TECHNICAL APPLICATIONS AND DATA MANAGEMENT. WS 2021/2022.

VORLESUNG 8

09.11.2021

MÜNCHEN

STUDIENGANG DIGITAL MANAGEMENT.

AGENDA

1. Aufgabenstellung Data Science
2. Case Study CNN

WAS HABEN WIR BIS JETZT GEMACHT?

ROADMAP	WAS HABEN WIR GEMACHT?
Vorlesung 1	Workflow Data Management, Datentypen und Datenqualität
Vorlesung 2	Einführung Data Science und Data Science Workflow, Grundlagen Data Management
Vorlesung 3	Grundlagen Stochastik: Wahrscheinlichkeitsrechnung, deskriptive und explorative Statistik
Vorlesung 4	Statistische Inferenz, lineare Regression
Vorlesung 5	Einführung Machine Learning, Unüberwachtes Lernen
Vorlesung 6	Überwachtes Lernen
Vorlesung 7	Neuronale Netze und Convolutional neural networks

1. AUFGABENSTELLUNG DATA SCIENCE

WAS IST IM RAHMEN PROJEKTARBEIT ZU TUN? 1. TEIL: SCHULTERBLICK DATA SCIENCE IN KW49.

- Zu erstellen ist eine Word-Datei à 4 Seiten für die gesamte Aufgabe mit:
  - Problem statement: „Welches Problem wollen wir lösen? Wieso ist es ein Problem? Was ist der Nutzen einer Lösung?"
  - Metriken zur Evaluation der Ergebnisse
  - Vorgehensweise Lösungsansatz anhand Data Science Workflow: „Wie gehen wir es an?"
  - Aufteilung Projektarbeit: wer in der Gruppe macht was? Eindeutige Zuordnung der Aufgaben ist wichtig!
  - Aktueller Status: gibt's Probleme? Wie kommen Sie voran?

- Vorstellung durch die Gruppen in der Vorlesung („Amazon"- Ansatz)

- Gemeinsame Diskussion

Templates finden Sie auf der Homepage des Kurses unter Materialien

WAS IST IM RAHMEN PROJEKTARBEIT ZU TUN? 2. TEIL: ABSCHLUSSPRÄSENTATION IN KW03

- Powerpoint-Präsentation: jede Gruppe präsentiert ihre Ergebnisse mit gesamthaft 20 - 25 Minuten

- Schriftliche Ausarbeitung je Teilnehmer mit ca. 12 Seiten:
  - Einleitung/ Projektübersicht
  - Related Work: welche Ansätze wurden verwendet? Was für ähnliche Ansätze gibt es? Wo sind die Unterschiede?
  - Vorgehensweise anhand Data Science Workflow (Get Data, Explore the Data, Model the Data, Visualise Results):
    - Beschreibung und Erklärung der eingesetzten Verfahren
    - Implementierung Datenaufbereitung, Datenmodellierung und Datenvisualisierung
  - Ergebnisse: Visualisierung Ergebnisse und Bewertung anhand Metriken
  - Reflektion: Was lief gut? Was lief schlecht?
  - Ausblick/ Future Work: „Was würden wir als nächste Schritte machen?“
  - Literaturverzeichnis

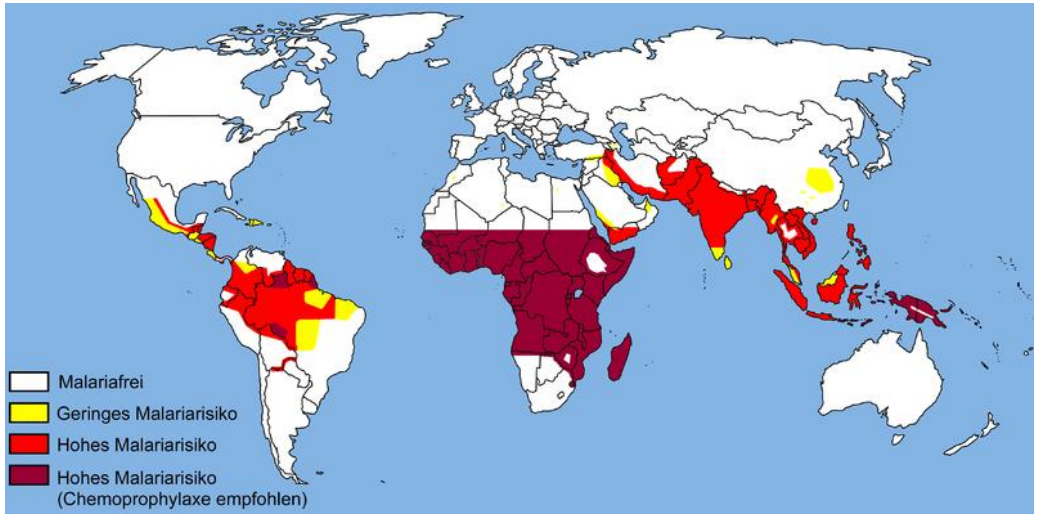
Templates finden Sie auf der Homepage des Kurses unter Materialien

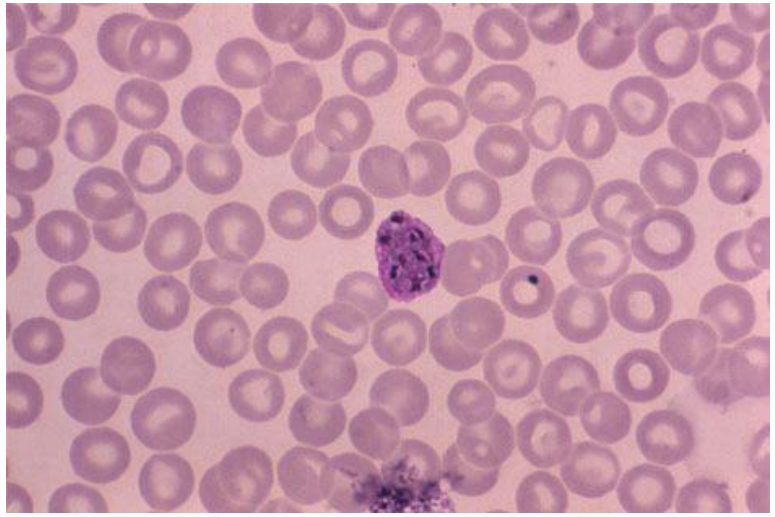
3. FALLBEISPIEL MALARIA

ANWENDUNGSFALL NEURAL NETWORKS: ERKENNEN INFIZIERTER ZELLEN MIT MALARIA.

- Malaria ist eine Infektionskrankheit, die durch den Stich einer Moskito übertragen wird.
- Häufigste Infektionskrankheit mit ca. 200 Mio. erkrankten Menschen pro Jahr.
- Diagnose: kostengünstigste Möglichkeit ist Analyse roter Blutkörperchen auf Plasmodien (einzellige Parasiten) per Mikroskop.

→ Aber: wie kann man sowas skalieren auf mehrere Mio. Leute?





Ziel: Automatisiertes Klassifizieren einer Zelle auf Infiziert oder Gesund per CNN

Quelle Bilder: Link

WIEDERHOLUNG GENERISCHER ABLAUF SUPERVISED LEARNING. ÜBERSICHT.

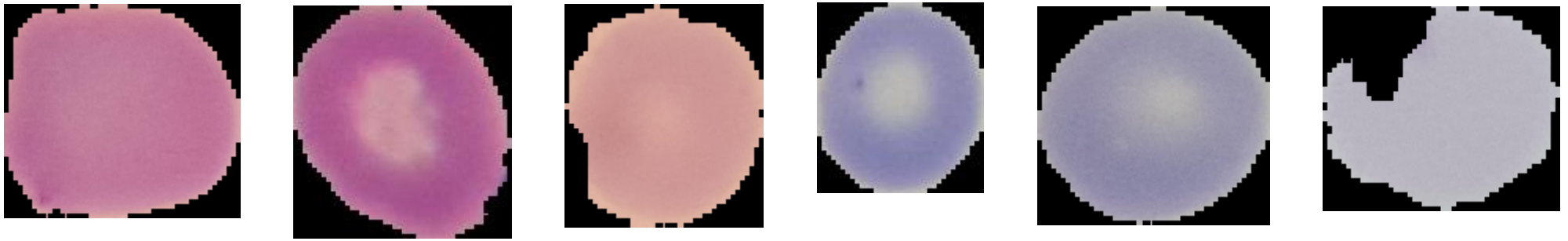
Notebook mit allen Schritten und Code liegt auf Github

1. Daten organisieren und hochladen.
2. Daten aufbereiten/ Data cleaning.
3. Daten aufteilen in Test- und Trainingsmenge (sowie ggf. Validierungsmenge).
4. Vorbereitungen: Machine Learning Verfahren wählen, Gewichte initialisieren, Kostenfunktion wählen.
5. Training: Schrittweise Optimierung Modellparameter bis das Modell möglichst gute Performance für die Trainingsmenge hat.
6. Modell(-güte) validieren anhand der Testmenge. Falls das Modell gut ist, weiter zu Schritt 7. Sonst zu Schritt 5.
7. Deployment: Modell einsetzen im „Live"-Betrieb inkl. kontinuierliches Überprüfen Güte Modell und Aktualisierung.

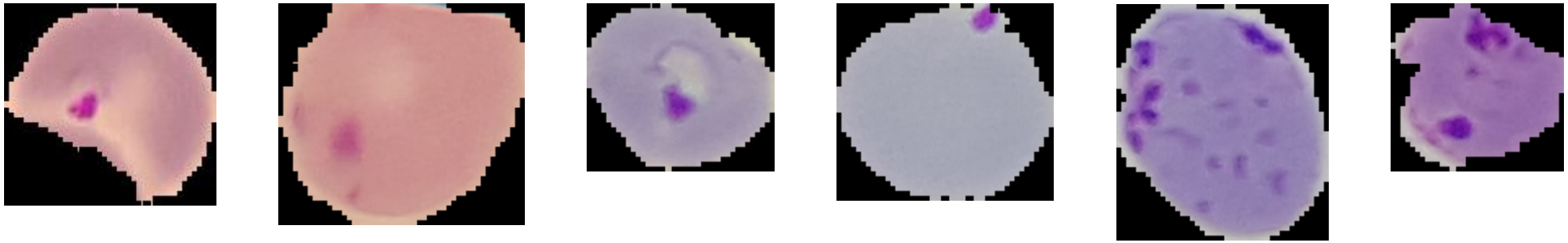
Ziel: Lernen eines möglichst genauen Modells H(Inputbild) = Infiziert oder Gesund

ERKENNEN INFIZIERTER ZELLEN MIT MALARIA. SCHRITT 1: DATEN ORGANISIEREN.

National Library of Medicine: 27560 Bildern



Gesunde Zellen



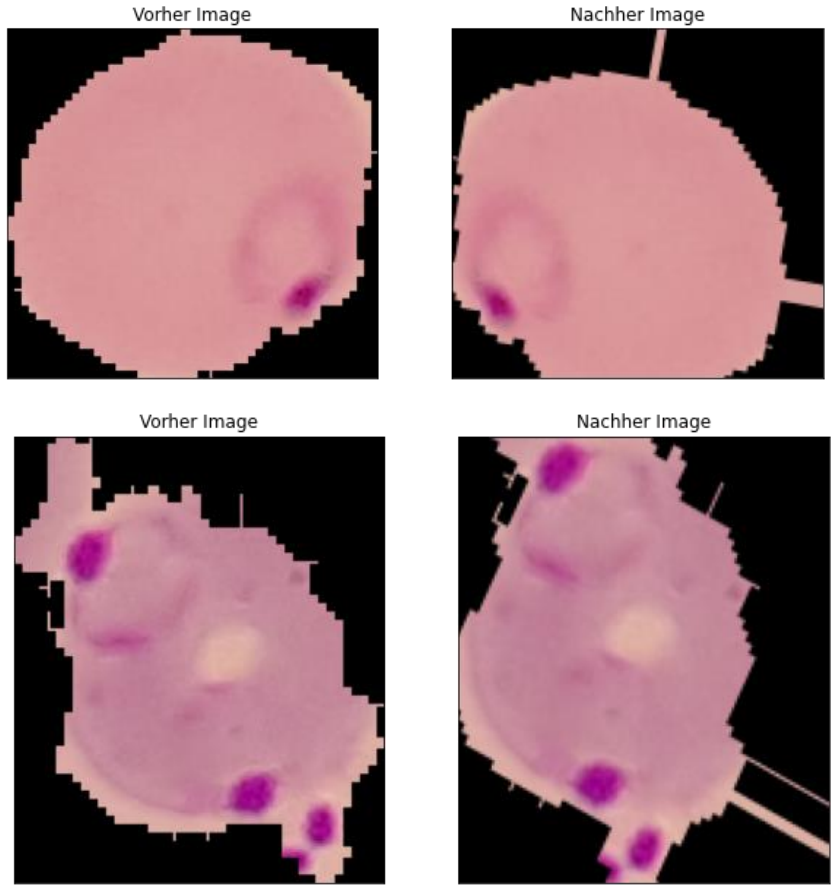
Infizierte Zellen

Wir sehen folgendes:

- Bilder haben verschiedene Größen.
- Zellen haben verschiedene geometrische Formen.
- Grund für Infektion scheint lila Objekt in Zelle zu sein. Dieses kann an verschiedenen Stellen in verschiedener Größe sein.

Quelle Bilder: National Library of Medicine: Malaria Dataset, Link

ERKENNEN INFIZIERTER ZELLEN MIT MALARIA. SCHRITT 2: DATEN AUFBEREITEN/ DATA CLEANING.



Data Augmentation generiert aus vorhandenen Bildern leicht veränderte Bilder als Input für das Modell.

Vorteile:

- Durch die veränderten Bilder wird der Algorithmus beim Lernen gezwungen zu generalisieren, das Modell wird somit robuster.
- Durch Data Augmentation kann ein viel größeres Datenset als vorhanden simuliert werden.
- Data Augmentation geschieht im Arbeitsspeicher.

Nachteile:

- Erhöhter Trainingsaufwand, da mehr Bilder.
- Modell kann Features lernen, die real nicht existieren (verzerrte Gesichter, Tiere in bestimmten Farben, …).

Transfer Learning ist wichtige Methode für das Sicherstellen der Stabilität von Machine Learning Modellen

Quelle Bilder: National Library of Medicine: Malaria Dataset, Link

DATA AUGMENTATION IM DETAIL: AUSGEWÄHLTE OPERATOREN.

Rotation: ImageDataGenerator(rotation_range=90)

Generiert Bilder mit zufällige Rotation um Wert Parameter

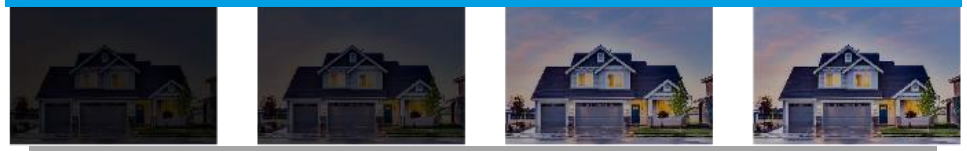
Vert. Schieben:ImageDataGenerator(width_shift_range=0.3)

Gen. Bilder mit zuf. Verschiebung links/ rechts < Parameter

Horiz. Schieben:ImageDataGenerator(heigth_shift_range=0.3)

Gen. Bilder mit zuf. Verschiebung oben/ unten < Parameter

Helligkeit:ImageDataGenerator(brightness_range=(0.1, 0.9))



Gen. Bilder mit zuf. Helligkeitswert Parameter; 0.0 = dunkel

Originalbild

HorizontalFlip: ImageDataGenerator(horizontal_flip=True)

Generiert Bilder inkl. zufälligem horiz. Vertauschen

Vertikaler Flip: ImageDataGenerator(vertical_flip=True)

Generiert Bilder inkl. zufälligem vertik. Vertauschen

Shear Intensity: ImageDataGenerator(shear_range=45.0)

Generiert Bilder durch Stretchen um best. Punkt um X Grad

Zoom: ImageDataGenerator(zoom_range=[0.5, 1.5])

Generiert Bilder per Zoom; < 1 = Vergrößerung

Nicht alle der erzeugten Bilder kommen in der Realität vor.
Darauf müssen Sie beim Einsatz von Data Augmentation Operatoren aufpassen!!

Quelle: Link

Vollständige Doku unter: Link
Tutorial unter: Link

ERKENNEN INFIZIERTER ZELLEN MIT MALARIA. SCHRITT 3: DATEN AUFTEILEN IN TEST-, TRAININGS UND VALIDIERUNGSMENGE.

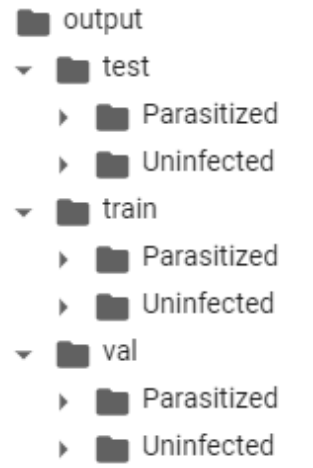
Originaldaten

13779 Bilder

13779 Bilder

```
import splitfolders
splitfolders.ratio(input_folder = '/content/cell_images',
                   output="output",
                   seed=1337, #reproduzierbare, zufällige Verteilung Daten
                   ratio=(.6, .2, .2), # 60% Trainings-, 20% Test, 20% Valid.daten
                   group_prefix=None)
```

Nach Aufteilung



5512 Bilder

16534 Bilder

5512 Bilder

Quelle: Splitfolder, verfügbar unter: Link

ERKENNEN INFIZIERTER ZELLEN MIT MALARIA. SCHRITT 4: MACHINE LEARNING VERFAHREN WÄHLEN.

Simples CNN

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	[(None, 150, 150, 3)]	0
conv2d (Conv2D)	(None, 150, 150, 32)	896
max_pooling2d (MaxPooling2D)	(None, 75, 75, 32)	0
dropout (Dropout)	(None, 75, 75, 32)	0
conv2d_1 (Conv2D)	(None, 75, 75, 64)	18496
max_pooling2d_1 (MaxPooling2	(None, 37, 37, 64)	0
dropout_1 (Dropout)	(None, 37, 37, 64)	0
conv2d_2 (Conv2D)	(None, 37, 37, 128)	73856
max_pooling2d_2 (MaxPooling2	(None, 18, 18, 128)	0
flatten (Flatten)	(None, 41472)	0
dense (Dense)	(None, 512)	21234176
dropout_2 (Dropout)	(None, 512)	0
dense_1 (Dense)	(None, 512)	262656
dropout_3 (Dropout)	(None, 512)	0
dense_2 (Dense)	(None, 1)	513

Total params: 21,590,593
Trainable params: 21,590,593
Non-trainable params: 0

Transfer Learning mit VGG

Layer (type)	Output Shape	Param #
input_3 (InputLayer)	[(None, 224, 224, 3)]	0
block1_conv1 (Conv2D)	(None, 224, 224, 64)	1792
block1_conv2 (Conv2D)	(None, 224, 224, 64)	36928
block1_pool (MaxPooling2D)	(None, 112, 112, 64)	0
block2_conv1 (Conv2D)	(None, 112, 112, 128)	73856
block2_conv2 (Conv2D)	(None, 112, 112, 128)	147584
block2_pool (MaxPooling2D)	(None, 56, 56, 128)	0
block3_conv1 (Conv2D)	(None, 56, 56, 256)	295168
block3_conv2 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv3 (Conv2D)	(None, 56, 56, 256)	590080
block3_conv4 (Conv2D)	(None, 56, 56, 256)	590080
block3_pool (MaxPooling2D)	(None, 28, 28, 256)	0
block4_conv1 (Conv2D)	(None, 28, 28, 512)	1180160
block4_conv2 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv3 (Conv2D)	(None, 28, 28, 512)	2359808
block4_conv4 (Conv2D)	(None, 28, 28, 512)	2359808
block4_pool (MaxPooling2D)	(None, 14, 14, 512)	0
block5_conv1 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv2 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv3 (Conv2D)	(None, 14, 14, 512)	2359808
block5_conv4 (Conv2D)	(None, 14, 14, 512)	2359808
block5_pool (MaxPooling2D)	(None, 7, 7, 512)	0
flatten_1 (Flatten)	(None, 25088)	0
dense_4 (Dense)	(None, 1)	25089

Total params: 20,049,473
Trainable params: 25,089
Non-trainable params: 20,024,384

Transfer Learning mit MobileNet

block_14_depthwise (DepthwiseCo	(None, 7, 7, 960)	8640	block_14_expand_relu[0][0]
block_14_depthwise_BN (BatchNor	(None, 7, 7, 960)	3840	block_14_depthwise[0][0]
block_14_depthwise_relu (ReLU)	(None, 7, 7, 960)	0	block_14_depthwise_BN[0][0]
block_14_project (Conv2D)	(None, 7, 7, 160)	153600	block_14_depthwise_relu[0][0]
block_14_project_BN (BatchNorma	(None, 7, 7, 160)	640	block_14_project[0][0]
block_14_add (Add)	(None, 7, 7, 160)	0	block_13_project_BN[0][0] block_14_project_BN[0][0]
block_15_expand (Conv2D)	(None, 7, 7, 960)	153600	block_14_add[0][0]
block_15_expand_BN (BatchNormal	(None, 7, 7, 960)	3840	block_15_expand[0][0]
block_15_expand_relu (ReLU)	(None, 7, 7, 960)	0	block_15_expand_BN[0][0]
block_15_depthwise (DepthwiseCo	(None, 7, 7, 960)	8640	block_15_expand_relu[0][0]
block_15_depthwise_BN (BatchNor	(None, 7, 7, 960)	3840	block_15_depthwise[0][0]
block_15_depthwise_relu (ReLU)	(None, 7, 7, 960)	0	block_15_depthwise_BN[0][0]
block_15_project (Conv2D)	(None, 7, 7, 160)	153600	block_15_depthwise_relu[0][0]
block_15_project_BN (BatchNorma	(None, 7, 7, 160)	640	block_15_project[0][0]
block_15_add (Add)	(None, 7, 7, 160)	0	block_14_add[0][0] block_15_project_BN[0][0]
block_16_depthwise_relu (ReLU)	(None, 7, 7, 960)	0	block_16_depthwise_BN[0][0]
block_16_project (Conv2D)	(None, 7, 7, 320)	307200	block_16_depthwise_relu[0][0]
block_16_project_BN (BatchNorma	(None, 7, 7, 320)	1280	block_16_project[0][0]
Conv_1 (Conv2D)	(None, 7, 7, 1280)	409600	block_16_project_BN[0][0]
Conv_1_bn (BatchNormalization)	(None, 7, 7, 1280)	5120	Conv_1[0][0]
out_relu (ReLU)	(None, 7, 7, 1280)	0	Conv_1_bn[0][0]
flatten_2 (Flatten)	(None, 62720)	0	out_relu[0][0]
dense_4 (Dense)	(None, 1)	62721	flatten_2[0][0]

Total params: 2,320,705
Trainable params: 62,721
Non-trainable params: 2,257,984

Auszug
(zu groß für 1 Screenshot)

ERKENNEN INFIZIERTER ZELLEN MIT MALARIA. SCHRITT 4: ÜBERSICHT TRANSFER LEARNING VERFAHREN.

Model	Size (MB)	Top-1 Accuracy	Top-5 Accuracy	Parameters	Depth	Time (ms) per inference step (CPU)	Time (ms) per inference step (GPU)
Xception	88	0.790	0.945	22,910,480	126	109.42	8.06
VGG16	528	0.713	0.901	138,357,544	23	69.50	4.16
VGG19	549	0.713	0.900	143,667,240	26	84.75	4.38
ResNet50	98	0.749	0.921	25,636,712	-	58.20	4.55
ResNet101	171	0.764	0.928	44,707,176	-	89.59	5.19
ResNet152	232	0.766	0.931	60,419,944	-	127.43	6.54
ResNet50V2	98	0.760	0.930	25,613,800	-	45.63	4.42
ResNet101V2	171	0.772	0.938	44,675,560	-	72.73	5.43
ResNet152V2	232	0.780	0.942	60,380,648	-	107.50	6.64
InceptionV3	92	0.779	0.937	23,851,784	159	42.25	6.86
InceptionResNetV2	215	0.803	0.953	55,873,736	572	130.19	10.02
MobileNet	16	0.704	0.895	4,253,864	88	22.60	3.44
MobileNetV2	14	0.713	0.901	3,538,984	88	25.90	3.83
DenseNet121	33	0.750	0.923	8,062,504	121	77.14	5.38
DenseNet169	57	0.762	0.932	14,307,880	169	96.40	6.28
DenseNet201	80	0.773	0.936	20,242,984	201	127.24	6.67
NASNetMobile	23	0.744	0.919	5,326,716	-	27.04	6.70
NASNetLarge	343	0.825	0.960	88,949,818	-	344.51	19.96
EfficientNetB0	29	-	-	5,330,571	-	46.00	4.91
EfficientNetB1	31	-	-	7,856,239	-	60.20	5.55
EfficientNetB2	36	-	-	9,177,569	-	80.79	6.50
EfficientNetB3	48	-	-	12,320,535	-	139.97	8.77
EfficientNetB4	75	-	-	19,466,823	-	308.33	15.12
EfficientNetB5	118	-	-	30,562,527	-	579.18	25.29
EfficientNetB6	166	-	-	43,265,143	-	958.12	40.45
EfficientNetB7	256	-	-	66,658,687	-	1578.90	61.62

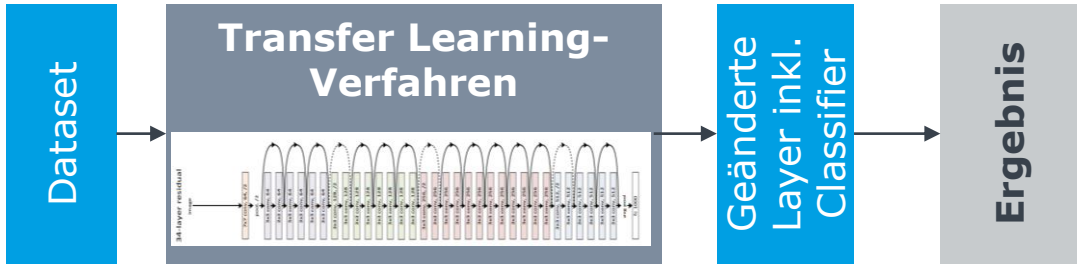
Erklärung Parameter:

- **Size:** Größe des zu speichernden Modells. Relevant für den Einsatz auf Geräten mit beschränktem Speicherplatz wie z.B. Handys oder IoT-Geräten.
- **Top-1 Accuracy:** gibt an, wie oft auf Daten des ImageNet-Datensatz genau die richtige Klasse erkannt wurde.
- **Top-5 Accuracy:** gibt an wie oft auf Daten des ImageNet-Datensatz eine von 5 gleichzeitigen Vorhersagen des Modells die richtige war.
- **Parameters:** Anzahl Parameter. Relevant falls das Modell angepaßt wird, da dies ein Indikator für die benötigte Rechenleistung und somit Dauer des Trainings ist.
- **Depth**: Tiefe des Netzes mit allen seinen Schichten.
- **Time per inference step**: wie lang Netz für die Inferenz braucht (also Auswertung Input in Netz).

ImageNet-Datensatz ist Benchmark für Bildererkennung

ERKENNEN INFIZIERTER ZELLEN MIT MALARIA. SCHRITT 4: WIE SETZEN WIR TRANSFER LEARNING EIN?

Anpassen/ Ersetzen spezifischer Netzumfänge:



Änderungen notwendig, da wir explizit gesunde von infizierten Zellen unterscheiden wollen.

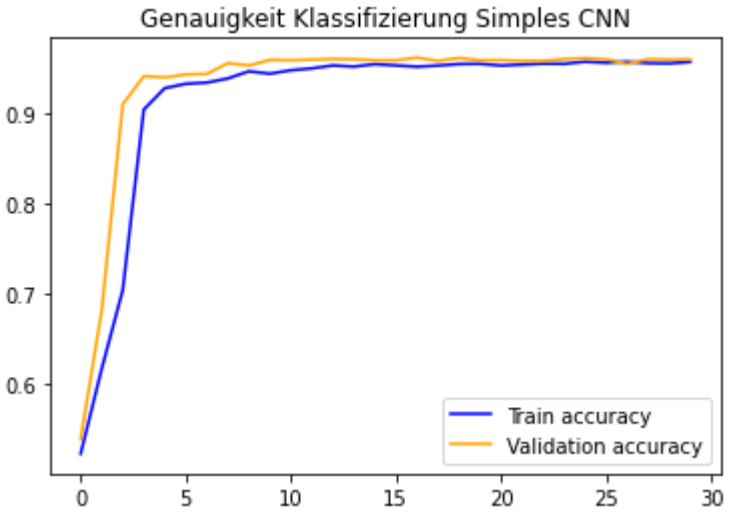
Deshalb müssen wir mindestens die letzte Schicht des Netzes anpassen, in der die Bilder klassifiziert werden.

Wie machen wir das?

- Definition einer Instanz des gewünschten Transfer Learning Modells.
- „Merken" der letzten Schicht des gewählten Modells.
- Anhängen der gewünschten, neuen Layer für Unterscheidung gesunde/ infizierte Zellen an die gemerkte, letzte Schicht.
- Modell kompilieren.
- Alle Schichten des Modells auf nicht trainierbar setzen und dann die hinzugefügten Schichten auf trainierbar setzen.
- Modell trainieren: die Schichten des Transfer Learning Modells bleiben gleich, wir trainieren also nur die neu hinzugefügten Schichten.

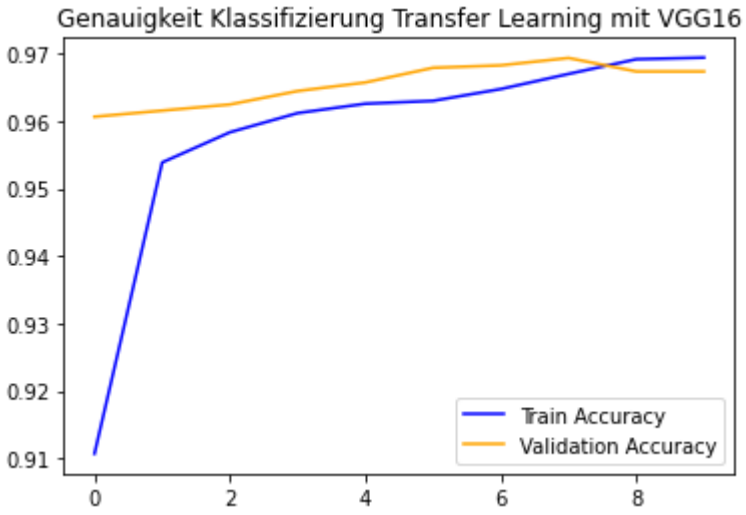
ERKENNEN INFIZIERTER ZELLEN MIT MALARIA. SCHRITT 5: TRAINING.

Simples CNN



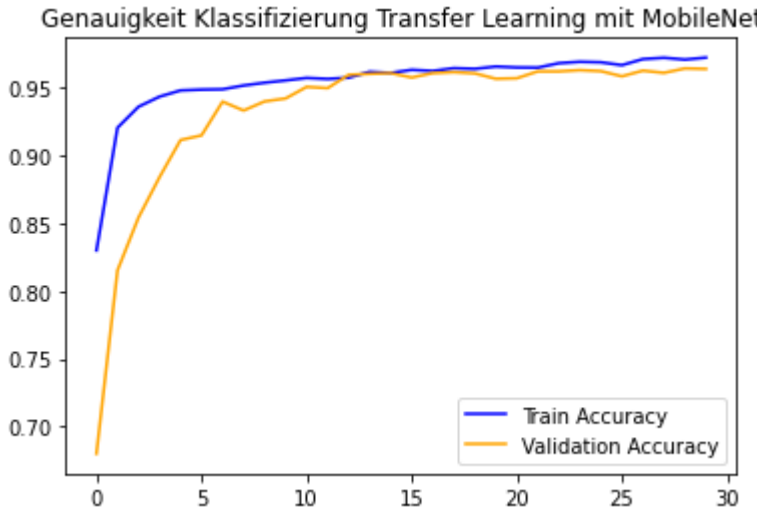
30 Epochen

Transfer Learning mit VGG16



10 Epochen

Transfer Learning mit MobileNet



30 Epochen

ERKENNEN INFIZIERTER ZELLEN MIT MALARIA. SCHRITT 6: VALIDIERUNG MODELLGÜTE PER TESTDATEN.

- Basic CNN:
  - Accuracy: 0.9554
  - Größe Modell: 247 MB

- VGG16:
  - Accuracy: 0.9623
  - Größe Modell: 153 MB

- MobileNet:
  - Accuracy: 0.9628
  - Größe Modell: 27 MB

Bei einem realen Einsatz des Modells würden wir deutlich mehr als die 10 oder 30 Epochen trainieren und auch die Hyperparameter tunen, um eine möglichst hohe Genauigkeit zu erhalten.

Dies dauert länger, aber bei einem real eingesetzten Modell sind falsche Klassifikationen sehr gefährlich!

CASE STUDY IN GRUPPENARBEIT

Basierend auf dem vorhandenen Notebook „VL 7 CNN and Transfer Learning - Malaria-v8.ipynb”:

1. **Data Augmentation**: wenden Sie verschiedene Data Augmentation Operatoren an auf Data Generator *train_image_gen.*
   - Welche der Operatoren bilden Zellen ab, die auch in der Realität existieren können?
   - Welche Operatoren sind eher nicht geeignet?
2. **Simple CNN-Netz:** experimentieren Sie mit dem Simple CNN-Netz. Ändern Sie die bestehenden Elemente oder bauen Sie die bekannten Elemente wie Layers, Dropout, Faltungen, MaxPooling etc. ein:
   - Welche ergeben eine höhere Genauigkeit?
   - Setzen Sie gerne die Anzahl der Epochen runter, um die Trainingsdauer zu beschleunigen.
3. **Transfer Learning:** probieren Sie andere Verfahren aus, wie bspw. Inception, ResNet, …
   - Welche ergeben eine höhere Genauigkeit?
   - Setzen Sie gerne die Anzahl der Epochen runter, um die Trainingsdauer zu beschleunigen.
4. Stellen Sie Ihre Ergebnisse, Hypothesen und Plots vor.

4. LITERATUR UND QUELLEN

QUELLEN:

Künstliche Intelligenz:

- Gröner, Heinecke: Kollege KI
- Burkov: The Hundred-Page Machine Learning Book, online verfügbar unter Link
- Nielsen: Neural Networks and Deep Learning, online verfügbar unter Link
- Russel, Norvig: Artificial Intelligence – a modern approach

Web-Links:

- https://karpathy.github.io/2015/05/21/rnn-effectiveness/
- https://colah.github.io/posts/2015-08-Understanding-LSTMs/
- https://distill.pub/2018/building-blocks/
- Attention: http://mlexplained.com/2017/12/29/attention-is-all-you-need-explained/

Notebooks:

- Sprache zu Text: https://colab.research.google.com/drive/1qFt8qxKtM05hRuRxsA1Lq4JtP7tstcgc