Deep Learning

TIAS IT Auditing
Data & Analytics specialization

Jori van Lier
March 22, 2019

Agenda

- Deep Learning introduction
- Extending Logistic Regression into Deep Learning
- Notebook practice with Deep Learning (+ 15min break at 10:30-10:45)
- Plenary notebook wrap-up
- Specialized Deep Learning architectures
- Lunch
- Towards ConvNets
- A simple ConvNet architecture & Transfer Learning
- Notebook practice
- Plenary notebook wrap-up
- Break
- Advanced ConvNets: localization, object detection, instance segmentation
- A ConvNet application at Schiphol Airport
- End

Deep Learning

- Deep Learning is **Machine Learning** with a specific model called an **Artificial Neural Network** (NN).

- These NNs have **many hidden layers**, which makes them “deep”.

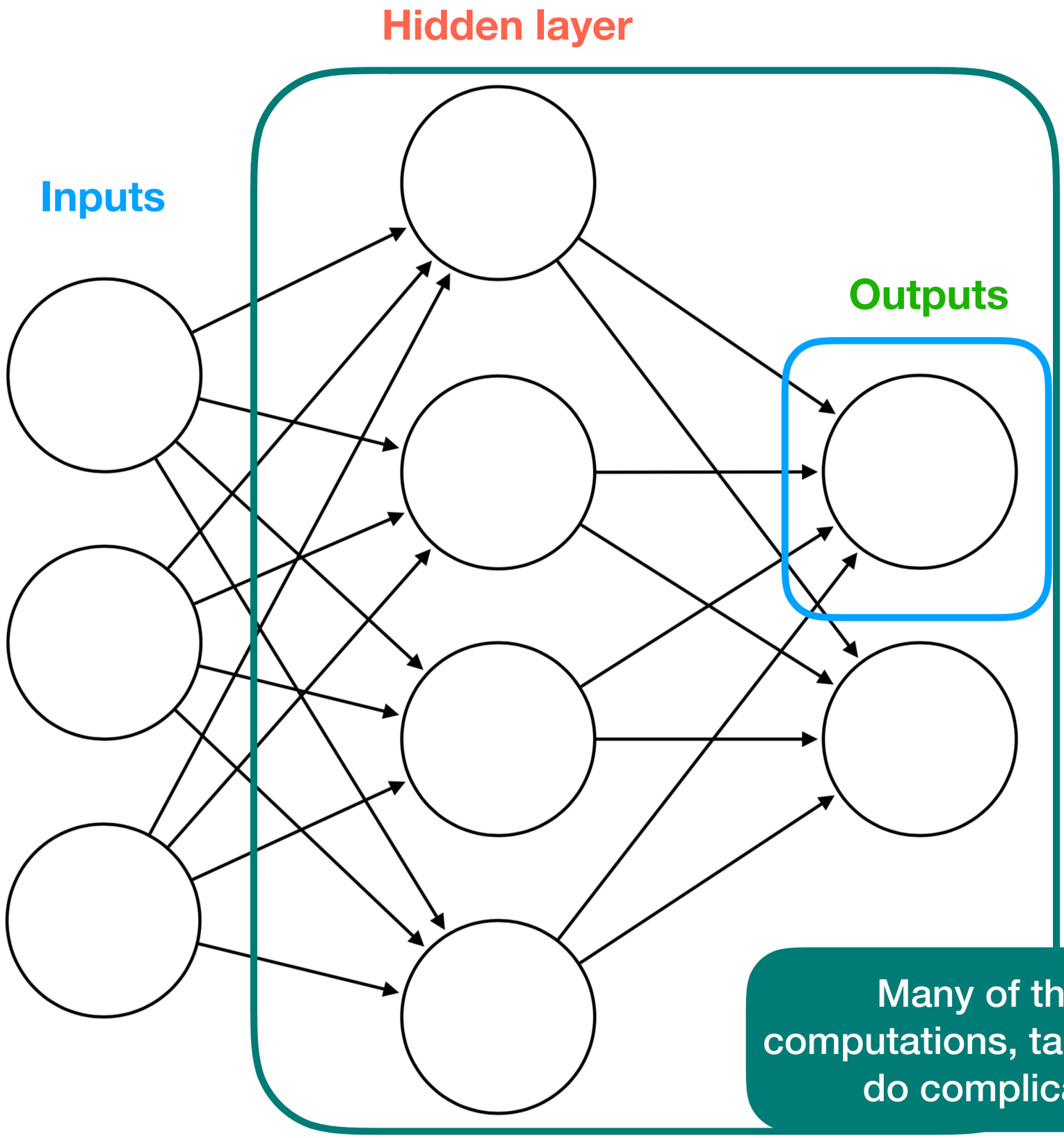
Artificial Neural Networks

- Artificial Neural Networks are a natural extension of Logistic Regression, which we've seen last week.

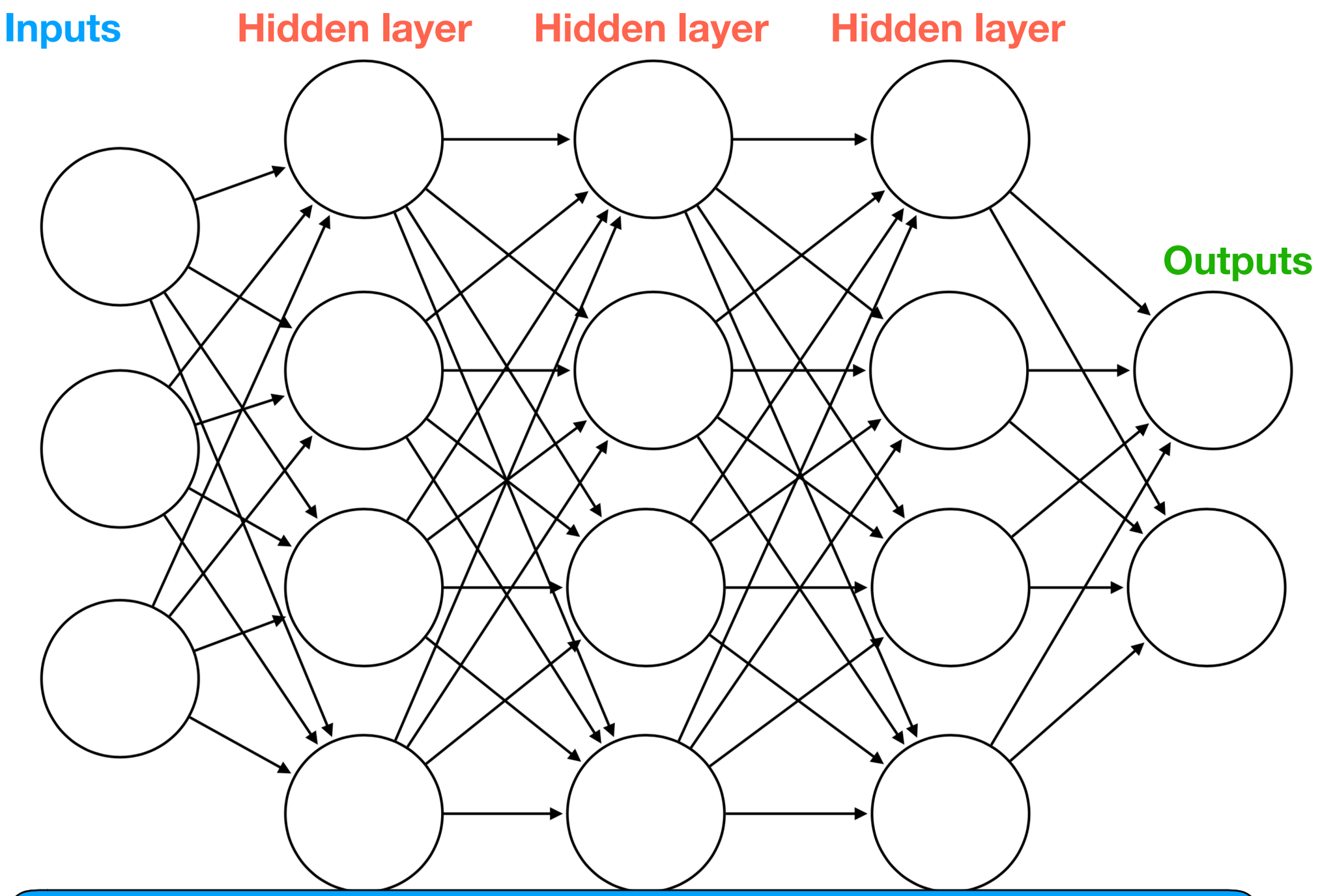
- So they can be used for classification problems, but also regression problems if we take away the sigmoid function.

- The conceptual model is very loosely based on the human brain.

A simple artificial neural network



“Deep” Learning



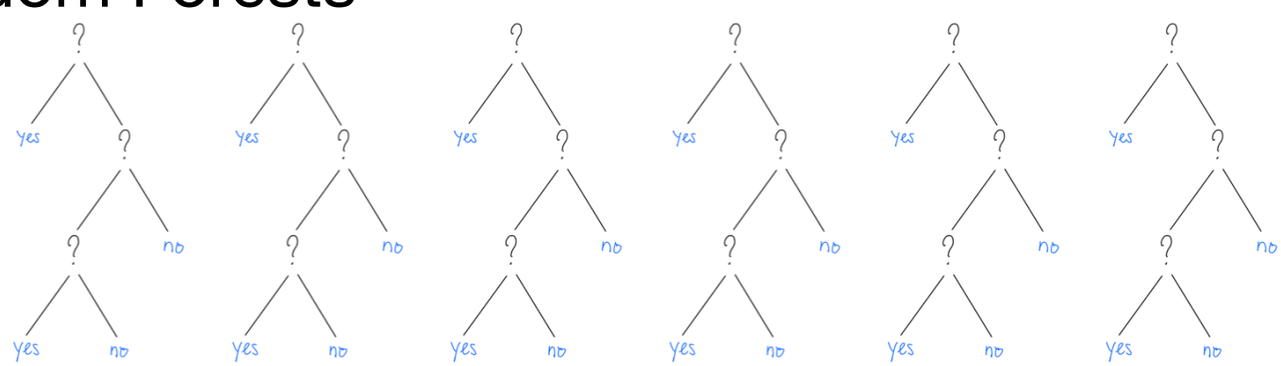
Add a couple of hidden layers, and suddenly an Artificial Neural Neural Network falls into the category “Deep Learning”.

The basic idea

- Brain contains neurons
- Sensory neurons respond to stumili, e.g. light (= Input)
- Motor neurons control the body (= Output)
- Interneurons connect neurons to other neurons in a network (= Hidden)
- Each neuron performs a simple task
- But, 10^{11} neurons working together can do incredible things

A bit of history

- The foundation for NNs already existed in the 80's.
- They performed very well on some difficult tasks, most notably handwriting recognition.
- But they required a lot of computation + manual tuning to work well.
- In the 90's, new techniques started gaining popularity because they worked (almost) just as well on many problems with much less effort:
  - Support Vector Machines
  - Tree-based methods, e.g. Random Forests
- NNs started falling out of favor



Resurgence

- In the 00’s and early 10’s, rapid advancements in computation power sparked a new interest in NNs.
- Most notably Graphical Processing Units (GPUs), which were initially developed for games, found a new use: very efficient parallel matrix computations.
    - Neural Networks involve pretty much only matrix computations!
    - Speedup compared to CPU is roughly 10x.
- A new generation of researchers started looking into them, which resulted in many advancements in network architectures and training techniques.
- Now, they’re everywhere.

Agenda

- Deep Learning introduction
- Extending Logistic Regression into Deep Learning
- Notebook practice with Deep Learning (+ 15min break at 10:30-10:45)
- Plenary notebook wrap-up
- Specialized Deep Learning architectures
- Lunch
- Towards ConvNets
- A simple ConvNet architecture & Transfer Learning
- Notebook practice
- Plenary notebook wrap-up
- Break
- Advanced ConvNets: localization, object detection, instance segmentation
- A ConvNet application at Schiphol Airport
- End

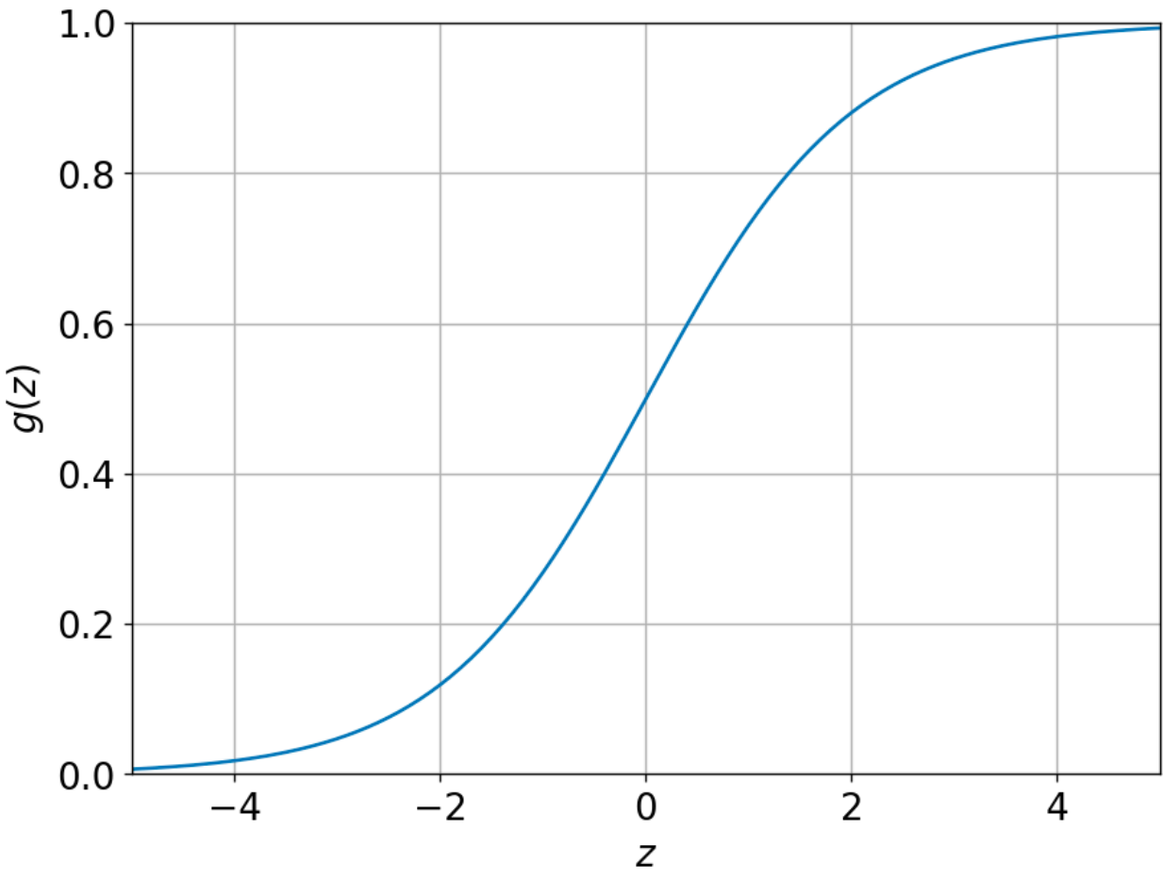
Let's revisit logistic regression

$$y = \frac{1}{1 + e^{-\theta \mathbf{x}}}$$

$$\theta \mathbf{x} = \sum_{i=0}^{n} \theta_n x_n$$

Convention: $x_0 = 1$, so
θ_0 only gets added (intercept)

The Sigmoid function used is used to squeeze the output between 0 and 1:



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

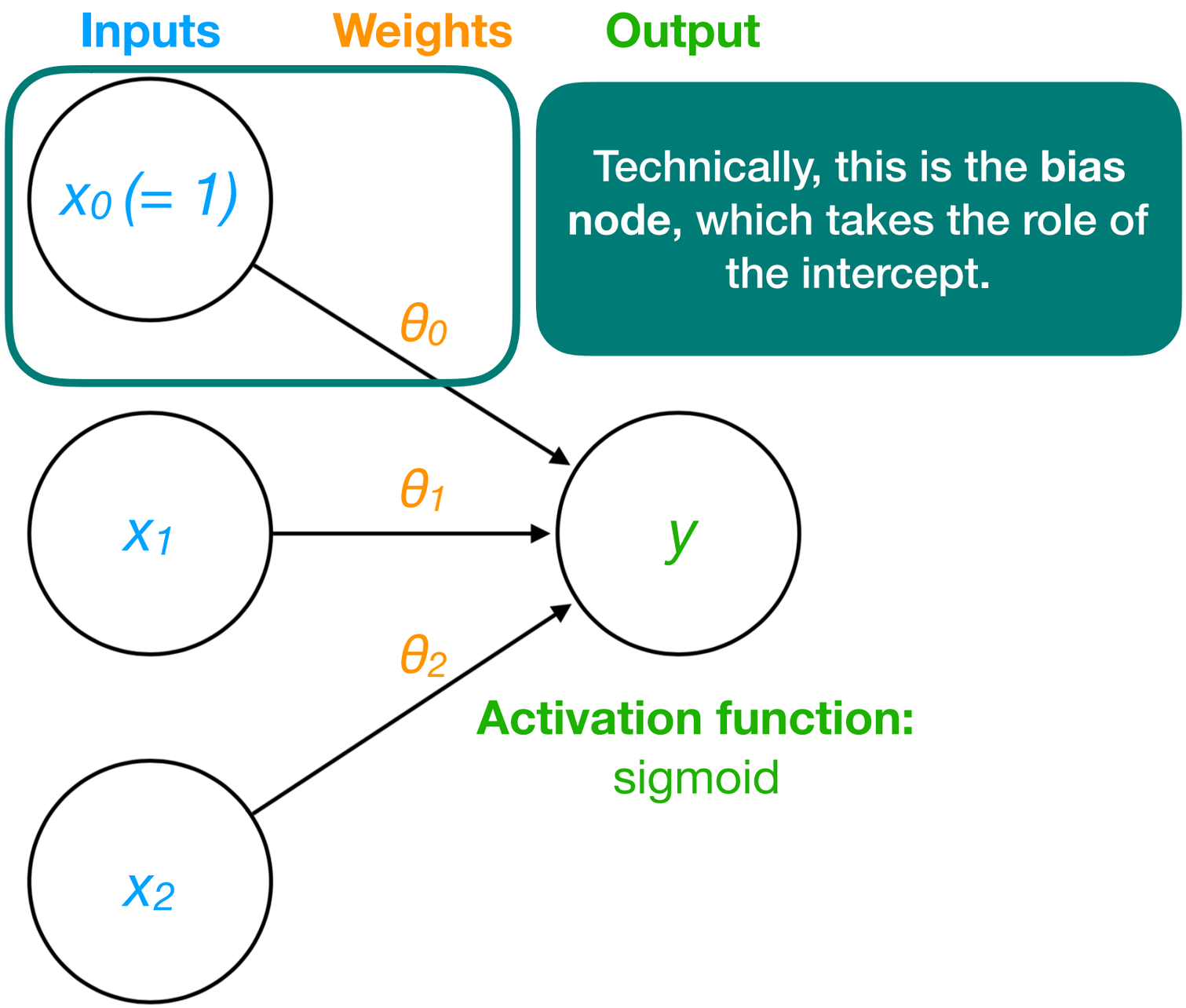
And formulate it a bit differently…

Logistic Regression

$$y = \frac{1}{1 + e^{-\theta \mathbf{x}}}$$

$$\theta \mathbf{x} = \sum_{i=0}^{n} \theta_n x_n$$

Neural Network



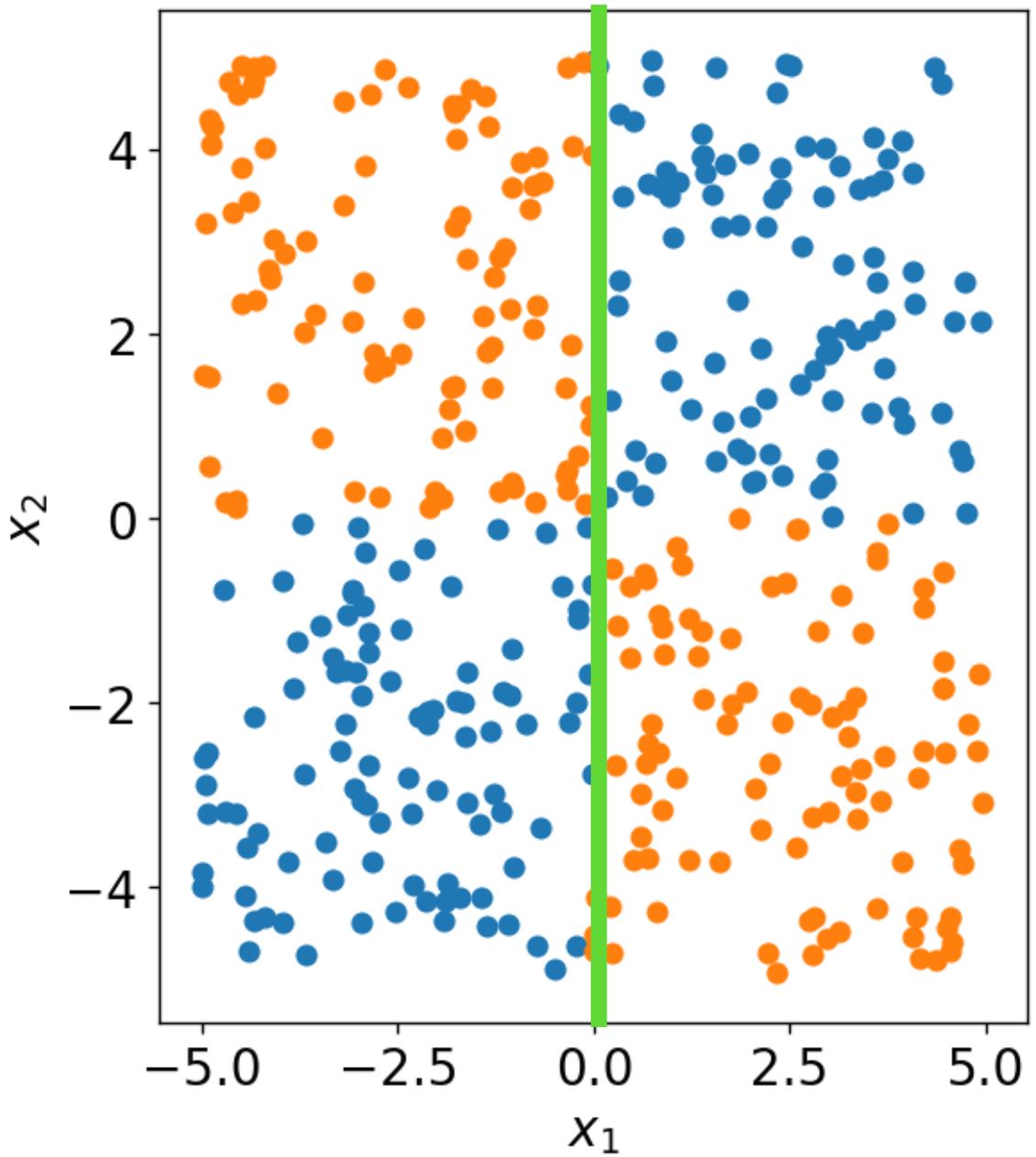
Technically, this is the **bias node**, which takes the role of the intercept.

Activation function:
sigmoid

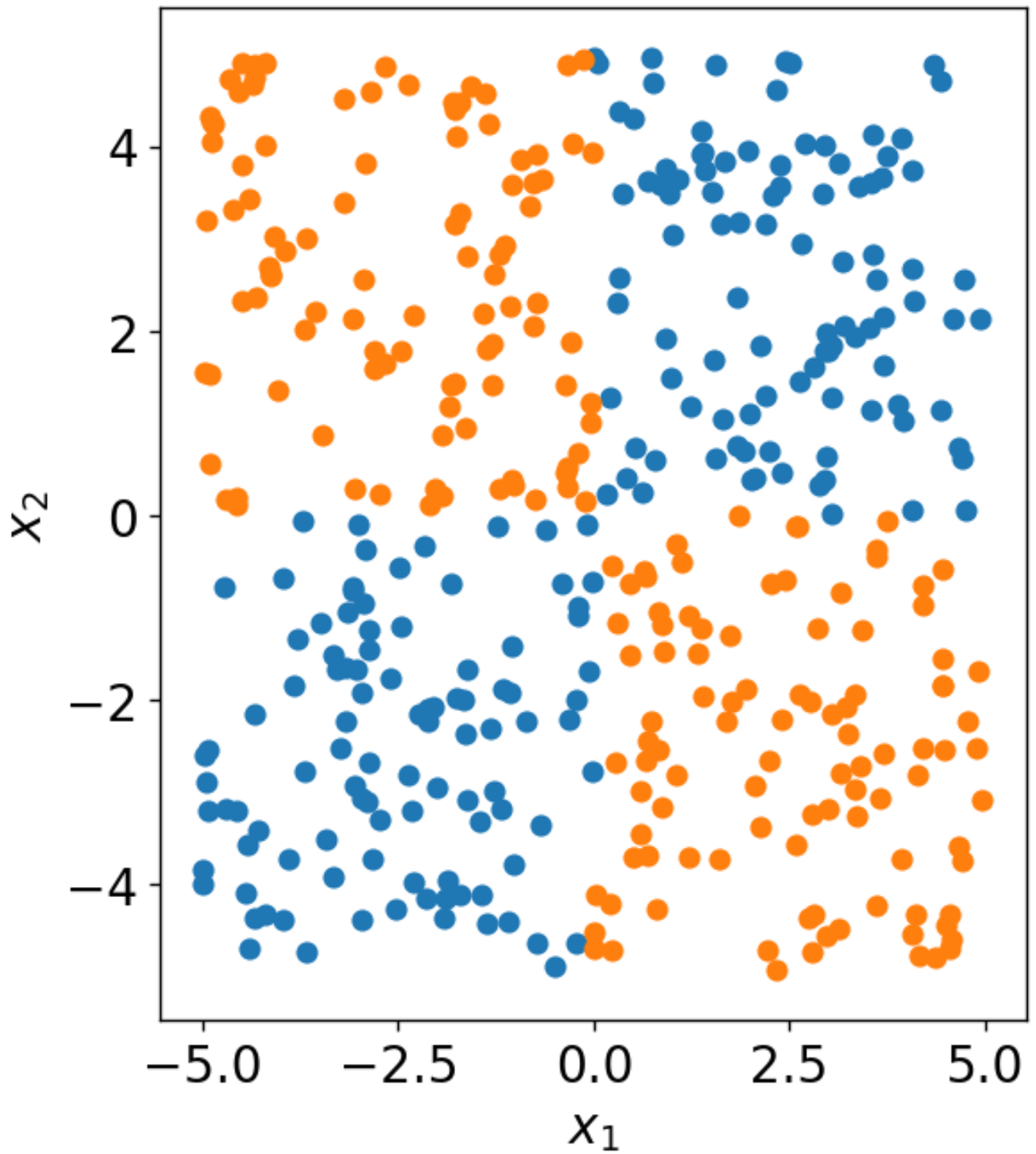
These two models are equivalent!

Why would we want to do this?

- Logistic regression can't solve everything!
- Remember: even though it uses the nonlinear sigmoid function, it is still a linear model and is therefore limited to modeling linear decision boundaries.
- In particular, it cannot learn to classify this dataset:
- This is known as the XOR problem.
- The Neural Network framework allows us to easily add more complexity to the Logistic Regression model.



Anyone know a simple solution?

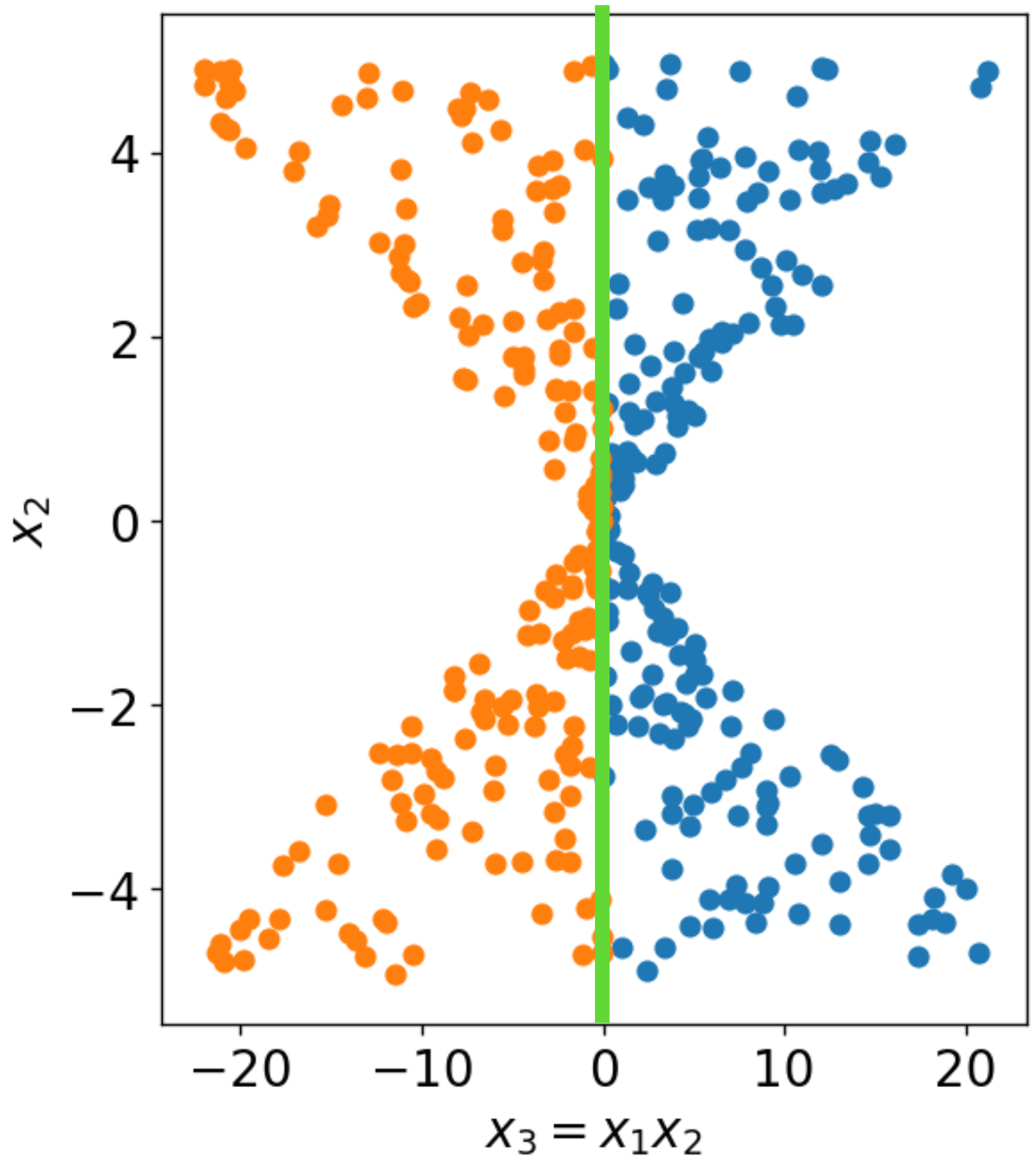


XOR problem: “trick” solution

- This problem can be solved by adding a new feature $x_3 = x_1 x_2$

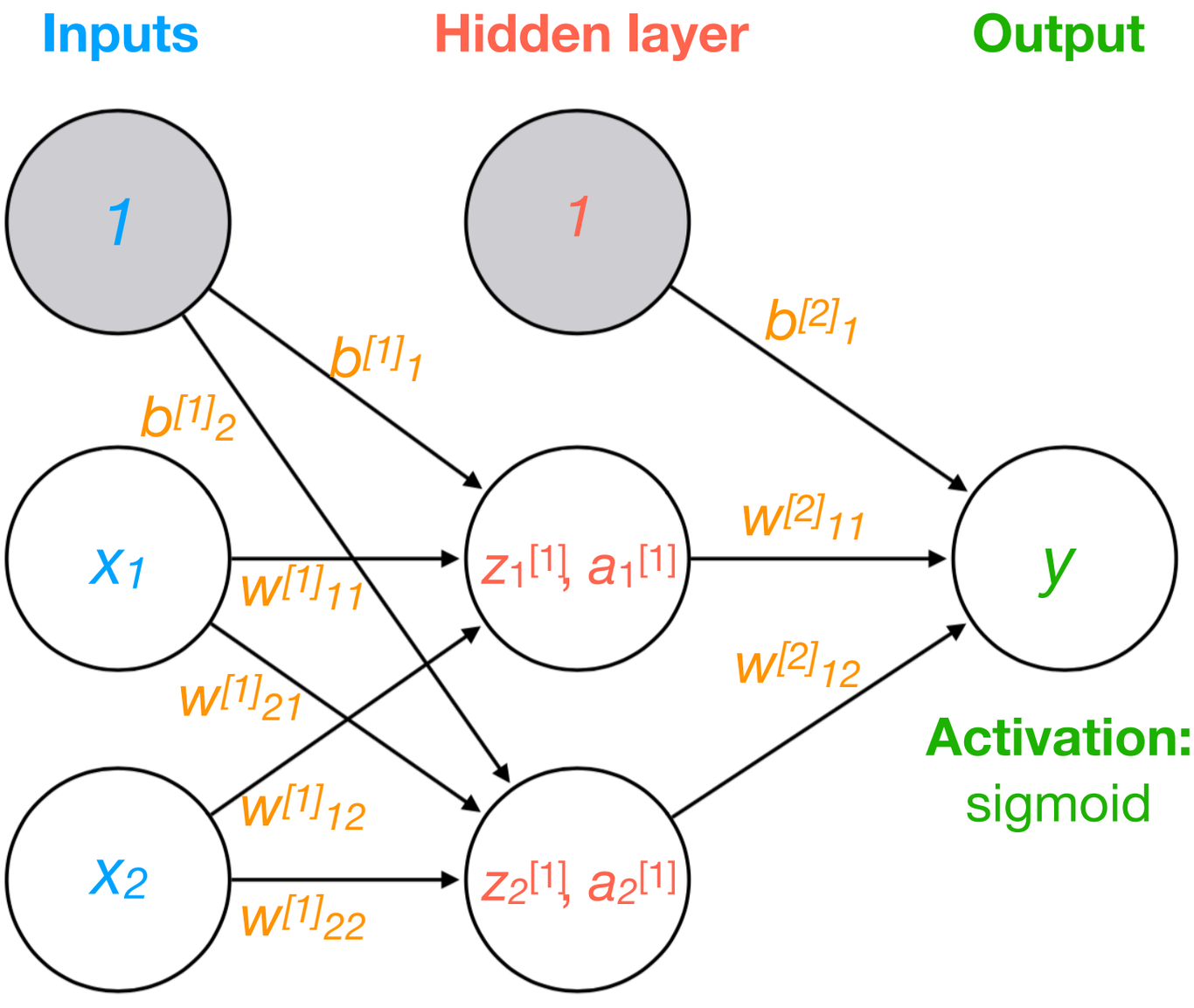
- Getting to this insight is feasible now, but not when there are 100s or 1000s of features!

This is an example of feature engineering!



The neural network solution

- Learn this solution, rather than adding them manually.



Activation:
sigmoid

$$z_1^{[1]} = x_1 w_{11}^{[1]} + x_2 w_{12}^{[1]} + b_1^{[1]}$$

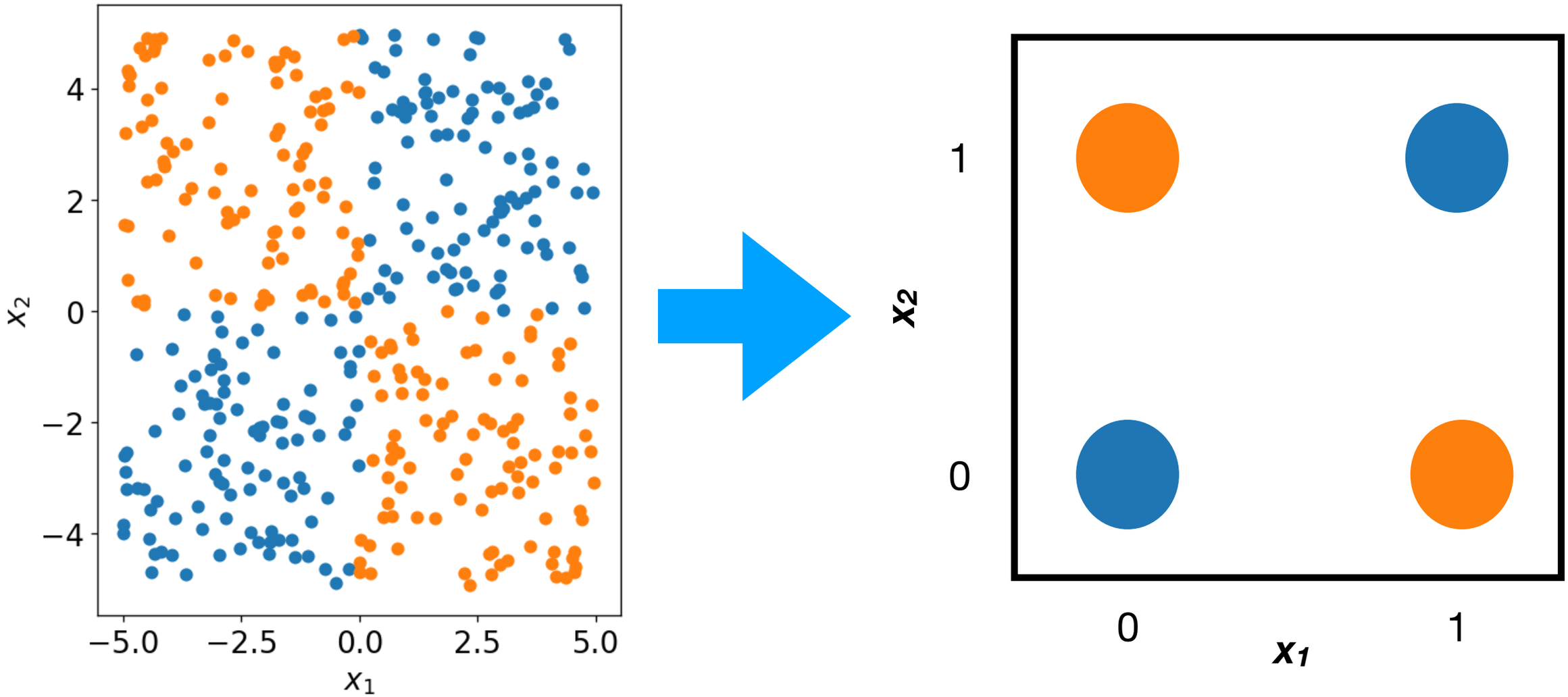
$$a_1^{[1]} = g\left(z_1^{[1]}\right)$$ *g(z)* is an activation function (e.g. sigmoid)

$$y = g\left(a_1^{[1]} w_{11}^{[2]} + a_2^{[1]} w_{12}^{[2]} + b_1^{[2]}\right)$$

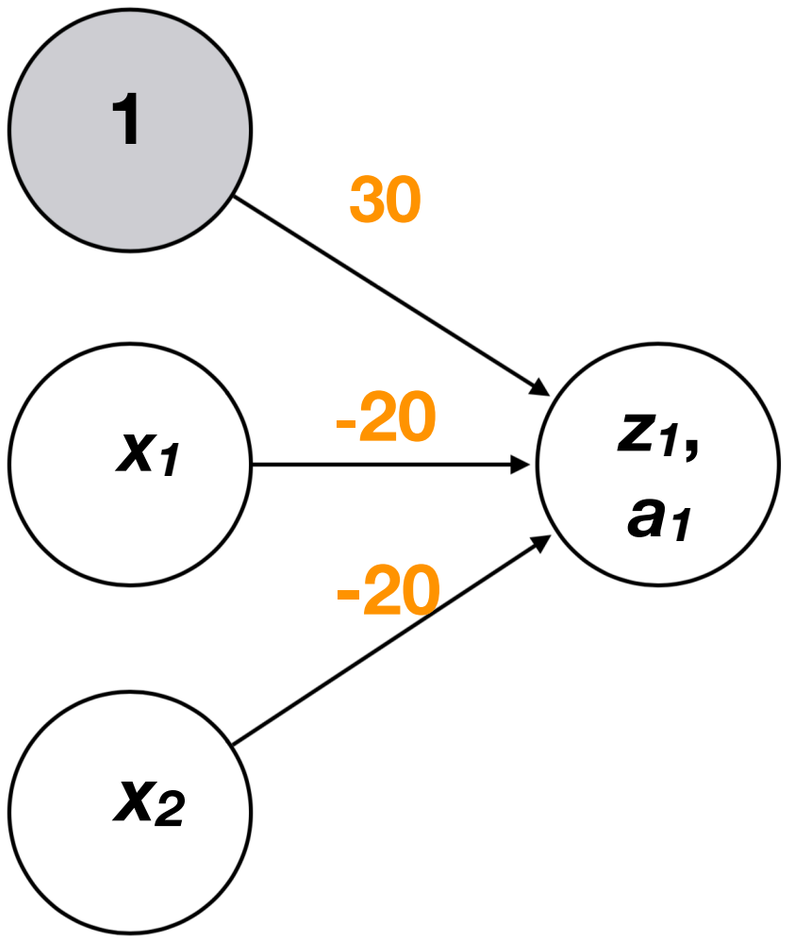
A lot of math, but at no point are x_1 and x_2 multiplied! So how does it solve XOR?

Let’s binarize the problem

This makes it easier to reason about…



For this example only: 0-1 stepwise activation function



$$z_1 = 30 - 20x_1 - 20x_2$$

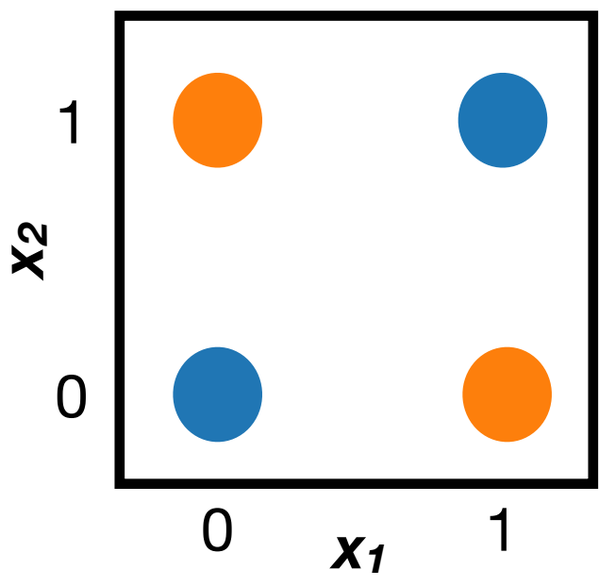
If z_1 is negative, a_1 is 0.
If z_1 is positive, a_1 is 1.

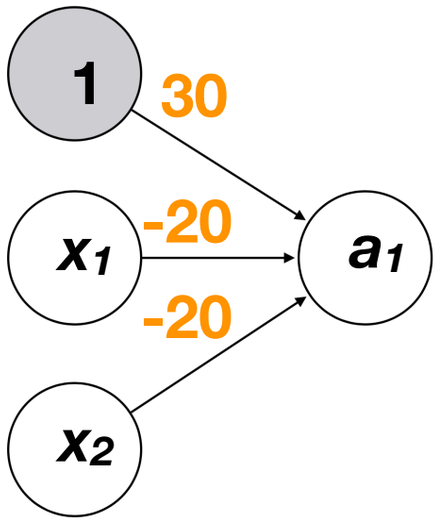
If x_1 and x_2 are 1:
z_1 = -10,
a_1 = 0

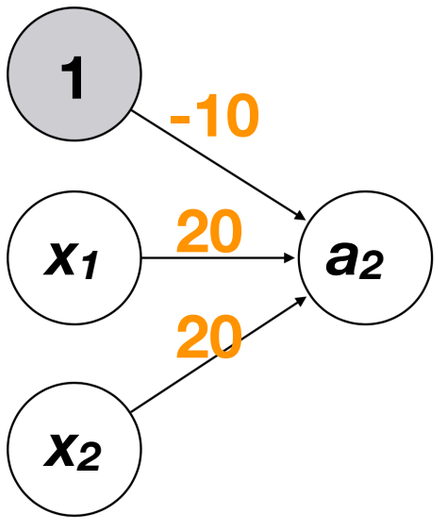
XOR with neural nets

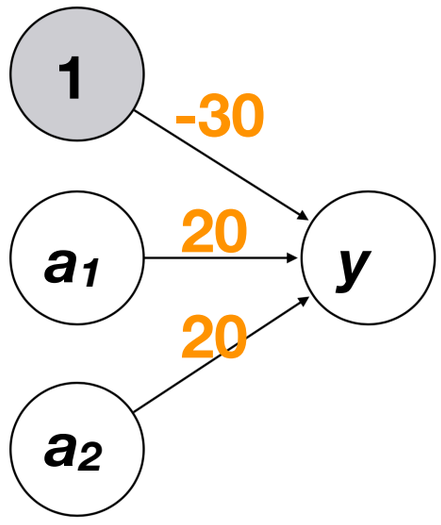
x_1	x_2	desired y
0	0	0
0	1	1
1	0	1
1	1	0

NOT (x_1 AND x_2)	x_1 OR x_2	a_1 AND a_2
1	0	0
1	1	1
1	1	1
0	1	0









So, we get a lot more flexibility! What else can we do?

- Add even more hidden layers: this is where “Machine Learning with Neural Networks” transitions into “Deep Learning”

- Different activation functions

  - We just saw a stepwise binary activation function (this one is not really used in practice though!)

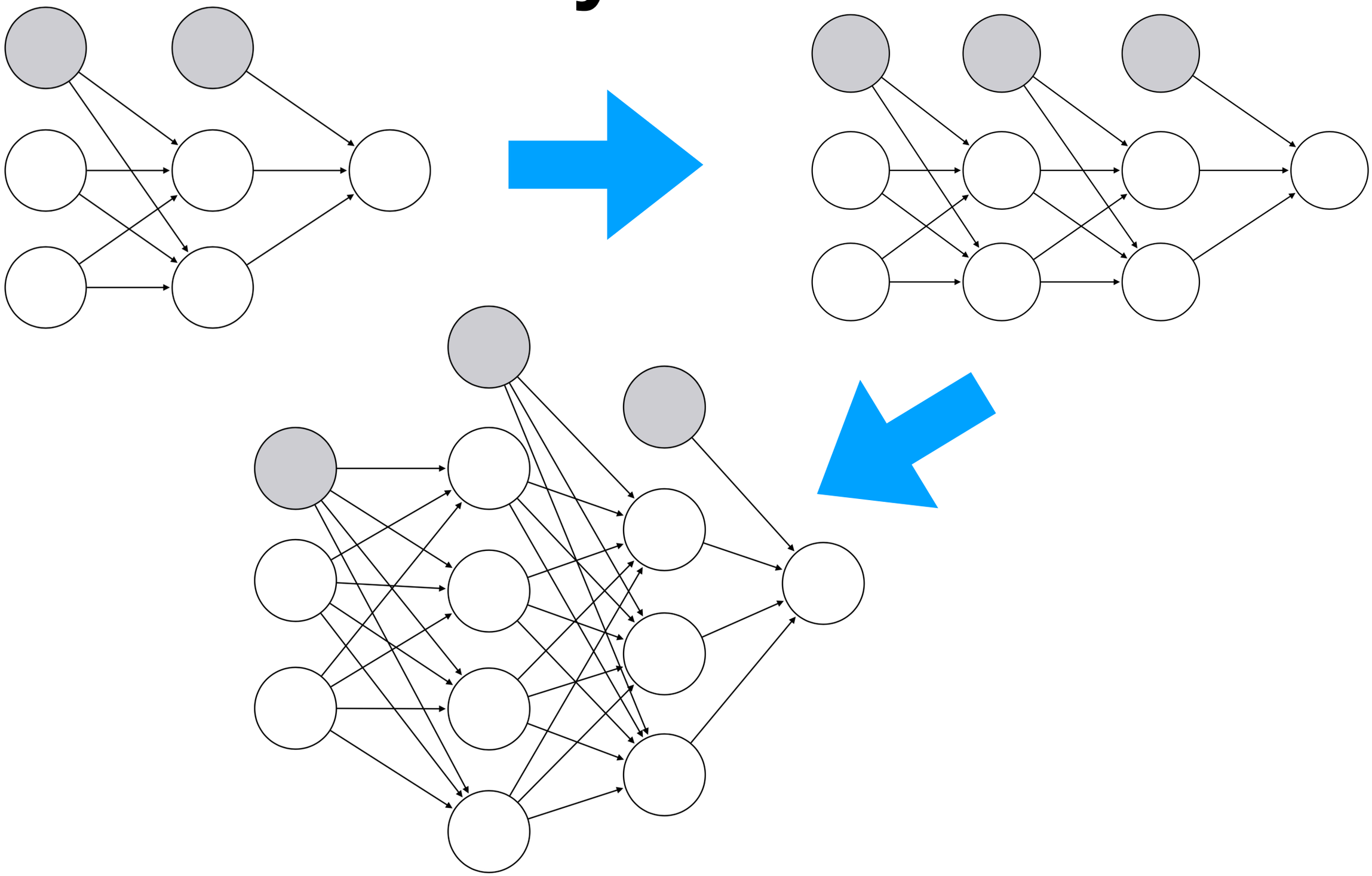
  - We’re not necessarily stuck with sigmoid…

- Multiple outputs

Flexibility: more hidden layers

- More hidden layers allows us to exploit more complicated relationships between features.
- Hyperparameters:
  - # of hidden layers
  - # of hidden units per layer
- Optimize these as you would any other!

Flexibility: more hidden layers

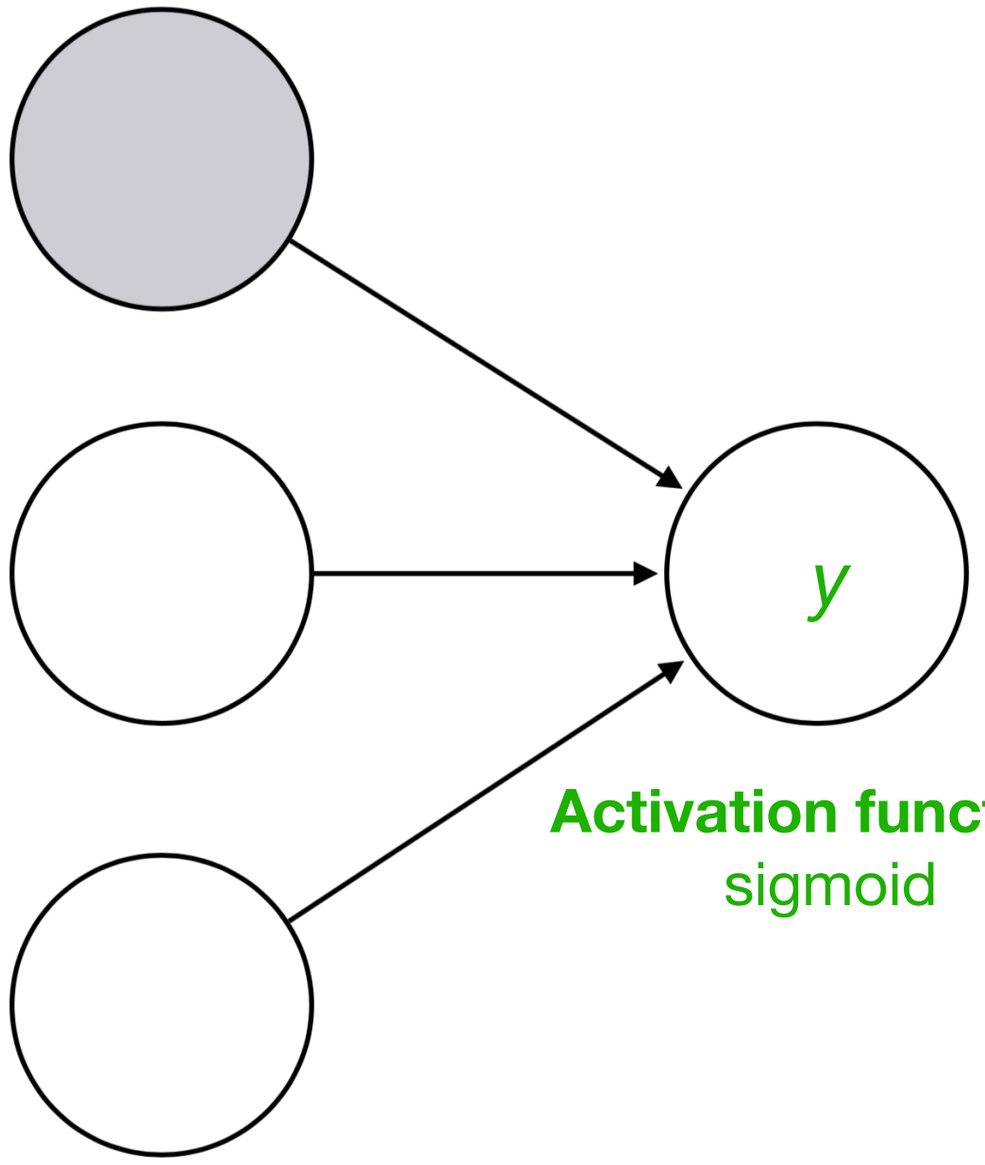


Flexibility: different activation functions

- What happens with this network if you use **no activation function** whatsoever?

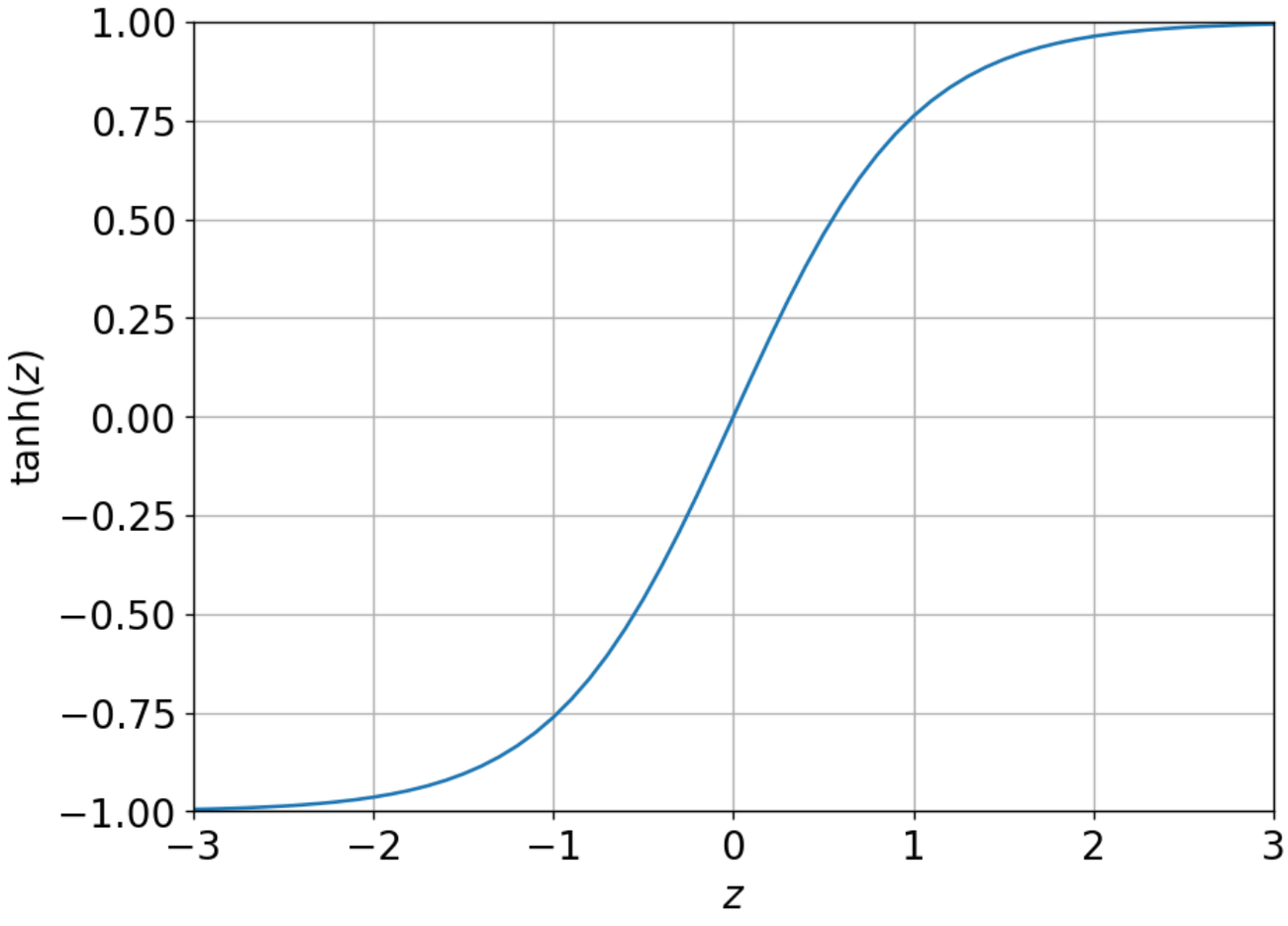
This neural network (without any hidden layers) with no activation function is equivalent to Linear Regression!

Other names:
- Linear activation function
- Identity activation function



tanh activation function

- Like sigmoid, but squeezes values between -1 and 1 instead of 0 and 1:

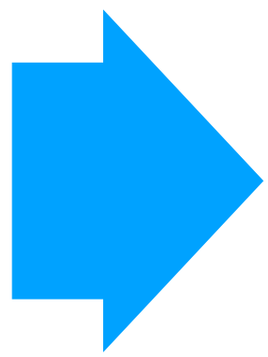


tanh is usually a better choice than sigmoid for the hidden layers, because it does a beter job of normalizing the outputs (mean 0).

Softmax activation function

- Multi-class generalization of sigmoid.
- Used exclusively at the end of the network.
- Sigmoid output: value between 0 and 1.
- Softmax output:

$$\begin{bmatrix} 0.01 \\ 0.95 \\ 0.04 \end{bmatrix}$$



For classification with 3 classes.

Sums up to one!

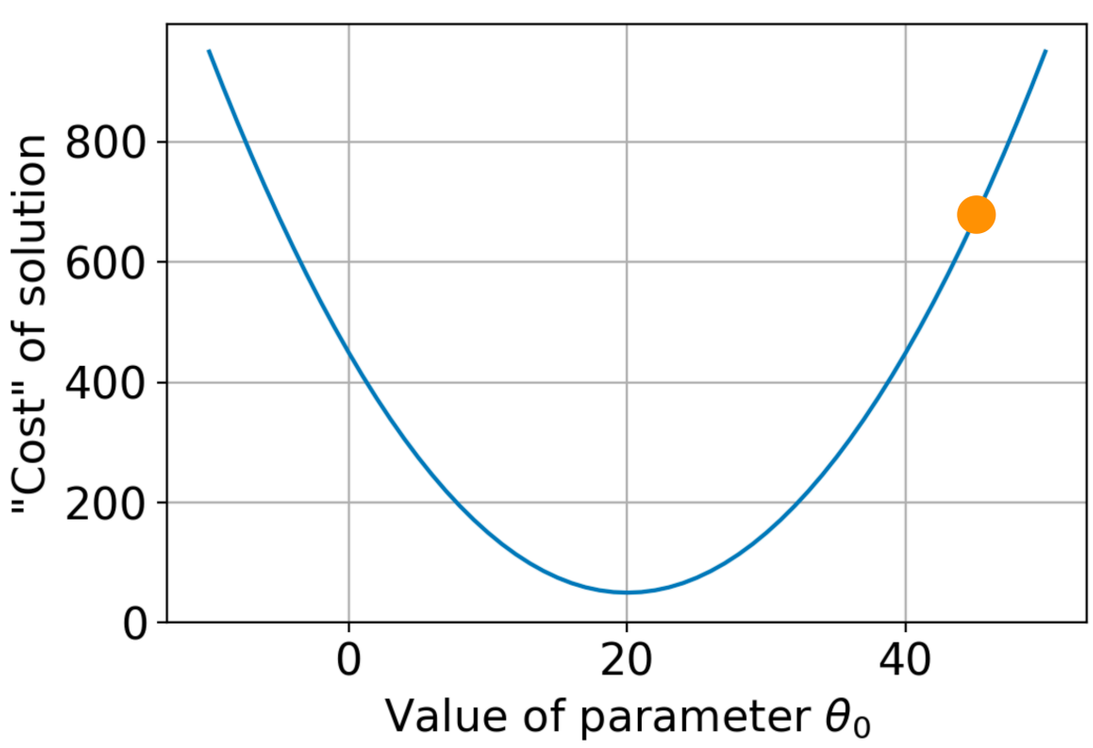
Flexibility: multiple outputs of different types

- Called multi-task-learning
- Example:
    - One output can be linear (“no activation function”)
    - One can be a sigmoid
    - Combination of regression and classification in a single network!
    - Example application: predict house price and also whether it will be sold within 1 month (binary yes/no).
- Can result in improved learning efficiency and prediction accuracy, when compared to training models separately.
- Warning: can also do the opposite!

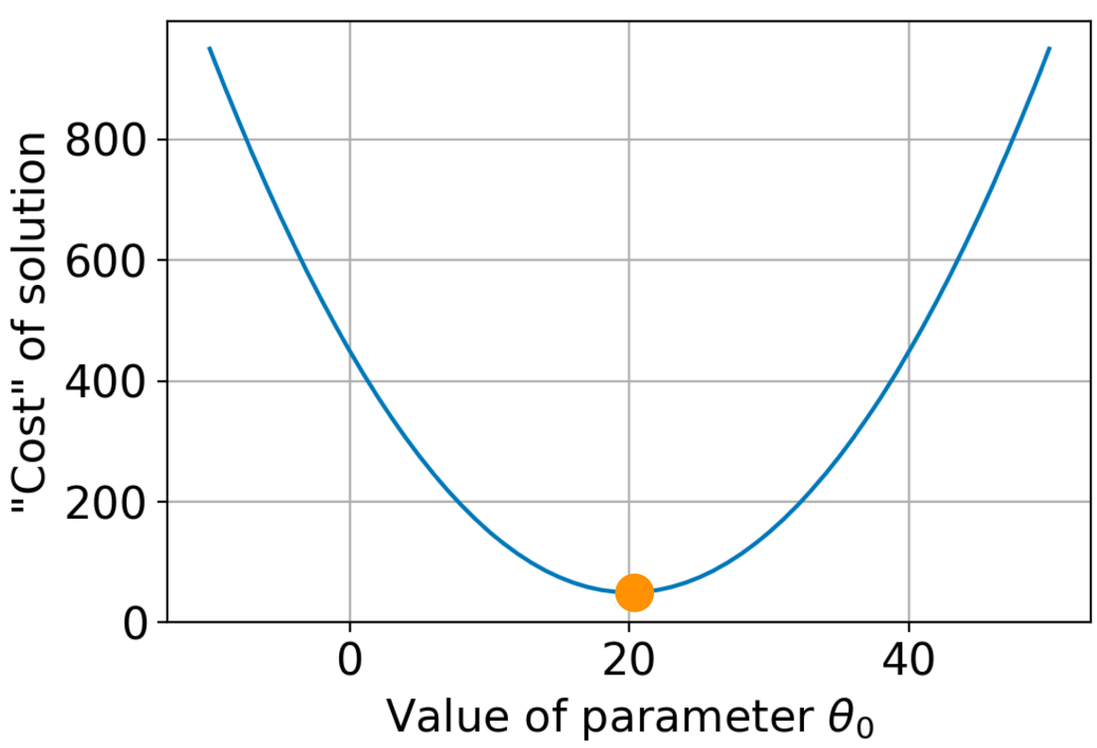
Determining the parameters

- Remember how we did this for Logistic Regression?

Initial guess



Final value after 4 steps

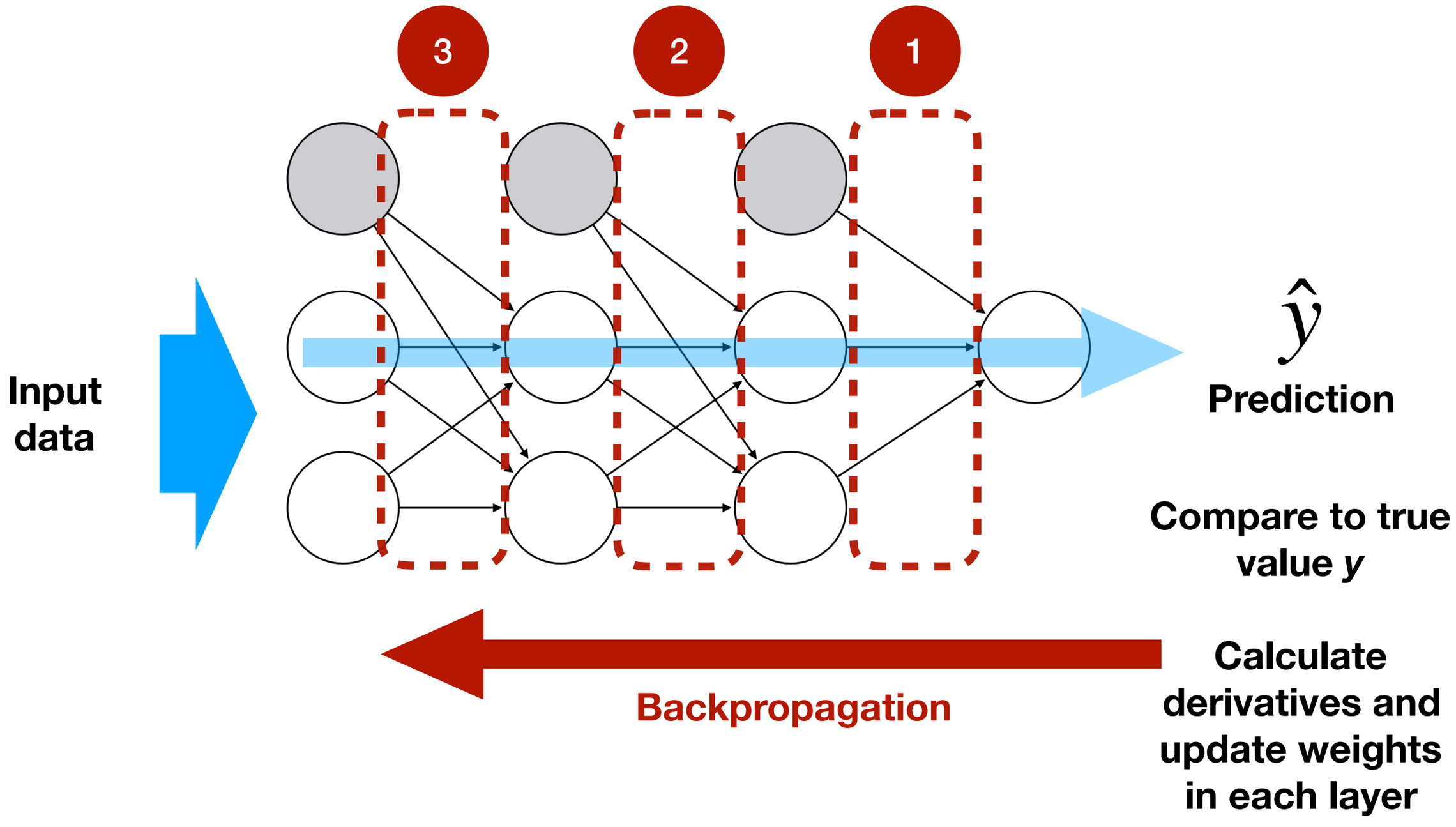


Algorithm: Gradient Descent

Gradient Descent doesn't work out of the box for NNs

- We need to find a way to update the hidden layers.
- We get the **error signal** at the end of the network.
- Need to **propagate** this signal back through the network.
- This algorithm is called **backpropagation** and is an extension of "vanilla" gradient descent involving the chain rule.

Backpropagation intuition

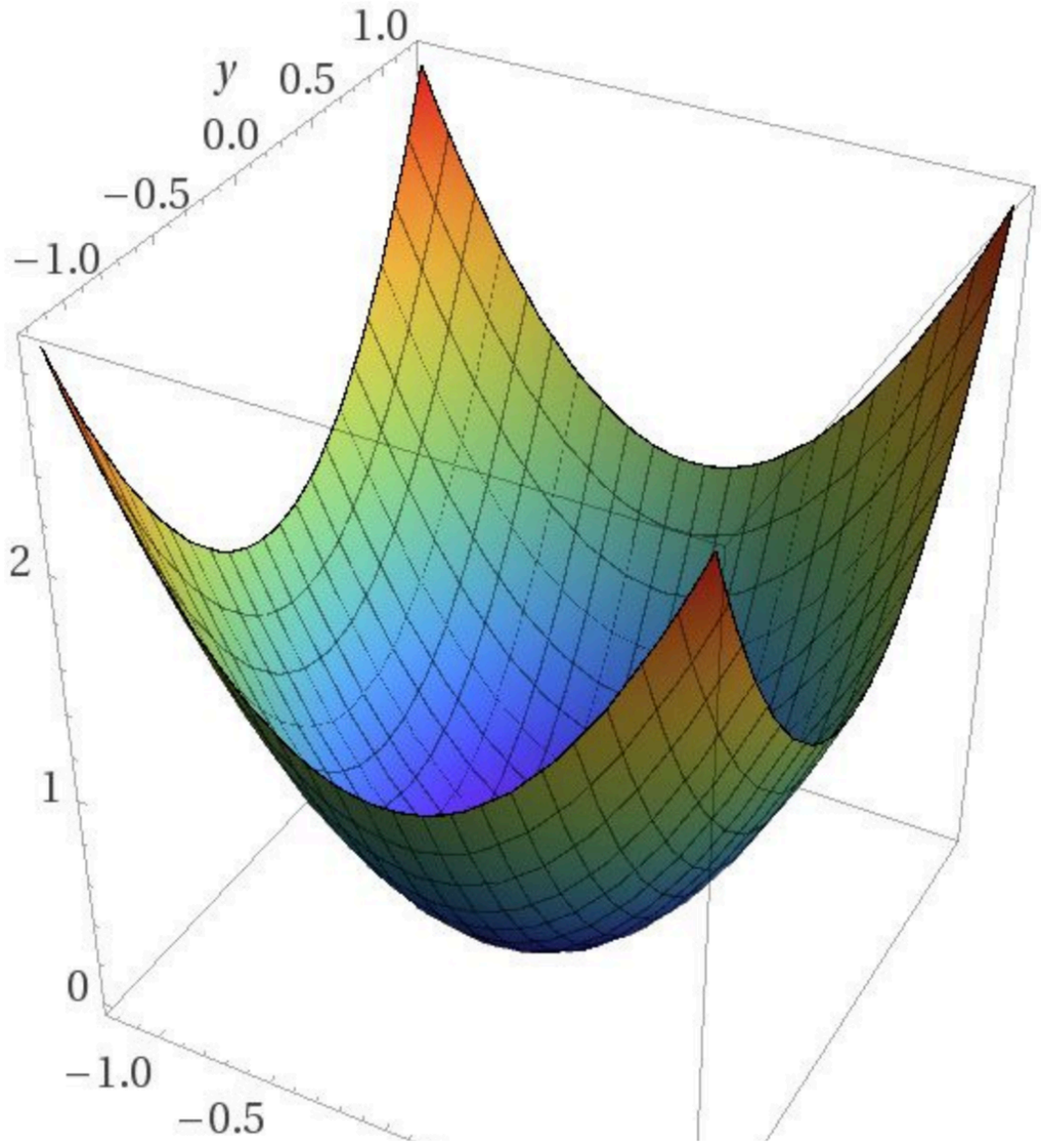


Difference in optimization difficulty

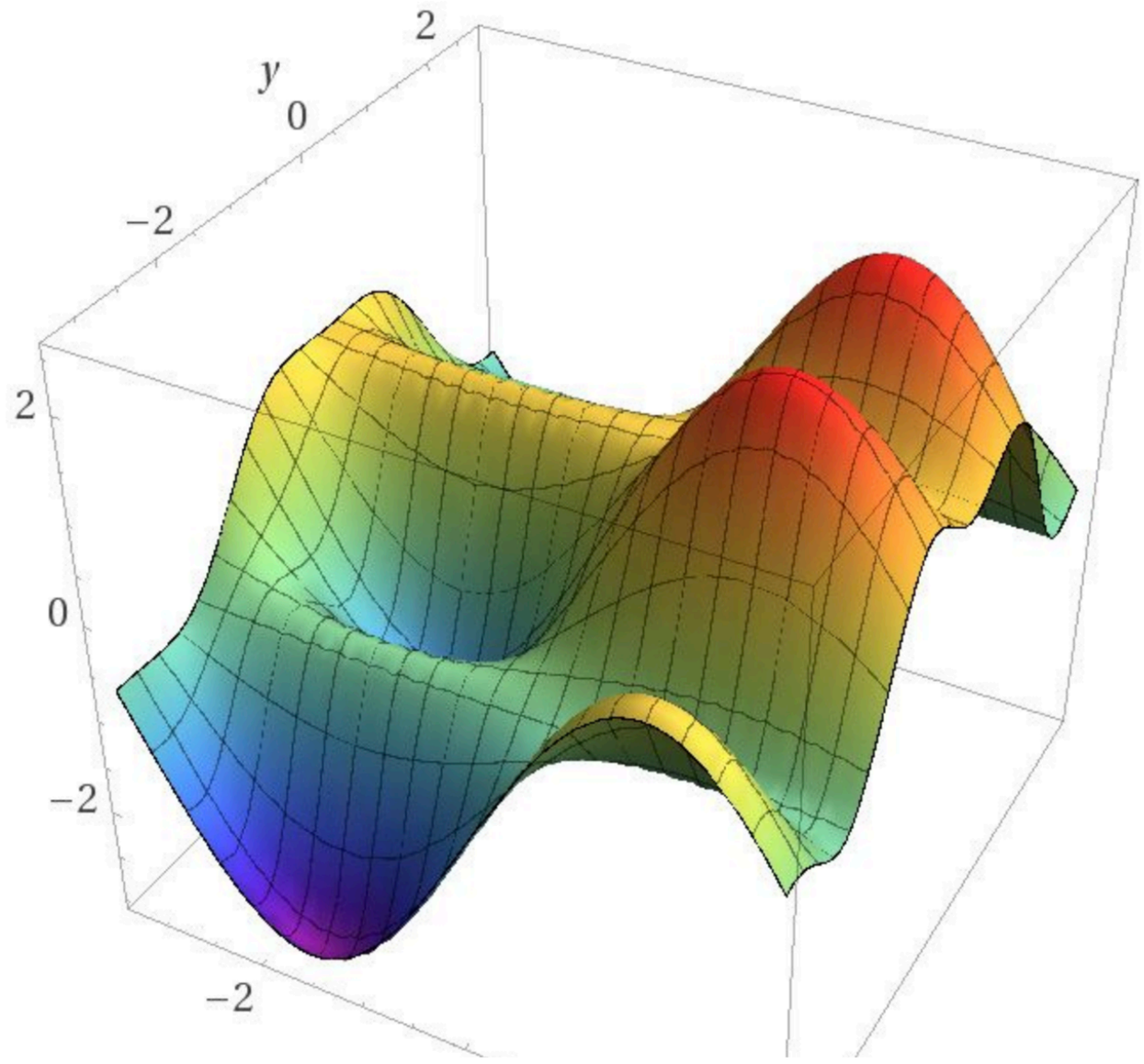
- The optimization problem for linear/logistic regression is **convex**: gradient descent will always find the **global minimum**, given enough time and an appropriate **learning rate**.

- In contrast, the optimization problem for Deep Learning is non-convex, and may settle in a **local minimum**.

convex v.s. non-convex



Linear/Logistic regression



Deep Learning

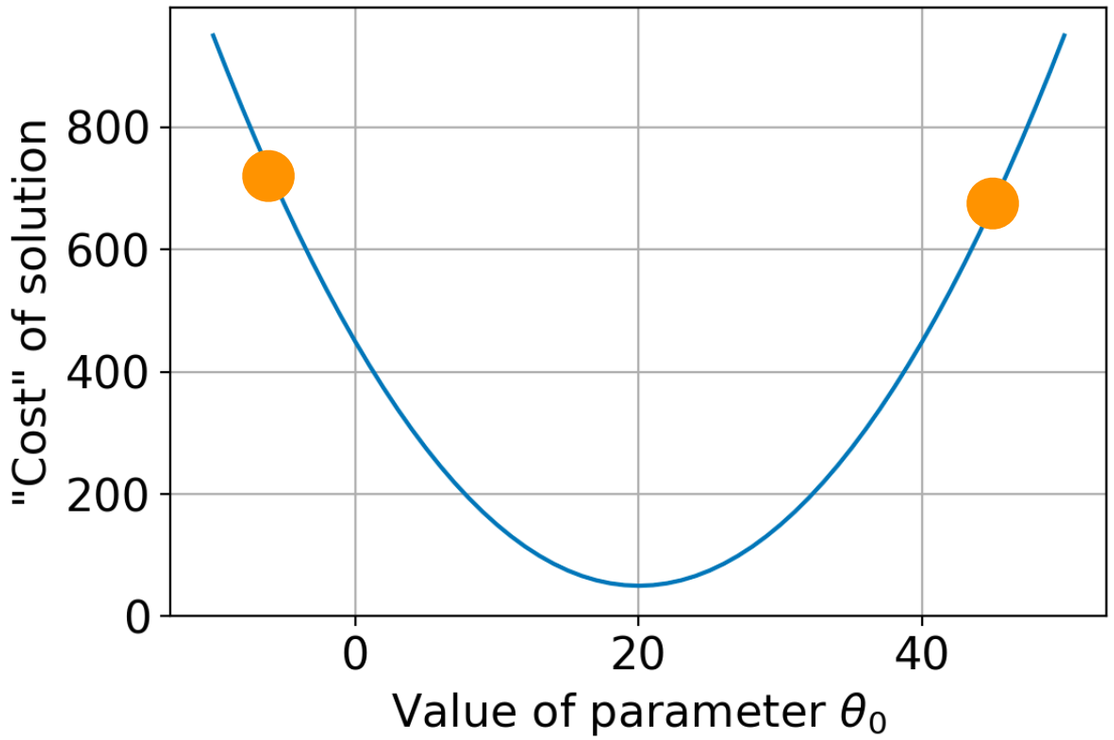
An "appropriate" learning rate

- Learning rate controls the size of the update step.

- Too small: it will take a very long time too converge.

- Too large: it will overshoot the minimum, fail to converge and possibly even diverge.

- Should be “just right”.



Learning rate attention points

- Pick an “appropriate” learning rate (LR).
- Typically 0.3, 0.03, 0.003, 0.0003
    - Most people try a range of options and pick the one with the best CV score after a reasonable amount of training.
- LR should be large enough to allow it to “jump” out of bad local minima.
- But small enough to allow it to settle deeply into a good local minimum.
    - Finding the **real** global minimum is very hard to do, and not required to train a good model, since a good local minimum is usually almost as good.
- A good strategy is to lower the LR over time.
    - Known as **learning rate annealing**.

Agenda

- Deep Learning introduction
- Extending Logistic Regression into Deep Learning
- Notebook practice with Deep Learning (+ 15min break at 10:30-10:45)
- Plenary notebook wrap-up
- Specialized Deep Learning architectures
- Lunch
- Towards ConvNets
- A simple ConvNet architecture & Transfer Learning
- Notebook practice
- Plenary notebook wrap-up
- Break
- Advanced ConvNets: localization, object detection, instance segmentation
- A ConvNet application at Schiphol Airport
- End

Notebook Practice

Classification with Deep Learning

https://colab.research.google.com
https://github.com/jvanlier/TIAS_ML_DL

Agenda

- Deep Learning introduction
- Extending Logistic Regression into Deep Learning
- Notebook practice with Deep Learning (+ 15min break at 10:30-10:45)
- Plenary notebook wrap-up
- Specialized Deep Learning architectures
- Lunch
- Towards ConvNets
- A simple ConvNet architecture & Transfer Learning
- Notebook practice
- Plenary notebook wrap-up
- Break
- Advanced ConvNets: localization, object detection, instance segmentation
- A ConvNet application at Schiphol Airport
- End

Agenda

- Deep Learning introduction
- Extending Logistic Regression into Deep Learning
- Notebook practice with Deep Learning (+ 15min break at 10:30-10:45)
- Plenary notebook wrap-up
- Specialized Deep Learning architectures
- Lunch
- Towards ConvNets
- A simple ConvNet architecture & Transfer Learning
- Notebook practice
- Plenary notebook wrap-up
- Break
- Advanced ConvNets: localization, object detection, instance segmentation
- A ConvNet application at Schiphol Airport
- End

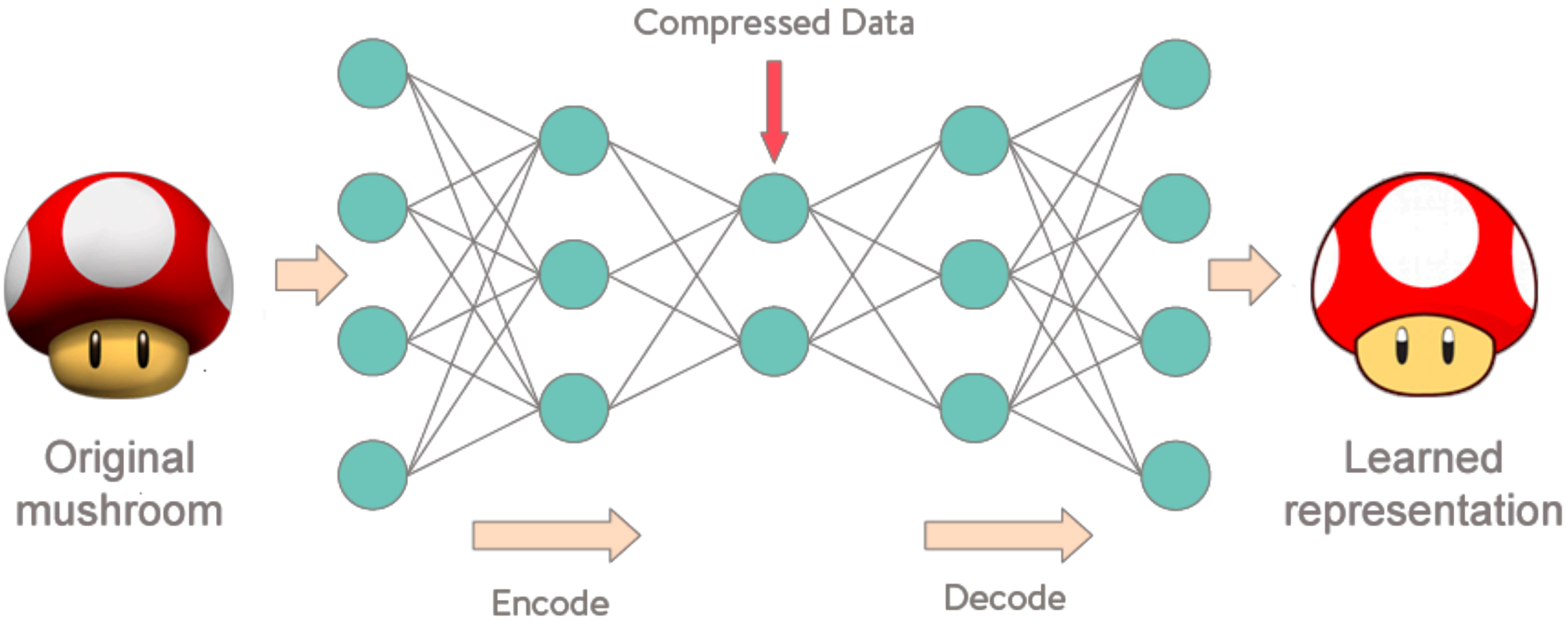
Specialized Deep Learning architectures

- Unsupervised Learning
    - Autoencoders: finding a compressed representation of something
- Supervised Learning
    - (classic) regression and classification on structured tabular data
    - Convolutional Neural Networks that work well with image data
    - Sequence Models that work well with time series
        - Recurrent Neural Network (RNN), Long-Short Term Memory (LSTM)
- Generative Adversarial Networks (GAN): generate images, e.g. faces
- Deep Reinforcement Learning (mainly an academic pursuit at this time)

We covered this just now!

We will cover this after lunch…

Autoencoders



The network learns to reconstruct the original image as accurately as possible from the compressed representation.

Sequence models

Used to classify and predict **sequence data**, or **time series**:

A series of data points indexed in time order, with - strictly speaking for time series, but not necessarily for sequence data - each successive point being equally spaced in time.

Time series example: value of Dow Jones Index at the end of every day.

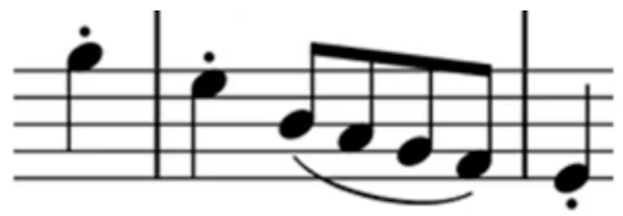
Examples of sequence data

Speech recognition

“The quick brown fox jumped over the lazy dog”

Music generation

genre
or
first few notes



Music generation

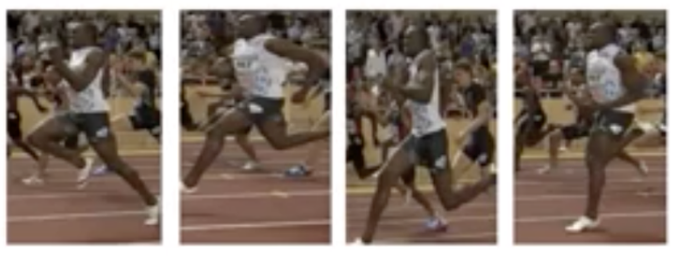
“There is nothing to like in this movie”

Machine translation

“Voulez-vous chanter avec moi?”

Do you want to sing with me?

Video activity recognition



“Running”

Named entity recognition

“Mike is going to the movies in Amsterdam”

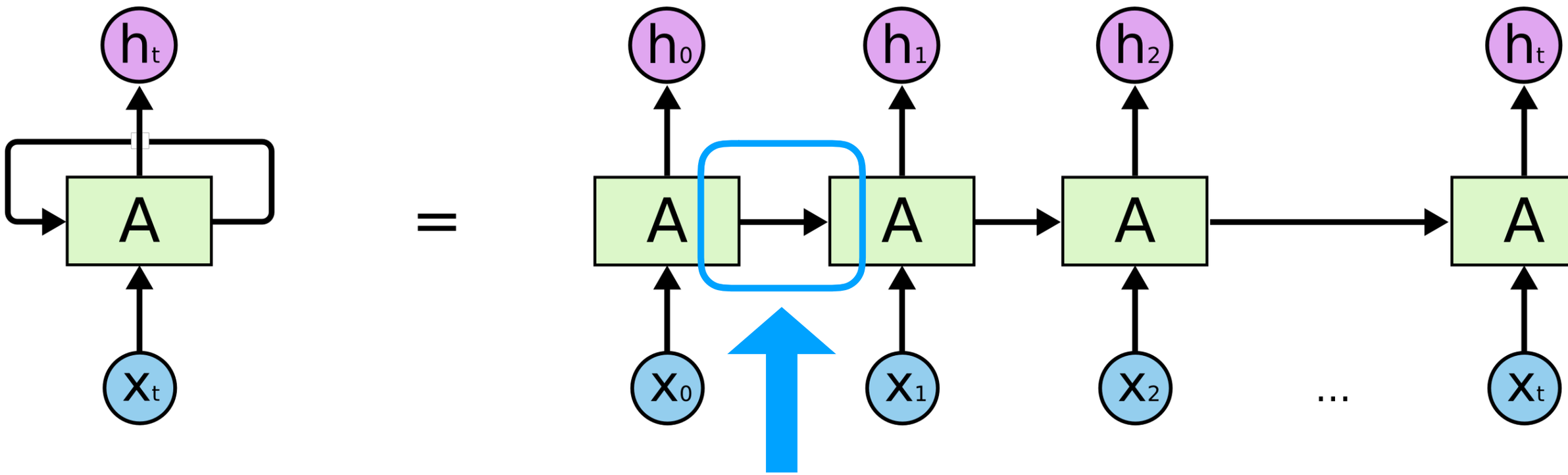
“**Mike** is going to the movies in **Amsterdam**”

Name **City**

A basic Recurrent Neural Network

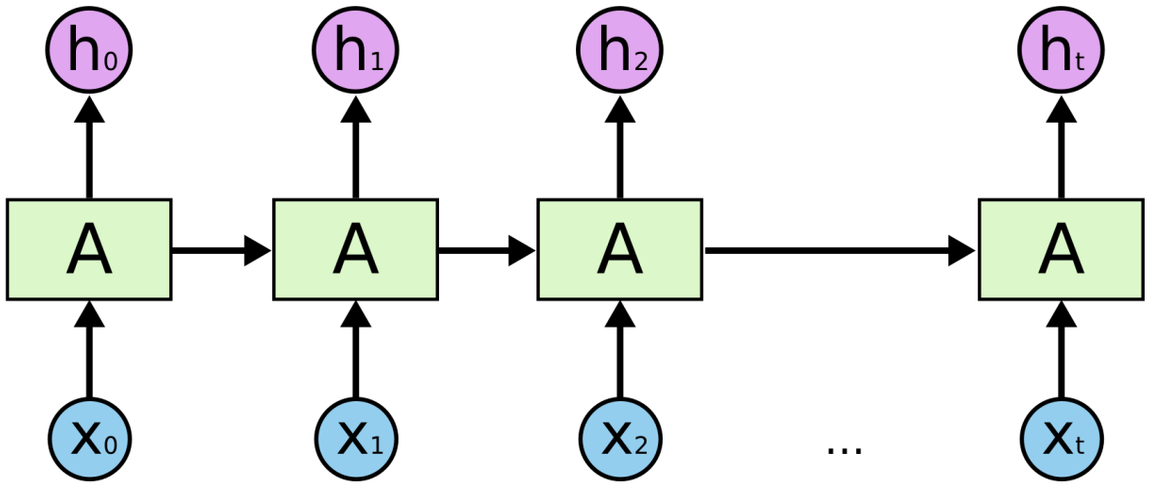
Compact architecture drawing

“Unrolled” architecture drawing



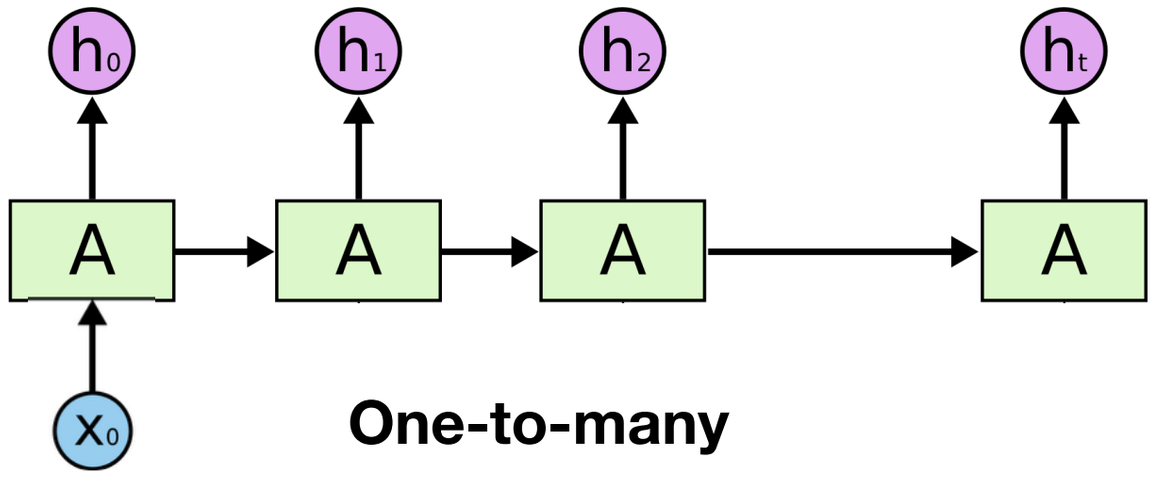
Distinguishing feature: the output of the previous tilmestep is used as input for the next timestep.

Types of RNNs



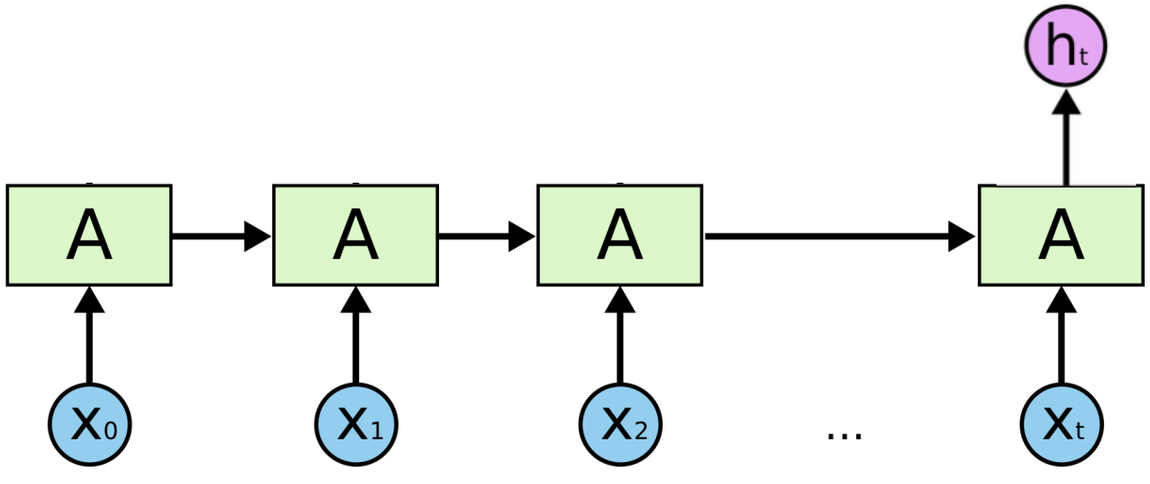
Many-to-many

Task: named entity recognition



One-to-many

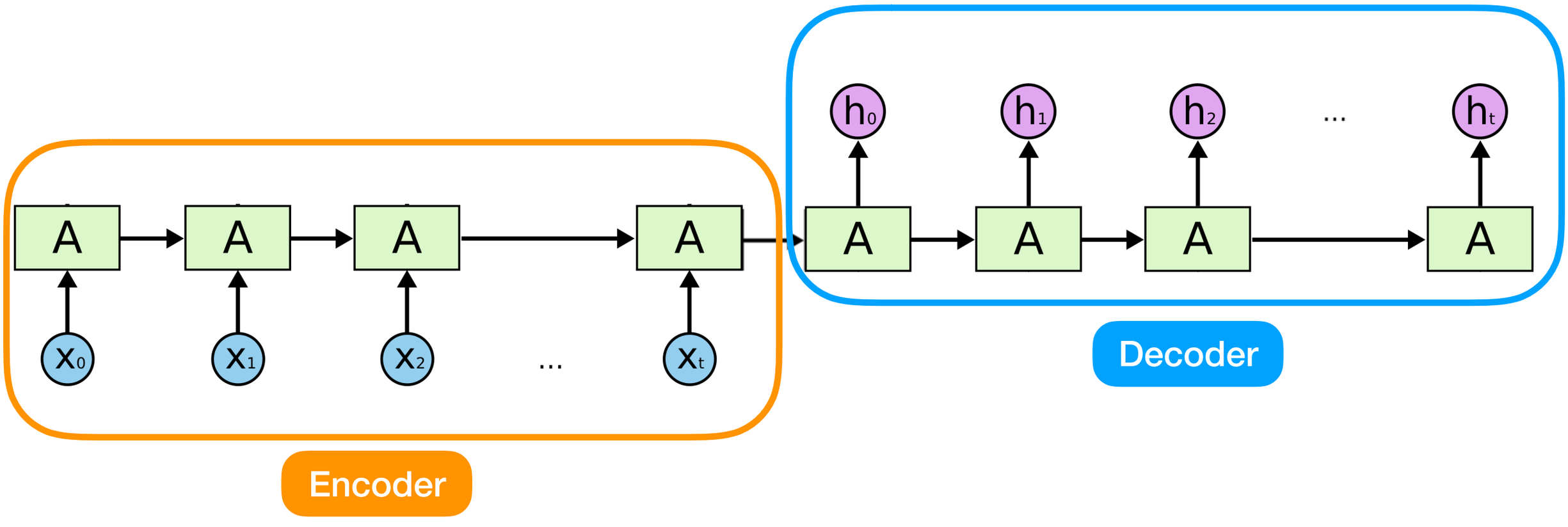
Task: music generation



Many-to-one

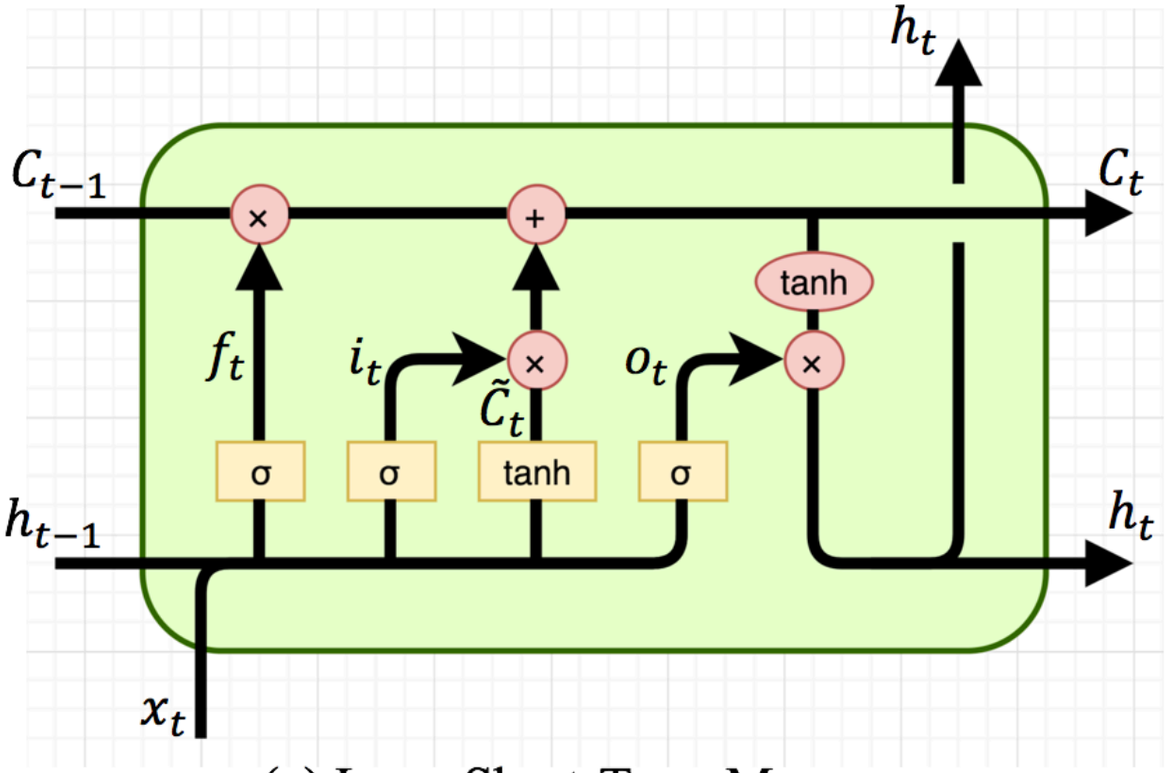
Task: sentiment classification

Many-to-many where the amount of inputs and outputs is different

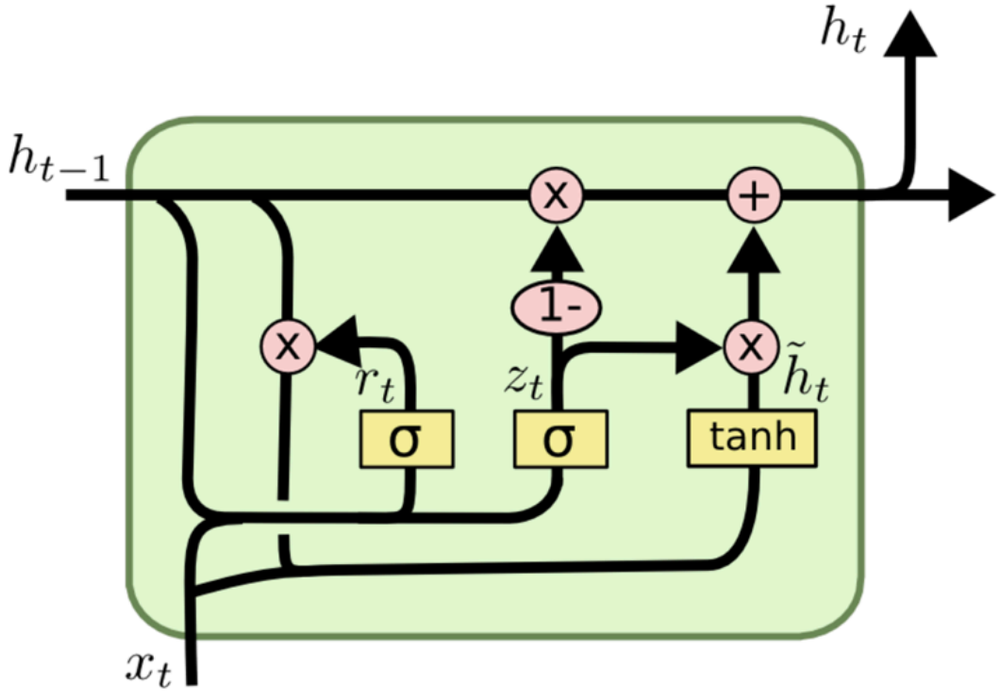


Task: machine translation

RNNs with memory



(a) Long Short-Term Memory



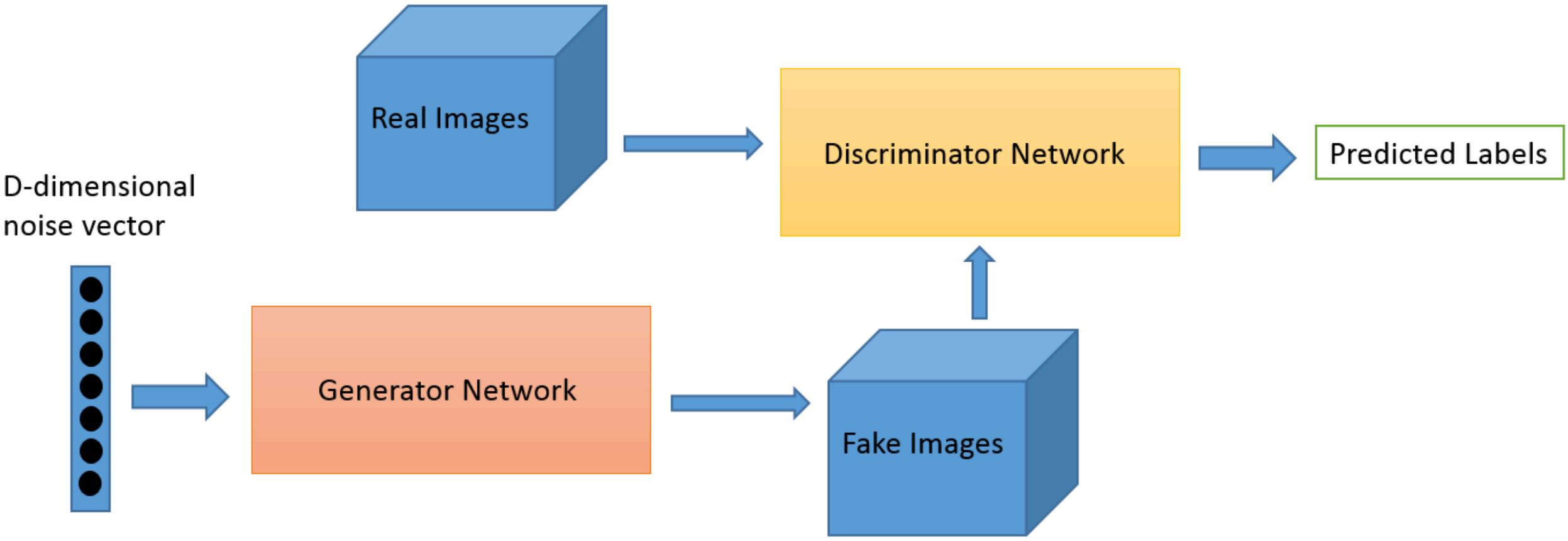
(b) Gated Recurrent Unit

These kind of networks have the capacity to “remember” previous input for multiple timesteps, e.g. the gender of a person.

GANs

- Generative Adversarial Networks
- Consists of two networks: a **Generator** and a **Discriminator**, that work together.
- The Generator learns to create new (fake) images that closely resemble the real images.
- The Discriminator learns to distinguish between the real and fake images (created by the Generator).

GAN - schematic overview



Credit: O'Reilly

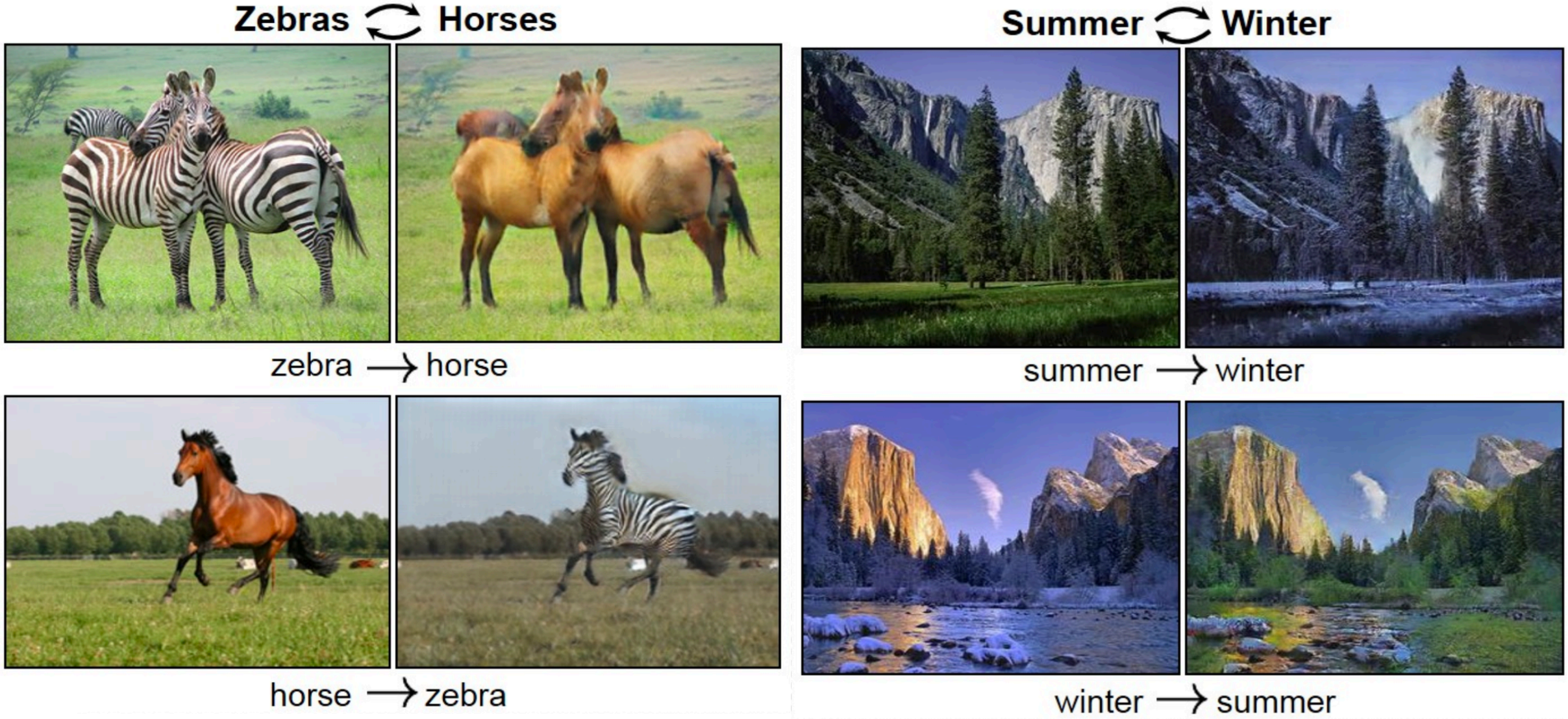
GANs can be used to create art

Somebody bought this for $432.000 …

GANs can be used to create faces

These people do not exist!

GAN style transfer



Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks

Jun-Yan Zhu* **Taesung Park*** **Phillip Isola** **Alexei A. Efros**

UC Berkeley

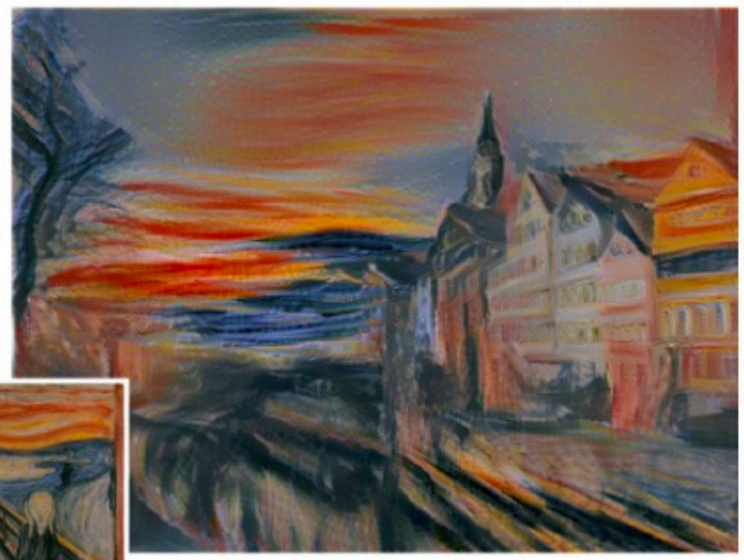
In ICCV 2017

A

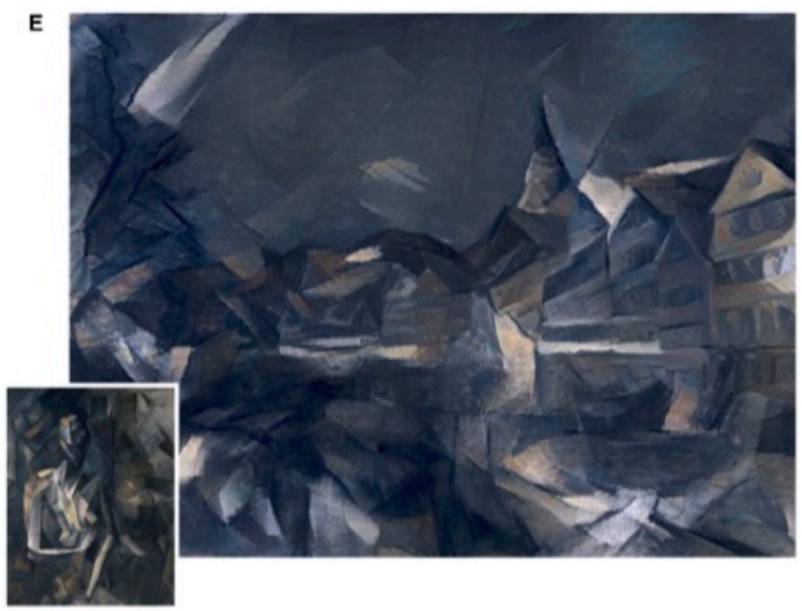
B

C

D



E



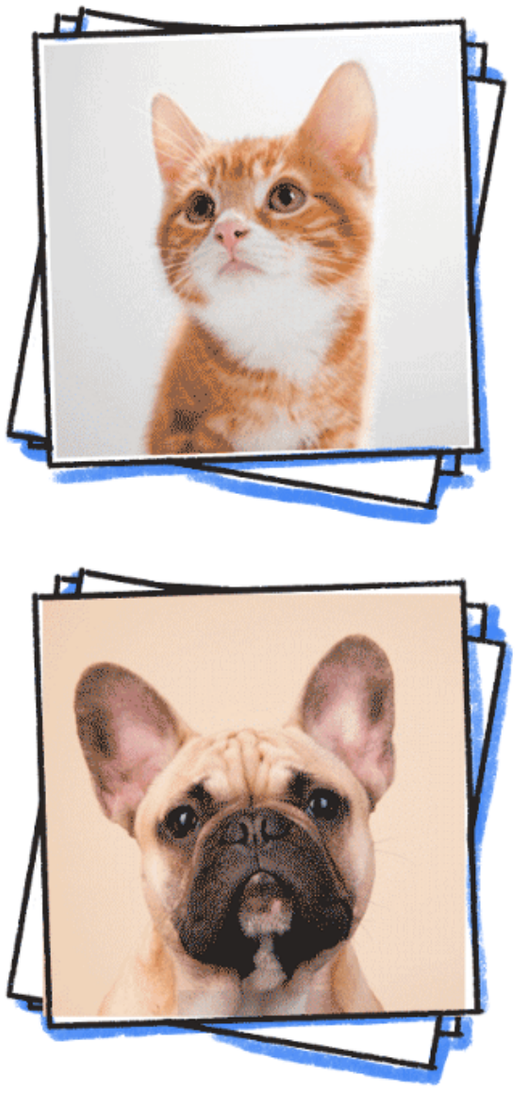
F

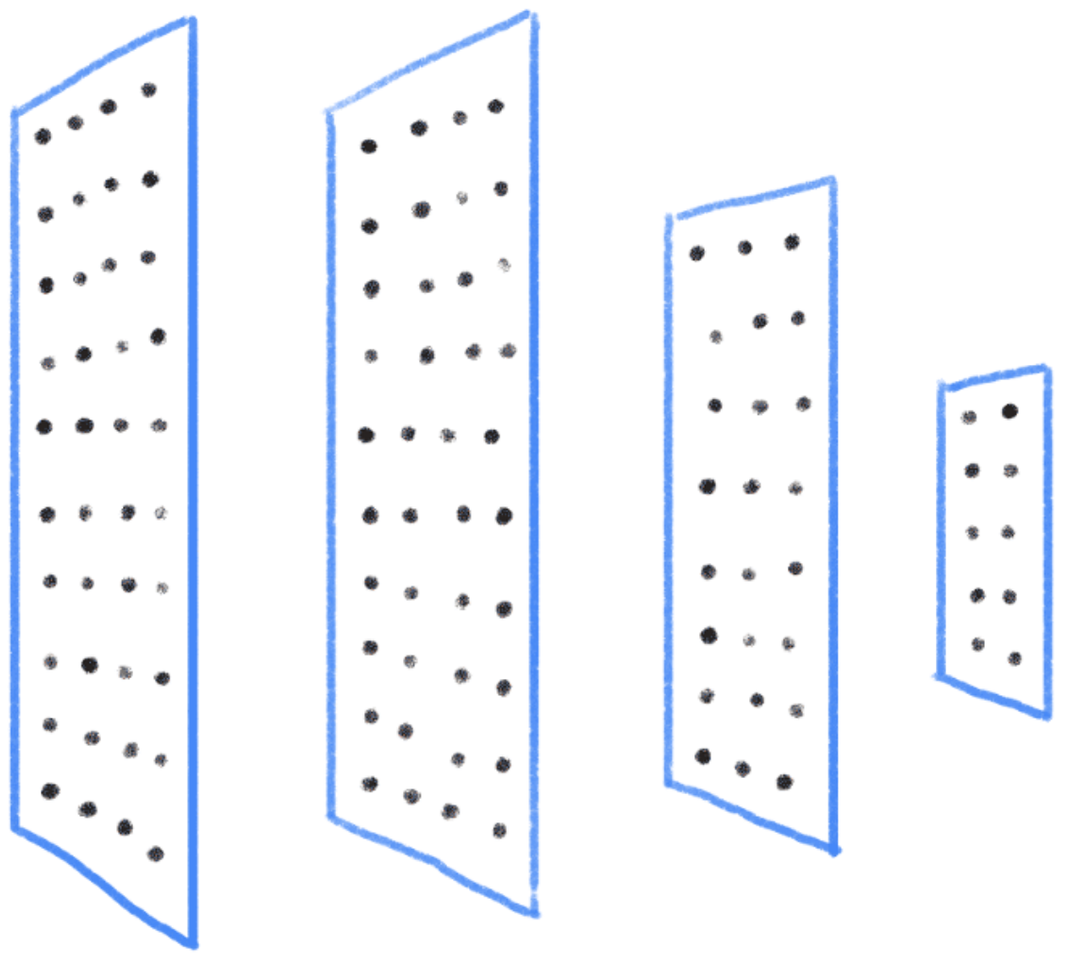
But, today, we'll focus on Convolutional Neural Nets (ConvNets)

CAT

(LABELED PHOTOS)

DOG





OUTPUT

Agenda

- Deep Learning introduction
- Extending Logistic Regression into Deep Learning
- Notebook practice with Deep Learning (+ 15min break at 10:30-10:45)
- Plenary notebook wrap-up
- Specialized Deep Learning architectures
- Lunch
- Towards ConvNets
- A simple ConvNet architecture & Transfer Learning
- Notebook practice
- Plenary notebook wrap-up
- Break
- Advanced ConvNets: localization, object detection, instance segmentation
- A ConvNet application at Schiphol Airport
- End

Agenda

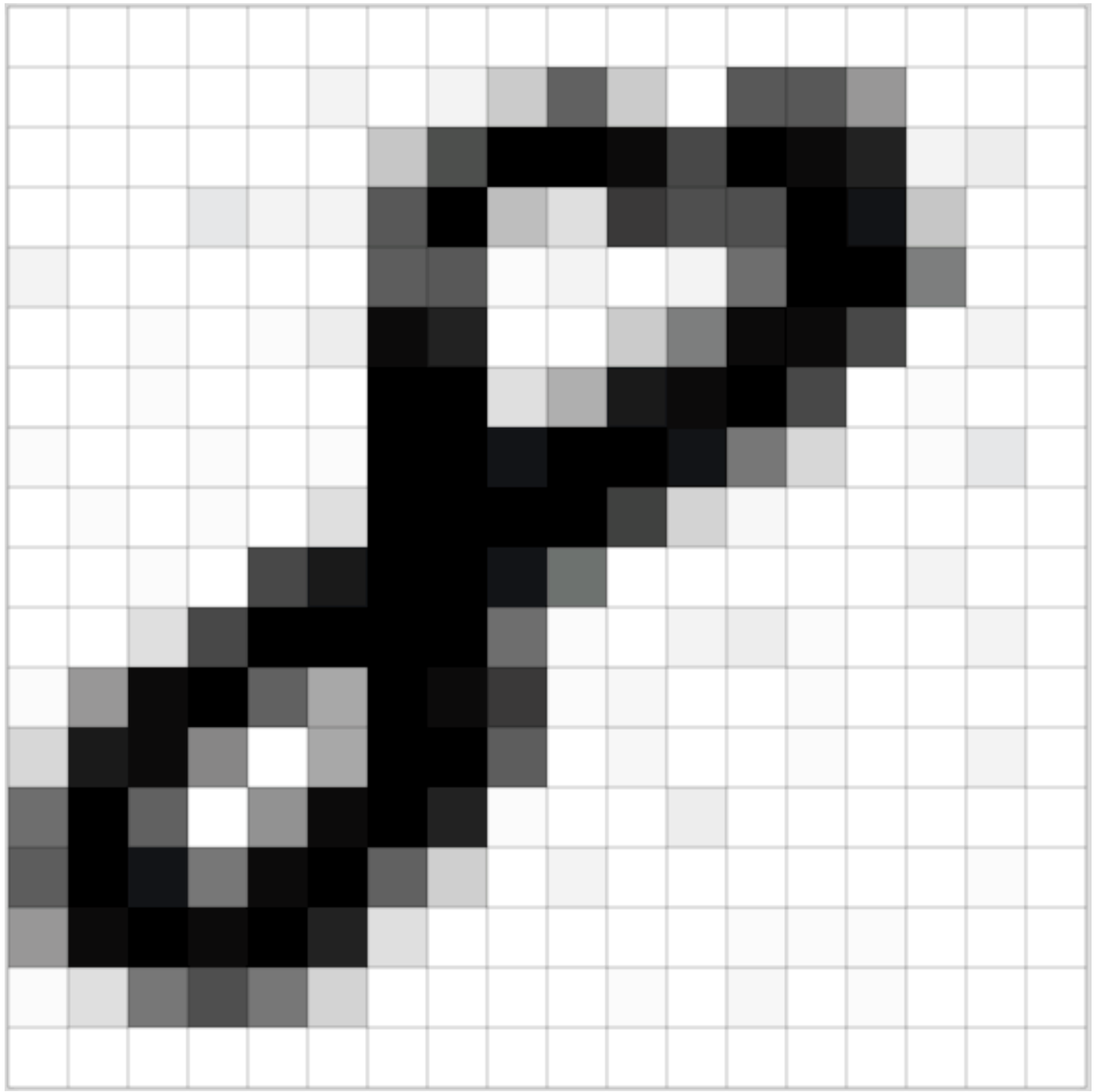
- Deep Learning introduction
- Extending Logistic Regression into Deep Learning
- Notebook practice with Deep Learning (+ 15min break at 10:30-10:45)
- Plenary notebook wrap-up
- Specialized Deep Learning architectures
- Lunch
- Towards ConvNets
- A simple ConvNet architecture & Transfer Learning
- Notebook practice
- Plenary notebook wrap-up
- Break
- Advanced ConvNets: localization, object detection, instance segmentation
- A ConvNet application at Schiphol Airport
- End

From simple Neural Networks to ConvNets

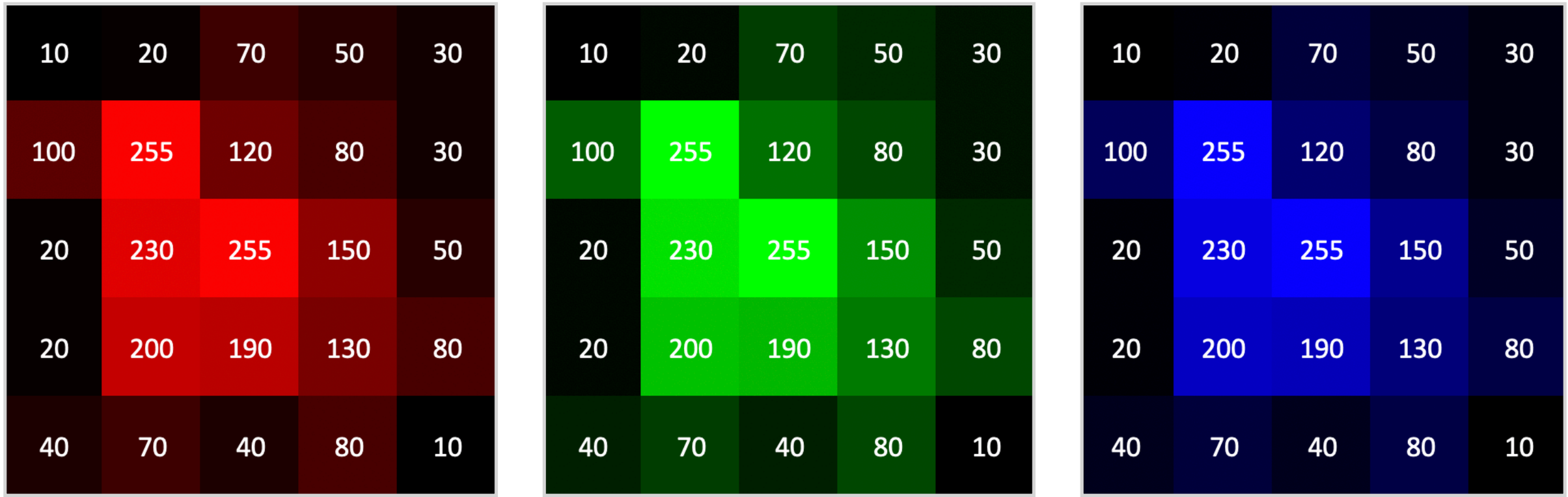
- A couple modifications are needed to make a Neural Network work well with image data:
  - Slightly more complicated input encoding
  - Convolution operator
  - Maxpooling operator
  - Many hidden layers
  - ReLU activation function
  - Dropout

Input encoding

- Image is divided into pixels
- Value of each pixel between 0 and 255.
- In fact, a color image has 3 primary colors or channels: R, G and B (Red, Green and Blue) Each channel has a value that ranges between 0 and 255!



Tensors: 3d tables

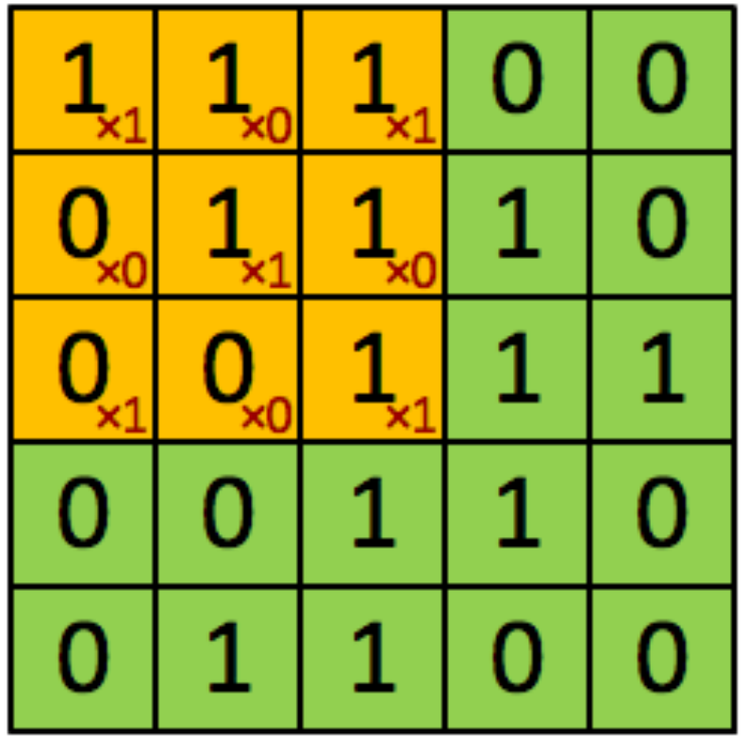


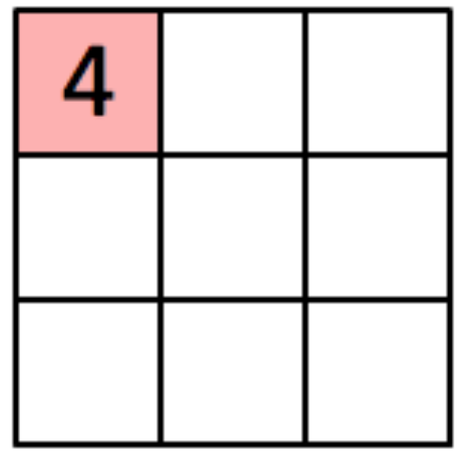
Each color of each pixel is mapped to its own input neuron.
5 x 5 x 3 = 75 inputs already! 720 x 576 x 3 would be 1,244,160 inputs!
These inputs are scaled to be between 0 and 1 (divided by 255).

To keep things simple, we'll continue with 2D grayscale examples…

Convolutions

- Basic idea: **filter** that **slides** over an image
- Multiply image value with filter value, sum everything, and write output in corresponding cell.





Convolutions as edge detectors

Input

10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0
10	10	10	0	0	0

*

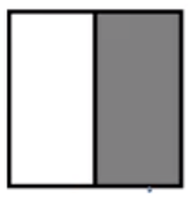
3x3 convolution filter

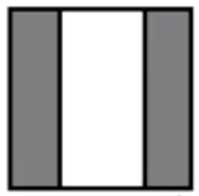
1	0	-1
1	0	-1
1	0	-1

=

output

0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0

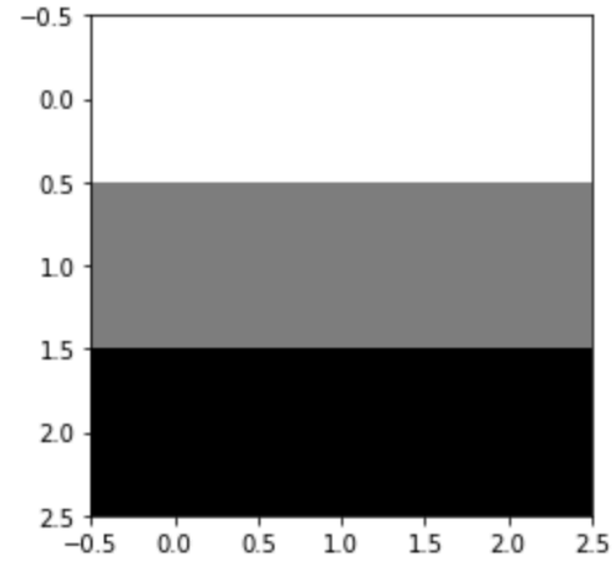


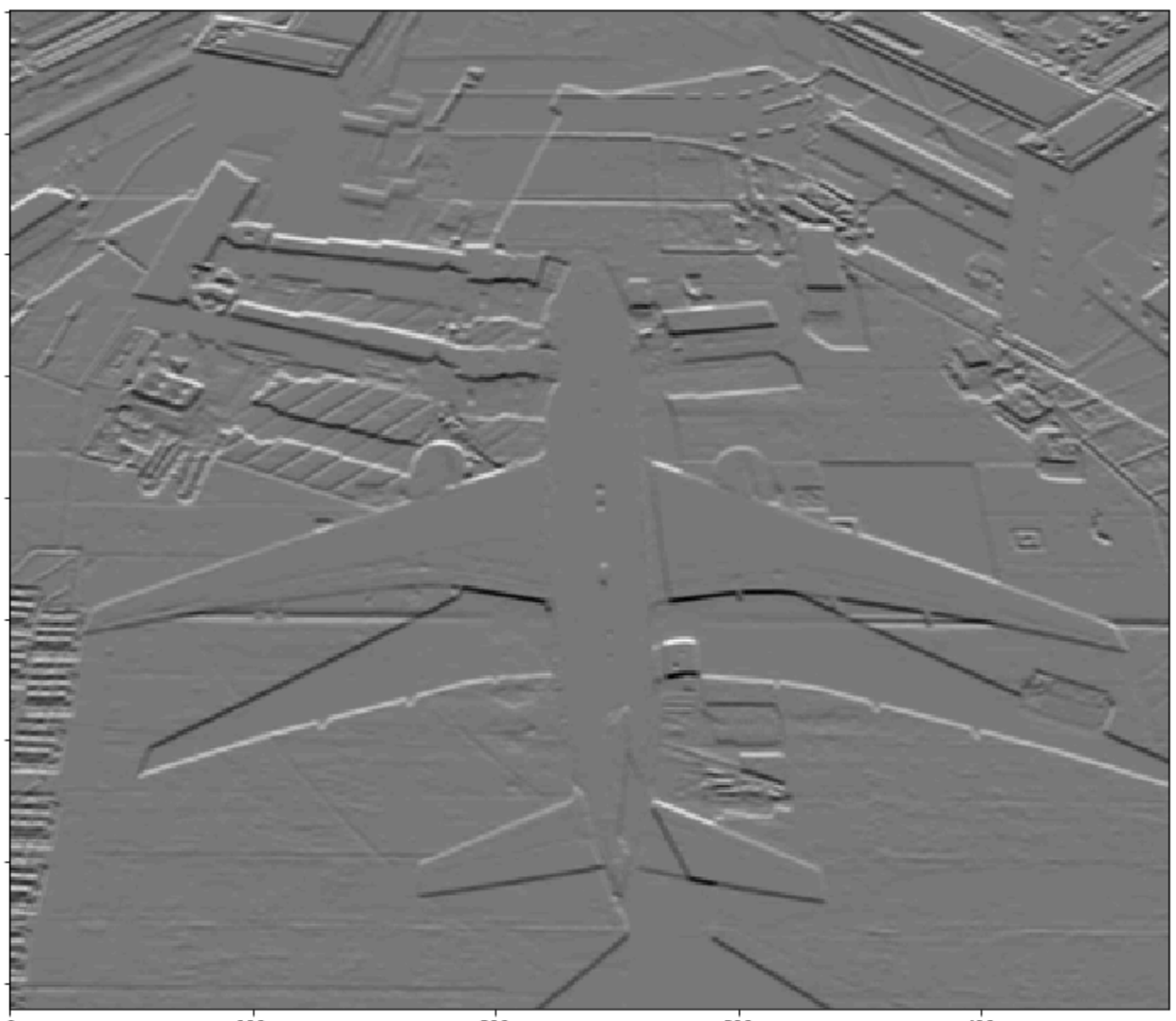


Horizontal edge detector:

```
array([[ 1,  0, -1],
       [ 1,  0, -1],
       [ 1,  0, -1]])
```

```
plt.imshow(hori, cmap='gray');
```

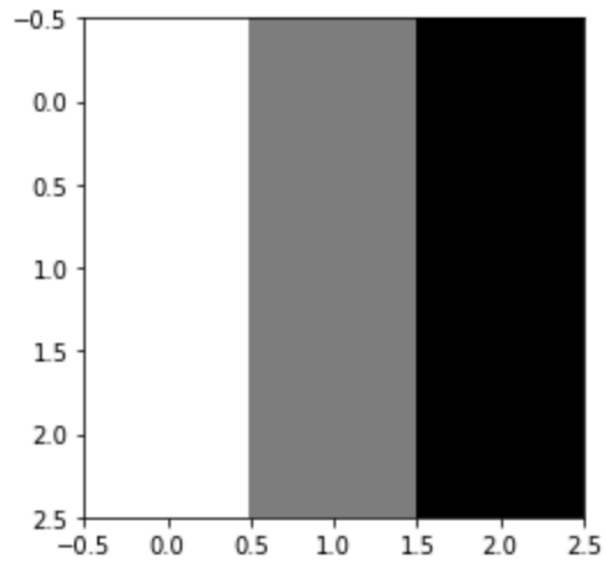


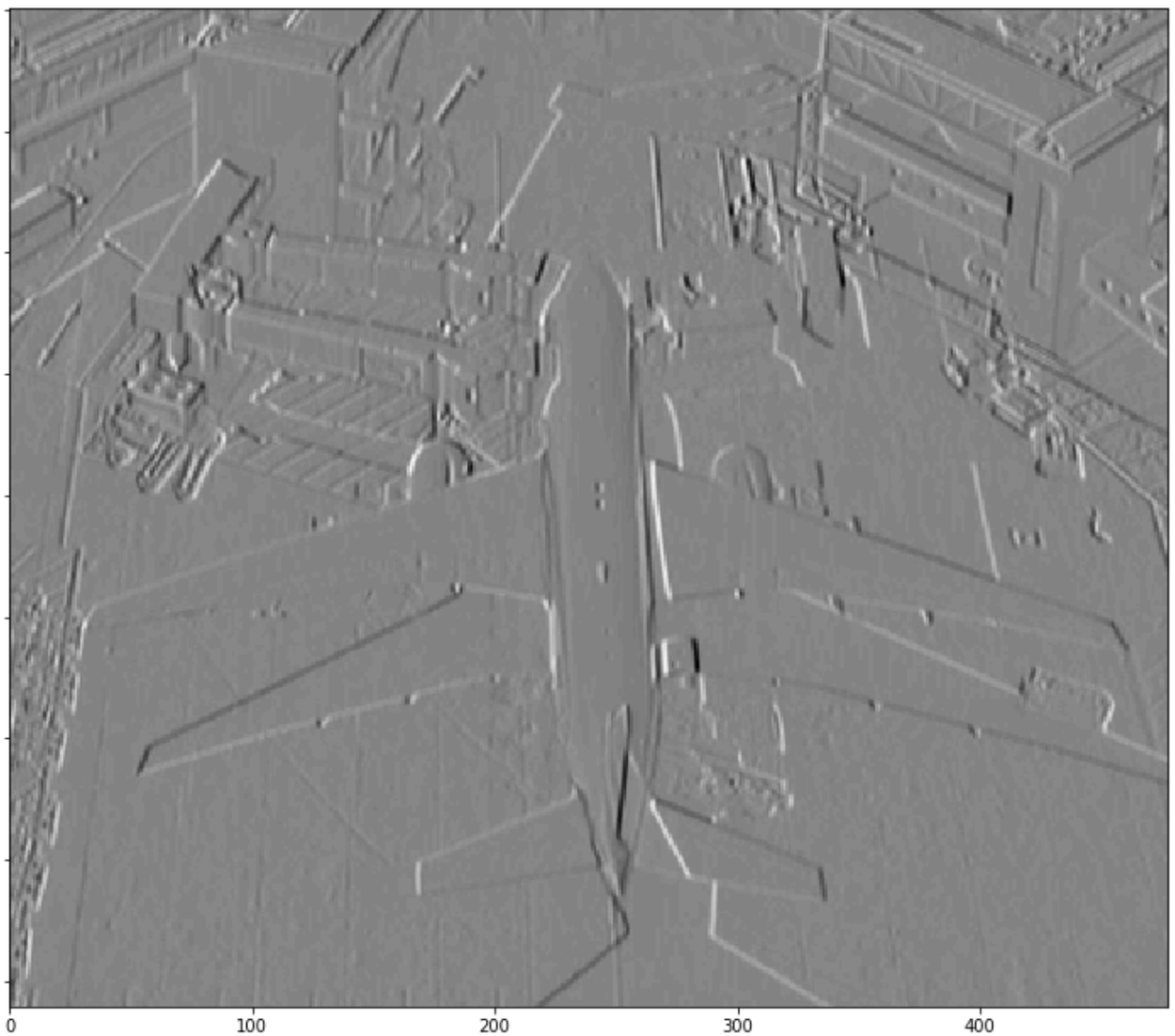


Vertical edge detector:

```
array([[ 1,  1,  1],
       [ 0,  0,  0],
       [-1, -1, -1]])
```

```
plt.imshow(vert, cmap='gray');
```





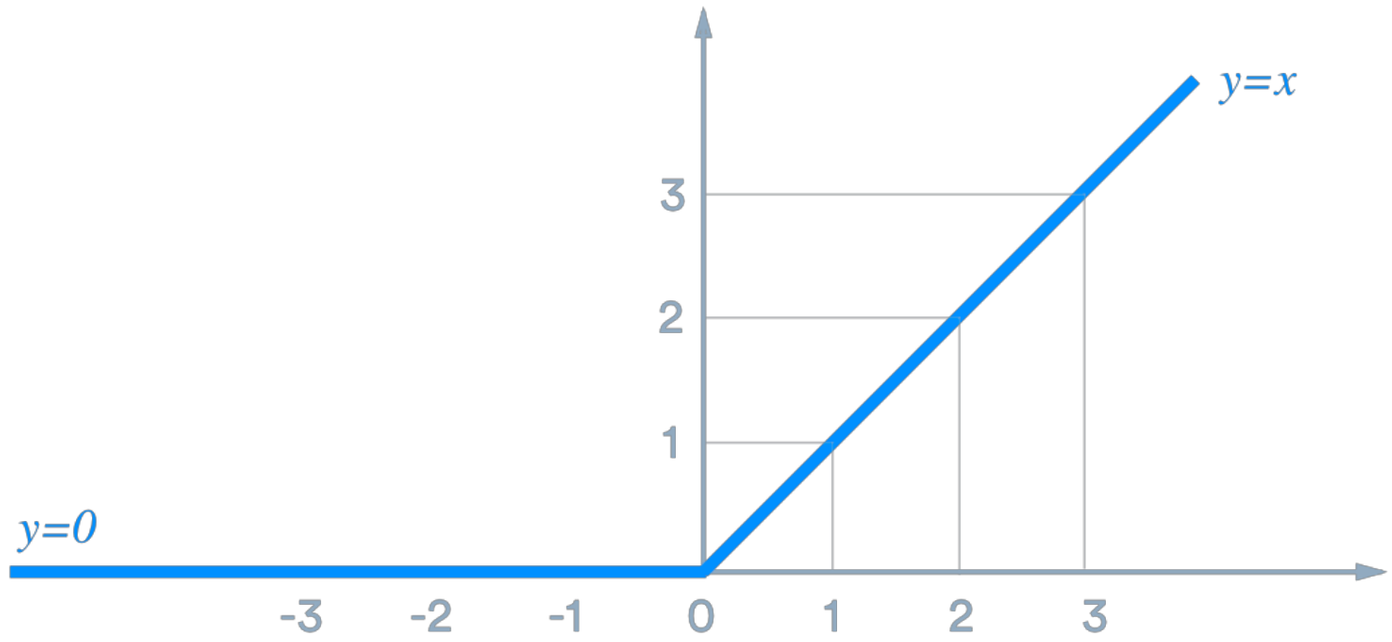
Convolutions are not new…

- Have been used in image processing for many years
- But: they were carefully crafted by domain experts for specific purposes, e.g.:
  - Various forms of edge detection
  - Blurring
  - Sharpening
- Key: ConvNet **automatically** learns the parameters of the filters!

ReLU activation function

- Trivially simple activation function:

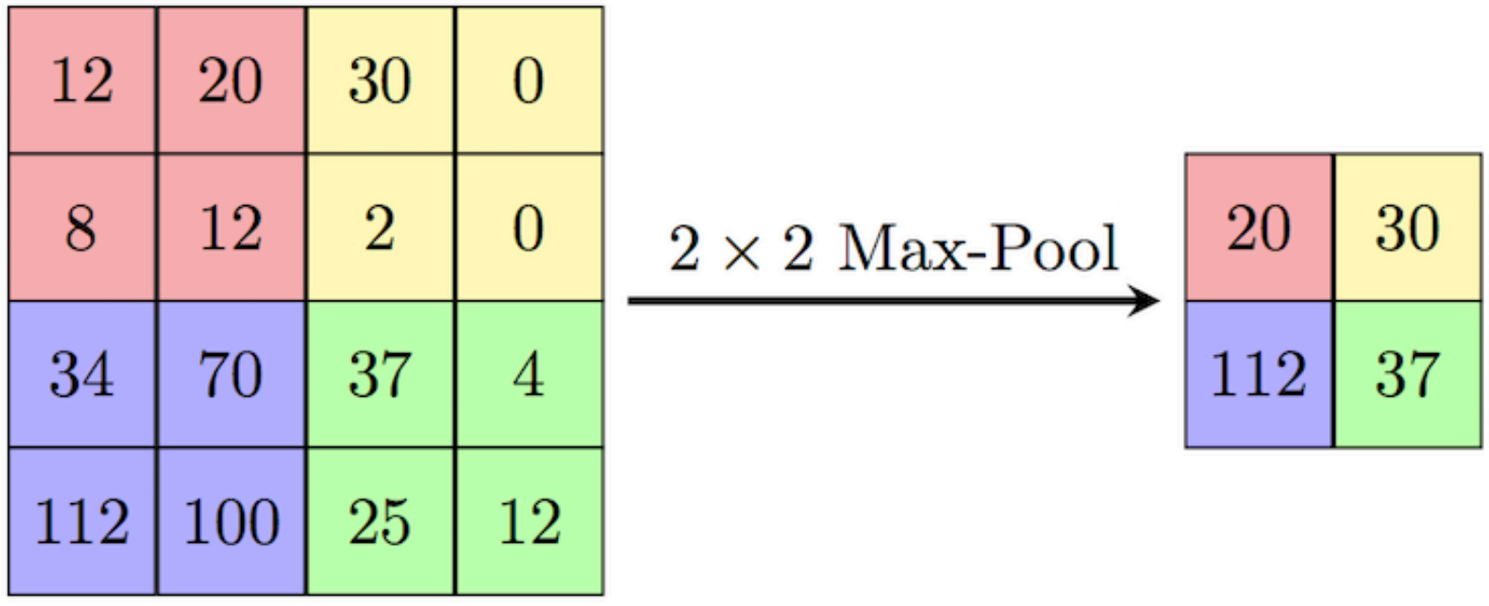
$$f(x) = \max(0,x)$$



- Why?
  - Tanh and sigmoid creates only small values, which complicates the training process when networks are very deep (more on this later).
  - By clipping negative values to 0, we create “sparse” networks that don’t actually use all the weights —> Increased performance in production!

Max-pooling

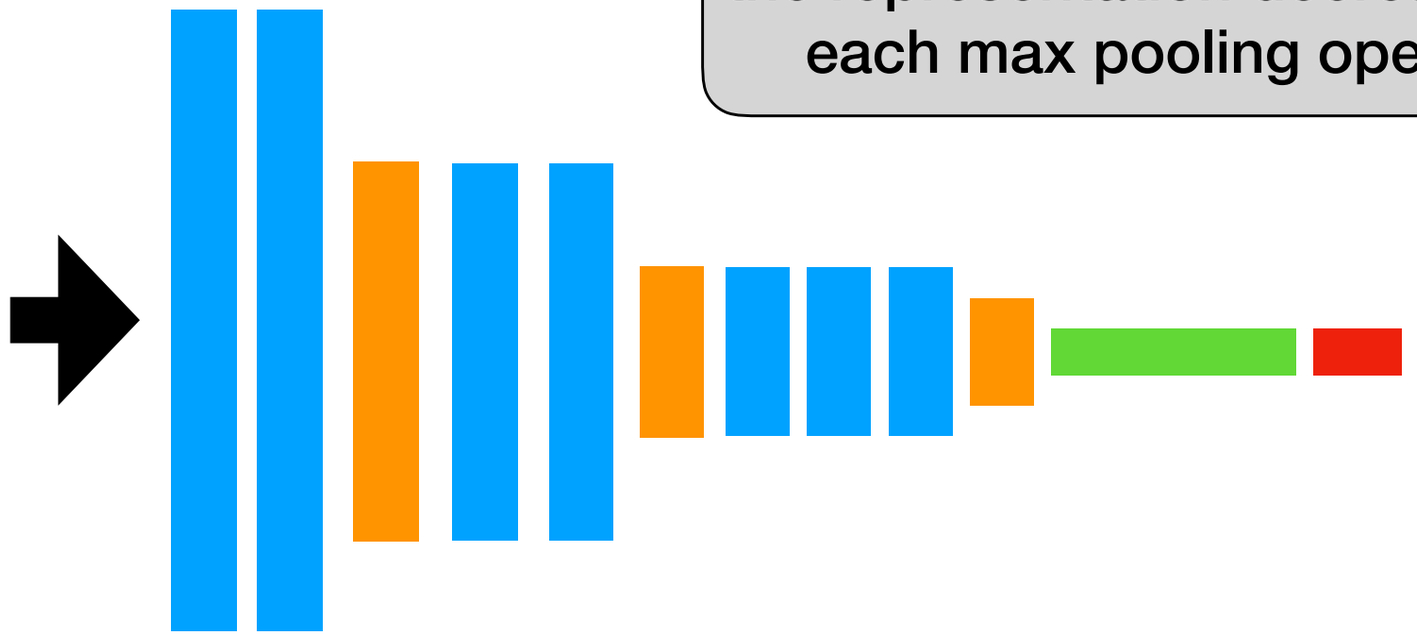
- A Convolution layer is typically followed by a Max-pooling layer



- This accomplishes three things:
  - It helps reduce over-fitting by providing an abstracted form of the representation.
  - It reduces computation cost by reducing the number of parameters to learn.
  - It makes it more robust to things being in a slightly different location.

Repeat!

Notice how the width & height of the representation decreases after each max pooling operation.



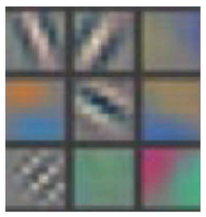
Convolution + ReLu

Max Pooling

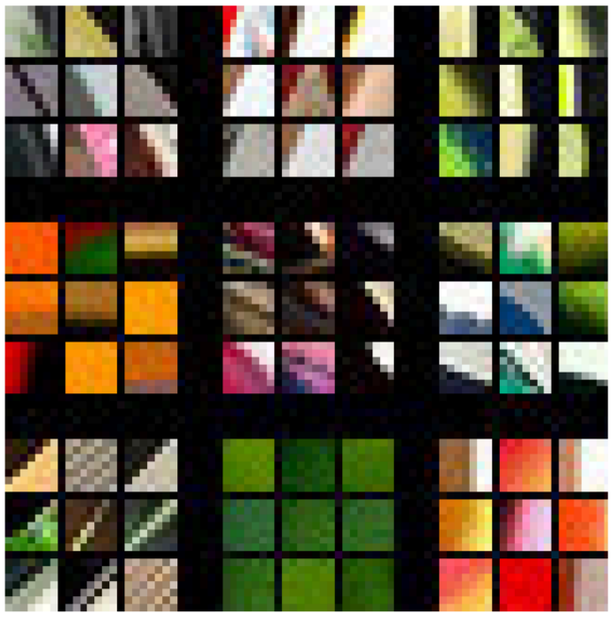
Fully Connected + ReLU

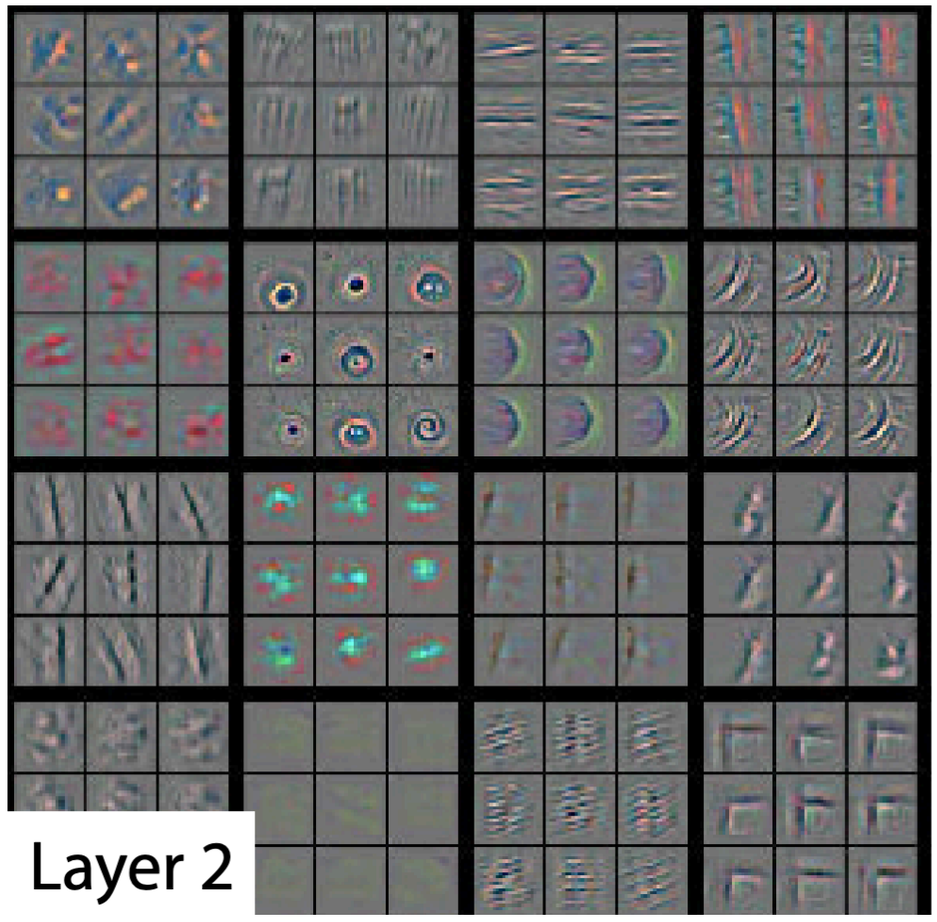
Softmax

What does each layer learn?
layer 1 & 2

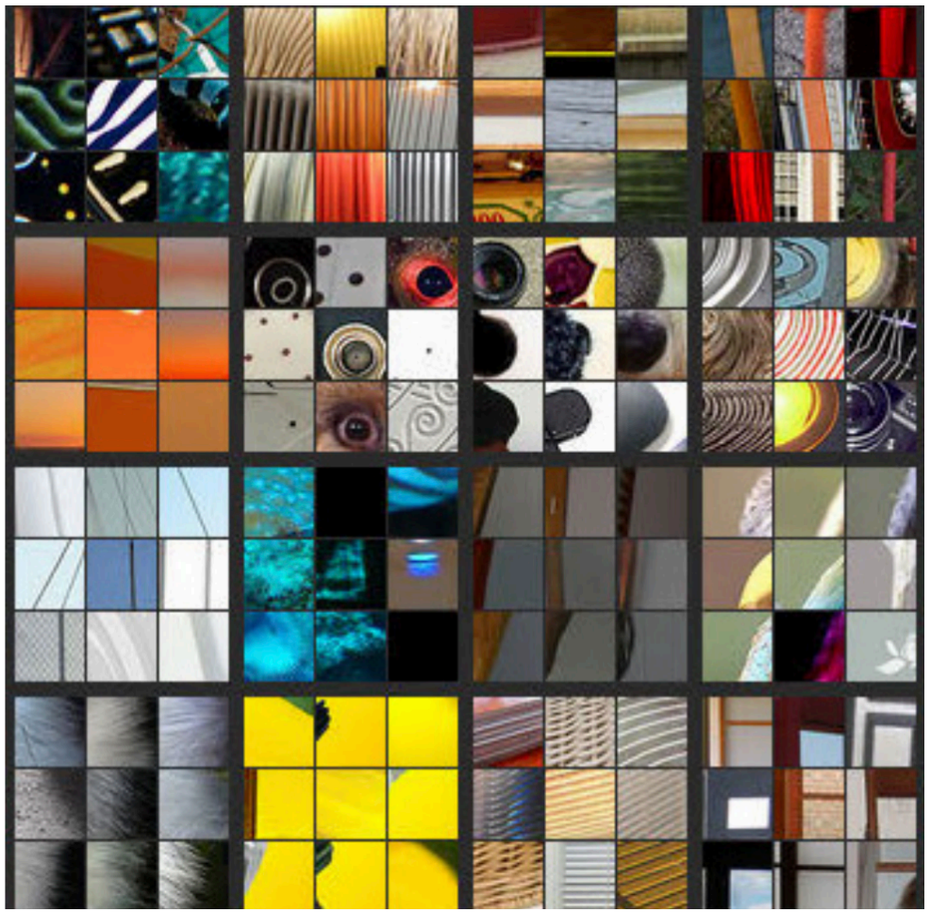


Layer 1



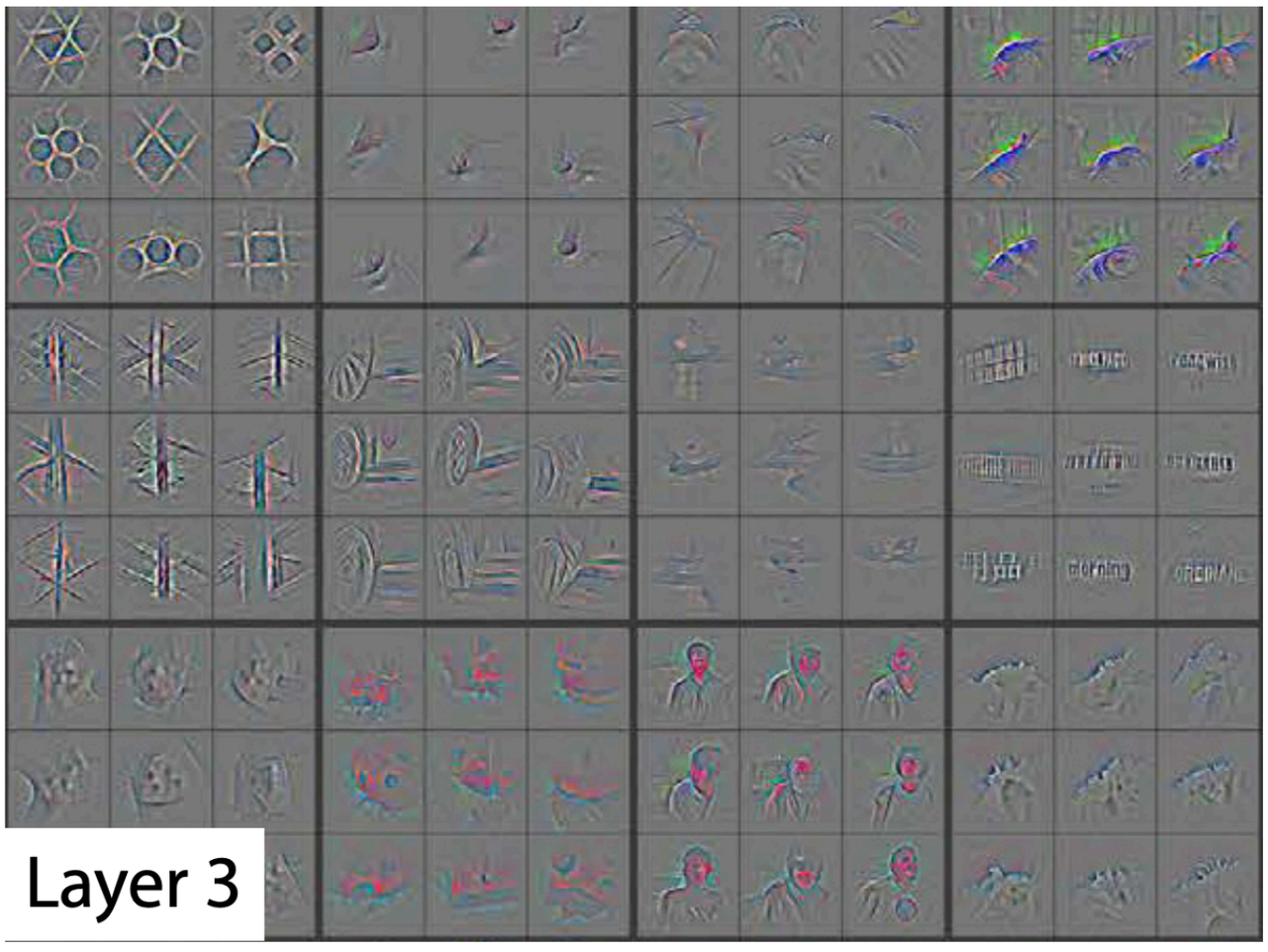


Layer 2



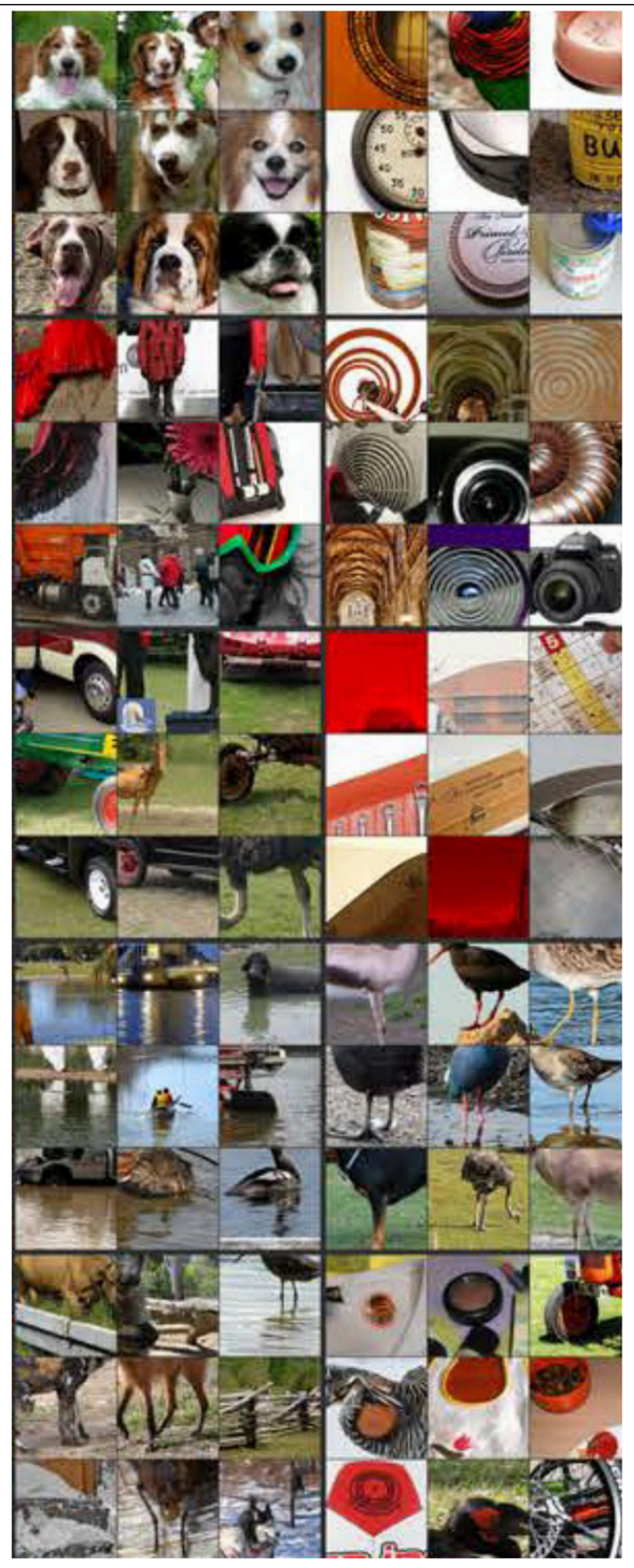
Zeiler and Fergus: Visualizing and Understanding Convolutional Networks, 2014

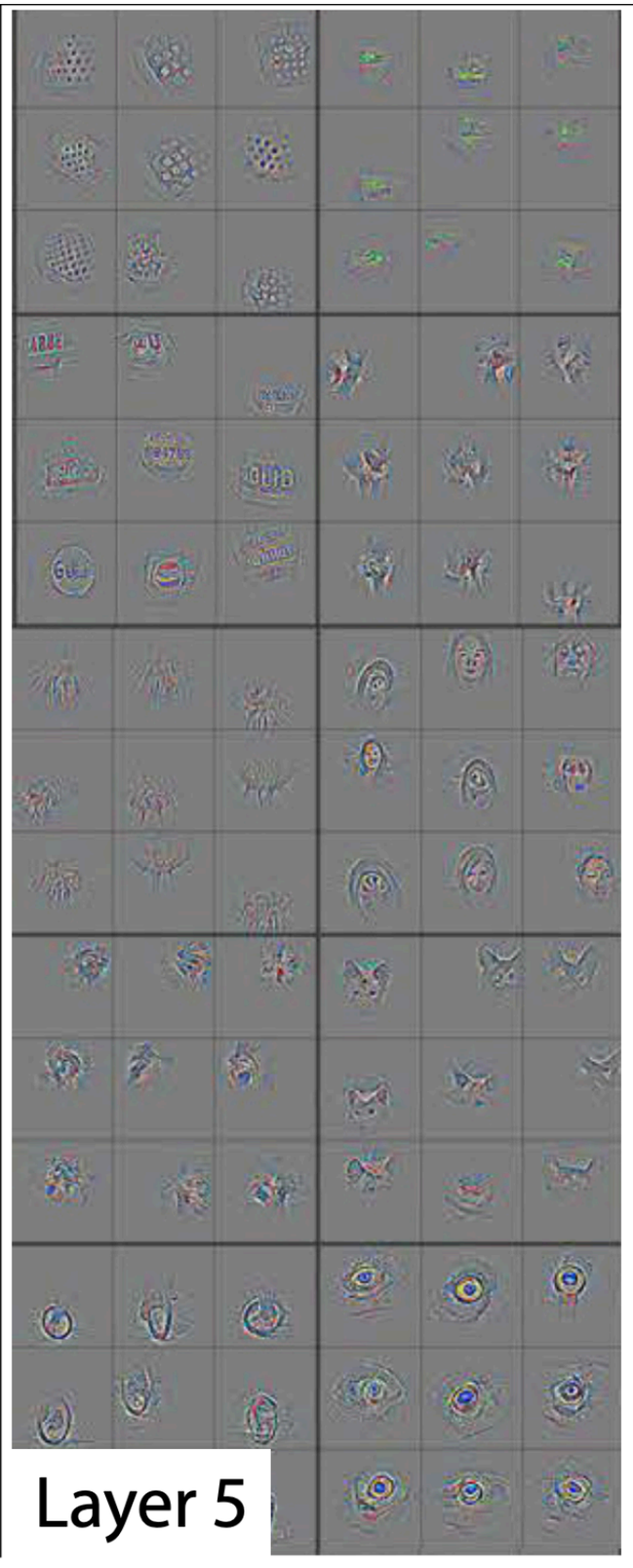
Layer 3

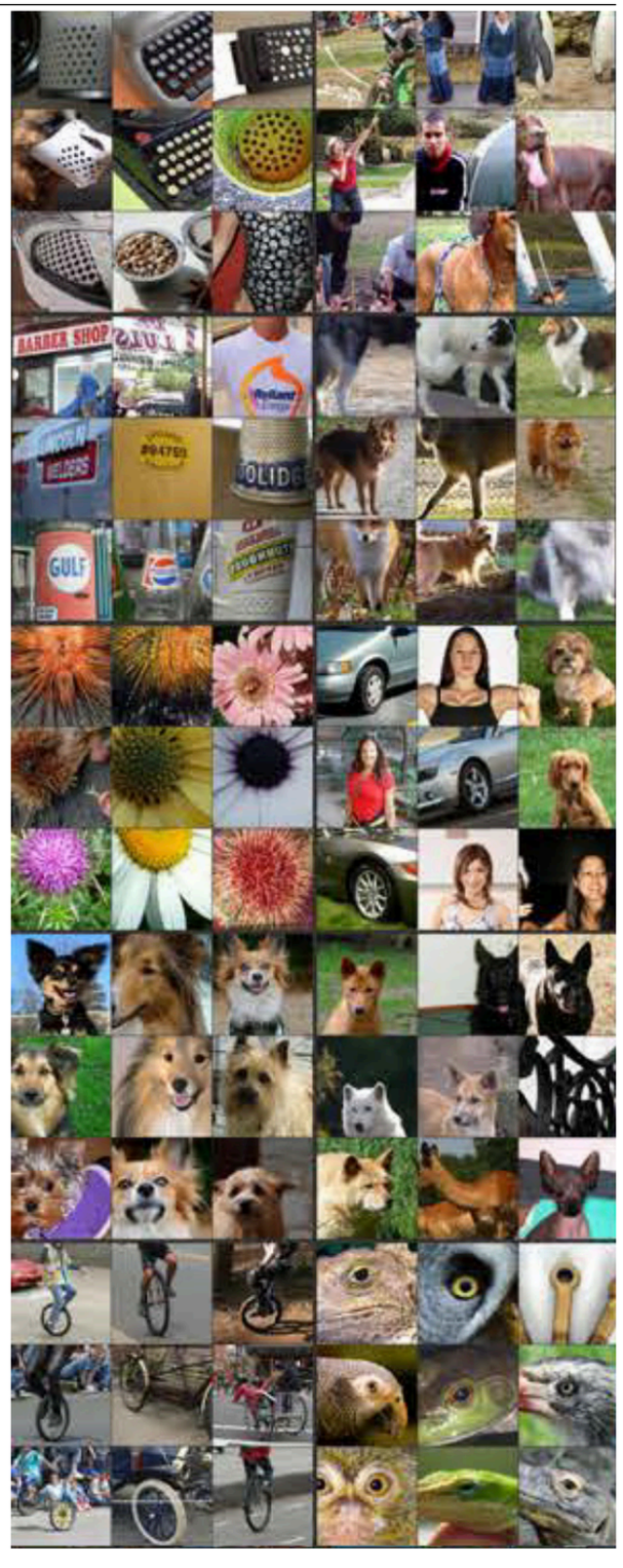


Zeiler and Fergus: Visualizing and Understanding Convolutional Networks, 2014

Layer 4 & 5







Zeiler and Fergus: Visualizing and Understanding Convolutional Networks, 2014

How about training?

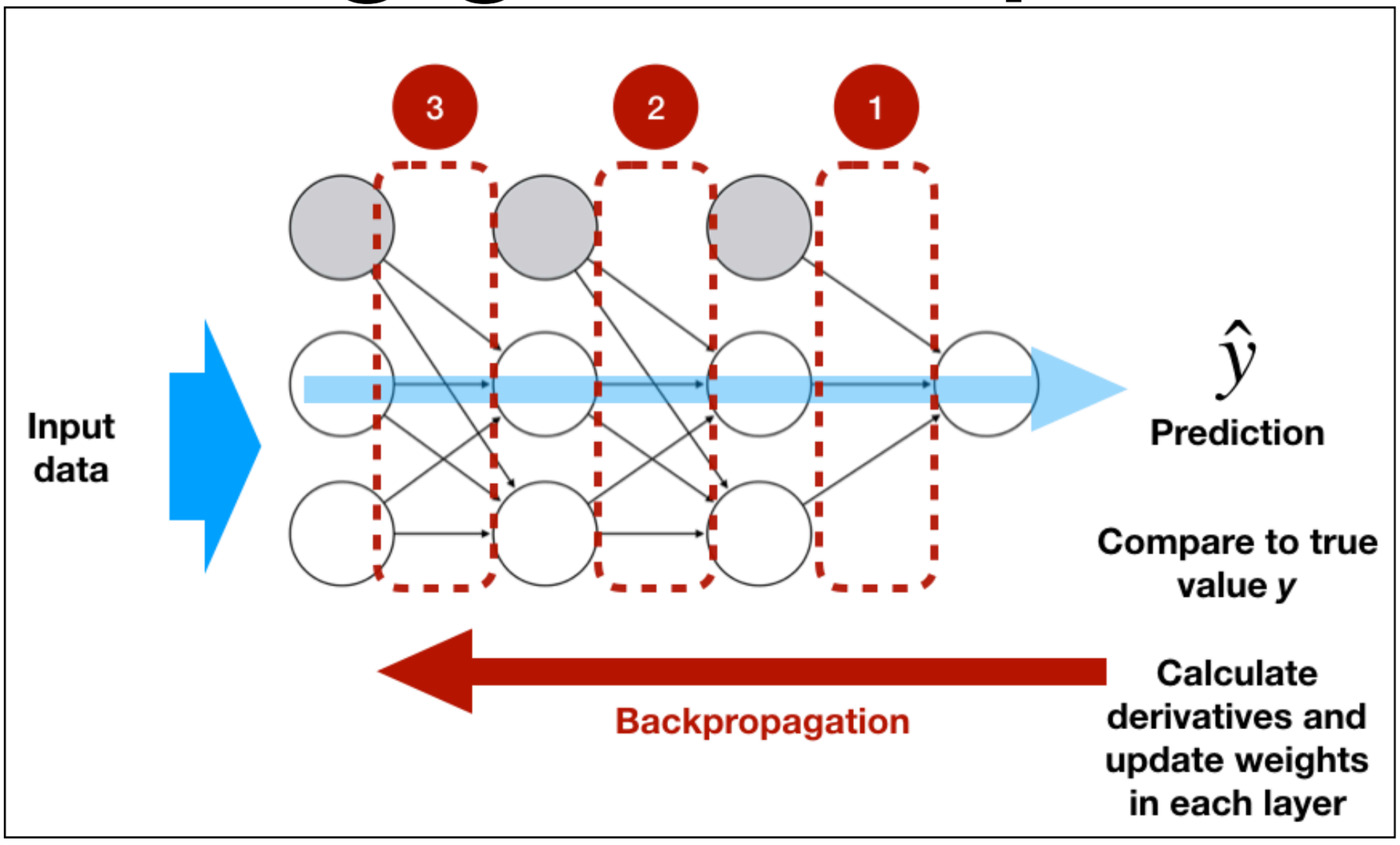
Training challenges - practical

- A lot of inputs, a lot of parameters (~100s of millions is not uncommon)
- Need a **Graphical Processing Unit** (GPU) to train efficiently: 10x speedup compared to CPU
    - GPU wil also speed up the predictions, but you can get away with doing this on CPU if you don’t have a huge dataset to be classified.

Training challenges - optimization

- Need **a lot of hidden layers** to find structure in images on multiple levels:
    - from edges, to textures, to simple shapes, etcc.
- The “error signal”, measured at the end of the network, can become very small (even 0) when traveling back through the network. The earliest layers will not be able to update themselves.
- This is known as the **vanishing gradient** problem.

Vanishing gradient problem



This is a side-effect of the way the backpropagation math works: multiplying many small (<1) numbers.

0.01 0.1 0.4 0.8

Dealing with the vanishing gradient problem

- Fortunately, there are ways to handle that problem:
  - Better random initialization of variables
    - The good news is: no need to worry about initialization details when you use high level frameworks like Keras!
  - ReLU activation function rather than tanh keeps values higher throughout the network.
  - More advanced architectures, such as Residual Neural Networks (ResNets), which skip layers.

These are some of the breakthroughs in the early 00's and 10's, which suddenly allowed us to train deep networks effectively.

Agenda

- Deep Learning introduction
- Extending Logistic Regression into Deep Learning
- Notebook practice with Deep Learning (+ 15min break at 10:30-10:45)
- Plenary notebook wrap-up
- Specialized Deep Learning architectures
- Lunch
- Towards ConvNets
- A simple ConvNet architecture & Transfer Learning
- Notebook practice
- Plenary notebook wrap-up
- Break
- Advanced ConvNets: localization, object detection, instance segmentation
- A ConvNet application at Schiphol Airport
- End

A simple ConvNet

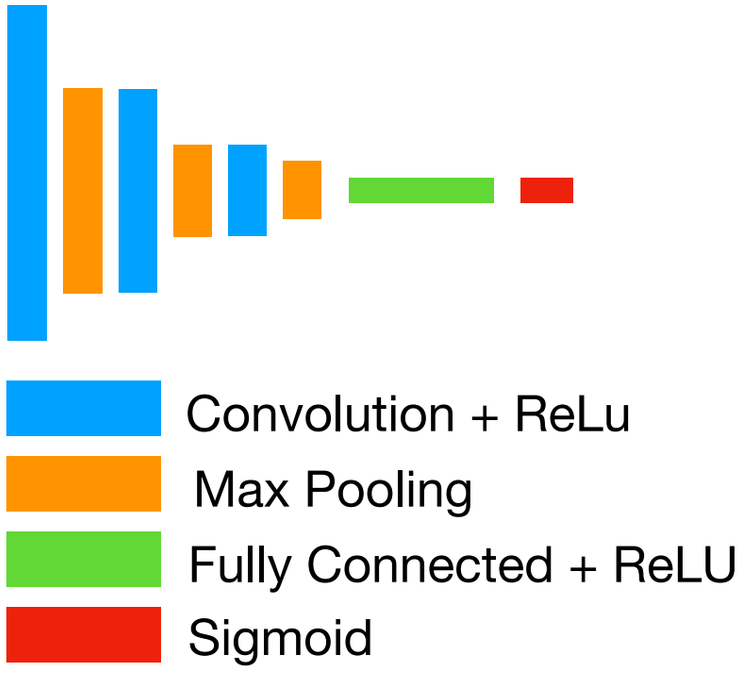
Using the Sequential API in Keras:

```
model = Sequential()
model.add(Conv2D(64, (3, 3),
                 input_shape=(DIM_Y, DIM_X, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(32, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Conv2D(32, (3, 3)))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))

model.add(Flatten())
model.add(Dense(64))
model.add(Activation('relu'))
model.add(Dense(1))
model.add(Activation('sigmoid'))
```



This already works pretty well on simple binary classification problems.

Transfer Learning

- One thing that has really accelerated Deep Learning recently is **Transfer Learning**

- Big networks, with 100-1000 millions of parameters, can take weeks to train on expensive GPU clusters.

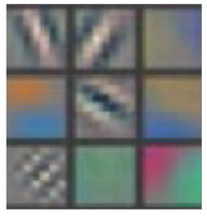
- These networks learn to recognize objects in that particular learning task… i.e. dogs, cats, airplanes, trucks…

  - See ImageNet: http://image-net.org/explore

- But… remember what the first layers learned?

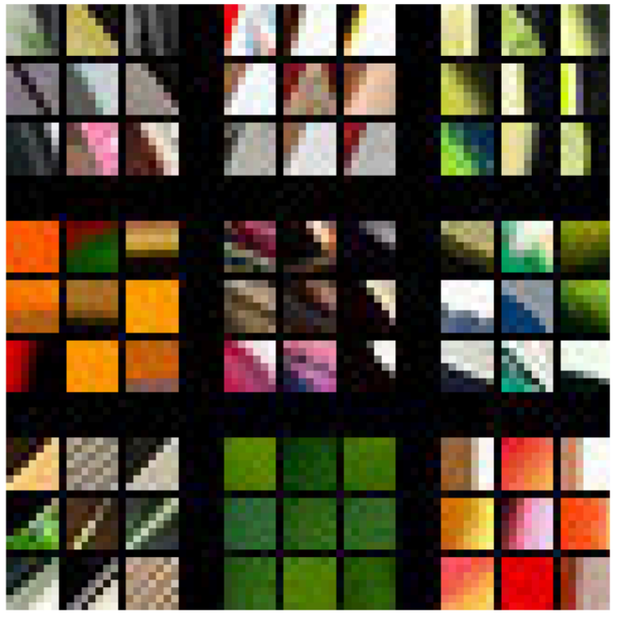
Transferable knowledge

- Edges/textures learned in the earlier layers are generic and apply to a lot of other problems as well!



Layer 1

- Starting each ConvNet from scratch is like re-inventing the wheel each time.

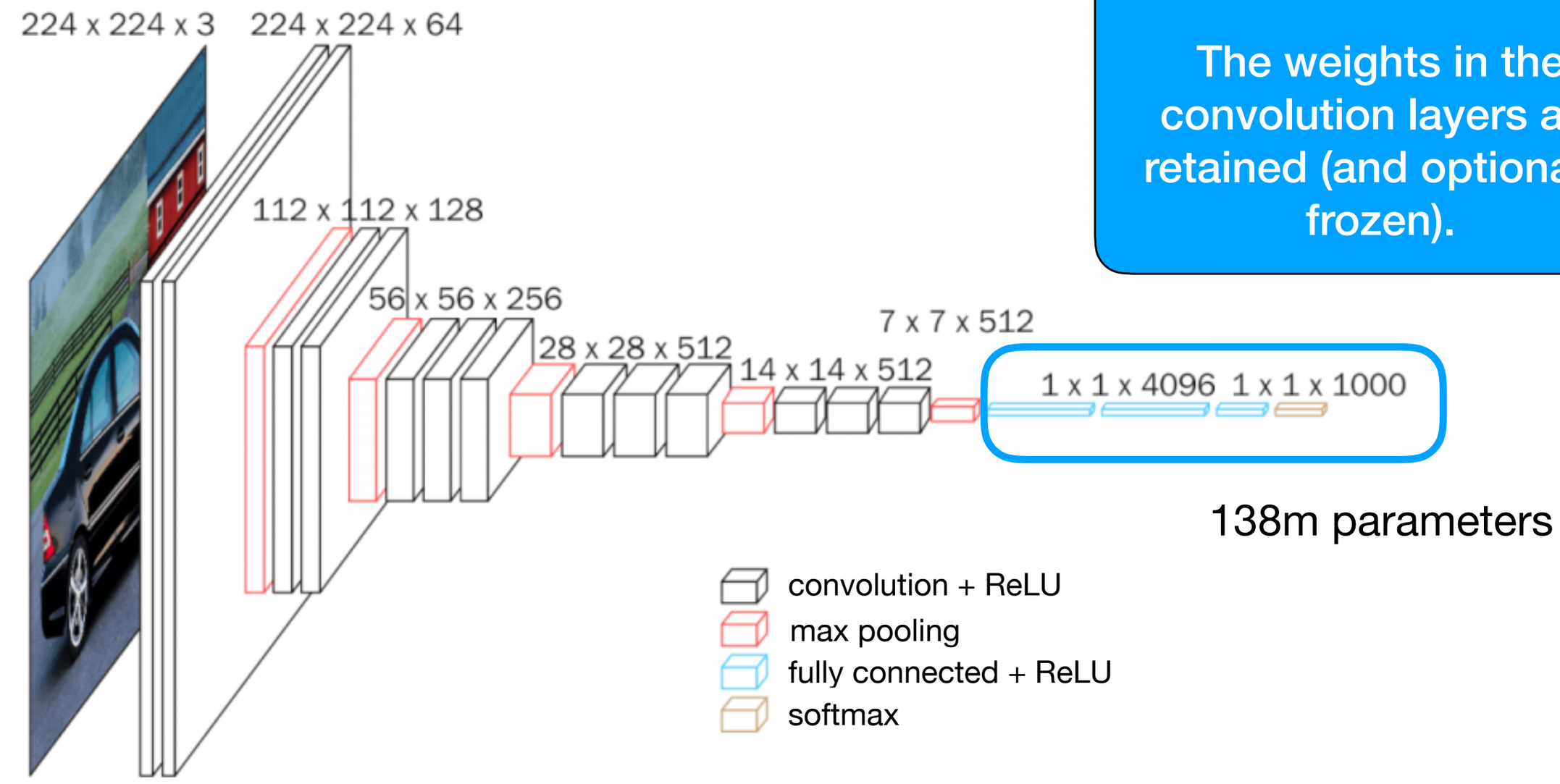


- Can save a *lot* of time to initialize weights from an existing network, instead of initializing randomly.

VGG-16

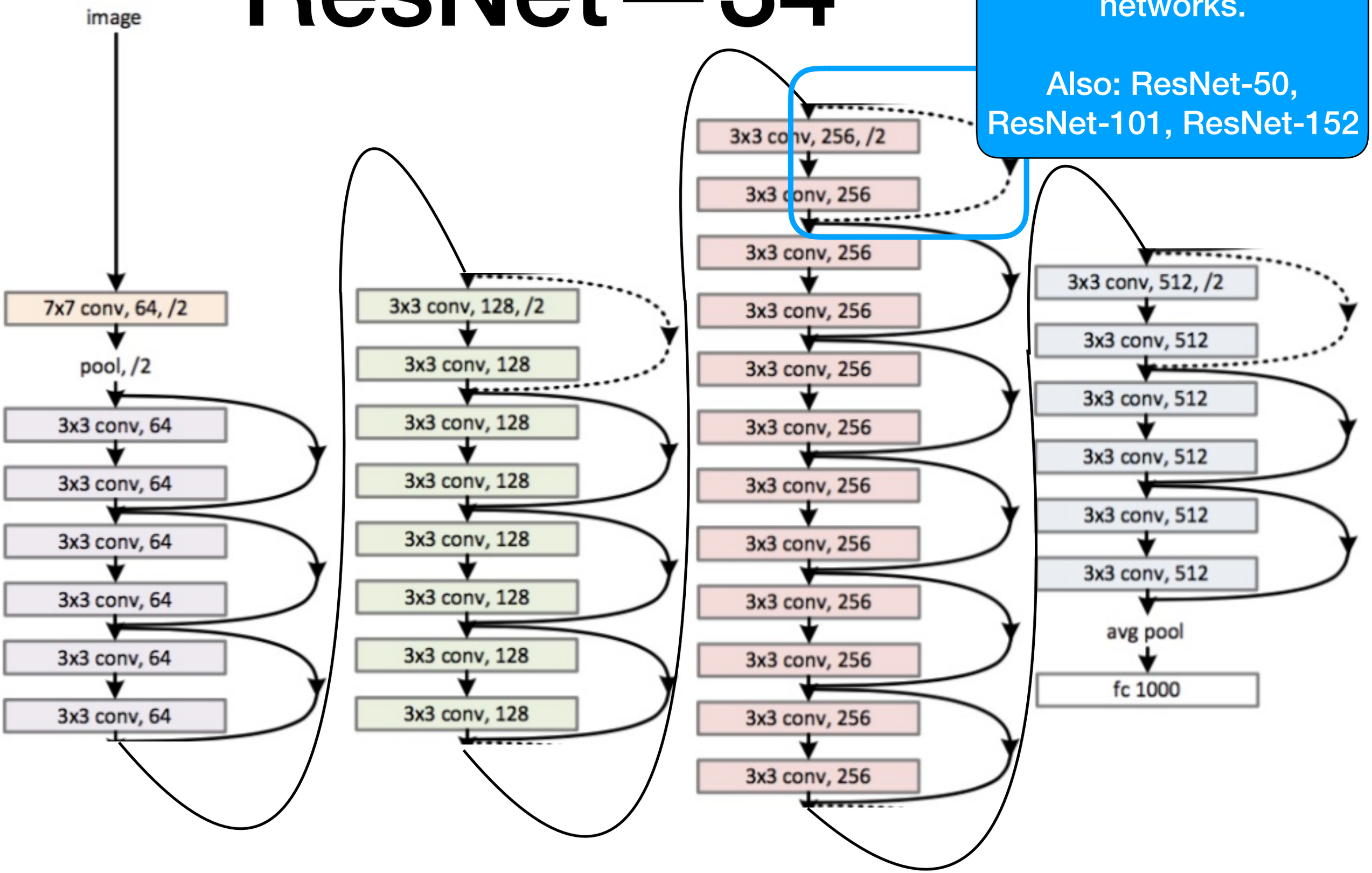
When transfer learning, the fully connected layers are initialized randomly.

The weights in the convolution layers are retained (and optionally frozen).



- Restriction: must use same input size as existing network (in this case a square of 224 x 224).

ResNet—34



These “skip connections” help with training very deep networks.

Also: ResNet-50, ResNet-101, ResNet-152

Network Depth

- Deeper networks usually have better accuracy.
- But:
    - They are harder to train
    - They are more data hungry.
    - They require more computation power. Both during training and during detection.
        - Potential issue for real-time deployments, e.g. self-driving cars.
        - Tradeoff between accuracy and detection speed.

Agenda

- Deep Learning introduction
- Extending Logistic Regression into Deep Learning
- Notebook practice with Deep Learning (+ 15min break at 10:30-10:45)
- Plenary notebook wrap-up
- Specialized Deep Learning architectures
- Lunch
- Towards ConvNets
- A simple ConvNet architecture & Transfer Learning
- Notebook practice
- Plenary notebook wrap-up
- Break
- Advanced ConvNets: localization, object detection, instance segmentation
- A ConvNet application at Schiphol Airport
- End

Notebook Practice

Classification with a Simple ConvNet

https://colab.research.google.com
https://github.com/jvanlier/TIAS_ML_DL

Agenda

- Deep Learning introduction
- Extending Logistic Regression into Deep Learning
- Notebook practice with Deep Learning (+ 15min break at 10:30-10:45)
- Plenary notebook wrap-up
- Specialized Deep Learning architectures
- Lunch
- Towards ConvNets
- A simple ConvNet architecture & Transfer Learning
- Notebook practice
- Plenary notebook wrap-up
- Break
- Advanced ConvNets: localization, object detection, instance segmentation
- A ConvNet application at Schiphol Airport
- End

Why was this relevant?

- This is a bit of a toy problem…

- But: techniques directly apply to any other image classification problem!

Examples of image classification problems - 1

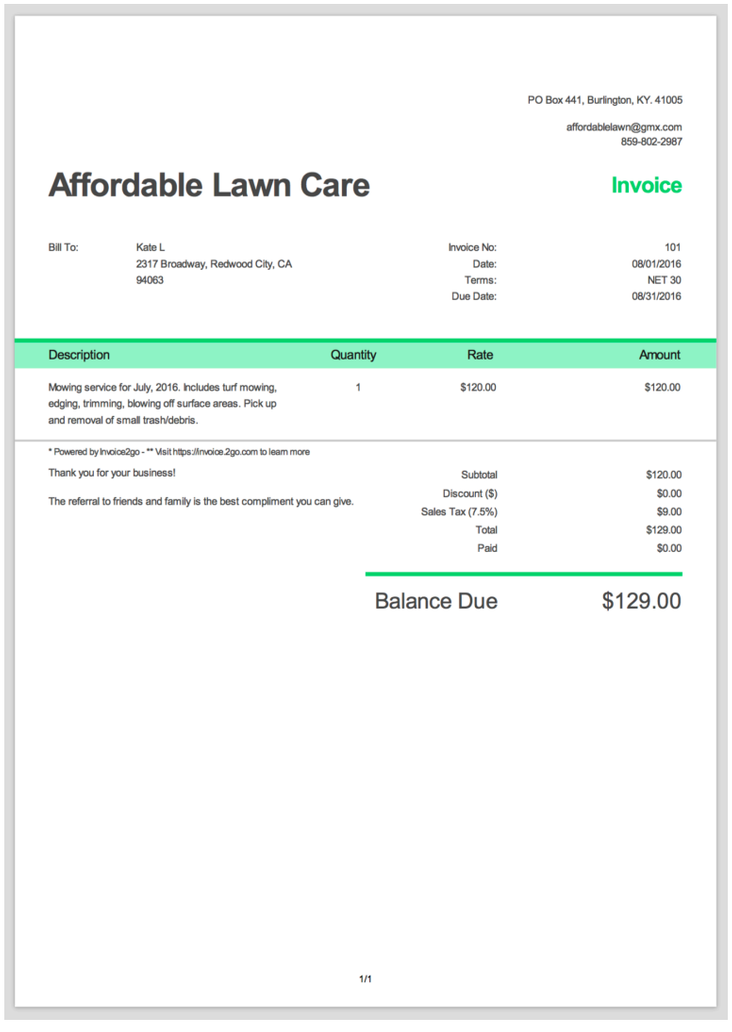
Is the rush-hour lane safe to open? Are there no parked cars, other obstacles, debris?

This is currently done manually.

Examples of image classification problems - 2

Is this piece of ashalt in need of repair?

Examples of image classification problems - 3



PO Box 441, Burlington, KY. 41005

affordablelawn@gmx.com
859-802-2987

Affordable Lawn Care

Invoice

Bill To: Kate L
2317 Broadway, Redwood City, CA
94063

Invoice No: 101
Date: 08/01/2016
Terms: NET 30
Due Date: 08/31/2016

Description	Quantity	Rate	Amount
Mowing service for July, 2016. Includes turf mowing, edging, trimming, blowing off surface areas. Pick up and removal of small trash/debris.	1	$120.00	$120.00

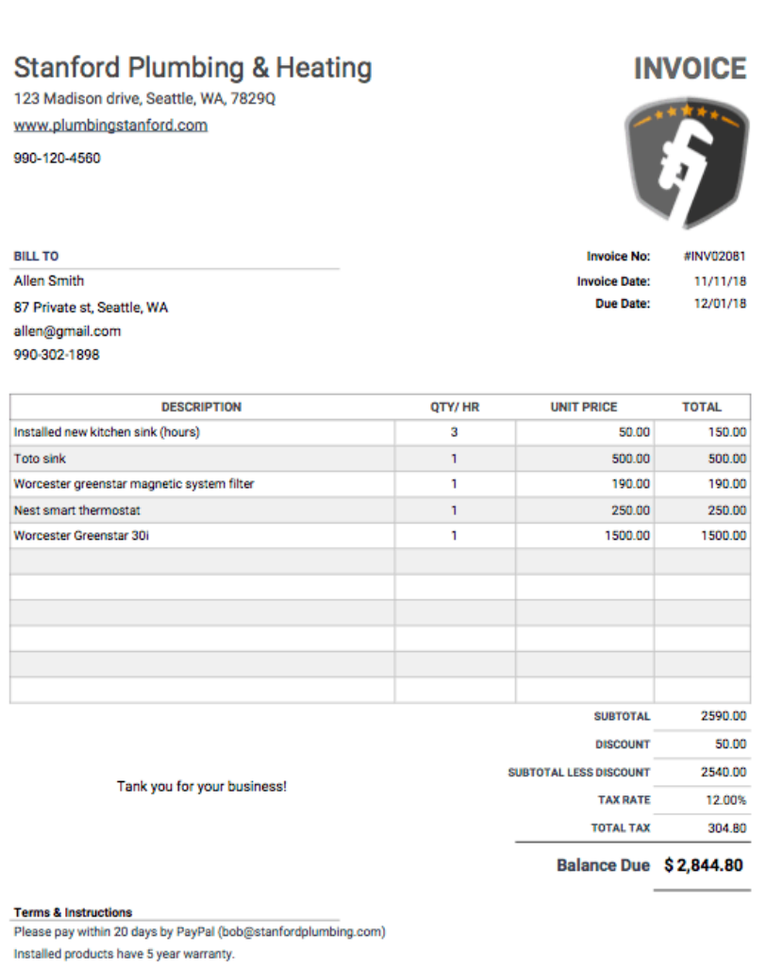
* Powered by Invoice2go - ** Visit https://Invoice.2go.com to learn more

Thank you for your business!

The referral to friends and family is the best compliment you can give.

Subtotal	$120.00
Discount ($)	$0.00
Sales Tax (7.5%)	$9.00
Total	$129.00
Paid	$0.00
Balance Due	**$129.00**

1/1



Stanford Plumbing & Heating
123 Madison drive, Seattle, WA, 7829Q
www.plumbingstanford.com
990-120-4560

INVOICE

Invoice No: #INV02081
Invoice Date: 11/11/18
Due Date: 12/01/18

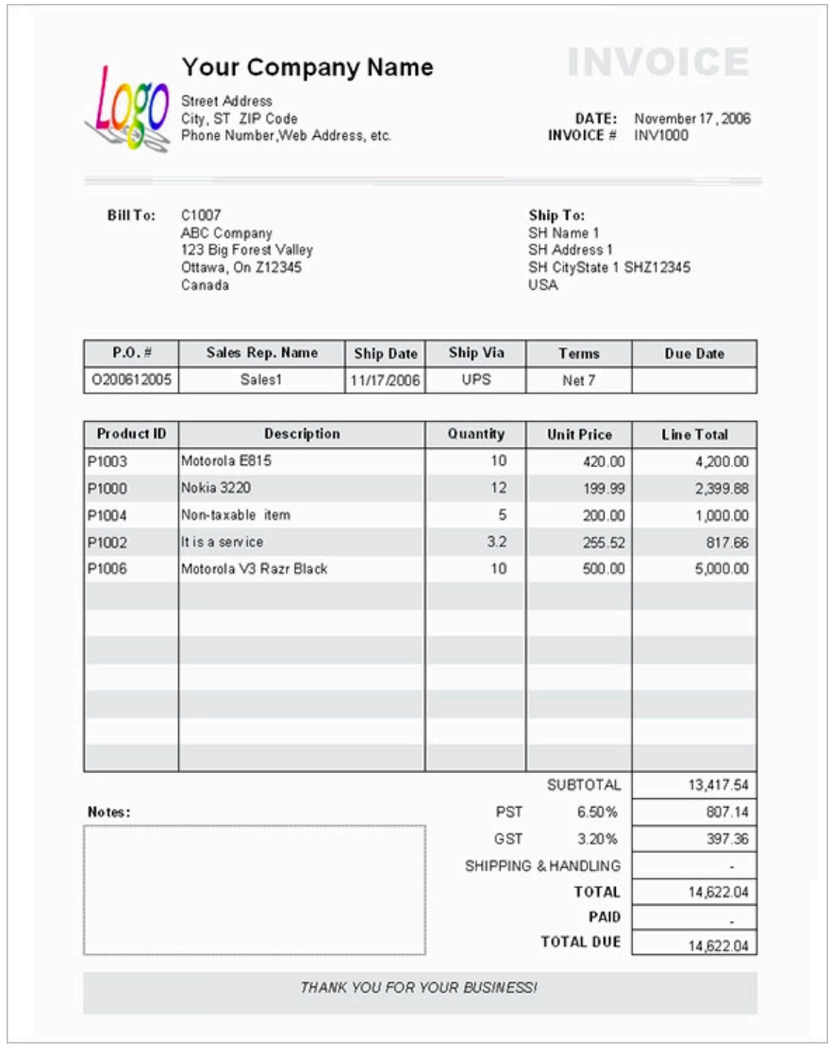
BILL TO
Allen Smith
87 Private st, Seattle, WA
allen@gmail.com
990-302-1898

DESCRIPTION	QTY/ HR	UNIT PRICE	TOTAL
Installed new kitchen sink (hours)	3	50.00	150.00
Toto sink	1	500.00	500.00
Worcester greenstar magnetic system filter	1	190.00	190.00
Nest smart thermostat	1	250.00	250.00
Worcester Greenstar 30i	1	1500.00	1500.00

SUBTOTAL	2590.00
DISCOUNT	50.00
SUBTOTAL LESS DISCOUNT	2540.00
TAX RATE	12.00%
TOTAL TAX	304.80
Balance Due	**$ 2,844.80**

Tank you for your business!

Terms & Instructions
Please pay within 20 days by PayPal (bob@stanfordplumbing.com)
Installed products have 5 year warranty.



Logo

Your Company Name
Street Address
City, ST ZIP Code
Phone Number,Web Address, etc.

INVOICE

DATE: November 17 , 2006
INVOICE # INV1000

Bill To: C1007
ABC Company
123 Big Forest Valley
Ottawa, On Z12345
Canada

Ship To:
SH Name 1
SH Address 1
SH CityState 1 SHZ12345
USA

P.O. #	Sales Rep. Name	Ship Date	Ship Via	Terms	Due Date
O200612005	Sales1	11/17/2006	UPS	Net 7	

Product ID	Description	Quantity	Unit Price	Line Total
P1003	Motorola E815	10	420.00	4,200.00
P1000	Nokia 3220	12	199.99	2,399.88
P1004	Non-taxable item	5	200.00	1,000.00
P1002	It is a service	3.2	255.52	817.66
P1006	Motorola V3 Razr Black	10	500.00	5,000.00

Notes:

SUBTOTAL		13,417.54
PST	6.50%	807.14
GST	3.20%	397.36
SHIPPING & HANDLING		-
TOTAL		14,622.04
PAID		-
TOTAL DUE		14,622.04

THANK YOU FOR YOUR BUSINESS!

Identify vendor —> choose vendor-specific bounding boxes to parse invoice automatically.

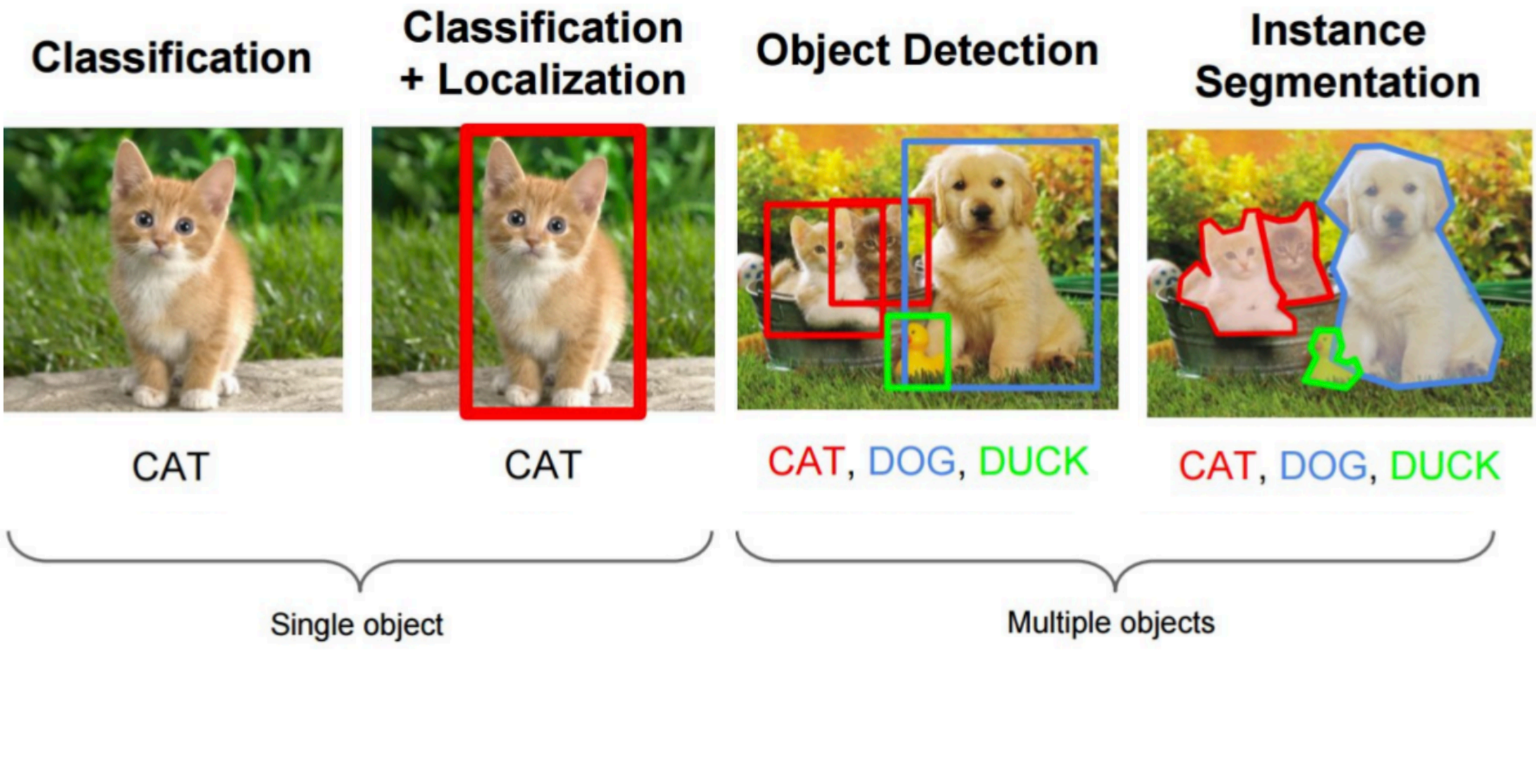
Agenda

- Deep Learning introduction
- Extending Logistic Regression into Deep Learning
- Notebook practice with Deep Learning (+ 15min break at 10:30-10:45)
- Plenary notebook wrap-up
- Specialized Deep Learning architectures
- Lunch
- Towards ConvNets
- A simple ConvNet architecture & Transfer Learning
- Notebook practice
- Plenary notebook wrap-up
- Break
- Advanced ConvNets: localization, object detection, instance segmentation
- A ConvNet application at Schiphol Airport
- End

Agenda

- Deep Learning introduction
- Extending Logistic Regression into Deep Learning
- Notebook practice with Deep Learning (+ 15min break at 10:30-10:45)
- Plenary notebook wrap-up
- Specialized Deep Learning architectures
- Lunch
- Towards ConvNets
- A simple ConvNet architecture & Transfer Learning
- Notebook practice
- Plenary notebook wrap-up
- Break
- Advanced ConvNets: localization, object detection, instance segmentation
- A ConvNet application at Schiphol Airport
- End

More advanced ConvNets



Towards localization

- This is a pretty simple extension: we've seen something like this before!

- Indeed, multitask learning. Just add 4 outputs to the classification ConvNet with ReLU activation functions:
  - xmin
  - ymin
  - width
  - height

- We're now doing both classification and regression in a single network.

Detecting multiple objects

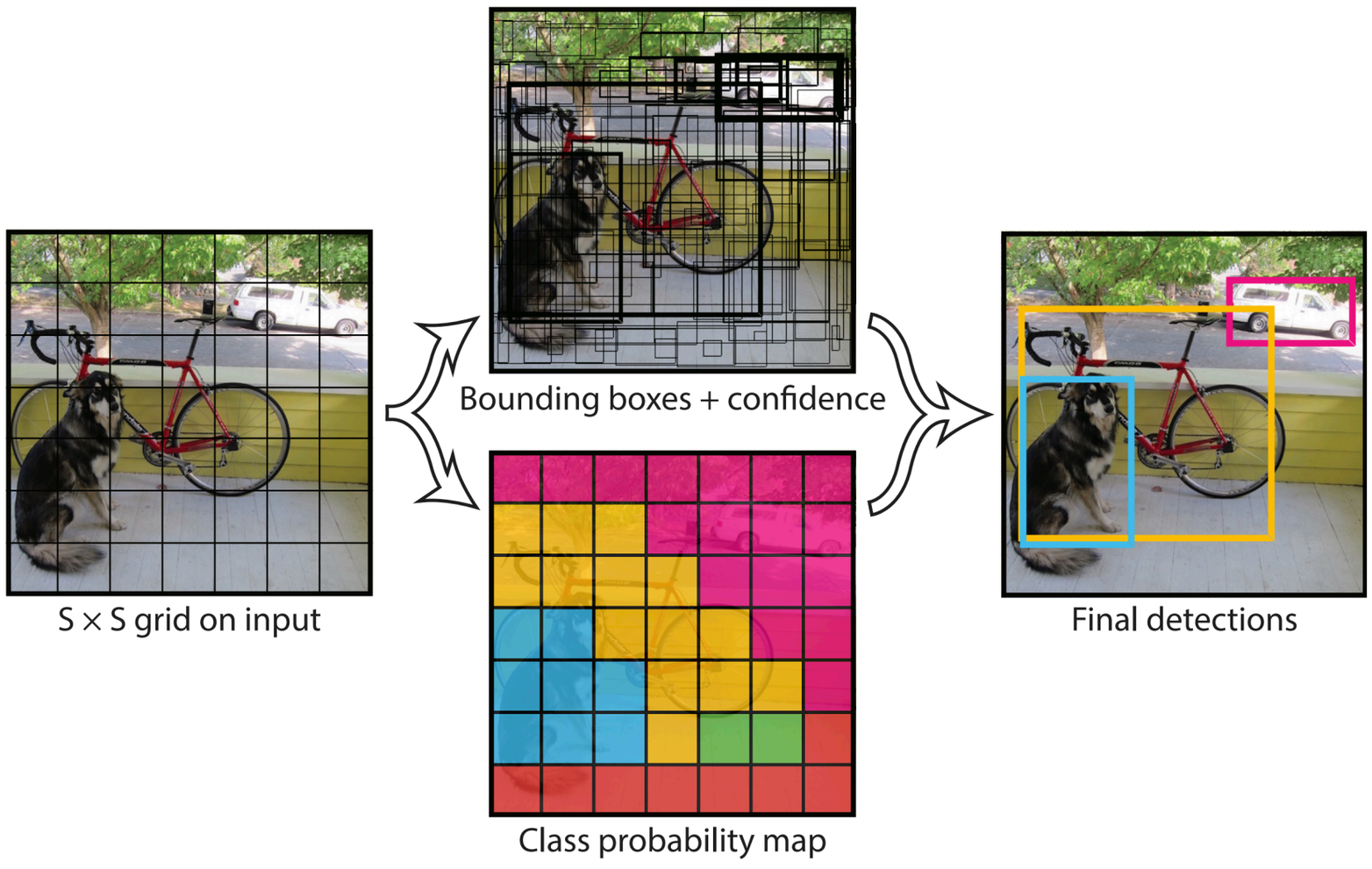
- Naive approach: sliding windows

- Run the CNN multiple times for each image. Then collect the results (e.g. license plate, headlight, grille).

- This is how the earliest Object Detection ConvNets worked, but it was very slow.

YOLO: You Only Look Once

- Only requires a single forward pass through ConvNet to find multiple objects in a single image! Very fast!



- YOLO was the first of this form, SSD (Single Shot Detector) improved on the same concept. SSD is more accurate, but YOLO remains faster.

Alternative approach: two-stage detectors

- Region-based CNN (R-CNN) uses **Selective Search** algorithm to get region proposals, and then proceeds to apply the CNN on each box:

(Fast(er)) R-CNN

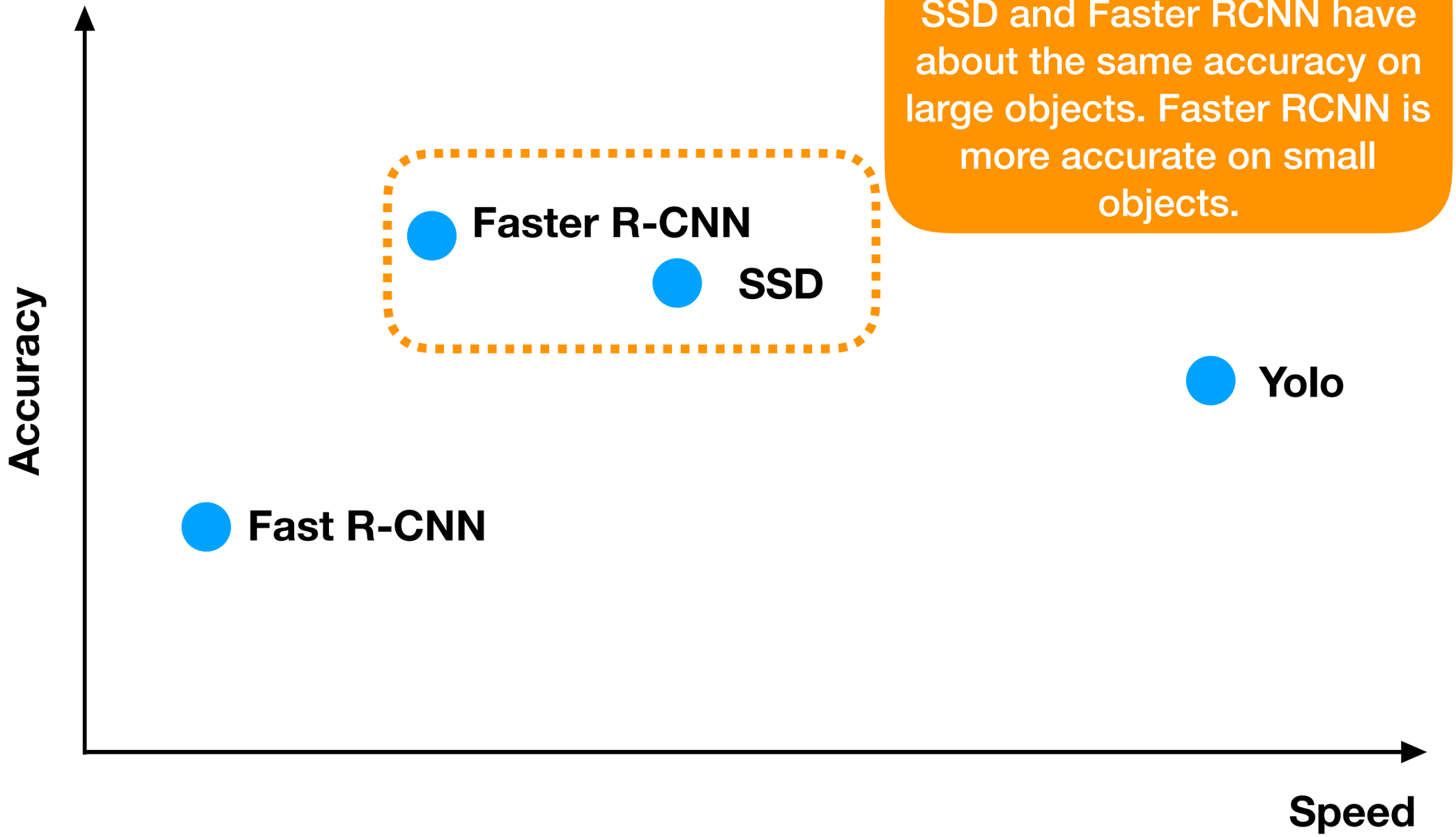
- First version was very slow, but has been sped up a lot in newer iterations:

- Fast R-CNN in 2015
    - Improvements to do only 1 forward pass through CNN.

- Faster R-CNN in 2016

    - No longer uses separate Selective Search algorithm - region proposals are now provided by a Neural Network!

Comparison


Nuance:
SSD and Faster RCNN have about the same accuracy on large objects. Faster RCNN is more accurate on small objects.
Faster R-CNN
SSD
Yolo
Fast R-CNN
Accuracy
Speed

Segmentation

- State of the art research has transitioned from object detection (bounding boxes) into instance segmentation.

Agenda

- Deep Learning introduction
- Extending Logistic Regression into Deep Learning
- Notebook practice with Deep Learning (+ 15min break at 10:30-10:45)
- Plenary notebook wrap-up
- Specialized Deep Learning architectures
- Lunch
- Towards ConvNets
- A simple ConvNet architecture & Transfer Learning
- Notebook practice
- Plenary notebook wrap-up
- Break
- Advanced ConvNets: localization, object detection, instance segmentation
- A ConvNet application at Schiphol Airport
- End

An application of Object Detection at Schiphol airport

Slides in different deck - these can not be shared unfortunately.

Agenda

- Deep Learning introduction
- Extending Logistic Regression into Deep Learning
- Notebook practice with Deep Learning (+ 15min break at 10:30-10:45)
- Plenary notebook wrap-up
- Specialized Deep Learning architectures
- Lunch
- Towards ConvNets
- A simple ConvNet architecture & Transfer Learning
- Notebook practice
- Plenary notebook wrap-up
- Break
- Advanced ConvNets: localization, object detection, instance segmentation
- A ConvNet application at Schiphol Airport
- End

References / further study

- Coursera Deep Learning Specialization by Andrew Ng
    - Good place to start if you'd like to go deeper. Pretty accessible. Also goes into sequence models. Workload 6-8 hours a week for ~ 20 weeks.
    - Bottom-up approach: math first.
- Fast.ai "Practical Deep Learning for Coders"
    - Top-down approach, very little math up front.
    - Does assume basic familiarity with Python programming
- Deep Learning book by Goodfellow, Bengio, Courville
    - Very detailed and math-heavy

Thanks!

Jori van Lier
jori@jvlanalytics.nl

Agenda with times

- 09:15 Deep Learning introduction
- 09:30 Extending Logistic Regression into Deep Learning
- 10:15 Notebook practice with Deep Learning (+ 15min break at 10:30-10:45)
- 11:15 Plenary notebook wrap-up
- 11:30 Specialized Deep Learning architectures
- 12:15 Lunch
- 13:15 Towards ConvNets
- 14:00 A simple ConvNet architecture & Transfer Learning
- 14:15 Notebook practice
- 15:15 Plenary notebook wrap-up
- 15:30 Break
- 15:45 Advanced ConvNets: localization, object detection, instance segmentation
- 16:00 A ConvNet application at Schiphol Airport
- 17:00 End