

Entwicklung eines autonomen Systems zur Bilderkennung mithilfe Neuronaler Netze auf dedizierter Hardware

Kolloquium - Bachelorarbeit

Manuel Barkey

Reutlingen, 29.01.2020



Motivation

- ▶ Überwachungssystem zur Wildtier-Erkennung
 - ▶ Auf Raspberry Pi
 - ▶ Automatisches Benachrichtigen



Motivation

- ▶ Überwachungssystem zur Wildtier-Erkennung
 - ▶ Auf Raspberry Pi
 - ▶ Automatisches Benachrichtigen

- ▶ Tag- und nachtgeeignet:
 - ▶ Infrarot-Kamera
 - ▶ Infrarot-LEDs



Motivation

- ▶ Überwachungssystem zur Wildtier-Erkennung
 - ▶ Auf Raspberry Pi
 - ▶ Automatisches Benachrichtigen
- ▶ Tag- und nachtgeeignet:
 - ▶ Infrarot-Kamera
 - ▶ Infrarot-LEDs
- ▶ Erkennung: Deep Learning
 - ▶ Nur bestimmte Tiere
 - ▶ Inferenz: Neural Compute Stick 2



Gliederung

Grundlagen

Training

Evaluierung

Applikation

Zusammenfassung und Ausblick



Gliederung

Grundlagen

Machine Learning
Hardware

Training

Evaluierung

Applikation

Zusammenfassung und Ausblick



Machine Learning

Erkennung von Zusammenhängen in großen Datenmengen,
ohne explizite Programmierung darauf.

► *Supervised Learning*



Machine Learning

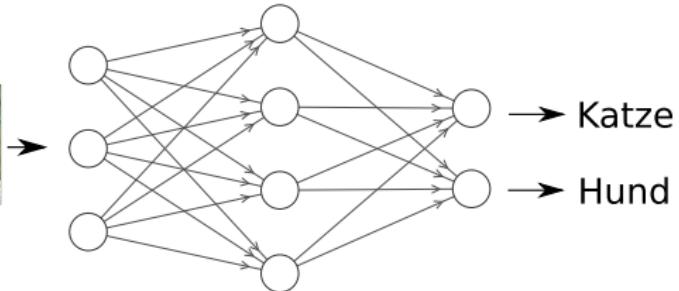
Erkennung von Zusammenhängen in großen Datenmengen,
ohne explizite Programmierung darauf.

► *Supervised Learning*



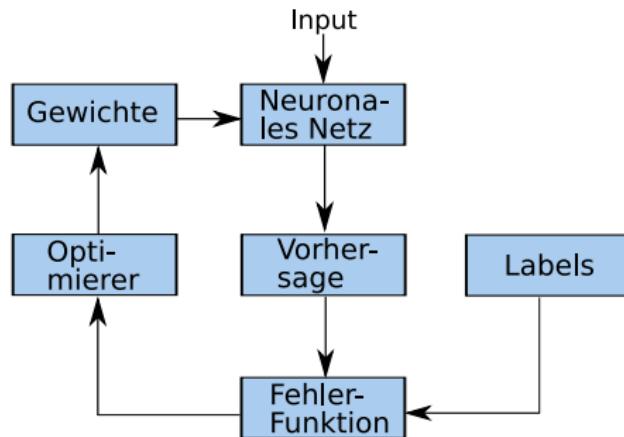
Neuronale Netze

Für komplexe Input
Daten, z.B. Bilder



Training & Inferenz

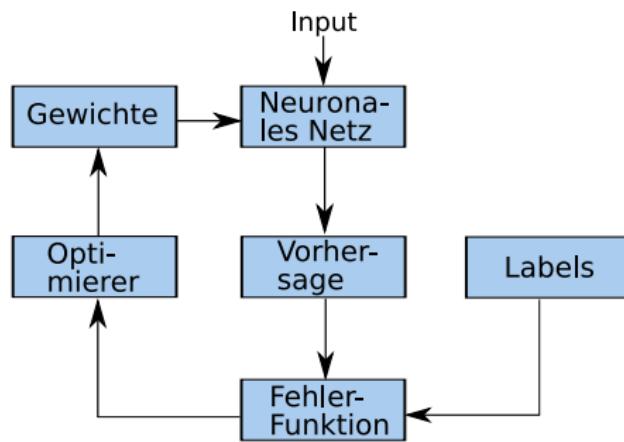
Training



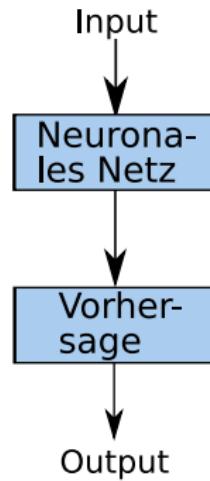
- ▶ Variable Parameter: *Gewichte*
- ▶ Bekannte Input Daten: *Labels*
- ▶ Mehrfaches Durchlaufen: *Epochen*

Training & Inferenz

Training



Inferenz



- ▶ Variable Parameter: *Gewichte*
- ▶ Bekannte Input Daten: *Labels*
- ▶ Mehrfaches Durchlaufen: *Epochen*

- ▶ Fixe Parameter
- ▶ Unbekannte Input Daten
- ▶ Einmaliges Durchlaufen

Intel Neural Compute Stick 2

Beschleuniger für die Inferenz von
Deep Learning Algorithmen

- ▶ Prozessor:
Intel Movidius Myriad X VPU
 - ▶ Effizient bei NN-spezifischen Rechenoperationen
- ▶ Anwendungen:
Edge Computing
 - ▶ Z.B. Überwachungskameras, Drohnen



Gliederung

Grundlagen

Training

Sammeln und Aufbereiten der Daten
Auswahl und Training des Modells

Evaluierung

Applikation

Zusammenfassung und Ausblick



Datensatz

Bilder + Labels mit Koordinaten der Bounding Boxen

- ▶ **OpenImages** Open Source Dataset von Google
- ▶ 9 Klassen mit Wildtieren (je 200 bis 2000 Bilder)
 - ▶ Braun Bär, Hirsch, Fuchs, Ziege, Igel, Eule, Hase, Waschbär, Eichhörnchen



Datensatz

Bilder + Labels mit Koordinaten der Bounding Boxen

- ▶ **OpenImages** Open Source Dataset von Google
- ▶ 9 Klassen mit Wildtieren (je 200 bis 2000 Bilder)
 - ▶ Braun Bär, Hirsch, Fuchs, Ziege, Igel, Eule, Hase, Waschbär, Eichhörnchen

Validierung und Overfitting

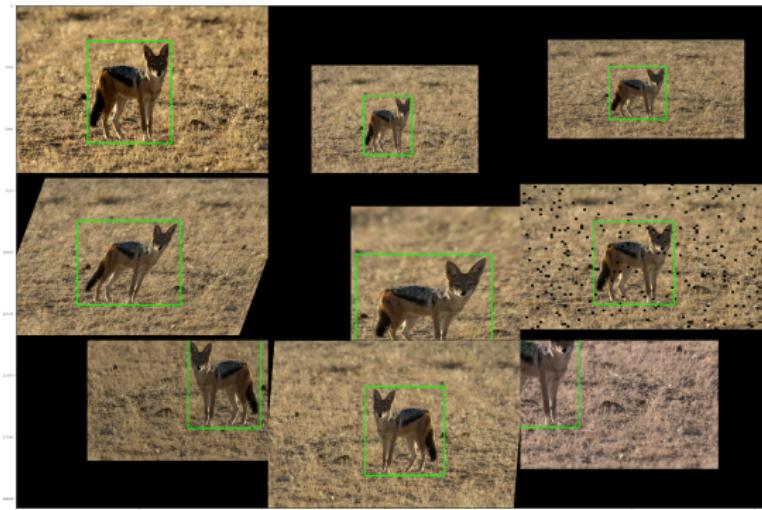
Kontrolle des Trainings durch Aufteilen der Daten in:



- ▶ **Overfitting:** Nur die Trainingsdaten werden gelernt

Augmentierung

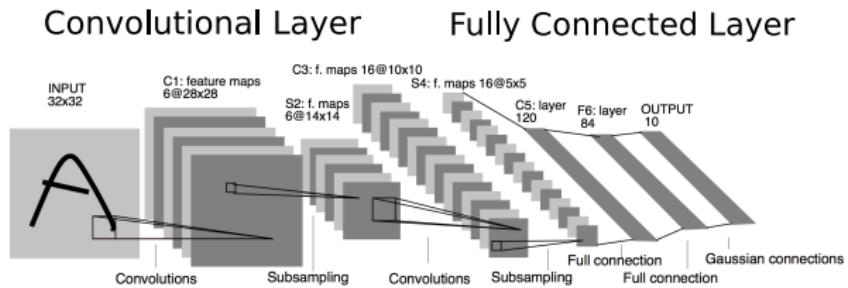
Künstlich mehr Daten erzeugen, verhindern von Overfitting



- ▶ Geometrisch: Verschieben, Spiegeln, Rotieren, Zoom
- ▶ oder: Farbwerte, Helligkeit, Kontrast, Rauschen, Dropout

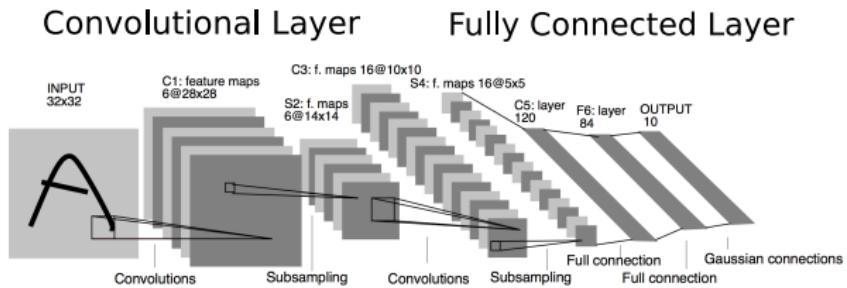
Convolutional Neural Network

- ▶ Faltung des Inputs mit Filter Matrix
- ▶ Erzeugen Feature-Maps
- ▶ Räumliche Invarianz



Convolutional Neural Network

- ▶ Faltung des Inputs mit Filter Matrix
- ▶ Erzeugen Feature-Maps
- ▶ Räumliche Invarianz



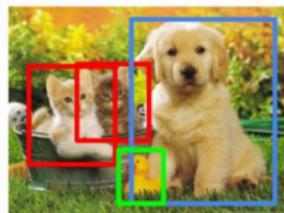
Objekt Detection

- ▶ Basis CNN
 - ▶ Feature Extraction
- ▶ Object Detection Framework
 - ▶ Single Shot Det. (SSD)
 - ▶ Region Based (R-CNN)

Classification



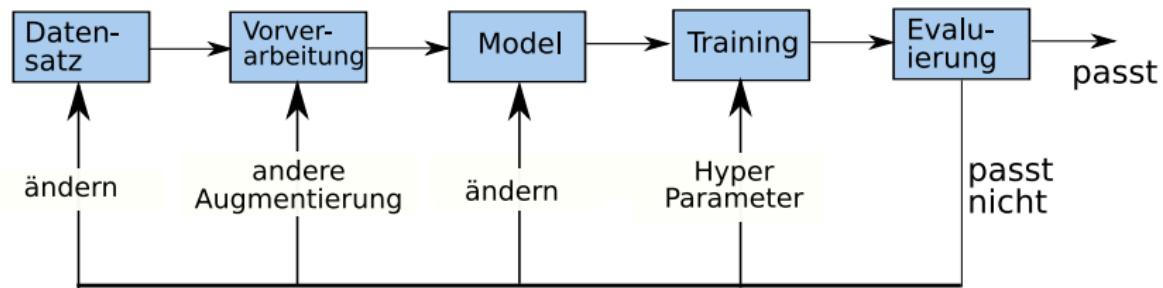
Object Detection



Trainingsworkflow

Framework: *Tensorflow*

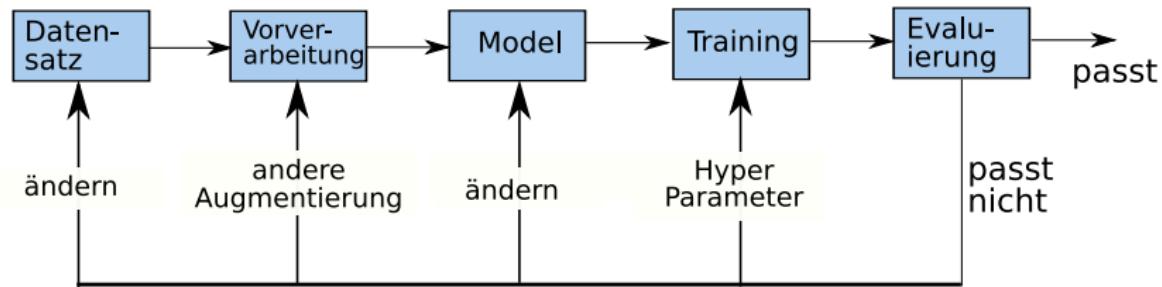
Mehrfaches Durchlaufen und anpassen des Trainingsprozess



Trainingsworkflow

Framework: *Tensorflow*

Mehrfaches Durchlaufen und anpassen des Trainingsprozess



- ▶ **Datensatz**
 - ▶ Augmentierung
 - ▶ Graustufen
- ▶ **Modelle:**
 - ▶ Faster R-CNNs
 - ▶ SSDs
- ▶ **Evaluierung:**
 - ▶ Genauigkeit (mAP)
 - ▶ Fehlerrate (Loss)

Gliederung

Grundlagen

Training

Evaluierung

Applikation

Zusammenfassung und Ausblick



Mean Average Precision (mAP)

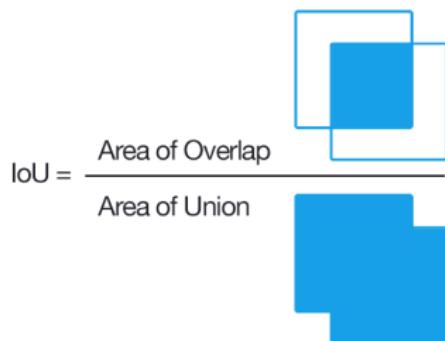
Intersection over Union (IoU)

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$


- ▶ $\text{IoU} > 0.5 \rightarrow \text{True Positive}$

Mean Average Precision (mAP)

Intersection over Union (IoU)



Recall: Trefferquote

$\frac{\text{True Positives}}{\text{alle Objekte im Bild}}$

Precision Genauigkeit

$\frac{\text{True Positives}}{\text{alle Predictions}}$

Average Precision für eine Klasse

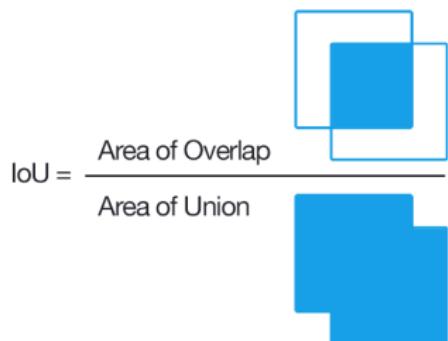
- ▶ $\text{IoU} > 0.5 \rightarrow \text{True Positive}$

$$AP = \frac{1}{N} \sum Precision(Recall)$$

Mean Average Precision (mAP)

Intersection over Union (IoU)

Recall: Trefferquote



$\frac{\text{True Positives}}{\text{alle Objekte im Bild}}$

Precision Genauigkeit

$\frac{\text{True Positives}}{\text{alle Predictions}}$

Average Precision für eine Klasse

- ▶ $\text{IoU} > 0.5 \rightarrow \text{True Positive}$

$$AP = \frac{1}{N} \sum \text{Precision}(\text{Recall})$$

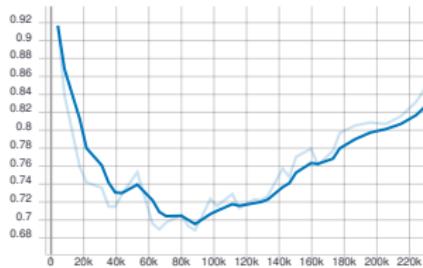
Loss

- ▶ **Lokalisierung:** Bounding Box Regression
- ▶ **Klassifizierung:** (Logarithmische) Fehlerberechnung

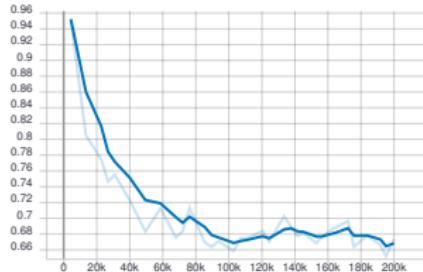
Visualisierung des Trainingsfortschritts

► Tensorboard

Ohne Augmentierung

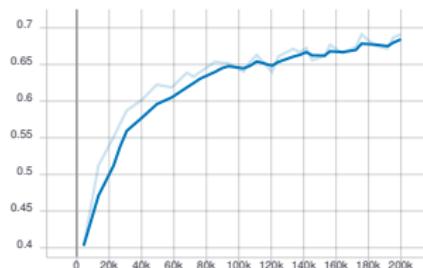
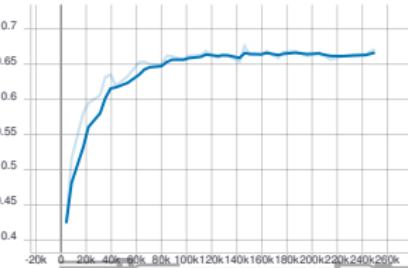


Mit Augmentierung



Loss

mAP



Vergleich Modelle: Genauigkeit - Inferenzzeit

Inferenzzeit auf dem NCS2

Architecture	Base CNN	Infer FPS	
		Sync.	Async.
SSD	Mobilenet	12,6	33,6
	InceptionV2	10,7	28,3
Faster R-CNN	InceptionV2	0,55	0,72

Vergleich Modelle: Genauigkeit - Inferenzzeit

Inferenzzeit auf dem NCS2

Architecture	Base CNN	Infer FPS	
		Sync.	Async.
SSD	Mobilenet	12,6	33,6
	InceptionV2	10,7	28,3
Faster R-CNN	InceptionV2	0,55	0,72

Genauigkeit

Model	mAP	Loss
SSD InceptionV2	0.55	4,4
+Augmentierung	0.6	4,1
Faster R-CNN	0.67	0.82
+Augmentierung	0.7	0,7
+Dropout/L2 Reg.	0.7	0.66



Vergleich Modelle: Genauigkeit - Inferenzzeit

Inferenzzeit auf dem NCS2

Architecture	Base CNN	Infer FPS	
		Sync.	Async.
SSD	Mobilenet	12,6	33,6
	InceptionV2	10,7	28,3
Faster R-CNN	InceptionV2	0,55	0,72

Genauigkeit

Model	mAP	Loss
SSD InceptionV2	0.55	4,4
+Augmentierung	0.6	4,1
Faster R-CNN	0.67	0.82
+Augmentierung	0.7	0,7
+Dropout/L2 Reg.	0.7	0.66

► Je **genauer**, desto **langsamer**!



Gliederung

Grundlagen

Training

Evaluierung

Applikation

Zusammenfassung und Ausblick



Test Inferenz auf eigene Bilder

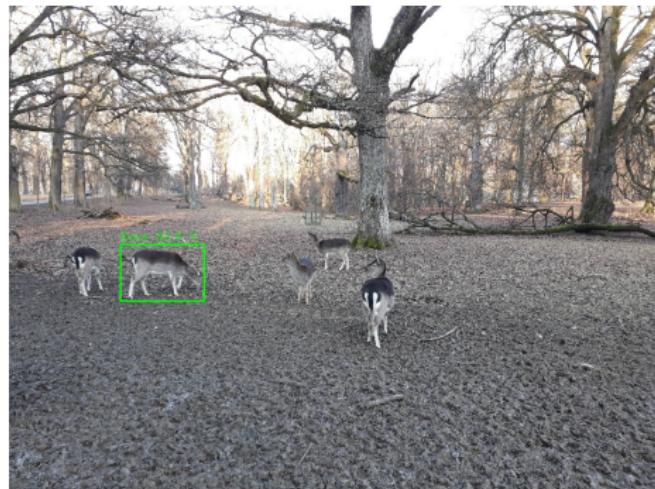


Abbildung: SSD+InceptionV2

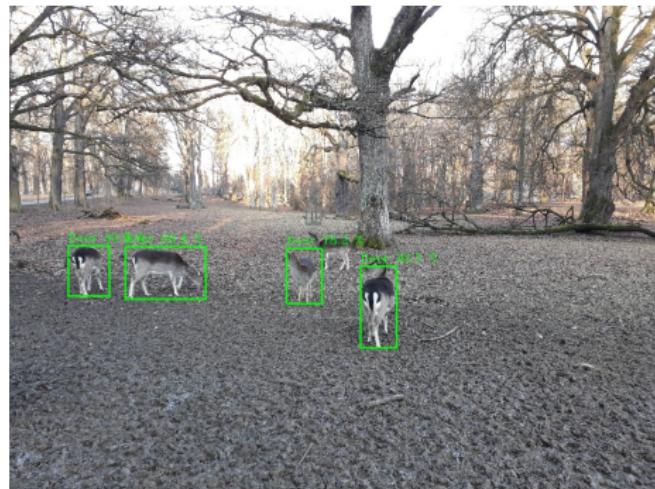
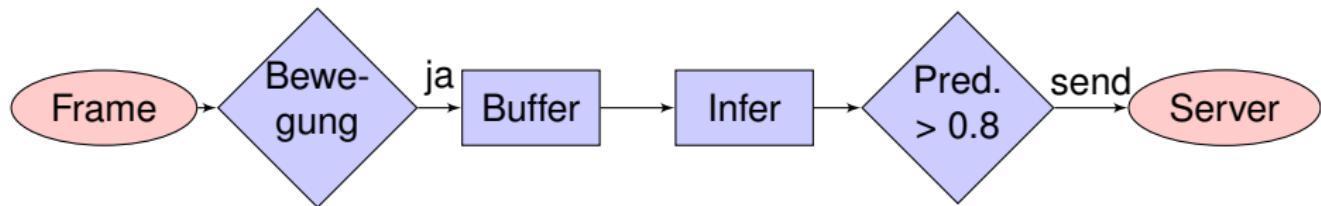


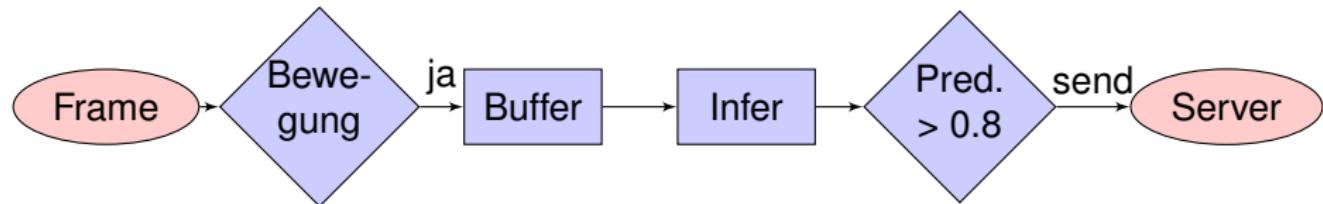
Abbildung: Faster R-CNN+InceptionV2



Applikation

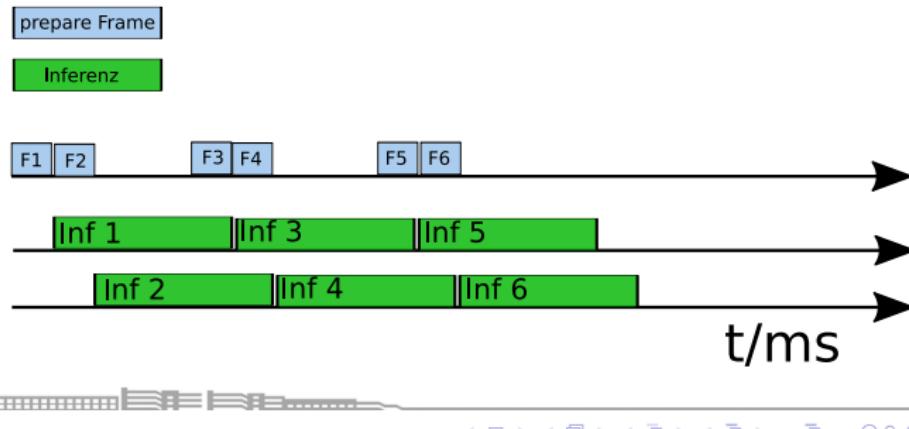


Applikation



Asynchrone Inferenz

- ▶ Inferenz läuft asynchron



- ▶ Parallele Inferenz-Requests

Gliederung

Grundlagen

Training

Evaluierung

Applikation

Zusammenfassung und Ausblick



Zusammenfassung

- ▶ Datensatz von *OpenImages* geladen und augmentiert
- ▶ Verschiedene *Tensorflow Modelle* trainiert und ausgewertet
- ▶ Anwendung für den *Raspberry Pi* implementiert
- ▶ Trainiertes Modell integriert

Ausblick

- ▶ Mobiles Internet für den Raspberry Pi (GSM-/LTE Modul)
- ▶ Nutzerbenachrichtigung z.B. per E-Mail
- ▶ Anwendung draußen testen

