UPPSALA
UNIVERSITET

Machine learning, big data and artificial intelligence – Block 5

Måns Magnusson
Department of Statistics, Uppsala University

December 2020

This week's lecture

- The Mini-Project and Master Thesis Projects
- Convolutional Neural Networks
- Transfer Learning

Mini-project

- Time to start think about the project.

Mini-project

- Time to start think about the project.
- Supervised problem of choice on real data.

UPPSALA UNIVERSITET

Mini-project

- Time to start think about the project.
- Supervised problem of choice on real data.
- 2-3 students.

Mini-project

- Time to start think about the project.
- Supervised problem of choice on real data.
- 2-3 students.
- Supply a project proposal of data and problem at the end of *16th of December 23.59*.
- *Hint!* Submit page 1-1.5 of the project as project proposal.
- Deadline is after all lectures on supervised learning

Mini-project

- Time to start think about the project.
- Supervised problem of choice on real data.
- 2-3 students.
- Supply a project proposal of data and problem at the end of *16th of December 23.59*.
- *Hint!* Submit page 1-1.5 of the project as project proposal.
- Deadline is after all lectures on supervised learning
- Feel free to combine it with your master thesis project.

UPPSALA UNIVERSITET

Mini-project

- Time to start think about the project.
- Supervised problem of choice on real data.
- 2-3 students.
- Supply a project proposal of data and problem at the end of *16th of December 23.59*.
- *Hint!* Submit page 1-1.5 of the project as project proposal.
- Deadline is after all lectures on supervised learning
- Feel free to combine it with your master thesis project.
- Check with me if you have questions.
- The project should result in a 4 page report (PDF) using the ICML LaTeX template.

Master thesis proposals

- Stochastic EM, probabilistic programming and Topic Models.

Master thesis proposals

- Stochastic EM, probabilistic programming and Topic Models.
- BERT for article segmentation/reconstruction.

Master thesis proposals

- Stochastic EM, probabilistic programming and Topic Models.
- BERT for article segmentation/reconstruction.
- CNN for newspaper segmentation.

UPPSALA UNIVERSITET

Master thesis proposals

- Stochastic EM, probabilistic programming and Topic Models.
- BERT for article segmentation/reconstruction.
- CNN for newspaper segmentation.
- BERT for legal predictions.

UPPSALA UNIVERSITET

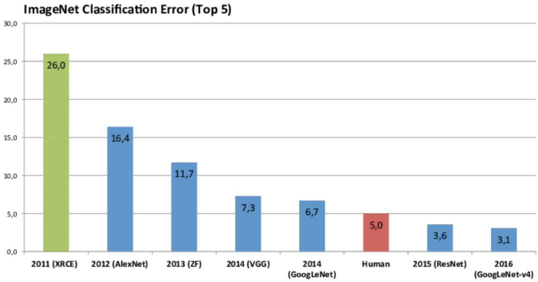
Master thesis proposals

- Stochastic EM, probabilistic programming and Topic Models.
- BERT for article segmentation/reconstruction.
- CNN for newspaper segmentation.
- BERT for legal predictions.
- Send me a note if you are interested (of any) before Thursday 23.59!

Convolutional Neural Networks

- *Convolutional* Neural Networks are behind great progress in the 2010s.
- It has revolutionized Computer Vision.
- Also called: ConvNets, Convolutional nets, Convolutional networks

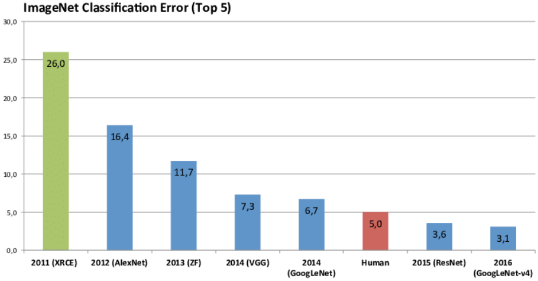
Figure: ImageNet performance (Roessler, 2019)



Convolutional Neural Networks

- *Convolutional* Neural Networks are behind great progress in the 2010s.
- It has revolutionized Computer Vision.
- Also called: ConvNets, Convolutional nets, Convolutional networks

Figure: ImageNet performance (Roessler, 2019)



Acknowledgements: Anders Eklund, Linköping University.

Convolutional Neural Networks

- Special architecture that works well for data with a grid structure
  1. 1D-grids: Time series
  2. 2D-grids: Gray-scale Images (pixels)
  3. 3D-grids: Color Images (pixels and chanels)
  4. 4D-grids: Color Video (pixels, chanels, frames)

Convolutional Neural Networks

- Special architecture that works well for data with a grid structure
  1. 1D-grids: Time series
  2. 2D-grids: Gray-scale Images (pixels)
  3. 3D-grids: Color Images (pixels and chanels)
  4. 4D-grids: Color Video (pixels, chanels, frames)

Computer Vision

- Problems
  - Image Classification

Computer Vision

- Problems
  - Image Classification
  - Image Segmentation

Computer Vision

- Problems
  - Image Classification
  - Image Segmentation
  - Object Detection

Computer Vision

- Problems
  - Image Classification
  - Image Segmentation
  - Object Detection
  - Object Localization

Computer Vision

- Problems
  - Image Classification
  - Image Segmentation
  - Object Detection
  - Object Localization
- Focus: 2D and 3D data

Computer Vision

- Problems
  - Image Classification
  - Image Segmentation
  - Object Detection
  - Object Localization
- Focus: 2D and 3D data
- Very Large Datasets:
  - ImageNet: 14M Images, 20k classes, 1M bounding boxes

Computer Vision

- Problems
  - Image Classification
  - Image Segmentation
  - Object Detection
  - Object Localization
- Focus: 2D and 3D data
- Very Large Datasets:
  - ImageNet: 14M Images, 20k classes, 1M bounding boxes
- Many different trained models and transfer learning

Example: Object Detection

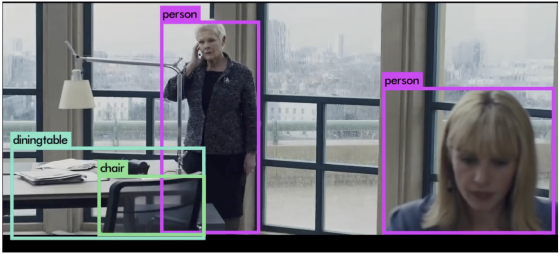


Figure: Object detection (see https://www.youtube.com/watch?v=VOC3huqHrss)

Example: Pneumonia detection

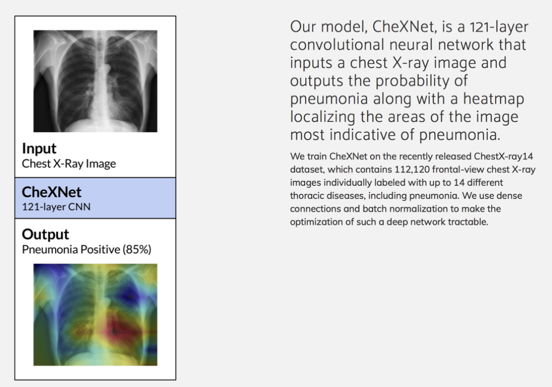


Figure: Rajpurkar et al. (2017). Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning. arXiv preprint arXiv:1711.05225.

Example: Fracture detection

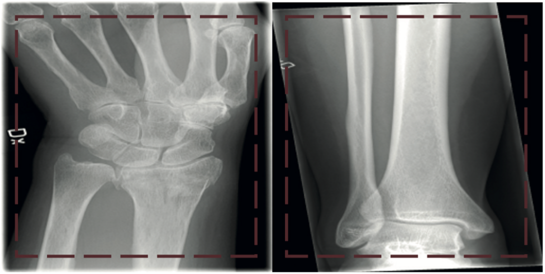


Figure 1. 2 images from the dataset. The area within the red box is the section presented to the network in order to classify the image. The left image is of a wrist fracture while the right image is without any apparent fracture.

Figure: Olczak et al, (2017) Artificial intelligence for analyzing orthopedic trauma radiographs, Acta Orthopaedica, 88:6, 581-586

What is an Image?

- 2-dimensional object
- Each pixel has:
  1. a value (light intensity)
  2. a coordinate

What is an Image?

- 2-dimensional object
- Each pixel has:
  1. a value (light intensity)
  2. a coordinate
- Grayscale: single channel
- Color: three channel (RGB)

What is an Image?

- 2-dimensional object
- Each pixel has:
  1. a value (light intensity)
  2. a coordinate
- Grayscale: single channel
- Color: three channel (RGB)
- Spatial and hiearchical correlation structures

MNIST example

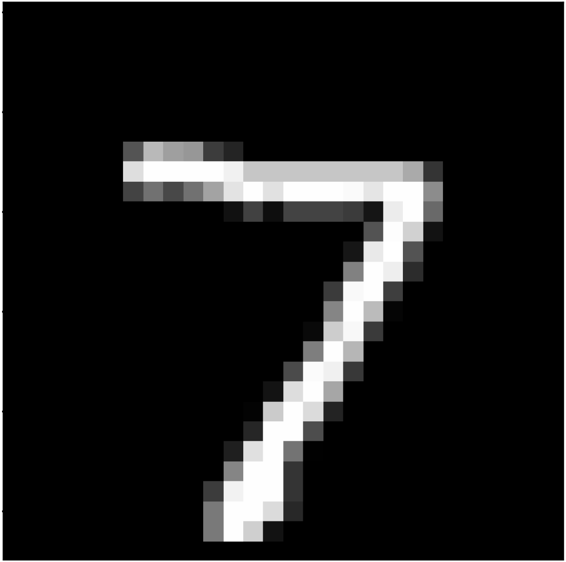


Figure: Example from the MNIST dataset (28 by 28 pixels)

How to train models for images?

- We want to learn representations of parts of images

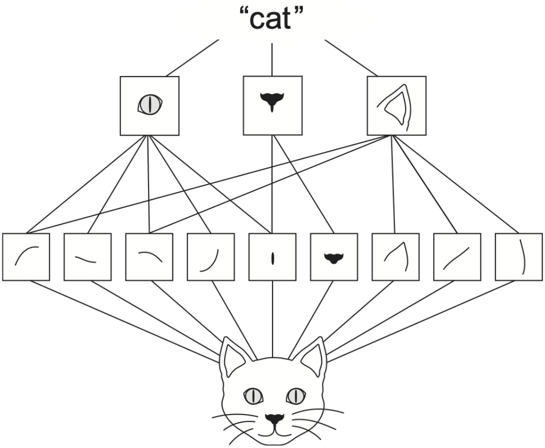


Figure: The representations of a cat (Chollet and Allair, 2018, Fig 5.2)

How to train models for images?

- We want to learn representations of parts of images

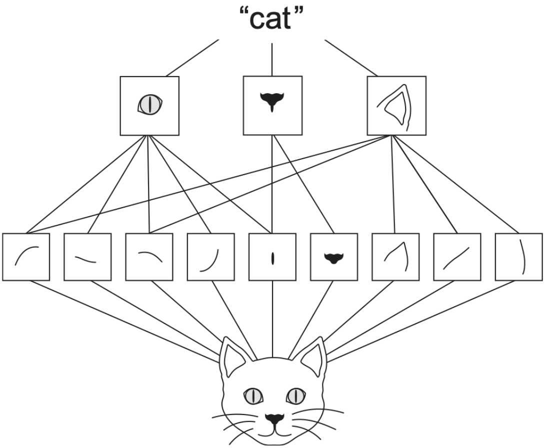


Figure: The representations of a cat (Chollet and Allair, 2018, Fig 5.2)

- CNN uses Convolutional Layers to learn parameter efficient representations:

Learning Representations for Images (again)

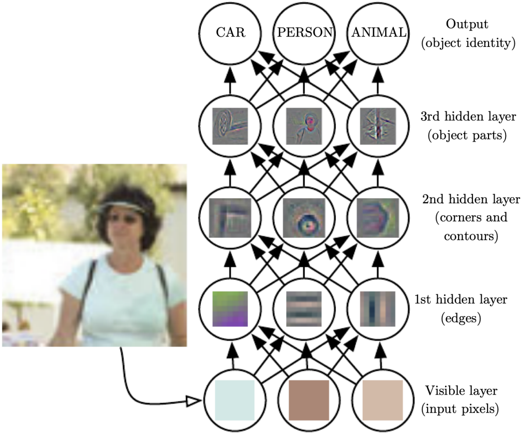


Figure: Learning representations for images (Goodfellow et al, 2017, Fig. 1.2)

Convolution

UPPSALA UNIVERSITET

- Different definitions are common, one example:

$$y(t) = \int x(\tau)k(t-\tau)d\tau = (x * k)(t)$$

- "Weighting together two functions"
- In CNNs:
  1. $x(t)$: Input
  2. $k(t)$: Kernel, filter, "feature"
  3. $y(t)$: Output, feature map

Discrete Convolution

- If t is discrete (as in a grid):

$$y(t) = (x * k)(t) = \sum_{\tau=-\inf}^{\inf} x(\tau)k(t-\tau)$$

- In the case of images we have 2 discrete dimensions

$$Y(i,j) = (X * K) \sum_{m} \sum_{n} X(m,n)K(i-m, j-n)$$

- Sometimes the cross-correlation is called convolution:

$$Y(i,j) = (X * K) \sum_{m} \sum_{n} X(m,n)K(i+m, j+n)$$

1. $X(i,j)$: Input (2D)
2. $K(i,j)$: Kernel, filter, "feature" (2D)
3. $Y(i,j)$: Output, feature map (2D)

Convolution of Images: 2D

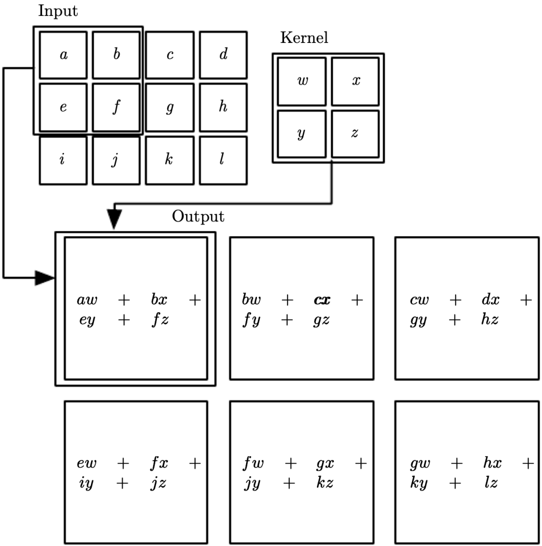


Figure: Convolution for an Image (Goodfellow et al, 2017, Fig. 9.1)

Convolution of images: Example

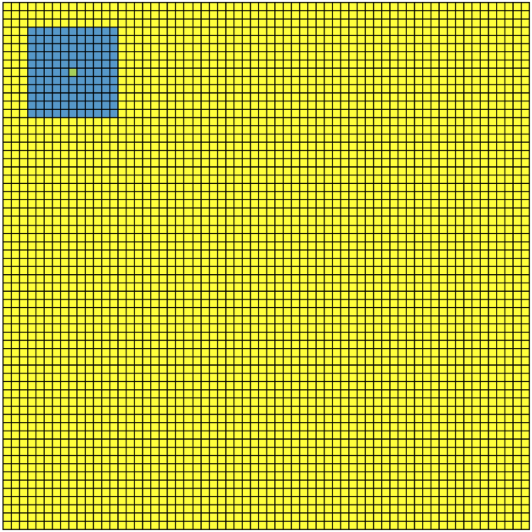


Figure: Convolution example.

Convolution of images: Examples

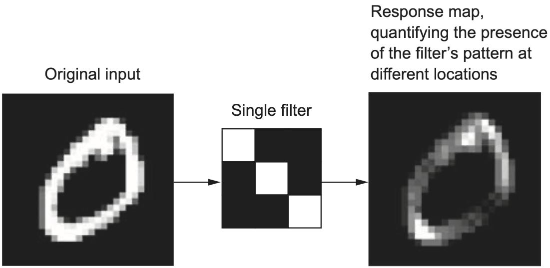


Figure: Convolution for an Image (Chollet and Allaire, 2018, Fig. 5.3)

Convolution of images: Example

$$X = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}, K = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$$

UPPSALA UNIVERSITET

Convolution of images: Example

$$X = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 \end{bmatrix}, K = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix}$$

$$Y = \begin{bmatrix} 2 & 1 & 2 & 0 \\ 2 & 1 & 2 & 0 \end{bmatrix},$$

Convolutions

- Benefits:
  1. Few parameters (filters)
  2. Captures local structures
  3. Efficient computations

Convolutions

- Benefits:
  1. Few parameters (filters)
  2. Captures local structures
  3. Efficient computations
- How to choose filters?
  1. Before: manually handcrafted
  2. Now: learn the filters using backpropagation $\rightarrow$ CNN

Convolution layer

- Input: Data or Feature Maps
- Parameters:
  - N filters/kernels of size $m \times m$
  - N bias terms (one per filter)
- Activation function: Applied element wise on feature maps
- Output: Feature Maps

Convolution layer

- Input: Data or Feature Maps
- Parameters:
  - N filters/kernels of size $m \times m$
  - N bias terms (one per filter)
- Activation function: Applied element wise on feature maps
- Output: Feature Maps
- In Keras:
```
layer_conv_2d(filters = 32, kernel_size =
c(3,3), activation = "relu", input_shape =
c(32,32,3))
```

Padding

- Handling edges
- *Padding*: add 0 around the image
- Nessecary to keep size of feature maps

Padding

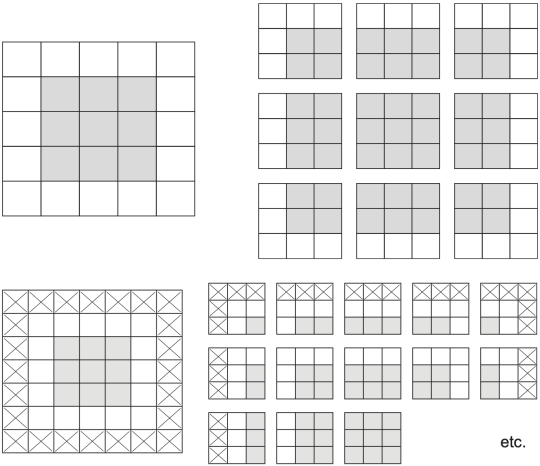


Figure: Padding and valid edge handling (Chollet and Allair (2018), Fig. 5.5, 5.6)

UPPSALA UNIVERSITET

Stride

- Skip every *n*th pixel
- Reduces the computations

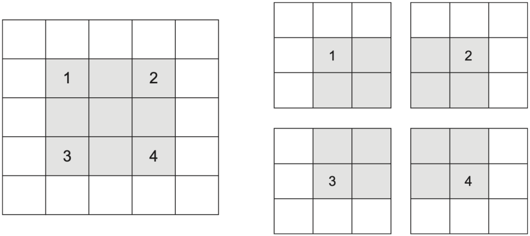


Figure: Strides (Chollet and Allair (2018), Fig. 5.5, 5.6)

UPPSALA UNIVERSITET

Why Convolution Layers?

- Captures local spatial structure
- Reduces the number of parameters (parameter sharing)
  1. The number and size of filters
  2. We use the same filters everywhere

Why Convolution Layers?

- Captures local spatial structure
- Reduces the number of parameters (parameter sharing)
  1. The number and size of filters
  2. We use the same filters everywhere
- Example: a 1 megapixel image (1000×1000 pixels)
  1. Dense network with 100 nodes: 100M parameters
  2. CNN network with 100 3×3 filters: 1000 parameters (900 from filters, 100 bias terms)

Convolution Neural Nets

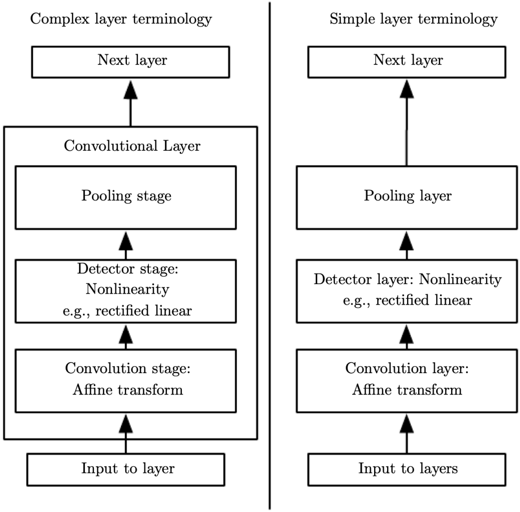


Figure: Convolution layer (Goodfellow et al, 2018, Fig. 9.7)

Detector stage

- Remember, in dense networks: $h = \sigma(XW + b)$
- In CNN:
  1. W "is the filter"
  2. X is the input feature map
  3. XW "is the convolutional feature map"
  4. b is a bias (one per filter)
  5. σ is the activation function (usually a ReLu)

Pooling layer

- Similar to a convolution
- We take a function f that return one value per pooling kernel

Pooling layer

- Similar to a convolution
- We take a function f that return one value per pooling kernel
- Most commonly $f = \max$
- Commonly a 2×2 pooling kernel with stride 2
- Why? Reduce the size of feature map, but keep the activation

Pooling layer

- Similar to a convolution
- We take a function f that return one value per pooling kernel
- Most commonly $f = \max$
- Commonly a 2×2 pooling kernel with stride 2
- Why? Reduce the size of feature map, but keep the activation
- In Keras:
  `layer_max_pooling_2d(pool_size = c(2, 2))(`

Max Pooling

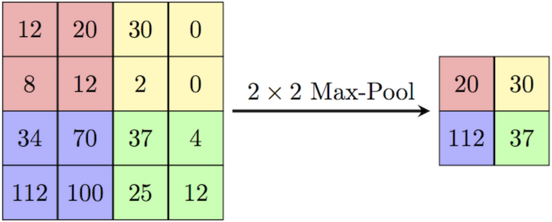


Figure: Strides (Computer Science Wiki: "Max-pooling / Pooling")

Data Augmentation

Figure: Data Augmentation (Chollet and Allair, 2018, Fig 5.10)

Data Augmentation

Figure: Data Augmentation (Chollet and Allair, 2018, Fig 5.10)

Popular CNN architectures

- AlexNet (2012), 5 convolutional layers
- VGGNet (2014), 16 convolutional layers
- ResNet (2015), 152 convolutional layers

VGG

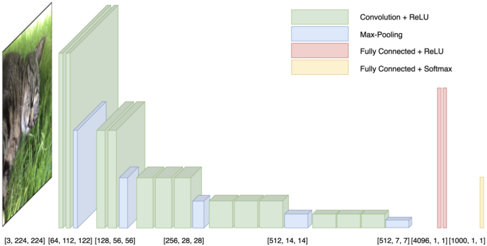


Figure 2.4: *Illustration of the data flow through the network VGG16. Data size, of the format $[c, h, w]$, is shown for the input image, output of each max-pooling layer, output after the first two fully connected layers, and the final network output.*

Jesper Westell, Multi-Task Learning using Road Surface Condition Classification and Road Scene Semantic Segmentation, LIU-IMT-TFK-A–19/570–SE

UPPSALA UNIVERSITET

Transfer learning

- "Transfer knowledge between problems"
- In practice: Transfer/reuse learned weights

UPPSALA UNIVERSITET

Transfer learning

- "Transfer knowledge between problems"
- In practice: Transfer/reuse learned weights
- Commonly: Use (large) pre-trained models for smaller problems
- One of the main reasons for the success of (convolutional) neural networks

Transfer learning

- "Transfer knowledge between problems"
- In practice: Transfer/reuse learned weights
- Commonly: Use (large) pre-trained models for smaller problems
- One of the main reasons for the success of (convolutional) neural networks
- Two types of transfer learning:
  - Feature extraction
  - Fine Tuning

Learning Representations for Images

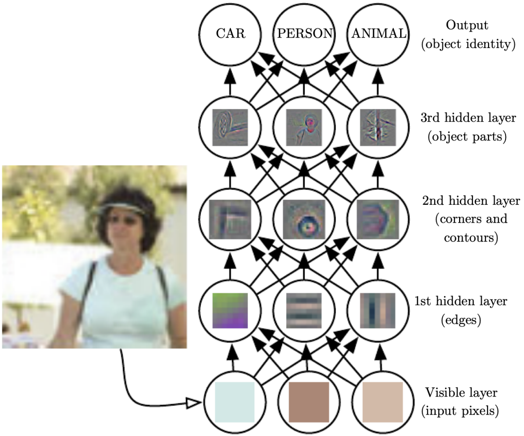


Figure: Learning representations can be crucial (Goodfellow et al, 2017, Fig. 1.2)

Feature Extraction

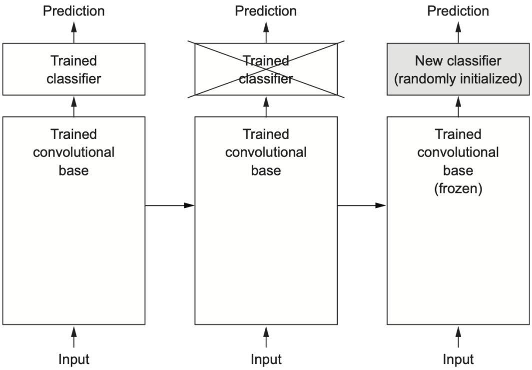


Figure: Using convnets as base for feature extraction (Chollet and Allair, 2018, Fig 5.12)

Fine-Tuning

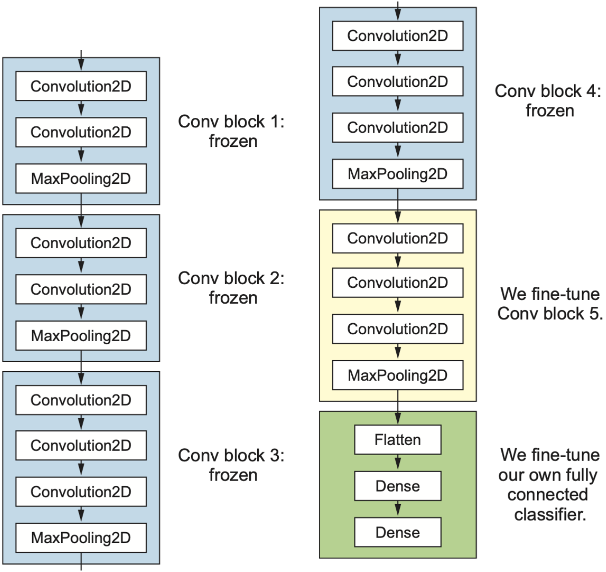


Figure: Finetuning a convolutional base (Chollet and Allair, 2018, Fig 5.15)

A quick demo

A demo using Keras for the MNIST datset in R.