



**Faculdade de Ciências da Universidade de Lisboa**

**Master's in Data Science**

Advanced Machine Learning
(2022/2023)

Final Project Report

# "Brain Tumor Detection using Deep Learning and Boosting Algorithms"

Cláudia Afonso (36273), Rita Rodrigues (54859)

## I. Introduction

Brain tumors are a complex and heterogeneous group of neoplasms that arise from an abnormal growth of cells within the brain or its surrounding tissues. They represent a significant health concern worldwide, accounting for approximately 2% of all cancers and causing substantial morbidity and mortality. While early detection of brain tumors is crucial for effective treatment and improved patient outcomes, their detection can be challenging due to several factors. The heterogeneity of tumor subtypes presents a significant obstacle in the detection of brain neoplasms, as these exhibit varying imaging characteristics and clinical features. For instance, glioblastoma multiforme (GBM), the most common and aggressive type of malignant brain tumor, often presents as a heterogeneous mass with necrotic areas, whereas meningiomas, the most common type of benign brain tumors, typically appear as well-defined, contrast-enhancing lesions.[1]

Another important challenge concerns the limited sensitivity and specificity of the conventional imaging modalities, such as magnetic resonance imaging (MRI) and computed tomography (CT) scans. Although widely used in clinical practice, these imaging techniques can be expensive, time-consuming and may not always provide a definitive diagnosis. In addition, such traditional imaging methods may not be sensitive enough to detect early-stage, small tumors or may produce false positives.[2] Thus, accurately distinguishing between tumor subtypes can be crucial for determining the most appropriate treatment and prognosis, while also avoiding unnecessary invasive procedures.

To address these challenges, deep learning has emerged as a powerful approach in the field of medical imaging, particularly for image classification in the context of brain tumors. The main aim of this project consists in developing a brain tumor detection system using deep learning approaches. The latter can efficiently extract the intricate details from brain scans, leading to improved accuracy in classifying distinct types of brain tumors. In addition, automating the detection process using deep learning can reduce the burden on radiologists and healthcare professionals, enabling faster diagnosis and streamlined treatment planning, thus allowing for more timely interventions and potentially better prognosis for patients.[3]

## II. Approach

In this work, our goal is to replicate and extend the brain tumor detection approach proposed in the scientific paper “MRI-based brain tumor detection using convolutional deep learning methods and chosen machine learning techniques”.[4] Specifically, using the same dataset, we will reproduce the input image preprocessing and some classification steps outlined in the paper, while complementing the published strategy with solutions of our own.

The approach presented in this project encompasses several key steps. We began by downloading from Kaggle the same diverse and representative dataset of MRI scans as the one used in the referenced paper, which includes various imaging planes and distinct tumor subtypes.[5] Next, the images were resized to ensure consistent input sizes across all MRI scans. We also employed data augmentation methods to increase the size of the dataset and improve the generalization of our machine learning models. For the core classification task, the same convolutional neural network (CNN) architecture as proposed in the referenced paper was adopted. This CNN architecture was specifically designed and tailored for the analysis of medical images, making it highly suitable for accurate brain tumor detection. In this way, by using the exact same CNN architecture as described in the paper, it is possible to fully replicate and validate the published approach while building upon it with additional techniques.

To extract meaningful and compact representations of the input images, we incorporated the proposed autoencoder architecture of the referenced paper.[4] This autoencoder was trained to learn efficient latent representations of the MRI scans, capturing meaningful and informative features related to particular types of brain tumors. To leverage the power of transfer learning and thus benefit from pre-existing knowledge, pre-trained CNN models such as VGG16 and ResNet50 were employed as feature extractors. These models have been previously trained on vast datasets, enabling them to learn intricate and discriminative features from more general image data. By using transfer learning, it is possible to harness the learned representations of these models and fine-tune them on our dataset to adapt them to the specific classification task at hand. Fine-tuning involved adjusting the parameters of the model to align with the nuances and characteristics of our dataset, thereby enhancing its performance and capability for brain tumor detection.

Ensemble learning techniques, specifically Adaboost, Gradient Boosting and XGBoost algorithms, were also explored in this project. Ensemble learning is a machine learning technique that combines several individual models, typically termed weak learners, to create a more accurate classifier. In addition to the aforementioned techniques, a novel approach known as Convolutional XGBoost was also employed. This method combines the convolutional neural network's feature extraction capabilities with the XGBoost algorithm, creating a powerful fusion model with improved prediction performance for brain tumor detection.[6]

Following training of the models on the dataset, a comprehensive analysis of the results was performed. The performance of each model was assessed using various evaluation metrics such as accuracy, precision, recall and F1 score.

## III. Implementation

The implementation of the brain tumor detection system presented in this project was carried out using the Python programming language, with popular frameworks such as TensorFlow and Keras used to build and train deep learning

models. Google Colab, a cloud-based development environment for collaborative work in Python with free access to GPU resources, was used to accelerate model training and evaluation.

Dataset

The brain tumor dataset used in this study was obtained from Kaggle and comprised 3264 T1-weighted contrast-enhanced MRI images of four different types: glioma, meningioma, pituitary gland tumor and healthy brain (Figure 1).[5] The images were obtained in sagittal, axial and coronal planes with varying sizes.

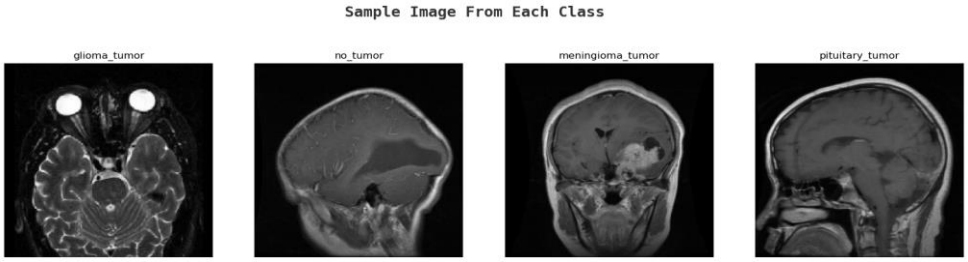



Figure 1 – Sample images for each of the four possible classes present in the chosen dataset.

Image pre-processing and data augmentation

To prepare the dataset for training, the images were resized to 80 by 80 pixels, as recommended by the authors of the referenced paper.[4] The dataset was also augmented by performing two types of image transformation: a 90-degree rotation and a vertical flip. This data augmentation technique has been shown to improve the performance of deep learning models for medical image analysis.[7] The dataset was augmented three times, resulting in a total of 9792 images. To train and evaluate the deep learning models, the brain tumor dataset was divided into a training set and a validation set. The training set consisted of 8812 images corresponding to 90% of the dataset, while the validation set contained 980 images corresponding to a total of 10% of the dataset.

CNN

The CNN architecture described in the paper was replicated in this project.[4] A diagram of the proposed architecture is exemplified in Scheme 1 of the Annex. The proposed CNN architecture for brain tumor detection consists of multiple convolutional layers, each applying a set of filters to extract features from the input data. The specific configuration includes two convolutional layers with 64 filters, followed by two convolutional layers with 32 filters and additional layers with 16 filters. The final two convolutional layers create network filters with a length of 8. The network layers use a 2x2 kernel function for convolution. The proposed CNN incorporates a hierarchical structure, connecting convolutional layers, pooling layers, and fully connected layers. It should be noted that not every convolutional layer requires a pooling layer afterwards. In this architecture, there are 8 convolutional layers and four pooling layers. The final pooling layer with a 2D output is flattened to a 1D layer, allowing it to be passed to the fully connected layers. To manage the output size of the convolutional layers and preserve the edges of the input data, padding is applied using adjacent cells with the same values. Batch normalization layers are used to prevent overfitting and a dropout layer with a rate of 0.1 is applied after max-pooling and fully connected layers.

The rectified linear unit (ReLU) activation function is used in all layers, with the exception of the last fully connected layer. ReLU introduces non-linearity to the model, enhancing the ability of CNN to learn complex and hierarchical representations from the input data. The “Adam” optimization algorithm is chosen as the optimizing function due to its effectiveness in handling large-scale datasets. An initial learning rate of 0.001 is selected through experimentation, providing a good balance between speed of convergence and accuracy.

The CNN is trained for 100 epochs with a batch size of 16. The latter determines the number of samples processed before updating the model’s parameters. A smaller batch size allows for more frequent updates, aiding convergence and preventing memory issues. The classification part of the CNN involves fully connected layers. The network includes a fully connected layer with 1024 neurons, followed by a final fully connected layer with 4 neurons representing the four types of image categories in the dataset. The softmax activation function is applied to the last layer for multi-class classification, thereby producing class probabilities. Overall, this CNN architecture and training configuration are designed to effectively extract features from brain tumor images and classify them into the appropriate categories. The choice of layers, activation functions, optimization algorithm, batch size and learning rate are based on empirical evaluations and best practices in deep learning.

The plot displaying the training and validation accuracies across epochs provides valuable insights into the performance of the CNN model (Figure 2A). As the training progresses, the accuracies steadily increase, indicating that the model is learning how to effectively classify the dataset images. Even though the values for the validation accuracy show occasional fluctuations or peaks, the overall trend is consistent with the training accuracy. This suggests that the model is not overfitting and is able to generalize well to unseen data, despite the intermittent challenges posed by certain

validation examples. The convergence of the training and validation accuracy curves demonstrates the effectiveness of this approach in accurately classifying brain tumors while maintaining generalization capabilities.

Similarly, the plot displaying the training and validation loss values across epochs reveals important information regarding the model's training process (Figure 2B). The training loss consistently decreased over time, indicating that the model successfully minimized learning errors and improved its performance on the training data. The validation loss, although exhibiting occasional peaks or fluctuations, exhibited an overall decreasing trend, indicating the model's ability to generalize well to unseen data. The alignment of the training and validation loss curves suggests that the CNN model managed to capture relevant features and generalize its learnings to new dataset images.

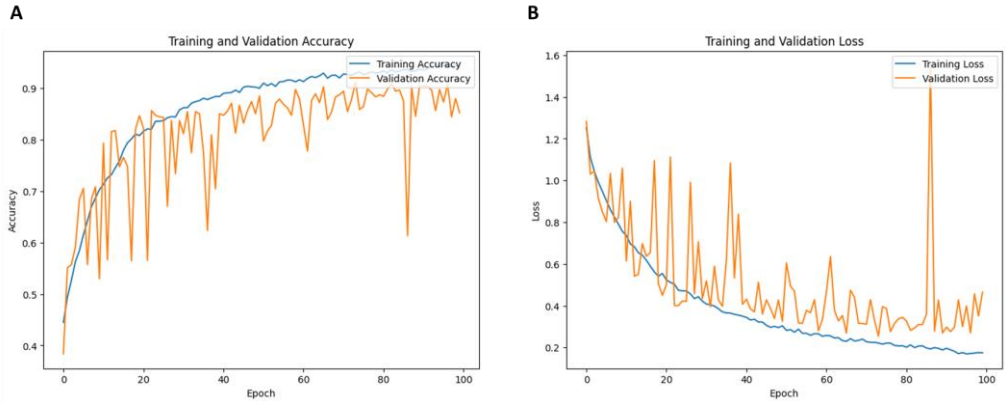



Figure 2 – Training and validation accuracies obtained across the 100 epochs of the proposed CNN.

Auto-encoder

We also replicated the architecture of a convolutional auto-encoder network as described in the referenced paper.[4] A diagram of the proposed architecture is exemplified in Scheme 2 of the Annex. This network is designed to predict the input images (X) rather than the target value (Y), using an auto-encoder for training and classification purposes. The overall architecture consists of two main parts: the convolutional auto-encoder network for data training and a separate convolutional network for classification, using the output encoder layer from the first part. The encoder part of the network included multiple convolutional layers with different filter lengths, while no pooling layers were used after each convolutional layer. However, a second stage of 2x2 max-pooling was applied after a sequence of two convolutional layers. The decoder part of the network mirrors the encoder architecture, but with the inclusion of up-sampling layers after a sequence of two convolutional layers. Batch normalization layers were incorporated after each convolutional layer and a dropout of 0.1 was applied after each max-pooling layer to prevent overfitting.

For accurate classification, the output of the encoder layer was further trained by additional convolutional layers, and the flattened layer was used to forward the output to a fully connected layer. The ReLU activation function was applied to all layers and batch normalization and dropout layers were employed to mitigate overfitting. A learning rate of 0.001 was chosen for the auto-encoder network, which was trained for 100 epochs with a batch size of 16.

Transfer Learning

Transfer learning is a deep learning technique that leverages the knowledge gained from training a model on a particular task to improve its performance on another related task.[8] It involves using a pre-trained model, which has been trained on a large dataset, as a starting point for a new task. The pre-trained model learns general features from the initial dataset, and these learned features can be transformed and fine-tuned for a different task with a smaller dataset. By using transfer learning, it is possible to benefit from the pre-trained model's knowledge and thus significantly reduce the training time and data requirements for the newer task.

Transfer learning has been widely applied in various domains, including computer vision tasks such as object recognition and image classification. In the context of brain tumor detection, transfer learning has shown promising results.[9] The pre-trained models, such as VGG16 [10] and ResNet50 [11] which have been trained on large-scale image datasets, can be adapted for brain tumor detection by fine-tuning the model on a specific brain tumor dataset. In the case of VGG16, the pre-trained model is initialized with the weights learned from the ImageNet dataset. The fully connected layers are replaced with new layers that are specific to the brain tumor classification task. The layers of the VGG16 model are set to be non-trainable, except for the newly added layers. This allows the model to extract relevant features from the brain tumor images while retaining the general knowledge learned from ImageNet. The adapted VGG16 model is then compiled with an "Adam" optimizer and categorical cross-entropy loss, suitable for multi-class classification. The training process involved 20 epochs.

Similarly, with ResNet50, the pre-trained model is initialized with weights learned from ImageNet. The layers of the ResNet50 model are frozen, preventing them from being trained. New classification layers are added on top of the ResNet50 model, allowing it to learn and classify brain tumor images. The added layers include a global average pooling layer, a fully connected layer with ReLU activation, a dropout layer to prevent overfitting, and a final softmax activation layer for multi-class classification. The adapted ResNet50 model is then compiled with an "Adam" optimizer and

categorical cross-entropy loss, suitable for multi-class classification. The training process involved 20 epochs. By employing transfer learning with VGG16 [10] and ResNet50 [11], we can benefit from their deep learning capabilities and their ability to extract meaningful features from images. These pre-trained models provide a solid foundation for brain tumor detection, reducing the need for large amounts of labeled data and significantly accelerating the training process.

Boosting

Boosting is a machine learning technique that combines multiple weak learners to create a strong learner. It is particularly useful when dealing with complex and challenging classification tasks. Boosting algorithms sequentially train a series of weak models, with each subsequent model focusing on correcting the mistakes made by the previous models. This iterative process leads to the creation of a powerful ensemble model capable of making accurate predictions. In the context of brain tumor detection, boosting algorithms can effectively handle the intricacies and variations present in brain tumor images, leading to improved classification performance.

One widely used boosting algorithm is AdaBoost. It works by iteratively training weak classifiers on different subsets of the data, assigning higher weights to misclassified samples at each iteration. This allows subsequent weak classifiers to focus more on the previously misclassified samples, improving overall classification accuracy. In our experiments, we employed the AdaBoostClassifier with 3 estimators. Another popular boosting algorithm is Gradient Boosting. Unlike AdaBoost, Gradient Boosting optimizes the loss function by iteratively fitting weak models to the negative gradient of the loss. This technique allows subsequent models to learn from the mistakes of previous models, gradually reducing the loss and improving predictive accuracy. We utilized the GradientBoostingClassifier with 3 estimators in our evaluation. Additionally, we also tested the XGBoost algorithm, an optimized implementation of gradient boosting. XGBoost combines the gradient boosting framework with additional enhancements such as parallelization, tree pruning, and regularization, making it highly efficient and effective. The XGBoost algorithm is known for its ability to handle large datasets and deliver excellent performance. In our experiments, we utilized the XGBClassifier with 3 estimators, a learning rate of 0.1, and a maximum depth of 5.

By leveraging these boosting algorithms, including AdaBoost, Gradient Boosting, and XGBoost, we aimed to improve the accuracy of brain tumor detection. These algorithms have shown great potential in various machine learning applications and are well-suited for handling the complexities of brain tumor classification tasks. In addition to the methods described above, we also explored the application of Convolutional XGBoost (C-XGBoost).[6] Convolutional XGBoost (C-XGBoost) is an innovative approach that combines the strengths of Convolutional Neural Networks (CNNs) and the XGBoost algorithm to enhance the accuracy and efficiency of image analysis tasks, particularly in brain tumor detection. C-XGBoost is an extension of the XGBoost algorithm, which is known for its effectiveness in handling structured data for regression, classification, and ranking tasks. By incorporating CNN layers into XGBoost, C-XGBoost is capable of handling grid-like structured data, such as images, and extracting complex patterns and features from the data.

The unique aspect of C-XGBoost lies in its integration of both CNNs and XGBoost. CNNs consist of multiple layers of convolutional, pooling, and fully connected layers that extract features and learn complex relationships within the data. Convolutional layers apply filters to the input data, pooling layers reduce data dimensionality, and fully connected layers facilitate learning of intricate feature relationships. On the other hand, XGBoost is employed to learn a decision tree model from the input data. The CNN layers are used to extract features from the data and feed them into the XGBoost model. In practice, the C-XGBoost model involves training the CNN part separately on brain tumor images to extract features. The last layer preceding the dense classification layer, in our case a dense layer with 1024 neurons, is used to extract the features. These features are then fed into the XGBoost framework, where the XGBoost algorithm minimizes the loss function and improves the model's accuracy. In our experiments, we used the XGBoost classifier with the objective function set to multiclass:softmax, a learning rate of 0.1, a maximum depth of 15 and a number of estimators equal to 500. By leveraging the power of both CNNs and XGBoost, C-XGBoost reduces model complexity, prevents overfitting, and improves training efficiency. This approach shows promising potential in enhancing image analysis and disease prediction tasks, such as brain tumor detection.

## IV. Results

To evaluate the performance of the proposed models for brain tumor detection, we compared the accuracy achieved by the different models. Figure 5 presents a bar plot illustrating the accuracy scores obtained by each model, providing a comprehensive overview of their relative performance. To establish a threshold for acceptable accuracy in brain tumor detection, a criterion of 90% accuracy was defined as the minimum performance requirement. This threshold was chosen based on the need for reliable and accurate classification of brain tumor images to ensure effective decision-making and minimize potential misdiagnosis or false negatives. Achieving an accuracy above this threshold would indicate a robust and reliable model performance, providing a high level of confidence in its ability to accurately classify brain tumor images.

Upon observing the bar plot comparing the accuracies of the different models (Figure 5), it becomes evident that the convolutional XGBoost and autoencoder models outperform the other ones, surpassing the defined threshold of 90% accuracy. Notably, Convolutional XGBoost demonstrates the highest accuracy among all models tested. The superior performance of convolutional XGBoost can be attributed to its ability to leverage the strengths of both

convolutional neural networks (CNNs) and gradient boosting algorithms. By incorporating CNN layers into the XGBoost framework, convolutional XGBoost can effectively capture intricate image patterns and learn discriminative features from the brain tumor images. This combined power enables Convolutional XGBoost to achieve higher accuracy scores and enhances its potential for accurate brain tumor classification. In contrast, the boosting models (AdaBoost, Gradient Boosting, XGBoost) exhibit lower accuracies, falling below the acceptable threshold. Given the critical nature of medical decision-making in brain tumor detection, these accuracy levels are deemed insufficient for reliable and precise classification. The boosting models' limitations in capturing complex image features and potential overreliance on weak classifiers might have contributed to their relatively poorer performance.

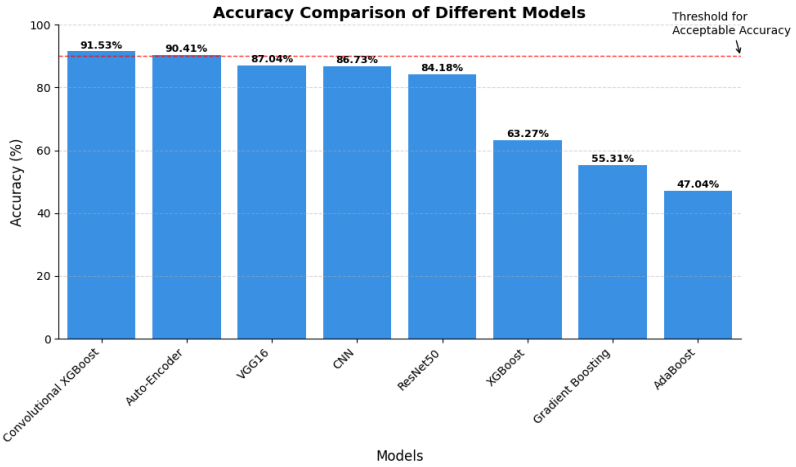



Figure 3 – Comparison of accuracy scores obtained for the different models tested in this project for the task of brain tumor classification in images.

After comparing the testing accuracy of our implemented models with the report results in the referenced paper, some differences can be observed. The CNN model in the paper achieved a testing accuracy of 93.45%, while our implementation achieved an accuracy of 86.73%. Similarly, the autoencoder model in the paper achieved a testing accuracy of 90.93%, whereas our implementation achieved an accuracy of 90.41%. These variations in accuracy can be attributed to several factors, including differences in the dataset, model architecture, hyperparameters, and training methodology. It is important to note that replicating the exact results reported in the paper may not always be possible due to these variations. Each implementation may involve different data preprocessing techniques, model configurations, or optimization strategies, which can lead to slight performance differences.

However, despite slight accuracy deviations, our implementation still demonstrates competitive performance. The CNN achieved an accuracy of 86.73%, and the autoencoder achieved 90.41%, indicating their effectiveness in accurately classifying brain tumor images. While we cannot directly validate the results of the paper due to the aforementioned differences, our results validate the potential of the CNN and autoencoder models in brain tumor detection. Based on the obtained results, the convolutional XGBoost model emerged as the top performer in our study, demonstrating the highest accuracy among all the tested models. Thus, we identify the convolutional XGBoost model as the best-performing model for brain tumor detection.

To gain further insights into the model's performance, we examined the classification report (Table 1) and generated a confusion matrix (Figure 4). The classification report provides detailed metrics such as precision, recall, and F1-score for each class, offering a comprehensive evaluation of the model's predictive capabilities. Additionally, the confusion matrix visually represents the distribution of predicted and actual class labels, enabling the assessment of the model's performance across different classes.

Table 1 – Values of the evaluation metrics obtained for the convolutional XGBoost model

| **Categories** | **Precision** | **Recall** | **F1-score** | **Support** |
|---|---|---|---|---|
| “No tumor” | 0.89 | 0.97 | 0.92 | 159 |
| “Glioma tumor” | 0.93 | 0.87 | 0.90 | 265 |
| “Meningioma tumor” | 0.90 | 0.90 | 0.90 | 286 |
| “Pituitary tumor” | 0.94 | 0.95 | 0.94 | 270 |
| | | | | |
| Accuracy | | | 0.92 | 980 |
| Macro average | 0.91 | 0.92 | 0.92 | 980 |
| Weighted average | 0.92 | 0.92 | 0.92 | 980 |

For the “No tumor” category (label 0), the model achieved a precision of 0.89, indicating that 89% of the predicted cases for this category were correct. The recall, which measures the percentage of actual positive cases correctly

identified, is 0.97. This indicates that the model accurately detected 97% of the actual cases for the “No tumor” category. The F1-score, which considers both precision and recall, is 0.92, reflecting a balanced performance for this category. For the “Glioma tumor” category (label 1), the model exhibited a precision of 0.93, indicating a high level of accuracy in predicting this category. The recall is 0.87, meaning that the model successfully identified 87% of the actual cases for the “Glioma tumor” category. The F1-score is 0.90, indicating a good balance between precision and recall. The “Meningioma tumor” category (label 2) demonstrated a precision of 0.90, indicating a high level of accuracy in predicting this category. The recall is also 0.90, indicating that the model effectively identified 90% of the actual cases for this category. The F1-score is 0.90, reflecting a well-balanced performance for this category. Finally, for the “Pituitary tumor” category (label 3), the model achieved a precision of 0.94, indicating a high level of accuracy in predicting this category. The recall is 0.95, indicating that the model successfully identified 95% of the actual cases for category 3. The F1-score is 0.94, reflecting a strong performance for this category.

Overall, the model achieved an accuracy of 0.92, indicating that it correctly classified 92% of the images in the testing dataset. The macro averages of precision, recall, and F1-score indicate a consistent performance across the different categories. The weighted average, which accounts for class imbalance, is 0.92, reflecting an overall balanced performance of the model across all categories.

The higher values and darker colors along the diagonal of the confusion matrix (Figure 4) suggest that the model performed well in correctly classifying brain tumor cases across all categories. This indicates that the model's predictions aligned with the true labels for a significant number of cases. However, some off-diagonal elements are also visible, which indicates some misclassifications. These cells represent instances where the model predicted a different category than the true label. The intensity of the color in these cells reflects the magnitude of misclassifications. Overall, the diagonal with higher values and darker colors in the confusion matrix suggests that the convolutional XGBoost model successfully differentiated between different brain tumor categories, demonstrating a robust performance in accurately classifying the tumors.

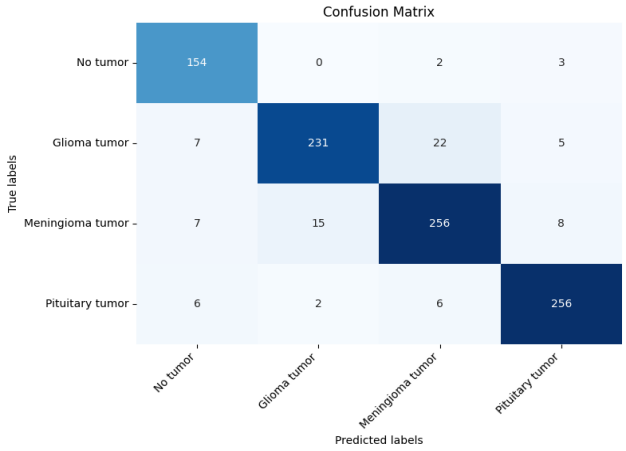



Figure 4 – Confusion matrix showing the counts of true positive, true negative, false positive, and false negative predictions obtained for the convolutional XGBoost model.

## V. Final comments

Our project on brain tumor detection used different deep learning and ensemble techniques. The combination of convolutional neural networks with the XGBoost algorithm demonstrated promising results in accurately classifying brain tumor images. The convolutional XGBoost model outperformed other models in terms of accuracy, making it a robust tool for brain tumor detection. Accurate and efficient diagnosis of brain tumors is crucial for timely treatment and favorable patient outcomes. The model’s high accuracy and performance have the potential to enhance the diagnostic process, aiding medical professionals in making informed decisions. By automating the classification of brain tumor images, it is possible to expedite the diagnosis process, potentially leading to early detection and improved patient care.

Throughout this project, we have gained valuable insights and learned several important lessons. The integration of convolutional neural networks and the XGBoost algorithm proved to be a powerful approach for handling complex image data. We have also realized the importance of adequate data preprocessing and feature extraction techniques to optimize model performance. While our project has demonstrated promising outcomes, it also has certain limitations. One of these pertains to the availability and size of the dataset. Incorporating a larger and more diverse dataset would enhance the generalizability of our model. Additionally, to ensure the reliability and validity of the models, it would be crucial to validate them using another dataset. Furthermore, exploring alternative architectures, such as deeper CNN models, could potentially improve the accuracy and robustness of the system.

In conclusion, our project showcases the usefulness and potential of convolutional XGBoost in brain tumor detection. By addressing the limitations and considering possible improvements or extensions, such as incorporating additional data sources and exploring alternative architectures, it is possible to further advance the accuracy and applicability of brain tumor detection, ultimately benefiting medical professionals and patients alike.

## VI. Bibliography


1. Ostrom QT, Gittleman H, Truitt G, Boscia A, Kruchko C, Barnholtz-Sloan JS. CBTRUS Statistical Report: Primary Brain and Other Central Nervous System Tumors Diagnosed in the United States in 2011-2015. Neuro Oncol. 2018;20(suppl_4):iv1-iv86.

2. Busby LP, Courtier JL, Glastonbury CM. Bias in Radiology: The How and Why of Misses and Misinterpretations. Radiographics. 2018;38(1):236-47.

3. Arabahmadi M, Farahbakhsh R, Rezazadeh J. Deep Learning for Smart Healthcare-A Survey on Brain Tumor Detection from Medical Imaging. Sensors (Basel). 2022;22(5).

4. Saeedi S, Rezayi S, Keshavarz H, S RNK. MRI-based brain tumor detection using convolutional deep learning methods and chosen machine learning techniques. BMC Med Inform Decis Mak. 2023;23(1):16.

5. Bhuvaji SK, A.; Bhumkar, P.; Dedge, S.; Kanchan, S. Brain Tumor Classification (MRI) - Kaggle 2020 [Available from: https://www.kaggle.com/datasets/sartajbhuvaji/brain-tumor-classification-mri.

6. Muyiwa Babayomi OAO, Abdulrasheed Adedolapo Kadiri. Convolutional XGBoost (C-XGBOOST) Model for Brain Tumor Detection. arXiv preprint arXiv:230102317. 2023.

7. Safdar MF, Alkobaisi SS, Zahra FT. A Comparative Analysis of Data Augmentation Approaches for Magnetic Resonance Imaging (MRI) Scan Images of Brain Tumor. Acta Inform Med. 2020;28(1):29-36.

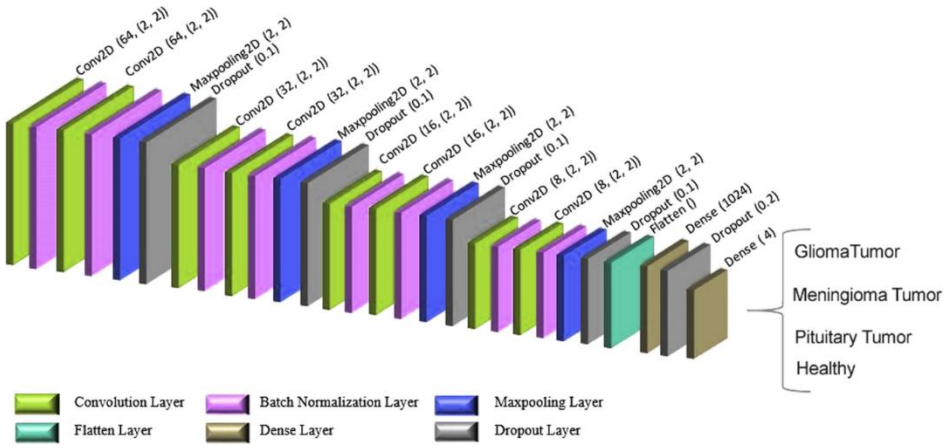
8. Pan SJ, Yang Q. A Survey on Transfer Learning. IEEE Transactions on Knowledge and Data Engineering. 2010;22(10):1345-59.

9. Shin HC, Roth HR, Gao M, Lu L, Xu Z, Nogues I, et al. Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning. IEEE Trans Med Imaging. 2016;35(5):1285-98.

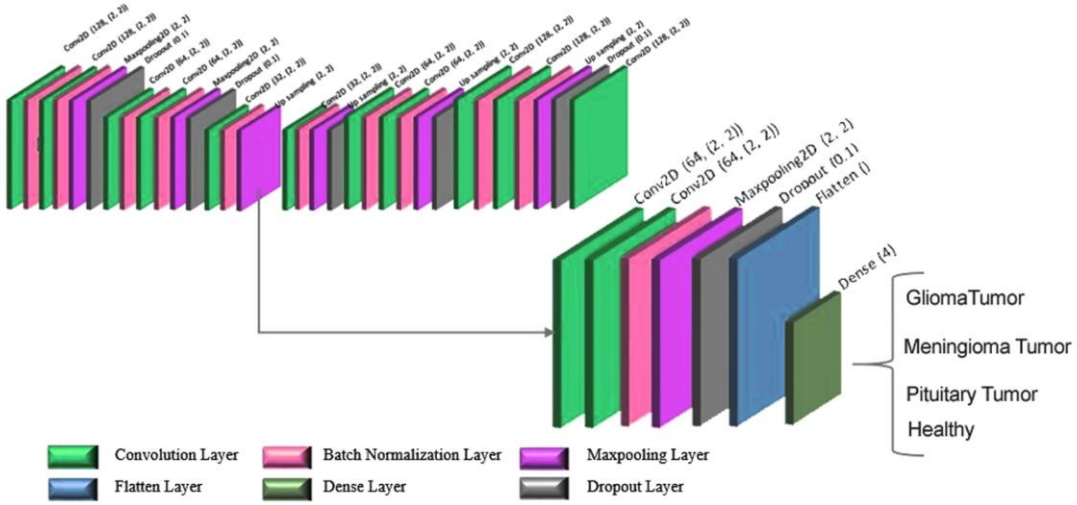
10. Simonyan KZ, A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv. 2014;1409.1556.

11. He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. In: IEEE, editor. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)2016. p. 770-8.


## VII. Annex




Scheme 1 – Architecture of the CNN proposed in the paper and replicated in this project. Adapted from the referenced paper.[4]




Scheme 2 – Architecture of the convolutional auto-encoder network classification part. Adapted from the referenced paper.[4]