Brain Tumor Classification in MRI images using CNN

Luca Lussu

***Abstract*—In this study, we explored the application of machine learning models for the diagnosis of brain tumors using a dataset of brain images from patients diagnosed with brain tumors. We implemented and tested several convolutional neural network (CNN) models to classify the images into two categories: ”tumor” and ”normal”. The tested models include a standard CNN version, a pre-trained version of ResNet [1], and an enhanced custom CNN version. In addition to defining and training the models, we also discussed performance optimization through the appropriate selection of loss functions and optimizers. The results show that the trained models achieved significant accuracy in classifying brain images, highlighting the effectiveness of machine learning in medical image analysis.**

I. Introduction

Early and accurate diagnosis of brain tumors is crucial for ensuring timely treatment and improving patients’ prospects for recovery. However, the interpretation of brain radiographic images can be complex and require specialized expertise from radiologists. In this context, the use of advanced image analysis techniques [2], such as machine learning and deep learning, represents a promising opportunity to enhance the diagnosis and management of brain tumors.

The Brain Tumor Dataset [3] used in this project consists of a set of brain radiographic images from patients diagnosed with brain tumors. Through the analysis of these images, we aim to develop classification models capable of distinguishing between images with tumor presence and normal images, in order to support medical diagnosis.

Various approaches have been explored for model development, including the use of convolutional neural networks (CNNs) and the employment of pre-trained architectures such as ResNet. The objective has been to obtain accurate and generalizable models, capable of precisely recognizing the presence of brain tumors in radiographic images.

This work documents the process of development, training, and evaluation of these models, as well as the results obtained. The ultimate goal is to provide a significant contribution to the medical community [4], facilitating early diagnosis and management of brain tumors through automated analysis of radiographic images.

II. Related Work

There are various methods for classifying magnetic resonance images; this article will primarily discuss those involving the use of custom *CNN* architectures and pre-trained architectures through transfer learning. However, positive results have also been found through the use of supervised learning algorithms that process categorical and numerical features extracted from magnetic resonance images.

For example, in a recently published work, as can be seen in the article by [5], a modified *CNN* model based on *ResNet18* and pre-trained achieved a nominal accuracy of 98.67%, significantly improving upon conventional *CNNs*. This model, called *OMRES*, uses the *RMSProp* algorithm with a *dropout* rate of 0.5, optimizing various hyperparameters such as learning rates, *batch* sizes, and the number of epochs.

III. Proposed Approach

A. Dataset and Preprocessing

The data used in this project consists of a dataset of brain magnetic resonance imaging (MRI) images from patients with brain tumors. The dataset has been divided into two classes: images with tumor presence and normal images. In total, the dataset contains 4600 images with 2513 brain tumor images and 2087 normal images. The ”Brain Tumor Dataset” used was downloaded from the Kaggle website. The images were preprocessed to reduce dimensions and standardize resolution, in order to facilitate processing by classification models. In our study, we specifically focus on two main attributes concerning the presence of brain tumors and their absence: ”Healthy” and ”Tumor”. The dataset was split into a training set (70

Below are some sample images taken from the dataset used in Fig 1

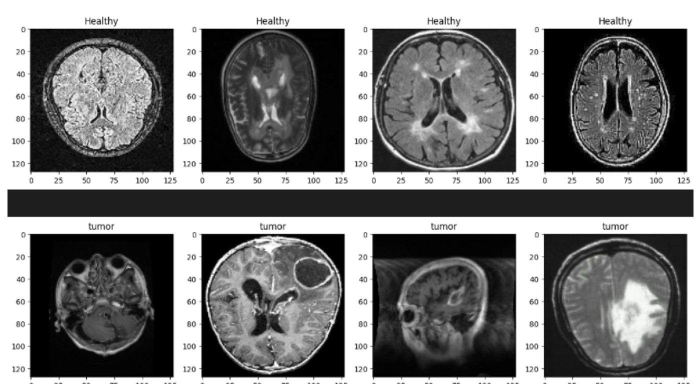

Fig. 1: Example of brain magnetic resonance imaging (MRI) images from patients

Subsequently, we can see the distribution of images in the dataset in Fig 2

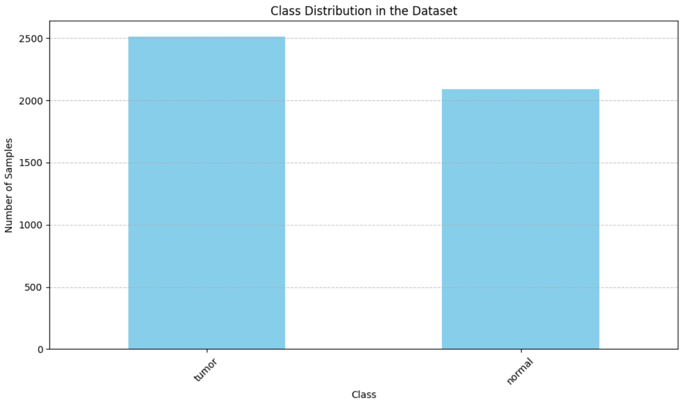

Fig. 2: Distribution of images in the dataset

In conclusion, the images were resized to 128x128 pixels and normalized to facilitate processing by classification models.

B. Model Architecture

The first proposed solution consists of using a custom convolutional neural network (CNN) for the classification of brain magnetic resonance imaging (MRI) images, developed specifically for this project. The network is composed of several convolutional layers for feature extraction followed by max pooling layers [6] for image dimension reduction, and dense FC layers [7] for binary classification.

The network architecture shown in Fig 3 is composed as follows:

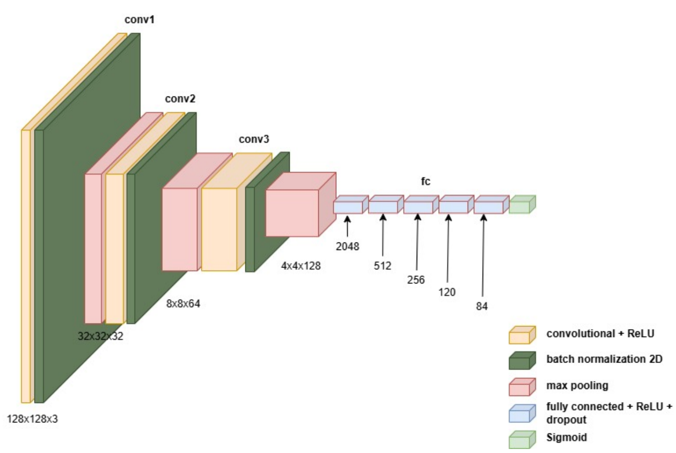

Fig. 3: Architecture of the custom convolutional neural network (CNN)

$$\texttt{INPUT} \rightarrow [\texttt{CONV} \rightarrow \texttt{BATCHNORM} \rightarrow \texttt{RELU} \rightarrow \texttt{MAXPOOL}] * 3 \rightarrow$$

$$\texttt{FC} \rightarrow \texttt{RELU} \rightarrow \texttt{DROPOUT} \rightarrow \texttt{FC}$$

The model can be analyzed in more detail by considering each individual layer, knowing that for each convolutional block, a 3x3 kernel size is applied as it has been shown to be the best for feature extraction:

1) As input, it receives images of dimensions $3 \times 128 \times 128$.
2) The first convolutional layer consists of 32 3×3 filters applied with padding 1 and stride 2. It is followed by batch normalization, a ReLU activation function, and max pooling with kernel size 2×2.
3) The second convolutional layer is composed of 64 3×3 filters with padding 1 and stride 2, followed by batch normalization, ReLU, and max pooling with kernel size 2×2.
4) The third convolutional layer uses 128 3×3 filters with padding 1 and stride 1, followed by batch normalization, ReLU, and max pooling with kernel size 2×2.
5) After the convolutional blocks, the feature maps are flattened and passed through a fully connected (FC) layer with 512 neurons and ReLU as the activation function.
6) Subsequently, there is a dropout layer with a rate of 0.6 to prevent overfitting.
7) This is followed by a second FC layer with 256 neurons and ReLU as the activation function, followed by another dropout layer.
8) The third FC layer is composed of 120 neurons and ReLU as the activation function, followed by a dropout layer.
9) The fourth FC layer is composed of 84 neurons and ReLU as the activation function, followed by a dropout layer.
10) The fifth and final FC layer, which is our output, is composed of 1 neuron and a sigmoid activation function.

The described architecture was obtained after several attempts and tests, in order to obtain an accurate and generalizable model for the classification of brain magnetic resonance imaging (MRI) images. This network has demonstrated the ability to effectively learn image characteristics and classify them accurately, as shown by the experimental results reported in the following section.

C. Pre-trained ResNet18 Model

In addition to the custom convolutional neural network (CNN), the option of using a pre-trained model for image classification was explored. This model is ResNet18, a convolutional neural network (CNN) pre-trained on ImageNet [8], the result of research work carried out by Microsoft Research and a series of tests where multiple versions of ResNet were evaluated, such as ResNet-18, ResNet-34, ResNet-50, ResNet-101, and ResNet-152, where ResNet18 is a lighter and faster model compared to the others with good performance.

The fundamental peculiarity of architectures of this type is represented by residual connections, which allow the transit of unprocessed information through the layers of the network. The design of ResNet is based on residual blocks, each of which is formed by several convolutional layers followed by a sum operation, which combines the original input with the output of the convolutional transformation. This approach favors efficiency in training and facilitates gradient propagation. ResNet18 represents one of the lightest variants of the ResNet family, with a total of 18 layers, which include both convolutional and fully-connected layers.

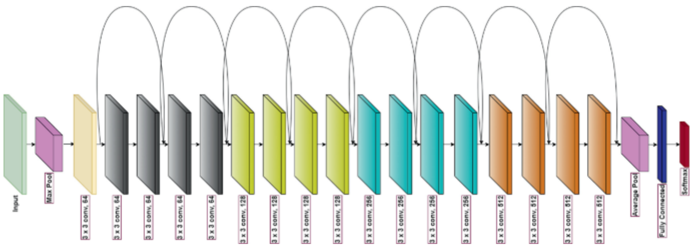

Fig. 4: Architecture of the pre-trained ResNet18 model

To adapt ResNet18 to the problem of binary classification of color MRI brain tumor images, it is necessary to make some modifications to its original structure. Since the images in our dataset are in color in the RGB format, no modification to the input channel dimension of the first convolutional layer is required.

However, considering that we are addressing a binary classification problem, we need to adjust the dimensions of the last fully connected layer so that it produces an output with two neurons instead of 1000, to adapt to the two classes in our dataset.

These modifications will allow us to use ResNet18 as a pre-trained feature extractor for our color MRI brain tumor image dataset, leveraging its learning capabilities on more generic and complex images provided by ImageNet, but customizing it for our specific binary classification context.

D. Alternative Version of CNN

An additional alternative to the previous version of the CNN seen above is the following architecture:

The proposed model, named `BetterBrainCNN`, is a convolutional neural network designed for the classification of brain magnetic resonance imaging (MRI) images. Below is a description of its architecture:

$$\texttt{INPUT} \rightarrow [\texttt{CONV} \rightarrow \texttt{BATCHNORM} \rightarrow \texttt{RELU} \rightarrow \texttt{CONV} \rightarrow$$

$$\texttt{BATCHNORM} \rightarrow \texttt{RELU} \rightarrow \texttt{MAXPOOL}] \times 4 \rightarrow$$

$$\texttt{FLATTEN} \rightarrow [\texttt{FC} \rightarrow \texttt{RELU} \rightarrow \texttt{DROPOUT}] \times 3 \rightarrow$$

$$\texttt{FC} \rightarrow \texttt{SIGMOID} \rightarrow \texttt{OUTPUT}$$

The model can be analyzed in more detail by considering each individual layer:

1) As input, it receives images of dimensions $3 \times 128 \times 128$.
2) The first convolutional layer consists of 64 3×3 filters applied with padding 1. It is followed by batch normalization, a ReLU activation function, and a second convolutional layer with 64 3×3 filters applied with padding 1. After each convolutional layer, there is batch normalization and a ReLU activation function. Finally, there is max pooling with kernel size 2×2 that reduces the dimensions to $64 \times 64 \times 64$.
3) The second convolutional layer is composed of 128 3×3 filters applied with padding 1. It is followed by batch normalization, a ReLU activation function, and a second convolutional layer with 128 3×3 filters applied with padding 1. After each convolutional layer, there is batch normalization and a ReLU activation function. Finally, there is max pooling with kernel size 2×2 that reduces the dimensions to $32 \times 32 \times 128$.
4) The third convolutional layer is composed of 256 3×3 filters applied with padding 1. It is followed by batch normalization, a ReLU activation function, and a second convolutional layer with 256 3×3 filters applied with padding 1. After each convolutional layer, there is batch normalization and a ReLU activation function. Finally, there is max pooling with kernel size 2×2 that reduces the dimensions to $16 \times 16 \times 256$.
5) The fourth convolutional layer is composed of 512 3×3 filters applied with padding 1. It is followed by batch normalization, a ReLU activation function, and a second convolutional layer with 512 3×3 filters applied with padding 1. After each convolutional layer, there is batch normalization and a ReLU activation function. Finally, there is max pooling with kernel size 2×2 that reduces the dimensions to $8 \times 8 \times 512$.
6) After the convolutional blocks, the feature maps are flattened and passed through a fully connected (FC) layer with 1024 neurons and ReLU as the activation function.
7) Subsequently, there is a dropout layer with a rate of 0.5 to prevent overfitting.
8) This is followed by a second FC layer with 512 neurons and ReLU as the activation function, followed by another dropout layer.
9) The third FC layer is composed of 256 neurons and ReLU as the activation function followed by a dropout layer.
10) The fourth and final FC layer, which is our output, is composed of 1 neuron and a sigmoid activation function.

The described architecture was obtained after several attempts and tests, in order to obtain an accurate model and test how it would perform with different convolutional and dropout layers.

IV. Experiments

In this section, we will describe the experimental details related to the development and evaluation of classification models based on convolutional neural networks (CNNs) discussed earlier and pre-trained architectures such as ResNet18. This is made possible through the use of a dataset of brain radiographic images from patients with brain tumors provided by Kaggle, and the utilization of custom scripts.

A. Development Environment

The project was developed in a Windows 11 environment, using Python scripts and libraries such as PyTorch, NumPy, Pandas, Matplotlib, Scikit-learn, and Jupyter Notebook. The code was executed on a computer with an AMD Ryzen 5 processor, 16GB of RAM, and an NVIDIA GeForce RTX 3060 graphics card with 6GB of memory.

B. Custom CNN Training

The custom network was defined and subsequently trained through a series of phases, during which various CNN configurations were tested on the training set and validated on the validation set, using accuracy as the main metric. The network described in the previous chapter underwent a detailed analysis. Its architecture, based on general CNN principles, was optimized through multiple modifications: deeper architectures were explored, and convolutional, pooling, and fully-connected layers were modified. The final configuration was chosen because it showed the most promising results.

The filters in the convolutional layers were all set to 3x3, as this format is more efficient and allows for increasing the depth of the network without significantly increasing the number of parameters. The batch size can influence various aspects of neural network performance; for this reason, training was conducted with different batch sizes. From the results on the validation set, the optimal size was found to be 64. During the experiments, the network was trained using the Adam optimizer with different learning rates. Values between 0.01 and 0.0001 were tested to determine the optimal value for the model and dataset. This range of values allowed for analyzing the impact of the learning rate on the training process and the final performance of the model. The results obtained indicated that a learning rate of 0.001 provided the best performance on the validation set, ensuring stable convergence and a significant reduction in the loss function during training.

During the training of the custom network, various regularization techniques were tested to improve the model's performance and prevent overfitting. In particular, dropout with a rate of 0.6 was applied after the fully-connected layers, and batch normalization layers [9] were added after the convolutional layers. These techniques allowed for reducing overfitting and improving the model's generalization. After various tests, better performance was observed with a dropout rate of 0.6. The network was finally trained on the training set for 19 epochs and tested on the validation set, as an early stopping mechanism [10] was implemented to prevent model overfitting.

The chosen parameters were:

- Epochs: 50
- Batch size: 64
- Learning rate: 0.001
- Dropout rate: 0.6
- Optimizer: Adam
- Loss function: BCELoss
- Patience: 5

The following image shows the loss and accuracy as the epochs progress.

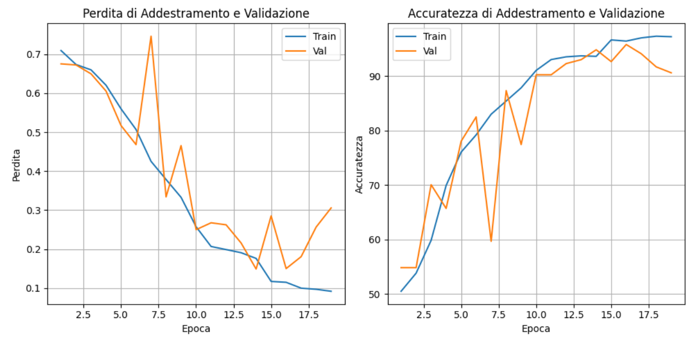

Fig. 5: Loss and accuracy over epochs

The model achieved an accuracy of 90.58% on the validation set, showing good performance in classification. The performance will be evaluated in the next section.

C. ResNet18 Training

For ResNet18 training, a transfer learning approach was used, leveraging a pre-trained model on ImageNet and adapting it to the task of classifying brain radiographic images. The model was trained on the training set and validated on the validation set, using accuracy as the main metric. As previously described, some modifications were made to the network to adapt it to the binary classification task. In particular, the last fully-connected layer was replaced with a new output layer with a single neuron and sigmoid activation function for binary classification.

The choice of the ResNet18 model was motivated by a series of tests evaluating multiple pre-trained architectures, including ResNet34, ResNet50, and ResNet101, which obtained inferior results compared to ResNet18. As with the custom network, different values of batch size and learning rate were tested. For this type of model, I chose to use the weights of the pre-trained model and perform fine-tuning. This latter technique can be very effective in adapting the model to a specific task, allowing the model to learn more relevant features for the new dataset, but with a higher computational cost.

The model results are shown in the following graphs:

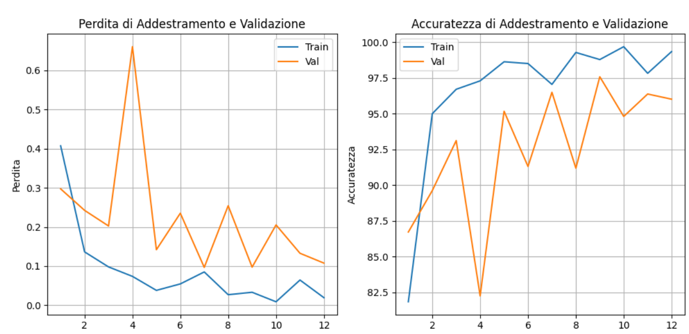

Fig. 6: Loss and accuracy over epochs

The pre-trained model achieved an accuracy of 96.01% on the validation set, showing better performance [11] compared to the custom network shown earlier.

D. Training of Alternative CNN Version

To evaluate further approaches and configurations, an alternative version of the custom network was developed, with some modifications to its architecture. In particular, additional convolutional and pooling layers were added to increase the complexity and depth of the network, adding a greater number of filters and increasing the size of the convolutional layers.

BetterBrainCNN consists of four convolutional blocks followed by a fully connected block. Each convolutional block is composed of two convolutional layers followed by batch normalization and ReLU activation, with a max pooling layer at the end to reduce the spatial dimensions of the image. The convolutional blocks progressively extract features of increasing complexity from the input images.

The fully connected block consists of four linear layers with ReLU activation functions between each pair of layers, followed by dropout to reduce the risk of overfitting. The final layer is a single unit with a sigmoid activation function, designed to produce a binary classification prediction.

The chosen parameters remain identical to the previous custom network. The results obtained are shown in the following graphs:

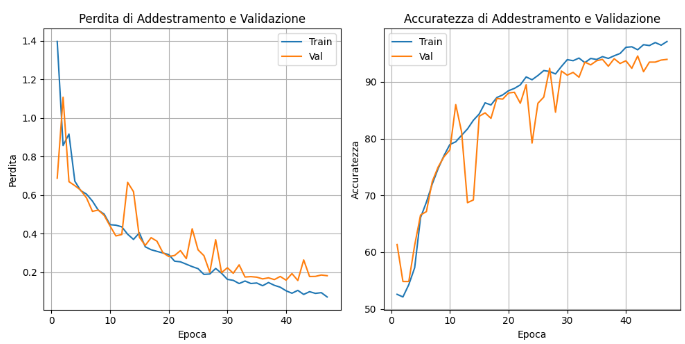

Fig. 7: Loss and accuracy over epochs

The model achieved an accuracy of 93.96% on the validation set, showing higher performance than the previous custom network.

E. Testing and Comparison Between Models

In this section, we will present the results of tests conducted on each model using appropriate metrics to evaluate classification performance. Various aspects will be examined, including accuracy, precision, confusion matrix, and ROC curve, in order to obtain a comprehensive view of the models' generalization capabilities and predictive effectiveness.

The three networks were tested using a test set that had not been previously used for training or validation, in order to evaluate the performance of the models on completely new data. The results obtained are reported in the following table:

Model	Accuracy (%)
Custom CNN	93.65
ResNet18	96.92
Deep CNN	94.38

TABLE I: Accuracy results of the models

Below are the confusion matrices of the models:

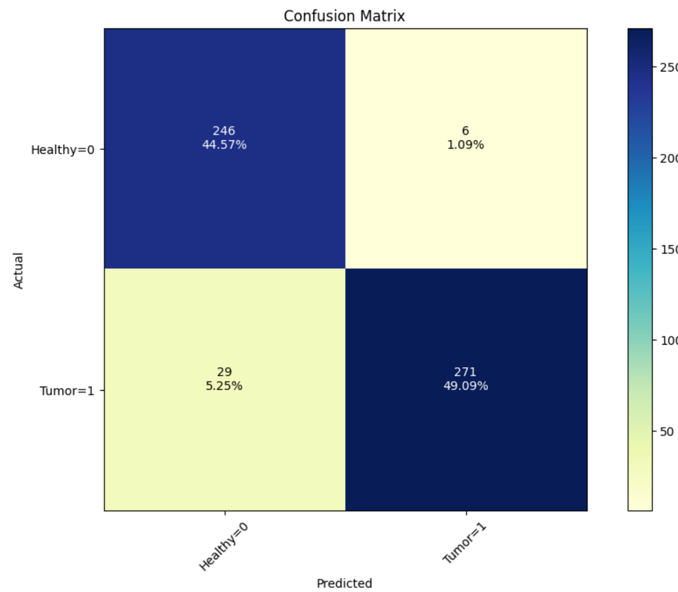

Fig. 8: Confusion Matrix for Custom CNN

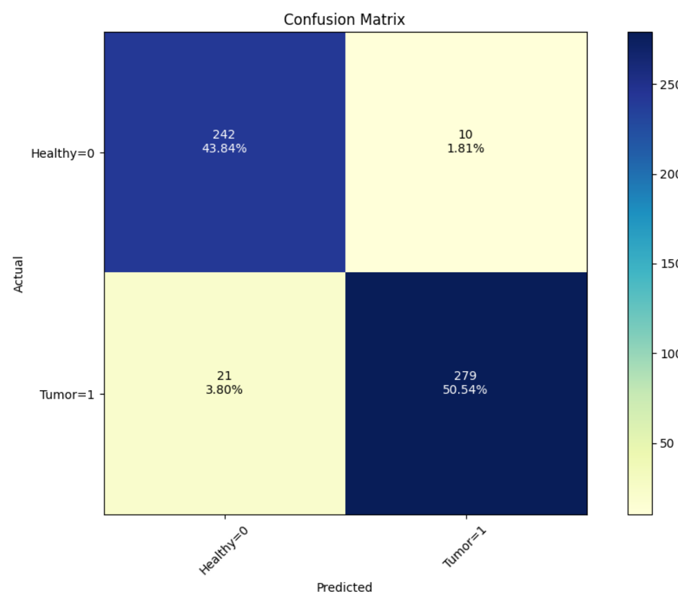

Fig. 9: Confusion Matrix for Deep CNN

The metrics between these two models are very similar, with an accuracy of 93.65% for the custom CNN and 94.38% for the Deep CNN, revealing that the network with a better ability to generalize and go in depth performs better than the custom network.

V. Conclusion

In this study, we have presented a comparison between three image classification models for predicting possible brain tumors, using deep learning techniques. The models were trained and validated on a dataset of brain radiographic images, with the aim of distinguishing between images with tumor presence and normal images. As we can observe through both metrics and graphical representations, the three models were compared in terms of accuracy, precision, confusion matrix, and ROC curve, in order to evaluate their performance and generalization capabilities.

In conclusion, we can say that the ResNet18 model achieved the best results, with an accuracy of 96.92

Below is the confusion matrix of the ResNet18 model, which achieved the best results:

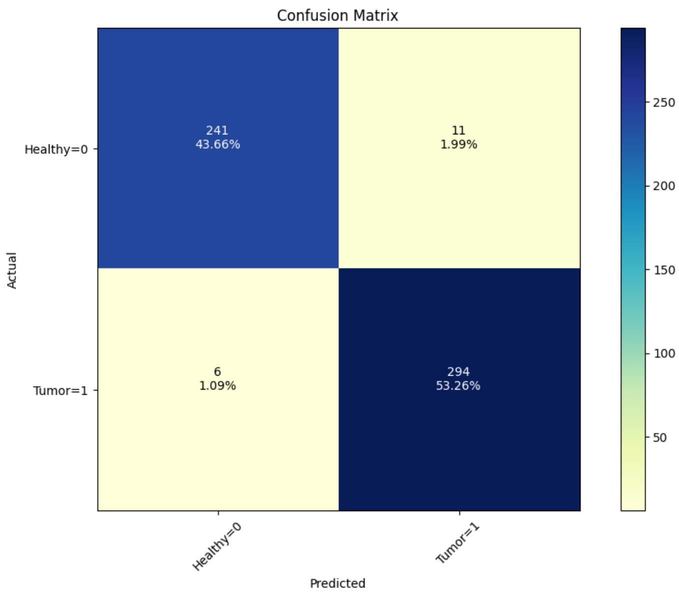

Fig. 10: ResNet18 Confusion Matrix

In the future, further deep learning and image analysis techniques could be explored to further improve model performance and develop more advanced solutions for brain tumor classification. The ultimate goal is to provide automated and accurate tools to support medical diagnosis and improve brain tumor management, contributing to saving human lives and improving patients' quality of life.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," pp. 770–778, 2016.

[2] T. Sadad, A. Rehman, A. Munir, T. Saba, U. Tariq, N. Ayesha, and R. Abbasi, "Brain tumor detection and multi-classification using advanced deep learning techniques," *Microscopy Research and Technique*, vol. 84, no. 6, pp. 1296–1308, 2021.

[3] P. Viradiya, "Brain tumor dataset," 2021.

[4] S. L. Fernandes, U. J. Tanik, V. Rajinikanth, and K. A. Karthik, "A reliable framework for accurate brain image examination and treatment planning based on early diagnosis support for clinicians," *Neural Computing and Applications*, vol. 32, pp. 15 897–15 908, 2020.

[5] M. S. . M. D. Somaya A. El-Feshawy Waleed Saad, "Iot framework for brain tumor detection based on optimized modified resnet 18 (omres)," *SpringerLink*, vol. 79, pp. 1081–1110, 2022. [Online]. Available: https://link.springer.com/article/10.1007/s11227-022-04678-y

[6] M. Sun, Z. Song, X. Jiang, J. Pan, and Y. Pang, "Learning pooling for convolutional neural network," *Neurocomputing*, vol. 224, pp. 96–104, February 2017.

[7] S. S. Basha, S. R. Dubey, V. Pulabaigari, and S. Mukherjee, "Impact of fully connected layers on performance of convolutional neural networks for image classification," *Neurocomputing*, vol. 378, pp. 112–119, 2020.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," *ImageNet: A Large-Scale Hierarchical Image Database*, vol. 11, no. 7, 2008. [Online]. Available: https://ieeexplore.ieee.org/document/5206848

[9] N. Bjorck, C. P. Gomes, B. Selman, and K. Q. Weinberger, "Understanding batch normalization," 2018.

[10] J. Chen and Y. Liu, "Fatigue modeling using neural networks: A comprehensive review," *Fatigue & Fracture of Engineering Materials & Structures*, vol. 45, no. 4, pp. 945–979, 2022.

[11] Y. Liu, Y. Zhou, S. Wen, and C. Tang, "A strategy on selecting performance metrics for classifier evaluation," *International Journal of Mobile Computing and Multimedia Communications (IJMCMC)*, vol. 6, no. 4, p. 16, 2014.