AdTrans project, 2023-1-PL01-KA220-HED-000158917

ADTRANS
ADVANCED DATA ANALYSIS FOR TRANSPORTATION AND MOBILITY ACADEMIC TEACHERS

Session 4

Perception Module & Image Processing

Paraskevi Fasouli

Co-funded by the European Union

Part 1:

Perception Module

Overview

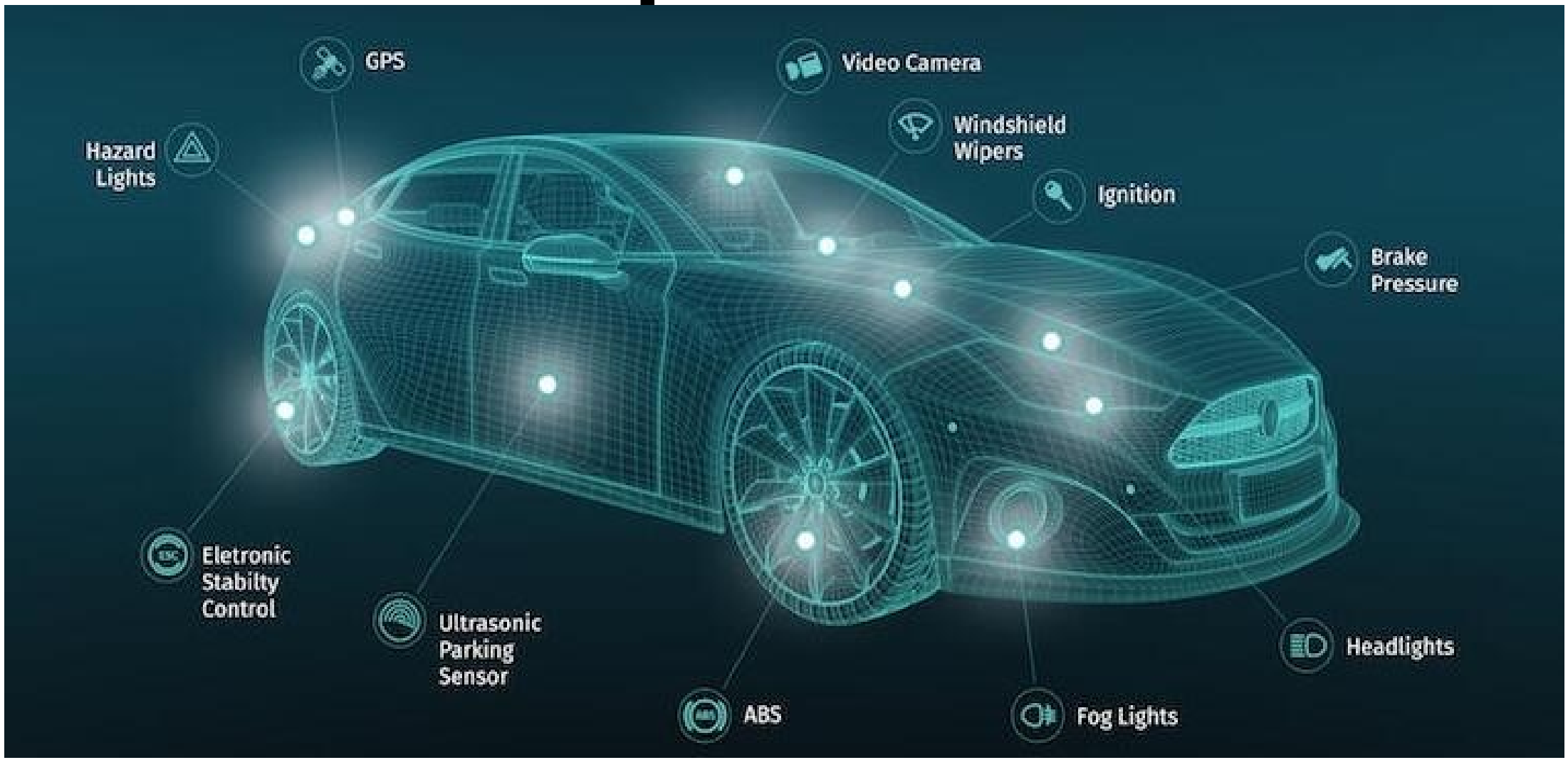
Car Sensor Explanation

Scene Understanding from Sensors

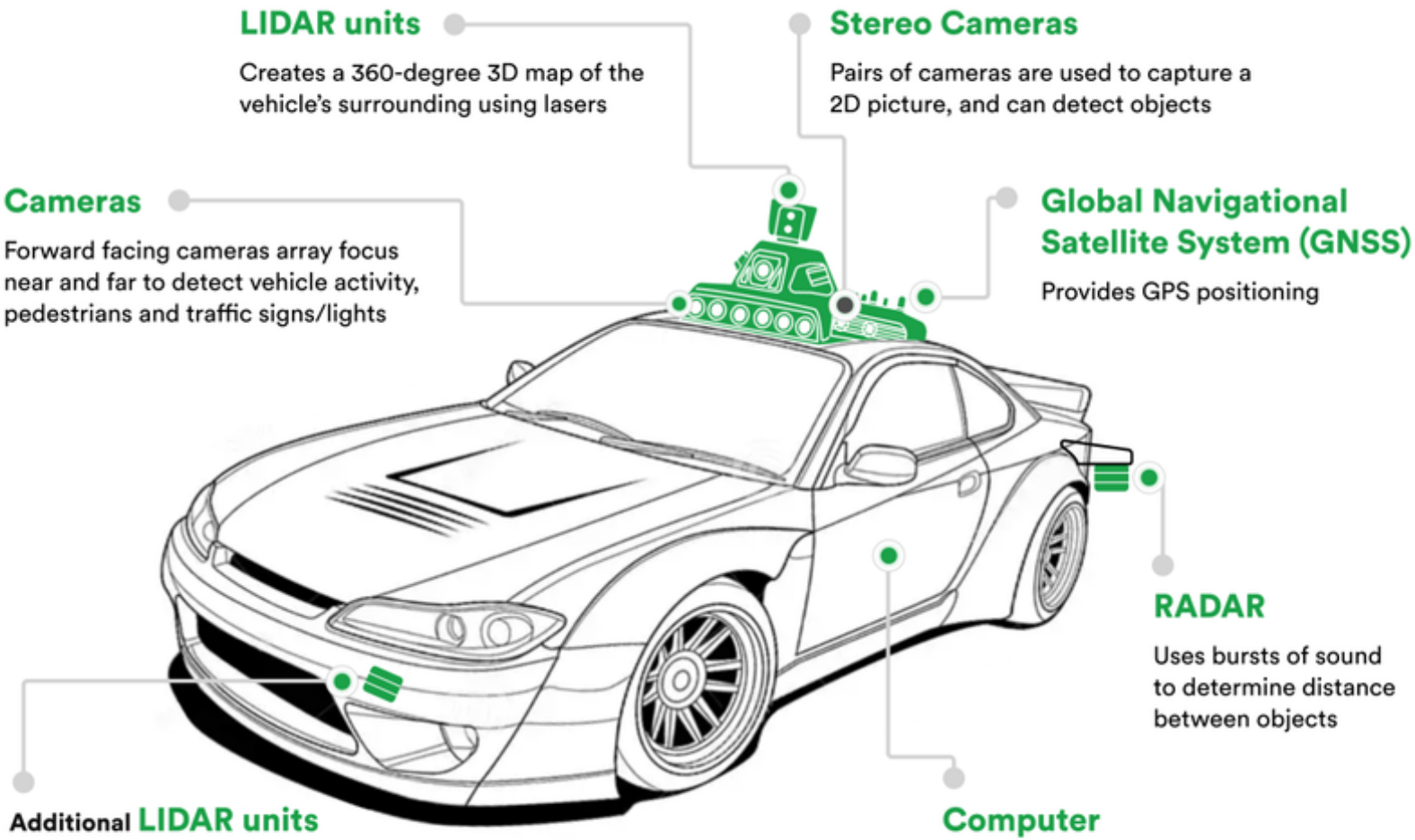
Convolution Explanation

CNN Architecture Explanation

Car Perception Sensors



Car Perception Sensors



Car Perception Sensors

Types of sensors:

Exteroceptive: Sensors that measure the surrounding

- e.g: Camera, LIDAR, RADAR

Proprioceptive: Sensors that measure the ego vehicle itself

- e.g: GNSS, IMU

Perception Sensors
Exteroceptive: CAMERAS

Properties:

- Resolution: Number of pixels in an image
- Field Of View (FOV)
- Dynamic range (Difference between the brightest and darkest pixel)

Stereo camera: Measures the depth using disparity maps (left and right images)

Benefits: Cheap, allowing deployment of multiple cameras (e.g Tesla)

Costs: Subjected to weather conditions (Visibility)

Perception Sensors
Exteroceptive: LIDARS

Properties:

- Uses time of flight (TOF) estimate of reflected light beams to measure the surrounding in 3D
- Number of beams LIDAR can emit
- Number of points per second LIDAR can collect
- Rotation rate
- Field Of View
- Detection range

Benefits: Not affected much by the environmental conditions

Costs: Expensive, Light absorbing or scattering surfaces can cause problems

Perception Sensors
Proprioceptive: GNSS

Global Navigation & Satellite System **Properties:**

- Measures the vehicle position and velocity
- Measurement technique: Trilateration

Benefits: Accuracy and Precision of the measurements

Costs: Cannot use for localisation in a map frame directly (must fuse with other sensors)

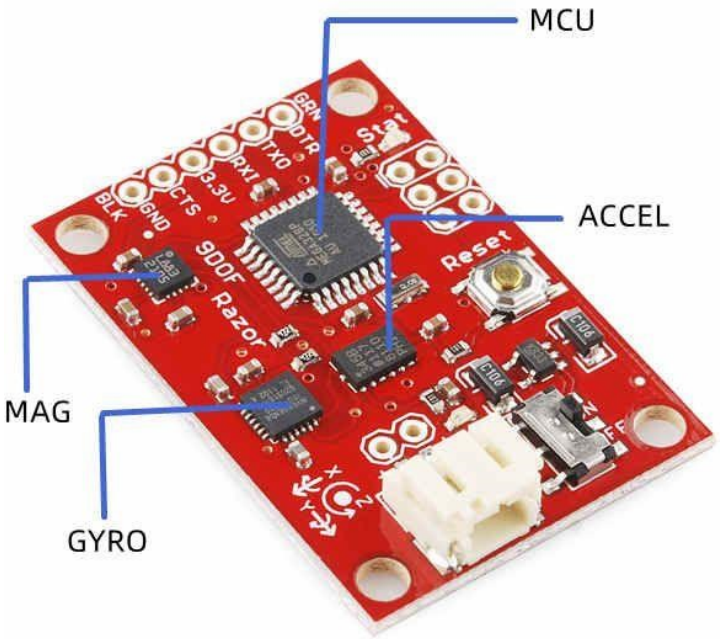
Perception Sensors
Proprioceptive: IMUS

Inertial Measurement Unit **Properties:**

- Measures body force, angular rates and sometimes the vehicle heading (3D orientation)
- Body force + dynamic vehicle model -> Position/Velocity and Acceleration
- Angular rate + dynamic vehicle model -> 3D Orientation

Benefits: Accuracy and Precision of the measurements

Costs: Cannot use for localization in a map frame directly (must fuse with other sensors)



Why Perceive the Environment?
Perception Module

Ego vehicle must be able to operate in an environment which is:

- Highly unstructured
- Extremely dynamic

Therefore, the ego vehicle must have an understanding about:

- Static objects
- Dynamic objects and their future trajectories
- Driving surfaces
- Lane marking and boundaries

Also determine the configuration of the ego vehicle in a given reference frame:

- What is the current position and the orientation

Why Perceive the Environment?
Perception Module

The information about the environment helps:

- **Behaviour model** to take better decisions
- **Trajectory planner** to produce the safest optimal trajectory
- **Localisation module** to localise the ego vehicle in the environment

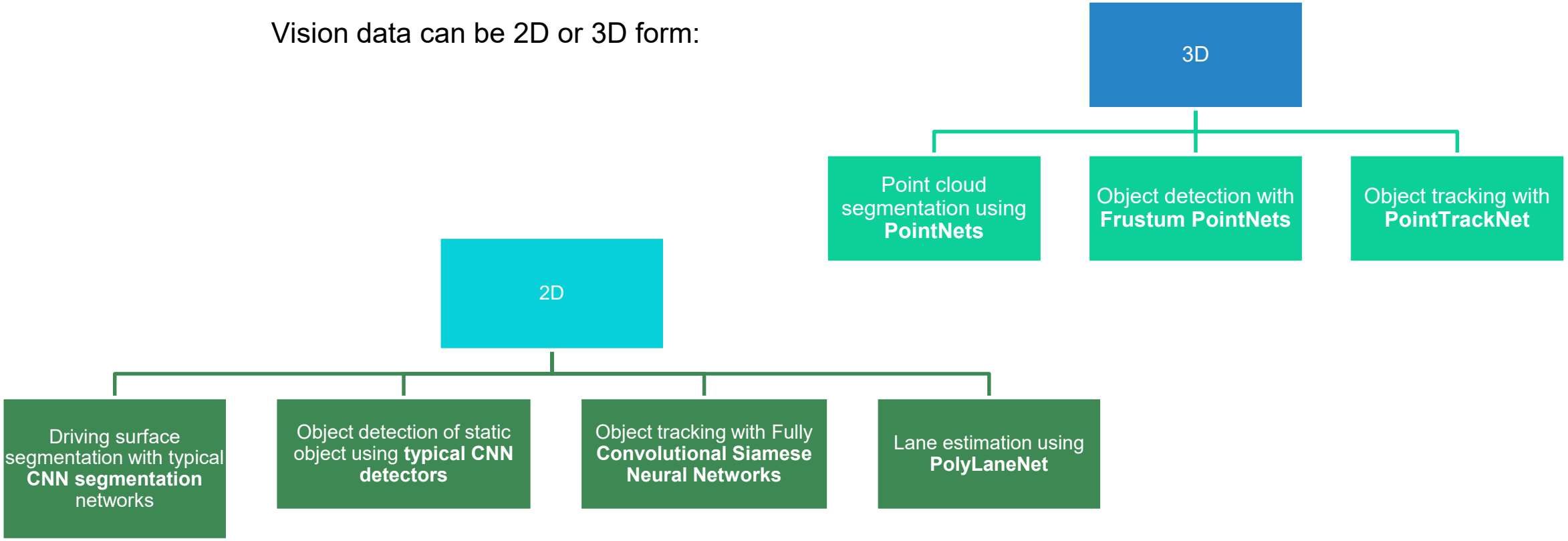
What are the environmental information:

- **Static objects:** Traffic signs, driving surfaces, road lanes etc.
- **Dynamic objects**: Pedestrians, Moving vehicles etc.

Why Perceive the Environment?

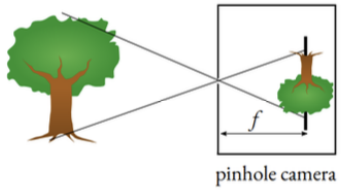
Perception Module

Vision data can be 2D or 3D form:

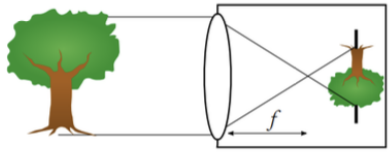


Data Acquisition For Perception

Exteroceptive Sensors: Scene Understanding

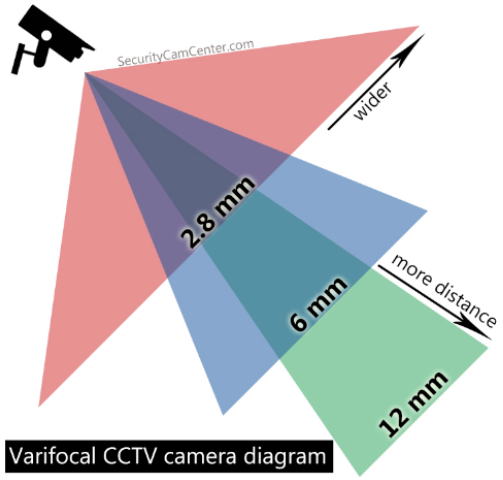


pinhole camera



camera with lens to focus image

Pinhole camera and lens camera



Varifocal CCTV camera diagram

FOV and Range vs Focal length

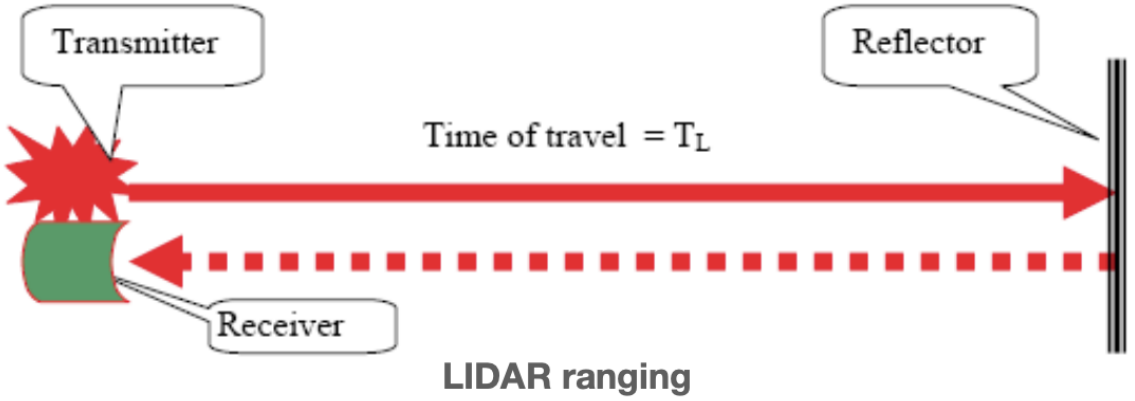
Observe the exterior of the ego vehicle using:
Cameras / Stereo Camera, LIDARs and Radars

Camera:	Stereo camera:
• There are pinhole cameras and lens cameras • A single camera can produce a single 2D image	• 2 Cameras placed apart parallel to each other • Used for depth image estimation

Data Acquisition For Perception
Exteroceptive Sensors: Scene Understanding

LIDAR: **3D Point cloud of reflective surfaces**

- Shoots laser beams to a surface (dynamic/static)
- Measures the time of flight (TOF)/elapsed time to determine the distance
- Measures the light intensities
- d=1/2ct c: speed of light
- Can determine the shape & material of the object surface
- With the above information 2D image reconstruction is possible



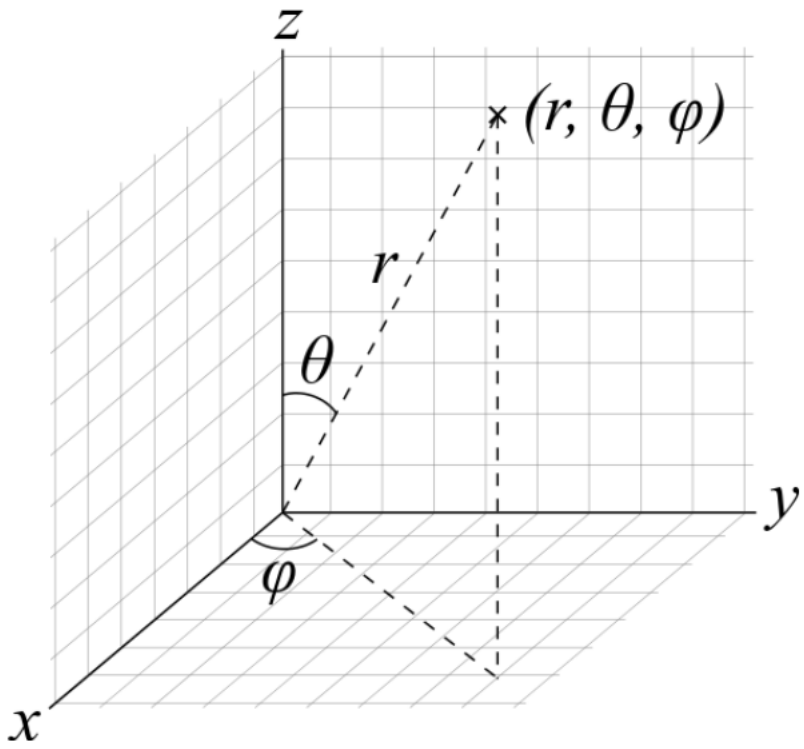
Data Acquisition For Perception
Exteroceptive Sensors: Scene Understanding

LIDAR Coordinate transformations:

- Measurement: Spherical coordinate system (r, ϕ, θ) *r : range, ϕ : azimuth angle, θ : elevation angle*

Cartesian coordinates (x, y, z) Transformation:

- x = rsin(θ)cos(ϕ)
- y = rsin(θ)sin(ϕ)
- z = rcos(θ)

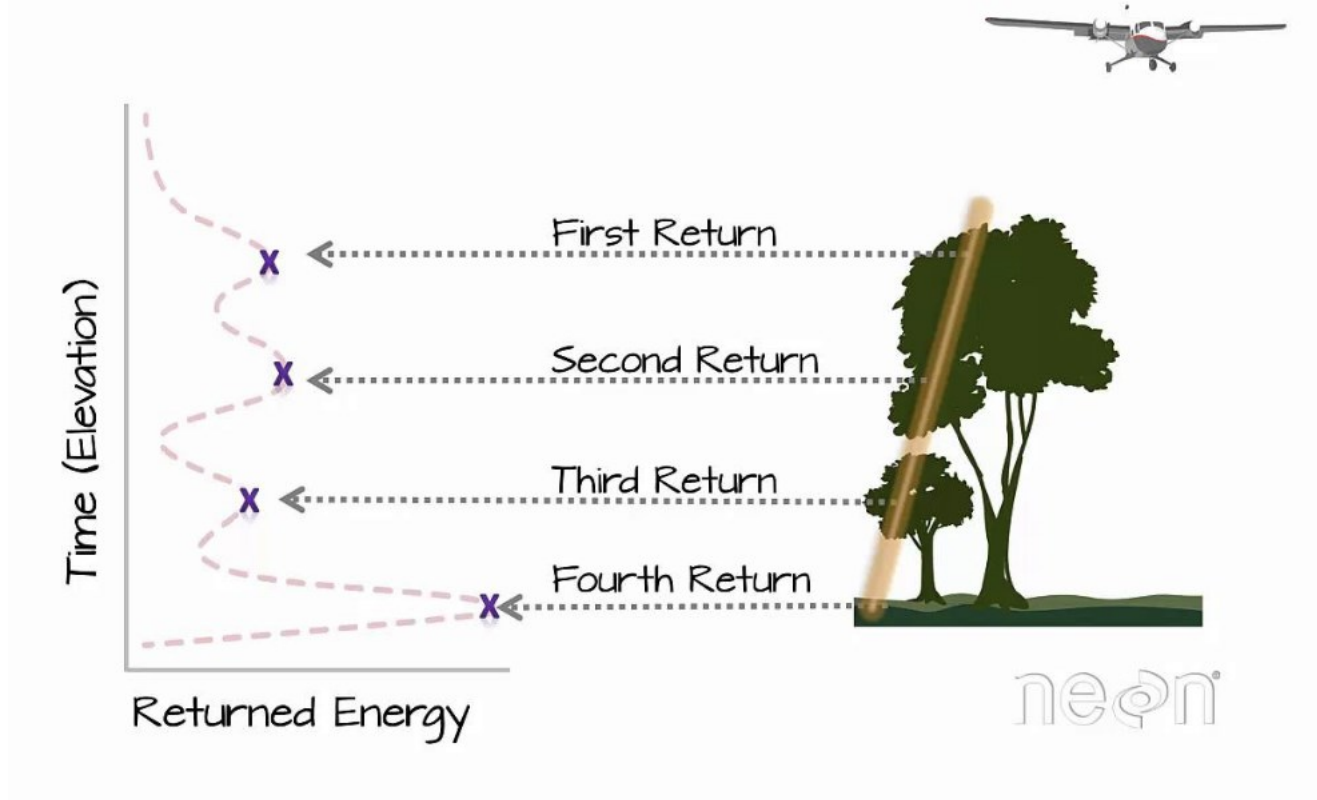


Spherical and cartesian coordinate system

Data Acquisition For Perception
Exteroceptive Sensors: Scene Understanding

LIDAR data structure:

- LIDAR collects data as the points reflected
- Each point is indexed and collected
- Take $p_1 = \begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} \ldots\ldots p_n = \begin{bmatrix} x_n \\ y_n \\ z_n \end{bmatrix}$
- Finally store them as
- $p = [p_1, p_2 \ldots . p_n]$
- $P = \begin{bmatrix} x_1 & y_1 & z_1 \\ \vdots & \vdots & \vdots \\ x_n & y_n & z_n \end{bmatrix}$

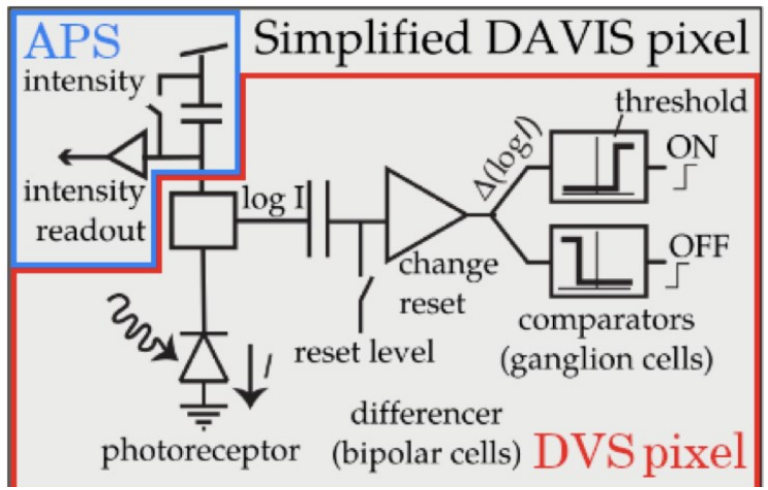


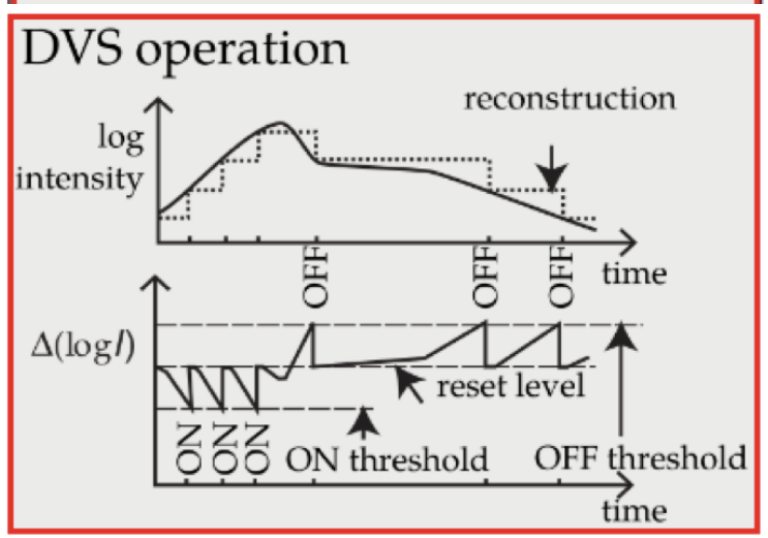
LIDAR to measure tree hight from the ground level

Data Acquisition For Perception
Exteroceptive Sensors: Scene Understanding

Dynamic Vision Sensor/ Event cameras

- Pixel triggers when there is a change in brightness from t_1 -> t_2
- Each pixel measures the ΔI asynchronously (Data driven)
- Tolerant to motion blur
- Produces a sparse output of events for each time step
- An event represents the brightness change on the object edges
- Has a very high temporal resolution
- When there is no event, output sparsity is very high





Part 2:

Deep Learning & Image Processing

Image Recognition Problems

Classic problem in computer vision, image processing, and machine vision is to determine if an image contains specific objects, features, or activities

Object recognition

- (classification into types of object like cat, dog, car, …)

Identification

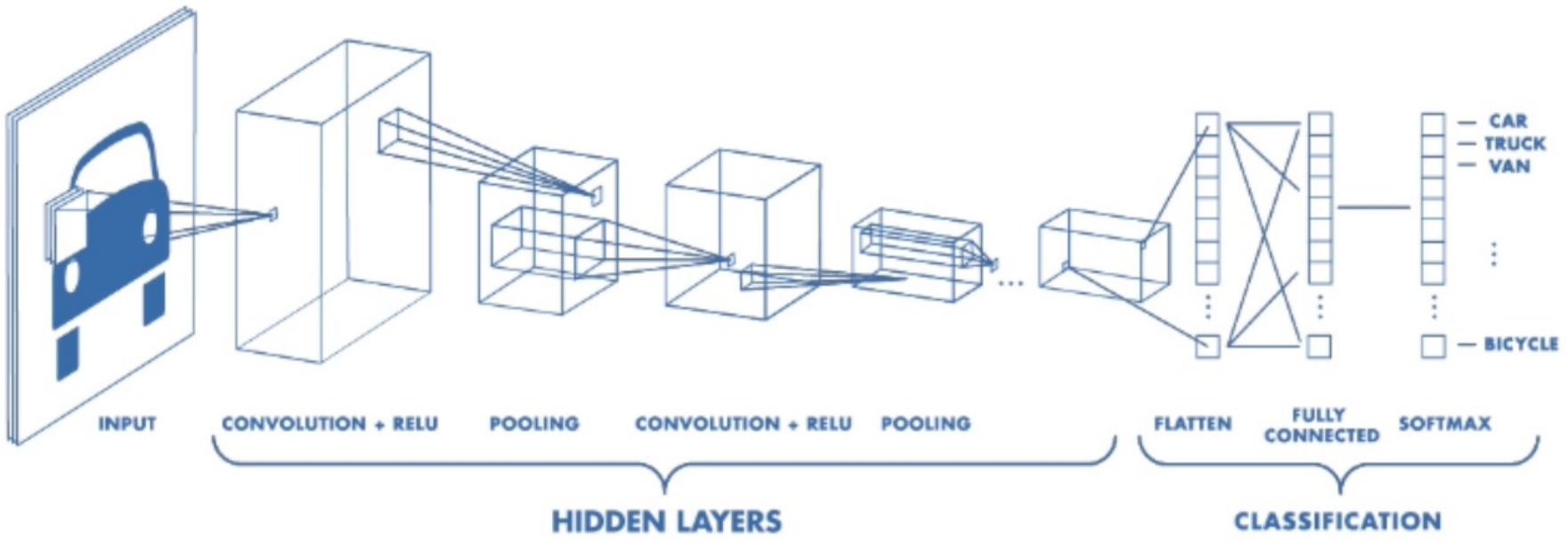
- (individual instance of an object is recognized, like face, voice, fingerprint, written digits)

Detection

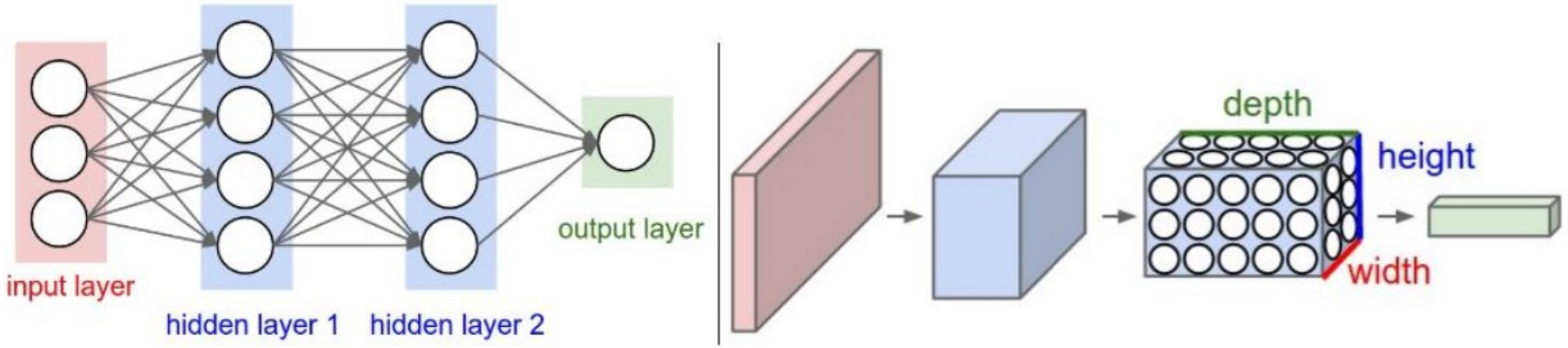
- (seek specific conditions, like abnormalities in medical images)

Many Convolutional Neural Nets are widely used (with GPUs)

Arrangement of CNN layers



Dense NN vs CNN



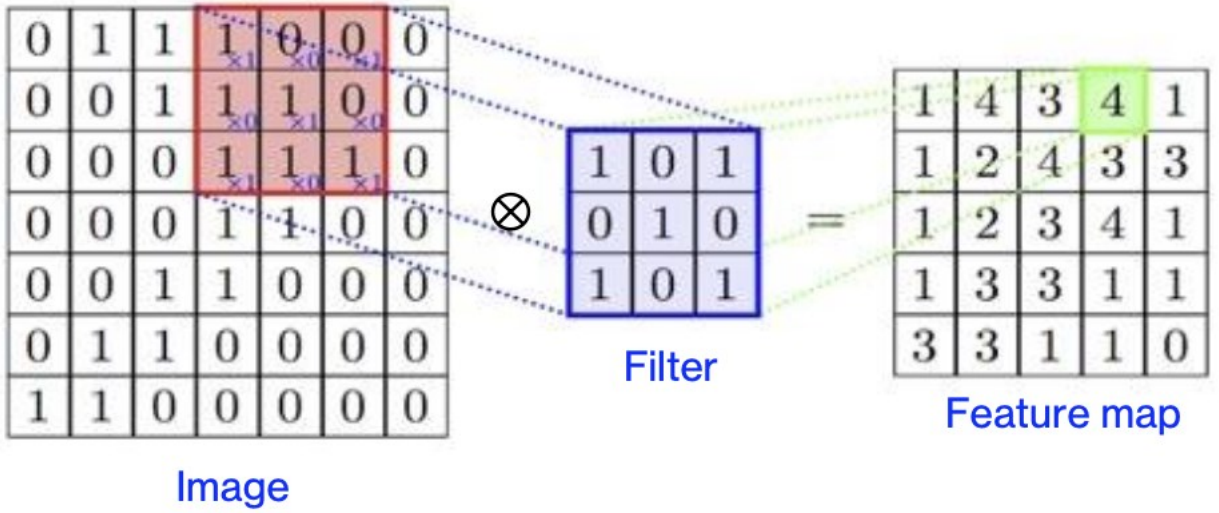
Left: A regular 3-layer dense NN

Right: A CNN arranges its neurons in 3 dimensions (width, height, depth). Every Layer transforms the 3D input volume to 3D output volume of neuron activations. Input (red layer) is an image so its width and height would be the image's dimensions and depth would be 3 (Red, Green, Blue channels)

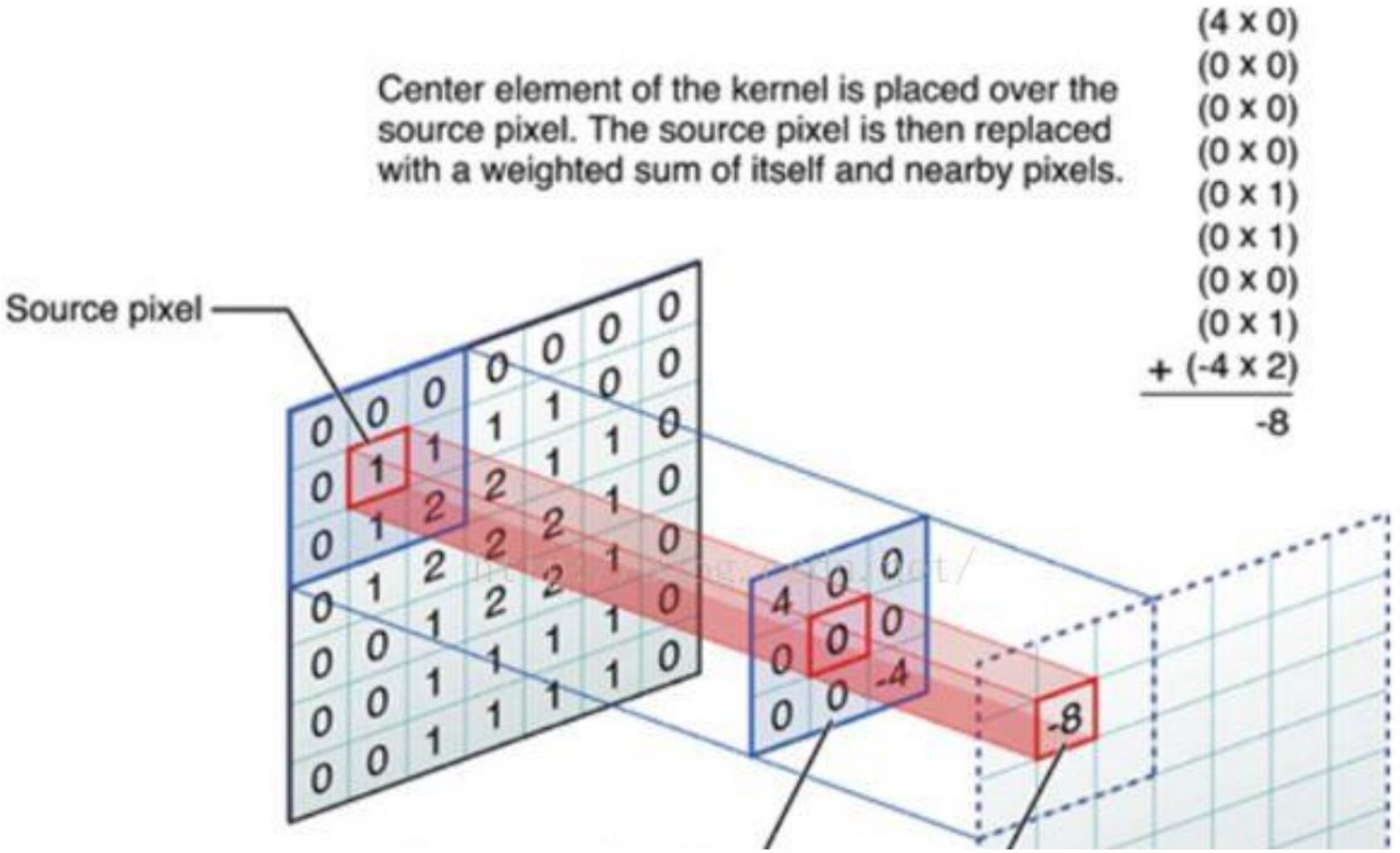
Image Filters / Convolutions

- Filters are small (3x3, 5x5, 7x7, …) arrays

- "Padding" **on** preserves image dimension,
- "Padding" **off** reduces by (n-1) pixels per plane for [n,n] filter



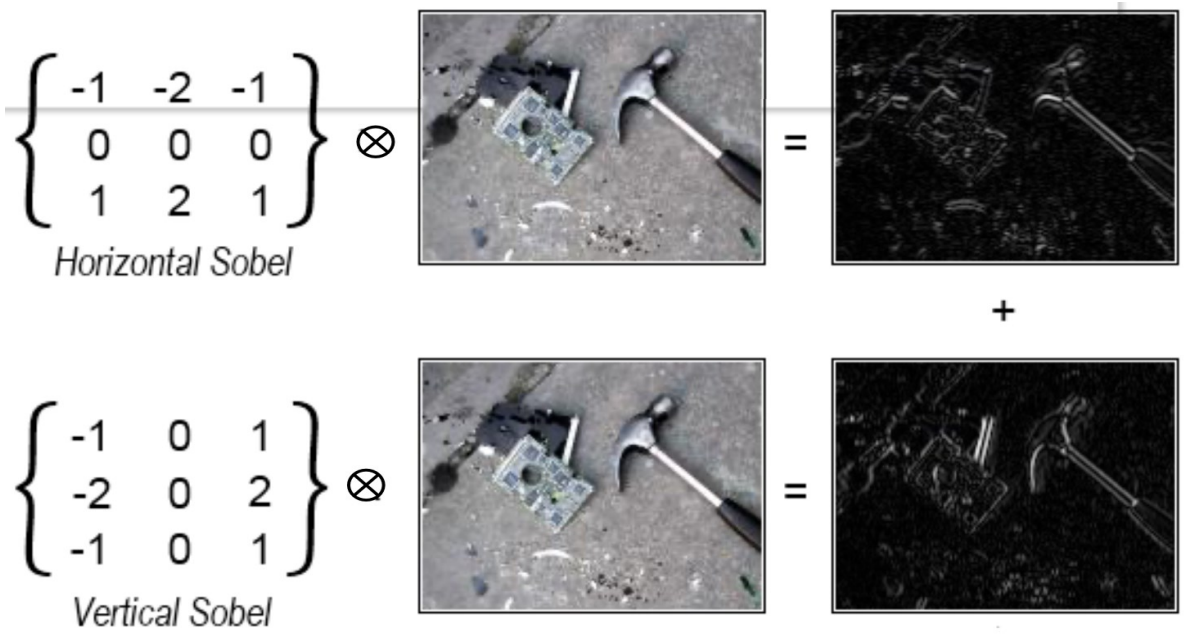
Convolutional Kernel



We use “Padding on” in the input volume with zeros in such way that the convolution layer does not alter the spatial dimensions of the input!

What do filters do?

- Different filters extract different features from image
- Larger filters have larger receptive area



2D Convolution (Image Processing)

Example of vertical edge filter (without padding)

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

$\otimes \quad G_x = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \quad =$

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

Example of vertical edge filter (without padding)

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

$\otimes \quad G_x = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \quad =$

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

Example of vertical edge filter (without padding)

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

$\otimes \quad G_x = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \quad =$

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

Example of vertical edge filter (without padding)

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

$\otimes \quad G_x = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \quad =$

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

Example of vertical edge filter (without padding)

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

$$\otimes \quad G_x = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \quad =$$

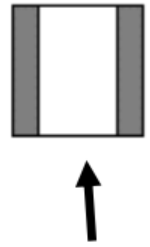
0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0

Example of vertical edge filter (without padding)

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

$\otimes \quad G_x = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} =$

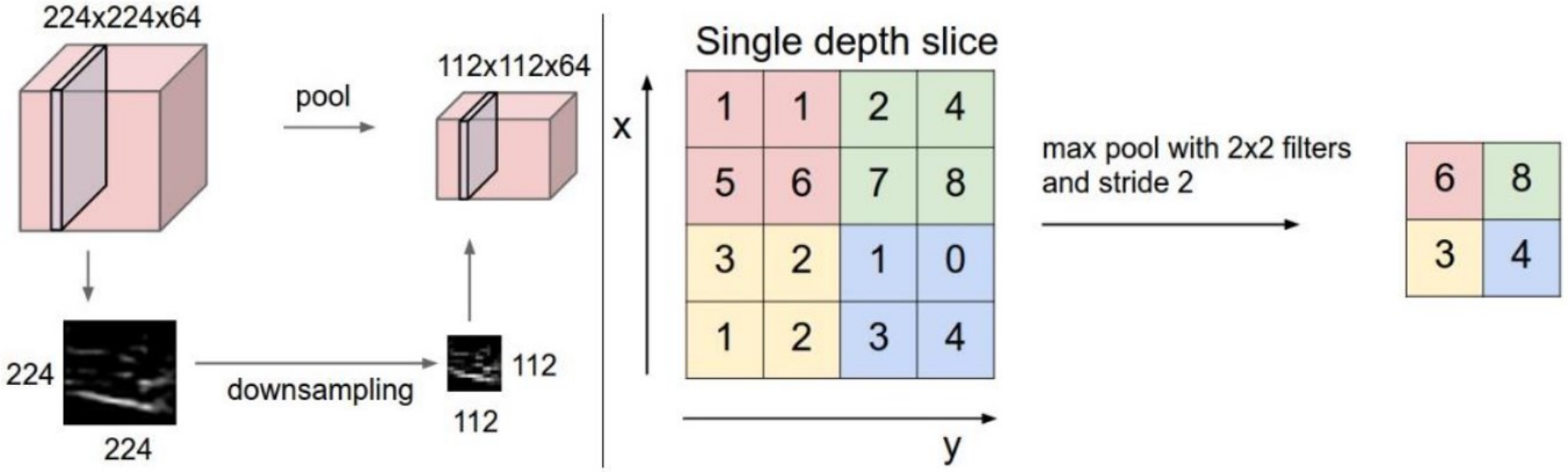
0	30	30	0
0	30	30	0
0	30	30	0
0	30	30	0



Strong response to vertical edge feature

Pooling Layer

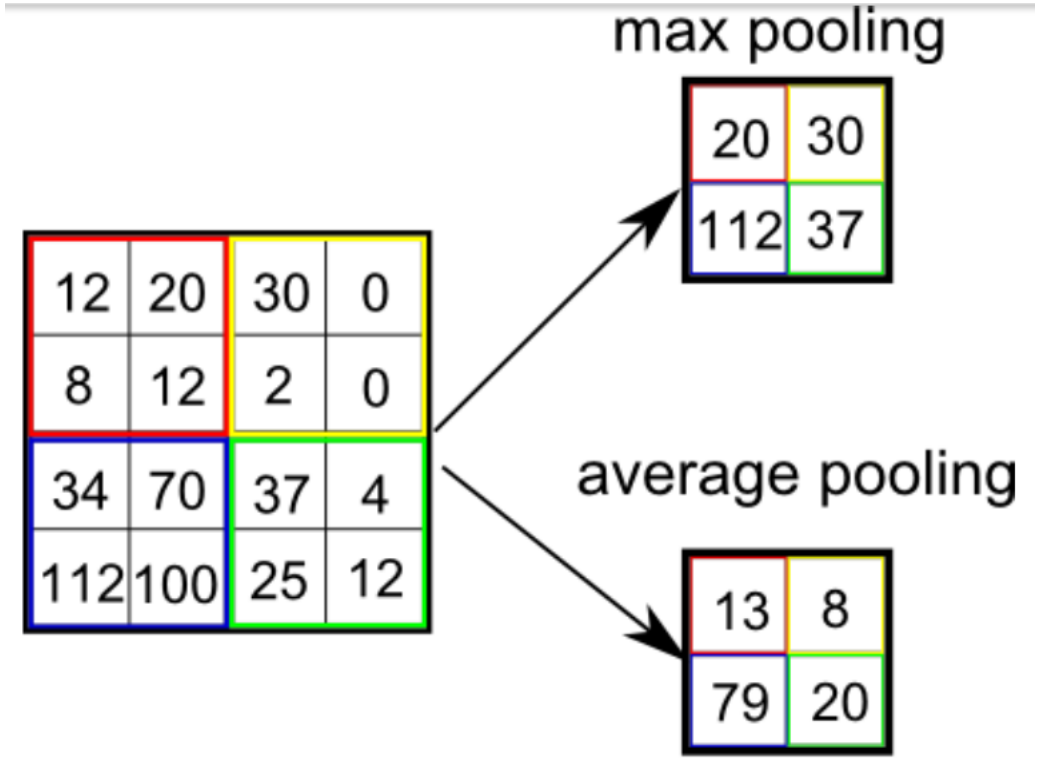
Pooling layer downsamples the volume spatially, independently in each depth slice of the input volume.



Left: The input volume of size [244x244x64] is pooled with filter size 2, stride 2 into output volume of size [112x112x64]. The volume depth is preserved.

Right: The most common downsampling operation is max, giving rise to **max pooling**, here shown with stride 2. Each max is calculated over 4 numbers (creates little 2x2 square)

Pooling



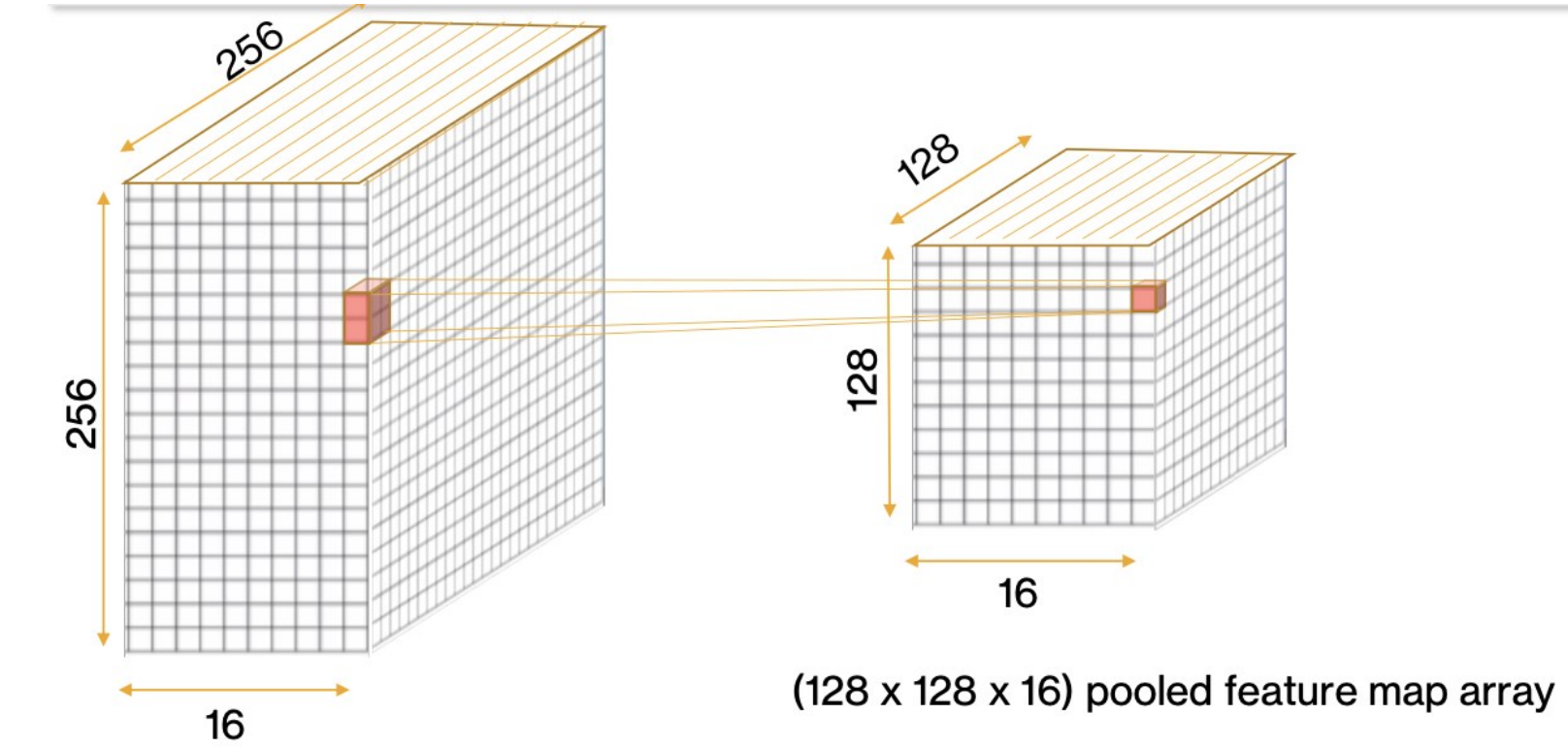
Difference between Max and Average Pooling

Stride controls how the filter works around the input volume.

In the example we had, the filter works around the input volume by shifting two units at a time.

Pooling

Pooling a (256x256x16) array with stride 2 and [2,2]



Why Pooling?

Why not just connect to a dense layer to classify features?

- **Dimensionality:** Connecting 64 feature maps of 256x256 pixels to a 1024 neuron dense layer would mean ≈ 4*109 weights.

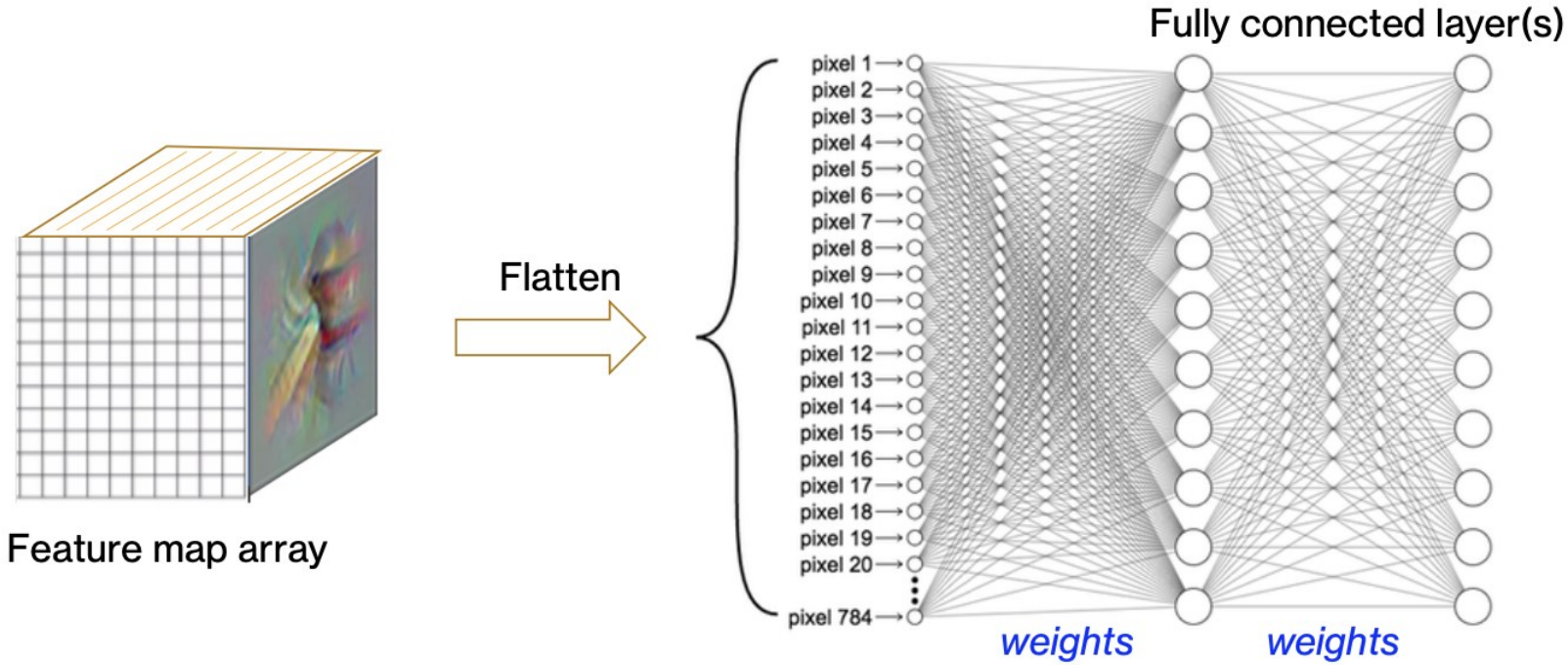
- Pooling reduces **sensitivity to position and size of feature**, i.e. introduces spatial and scale invariance – good for some applications.

Effect of Repeated Convolution & Pooling

- Hidden layers are composed of different arrangements of alternating convolution + activation and pooling layers.
- The first layer of filters identifies very basic features, like edges (vertical, horizontal, tilted), curves, or dark/light regions.
- Repeated pooling and filtering lets ConvNet build and identify more complex groups of features from low-level geometric building blocks.
- Vastly improves discrimination and recognition of complex patterns.

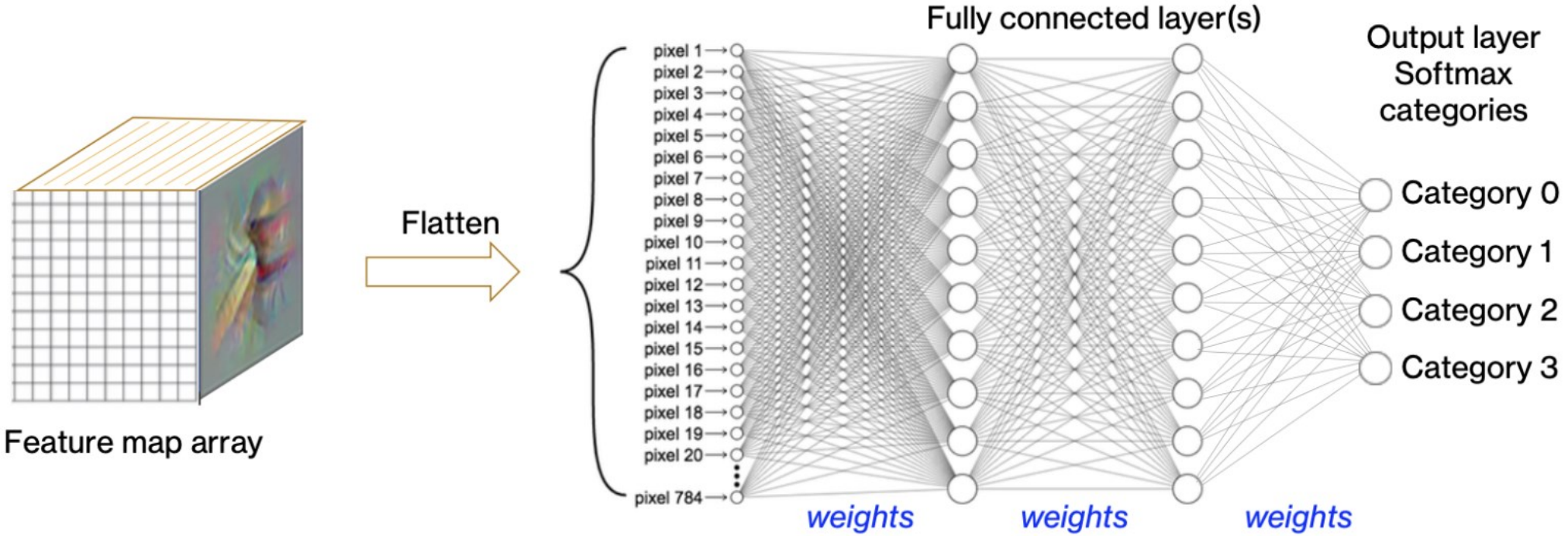
Fully Connected Dense Layers

- Output of last conv layer, flattened and connected to dense layer
- Maybe several dense layers with full neural interconnection
- These layers recognize patterns coming from feature maps



Output Layer

- Final layer of output neurons
- Each corresponds to one category in training/validation and (hopefully) test data



Cross-Entropy Loss

Used when the target output represents a probability distribution

e.g., a single output unit that indicates the classification decision (yes, no) for an input

Output $y \in [0, 1]$ denotes Bernoulli likelihood of class membership

Target t indicates true class probability (typically 0 or 1)

Note: single value represents probability distribution over 2 alternatives

The Categorial Cross Entropy for C class has following formula:

$$L = -\sum_{i=1}^{C} t_i \log y_i$$

For binary classification, we have following formula (C = 2):

$$L = -t \log y + (t-1)\log(1-y)$$

Categorical Outputs

Each input can belong to one category
  y_j denotes the probability that the input's category is j

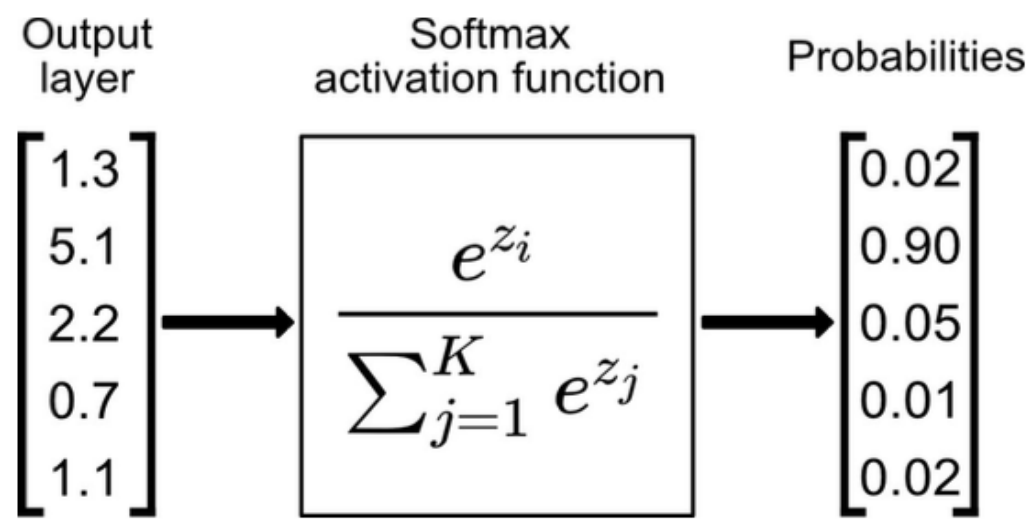
To interpret y as a probability distribution over the alternatives:

$$\Sigma_j y_j = 1 \text{ and } 0 \leq y_j \leq 1$$

Activation function:

$$y_j = \frac{e^{zi}}{\sum_{j=1}^{K} e^{zj}}$$

- Exponentiation ensures nonnegative values
- Denominator ensures sum to 1



Known as **softmax**, and formerly, Luce choice rule

End of Session 4