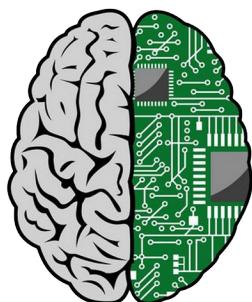


# ML für 10€ Workshop

**ML für 10€**  
**Mit ESP32-CAM und Edge Impulse**



**EDGE  
IMPULSE**



**Workshop zu den  
Labortagen 2023**  
Mark Hoffmann, B. Eng.



# ML für 10€ Workshop

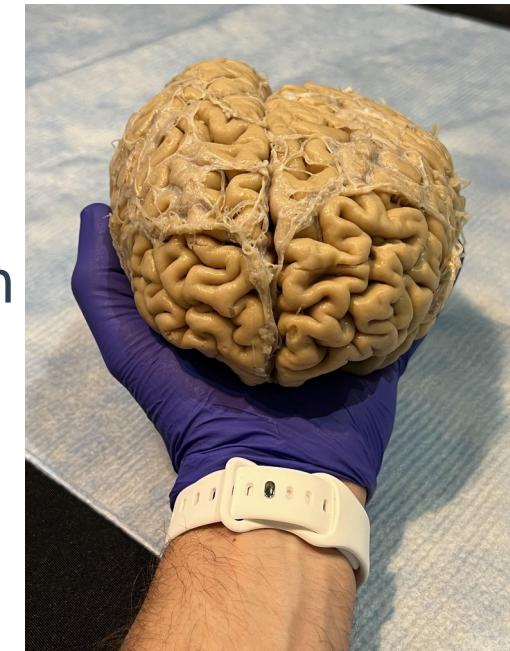
## Inhalt

- KI & Machine Learning
- Training vs. Inferenz
- Frameworks & Hardware
- Klicken vs. Coden
- Meine Master Thesis
- Vorstellung Bausatz (ESP32-CAM)
- Abarbeiten der Anleitung
- Weitere Anwendungen

# ML für 10€ Workshop

## Menschliche Intelligenz

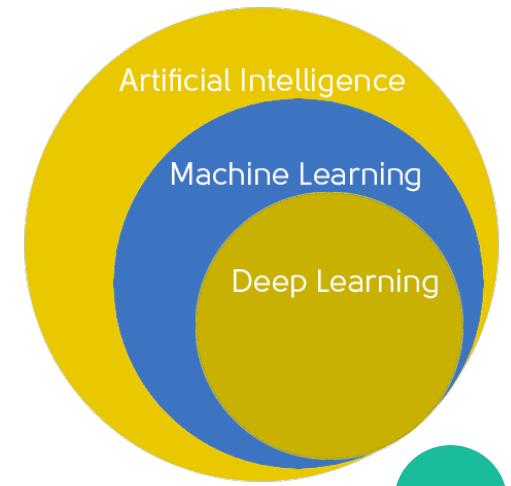
- Betrifft **Problemlösungsfähigkeiten**
- Messbar nicht nur per „üblichem“ **IQ-Test**:  
**Einstein, Mozart, Da Vinci**;  
aber auch Empathie, Humor, Kommunikation,  
Ethik und Moral, Überlebensfähigkeit,  
Gedächtnis, Lernen aus Fehlern, Körperkoord.
- Gehirn ist **komplexestes Objekt** im Universum
- 100 Milliarden Neuronen, Billiarden Synapsen
- **Gewicht 1,3 kg**, fettestes Organ im Körper;  
Konsistenz wie **Wackelpudding**
- Ausfallerscheinungen, Emotionen, Fehler



# ML für 10€ Workshop

## Unterschied: KI, ML, DL

- **Künstliche Intelligenz (KI)**: Systeme, die menschenähnliche Intelligenz nachahmen
- **Machine Learning (ML)**: Nutzung von Algorithmen und Modellen, Computer aus Daten lernen zu lassen - ohne explizite Programmierung
- **Deep Learning (DL)**: Spezielle Form des ML, basiert auf tiefen neuronalen Netzwerken - gut für Daten wie Bilder und Texte geeignet



# ML für 10€ Workshop

## Aktuelle KI Dienste

### Persönlich relevante Beispiele

- **ChatGPT** & Bard - (textuelle Fragen)
- DALL-E, Stable Diffusion, Midjourney - (Bildgenerierung)
- GitHub Copilot - (Programmierhilfen)
- Siri, **Alexa**, Google Assistant - (Personal Assistants)
- Fast alle Autohersteller - **Tesla** - (Autonomes Fahren)
- Computer aided diagnosis - **Fotofinder** - (Hautläsionen)
- **Skynet** bei Terminator und die Agenten bei Matrix - (**im Film:** menschenfeindliche Intelligenzen mit **Zugriff auf das Militär**)

# ML für 10€ Workshop

## Gefahren bei KI

- Fehlende Ethik und Moral
- Vorurteile und Diskriminierung
- Unvorhersehbarkeit, fehlende Nachvollziehbarkeit
- Arbeitsplatzverluste, Abhängigkeit
- Überwachung, Privatsphäre, Datenschutz
- Verlust von Kontrolle, autonome Waffen, Existenzialrisiko
- Verschiedene Betrugsarten

# ML für 10€ Workshop

## Bletchley Park

- Arbeitsstätte von Alan Turing, Entschlüsselung der Enigma, heute Museum
- **Global AI Safety Summit (1.11.23-2.11.23)**
- Teilnahme 28 Länder (inkl. China), Kamala Harris, Elon Musk, Ursula von der Leyen
- EM: „Niemand wird mehr arbeiten müssen.“
- GCHQ-Chefin: „Wir wissen nicht, was passieren wird, aber wir planen für das Schlimmste.“



# ML für 10€ Workshop

## Gefahren bei KI

### Beispiel: Paperclip Maximizer (Gedankenexperiment)

- KI hat „absurdes“ Ziel
- Unterordnung von allen sonstigen Zielen
- Kein(e) Human(s) in the loop, gerät außer Kontrolle
- Aber auch Einbindung in die menschliche Umgebung mit graduelltem Übergang



YouTube Video→ [https://youtu.be/3mk7NVFz\\_88](https://youtu.be/3mk7NVFz_88)

# ML für 10€ Workshop

## ML „im kleinen Rahmen“ (hier und heute)

- **Theorie:** Training vs. Inferenz
- **CNNs:** Convolutional Neural Networks
- **Thesis:** Hautkrebsdetektion
- **Workshop Objektklassifikation:** Unterscheidung von **zwei Objekten** mit Hilfe von **ESP32-CAM** Boards  
(bei Einsatz von Arduino Libraries)

# ML für 10€ Workshop

## Training vs. Inferenz

- **Training** nimmt Unmengen an Daten (z.B. Bilder) und versucht diese „wieder und wieder“ richtig zu klassifizieren
- **Inferenz** (Englisch „to infer“ = **schlussfolgern**) nimmt z.B. „ein Bild“ und klassifiziert es

# ML für 10€ Workshop

## Training

- Benötigt **Unmengen** an **Trainingsdaten** (Bilder >25.000 z.B. bei Hautkrebs (Thesis))
- Seit Anfang 2000er **Big Data** (Exabytes an Daten) nutzbar
- Viel **Matrizenmultiplikation** nötig / paralleles Abarbeiten
- Einsatz von **GPUs & TPUs** (statt **CPUs**), da diese Matrizenmultiplikation schneller vollführen können
- Empfehlung Betrieb im **Rechenzentrum** (statt GPU und TPU zu 1000€+)
- **Dauert** (meist zwischen 20 Minuten und mehreren Tagen)

# ML für 10€ Workshop

## Convolutional Neural Networks - CNNs

- **Convolution** - zu Deutsch **Faltung** - **kein Origami (!)**
- Gut geeignet für **Erkennung von Objekten**, Mustern in **Bildern**
- **Bildsegmentierung** – Unterteilung in verschiedene Regionen, z.B. zur Tumorerkennung
- Gestenerkennung / **Posenerkennung**
- Mustererkennung in Zeitreihen, z.B. zur **Spracherkennung**
- **Videoklassifikation**, z.B. Polizeikameras bei Stau

# ML für 10€ Workshop

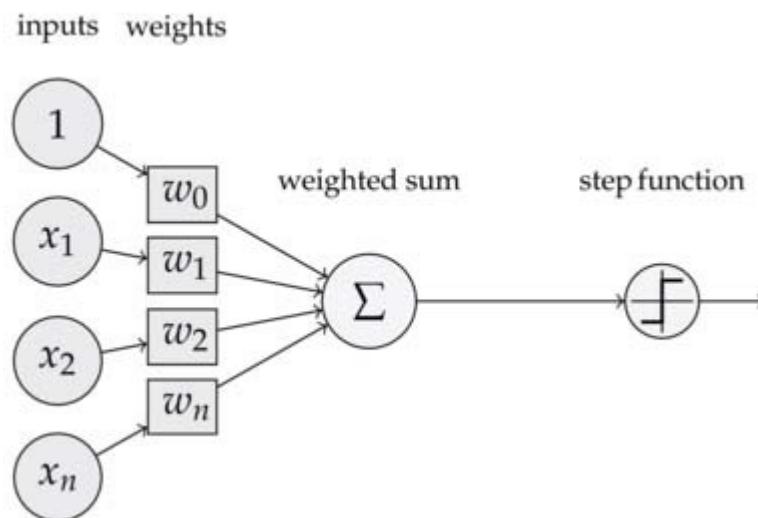
## Theorie zu CNNs

### Konzepte

- **DNN:** Zugang über das **Perzeptron** und **künstliche Neuronen** mit **Gewichten** - ähnlich dem menschlichen Gehirn
- oder:
- **CNN:** Zugang per **Convolutions (Faltungen)** mit **Kerneln**

# ML für 10€ Workshop

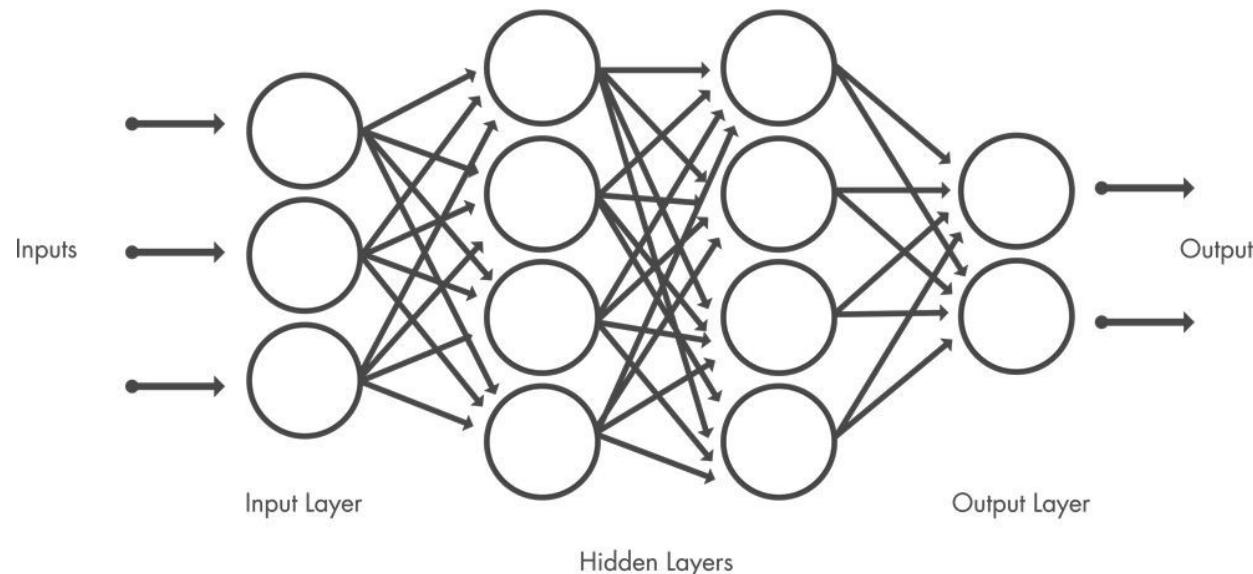
## Das Perzeptron



**Künstliches Neuron**  
Frank Rosenblatt - 1958

# ML für 10€ Workshop

## Deep Neural Network



Zu sehen: Eine **Eingabeschicht**,  
eine **Ausgabeschicht**  
und viele **verborgenen Schichten** dazwischen

Nicht zu sehen: **Gewichte, Summen, Aktivierungsfkt.**

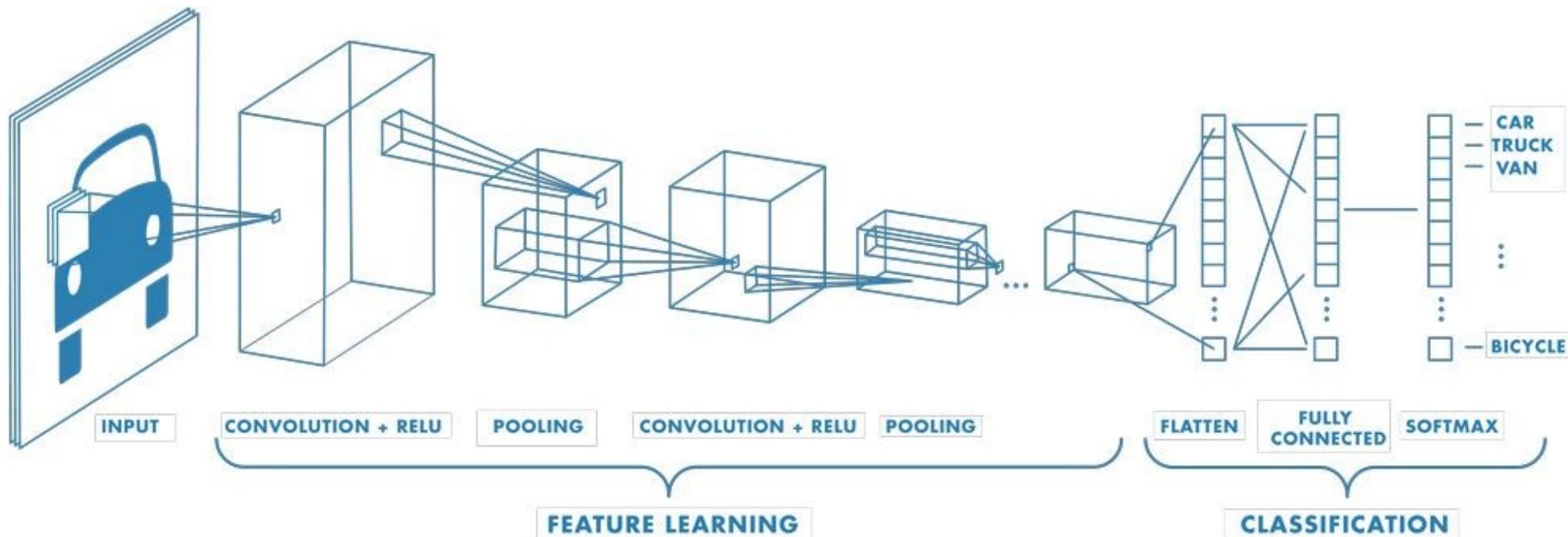
# ML für 10€ Workshop

## Training – CNNs

- Ein CNN besteht aus Dutzenden oder Hunderten von **versteckten Schichten**
- Jede **Schicht** lernt **verschiedene Merkmale** eines Bildes zu erkennen
- Die Ausgabe eines jeden geschichteten Bildes als **Eingabe für die nächste Schicht verwendet**
- Es werden Filter mit **sehr einfachen Merkmalen**, wie z. B. Helligkeit und Kanten angewendet - mit zunehmender Komplexität bis hin zu Filtern von Merkmalen, die das Objekt **eindeutig definieren**

# ML für 10€ Workshop

## CNNs - Convolutional Neural Network



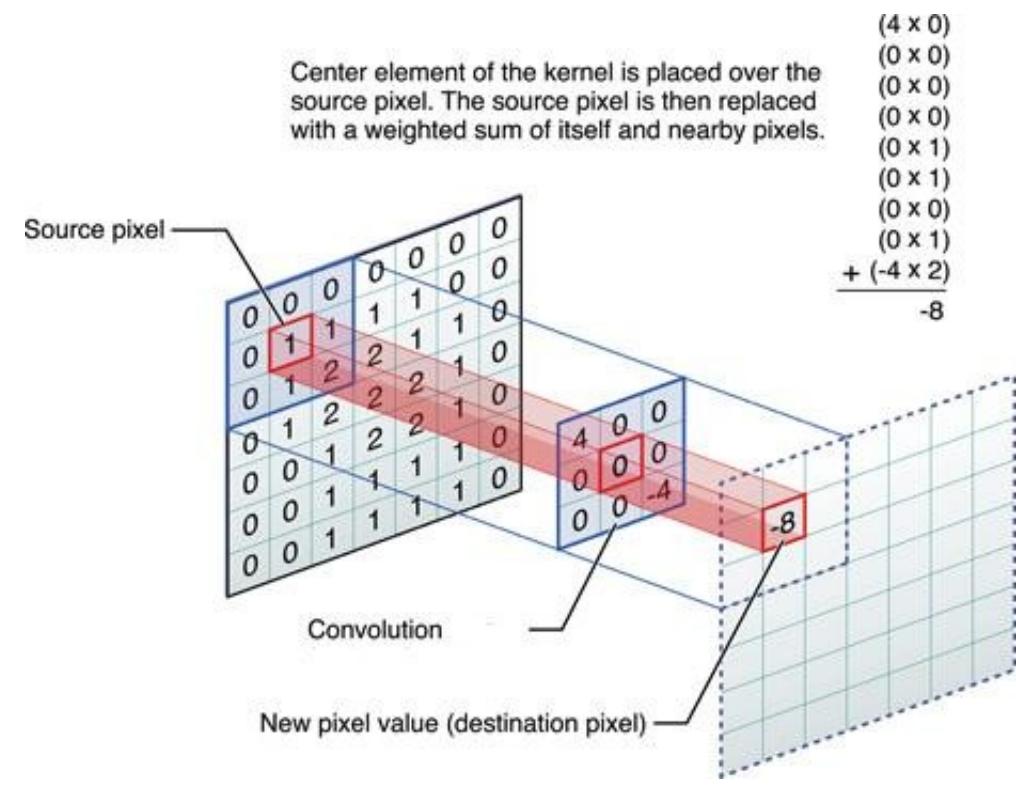
Hier zu erkennen - die Arbeitsschritte: **Convolution**, **ReLU**, **Pooling**, **Flattening**, **Fully Connected** und **Softmax**

# ML für 10€ Workshop

## CNNs – Convolutional Neural Networks

### Faltung

- Mathematische Operation
- Anwendung von **Kerneln** bzw. Filtern auf ein Bild
- „More than meets the eye“  
→ bringt mehr als man auf den ersten Blick erwartet
- Ermöglicht das **Finden** von Strukturen in größeren Bildern



(ausführlicher erklärt → [is.gd/ayanix](http://is.gd/ayanix))

# ML für 10€ Workshop

## CNNs - Convolutional Neural Networks

### Faltung

- Ein S/W Bild hat Pixelwerte mit Farbe im Bereich 0 (Schwarz) bis 255 (weiß)
- Ein R/G/B Bild hat drei Kanäle analog dazu

0	2	15	0	0	11	10	0	0	0	0	9	9	0	0	0	0
0	0	0	4	60	157	236	255	255	177	95	61	32	0	0	29	
0	10	16	119	238	255	244	245	243	250	249	255	222	103	10	0	
0	14	170	255	255	244	254	255	253	245	255	249	253	251	124	1	
2	98	255	228	255	251	254	211	141	116	122	215	251	238	255	49	
13	217	243	255	155	33	226	52	2	0	10	13	232	255	255	36	
16	229	252	254	49	12	0	0	7	7	0	70	237	252	235	62	
6	141	245	255	212	25	11	9	3	0	115	236	243	255	137	0	
0	87	252	250	248	215	60	0	1	121	252	255	248	144	6	0	
0	13	113	255	255	245	255	182	181	248	252	242	208	36	0	19	
1	0	5	117	251	255	241	255	247	255	241	162	17	0	7	0	
0	0	0	4	58	251	255	246	254	253	255	120	11	0	1	0	
0	0	4	97	255	255	255	248	252	255	244	255	182	10	0	4	
0	22	205	252	246	251	241	100	24	113	255	245	255	194	9	0	
0	111	255	242	255	158	24	0	0	6	39	255	232	230	56	0	
0	218	251	250	137	7	11	0	0	0	2	62	255	250	125	3	
0	173	255	255	101	9	20	0	13	3	13	182	251	245	61	0	
0	107	251	241	255	230	98	55	19	118	217	248	253	255	52	4	
0	18	146	250	255	247	255	255	249	255	240	255	129	0	5	0	
0	0	23	113	215	255	250	248	255	255	248	248	118	14	12	0	
0	0	6	1	0	52	153	233	255	252	147	37	0	0	4	1	
0	0	5	5	0	0	0	0	0	0	14	1	0	6	6	0	

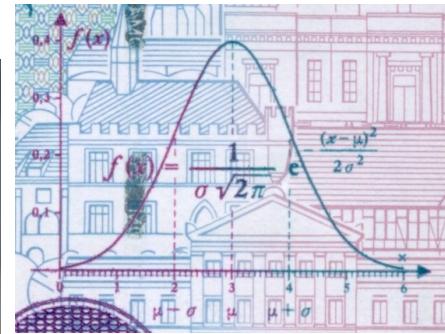
Farbwerte

# ML für 10€ Workshop

## CNNs - Convolutional Neural Networks

### Faltung

- Kernel ungerade, z.B. 3x3, 5x5, 7x7, ...
- Wird Zeile für Zeile über das Bild geschoben
- **Gauss** (Weichzeichnung)

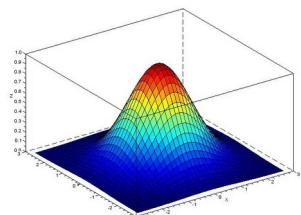


Gauss'sche  
Normalverteilung



10 DM Schein

0	0	0	5	0	0	0
0	5	18	32	18	5	0
0	18	64	100	64	18	0
5	32	100	100	100	32	5
0	18	64	100	64	18	0
0	5	18	32	18	5	0
0	0	0	5	0	0	0



Gauss Kernel

# ML für 10€ Workshop

## CNNs – Convolutional Neural Networks

### Faltung

- **Kerneltypen:**

Gauss: Weichzeichnung

Canny Edge bzw. **Sobel**: Kantendetektion

1	0	-1
2	0	-2
1	0	-1

Sobel Kernel  
(3x3)



- **Kernel Matching:** Bestimmen Kernel im Bild **wiederfinden!**

# ML für 10€ Workshop

## Faltungsanwendung - Bsp.: PA2

Per **Kantendetektion** und **FPGA**:  
**Karten** erkennen lassen

- Der **Dr. Bauer Trick** für die PA2:
- Kamerabilder → **Kanten detektieren** lassen
- Vier Orientierungen von Linien - **Faltung** auf ein Schwarz-Weiß Bild
- Ein **Kernel-Match** (z.B. 5 Pixel weiß in einer Reihe) ergibt einen Punkt / Hit
- Alle Hits zusammenzählen und gemäß „**Regelwerk**“ aufschlüsseln ... et voila ... **Erkennung** (LED an)



Cyclone IV FPGA  
Board



OV7670

# ML für 10€ Workshop

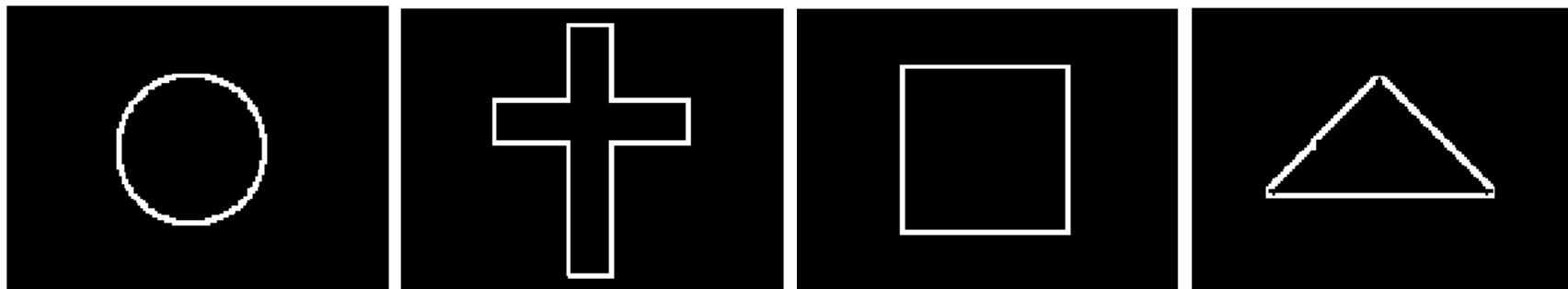
## Faltungsanwendung - Bsp.: PA2

Per **Kantendetektion** und **FPGA**:  
**Spielkarten** erkennen lassen

- Der **Dr. Bauer Trick** für die PA2:

- „Hit“ Counting
- Regelwerk (empirisch)
- LED am FPGA geht an

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



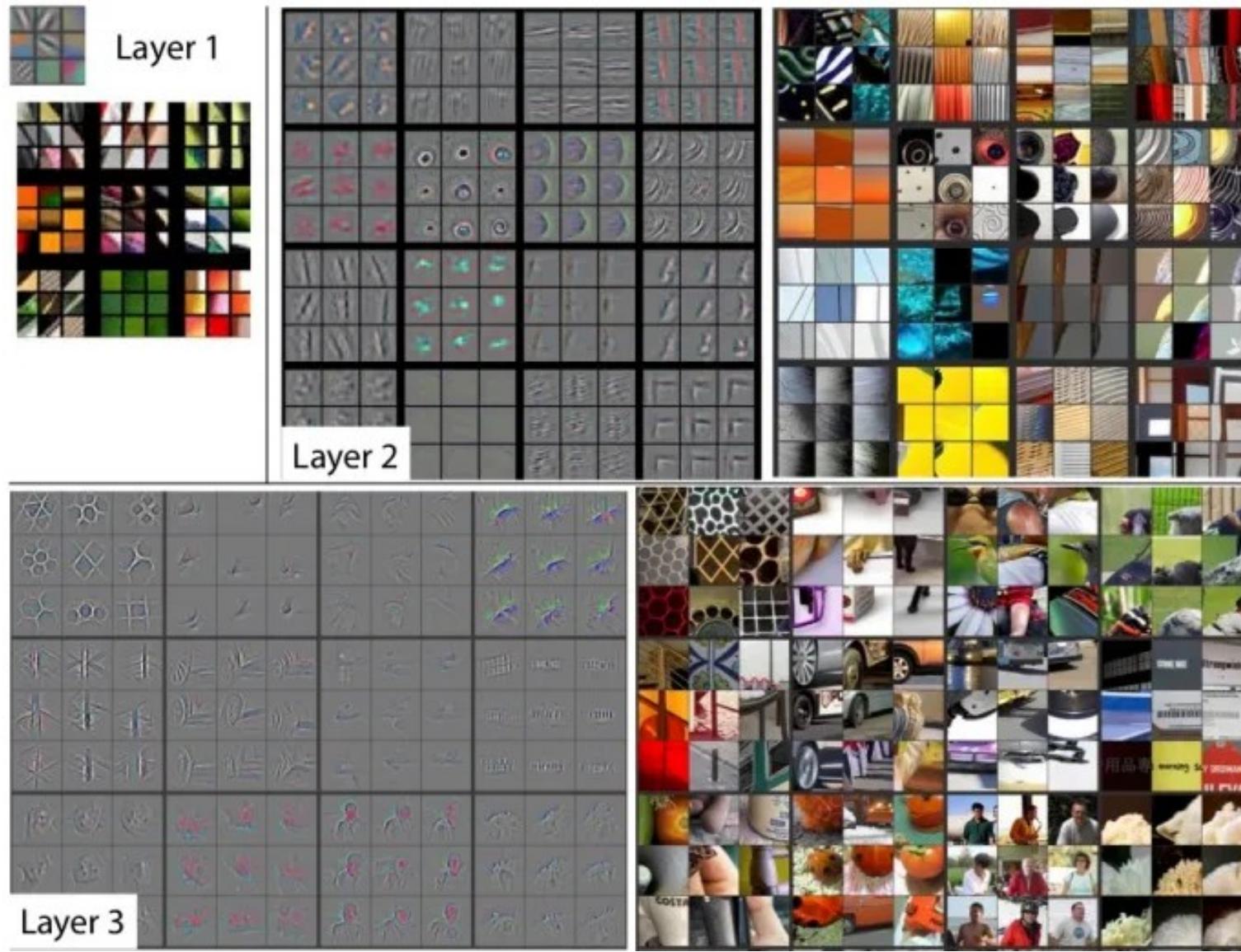
# ML für 10€ Workshop

## Weiterführung der PA2:

### CNNs - Convolutional Neural Networks

- Die „**große Magie**“ bei CNNs ist, dass die Kernel je Schicht „**von alleine**“ gefunden werden und aufeinander aufbauen ...
- Am **Beispiel** von folgenden Schichten ...

# ML für 10€ Workshop



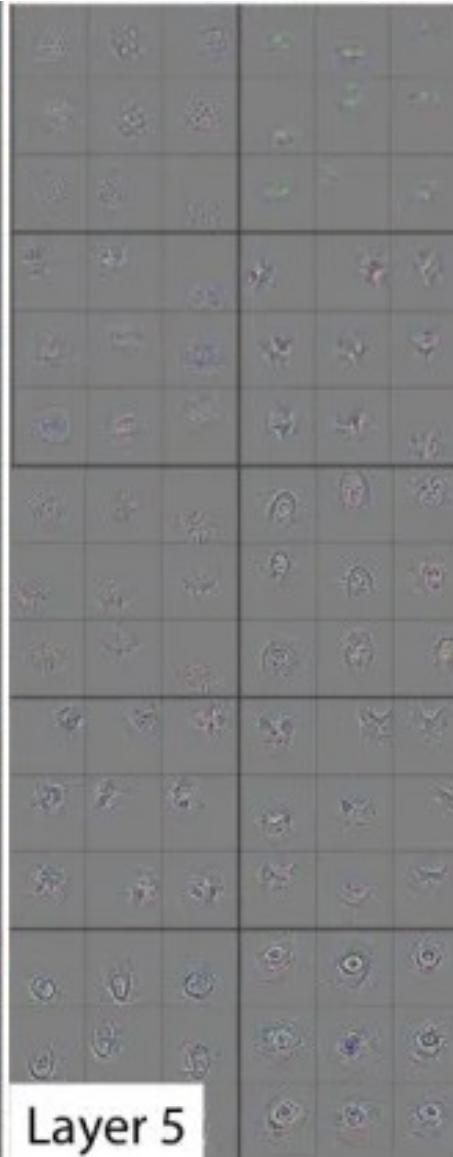
# ML für 10€ Workshop



Layer 4



Layer 5



# ML für 10€ Workshop

## CNNs – Convolutional Neural Networks

Zurück zu den **CNN Arbeitsschritten**:

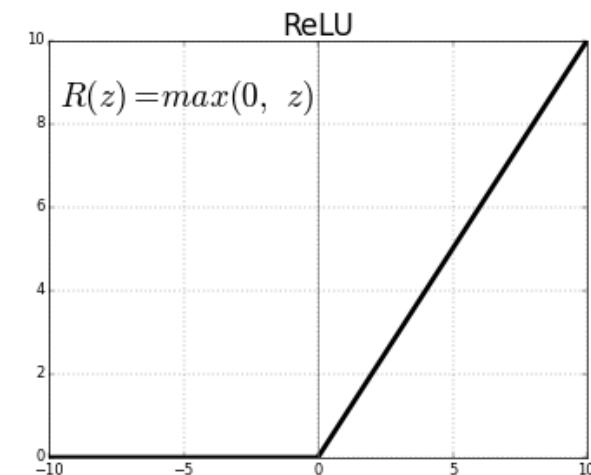
- ReLU
- Pooling
- Flattening
- Fully Connected
- Softmax

# ML für 10€ Workshop

## CNNs - Convolutional Neural Networks

### ReLU - Rectified Linear Unit - Aktivierungsfunktion

- Zuordnung **negativer Werte zu null; Beibehaltung positiver Werte**
- Nur die aktivierte Merkmale werden in die nächste Schicht übertragen
- **Nichtlinearität** einführen; bei mehrerer Matrizenmultiplikationen hintereinander, bliebe das Ergebnis linear
- → Schnelleres und effektiveres Training

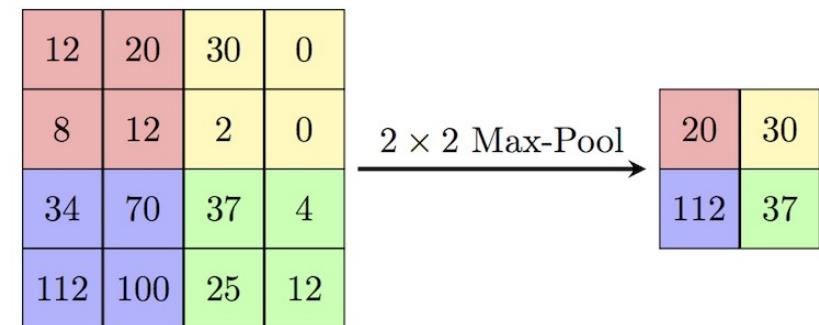


# ML für 10€ Workshop

## CNNs - Convolutional Neural Networks

**Pooling** - aus dem Englischen „sich vereinigen, zusammenschließen, bündeln“

- Hier: kleine Bereiche im Bild werden zusammenfasst und die Datenmenge reduziert
- Erzeugt **Invarianz** gegenüber **kleinen Verschiebungen** von Strukturen im Bild
- Wichtige Merkmale werden beibehalten

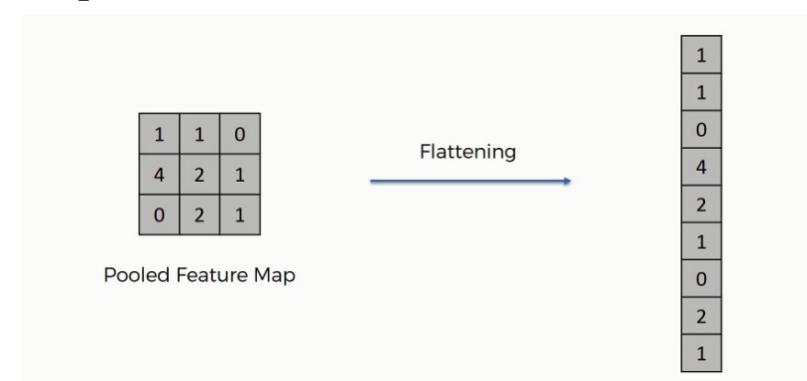


# ML für 10€ Workshop

## CNNs - Convolutional Neural Networks

### Flattening

- Dabei wird eine Datenstruktur in einen eindimensionalen Vektor umgewandelt
- Z.B. ein Farbbild, normalerweise in drei Kanälen (RGB) und mit Höhe und Breite repräsentiert ...
- ... das Flattening **wandelt die Pixelwerte aus diesen drei Kanälen und allen Zeilen und Spalten des Bildes in eine lange Liste**

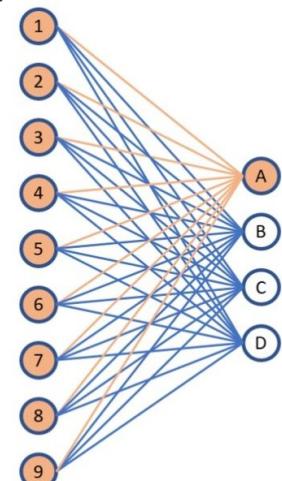


# ML für 10€ Workshop

## CNNs - Convolutional Neural Networks

### Fully Connected Layer

- Die vollständig verbundene Schicht ist dafür verantwortlich, die Merkmale oder Informationen, die aus den vorherigen Schichten stammen, miteinander zu verknüpfen, um eine **endgültige Ausgabe** zu erzeugen, also **Klassifikation** bei Bildern
- Diese Schicht besteht aus **Neuronen, die mit allen Neuronen der vorherigen Schicht verbunden** sind
- In einem Klassifikationsproblem kann der Fully Connected Layer die **Wahrscheinlichkeiten für verschiedene Klassen** oder Kategorien ausgeben
- Dieser Schritt führt zur **endgültigen Klassifizierung** des untersuchten Objekts

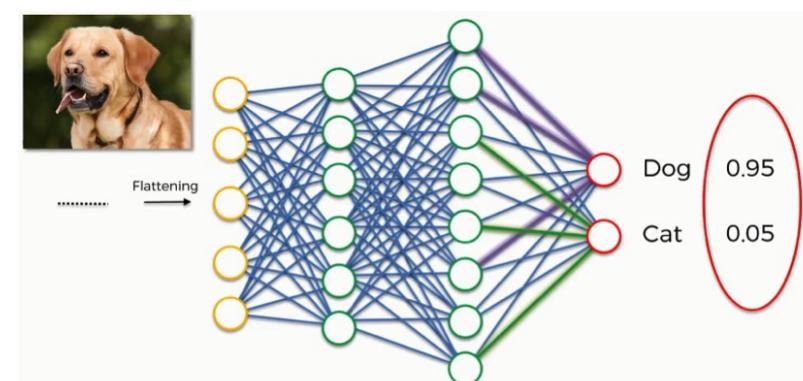


# ML für 10€ Workshop

## CNNs - Convolutional Neural Networks

### Softmax

- Diese Funktion nimmt einen Vektor von Werten und transformiert ihn so, dass **die resultierenden Werte zwischen 0 und 1 liegen und sich zu 1 summieren**
- Zeigt die **Wahrscheinlichkeiten** von verschiedenen Klassifikationsklassen
- Die Klasse mit der **höchsten Wahrscheinlichkeit** wird ausgewählt



# ML für 10€ Workshop

## ML-Modelle

- **Beispiele:** ResNet, Inception, VGG, AlexNet, Xception, MobileNet (v1 & v2), LeNet, ...
- **Schichten** - entweder komplett selber aufsetzen (**Learning from Scratch**), 1:1 aus dem Internet nehmen oder per **Transfer Learning** andere Klassifikationsklassen „hinten anhängen“, z.B. **Melanom** und Standard **Nävus** an ImageNet; auch **Einfrieren** von Schichten möglich

# ML für 10€ Workshop

## Datensätze

- **COCO** mit 80 Objektklassen  
(über 118.287 Trainingsbilder)
- **ImageNet** mit 1000 Objektklassen  
(1.281.167 Trainingsbilder)
- **HAM 10000** mit 7 Diagnosen  
(10.015 Trainingsbilder)
- **ISIC 2019** mit 9 Diagnosen  
(25.332 Trainingsbilder)



## DOCUMENTATION

### Conv

The convolution operator consumes an input tensor and a filter, and computes the output.

#### ATTRIBUTES

##### auto\_pad: string

auto\_pad must be either NOTSET, SAME\_UPPER, SAME\_LOWER or VALID. Where default value is NOTSET, which means explicit padding is used. SAME\_UPPER or SAME\_LOWER mean pad the input so that the output size match the input. In case of odd number add the extra padding at the end for SAME\_UPPER and at the beginning for SAME\_LOWER. VALID mean no padding.

##### dilations: int[]

dilation value along each axis of the filter.

##### group: int

number of groups input channels and output channels are divided into.

##### kernel\_shape: int[]

The shape of the convolution kernel. If not present, should be inferred from input W.

##### pads: int[]

Padding for the beginning and ending along each axis, it can take any value greater than or equal to 0. The value represent the number of pixels added to the beginning and end part of the corresponding axis. pads format should be as follow [x1\_begin, x2\_begin...x1\_end, x2\_end,...], where xi\_begin the number of pixels added at the beginning of axis  $i$  and xi\_end, the number of pixels added at the end of axis  $i$ . This attribute cannot be used simultaneously with auto\_pad attribute. If not present, the padding defaults to 0 along start and end of each axis.

# ML für 10€ Workshop

## Inferenz

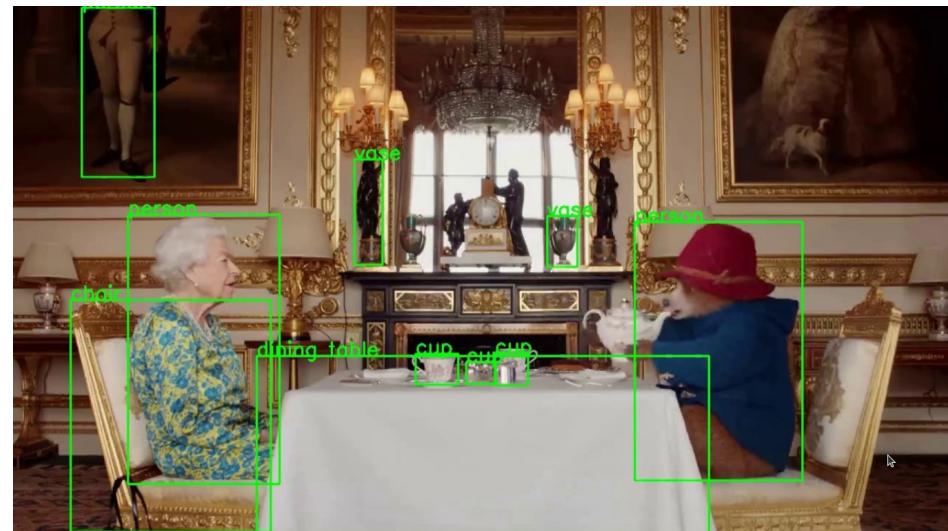
- Gerne eingesetzt bei **Edge Computing** und **IoT** (in Kameras)
- Dauer: Bruchteile von Sekunden / **Echtzeit**
- Z.B per Google Coral Edge TPU Beschleunigung  
(USB Erweiterung bei Raspberry Pis)
- Auch: **ESP32-CAM** ;-)

# ML für 10€ Workshop

## Inferenz - Beispiel

**YOLO v5 Algorithmus**  
„You only look once“)

- **Beispiel:**  
Die Queen und Paddington Bär  
(70 Jahre Thronjubiläum)
- Zu sehen: **Bounding Boxen**  
und **Klassifikationsergebnisse**
- **Zeigen!**



Code für ein beliebiges YouTube Video  
→ <https://is.gd/sufaba>

# ML für 10€ Workshop

## Frameworks

- **ML-Frameworks** - bei **Python** Einsatz:  
**TensorFlow** (Lite) (Google)  
Keras (jetzt integriert in TF)  
Pytorch (Facebook)  
Scikit-Learn (OS-Community)  
Caffe (Berkeley Vision and Learning Center)  
Auch: MXNet, Theano, CNTK, Chainer, DL4J, ...
- Bildverarbeitung - ebenso **Python** Einsatz:  
**OpenCV**, Scikit-Image

# ML für 10€ Workshop

## Hardware

- **Training:**  
GPUs & TPUs von Nvidia (CUDA Treiber)  
GPUs von AMD (ROCm Treiber)



Nvidia T4 TPU

- **Inferenz:**  
Google Coral (ASIC)  
Nvidia Jetson (SBC)  
Intel Neural Compute Stick



Nvidia Jetson

- **Lokal (\$\$\$):**  
Workstation (z.B. über die Hochschule)

Google Coral USB

- **Cloud (10 US-\$/Monat):**  
U.a. Google Colab Pro

# ML für 10€ Workshop

## Coden

- **Programmierung:**

Hauptsächlich Python mit **Jupyter Notebooks** statt in traditionellen IDEs (einzelne bestätigende Codeblöcke - **Zellen** statt **Fliesstext**)

- **Umgebung:**

Entweder: **Selber aufsetzen** auf lokalem PC (Anaconda)  
Oder: **Cloud Nutzung**, v.a. mit Google Colab (Pro) und Kaggle

# ML für 10€ Workshop

## Anaconda & Conda

### Lokale Lösung

- Einrichten einer ML Python Umgebung inkl. GPU / TPU Treiber über die Python Distribution Anaconda mit Paketmanagementssoftware Conda möglich (empfohlen!)
- Unterstützt: Linux, Windows, macOS
- **Ubuntu** Anleitung → <https://is.gd/foquser>, <https://is.gd/alamew>



ANACONDA®

# ML für 10€ Workshop

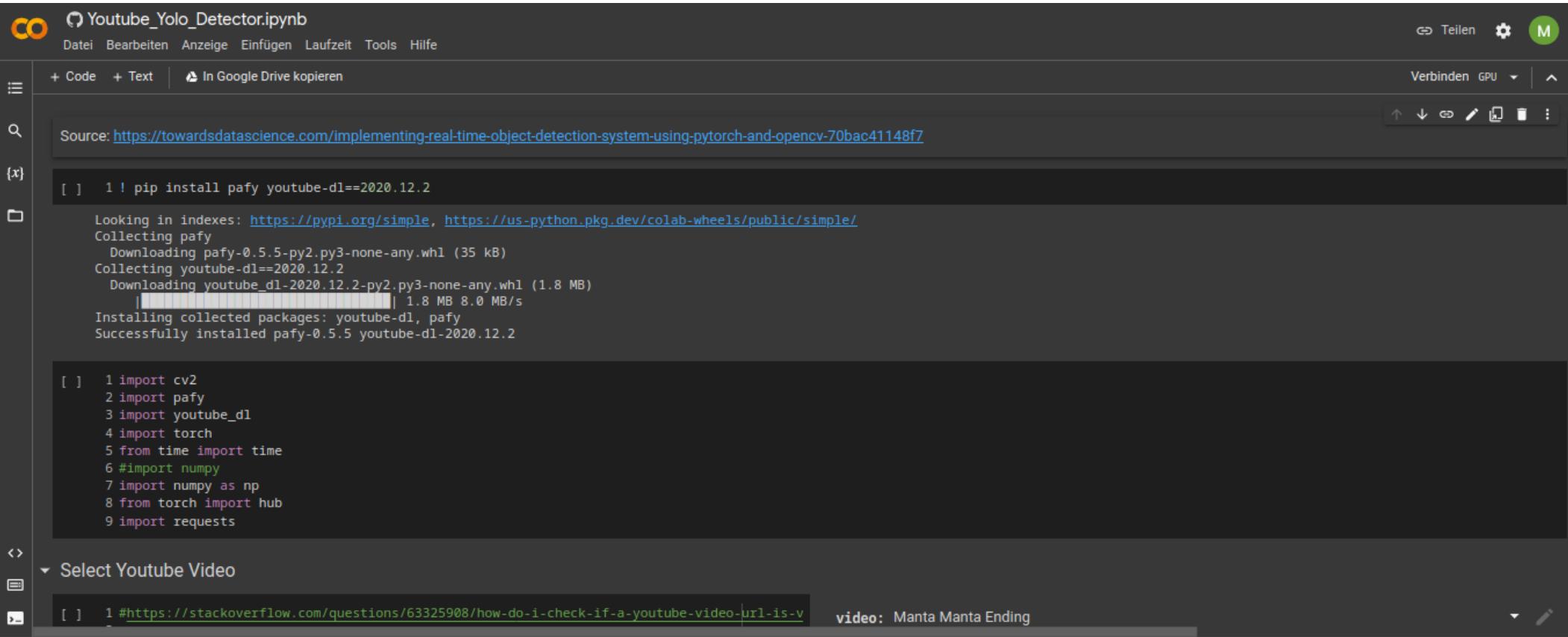
## Google Colab (Pro)

### Rechenzentrum Lösung

- Jupyter Notebooks (Python & R)
- Speichern bei Google Drive, Export zu GitHub
- **11€/Monat** (per Kreditkarte, bessere GPUs und TPUs, längere Laufzeit)
- Vollwertiges **aktuelles Ubuntu** (apt install ...; pip install ...)
- Python der Wahl mit „gefühlt“ weniger Abhängigkeitsproblemen

# ML für 10€ Workshop

## Google Colab (Pro)



The screenshot shows a Google Colab notebook titled "Youtube\_Yolo\_Detector.ipynb". The interface includes a toolbar with file operations like "Teilen" and "GPU", and a sidebar with navigation icons. The main area displays a code cell with the following content:

```
[ ] 1 ! pip install pafy youtube-dl==2020.12.2
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting pafy
  Downloading pafy-0.5.5-py2.py3-none-any.whl (35 kB)
Collecting youtube-dl==2020.12.2
  Downloading youtube_dl-2020.12.2-py2.py3-none-any.whl (1.8 MB)
    |██████████| 1.8 MB 8.0 MB/s
Installing collected packages: youtube-dl, pafy
Successfully installed pafy-0.5.5 youtube-dl-2020.12.2

[ ] 1 import cv2
2 import pafy
3 import youtube_dl
4 import torch
5 from time import time
6 #import numpy
7 import numpy as np
8 from torch import hub
9 import requests
```

Below the code cell, there is a section titled "Select Youtube Video" with a dropdown menu. A status bar at the bottom shows the URL "#https://stackoverflow.com/questions/63325908/how-do-i-check-if-a-youtube-video-url-is-v" and the message "video: Manta Manta Ending".

# ML für 10€ Workshop

## Paper / GitHub Beispiele

- Universitäres Umfeld
- Paper, z.B. über ResearchGate  
→ <https://www.researchgate.net> mit GitHub Beispielcode
- Paper zu ISIC - International Skin Imaging Collaboration mit Teilnahme an Contests, z.B. Medical Image Computing and Computer Assisted Intervention (MICCAI), October 2023 in Vancouver  
Dafür angeboten: Trainingsdatensätze (z.B. ISIC 2019 Challenge → <https://is.gd/icoxuku> > 25.000 Bilder von Hautläsionen)

# ML für 10€ Workshop

## Python Code für Hautkrebserkennung

Schritte im Training gemäß **Master Thesis**

- MobileNet V2 - Blumenbilder mit Google Coral  
→ <https://is.gd/ligeni>
- ISIC 2019 EfficientnetB0 (Probleme mit nicht ausbalanciertem Datensatz) → <https://is.gd/wavowi>
- Erst mal einfacherer Aufbau (MVP - Minimalprojekt - Ziffernerkennung)

# ML für 10€ Workshop

## Klicken

**Lösungen im Browser - ohne eigenes Coden:**

- **Google Teachable Machine**
- **Edge Impulse**
- Microsoft Azure
- Amazon AWS SageMaker
- Auch: Lobe, Runway ML, Nvidia DeepStream, Neuromation, DataRobot, RapidMiner, IBM Watson Studio

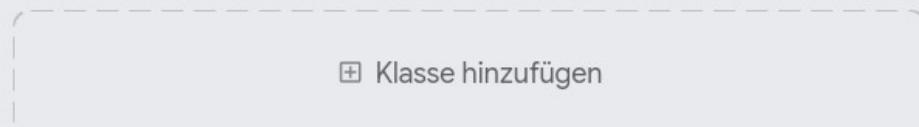
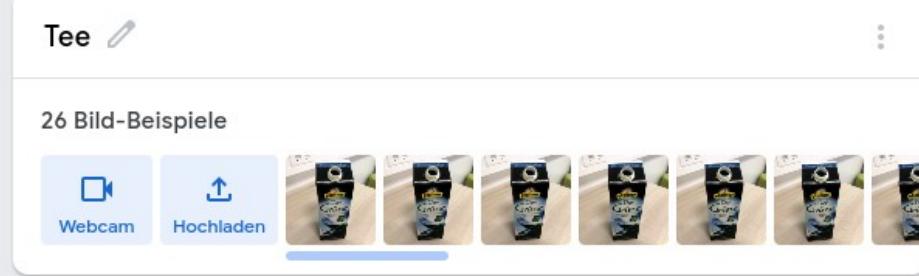
# ML für 10€ Workshop

## Google Teachable Machine

- **Sehr einfache Bedienung** (keine Programmierkenntnisse!)
- Klassifikation von **Posen**, **Bildern** und **Tönen**
- Verwendung von Dateien oder Aufnahmen der Webcam
- Erzeugt **TensorFlow (Lite)** Modelle (auch quantisiert)
- Läuft im Browser → <https://is.gd/ejicaj>
- Android App → <https://is.gd/surife>
- Beispiel PA2 Muster → <https://is.gd/dujoye>

# ML für 10€ Workshop

≡ Teachable Machine



Training

Modell ist trainiert

Erweitert

Vorschau

Modell exportieren

Eingabe AN

Webcam



Ausgabe

Becher

100%

Tee

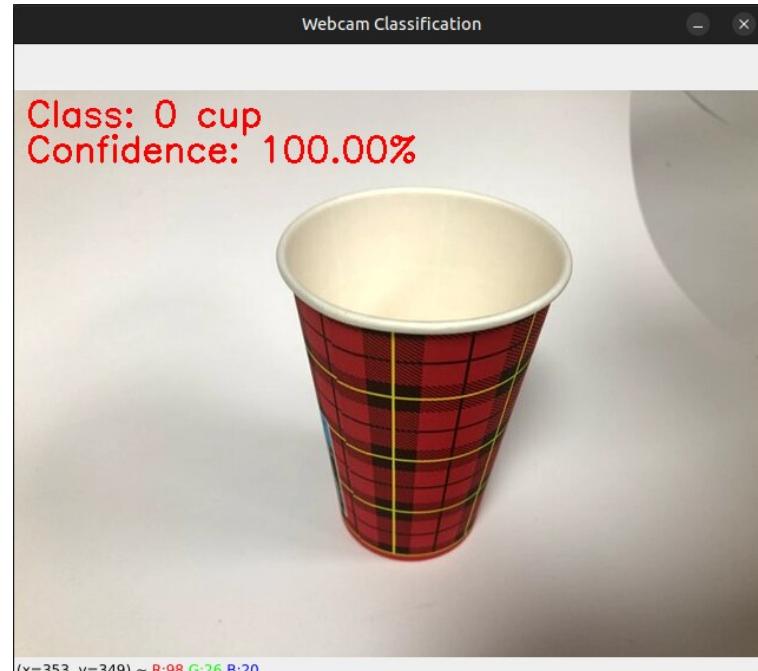
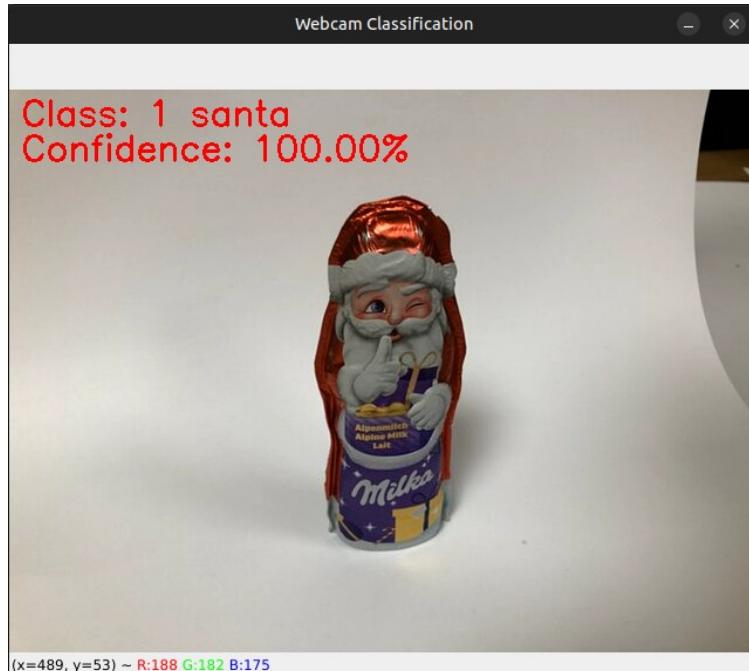
100%

Erzeugt TensorFlow (.js, Lite, Standard) Modelle (!)  
→ <https://teachablemachine.withgoogle.com>

# ML für 10€ Workshop

## Google Teachable Machine

Python Code für Inferenz mit TensorFlow (mit beliebigem PC)  
→ <https://is.gd/ecezep>



# ML für 10€ Workshop

## Edge Impulse

- Cloud Lösung für **Training** und Bereitstellung von ML Modellen für IoT, µC, Embedded Systeme; Ziel: keine Internetanbindung
- Anbindung von **Sensoren** - z.B. Beschleunigungsmesser, Gyroskope oder Mikrofone
- Verschiedene maschinelle Lernalgorithmen möglich, z.B. **MobileNet V1** – also **Training** nur durch Klicken
- Kostenlos (teilweise); holländische Firma



# ML für 10€ Workshop

MongoQ / Labortage 2023

An impulse takes raw data, uses signal processing to extract features, and then uses a learning block to classify new data.

The screenshot shows the Edge Impulse web interface. On the left is a sidebar with various options: Dashboard, Devices, Data acquisition, Impulse design (with sub-options Create impulse, Image, Object detection), EON Tuner, Retrain model, and Live classification. Below this is a 'Try Enterprise Free' section with a 'Start free trial' button. The main area is titled 'MongoQ / Labortage 2023'. It displays four blocks: 'Image data' (red background) with input axes 'image' (48x48), 'Input features' (checkbox checked), and 'Output features' (4: circle, cross, square, triangle); 'Image' (white background) with input axes (1) 'image' selected; 'Object Detection (Images)' (purple background) with input features 'Image' selected and output features (4: circle, cross, square, triangle); and 'Output features' (green background) with a checkmark icon. A 'Save Impulse' button is located at the bottom right of the purple block.

EDGE IMPULSE

Dashboard

Devices

Data acquisition

Impulse design

- Create impulse
- Image
- Object detection

EON Tuner

Retrain model

Live classification

Try Enterprise Free

Start free trial

M

Image data

Input axes

image

Image wi... Image hei...

48 48

Resize mode

Fit sl

For object detection use a square image size, e.g. 96x96, 160x160 or 320x320.

Image

Name

Input axes (1)

✓ image

Object Detection (Images)

Name

Input features

✓ Image

Output features

4 (circle, cross, square, triangle)

Output features

4 (circle, cross, square, triangle)

Save Impulse

# ML für 10€ Workshop

## Bastelzeit

Dies ist ja auch ein **Workshop**,  
also **basteln** wir ;-)

# ML für 10€ Workshop

**Task:** Klassifikation

**Weihnachtsmann „Santa“,  
Becher „Cup“ oder  
Hintergrund „Empty“**

# ML für 10€ Workshop

## Bausatz HW - ESP32-CAM (~6€)

### Technische Daten

Dual-Core 32-Bit-CPU @240 MHz - ESP32-D0WDQ6-V3

Eingebauter 520 KB SRAM, externer 4M PSRAM

WiFi 802.11 b/g/n

Bluetooth 4.2 BLE

Schnittstellen - UART/SPI/I2C/PWM/ADC/DAC

Kamera - OV2640 (2MP)

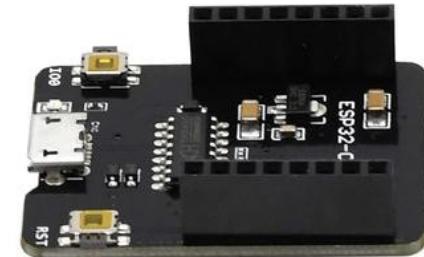
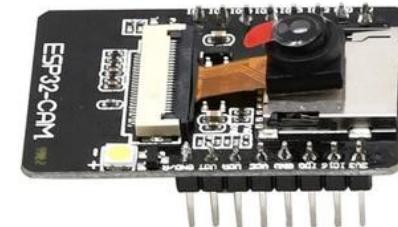
Onboard Flash LED

Onboard rote LED

Slot für MicroSD-Karte

U.FL Anschluss für externe Antennen

CH340 Board (USB-TTL-Wandler) Untersatz



# ML für 10€ Workshop

## Bausatz aufgebaut



(3D-Druck von → <https://is.gd/unixup>)

# ML für 10€ Workshop

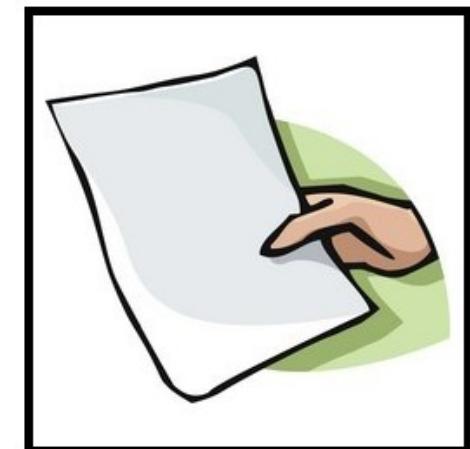
## Bausatz Handout

Anleitung

→ <https://is.gd/iremav>

Video (ähnlich)

→ <https://is.gd/amexos>



# ML für 10€ Workshop

## Edge Impulse

- **Keine Programmierung** nötig!
- Bloß **Aufnahme von insgesamt 140 Bildern** (Objekte gedreht)
- **Freie Accounts** für alle (Austeilen!)
- **Einloggen** über → <https://is.gd/ezawap>
- Anlegen eines neuen Accounts (optional!) → <https://is.gd/adujuw>



# ML für 10€ Workshop

**Eigenes WLAN**  
für diesen Workshop

**Linksys WRT54G AP**

**SSID:** esp32-wrt54g  
**PW:** esp32-cam



# ML für 10€ Workshop

## Eigenes WLAN für diesen Workshop

Über die **IP 192.168.1.1** des **WRT54G**  
die **IP Adresse** des Boards passend zur **MAC Adresse** ermitteln

DHCP			
DHCP Clients			
Host Name	IP Address	MAC Address	Client Lease Time
esp32-ACB3C4	192.168.1.114	24:DC:C3:AC:B3:C4	1 day 00:00:00
x220	192.168.1.131	[REDACTED]	1 day 00:00:00

Auto-Refresh is On

DD-WRT

# ML für 10€ Workshop

## Aufbau „Theater“

- Zusammenbau eines Aldi Kartons mit etwas A3-Papier und Blu Tack
- FOTO

# ML für 10€ Workshop

# Inferenzergebnisse

- Momentan Inferenzergebnis über **Arduino Serial Monitor**
  - Besser: Inferenzergebnis **Übertragung** per **WLAN** an Laptop
  - Nur einen **kurzen String** übertragen (keine Bounding Boxen)
  - Erfordert Anpassung des **Arduino C-Codes**
  - Python Skript vorhanden (**Demo läuft**)
  - Arduino Quelltext sollte mit eigenem Code  
→ <https://is.gd/tazebe> ergänzt werden

The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** "usb\_serial\_print\_speed | Arduino 1.8.9"
- File Path:** "/dev/cu.usbmodem62615701"
- Code Area:** The code for the sketch "usb\_serial\_print\_speed". It includes a loop that prints the current value of 'count' every second.
- Serial Monitor Area:** Displays the output of the serial port:

```
count=11000092755, lines/sec=306471
count=11000092756, lines/sec=306471
count=11000092757, lines/sec=306471
count=11000092758, lines/sec=306471
count=11000092759, lines/sec=306471
count=11000092760, lines/sec=306471
count=11000092761, lines/sec=306471
count=11000092762, lines/sec=306471
count=11000092763, lines/sec=306471
count=11000092764, lines/sec=306471
count=11000092765, lines/sec=306471
count=11000092766, lines/sec=306471
count=11000092767, lines/sec=306471
count=11000092768, lines/sec=306471
count=11000092769, lines/sec=306471
count=11000092770, lines/sec=306471
```
- Bottom Buttons:** "Autoscroll" (checked) and "Show timestamp"
- Terminal Area:** Shows the terminal output:

```
TeensyPipeMonitor ctor, port_usb:14200000
disconnect
TeensyPipeMonitor open usb:14200000
opened, dev=/dev/cu.usbmodem62615701, name=Serial
```

# ML für 10€ Workshop

Live!

Demonstration



# ML für 10€ Workshop

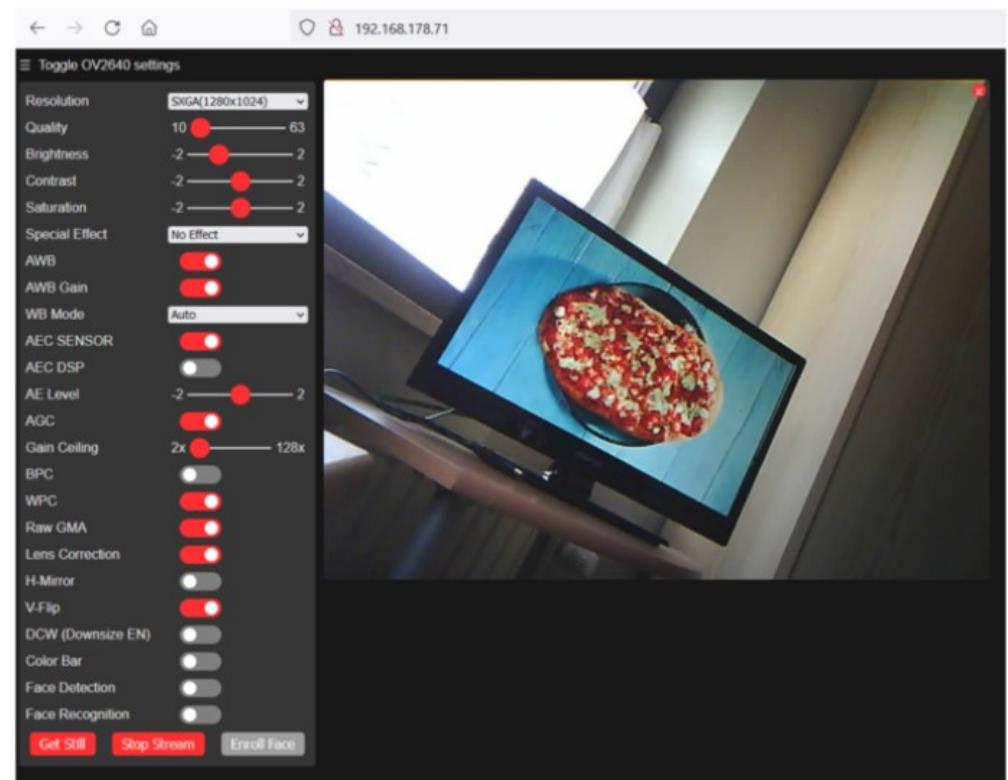
## Andere Anwendungen für ESP32-CAM Boards



# ML für 10€ Workshop

## ESP32-CAM „nur“ als Webcam

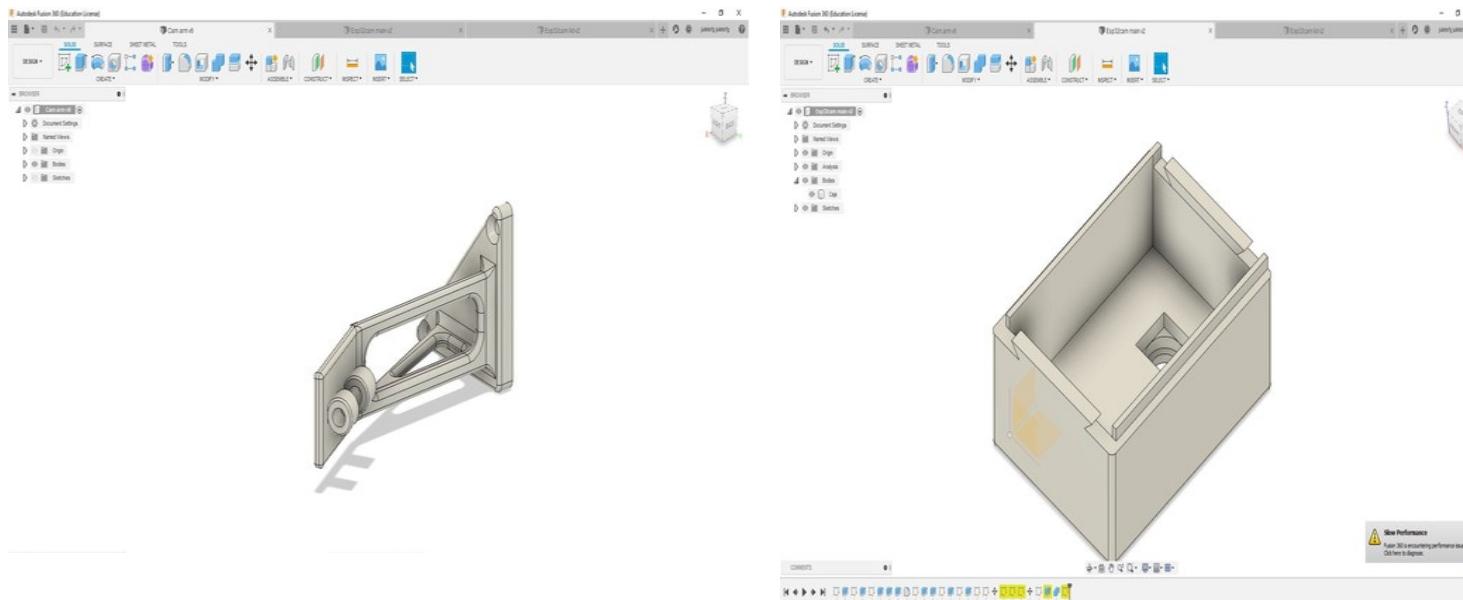
- Ja, ist **machbar** ...  
→ <https://is.gd/asicim>



# ML für 10€ Workshop

## Einsatz als Webcam für 3D-Drucker

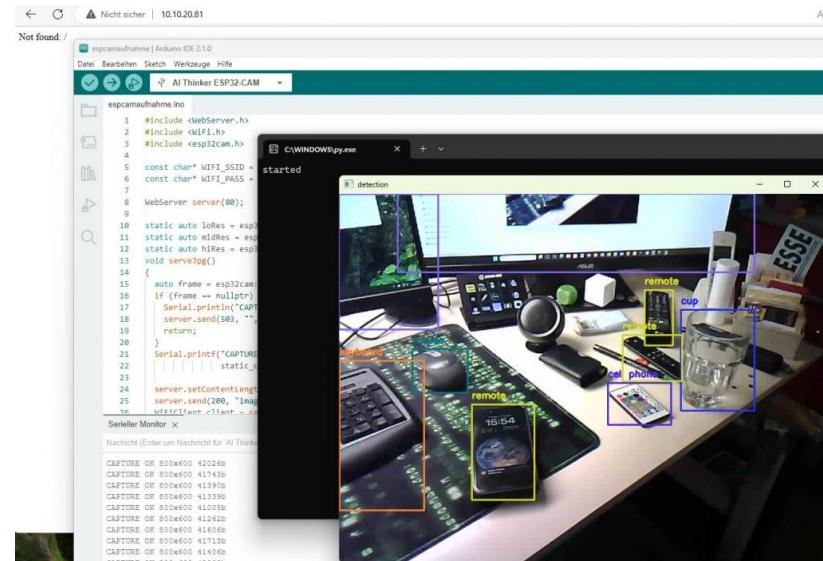
- ESP32-CAM Aufhängung für Ender 3 - <https://is.gd/uxuhit>
- OctoPrint Plugin - <https://is.gd/efubaq>



# ML für 10€ Workshop

## ESP32-CAM „nur“ als Webcam ... ... aber mit YOLO ;-)

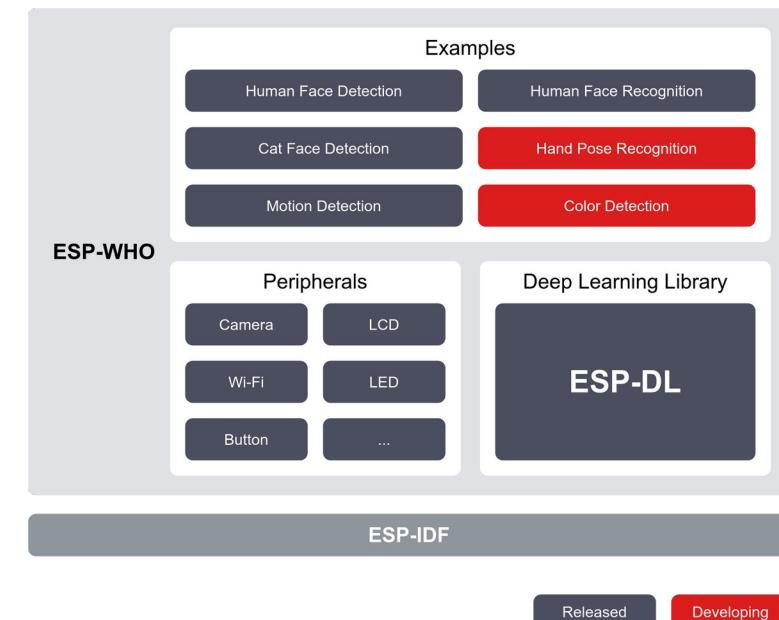
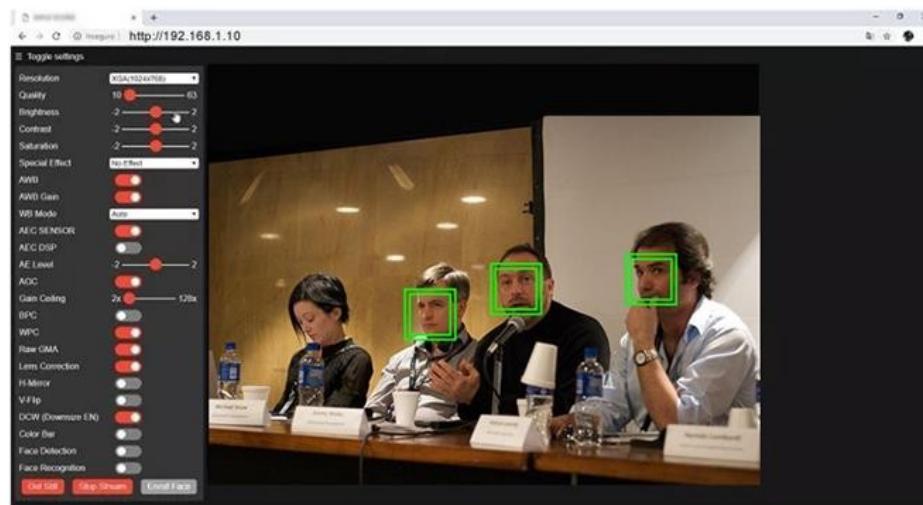
- Ja, machbar, aber „da könnte man ja auch eine reguläre Webcam einsetzen ...“
- ESP32-CAM und Objekt Detektion mit **YOLO** Algorithmus  
→ <https://is.gd/gutuga>



# ML für 10€ Workshop

## Gesichtserkennung mit ESP-IDF & ESP-WHO

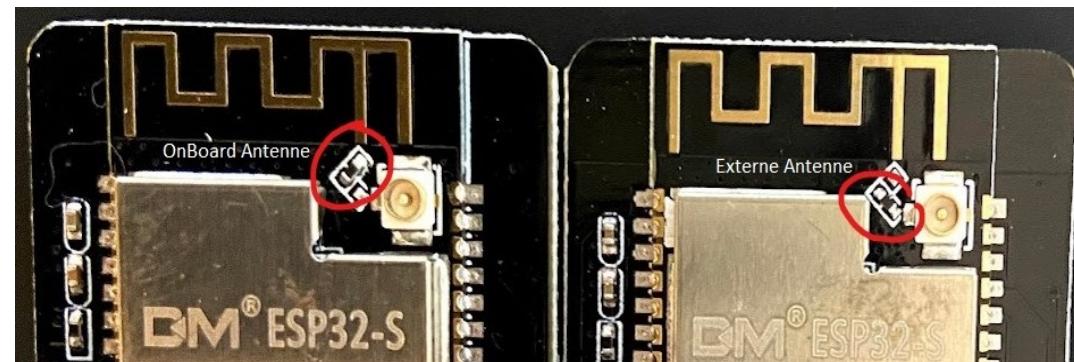
- Einsatz von ESP-IDF
- Gute (!) Anleitung → <https://is.gd/sajehi>
- ESP-WHO → <https://is.gd/gopoya>



# ML für 10€ Workshop

## Antenne

- Eine **externe Antenne** kann angeschlossen werden - durch **Umlöten eines Widerstands**
- Empfehlung, das ESP32-CAM Board, unbedingt **ohne die USB-TTL-Wandler** Platine über ein Netzteil mit mindestens 1A zu versorgen - bei stabilen 5V → <https://is.gd/oxiwik> („steht im Internet“)



# ML für 10€ Workshop

## Persönliche Empfehlung

Ideen für eigene weitergehende Beschäftigung

- Building a **neural network FROM SCRATCH** (no Tensorflow/Pytorch, just numpy & math)  
→ <https://youtu.be/w8yWXqWQYmU>
- Deep Learning with FPGAs - A getting started tutorial on **FPGA** implement of CNN using **OpenCL**  
→ <https://is.gd/epomor>
- Implement Convolutional neural networks (CNN) in **FPGA** “zu Fuß“ programmieren → <https://is.gd/fuvoro>
- **PyFPGA** - Tool für Steuerung und Programmierung von FPGAs, Frontend für Herstellertools (z.B. Xilinx ISE, Intel Quartus)  
→ <https://is.gd/omoset>
- **MyHDL** - HDL, u.a zur Simulation und Synthese bei FPGAs  
→ <https://is.gd/ezecac>

# ML für 10€ Workshop

**Vielen Dank für eure Aufmerksamkeit!**

Evtl. habt ihr noch Fragen ...

