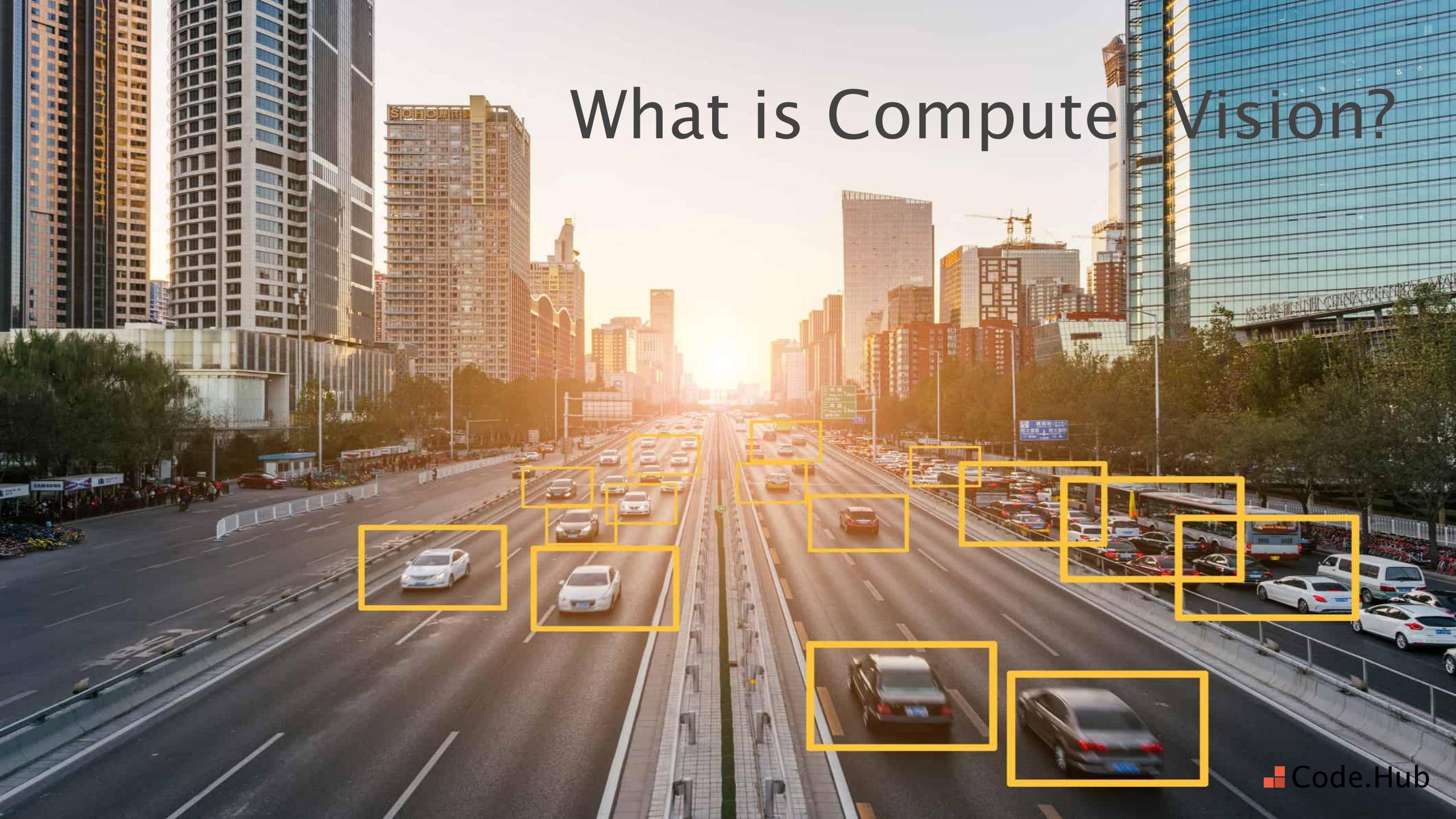


Deep Learning in Computer Vision

Thanos Tagaris

Contents

- **What we'll cover:**
 - intro to Computer Vision
 - Convolutional Neural Networks
 - Interpretability in visual models
 - hands-on with tensorflow/keras
- **Prerequisites:**
 - basic understanding of Machine Learning and Neural Networks
 - basic knowledge of python



What is Computer Vision?

What is Computer Vision?

*Computer Vision is a field of research aiming at developing
human-like vision capabilities for computers.*



high-level understanding of images and video

Most common tasks

- **Classification**

*determine what does the image show
(assumes that the image has a single interpretation)*

- **Segmentation**

divide the image into segments according to its content

- **Object Detection**

detect the different objects inside the image

Note: all of the above are supervised learning tasks

Image Classification



tabby cat



great white shark



lemur

identifies what the image is showing

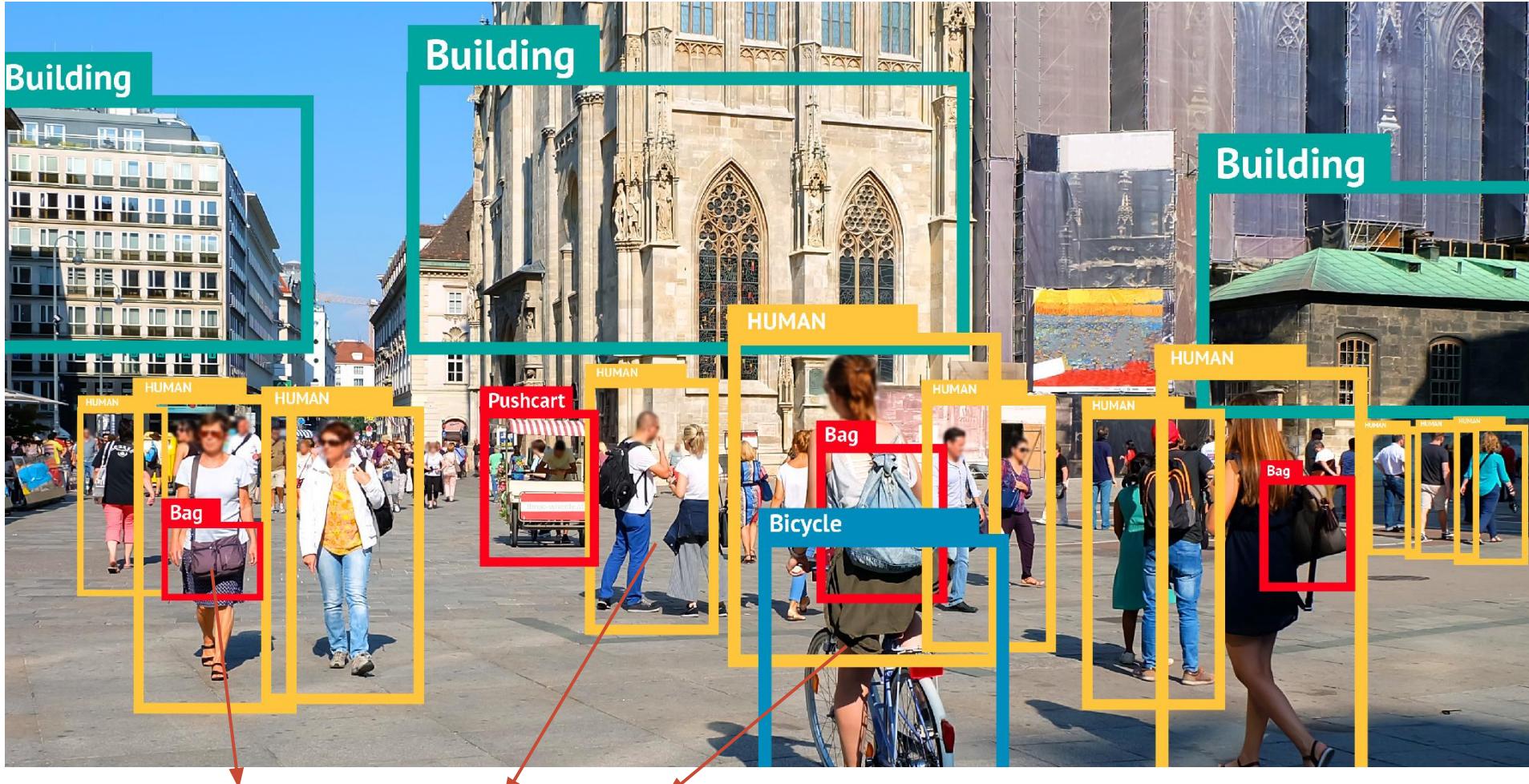
Image Segmentation



identifies the area of the image that is a “road”

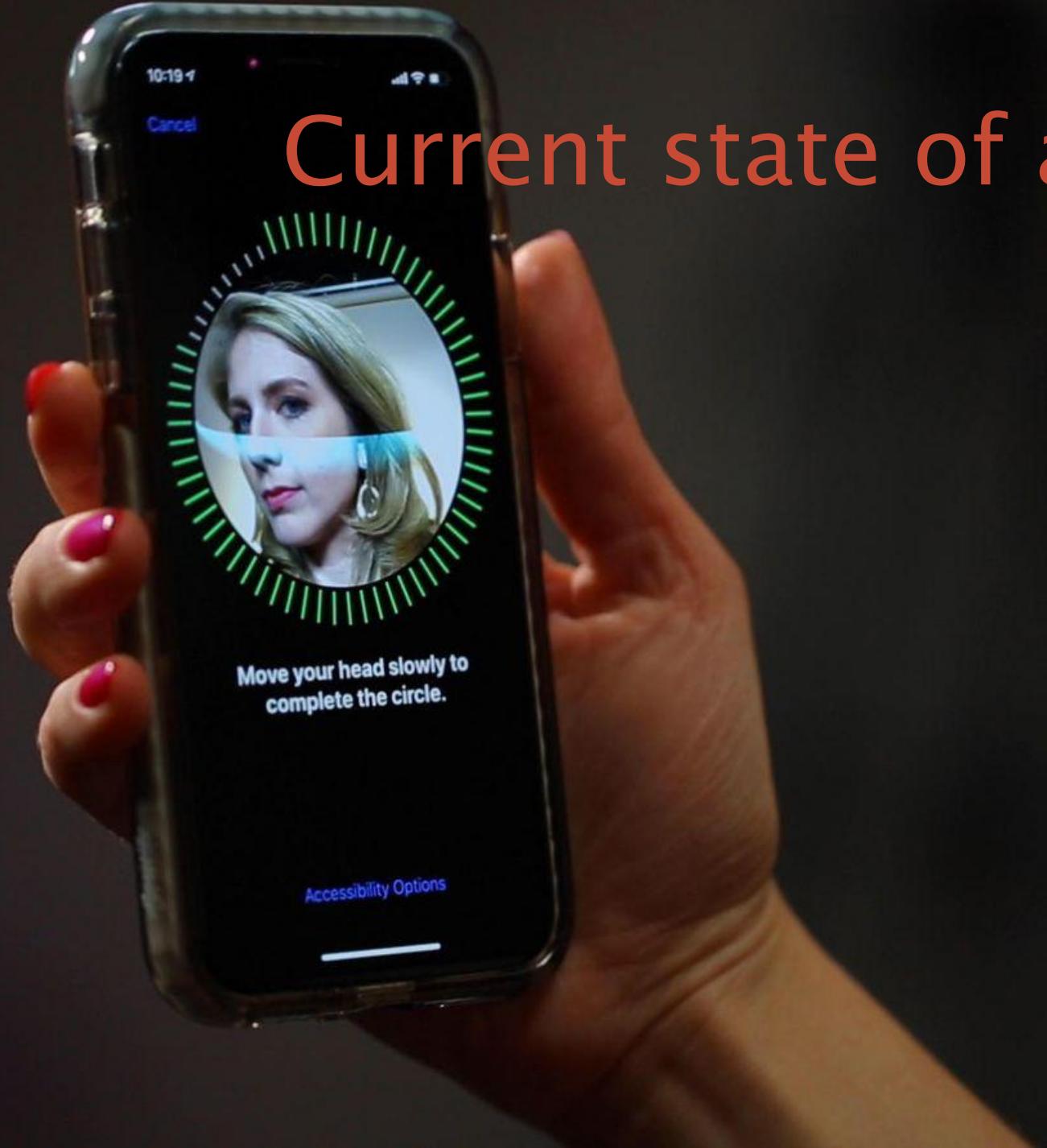


Object Detection



detects the objects in the image

Current state of affairs (1/6)



Current state of affairs (2/6)

← → C 🔒 photos.google.com/search/cat

🔍 cat X

Τρί, 30 Ιουλ

Current state of affairs (3/6)

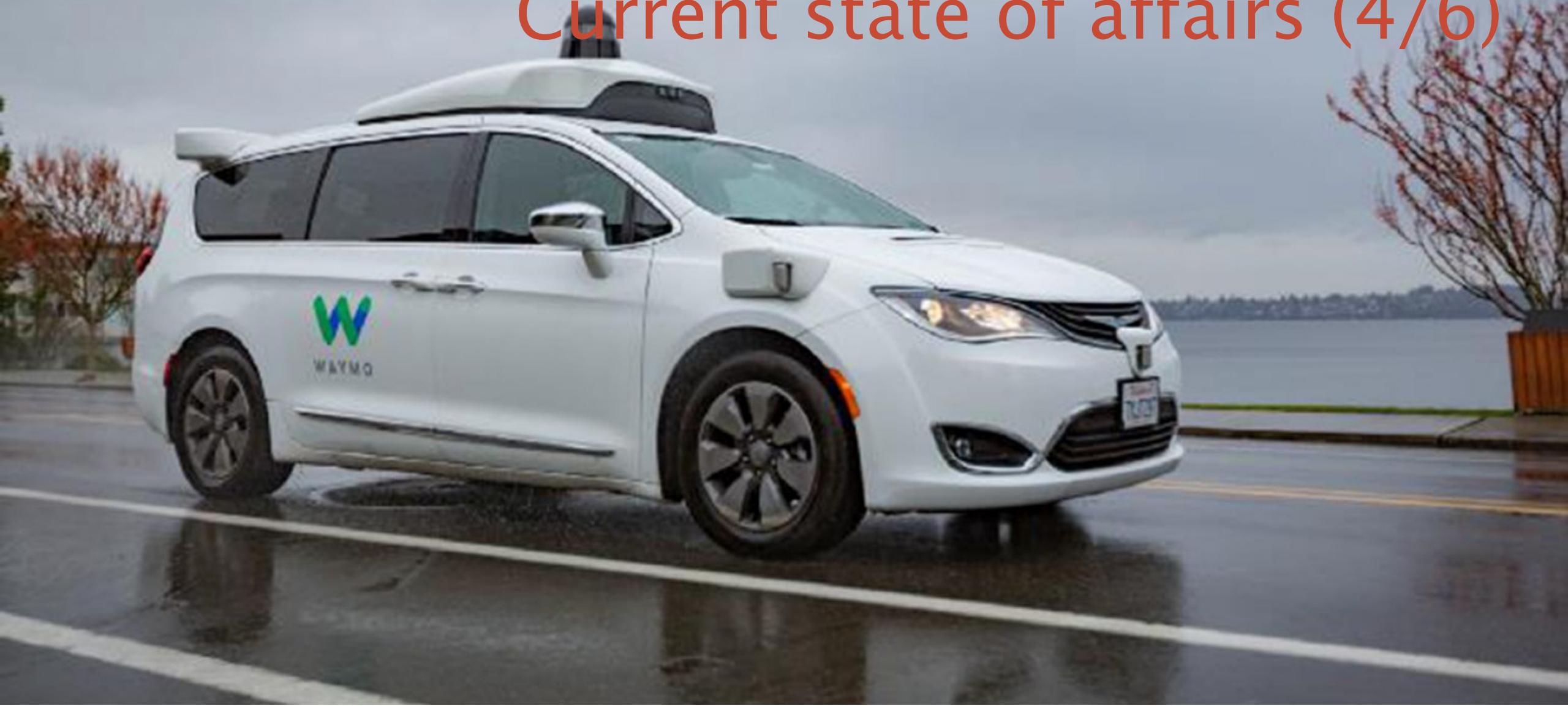
Google AI can now look at your retina and predict the risk of heart disease

After analyzing data from over a quarter million patients, the neural network can predict the patient's age (within a 4-year range), gender, smoking status, blood pressure, body mass index, and risk of cardiovascular disease.



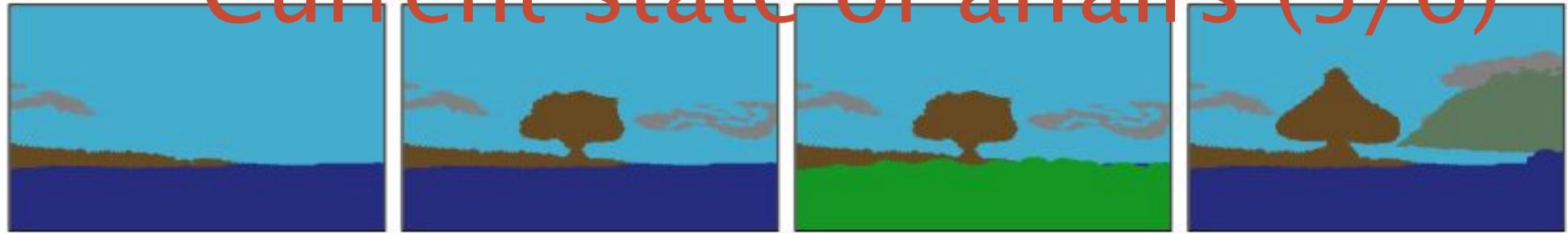
by **Francesca Schiopca** — February 20, 2018 – Updated on February 21, 2018 in Diseases, Science

Current state of affairs (4/6)



Current state of affairs (5/6)

cloud	sky
tree	mountain
sea	grass



Semantic Manipulation Using Segmentation Maps



Stylization using Guide Images

Current state of affairs (6/6)



Challenges in image recognition

What makes a cat, a cat?

- its tail?
- its paws?
- its ears?
- its eyes?
- its face?



Challenges in image recognition

Challenges:

- Hard to quantify what actually **makes** a “cat”
- Do all cats look the **same**?
- Are all cats located in the same **position** in the image?
- Are all cats located in the same **context**?

Consequently:

- Hard to approach image-related tasks with **traditional programming** (e.g. rule-based systems)
- Need for more advanced **Machine Learning** algorithms.

Behind the hood...

... Machine Learning

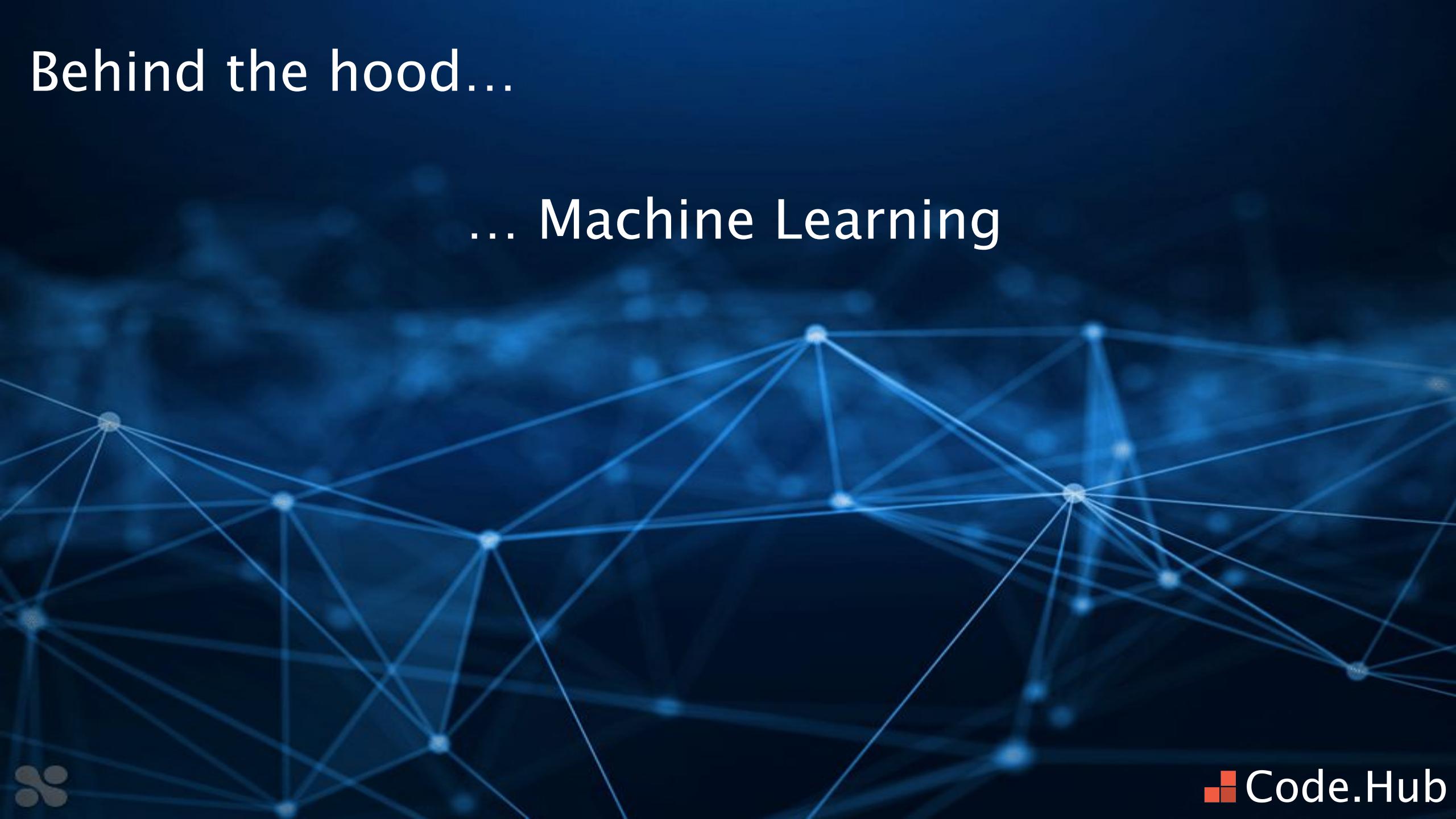
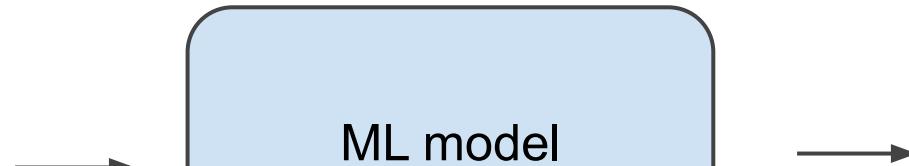


Image classification

Cats vs Dogs



labelled image
dataset



ML model

cat
cat
cat
dog
dog
cat
cat
cat
dog
dog

predict the class
labels

Demystifying ML: caveats

- Key to success: **data**
 - Require lots of data → the more the better
 - Data needs to be manually annotated → very costly
 - Quality of data is very important → machine bias
- **Interpretability** is an issue
 - More complex algorithms (e.g. Deep Neural Networks) are considered as “black boxes”.
 - More difficult to assess performance.
 - Industries are slow to adopt.
- In most cases they **can't adapt** over time.
- Not easy to insert **domain knowledge** in ML pipeline.

Case study

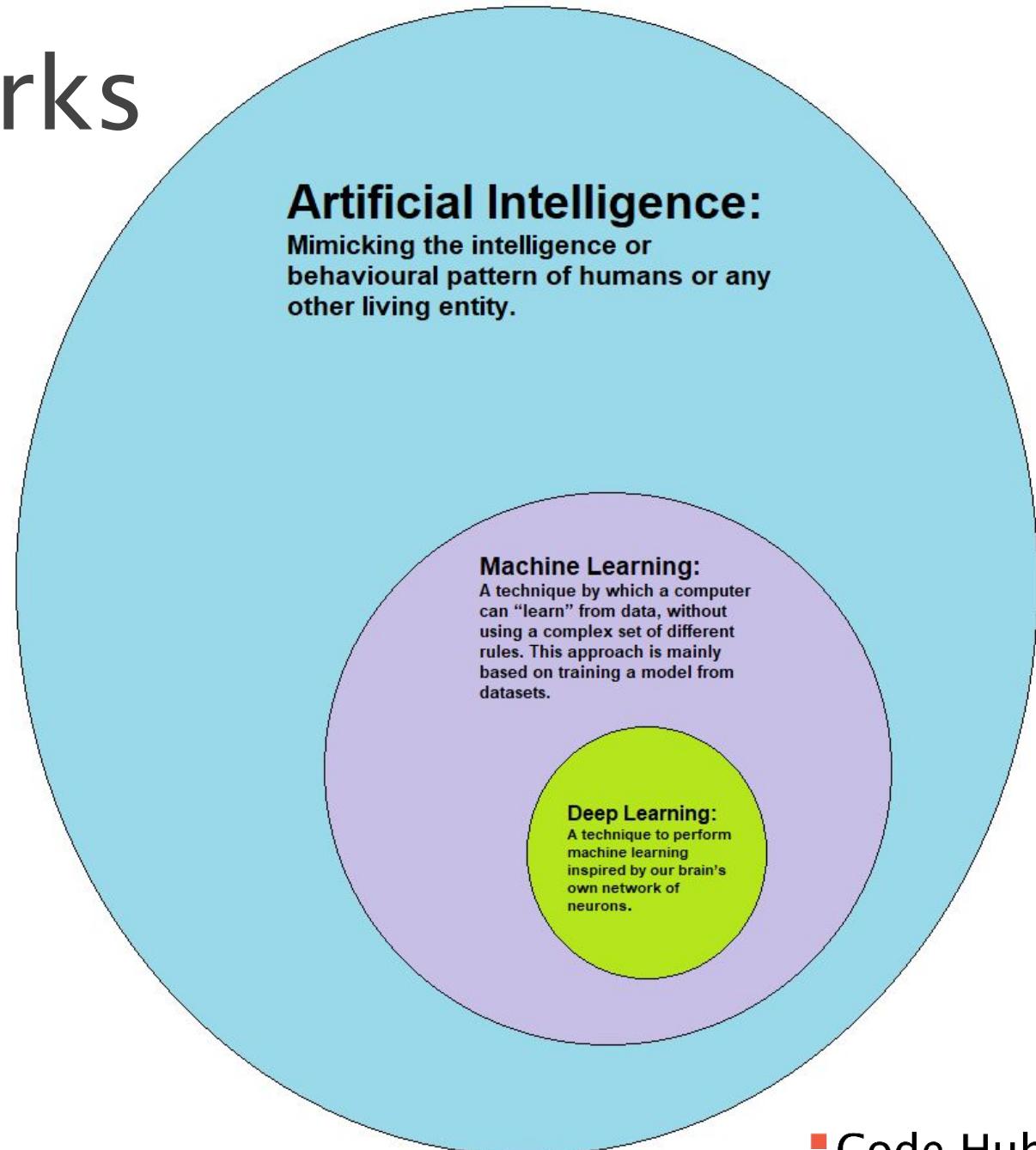
Handwritten digit classification with logistic regression

Deep Learning...

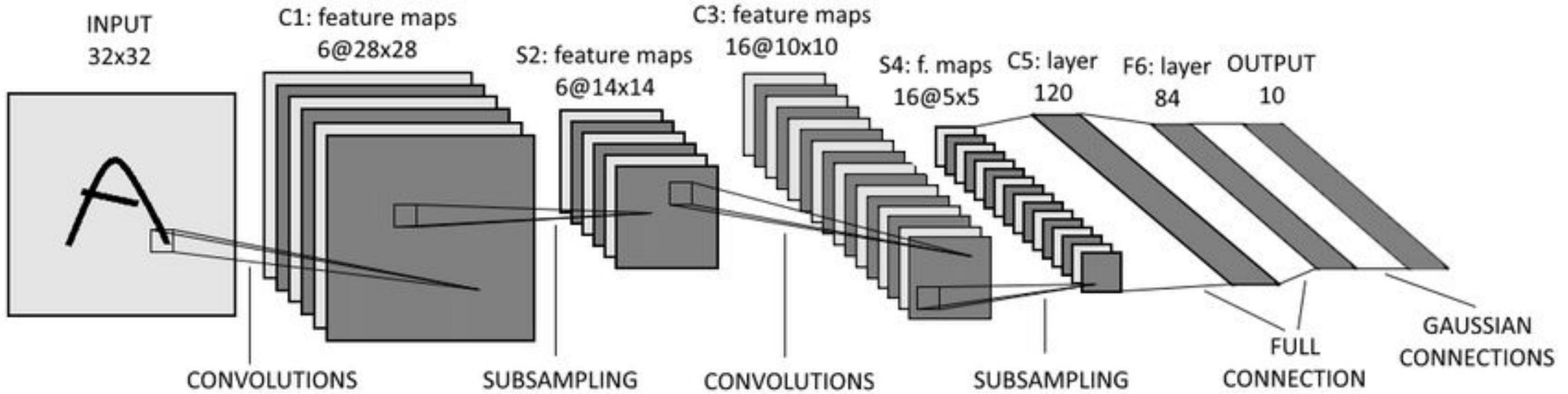


Deep Neural Networks

- Neural Networks with lots of hidden layers.
- Can extract high-level features from the data.

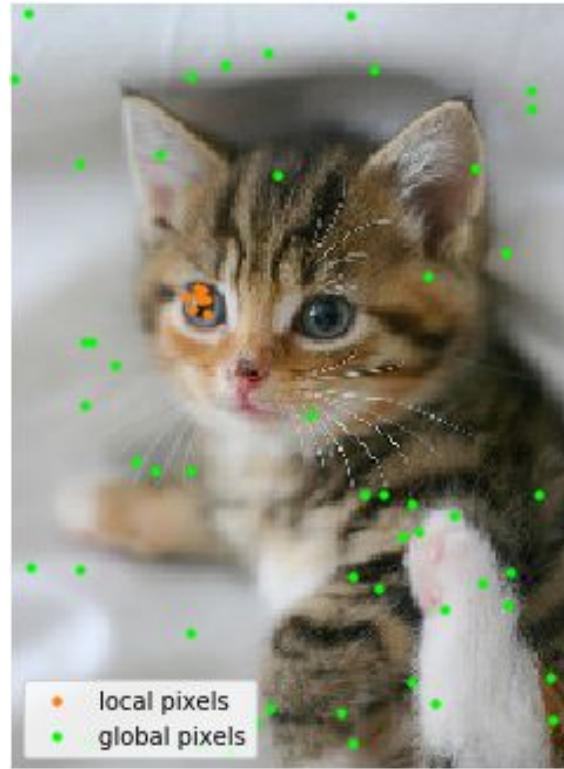


Convolutional Neural Networks

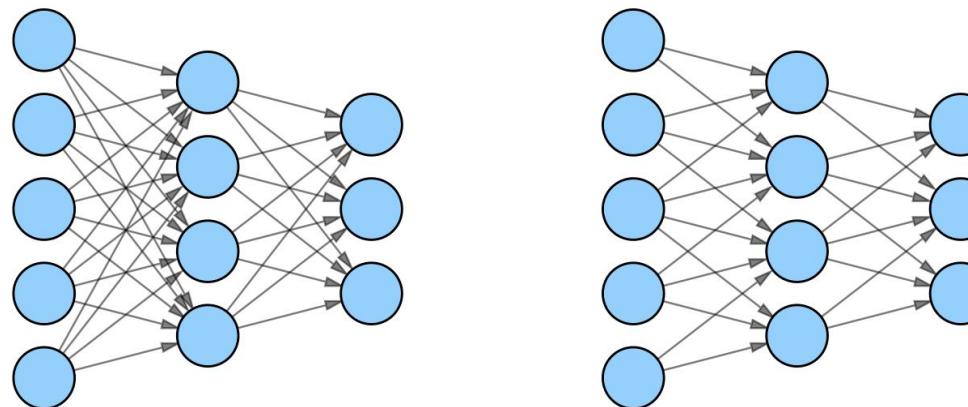


- Input typically represented in **2D** (natural orientation of image)
- Exploit concepts like **sparse connectivity** and **weight sharing**.
- Work very well with images.
- Extract **high-level spatial features**.

Sparse Connectivity: Intuition

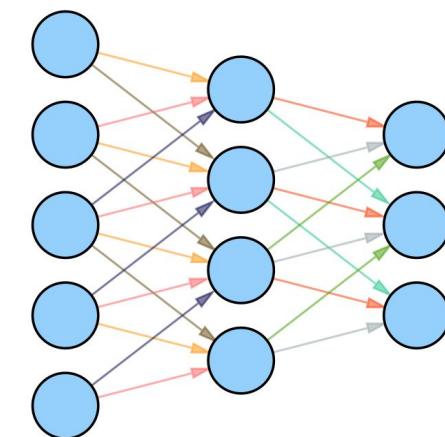
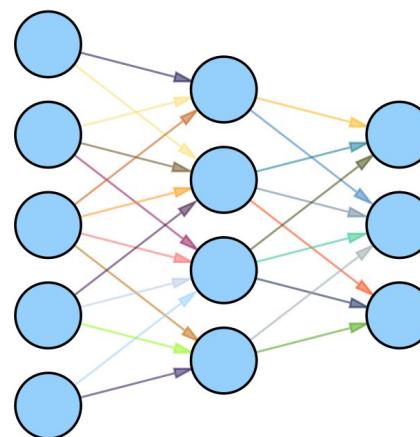


- Pixels exhibit strong **local** semantic correlation
- More important to look at local areas rather than global ones
- Local features can be captured by **locally (sparsely) connected** Neural Networks



Weight Sharing: Intuition

- Objects (e.g. cat's eye) can appear in **various locations** in an image.
- It makes sense for a Network to be able to detect the **same features in any location** of the image.
- This is possible through **weight sharing**.



Convolution Layer

- **Operation:** 2D convolution between **input** and a **kernel**
- **Parameters:**
 - **kernel's** weights are all trainable
 - bias (tied/untied) [optional]
- **Hyperparameters:**
 - kernel size (here 3x3)
 - strides (here 1x1)
 - initialization
 - padding (here “valid”)
 - etc.

1 <small>$\times 1$</small>	1 <small>$\times 0$</small>	1 <small>$\times 1$</small>	0	0
0 <small>$\times 0$</small>	1 <small>$\times 1$</small>	1 <small>$\times 0$</small>	1	0
0 <small>$\times 1$</small>	0 <small>$\times 0$</small>	1 <small>$\times 1$</small>	1	1
0	0	1	1	0
0	1	1	0	0

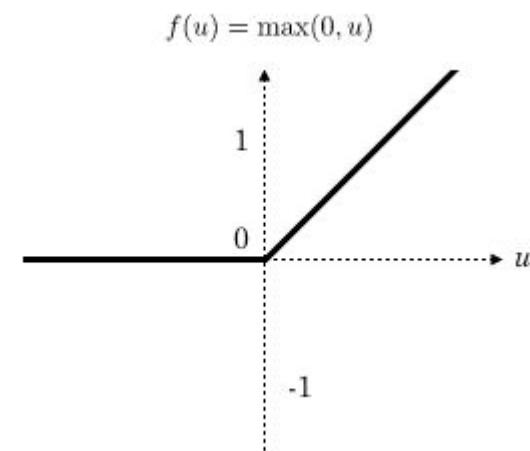
Image

4		

Convolved Feature

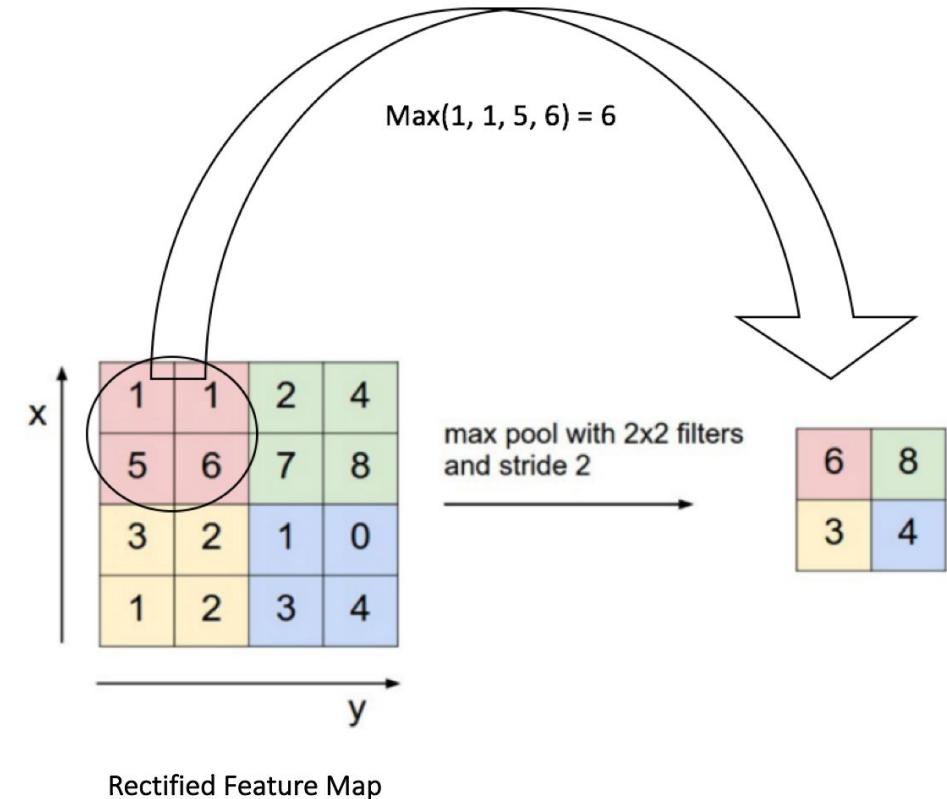
Activation Function

- Convolution layers perform an **affine** transformation
- A network consisting of only conv layers can't model non-linear functions
- Activation functions are **non-linear** functions typically applied after linear layers.
- Most popular activation functions:
 - **Rectified Linear Unit (ReLU)**
 - Other functions from the ReLU family
(e.g. PReLU, leaky ReLU, elu, ...)
 - Saturating ones (e.g. sigmoid, tanh, ...)



Pooling Layer

- **Operation:** 2D aggregation in input
- **Parameters:**
 - none
- **Hyperparameters:**
 - operation (here “max”)
 - size (here 2x2)
 - stride (here 2x2)
 - padding (here “valid”)
- **Benefits:**
 - reduces the spatial dimensions (less parameters for subsequent layers)
 - makes output invariant to small changes in input
 - makes output equivariant to changes in scale



Case study

Handwritten digit classification with CNN

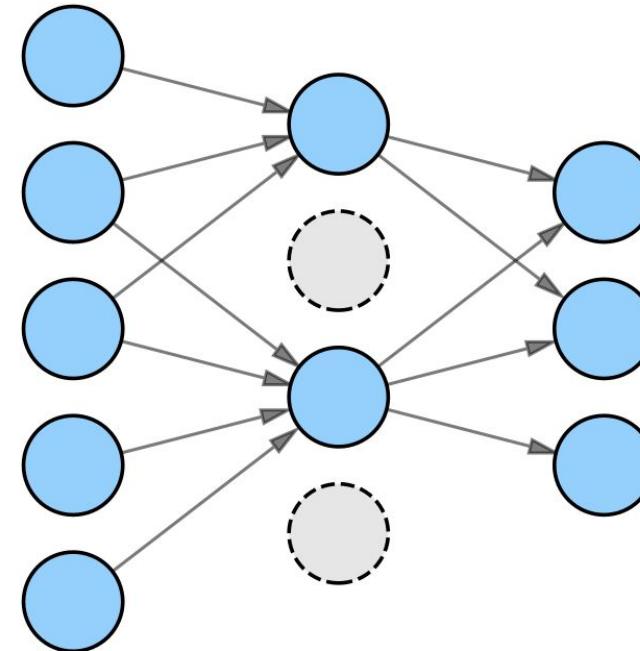
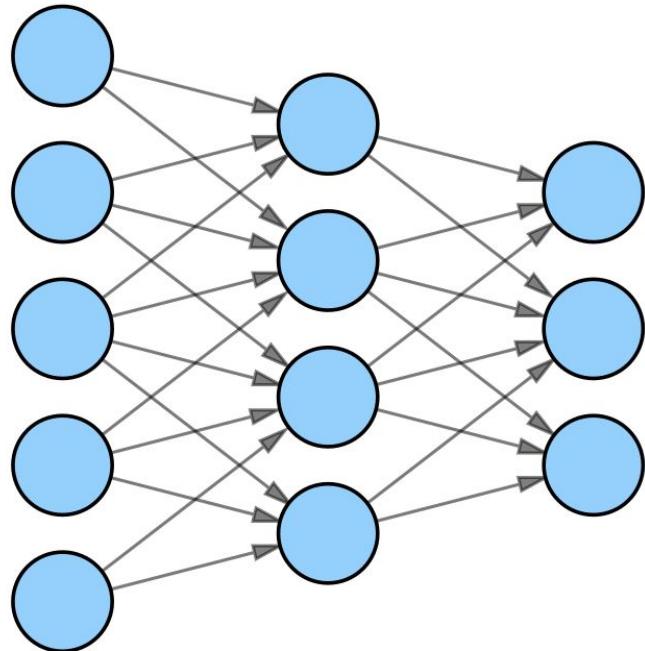
Improvements...

... on the architecture and the training procedure



Dropout

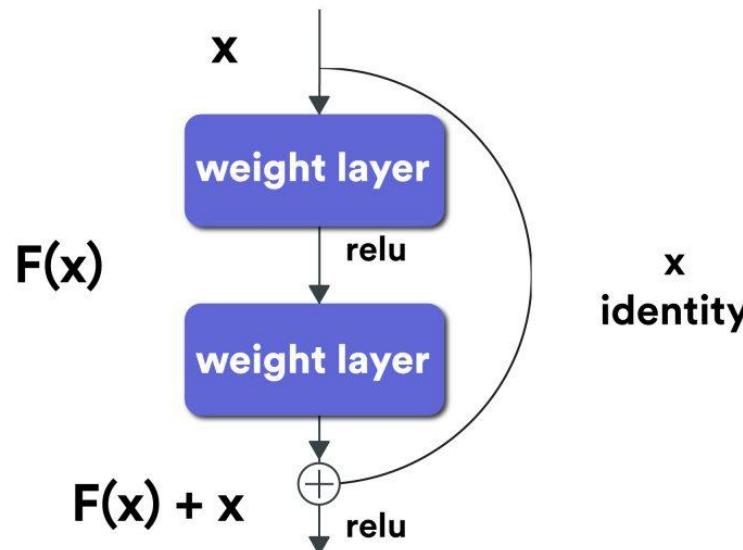
- Most popular **regularization** technique for Neural Networks
- Works by randomly **removing** some activations from a layer at each iteration.



- Reduces interdependence between specific neurons.
- In turn, this **reduces overfitting**.

Skip Connections

- The networks we've seen up till now process the data **sequentially**
- Idea: disrupt this sequential flow by allowing data (and consequently gradients) to **skip** certain layers
- Can be implemented two ways: **addition** (e.g. ResNet), **concatenation** (e.g. DenseNet)



- Can help with the vanishing/exploding gradient problem of Deep Neural Nets

Batch Normalization

- Technique for **increasing training speed and stability**, while **reducing overfitting**.
- **Normalizes** the input of a layer by batch statistics computed during training.
- Then **re-scales** and **re-shifts** the normalized input via two trainable parameters.
- Made way for different intra-layer normalization techniques (e.g. *layer norm*, *group norm*, *instance norm*)

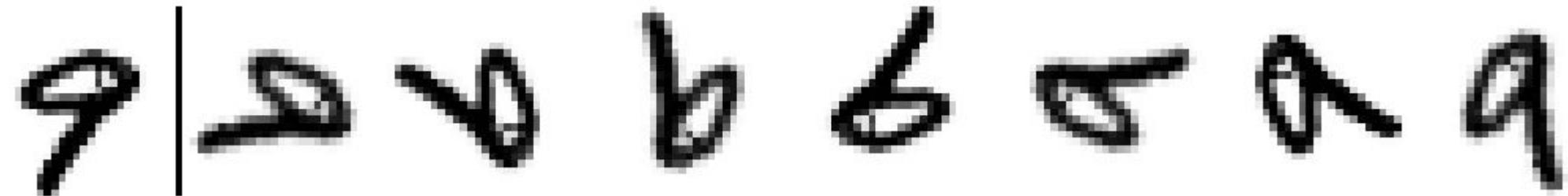
Data Augmentation

- More data helps
- Idea: artificially create more images
- If done properly, these will **increase the network's performance**
- These images will have the **same semantic content** as the originals, but be regarded as completely different images by the network



Data Augmentation - caveats

- Too much augmentation can **alter the semantic content** in an image.



- This can **add noise** to the dataset and **hurt performance**.
- Augmentation scheme needs to be selected **ad-hoc** for each application.

Transfer Learning

- Models trained on **more data** perform better
- Some problems **don't have enough data**
- Idea: train a model on a task that has **lots of data** (let's call this **A**, e.g. *ImageNet*) and then use the knowledge it extracted on another dataset that has **fewer data** (**B**).
- This is typically done by (sometimes partially) continuing the training of a model trained on **A**, on data from **B**.
- This usually leads to a **better performance** on task **B**.

Interpretability in visual models

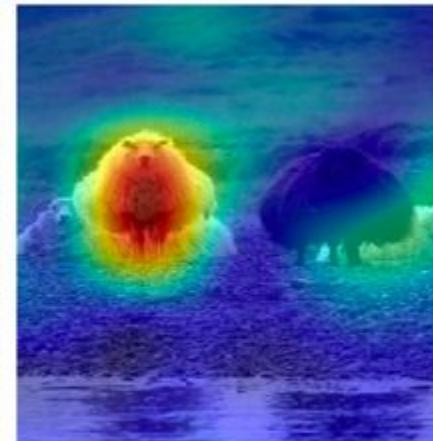
- What is **interpretability** in Machine Learning?
 - the ability to predict a network's output?
 - the ability to “export” the network's logic into a set of rules?
 - the ability to explain the network's prediction?
- We'll equate **interpretability** with **explainability**
 - it is the best we can achieve in practice
 - it is enough for most commercial applications

Saliency map

- What form do explanations come in visual models?



(a) Sheep - 26%, Cow - 17%



(b) Importance map of 'sheep'



(c) Importance map of 'cow'

- Image with the same dimensions as the original which shows the **contribution** of each pixel towards the final prediction.

Visual Interpretability Techniques

- Hot research topic
- A lot of techniques for explaining the predictions of a CNN
- Most common:

Grad-CAM





Thank you

Slides and notebook available at:

github.com/djib2011/material/open-conf-2020

TensorFlow tutorial:

github.com/djib2011/tensorflow-df-tutorial