Brain Scan Classification – Deep Learning Final Work

Eylon Yehiel

Abstract

This project develops a deep learning system for automatic classification of brain tumors using MRI and CT scan images. Leveraging convolutional neural networks (CNNs), we implemented multiple approaches including models trained from scratch and transfer learning using the VGG16 architecture. The system was trained and evaluated on a dataset of 9,618 images from Kaggle, achieving accuracy rates above 97% for both MRI and CT scan classifications. The project demonstrates the potential of deep learning in medical imaging diagnostics, particularly for illness detection and classification.

1. Introduction

Brain tumor detection and classification are critical tasks in medical diagnosis that traditionally rely heavily on expert human interpretation of medical imaging. This manual process is time-consuming and susceptible to human error. This project addresses this challenge by developing an automated classification system using deep learning techniques.

Motivation

- Reduce the potential for human error in tumor detection.
- Provide a reliable second opinion tool for medical professionals.
- Improve efficiency in medical image analysis.
- Create a scalable solution for different types of medical imaging.

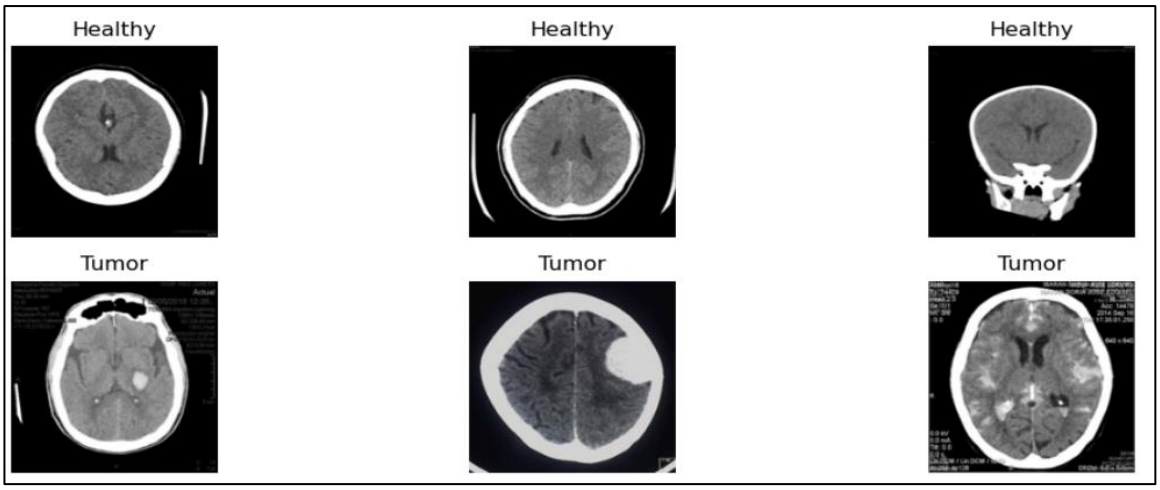
Objectives

1. Develop accurate classification models for both MRI and CT scan images with option for scaling it for another types of scans or photos.
2. Compare performance between models trained from scratch and transfer learning approaches, using and not using data augmentation.
3. Evaluate the effectiveness of data augmentation in improving different model performances.

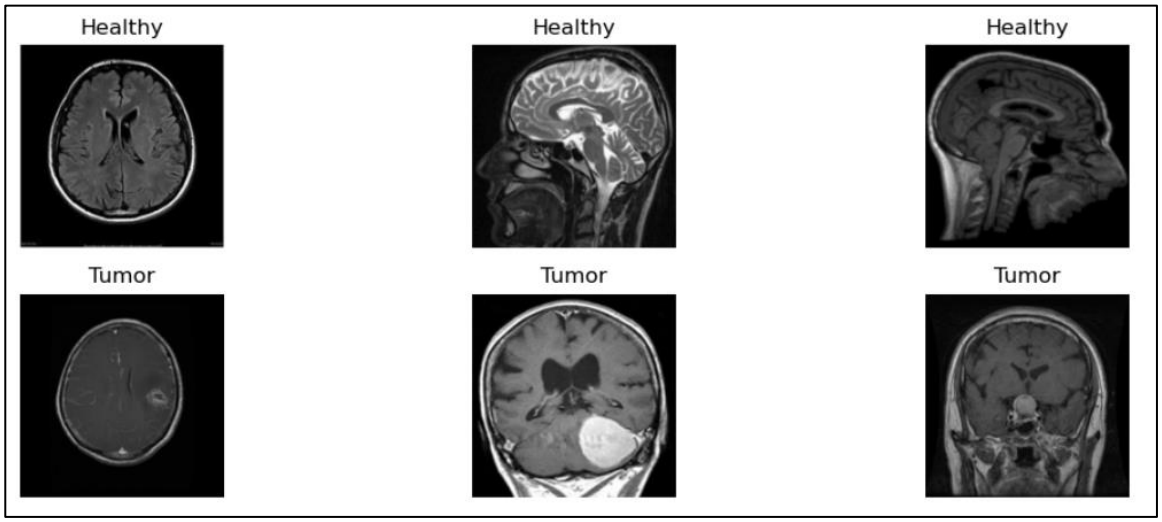
2. Dataset

Data has been downloaded from Kaggle ("Brain tumor multimodal image dataset") and contains 9,599 total images: 4,981 MRI scans and 4,618 CT scans. The images are labeled with binary classes "0" for health and "1" for tumor. Images size is 180X180 pixels with 3 color channels (RGB). Although the ability to resize to higher resolution, I chose to keep it as is in manner of runtime. Before the data inserted to network it has been rescaled – every pixel in every picture is divided by maximal pixel value (255). In pretrained models I used the model's predefined preprocessing function.

From the whole dataset, 20% defined as test set, and from the rest 80% - another 20% defined as validation set. The Train-validation-test split division at the end is 64-16-20.



CT input examples



MRI input examples

3. Methodology

For both CT and MRI scan images I used the same models with the same architectures to compare models' illness recognition ability on both cases. CNN's – Convolutional Neural Networks are proven to have the most efficiency in image recognition due to their depth that enables the networks to learn patterns in pictures. For each scan type in separate – CT and MRI -, several CNN architectures have been investigated:

1. **Model from scratch:** after many iterations of try and error of changing number of units, adding and removing dense, normalization and dropout layers, and changing activation function – best combination chosen. Input form is (180,180,3) as mentioned. A rescaling layer [0,1] inserted at the start to help the model make calculations. The network has five Conv2D layers with increasing numbers of filters (32, 64, 128, 256, and 256), each using a 3x3 kernel and ReLU activation to capture patterns such as edges, textures, and higher-level features. These are interspersed with MaxPooling2D layers, which reduce the spatial dimensions of the feature maps, lowering the data dimensions to retain the most significant features. After feature extraction, the Flatten layer converts the 2D feature maps into a vector, which is passed to a Dense output layer with a single neuron and a sigmoid activation function. This final layer outputs a probability value, indicating the likelihood that the input belongs to one of the two classes. The net compiled with binary-cross-entropy loss, RMSprop optimizer and evaluated with accuracy score. The model trained on 30 epochs of the train set with batch size of 30. Best parameters are saved due to validation loss monitor.

2. **Data augmentation addition:** To generalize the model to capture vary forms, angles and more potentially noisy attributes, we would like to teach the model how it is looks like. So, it is executed by adding an augmentation layer before the rescale layer and proceeding with same architecture as 'model from scratch'. Augmentation actions are random horizonal flip, random rotation of ±36 degrees (10% of 360 degrees), random zoom the images up to 20% and random brightness adjustments (±20%). I added a Dropout layer to prevent overfitting and gave 100 epochs in the favor of repeated learning.

3. **Transfer learning – VGG16:** To take advantage of previous work and training progress I chose to examine the use of feature extraction from pretrained VGG-16 model with "ImageNet" weights. Conducted by taking one before last layer's predictions and train on them a simple network with only Flatten and Dense layer of 256 units**.**

4. **Transfer learning with data augmentation:** To do so, I push the augmentation layer to enrich image representations after input layer, and after it concatenated it with VGG16 architecture.

5. **Fine tuning pretrained model:** Freeze all but the last 4 layers of the VGG model, enabling fine-tuning only on the final layers, allows the VGG model's layers to be trainable, and helps leverage pre-trained features while adapting to the new dataset. RMSprop optimizer is used with a very small learning rate to prevent overfitting during fine-tuning.

Loss and accuracy values are tracked along each model training for emphasizing overfit points when the validation loss is still relatively high despite the fact accuracy might be high. For this reason, the checkpoint of each model – saving the best performances – defined by lowest validation loss value throughout all epochs.

For networks evaluation and comparison, we will use the known sensitivity analysis indices **– accuracy, precision, recall** and **F1 score**. I found **confusion matrix** and **classification report** as the most informative representations of those matrices. An extremely long training time is also a consideration, so we look for a balance between runtime and performance.

For the reason we are dealing with medical subjects – a good model's quality evaluation will be **recall of positive cases**, **means the ratio between model's successful predictions of illness and the actual (true) such cases** in our train dataset.

4. Results

MRI:

	Accuracy
feature_extraction_vgg16_mri	0.991
fine_tuning_vgg_mri	0.989
from_scratch_mri	0.982
feature_extraction_with_augmentation_mri	0.981
from_scratch_with_augmentation_mri	0.975

	Positive recall
fine_tuning_vgg_mri	0.992
feature_extraction_vgg16_mri	0.990
from_scratch_mri	0.983
feature_extraction_with_augmentation_mri	0.983
from_scratch_with_augmentation_mri	0.978

As can be noticed, there is no absolute constancy in the findings and there is no superior method for MRI scans among the selected models. In terms of our chosen evaluation metric – **VGG16 fine-tuned model shows best results with 99.2% of true illness predicted correctly**, and with error of no more than 5 entities classified as healthy. It should be noted that other metrics might yield different ratings (e.g., for precision of healthy samples – VGG feature extraction model gains 99.3% with only 3 misclassified samples. Another interesting result is the **negative effect caused by**

data augmentation for both metrics – the reason can be the use of Dropout layer aside augmentation process.

CT:

	Accuracy
from_scratch_with_augmentation	0.982
from_scratch	0.972
feature_extraction_vgg16	0.970
fine_tuning_vgg	0.961
feature_extraction_with_augmentation	0.955

	Positive recall
from_scratch_with_augmentation	0.966
feature_extraction_vgg16	0.959
from_scratch	0.957
feature_extraction_with_augmentation	0.942
fine_tuning_vgg	0.938

In general, CNNs are better recognizer of MRI scans than CT scans. The most interesting outcome of this work is that **model from scratch with data augmentation is superior CT scans predictor for both metrics – accuracy (98.2%) and positive recall (96.6%).** But, on the other hand, for MRI predictions – same model with same parameters shows the lowest ranks. This phenomenon might be good material for future investigation.

5. Summary

CNNs generally performed better on MRI scans than CT scans, emphasizing the inherent differences in imaging modalities and the need for further exploration into their unique challenges. The contrasting results between MRI and CT data, particularly the effectiveness of data augmentation, provide an intriguing avenue for future research to understand and optimize model performance for varying image types.

In conclusion, while the project successfully demonstrated high performance in brain tumor detection, it also uncovered significant insights and posed new questions for future exploration, particularly regarding the variability of model performance across different imaging modalities and the role of data augmentation. These findings reinforce the importance of tailoring deep learning approaches to the specific nuances of medical imaging data.

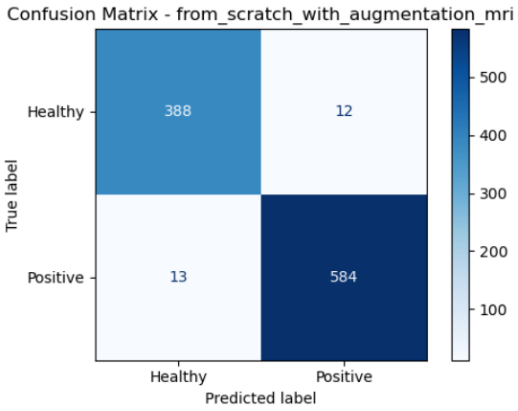
6. Links:

- Project repository: https://github.com/EylonYehiel/Brain-Scan-Classification-Deep-Larning
- Data source: https://www.kaggle.com/datasets/murtozalikhon/brain-tumor-multimodal-image-ct-_and-mri

7. Appendices:

from_scratch_with_augmentation_mri:

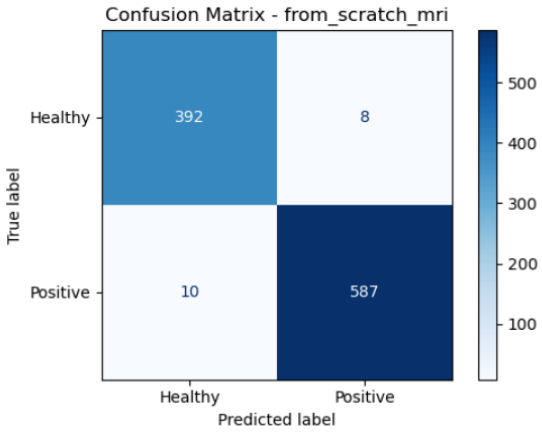
32/32 ——— 13s 379ms/step



	Healthy	Positive	macro avg	weighted avg
precision	0.968	0.980	0.974	0.975
recall	0.970	0.978	0.974	0.975
f1-score	0.969	0.979	0.974	0.975
support	400.000	597.000	997.000	997.000

from_scratch_mri:

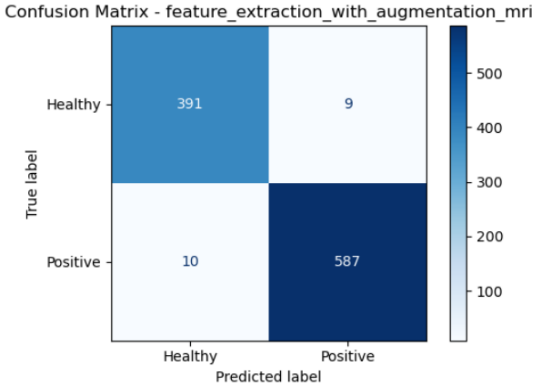
32/32 ——— 13s 373ms/step



	Healthy	Positive	macro avg	weighted avg
precision	0.975	0.987	0.981	0.982
recall	0.980	0.983	0.982	0.982
f1-score	0.978	0.985	0.981	0.982
support	400.000	597.000	997.000	997.000

feature_extraction_with_augmentation_mri:

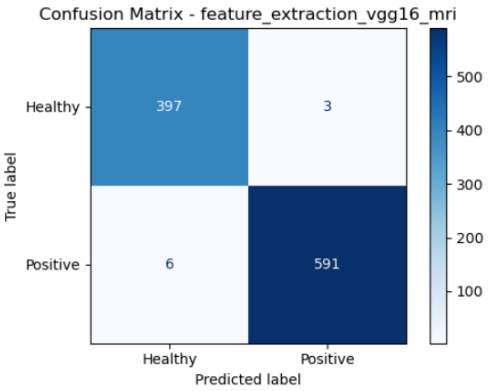
32/32 ——— 164s 5s/step



	Healthy	Positive	macro avg	weighted avg
precision	0.975	0.985	0.98	0.981
recall	0.978	0.983	0.98	0.981
f1-score	0.976	0.984	0.98	0.981
support	400.000	597.000	997.00	997.000

feature_extraction_vgg16_mri:

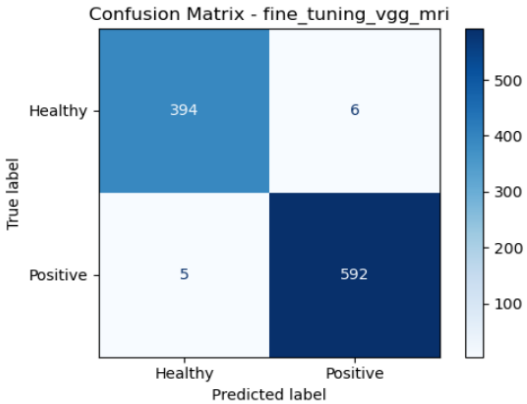
32/32 ——— 1s 16ms/step



	Healthy	Positive	macro avg	weighted avg
precision	0.985	0.995	0.990	0.991
recall	0.992	0.990	0.991	0.991
f1-score	0.989	0.992	0.991	0.991
support	400.000	597.000	997.000	997.000

fine_tuning_vgg_mri:

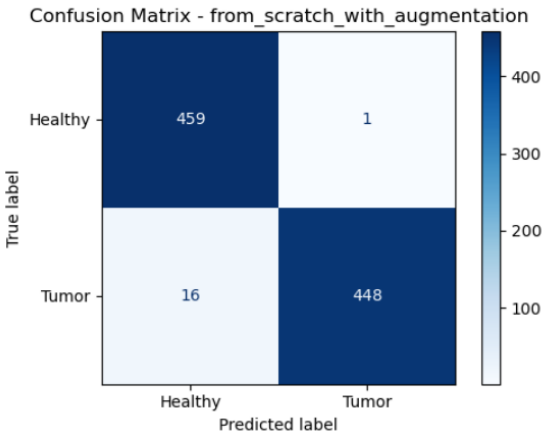
32/32 ——— 179s 6s/step



	Healthy	Positive	macro avg	weighted avg
precision	0.987	0.990	0.989	0.989
recall	0.985	0.992	0.988	0.989
f1-score	0.986	0.991	0.989	0.989
support	400.000	597.000	997.000	997.000

from_scratch_with_augmentation:

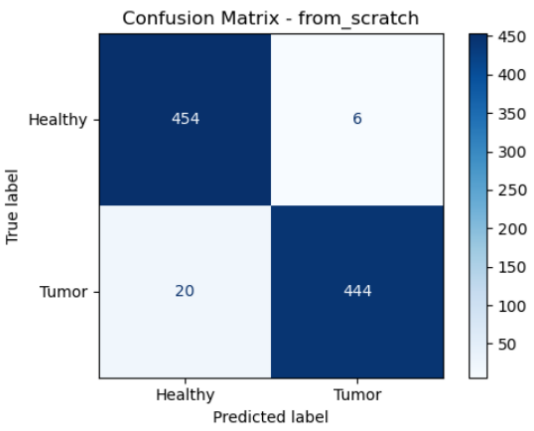
29/29 ——— 5s 153ms/step



	Healthy	Tumor	macro avg	weighted avg
precision	0.966	0.998	0.982	0.982
recall	0.998	0.966	0.982	0.982
f1-score	0.982	0.981	0.982	0.982
support	460.000	464.000	924.000	924.000

from_scratch:

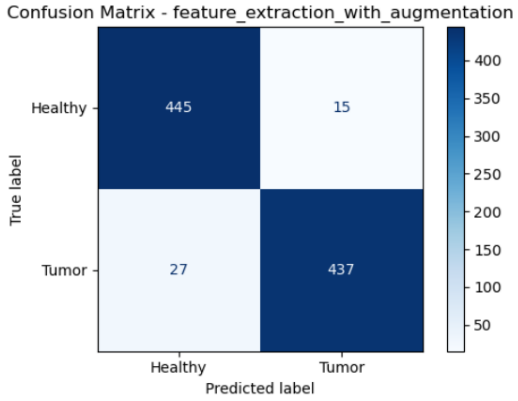
29/29 ——— 4s 145ms/step



	Healthy	Tumor	macro avg	weighted avg
precision	0.958	0.987	0.972	0.972
recall	0.987	0.957	0.972	0.972
f1-score	0.972	0.972	0.972	0.972
support	460.000	464.000	924.000	924.000

feature_extraction_with_augmentation:

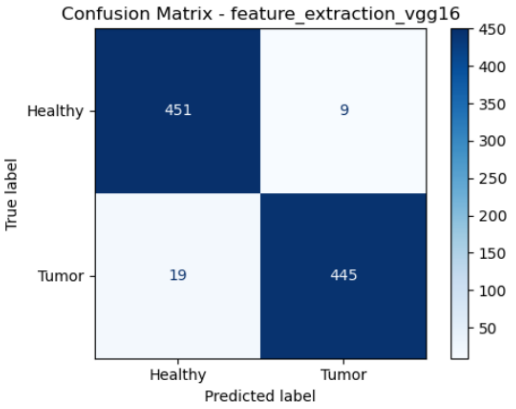
29/29 ——— 83s 3s/step



	Healthy	Tumor	macro avg	weighted avg
precision	0.943	0.967	0.955	0.955
recall	0.967	0.942	0.955	0.955
f1-score	0.955	0.954	0.955	0.955
support	460.000	464.000	924.000	924.000

feature_extraction_vgg16:

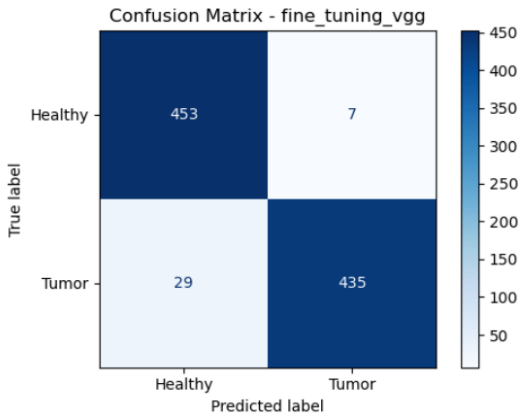
29/29 ——— 0s 6ms/step



	Healthy	Tumor	macro avg	weighted avg
precision	0.96	0.980	0.97	0.97
recall	0.98	0.959	0.97	0.97
f1-score	0.97	0.969	0.97	0.97
support	460.00	464.000	924.00	924.00

fine_tuning_vgg:

29/29 ——— 124s 4s/step



	Healthy	Tumor	macro avg	weighted avg
precision	0.940	0.984	0.962	0.962
recall	0.985	0.938	0.961	0.961
f1-score	0.962	0.960	0.961	0.961
support	460.000	464.000	924.000	924.000