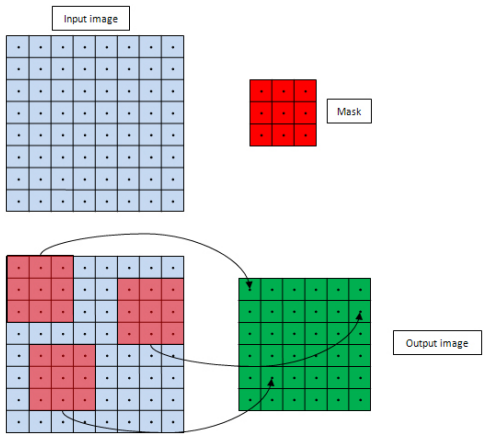


Figure: developer.amd.com

2 Recursion

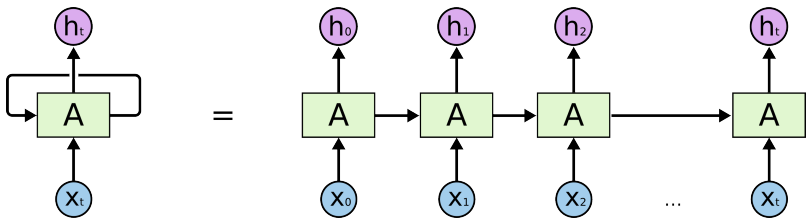


Figure: credits: colah.github.io

Bells and whistles

1. Batch normalization, layer normalization
2. Dropout
3. Activation functions
4. Residual connections
5. etcetera, etcetera

Activation functions

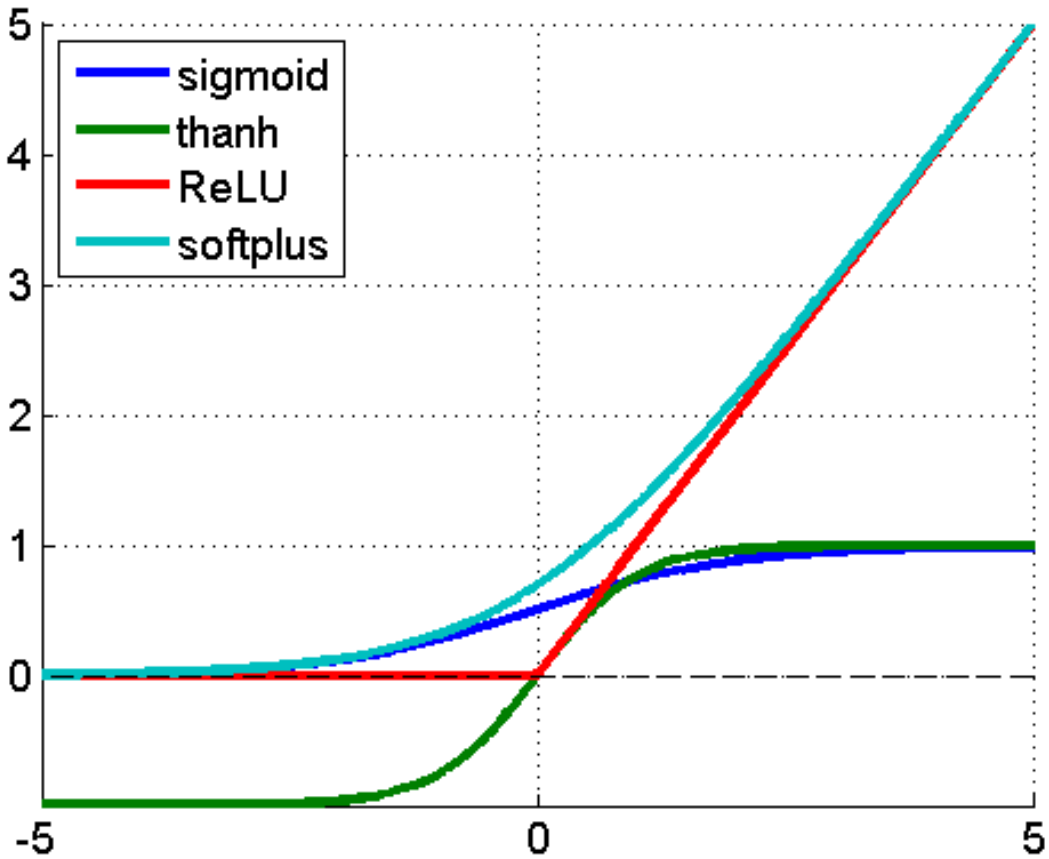


Figure: Common activation functions (credits Vanessa Imiloa)

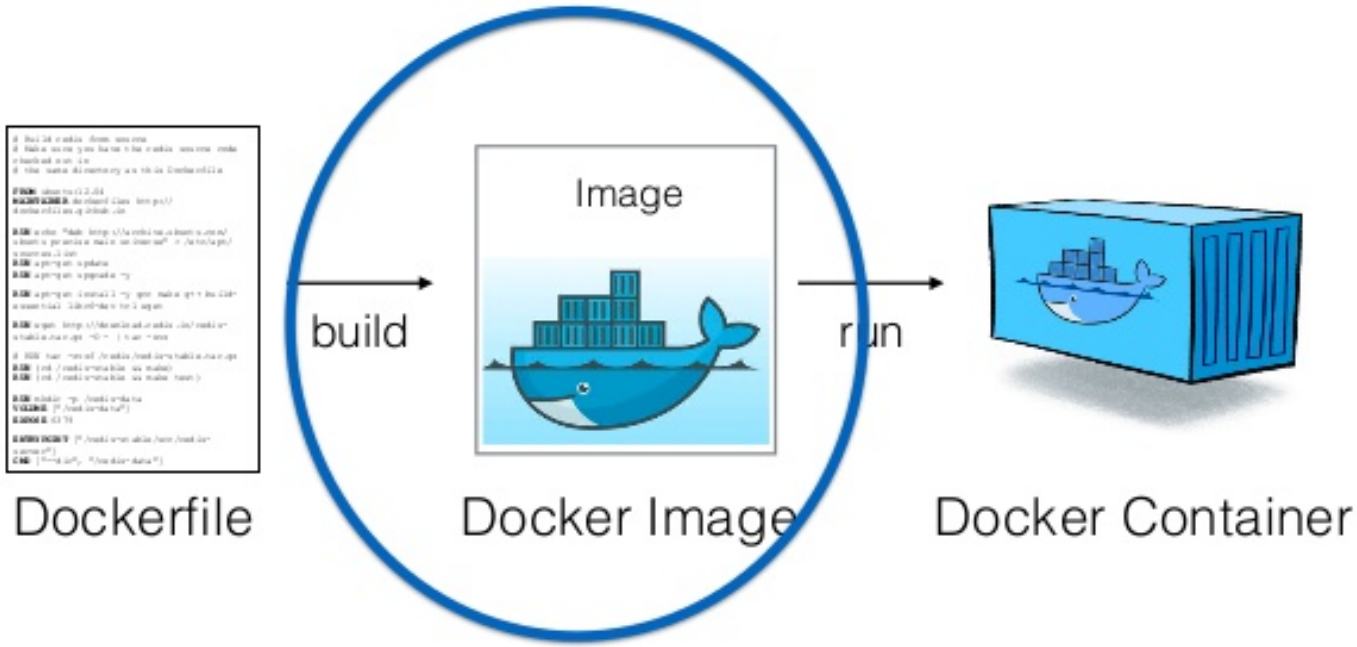
Play with activation functions here

Cloud services

1. AWS
2. Azure
3. Google cloud

Docker

Single Container Docker Workflow



Docker pipeline

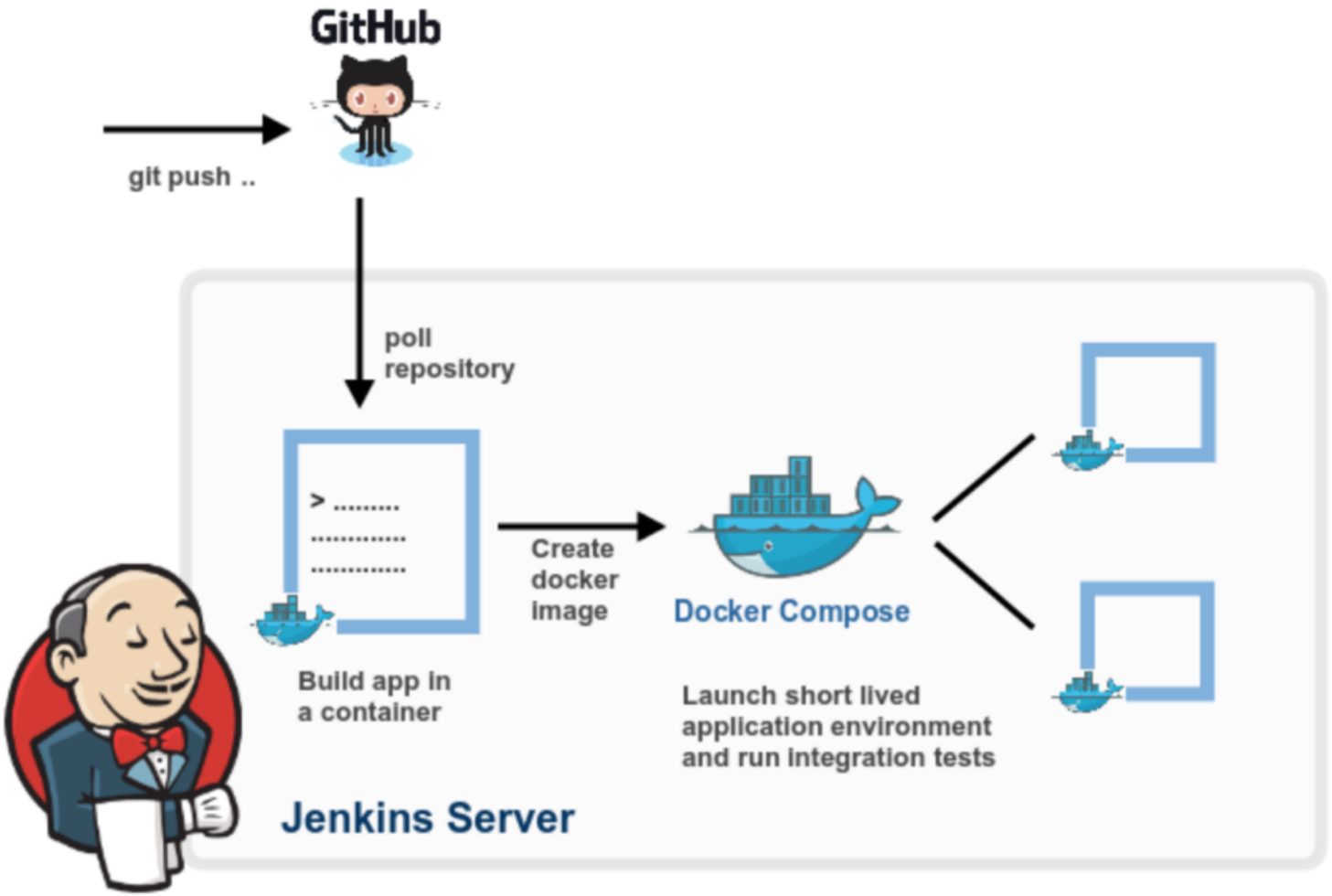


Figure: Source: rancher.com

Gradient Descent

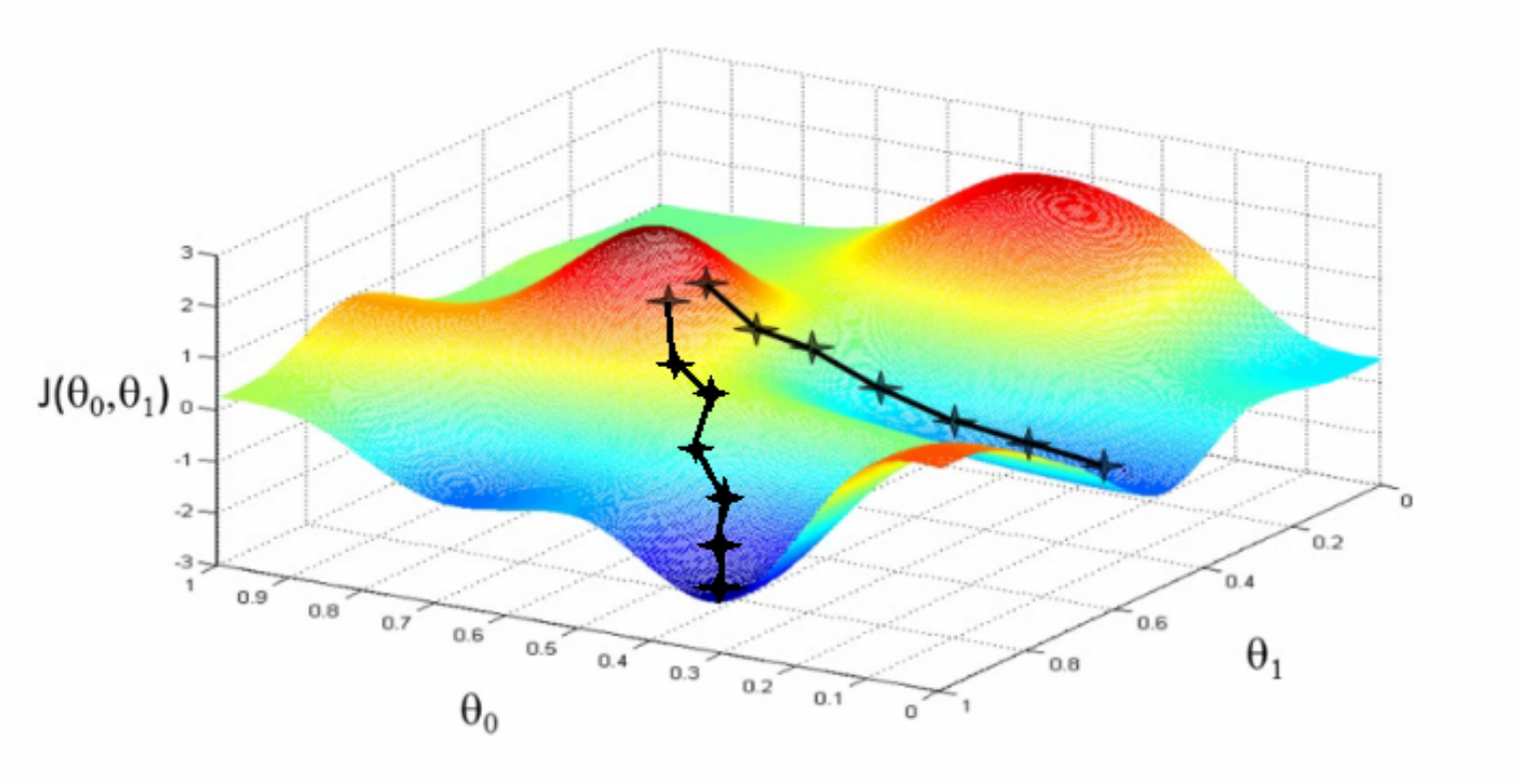


Figure: Credits: ML blog Vasilis Vryniotis

Deploy

DIY

1. Fast inference at testtime
2. Microservices
3. Post processing

Quantization

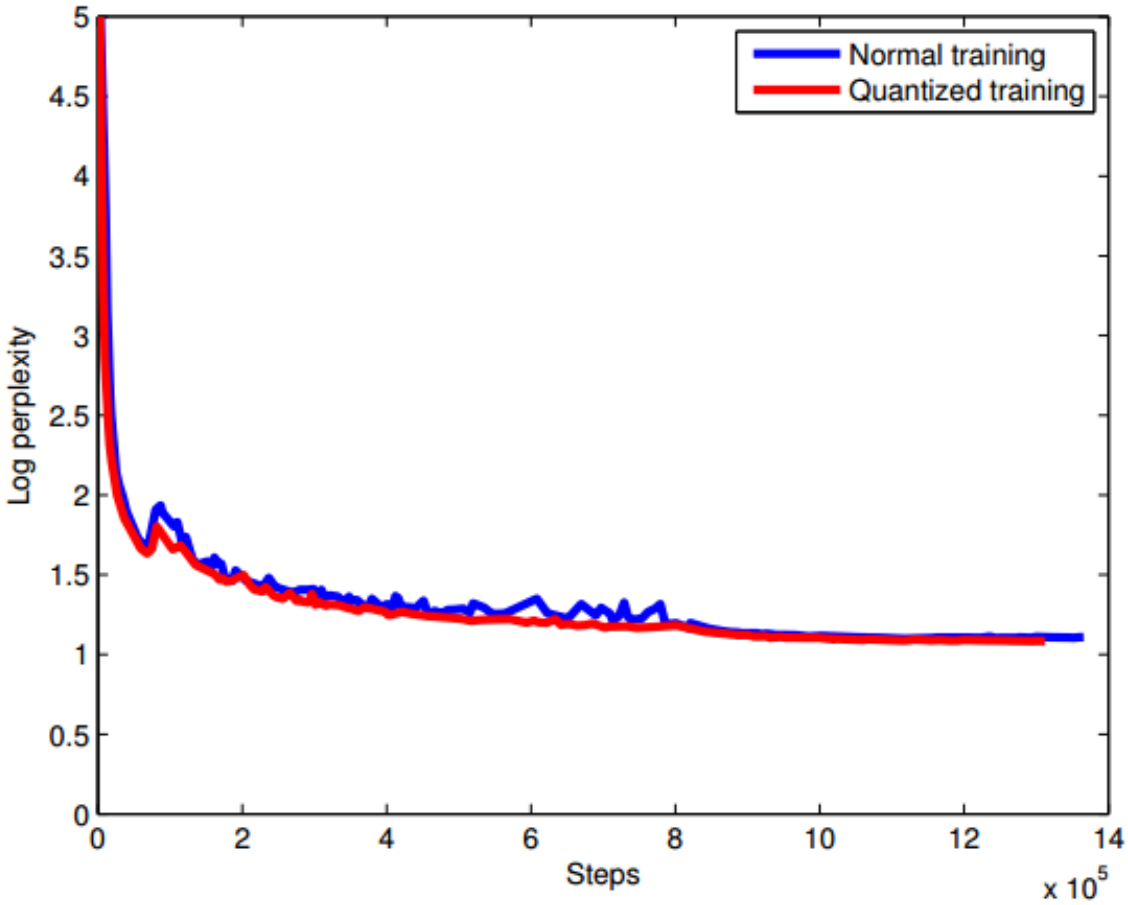


Figure: Credits: Arxiv GNMT paper

Quantization

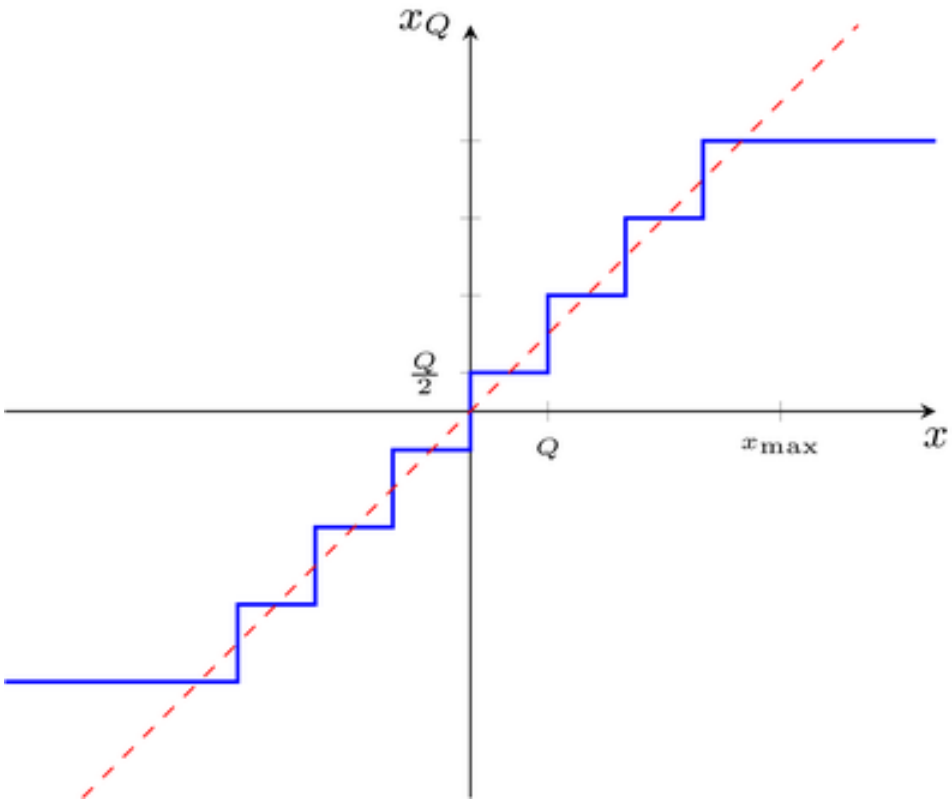


Figure: Credits: Sascha Spors

Speed up

	BLEU	Log Perplexity	Decoding time (s)
CPU	31.20	1.4553	1322
GPU	31.20	1.4553	3028
TPU	31.21	1.4626	384

Figure: Credits: Arxiv GNMT paper

Micro services

Micro services

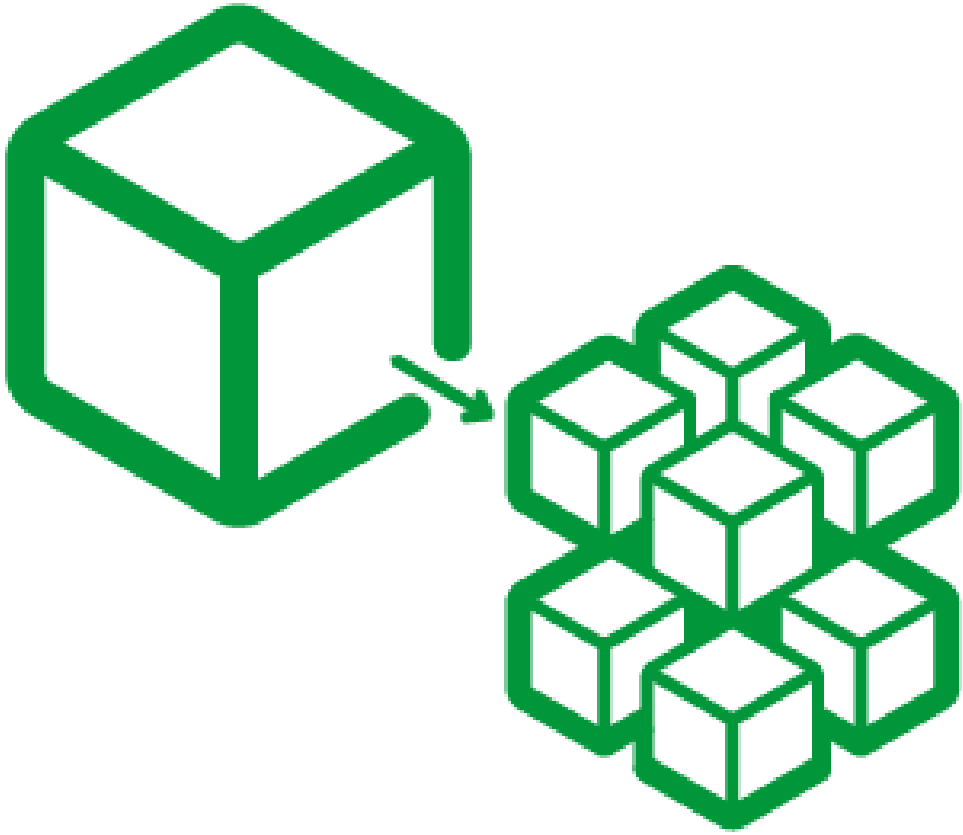


Figure: Source: nginx.com

Post processing

1. Plot low confidence / high uncertainty
2. Plot confusion matrix
3. Inspect neurons

Regularization

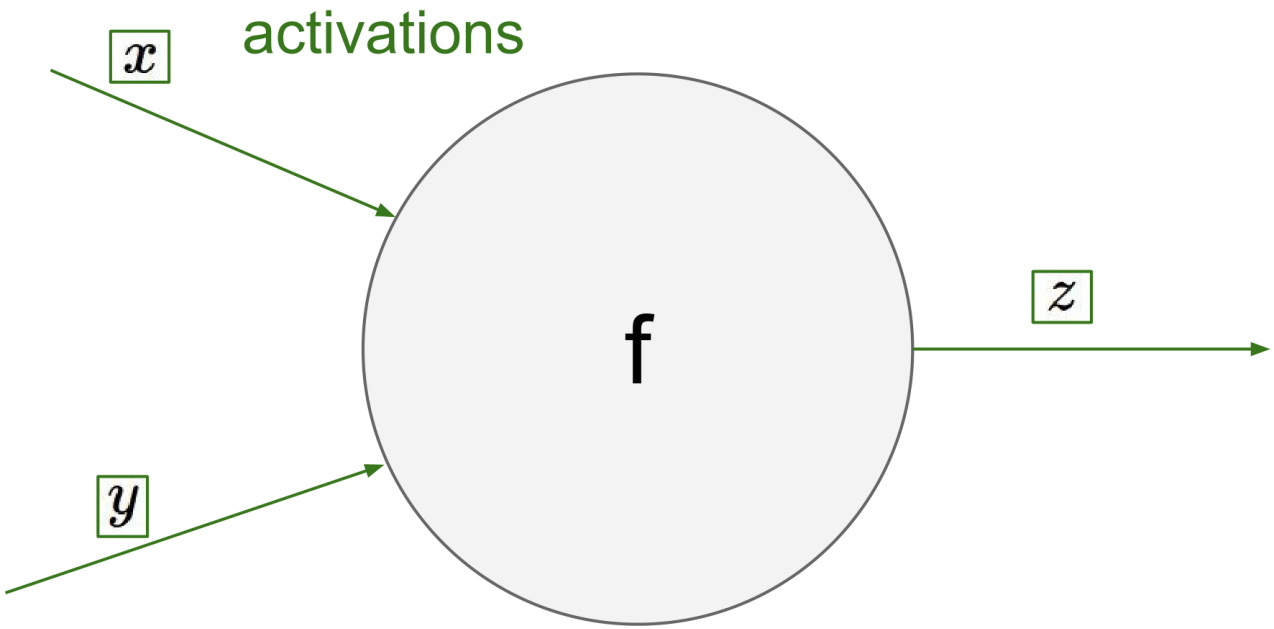
Figure: Example for regularization

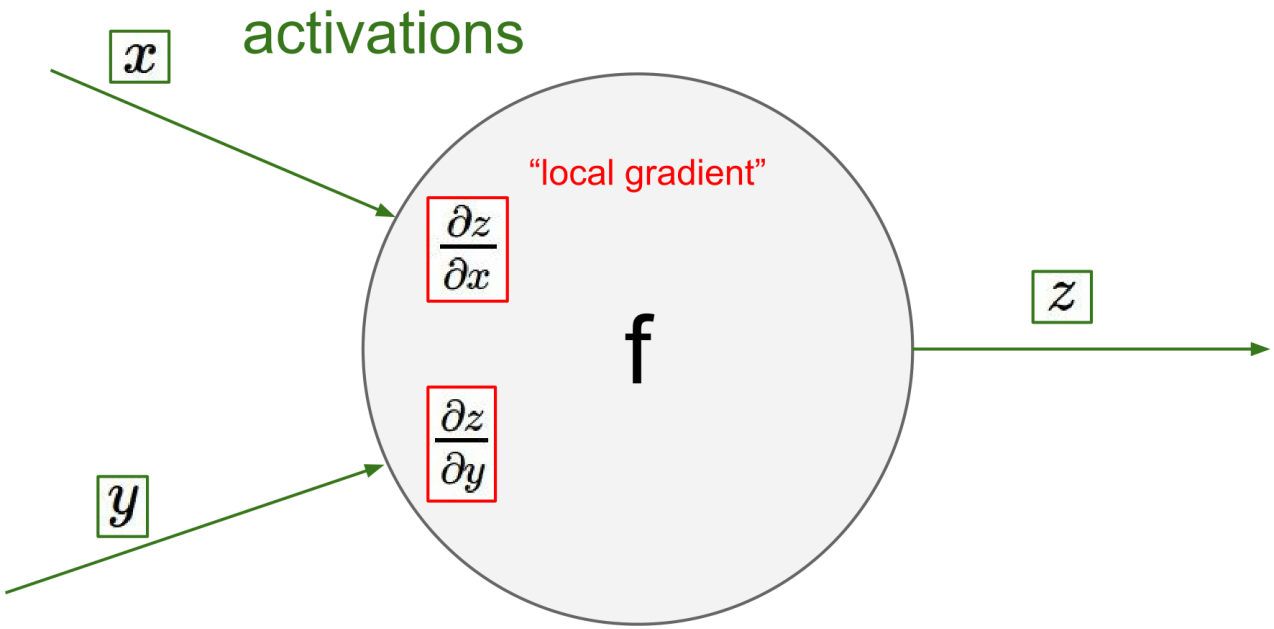
Wrap up

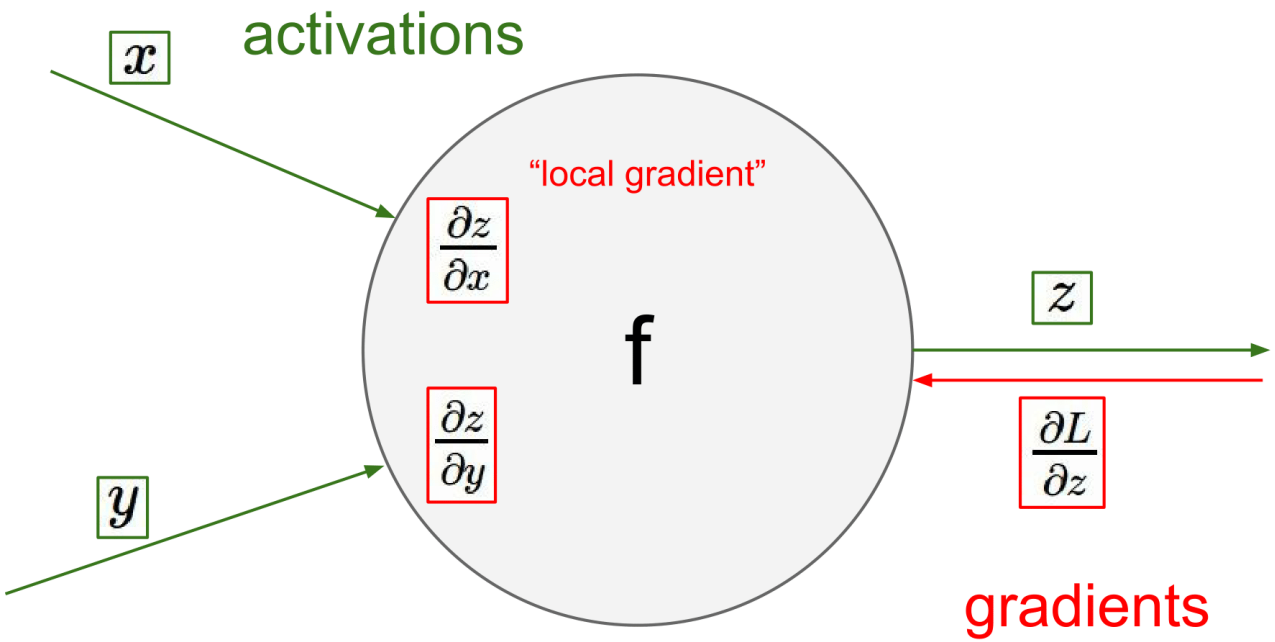
Questions

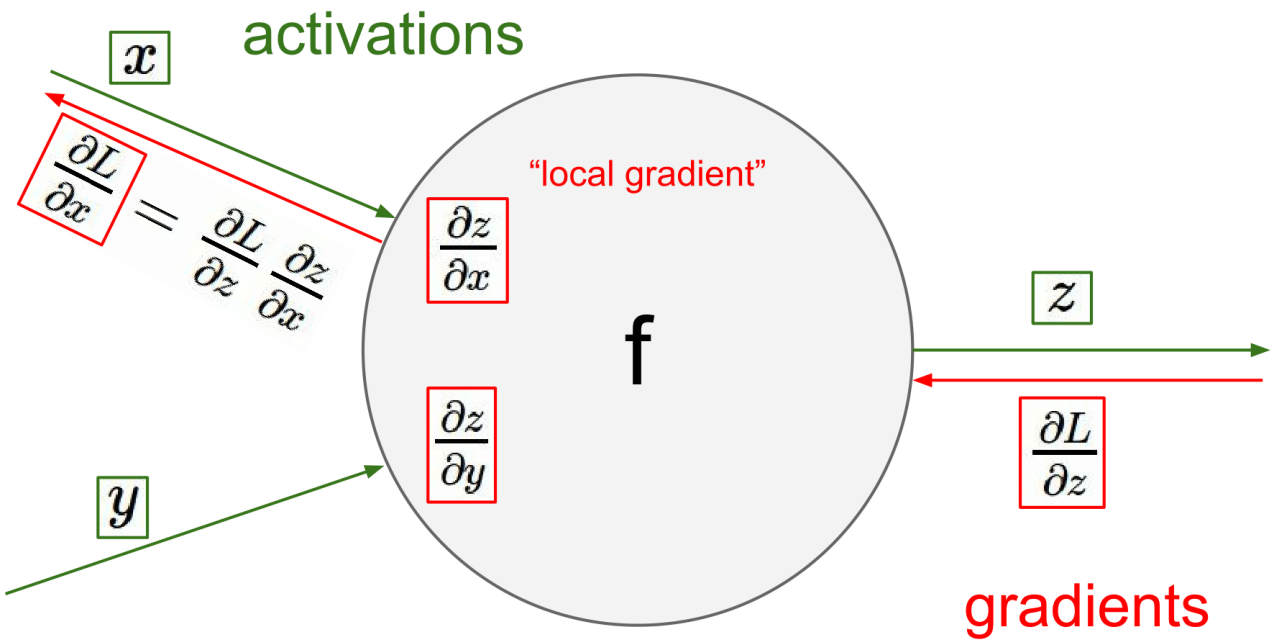
Further learning

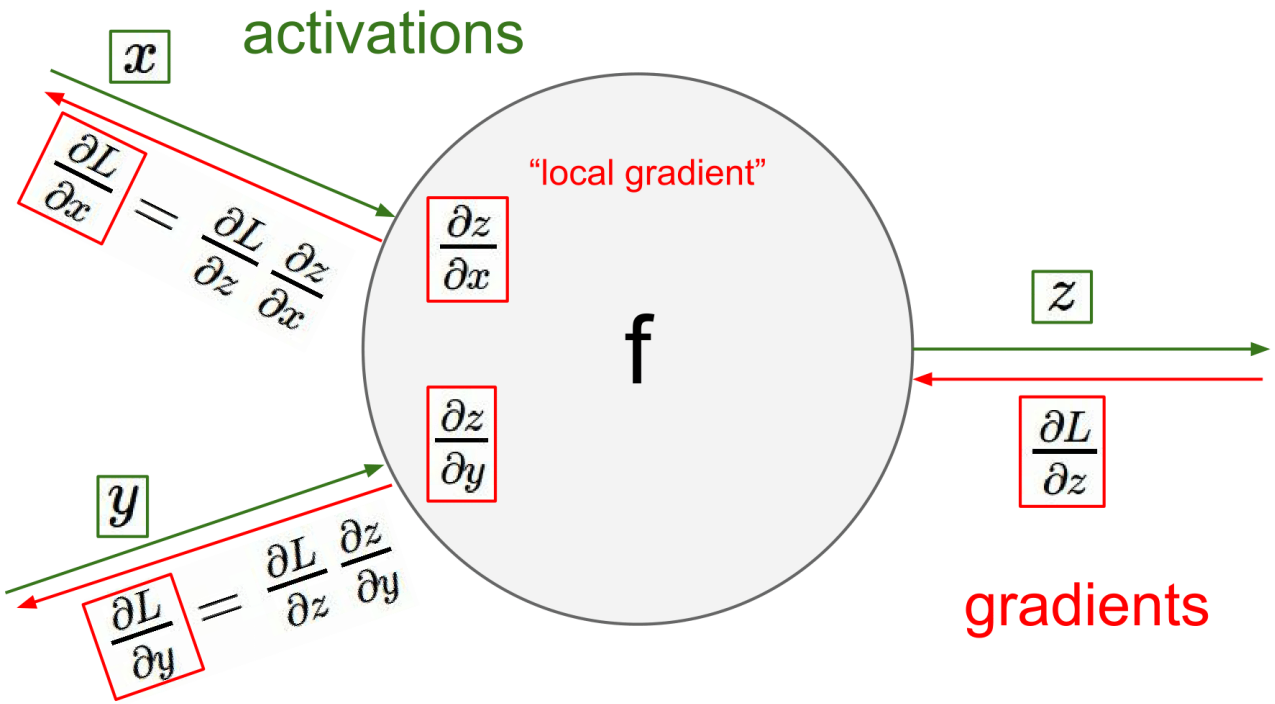
1. Blogs
    1. The Unreasonable Effectiveness of Recurrent Neural Networks, Andrej Karpathy
    2. Chris Olah's blog colah.github.io
2. Courses
    1. Course on CNN: cs231n
    2. Udacity on Deep learning
    3. Fast.ai: Deep learning for Coders
3. My page (robromijnders.github.io)

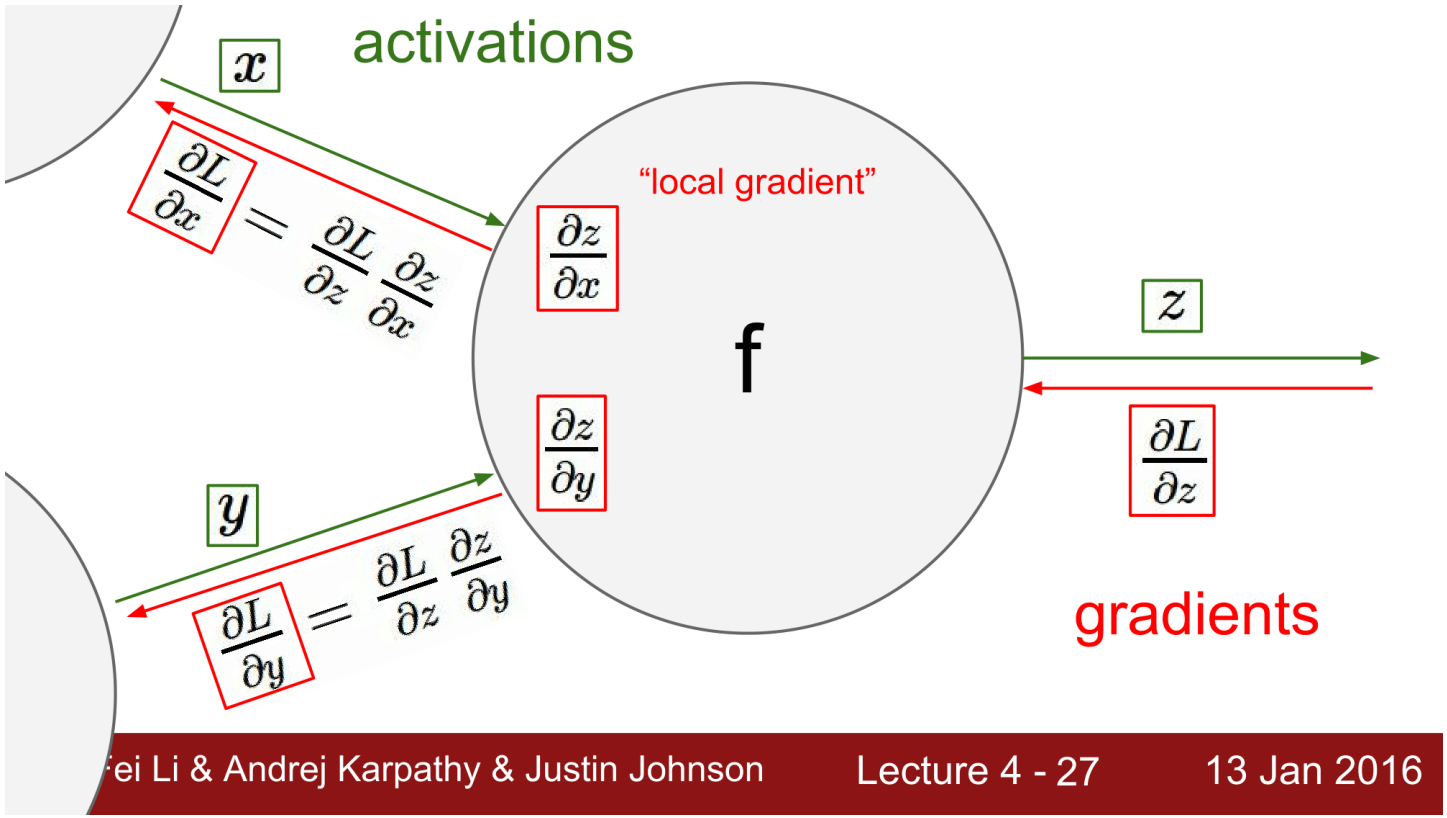

activations
x
y
f
z


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


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
"local gradient"
f
z
gradients
y


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
ei Li & Andrej Karpathy & Justin Johnson
Lecture 4 - 27
13 Jan 2016

Softmax

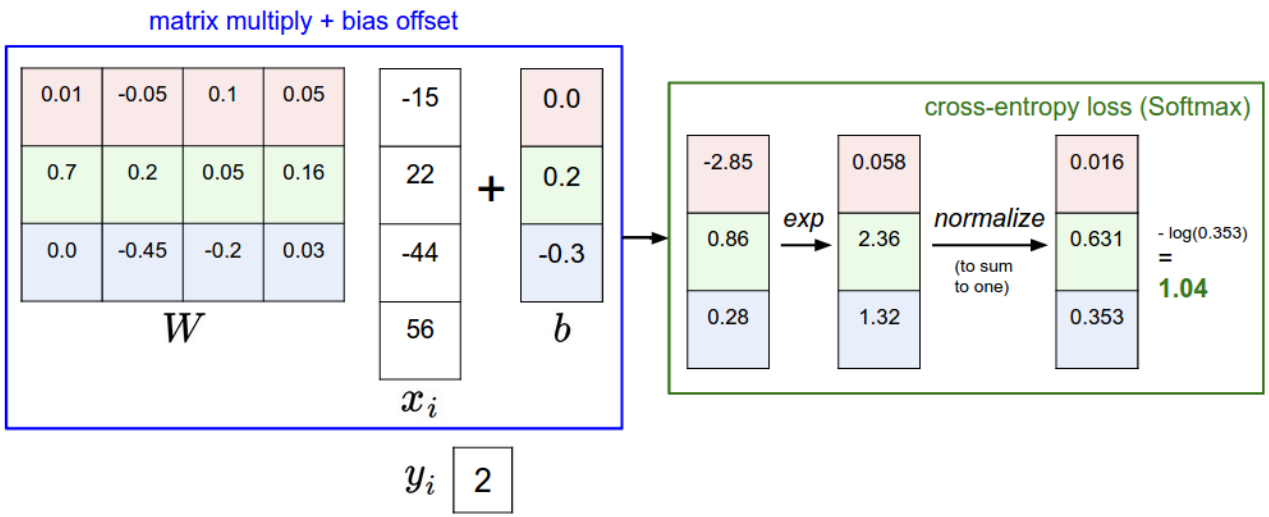


Figure: Adapted from cs231n.github.io/linear-classify/

Example

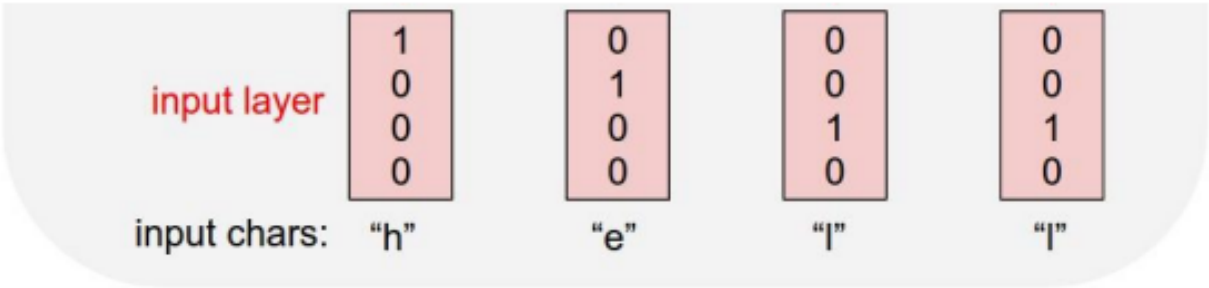


Figure: cs231n, Andrej Karpathy

Example

$$h_t = \tanh(W_{hh} h_{t-1} + W_{xh} x_t)$$

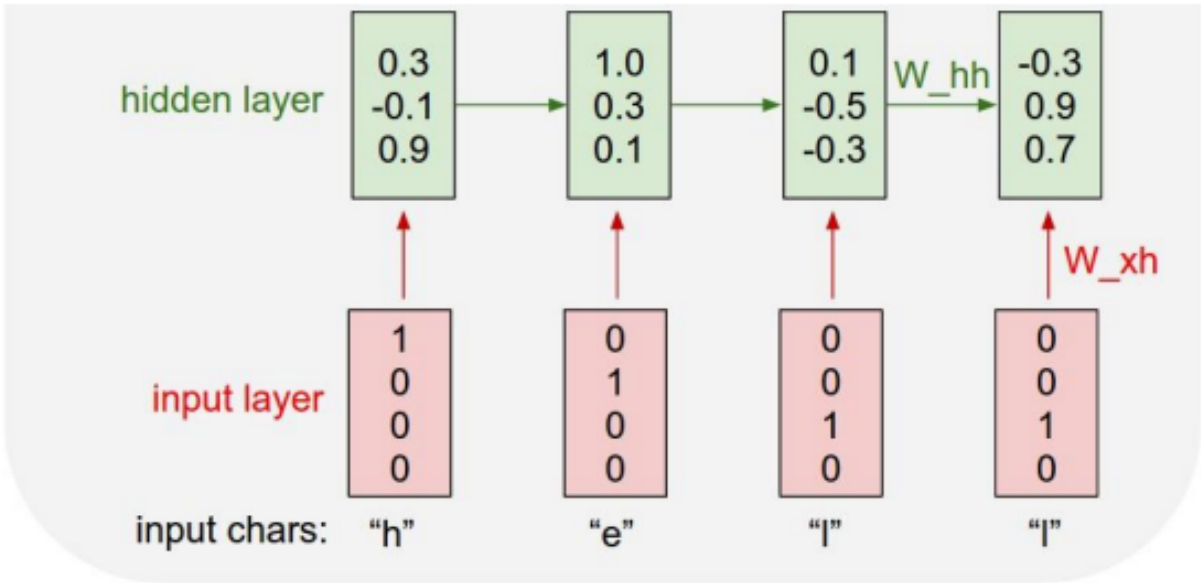


Figure: cs231n, Andrej Karpathy

Example

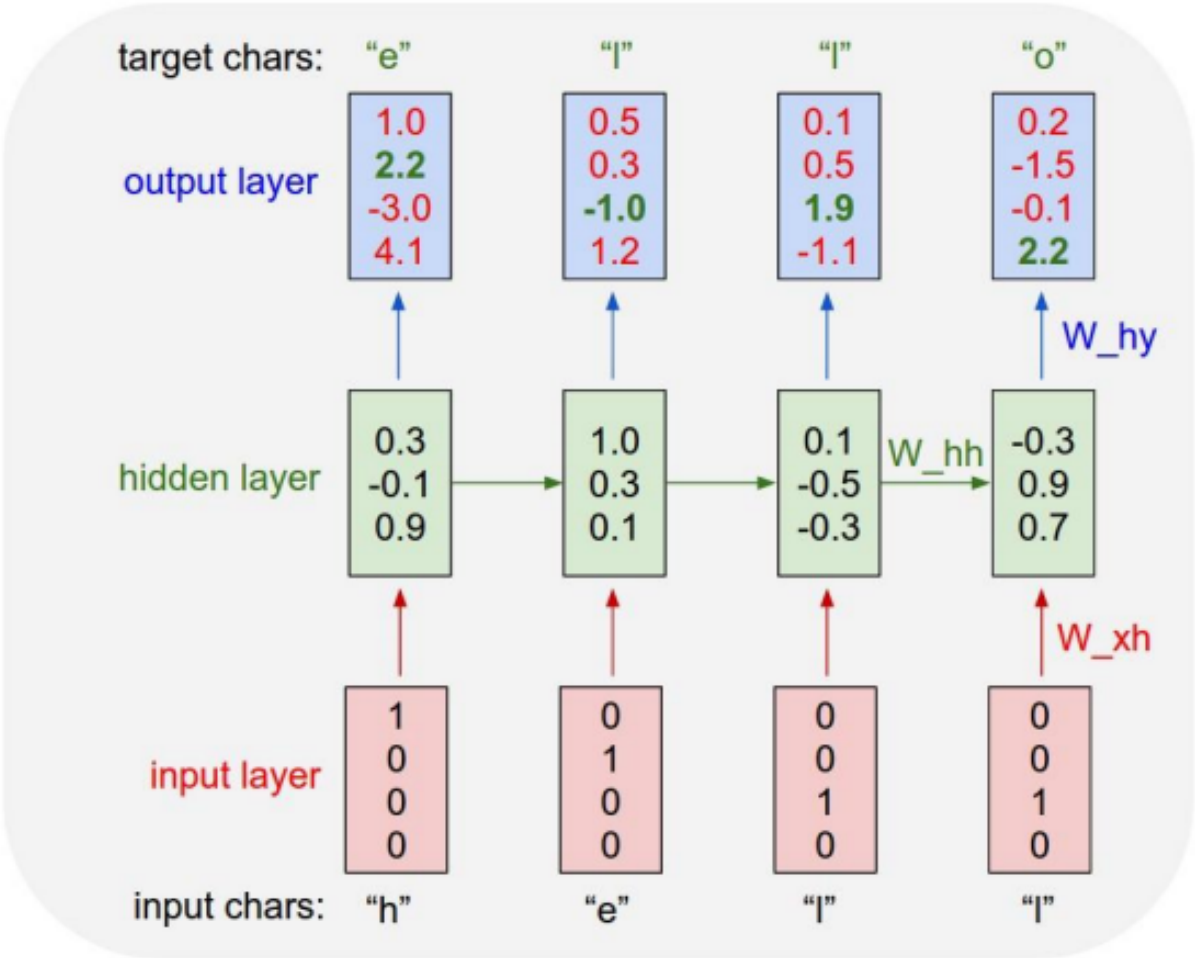


Figure: cs231n, Andrej Karpathy