HIGHTECH Agenda Bayern

KI PRODUKTIONS-NETZWERK AUGSBURG

Universität Augsburg University · DLR · Fraunhofer IGCV · Technische Hochschule Augsburg

KI - Bildungsprogramm

2. Praxistag KI-Zusatzzertifikat für Auszubildende

"KI in der Produktion"

Computer Vision

Eine Definition

Computer Vision ist ein Feld innerhalb der künstlichen Intelligenz (KI), das es Computern und System ermöglicht aussagefähige Informationen aus digitalen Bildern, Videos und anderen Visuellen Eingaben zu gewinnen – und auf Grundlage dieser Informationen Maßnahmen zu ergreifen oder Empfehlungen abzugeben.

IBM (2024)

Aber

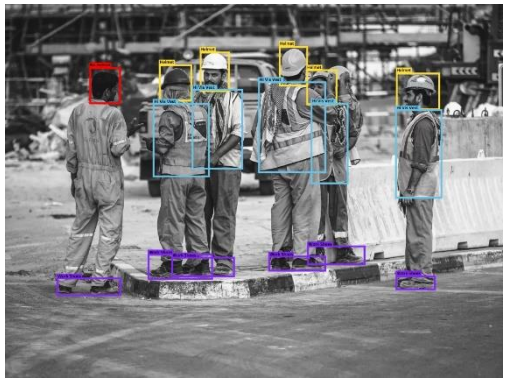
Obwohl die Wahrnehmung für den Menschen eine mühelose Tätigkeit zu sein scheint, erfordert sie anspruchsvolle Berechnungen.

Russel & Norvig (2010)

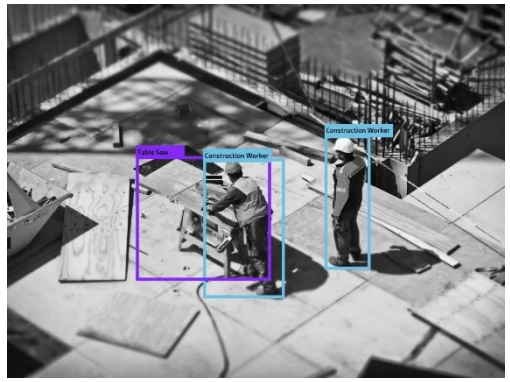
Computer Vision

Use Cases - Industrie

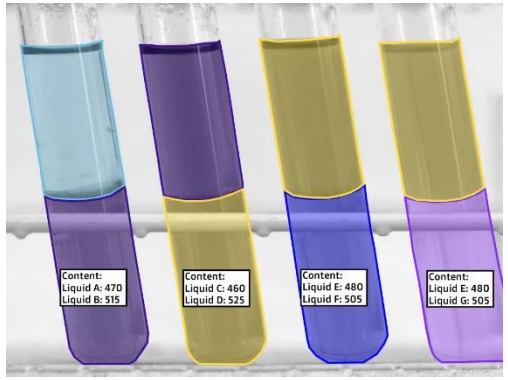
Logistik



Sicherheit

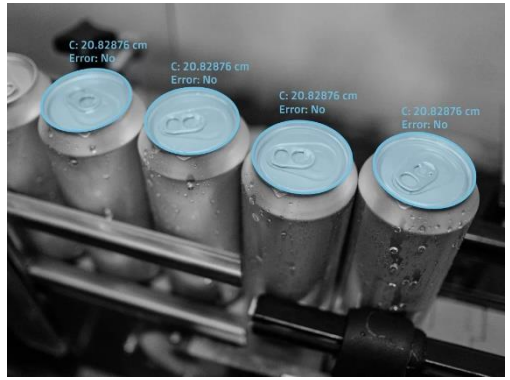


Objekterkennung



Füllstandsüberwachung

Sortierverfahren



Fehlererkennung

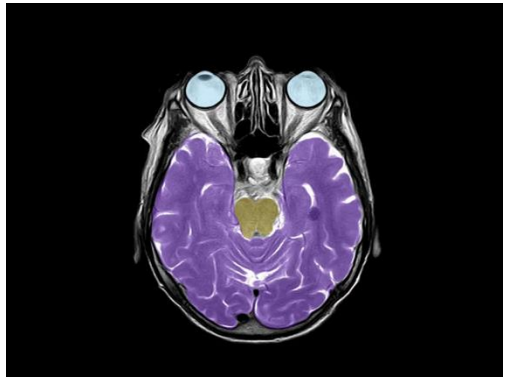
Quelle: *Humans in the Loop (https://humansintheloop.org)*

HIGHTECH Agenda Bayern

Universität Augsburg University

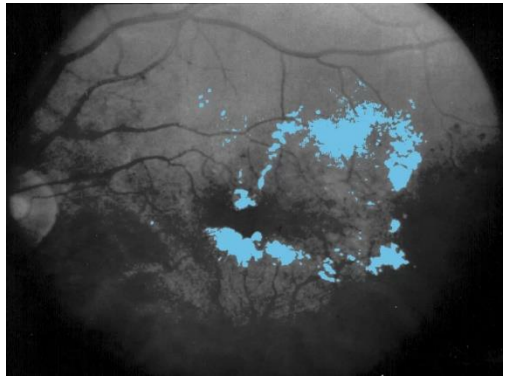
Computer Vision

Use Cases – Medizin & Einzelhandel



Radiologie

Lagerstands-
überwachung



Glaukom- und
Katarakterkennung

Kundenverhalten

Quelle: *Humans in the Loop (https://humansintheloop.org)*

HIGHTECH Agenda Bayern
Universität Augsburg University

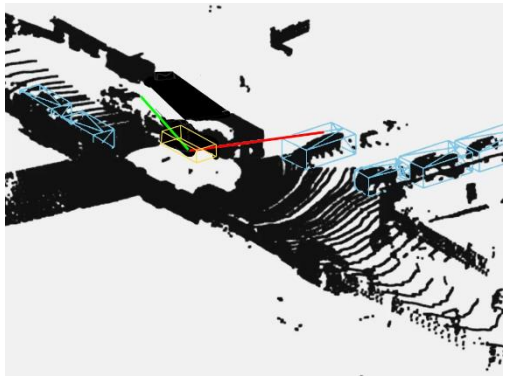
Computer Vision

Use Cases - Automotive

Kabinenüberwachung

Objekterkennung

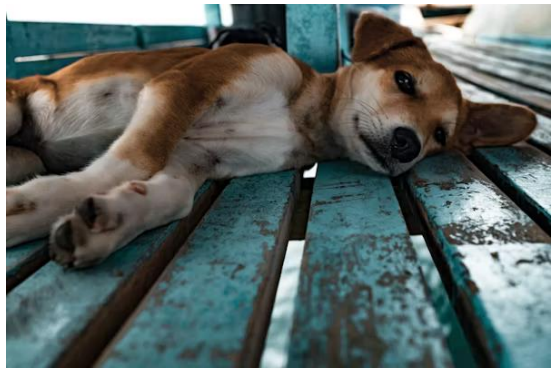
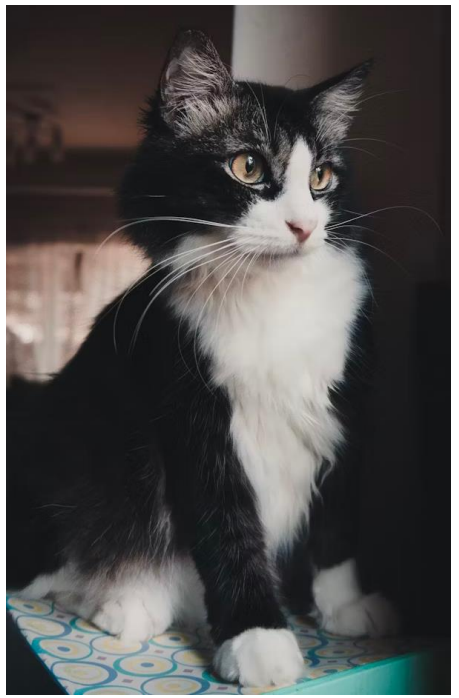
Verkehrsüberwachung

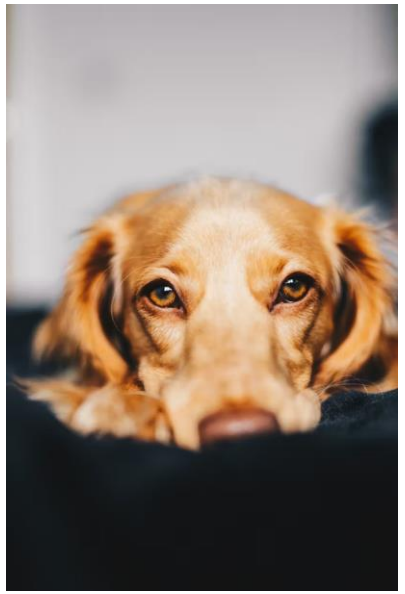
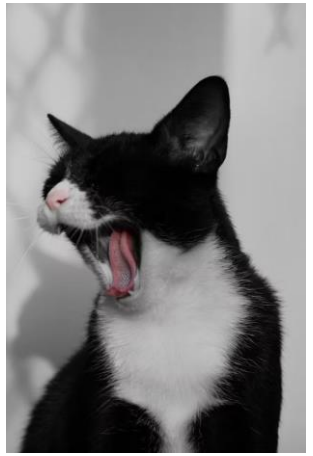


LIDAR & Sensorfusion

Quelle: *Humans in the Loop (https://humansintheloop.org)*

Trainingsdaten

HIGHTECH Agenda Bayern
Universität Augsburg University

Testdaten

HIGHTECH Agenda Bayern
Universität Augsburg University

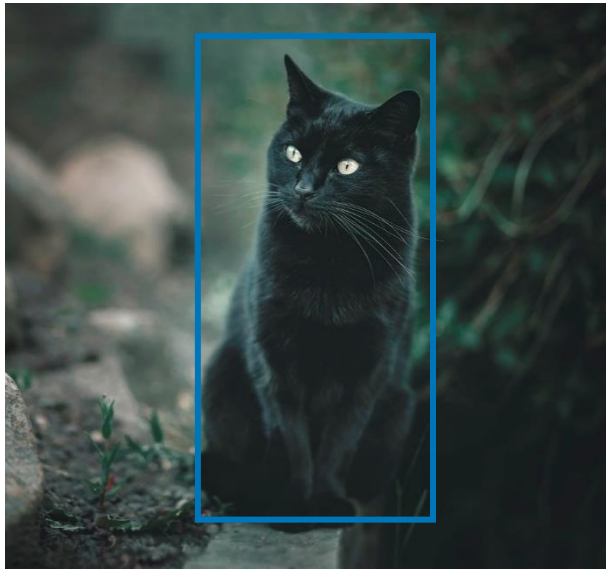
Komplexität der Aufgabe

Bildklassifizierung vs. Objekterkennung

Ganzes Bild hat das Label „Cat“

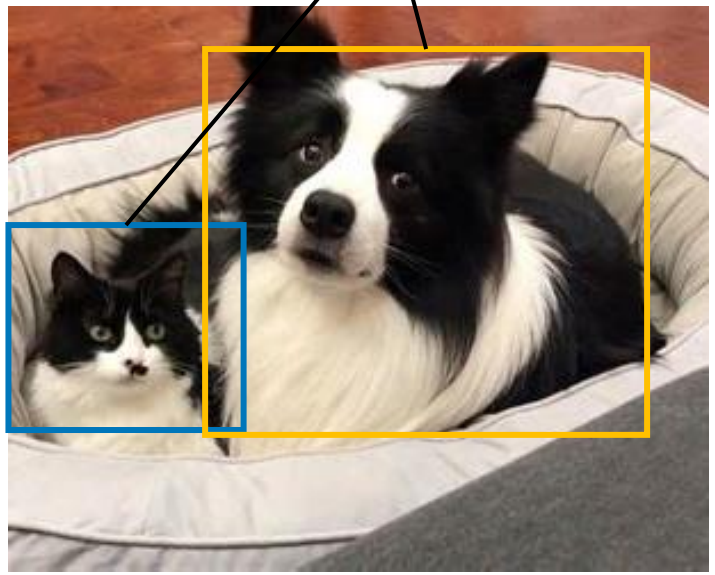
Classification
Cat

Ganzes Bild hat das Label „Cat“



Classification & Localization
Cat

Jede Klasse hat ein eigenes Label



Object Detection
Cat, **Dog**

HIGHTECH Agenda Bayern
Universität Augsburg University

Erstellen eines Datensatzes

Worauf sollte man bei der Erstellung eines Datensatzes achten? Was sind Kriterien "guter Daten"?

Arbeiten mit Jupyter Notebooks

Objekterkennung mit Bounding-Boxen - *das Modell erkennt verschiedene Objekte, wie Autos, Menschen, Hydranten, etc. und identifiziert sie im Bild, indem eine Box um das jeweilige Objekt gezogen wird. Die Zahl gibt jeweils an, wie sicher sich das Modell ist, dass es sich beim identifizierten Objekt tatsächlich um ein solches handelt.*

Um das Verständnis zu vertiefen und um auf wichtige Aspekte aufmerksam zu machen haben wir eine Reihe von Fragen und Aufgaben in dieses Notebook eingebaut:

Beispiel

1. **Verständnisfragen**

Frage: In den blauen Boxen stehen Verständnisfragen

2. **Arbeitsaufträge**

Auftrag: In grünen Boxen finden sich Arbeitsaufträge.

Besonders komplizierte Aspekte oder Bereiche, in denen leicht Fehler gemacht werden können sind mit einer roten Box gekennzeichnet:

Achtung: Hier müssen Sie gut aufpassen. Nehmen Sie sich ausreichend Zeit, die nachfolgenden Angaben zu lesen oder Fragen Sie ein Mitglied des Teams um Hilfe.

Was *ist* Objekterkennung?

Da es sich hierbei um ein Problem des **überwachten Lernens** handelt, reicht es nicht einfach, dem Algorithmus nur die Bilddaten zuzuführen. Damit eine Zuordnung von Bild → Objekt stattfinden kann, müssen die Daten entsprechend **annotiert** sein.

▸ **Exkurs: Algorithmendesign vs. Datenkuration**

Das Annotieren von Daten kann teilweise automatisiert werden, wird aber in den meisten Fällen noch von Menschen per Hand durchgeführt.
In diesem Notebook widmen wir uns also drei Dingen:

1. **Dem Sammeln von Bilddaten**
2. **Der Datenannotation**
3. **Finetuning eines SOTA-Objekterkenners**

1.1. Wo bekommt man die Daten her?

Je nach Problemstellung kann es leichter oder schwieriger sein, an (gute) Daten heranzukommen. Für einige Domänen existieren bereits große - teilweise auch gut aufbereitete - Datensätze, die oft auch frei zugänglich im Internet abrufbar sind.

Beispiele sind:

- ImageNet
- NIST-Datensätze
- MS COCO
- u.v.m.

Die Klassen des **MS COCO** - Datensatzes. **COCO** steht dabei für Common Objects in COntext.

Fragen, Aufträge und wichtige Hinweise

Informationen und Hintergrundwissen

Codezeilen

1.4. Aufnehmen der Bilder

Bevor wir anfangen, legen wir zwei übergeordnete Dinge fest:

- Den Speicherort der Bilder.
- Die Anzahl an Bildern, die wir sammeln wollen.

Auftrag: Führen Sie die nächste Codezelle aus und legen Sie einen Gruppennamen fest.

```
[ ]: GROUP, PATH = set_metadata(TASK)
```

Anschließend werden die Bilder mithilfe der Webcam aufgenommen, mit einer ID versehen und dann abgespeichert.

Setzen des `Device`-Parameters:

▸ Ein Hinweis zu CV2

HIGHTECH Agenda Bayern
Universität Augsburg University

Arbeiten mit Jupyter Notebooks

Codezeilen

Codezeilen müssen der Reihe nach ausgeführt werden:

```
import cv2                                                    # C
import os                                                    # E
from ultralytics import YOLO                                 # M
from IPython.display import Video                            # Z

# Die folgenden Funktionen dienen der Unterstützung beim Erstellen der
################################################################################
from helpers.image_helpers import capture_images             # Helferf
from helpers.model_helpers import inference_webcam           # Helferf
from helpers.model_helpers import inference_video            # Helferf
from helpers.model_helpers import choose_model               # Helferf
from helpers.model_helpers import set_training_config        # Helferf
from helpers.model_helpers import get_trained_model          # Helferf
from helpers.file_handlers import prepare_folder_structure   # Helferf
from helpers.file_handlers import create_config              # Helferf
from helpers.file_handlers import move_file                  # Helferf
from helpers.config import set_metadata                      # Helferf
from helpers.config import set_image_data                    # Helferf
from helpers.file_handlers import cleanup_images

TASK = 'DETECT'
```

[1] [1] ✓ 9.7s

Manche Codezeilen erfordern Eingaben, diese erscheinen am oberen Rand des Bildschirms:

Gruppenname: (Press 'Enter' to confirm or 'Escape' to cancel)

Bei Fehlermeldungen stehen wir gerne zur Verfügung!

⊗ 1.3s

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In[1], line 1
----> 1 _, NUM_IMGS, DELAY, DEVICE = set_image_data(device=5, task=TASK)
      3 capture_images(num_imgs=NUM_IMGS,
      4                name=GROUP,
      5                img_path=PATH,
      6                device=DEVICE,
      7                delay=DELAY,
      8                show=True)

NameError: name 'set_image_data' is not defined
```


HIGHTECH Agenda Bayern
Universität Augsburg University

Analyse und Bewertung der Ergebnisse der Objekterkennung

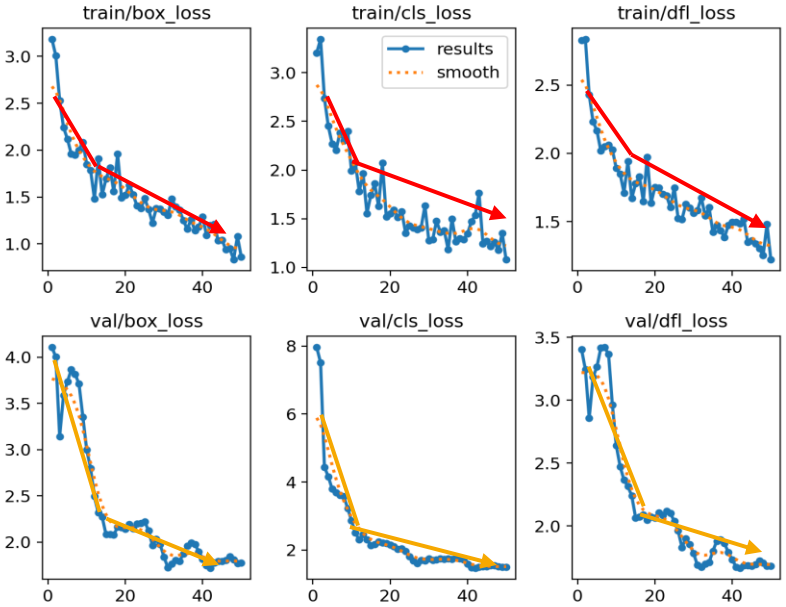
Was ist eine Epoche?

Eine Epoche besteht aus einem Durchgang durch alle Trainingsdaten, wobei das Modell seine Gewichte anpasst, um Fehler (Loss) zu minimieren.

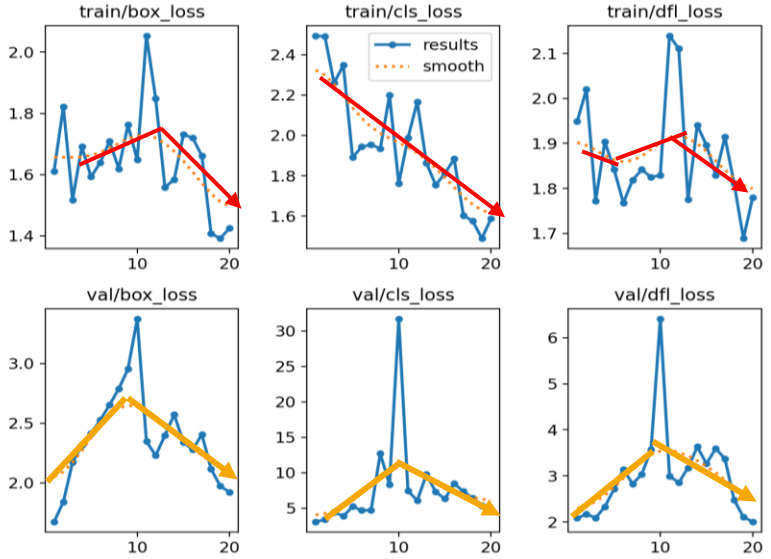
```
Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
 1/25      8.23G      1.462      2.446      1.652        256        800: 100% 100/100 [01:15<00:00,   1.32it/s]
           Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 8/8 [00:05<00:00,   1.60it/s]
             all        230        385      0.396      0.203     0.0524     0.0181
             cat        230        200        0.1        0.3     0.0814     0.0281
             dog        230        184      0.089       0.31     0.0757     0.0263
       dogs-cats        230          1          1          0          0          0

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
 2/25      8.97G      1.166      1.792      1.436        184        800: 100% 100/100 [01:11<00:00,   1.39it/s]
           Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 8/8 [00:03<00:00,   2.08it/s]
             all        230        385      0.477      0.227      0.107     0.0356
             cat        230        200      0.173      0.435      0.162     0.0523
             dog        230        184      0.258      0.245       0.16     0.0545
       dogs-cats        230          1          1          0          0          0

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
 3/25      8.97G      1.139        1.7       1.41        219        800: 100% 100/100 [01:06<00:00,   1.50it/s]
           Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 8/8 [00:03<00:00,   2.06it/s]
             all        230        385      0.118      0.118     0.0664     0.0179
             cat        230        200      0.176       0.18     0.0963     0.0256
             dog        230        184      0.179      0.174      0.103     0.0282
       dogs-cats        230          1          0          0          0          0

Epoch    GPU_mem   box_loss   cls_loss   dfl_loss  Instances       Size
 4/25      8.97G      1.139       1.68      1.411        255        800: 100% 100/100 [01:07<00:00,   1.48it/s]
           Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 8/8 [00:04<00:00,   1.87it/s]
             all        230        385     0.0624      0.184       0.06     0.0165
             cat        230        200     0.0793      0.335     0.0883     0.0227
             dog        230        184      0.108      0.217     0.0916      0.027
       dogs-cats        230          1          0          0          0          0
```


HIGHTECH Agenda Bayern
Universität Augsburg University

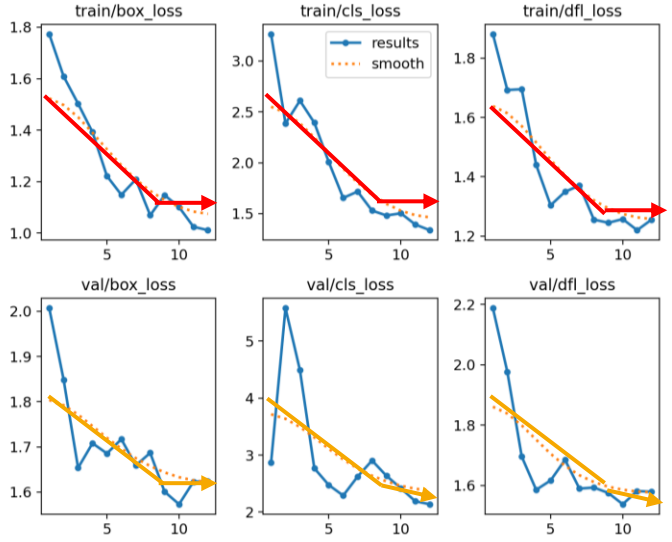
Beispiele für Loss-Grafiken



Ideal wäre es, wenn der Verlust gegen null geht, was bedeutet, dass das Modell immer genauer wird.



Wenn die Verlustkurven stark schwanken und sich nicht abflachen, hat das Modell die Datenmuster noch nicht vollständig erlernt und benötigt weiteres Training.



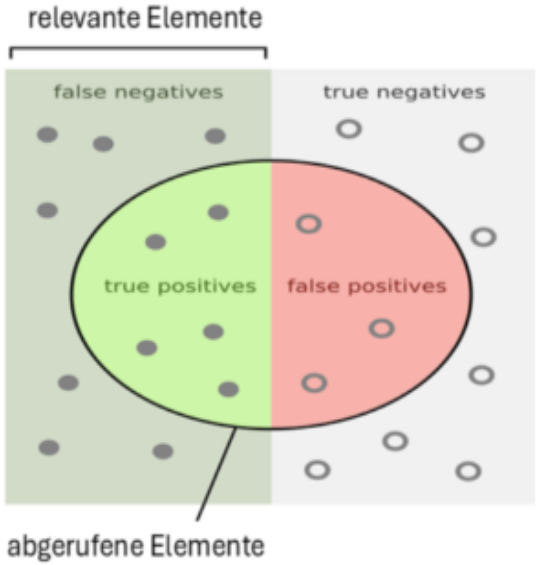
Flachen die Verlustkurven ab, hat das Modell die Datenmuster erlernt, und weiteres Training bringt kaum Verbesserung.

TIPP: Steigt der Validierungsverlust, während der Trainingsverlust sinkt, deutet dies auf Überanpassung (Overfitting) hin.

HIGHTECH Agenda Bayern
Universität Augsburg University

Precision und Recall

Die **Precision** (auch als **positiver prädiktiver Wert** bezeichnet) ist der Anteil der relevanten Instanzen unter den abgerufenen Instanzen.



Recall (auch als **Empfindlichkeit** bekannt) ist der Anteil der relevanten Instanzen, die abgerufen wurden.

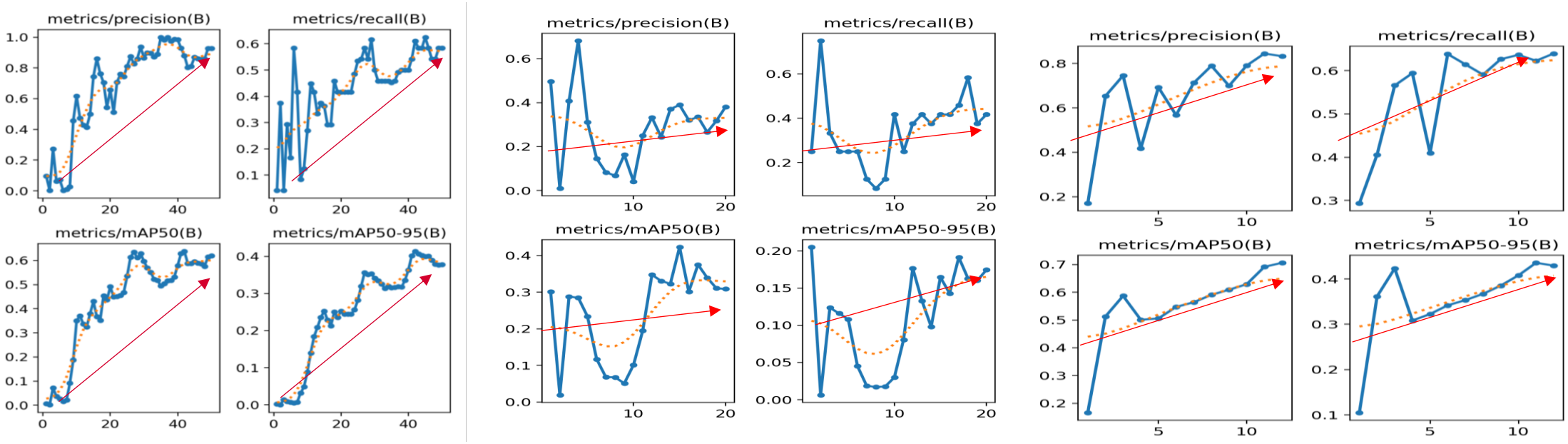
Wie viele abgerufene Elemente sind relevant?

$$\textbf{Precision} = \frac{\text{Relevante abgerufene Instanzen}}{\text{Alle abgerufenen Instanzen}}$$

Wie viele relevante Elemente wurden abgerufen?

$$\textbf{Recall} = \frac{\text{Relevante abgerufene Instanzen}}{\text{Alle relevanten Instanzen}}$$

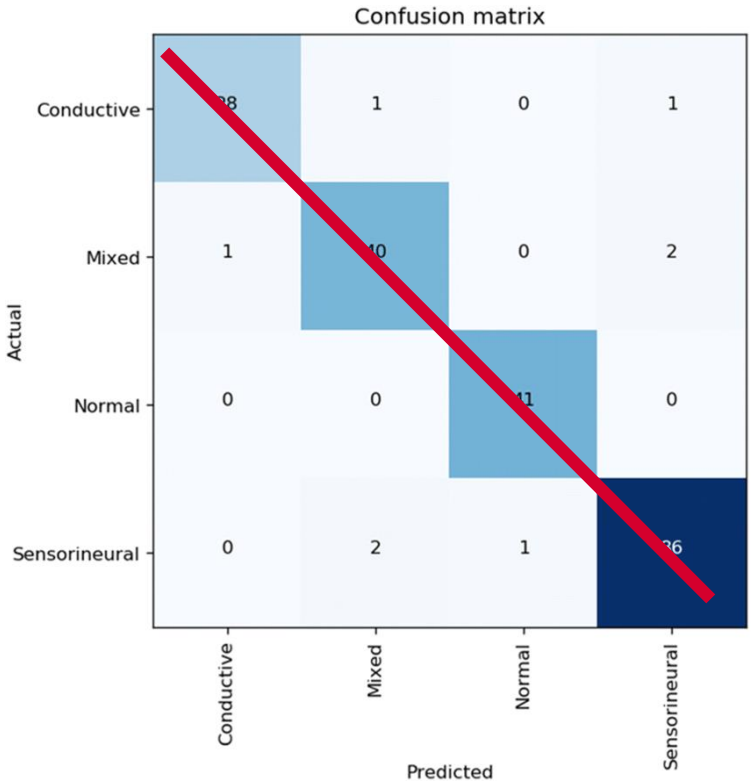
Beispiele für Precision und Recall

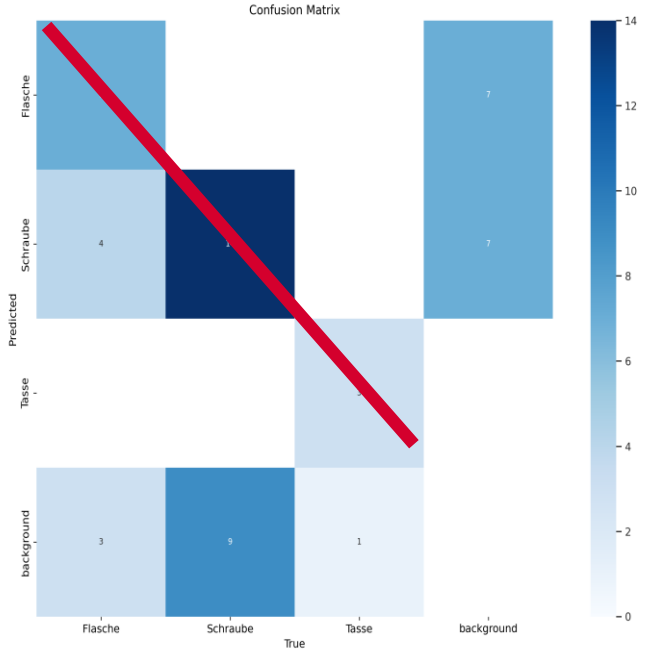


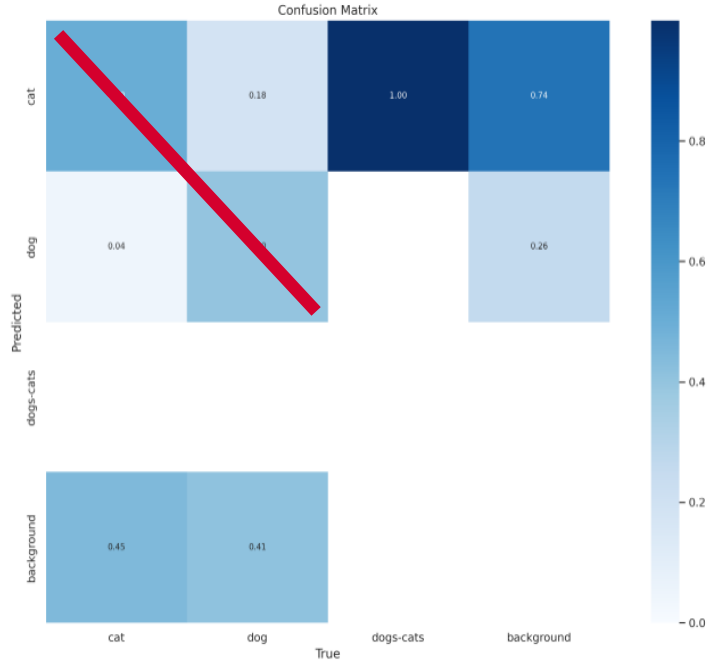
- Wenn sowohl Precision als auch Recall näher an 1 kommen, zeigt dies, dass das Modell sowohl genau als auch empfindlich ist.

- In verschiedenen Anwendungen haben Precision und Recall unterschiedliche Prioritäten: Im Gesundheitswesen ist hoher Recall wichtig, während bei Spam-Filtern hohe Precision bevorzugt wird.

Beispiele für Konfusionsmatrizen







Werden die Diagonalwerte in der Konfusionsmatrix dunkler, bedeutet das, dass das Modell mehr richtige Vorhersagen trifft. Steigende Werte in den Zellen für "True Positives" und "True Negatives" zeigen, dass das Modell beide Klassen korrekt klassifiziert.

Am Ball bleiben!

"Es ist nicht genug zu wissen – man muss auch anwenden. Es ist nicht genug zu wollen – man muss auch tun."

Johann Wolfgang von Goethe

B4KMU.de

AuxIndustryInsights

KI-Campus

KI PRODUKTIONS-NETZWERK AUGSBURG

Universität Augsburg University · DLR · Fraunhofer IGCV · Technische Hochschule Augsburg

HIGHTECH Agenda Bayern

Vielen Dank für Ihre Aufmerksamkeit!

@ amuuniversitataugsburg9607

@ mint_uni.augsburg

@ amu-uni-a

mint@amu.uni-augsburg.de

www.b4kmu.de

0176 / 5354871