

Brain MRI Images for Brain Tumor Detection

zhaw Kathrin Noser
Dominique Neff
André Meier

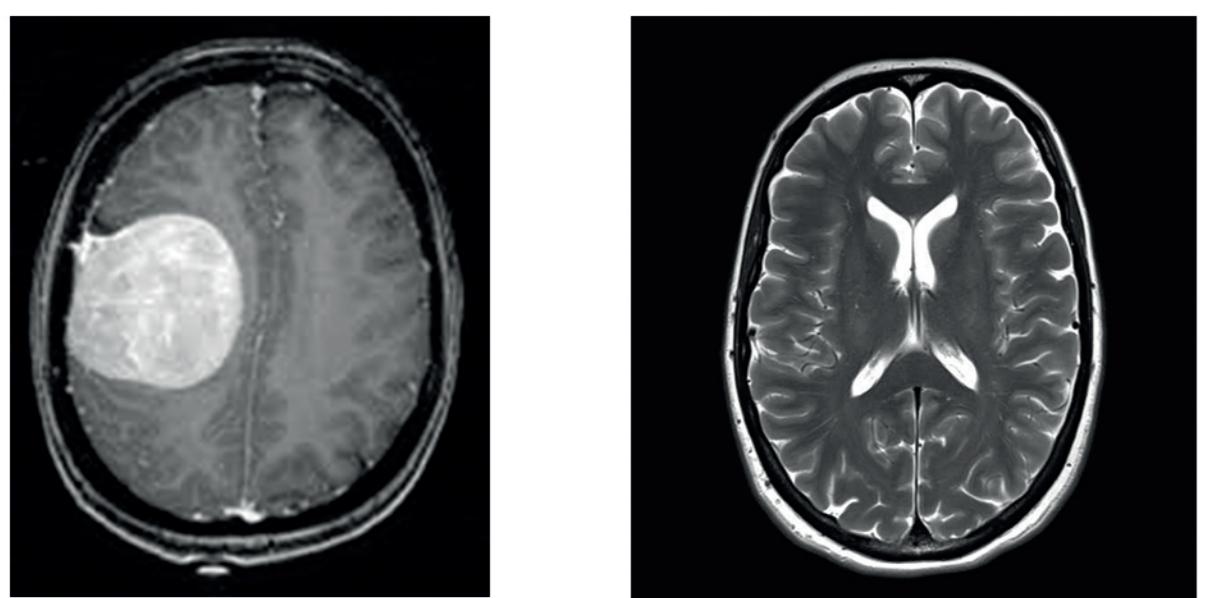


Motivation & Dataset

Ein Hirntumor gilt als eine der aggressivsten Krankheiten, sowohl bei Kindern als auch bei Erwachsenen. Jedes Jahr wird bei etwa 11.700 Menschen ein Hirntumor diagnostiziert. Die 5-Jahres-Überlebensrate für Menschen mit einem krebsartigen Hirn- oder ZNS-Tumor liegt bei etwa 34 % für Männer und 36 % für Frauen. Hirntumore werden unterteilt in: Gartiger Tumor, bösartiger Tumor, Hypophysentumor, usw. Die beste Technik zur Erkennung von Hirntumoren ist die Kernspintomographie (MRI). Durch die Aufnahmen wird eine große Menge an Bilddaten erzeugt. Diese Bilder werden von einem Radiologen untersucht. Eine manuelle Untersuchung kann aufgrund der Komplexität von Hirntumoren und deren Eigenschaften fehleranfällig sein.

Wir versuchten in diesem Projekt eine geeignete Deep Learning Lösung zur Erkennung von Hirntumoren zu finden. Dazu wurden verschiedene Modelle getestet.

Der Datensatz beinhaltet 253 MRI Bilder von Hirnscans in 2 Klassen. Die Bilder liegen grösstenteils im *.jpg Format vor und sind unterschiedlich gross.



Preprocessing

Die Breite und Höhe der Bilder reicht von unter 200px bis zu über 1400px. Die Proportionen zwischen Höhe und Breite sind nicht einheitlich. Wir müssen also die folgenden Hauptverarbeitungsschritte durchführen.

- Ändern der Grösse der Bilder auf eine identische Grösse. Dabei ist es darauf zu achten die Proportionen nicht zu stark zu verzerrn.
- Einlesen der Bilddaten in Arrays.
- Konstruieren der entsprechenden Label Vektoren
- Aufteilung der Daten in Trainings-, Validierungs- und Testmengen.
- Normalisieren

Modelling

Als Benchmarkmodell wird ein simpler Random Forest Algorithmus verwendet. Weiter werden verschiedene Convolutional Neural Networks, Ensembles sowie Transferlearning VGG16 Modelle getestet. Um effizient viele verschiedene Modelle und Parameter zu testen wurde ein Script programmiert über das Parameter per Excel in das Script überliefert werden und nach dem Training alle Scores und Plots zurückliefern. So wurden über 100 Modelle getestet. Die besten davon sind in den Results ersichtlich.

Results

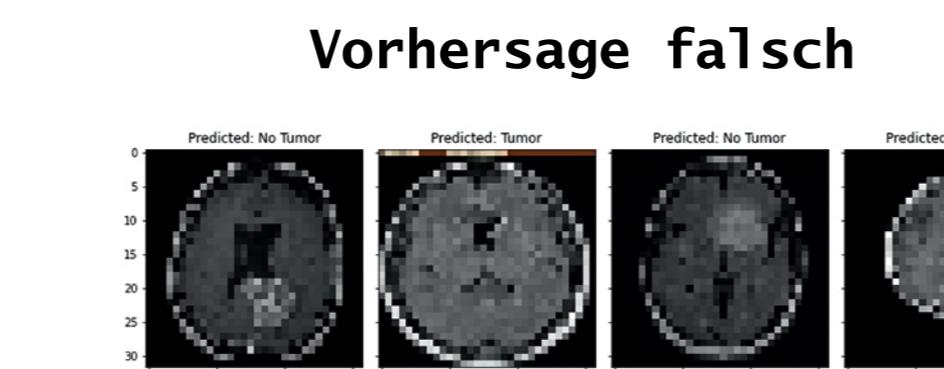
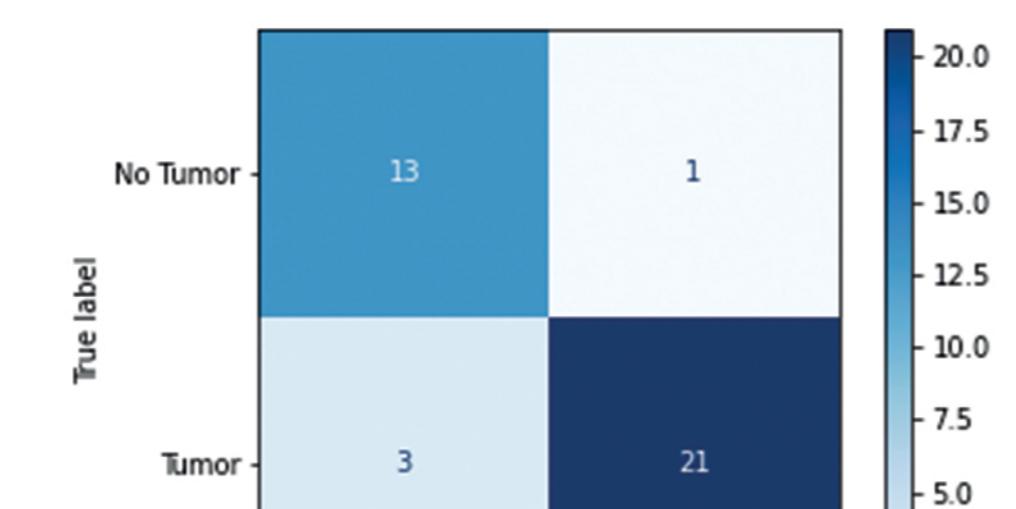
Baseline Model

Random Forest

Anzahl Trees = 40

Scores:

Accuracy: 89.47 %
F1: 91.30 %
Precision: 95.45 %
Recall: 87.50 %



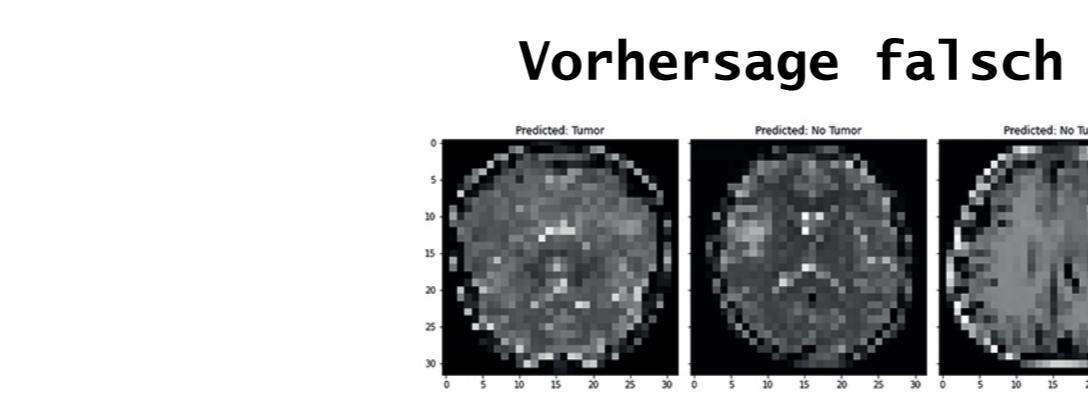
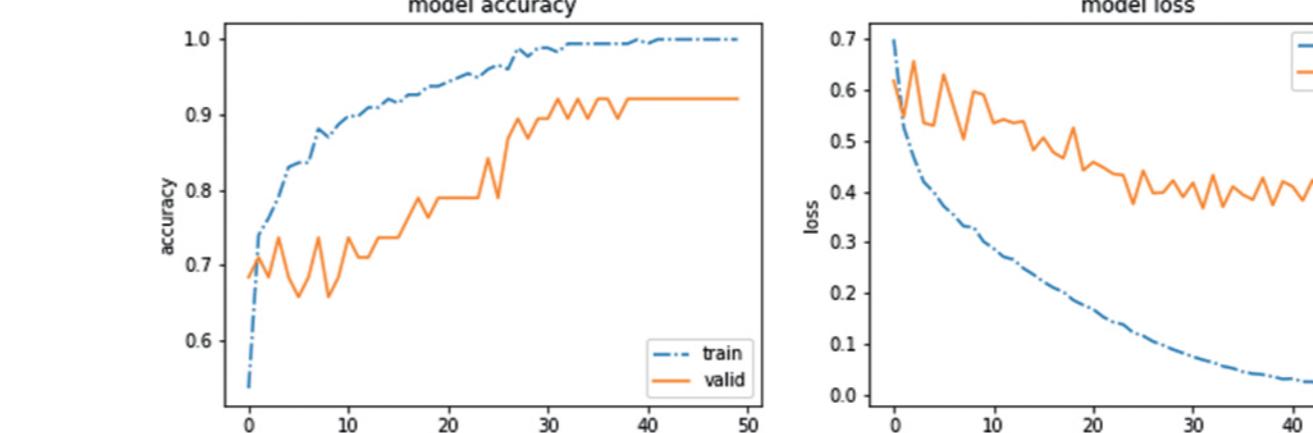
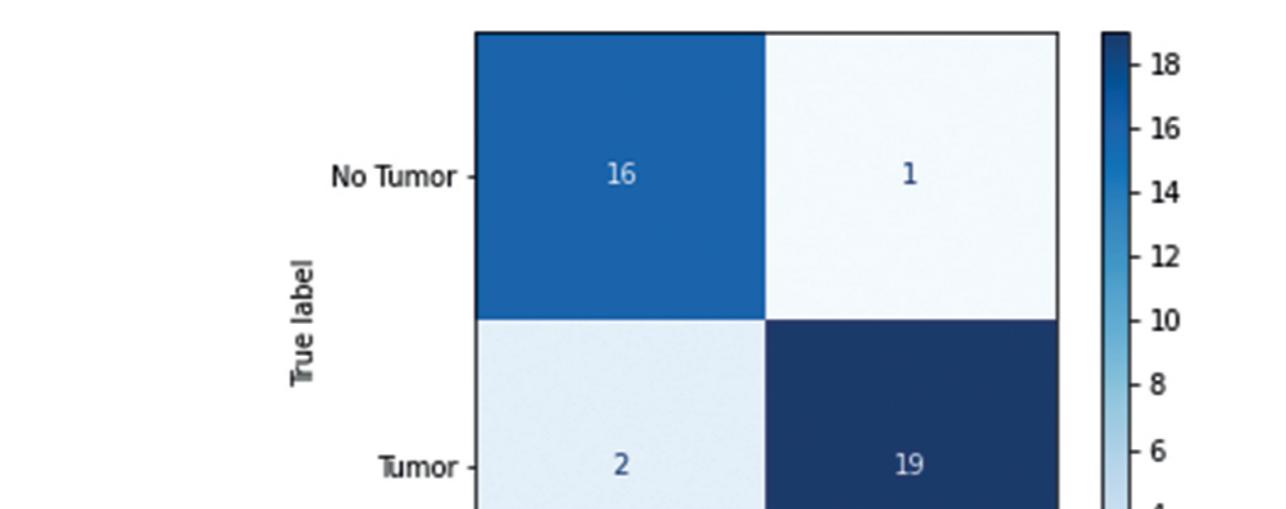
Best Model

CNN mit VGG16

Batch Size = 64
Epochen = 30
Anzahl Hidden Layers = 2
Neuronen = [200, 100]
Activation Output = Softmax
Augmentation = false
Dropout = false

Scores:

Accuracy: 92.11 %
F1: 92.68 %
Precision: 95.00 %
Recall: 90.48 %
NLL-Cross: 20.91 %

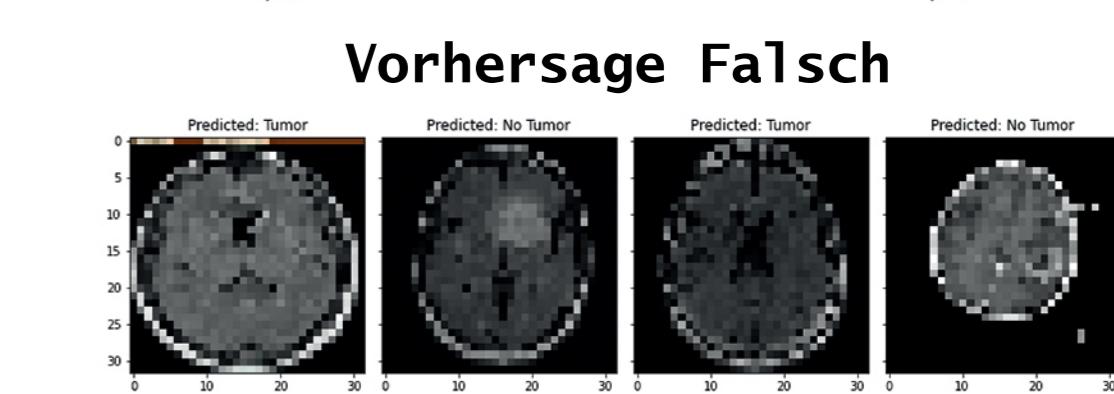
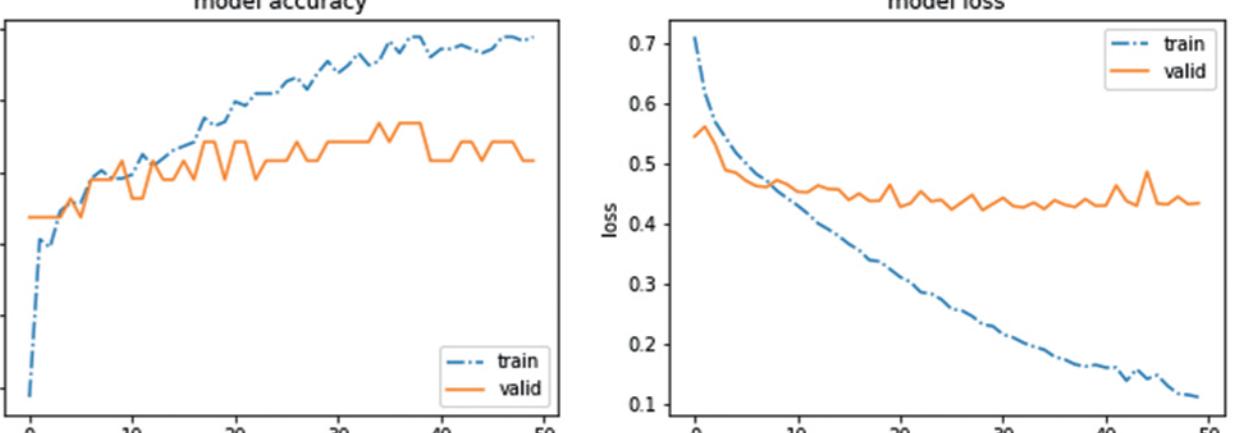
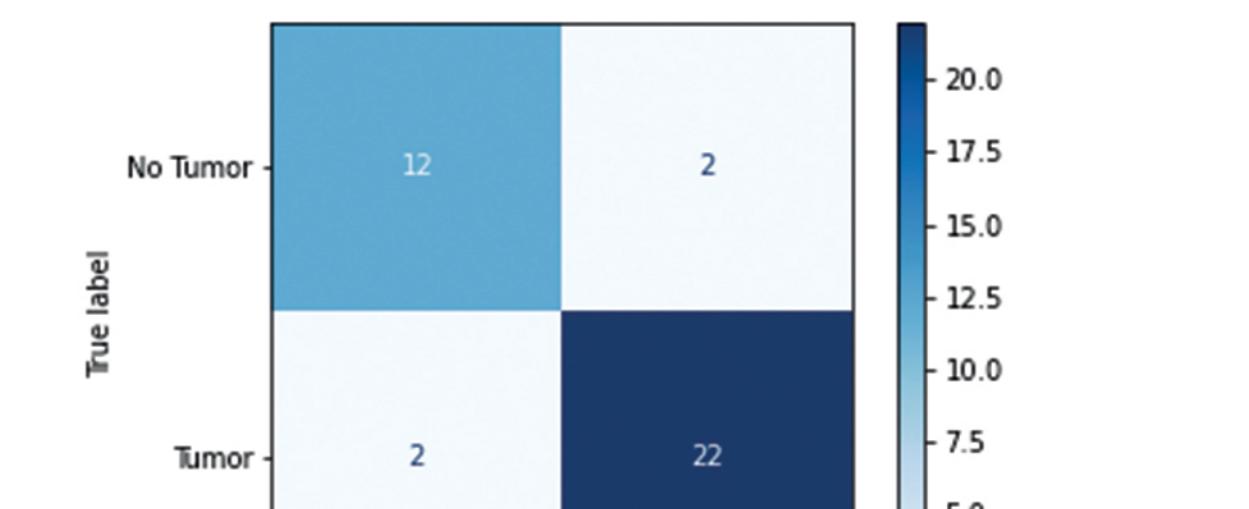


CNN

Batch Size = 64
Epochen = 50
Anzahl Hidden Layers = 0
Convolutional Layer mit Max-Pooling = 1
Kernels = 16
Kernel Size = [2, 2]
Padding = Same
Activation = ReLU
Activation Output = Softmax
Augmentation = false
Dropout = false

Scores:

Accuracy: 89.47 %
F1: 91.67 %
Precision: 91.67 %
Recall: 91.67 %
NLL-Cross: 23.46 %



Summary

Die Modelle haben eine Accuracy von 50% deutlich überschritten und sind somit besser als zufälliges Raten.

Von allen getesteten Modellen zeigte sich, dass Modelle mit Transferlearning (VGG16) am besten abgeschnitten haben. Es wurden verschiedene Image Sizes ausprobiert jedoch wurden keine Verbesserungen mit grösseren Bildern beobachtet.

Mit anderem Seed für den Daten-Split können die Resultate stark variieren. Einzig das Modell CNN mit VGG16 war mit Abstand das Robusteste. Die Accuracy lag stets über 80%. Bei anderen Modellen ist die Accuracy zum Teil unter 80% gefallen

Im Allgemeinen war die Performance mit Seed = 128 deutlich besser als andere Modelle welche mit Seed = 30 getestet wurden. Das kann daran liegen, dass die Bilder entsprechend vorteilhafter gesplittet wurden.

Mit MC-Dropout wurden ebenfalls gute Resultate erzielt. Diese Modelle schafften es jedoch nicht in die Top 3 von unserem Projekt.

Lessons Learned

- Der Datensatz ist mit 253 Bildern viel zu klein.
- Die Klassen waren nicht gut ausbalanciert. ("Yes" hat beinahe doppelt soviele Bilder als "No"). Konnte jedoch durch geeignete Klassengewichte korrigiert werden.
- Die Data Augmentation hat bei einfacheren Modellen nicht funktioniert. Die Situation wurde sogar verschlechtert.
- Bei komplexeren Modellen zeigte sich jedoch eine Verbesserung.
- Bei einer so einfachen Klassifizierung mit so wenigen Daten, sind Zufallseffekte ein Problem beim Vergleich der Modelle.
- Bei diesem Projekt brachten einfachere Modelle bei wenig Daten die bessere Performance.
- Modelltuning ist Erfahrungssache. Es gibt kein Patentrezept.