

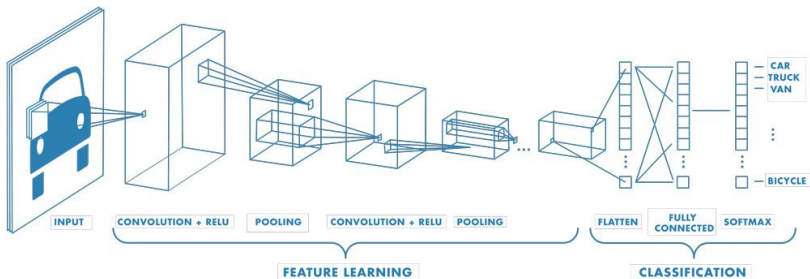
Neural Networks 2 - Convolutional Neural Networks

18NES2 - Lecture 5, Winter semester 2025/26

Zuzana Petříčková

October 21, 2025

Today's Lecture: Introduction to Convolutional Neural Networks



Source:

<https://matlabacademy.mathworks.com/details/deep-learning-onramp/deeplearning>

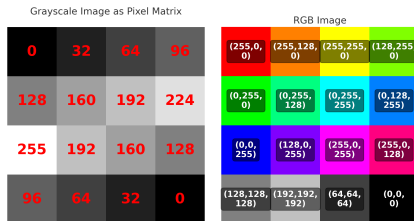
Introduction to Convolutional Neural Networks

Today's Lecture:

- Motivating example: bird species classification
- Convolution operation – intuition, purpose, and parameters
- Convolutional neural network architecture (layers, filters, pooling)
- Classic CNN architecture and Fashion MNIST example

Reminder: Digital Image Representation

- A digital image is a matrix (tensor) of pixels.
- Each pixel (short for "picture element") describes the color at a specific position in the image.



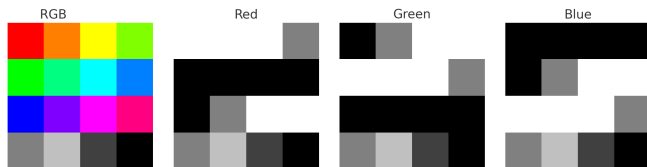
Grayscale Image

- Each pixel is a single value indicating brightness (e.g., 0 = black, 255 = white).
- For machine learning, pixel values are usually normalized to the interval $[0, 1]$.

Reminder: Digital Image Representation

Color Image (RGB)

- Each pixel consists of three components: R (red), G (green), B (blue).
- The image is represented as a 3D tensor of shape (height $\times$ width $\times$ 3).
- These components are called color channels.



Example: `convolution_introduction.ipynb`

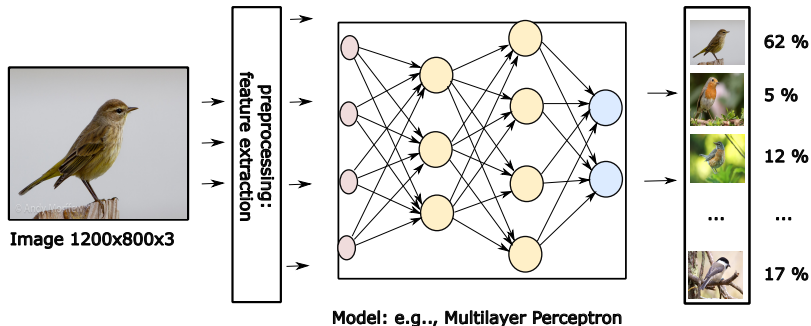
Motivating Example: Image Classification

Bird Species Recognition



Motivating Example: Bird Species Recognition

Classical Machine Learning Approach

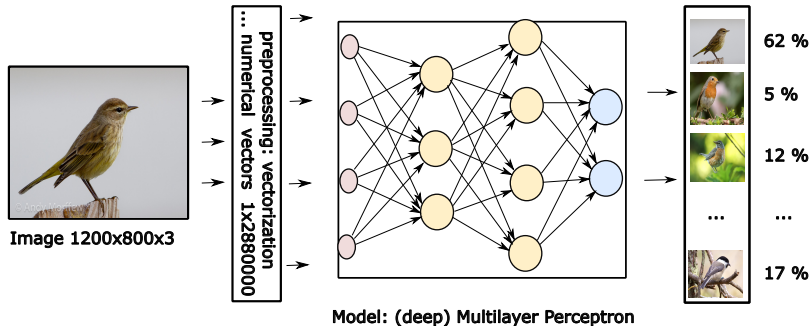


Thorough preprocessing of the data: Feature Extraction

- edge detection, LBP histograms, etc.
- information loss; requires careful feature design

Motivating Example: Bird Species Recognition

What if we train a neural network directly on the data?

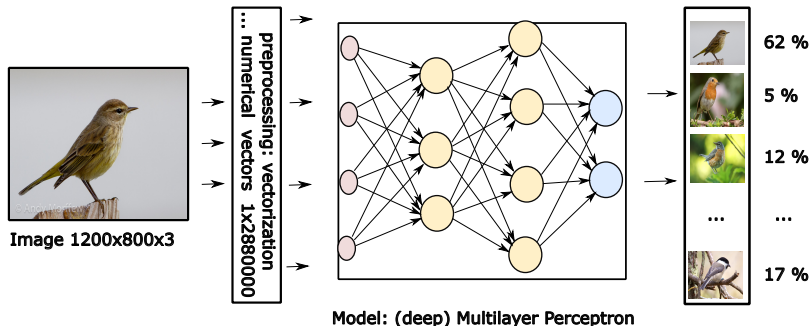


Deep Learning Principle

- let the model learn useful features from the data
- preprocess the data just slightly (e.g., vectorization, normalization)

Motivating Example: Bird Species Recognition

What if we train a neural network directly on the data?

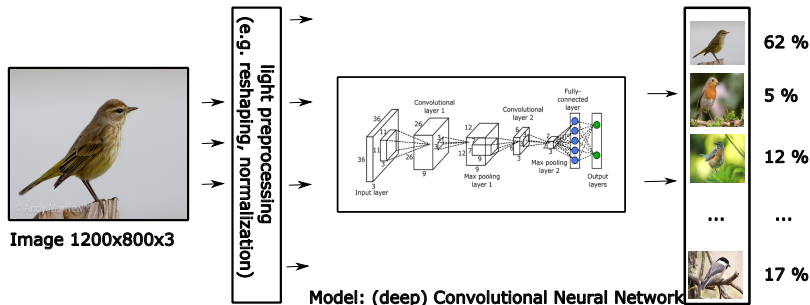


Drawbacks of the classical approach (fully connected layers only):

- high number of features
- loss of spatial relationships between pixels
- it is difficult to train the model effectively

Motivating Example: Bird Species Recognition

How could we improve this?

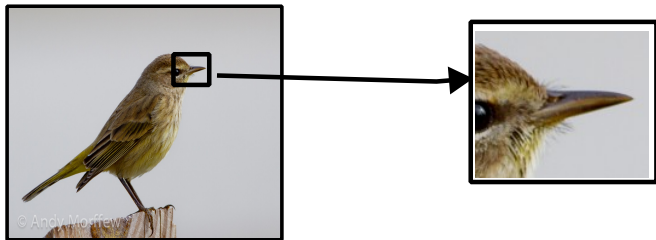


Convolutional Neural Network (CNN):

- a neural network with convolutional layers
- takes spatial arrangement of pixels into account
- fewer parameters, easier to train compared to fully connected networks

Motivating Example: Bird Species Recognition

Patterns in data: for example, beak shape



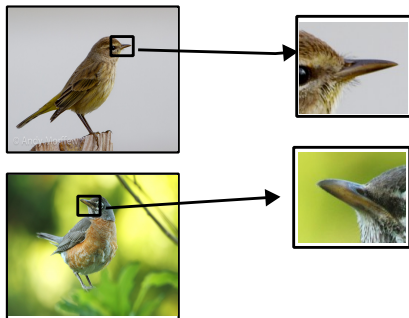
Let's create a beak detector:

- a simple model (e.g., a single-layer neural network) that detects beaks in images

But: the beak may appear in different locations within the image

Motivating Example: Bird Species Recognition

Patterns in data: for example, beak shape

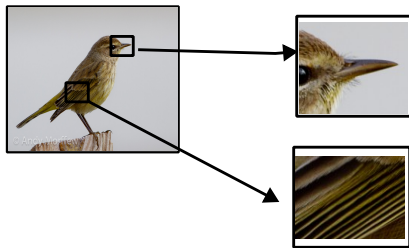


The beak may appear in different image regions

- the detector should find a beak in any image and at any location
- the detector must slide over the input image

Motivating Example: Bird Species Recognition

There are multiple patterns in the data (e.g., beak, feather, eye):



The idea:

- create a set of detectors for different features (patterns)
- detectors should recognize features anywhere in the image
→ detectors slide over the image
- these detectors form the initial layers of a convolutional neural network

Convolutional Neural Network

- A neural network that includes convolutional layers

Convolutional Layer

- Consists of a set of filters (also called kernels or detectors)
- Each filter performs a convolution operation over the input image
- The output of the convolution (a feature map) is passed to the next layer

Convolution Operation

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

Image 6x6 (black and white)

Convolutional Layer:

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1 (3x3)

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2 (3x3)

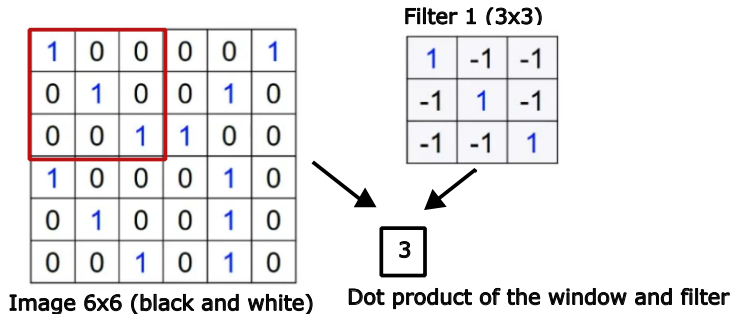
■ ■ ■

- The convolutional layer contains several filters
- Each filter detects a pattern (feature) of size 3×3 pixels (e.g., diagonal edge, vertical edge, etc.)

Example source: Petr Doležel – Convolutional Neural Network,

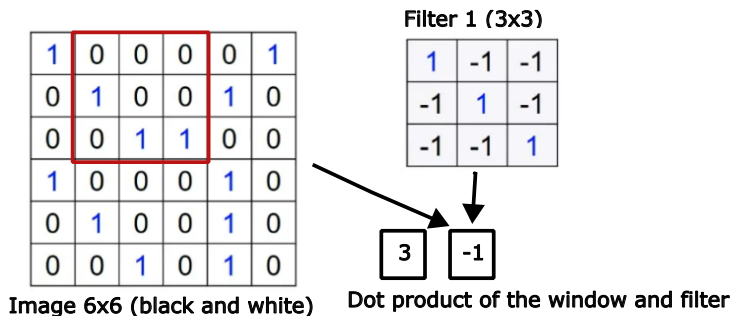
<https://www.youtube.com/watch?v=-2vEi-Aa0FA>

Convolution Operation



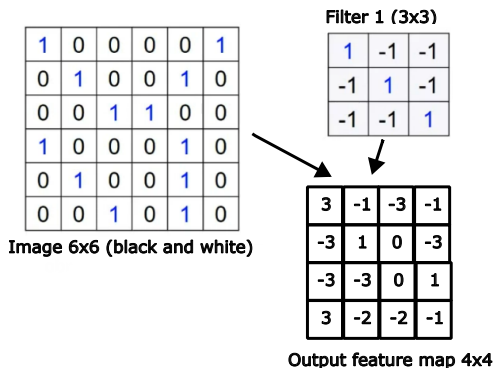
- We compute the dot product:
$$y = \sum_{i=1}^9 w_i x_i + b \text{ (for flattened matrices)}$$

Convolution Operation



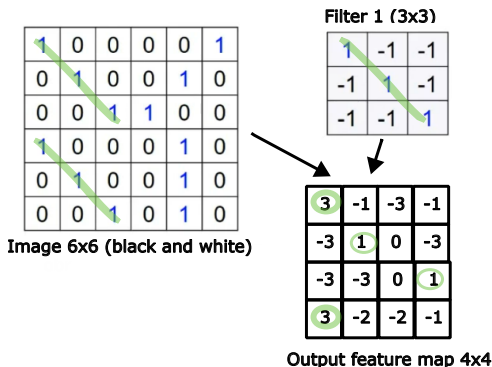
- Move the sliding window and compute another dot product

Convolution Operation



- By sliding the window over the image, we apply the filter to the entire input
- The result is a new 2×2 tensor – a **feature map**

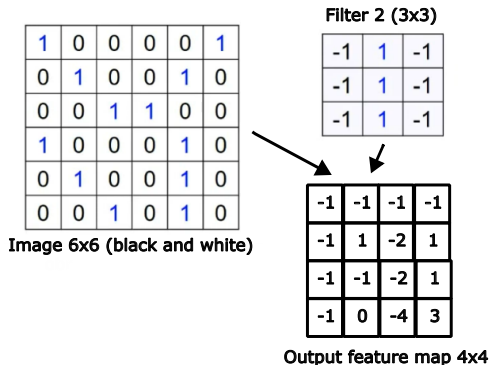
Convolution Operation



Feature Map

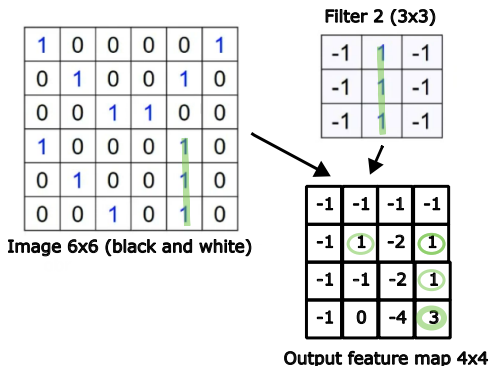
- Indicates where (and how strongly) the pattern represented by the filter appears in the input image
- Example: diagonal edge filter

Convolution Operation



- We can similarly apply a second filter

Convolution Operation

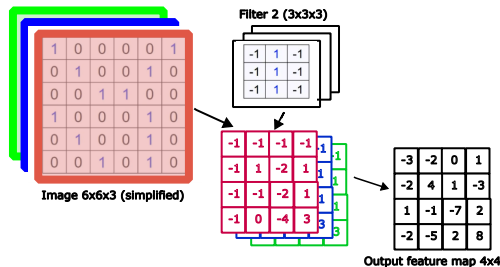


Second Feature Map

- Indicates where (and how strongly) the pattern represented by the second filter appears in the input image
- Example: vertical edge filter

Convolution Operation

Color Image: 3 input channels – R, G, B



- Each filter has weights for **all input channels** (R, G, B)
- Computation: convolution is performed separately on each channel and the results are **summed**
- Each filter produces **one aggregated** output feature map
- The number of filters defines the **number of output channels**

Convolution Operation

Example: Zebra



Source:

<https://matlabacademy.mathworks.com/details/deep-learning-onramp/deeplearning>

Convolution Operation

Example: Zebra



-1	1	1	-1
-1	1	1	-1
-1	1	1	-1
-1	1	1	-1



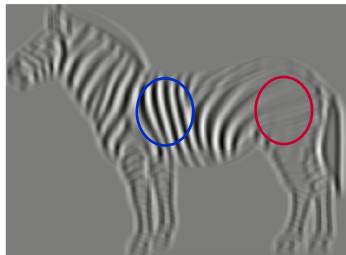
- Output after applying the vertical stripe filter

Convolution Operation

Example: Zebra



-1	1	1	-1
-1	1	1	-1
-1	1	1	-1
-1	1	1	-1



- Clearly shows where the pattern is strongly present and where it is not

Convolution Operation

Example: Zebra



Vertical stripes

Diagonal cross

Horizontal edge

White blob

Diagonal edge



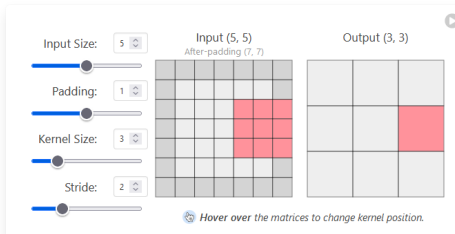
- Examples of other filters and their resulting feature maps

Convolution Operation

Convolution Operation Parameters

- Dimensions of the input image
- Padding – how borders are handled
- Filter size
- Stride – the step used to move the filter across the image

Understanding Hyperparameters



Great interactive visualization:

<https://poloclub.github.io/cnn-explainer/>

Convolution Parameters: Padding and Stride

Padding

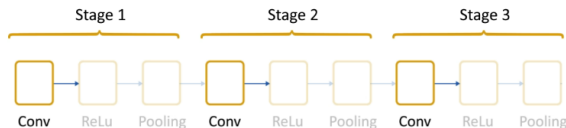
- Adds extra border pixels around the input image
- Common options:
 - **Valid** – no padding (output shrinks)
 - **Same** – padding added to keep output size the same as input
- Helps preserve spatial size and improve edge detection

Stride

- Controls how far the filter moves at each step
- **Stride = 1**: typical setting, dense coverage
- **Stride > 1**: reduces output size, performs downsampling

Tip: Try out different values in CNN Explainer

Classic CNN Architecture



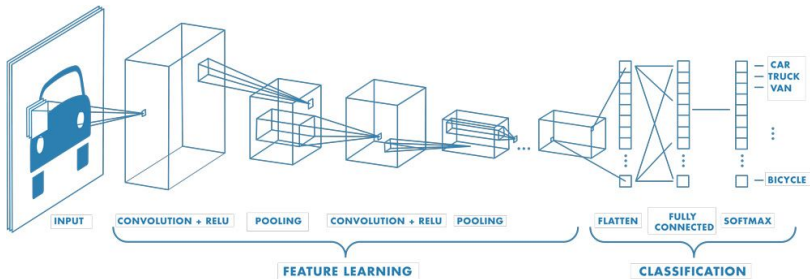
Core idea: stack convolutional layers (or blocks) on top of each other

- The first convolutional layer detects simple features (e.g., edges, blobs)
- Each following layer extracts higher-level features

Hierarchical structure of features:

edges → shapes → object parts → whole objects

Classic CNN Architecture



Main components of a CNN:

- **CNN base** - Convolutional blocks for feature extraction
- **Flattening layer** – converts the feature maps into a 1D vector
- **Head** - Fully connected neural network for classification

Image source: <https://matlabacademy.mathworks.com/details/deep-learning-onramp/deeplearning>

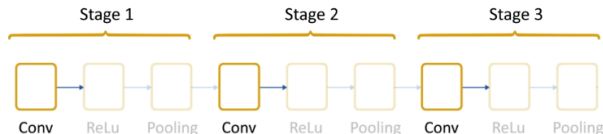
[//matlabacademy.mathworks.com/details/deep-learning-onramp/deeplearning](https://matlabacademy.mathworks.com/details/deep-learning-onramp/deeplearning)

Classic CNN Architecture



Typical structure of a convolutional block:

- Convolutional layer
- Nonlinear activation function (e.g., ReLU)
- Pooling layer



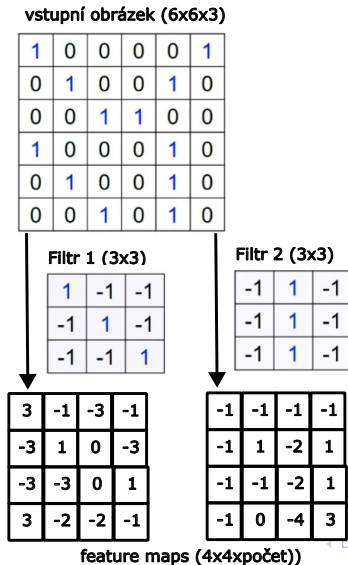
Pooling (Subsampling) Layer

- Reduces spatial resolution while preserving most of the relevant information
- A sliding window (e.g., 2×2) moves across the feature map, often with stride = 2
- Common operations: MAX (max pooling), AVERAGE (average pooling); no weights involved

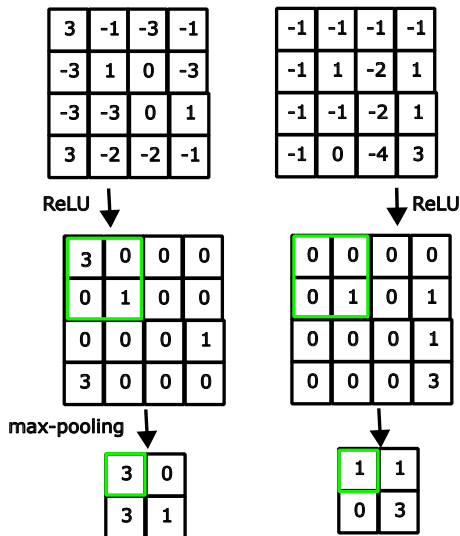
Why pooling?

- Condenses the information stored in the feature map
- Keeps track of where and how strongly a feature occurs
- Reduces the data size (e.g., $2 \times 2 \rightarrow 1$ value = 75% reduction)

Convolutional Block – Example: Convolutional Layer



Convolutional Block – Example: Pooling Layer



Convolutional Block

Why not just stack convolutional layers without pooling?

- The number of parameters grows with each added layer
- Image size stays (almost) the same, especially with "same" padding
⇒ the size of the feature maps (and computation) keeps growing

Pooling layer:

- Reduces data size while preserving information about feature presence and strength
- e.g., $2 \times 2 \rightarrow 1$ value = quarter size

Alternating convolution and pooling – bipyramidal effect:

- Spatial size decreases, number of feature maps increases

Bipyramidal Architecture

- One of the oldest architecture types: wide and shallow, with a deeper fully connected part – close to the basic layered schema

Typical structure of a bipyramidal architecture:

- The number of filters typically doubles in deeper layers (e.g., 32, 64, 128, ...)
- Most commonly used filter size: 3×3
- ReLu activation
- Max-pooling 2×2 is often paired with filter doubling
- When using several convolutional layers, we may not need many fully connected layers
- (Optionally) One or more fully connected layers are added for classification

CNN_fashion_mnist.ipynb

- Practical example: MNIST dataset (handwritten digits)

Training a Convolutional Neural Network

- Typically trained using a variant of backpropagation (e.g., SGD)
- Mini-batch learning – the model requires a large amount of data
- A high number of trainable parameters

How to choose a suitable architecture in practice?

- We usually don't optimize the number of layers or neurons manually
- We pick a proven topology from the literature for the given type of task:
 - Bipyramidal architecture
 - One of the modern architectures (e.g., <https://keras.io/api/applications/>)

Examples

CNN_fashion_mnist.ipynb

- Practical example: Fashion MNIST dataset (digits)

Useful links:

- Interactive CNN visualization:
<https://poloclub.github.io/cnn-explainer/>
- MathWorks – activation visualization (face)