Regional Brain Tumor Detection Web App

Nicole Lowenstein[301433773], Vedant Ashish Jain[301438620], Sahba Hajihoseini[301433997], Japneet Sason[301392978], and Risa Kawagoe[301417365]

{nla70,vaj1,sha253,jsason,rkawagoe}@sfu.ca

Abstract. In this project, we developed a machine learning model that can detect tumors on brain MRI images. Additionally, the model can predict the shape, size, and location of the tumor. We believe that this medical analysis and detection tool is a great resource for doctors and radiologists to make diagnoses more precise and obtain information of the disease that we could not otherwise know. This has the potential to save many lives, as it is easier to take a course of action with more knowledge of the disease. We used a U-NET convolutional neural network with 28 layers for our model and we trained it with 368 files divided into training, test, and validation sets with a 80:10:10 split. We achieved a model that is 99.48% accurate. We also have a 1.64% loss and a 59.87% dice loss.

Keywords: Convolutional Neural Network · Brain Tumors · MRI · Computer-Assisted Diagnosis

1 Introduction

Brain tumors pose a significant threat to public health, often requiring early and precise diagnosis for effective treatment. Our project aims to address this critical need by creating an advanced neural network model that can be used to identify and analyze brain tumors from magnetic resonance imaging (MRI) scans. The goal of this project is to equip medical practitioners with a tool that can not only detect the existence of tumors but also provide important details about their location, size, and shape. This could completely revolutionize the diagnostic procedure. Having detailed and precise data can enable medical professionals to make better decisions, thereby improving treatment programs' effectiveness and potentially saving lives.

The combination of medical imaging and machine learning has seen amazing breakthroughs in recent years, altering how brain cancers are detected and diagnosed. Traditional diagnostic procedures are being reshaped by modern techniques, with neural network-based approaches becoming more popular due to their potential to improve efficiency and accuracy. Our project primarily focuses on exploring the relationship between medical imaging and machine learning in light of these breakthroughs. In order to take advantage of this combination, we have implemented the U-NET architecture and are using convolutional neural networks (CNNs) to create a brain tumor detection model that is dependable.

The U-NET architecture's capacity to recognize complicated spatial relationships in images is what allows it to adapt to the complexities of brain MRI scans.

1.1 Literary Survey

The potential of neural networks to enhance current diagnostic methods has made brain tumor detection via neural networks a popular research topic. Convolutional Neural Networks (CNNs) have been very helpful as they provide a more automated and efficient method of feature extraction than time-consuming, manual techniques. Many researchers have presented various CNN architectures, the majority of which are able to classify brain tumors into various categories in addition to detecting them. These models have been thoroughly tested on large datasets for comprehensive training and testing.

Sultan et al., for example, proposed a CNN model with 16 layers that produced prediction accuracies of 96.1% and 98.7% on datasets involving 3064 and 516 pictures, respectively [1]. Using the Fuzzy C-Means clustering technique, Hossain et al. achieved an impressive 97.9% prediction accuracy, surpassing previous models [2]. Ertosun et al. developed a hybrid CNN model for multiclass glioma tumors using a unique approach. For Grade II, Grade III, and Grade IV glioma tumors, they achieved classification accuracies of 96.0%, 71.0%, and 71.0%, respectively [3]. Using CNN and genetic algorithms, Anaraki et al. detected glioma tumors with 90.9% prediction accuracy, achieving 94.2% prediction accuracy for the diagnosis of pituitary, meningioma, and glioma tumors [4].

The study titled "Accurate brain tumor detection using deep convolutional neural network," by Md. Saikat Islam Khan et al., makes a substantial contribution to this field [5]. Their application of a 23-layer CNN and transfer learning with the VGG16 architecture shows promising results, with classification accuracy for tested datasets reaching up to 97.8% and 100%. Our project has been inspired by this research, which addresses issues with low amounts of data and shows how pre-trained models can improve diagnostic performance.

Despite their success in categorizing huge picture datasets, CNNs face difficulties, particularly in the medical imaging area. CNN models frequently need huge quantities of data for training, which can be challenging to obtain. In addition, they face difficulty in identifying photos that have minimal changes, including rotations or tilts. Researchers are using data augmentation approaches more often to overcome these issues, adding new variants on a regular basis during training. We used methods to improve the strength of our neural network model as we understand how important it is to overcome limitations in our research.

1.2 Roadmap

We organized our report into several key sections to provide a structured review of our project.

1. **Materials:** Description of data, signals, images, etc.

2. **Methods:** Outline of the proposed methodology, algorithmic details, and figures that make up our neural network model's fundamental building blocks.
3. **Results:** We provide figures that highlight our accomplishments and demonstrate the accuracy of our brain tumor detection model. We present both qualitative and quantitative data.
4. **Accomplishments:** An overview of our project's lessons learned and successes, including the challenges we overcame.
5. **Contributions:** We provide a clear list of each member's contributions towards the completion of the project.
6. **Conclusions and discussions:** After summarizing our achievements, we review and discuss the project as a whole, offering insights into our results.
7. **Future work:** We outline potential directions for future research, suggesting areas for improvement for those who may continue this work.
8. **Acknowledgements:** Acknowledging the sources who assisted with our project by contributing ideas, information, code, and support.
9. **Appendix:** A list of supplementary material and information that would otherwise interrupt the flow of the reading.
10. **References:** A list of all the references we used for our project, including articles, websites, and additional sources.

2 Materials

In our project, we utilized the BraTS 2020 dataset [6–10] for brain tumor segmentation using a convolutional neural network (CNN). This dataset is a rich collection of multimodal MRI scans, focusing on glioblastoma (GBM/HGG) and lower-grade glioma (LGG), sourced from multiple institutions. These scans are accompanied by pathologically confirmed diagnoses [6].

BraTS [11] is a yearly-held brain tumor segmentation challenge that aims to evaluate different cutting edge methods for brain tumor segmentation in mulimodal (MRI) scans. Specifically, the 2020 edition, held virtually in Lima, Peru, focused on challenges involving the segmentation of heterogeneous brain tumors (gliomas) and patient survival predictions [7]. We chose to work with this dataset because it was readily available, unlike the more recent BraTS 2023 dataset which required us to ask for access and it was never granted, and because our project task was similar to what the dataset was originally used for.

The BraTS 2020 dataset includes enhanced 3T multimodal MRI scans. Our study specifically employed the NIfTI format MRI scans from the BraTS 2020 dataset. These scans cover a range of modalities including native (T1), post-contrast T1-weighted (T1Gd), T2-weighted (T2), and T2 Fluid Attenuated Inversion Recovery (T2-FLAIR), offering a comprehensive view necessary for accurate segmentation. The dataset, encompassing scans from 19 diverse institutions, reflects variations in clinical protocols and scanning equipment, adding to the robustness of our study.

For our brain tumor segmentation project, we utilized the main training dataset from BraTS 2020, which comprised MRI scans from 368 patients. To

tailor this dataset for a comprehensive evaluation of our convolutional neural network (CNN) model, we devised a strategic partitioning scheme. This involved dividing the main dataset into three distinct subsets(Figure 1): a primary training set, a validation set, and a testing set.

Out of the total 368 patient images, we allocated 294 for the primary training set. This substantial portion of the dataset was used for the initial training and tuning of our CNN model. The remaining images were equally divided between the validation and testing sets, with 37 patient images allocated to each. This distribution allowed for a balanced and effective evaluation of the model, ensuring robust validation and rigorous testing of the model's performance and its generalization capabilities. This meticulous approach in dividing the dataset was crucial for achieving a thorough and nuanced understanding of the model's efficacy in segmenting brain tumors.

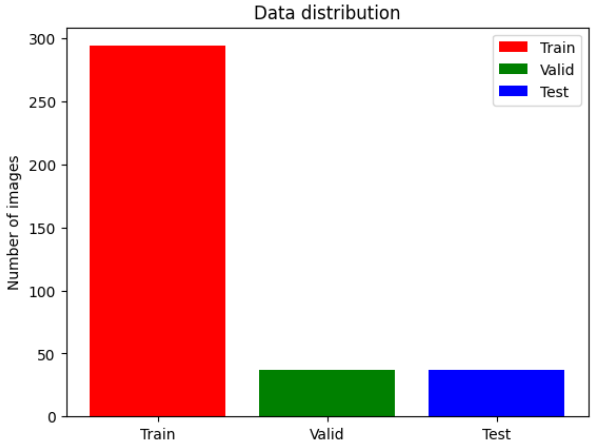

Fig. 1. Distribution for the data set

3 Methods

3.1 Proposed Method

Our project is centered around the U-Net architecture, a convolutional neural network optimized for complex structures in medical images, particularly effective in brain tumor segmentation tasks. U-Net's unique architecture, which features a contracting path to capture context and a symmetric expanding path for precise localization, makes it exceptionally well-suited for medical image analysis. This structure enables the model to effectively capture both high-level semantic and low-level spatial information, crucial for accurate segmentation in MRI scans.

3.2 Algorithmic Details

We constructed our model using the Keras library, known for its user-friendly and modular approach. Our implementation of the U-Net architecture involves multiple convolutional (Conv2D) layers, where each layer applies a set of learned filters to extract features from the images. These layers are interspersed with MaxPooling2D operations to reduce the spatial dimensions of the feature maps, thereby increasing the field of view and computational efficiency. To combat the issue of overfitting, especially prevalent in medical imaging due to the limited annotation in data, we integrated dropout layers. These layers randomly deactivate a fraction of neurons, thus enabling the model to learn more robust features. The model also includes UpSampling2D layers in its expanding path to reconstruct the segmentation maps from the compressed feature representation.

3.3 Loss Function

The Dice coefficient, integral to our custom loss function, is particularly effective for datasets with imbalanced classes in segmentation tasks. It is defined in terms of True Positives (TP), False Positives (FP), and False Negatives (FN), which are standard terms in classification problems. In the context of image segmentation:

- TP is the number of pixels correctly identified as part of the class (e.g., tumor).
- FP is the number of pixels incorrectly identified as part of the class.
- FN is the number of pixels incorrectly identified as not part of the class.

The Dice coefficient is formulated as follows:

$$\text{Dice} = \frac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}} \tag{1}$$

This formula calculates the ratio of twice the number of pixels correctly classified as the class (the intersection of the predicted segmentation and the ground truth) over the total number of pixels in both the predicted segmentation and the ground truth. In our implementation, the Dice loss, which is one minus the Dice coefficient, is used. This formulation frames the problem as a minimization one, where a lower Dice loss, and therefore a higher dice coefficient, indicates better overlap between the predicted segmentation and the ground truth. As we can see from Figure 2, we get a higher dice coefficient as we increase the epoch, giving us a more accurate model.

3.4 Data Processing and Visualization

Data preprocessing is a pivotal step in our methodology. We standardize the MRI scans by normalizing their intensity values and resizing them to a uniform dimension, ensuring consistency across the dataset. For visualization, we use Matplotlib, a versatile plotting library in Python. This tool allows us to

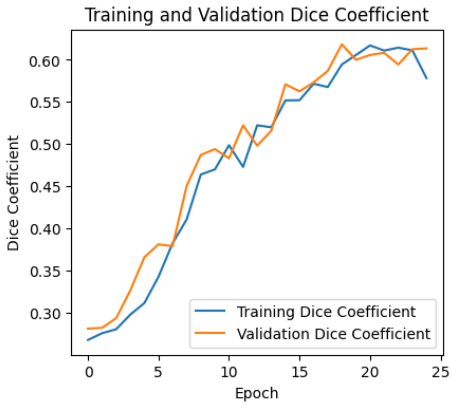

Fig. 2. Dice coefficient over 25 epoch

generate insightful visualizations of the MRI slices and their corresponding segmented outputs. These visualizations are useful in assessing the performance of our model on individual cases and provide an intuitive understanding of the model's capabilities in segmenting different tumor regions within the brain.

3.5 Training Process

The training of our model is a meticulous process, where we feed the preprocessed MRI scans into the network. We employ an Adam optimizer, a popular choice due to its efficiency in handling sparse gradients and adaptive learning rate capabilities. Throughout the training, we monitor various metrics, including loss and accuracy, to evaluate the performance of the model. This continuous monitoring allows us to make necessary adjustments to the model's parameters, ensuring optimal learning and convergence.

3.6 Explanatory Figures and Schematic Diagrams

The architectural details of our model are depicted using the `plot_model` function in Keras (this can be found in Appendix A), offering a schematic representation of the network as pictured in **??**. This visualization is complemented by additional figures created using Matplotlib, which illustrate the segmentation results on MRI slices.

4 Results

This section presents both qualitative and quantitative results obtained from our model. We showcase the effectiveness of our U-Net based CNN model in detecting and segmenting brain tumors from MRI scans. The results are presented through a series of figures and tables, each designed to convey a clear message about the model's performance and capabilities.

4.1 Qualitative Results

We visually demonstrate the model's segmentation accuracy on MRI scans. Figure 3 displays side-by-side comparisons of the original MRI images, the ground truth, and the model's predictions for a single slice of the axial plane.

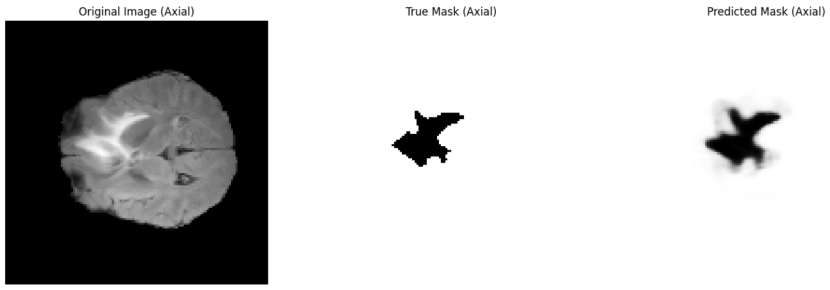

Fig. 3. Comparison of the original MRI image (left), ground truth segmentation (middle), and our model's prediction (right). This visualization highlights the precision and accuracy of our model in identifying and delineating tumor regions.

4.2 Quantitative Results

The model's performance is further evaluated using several metrics, including accuracy, Dice coefficient, precision, and recall. These metrics provide a comprehensive understanding of the model's effectiveness in tumor segmentation. We thought the most important parameters to describe our results, besides the dice coefficient which was already shown in Figure 2, were the accuracy and loss of our model. Figure 4 illustrates the accuracy and loss of our model. In order to plot these, we used to following loss function,

$$\text{Loss} = -\sum_{c=1}^{M} y_{o,c} \log(p_{o,c}) \tag{2}$$

where M is the number of classes(4 in our case), y is a binary indicator if the class is classified correctly, and p is the predicted probability of observing within the class. As for the accuracy, we used model.evaluate() from tensorflow to obtain the accuracy over 25 epoch.

We have summarized the metrics of the model in Table 1

4.3 Discussion of Negative Results

In our experimentation, we encountered certain challenges that are crucial to address for a comprehensive understanding of our model's performance. Notably,

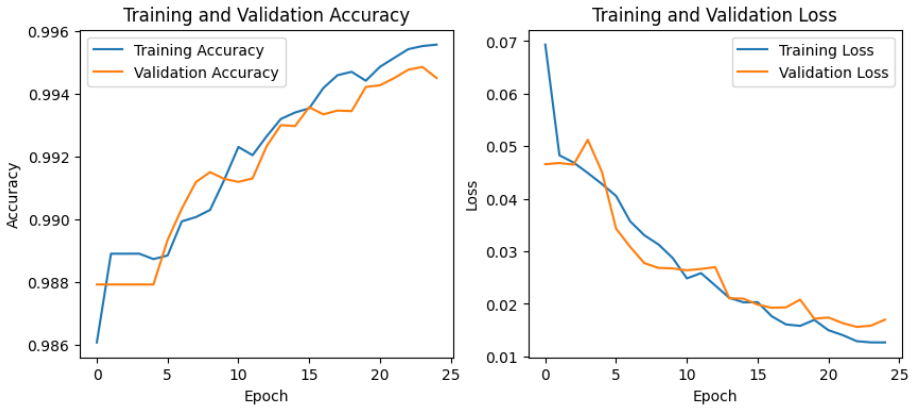

Fig. 4. Accuracy (left) and loss (right) for the training and validation data subsets over 25 epoch.

Table 1. Performance metrics of the CNN model.

Metric	Value
Accuracy	99.48%
Dice Coefficient	0.5987
Loss	1.637%

our model demonstrated reduced effectiveness on MRI scans with atypical tumor appearances. These cases often resulted in lower Dice scores, which could be attributed to the model's difficulty in accurately distinguishing tumor tissues from normal brain tissues in less common presentations.

Furthermore, we observed a notable occurrence of false positives, especially in cases where tumors were not present. This phenomenon of the model erroneously predicting the presence of a tumor can be linked to several factors:

1. **Data Imbalance**: Our training dataset may have had an imbalance in the representation of tumor versus non-tumor cases. Such imbalances can lead the model to be biased towards predicting the presence of tumors more frequently.
2. **Feature Overlap**: The features of non-tumorous regions in some MRI scans might closely resemble those of tumors, causing the model to misinterpret these as positive cases. This is particularly challenging in medical imaging, where benign anomalies can often mimic malignancies.
3. **Model Generalization**: While the model shows high accuracy, its ability to generalize to unseen, atypical cases may still be limited. This limitation often becomes evident when the model is exposed to diverse data outside of its training scope.
4. **Noise and Artifacts in MRI Scans**: MRI scans can sometimes contain noise and artifacts, which may lead to false-positive predictions. The model's

current preprocessing steps may not fully eliminate these factors, affecting its predictive accuracy.

Addressing these issues will be crucial in future iterations of our model development. Enhancements in data preprocessing, more balanced dataset composition, and advanced feature extraction techniques may help in reducing the rate of false positives and improving the model's ability to handle atypical tumor appearances.

5 Accomplishments

This was the first time we worked with real medical images and 3D MRI files. We had to learn how to open and use .nii files, which is an industry standard to store MRI images. We were able to create a fully functional machine learning model using tensorflow, which we have made accessible through a react web app for easier and better user interface. For now, each user will have to download all the necessary libraries and dependencies to run the model locally. Our model has a very high accuracy and a very small loss, which is ideal. This was accomplished by fine-tuning the hyper-parameters and through multiple training processes. We also managed to get very accurate tumor mask predictions with only some issues with the false positives when there is no tumor present. To finalize the project, we had to learn how to connect our tensorflow model with the react web app we made. This involved a lot of research and discussion, but we finally were able to do it and come up with the final result.

6 Contributions

- **Nicole Lowenstein**: Vedant, Sahba, and I came up with the project idea and began looking for data sets. I did some research into different architectures and finally decided on the U-net model. I wrote some code to open and visualize the files, but it was not used. I helped Vedant and Sahba with the training process and worked with them to implement a Flask server. I wrote most sections of the report and then corrected and added some important information that was missing.
- **Vedant Ashish Jain**:I co-developed the neural network model and processed the dataset, enhancing our model's performance. I also refined the visualization code, which improved our understanding of the results. After initial training, Sahba and I fine-tuned the model and developed a Flask server with Nicole. I integrated the front and back end of the app with Risa's help and later focused on adding new features like GIF creation and adapting the model for single JPEG image analysis.
- **Sahba Hajihoseini**: My work mainly focused on implementing the backend. I researched to find the most suitable architecture for the model, loss function, and dataset. I implemented the U-net model and tailored different aspects of it through extensive testing. I also crafted the loss function to meet

the requirements of our project. Moreover, I built the functions required to handle and split the dataset. I worked closely with Vedant and Nicole to tune different parameters of the model to get the highest accuracy possible. I also collaborated with Vedant and Risa to connect the back and front.
- **Japneet Sason**: For the front-end, I took on the task of studying and using React. While my code in the "japneet-test" branch was not used, I got familiar with React from the experience. I spent time on the report, most notably the background study and article collection for the literature survey and introduction/discussion sections. Additionally, I want to give credit to Risa for her work on the front-end, and for picking up my shortcomings in React.
- **Risa Kawagoe**: Designed the flow of user interaction and the user interface using Figma. Implemented the entire frontend for the production version in React. Wrote code for converting a pytorch model (.pt) to a ONNX model (.onnx) to load the model with ONNX runtime library, and for preprocessing input images. Worked on connecting the frontend and backend using the backend Flask server that Sahba, Vedant, and Nicole implemented; adjusted some implementation in server code to match the interfaces between frontend and backend.

7 Conclusion and Discussions

In conclusion, our project aimed to address the need for early and precise diagnosis of brain tumors through the development and evaluation of a complex U-Net based CNN model. Our model achieved notable results in both qualitative and quantitative assessments, demonstrating its capability to accurately detect and isolate brain tumors in MRI scans. The visual representations in Figure 3 highlight the precision and accuracy of our model, giving a clear picture of how well it performs in segmentation. The quantitative analysis, which is shown in Table 1 and Figure 4, indicated a low loss of 1.637%, a Dice coefficient of 0.5987, and an exceptional accuracy of 99.48%. These results support our model's efficiency in tumor detection and add to current research using CNNs for medical image analysis.

Our testing did, however, highlight certain difficulties and potential areas for development. Lower Dice scores were obtained by the model in cases with atypical tumor features. The number of false positives raised questions regarding potential noise in MRI scans, feature overlap, data imbalance, and model generalization, particularly when tumors were absent. In a critical analysis of the project, we acknowledge the need for continued work to improve model generalization, handle false positives, and improve preprocessing techniques. The observed constraints highlight the difficulty of analyzing medical images and the significance of ongoing model improvement.

8 Future Work

Suggested future work to pick up where we left off:

1. **Online Deployment of Model Testing Website:** Our current implementation involves a local Flask server for model testing, which has been effective for initial development and trials. However, for broader accessibility, we plan to deploy our website online. This transition will enable remote access, making it convenient for users to test and interact with our model from anywhere. The online platform will also facilitate a more robust user experience, allowing for real-time feedback and a more interactive interface.
2. **Implementation of Image Annotation:** In our current project phase, we utilized MRI images with pre-existing annotations. Moving forward, we intend to integrate a feature for border box annotation in our website. This addition will enhance the model's capability in accurately predicting tumor locations and dimensions. By allowing users to annotate images directly on the website, we can gather more precise data, which is crucial for improving the accuracy of our segmentation model.
3. **Introduction of Data Augmentation:** To date, our model training has not incorporated data augmentation techniques. We recognize the potential benefits of data augmentation in enhancing model robustness and generalizability. In future iterations, we plan to implement various data augmentation strategies, such as rotation, scaling, and flipping of images. These techniques will help in creating a more diverse training dataset, reducing overfitting, and improving the model's performance on unseen data.
4. **Integration with Detectron2:** A significant enhancement we aim to introduce is the integration of Detectron2, Facebook AI Research's advanced library for object detection and segmentation. Detectron2 offers state-of-the-art algorithms for detection and segmentation, which we believe will significantly improve our model's output quality. By leveraging Detectron2's powerful features, we expect to achieve more accurate segmentation results, especially in complex scenarios where precision is paramount.

These advancements aim to enhance the functionality, accuracy, and user experience of our segmentation model. They also provide a foundation for future development by other students or researchers in this field.

Acknowledgements

Firstly, we would like to acknowledge and thank our professor, Ghassan Hamerneh, for all his help and useful insights throughout all the stages of our project. We would also like to thank out TAs, Ben Cardoen and Weina Jin, for being present in our project discussions and being willing to help whenever they saw fit. We would also like to acknowledge the BraTS challenge for making their data public and accessible for us to use.

Appendix

We will include some detailed, low-level information to enhance the reader's understanding of our project without interfering with the flow of the article.

Appendix A

In Figure 5 and Figure 6 we present a detailed schematic of the convolution layers of our model.

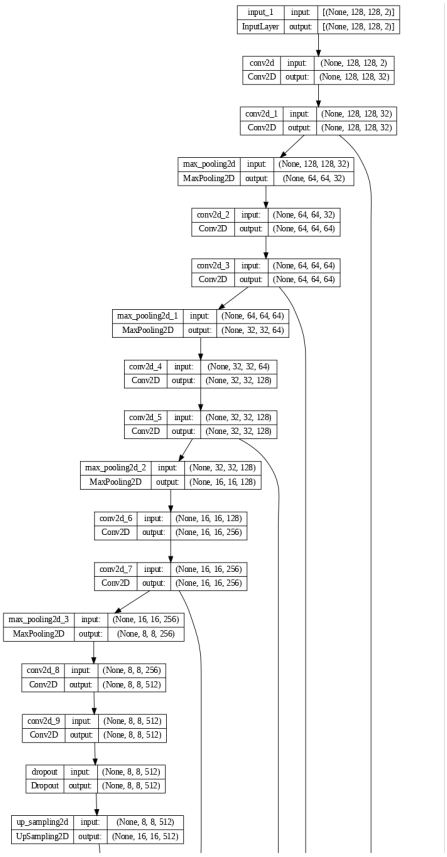

Fig. 5. Schematic of the model

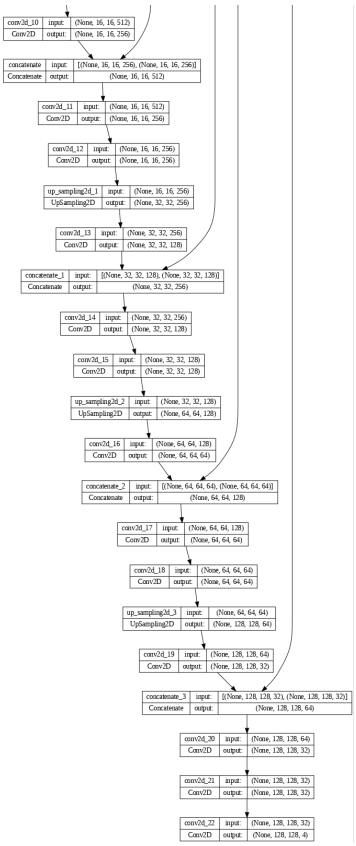

Fig. 6. Schematic of the model (cont'd)

Appendix B

We have written a detailed README.md file in our GitHub repository with instructions on how to run the program and how to install the necessary libraries and dependencies. Our repository is private but people with access to it will be able to use the following url.

References

1. H. H. Sultan et al. Multi-classification of brain tumor images using deep neural network. *IEEE Access*, 7:69215–69225, June 2019.
2. T. Hossain et al. Brain tumor detection using convolutional neural network. In *Proceedings of the IEEE International Conference on Sensors*, pp. 1–6. IEEE, May 3 2019.
3. M. G. Ertosun and D. L. Rubin. Automated grading of gliomas using deep learning in digital pathology images: A modular approach with ensemble of convolutional neural networks. In *AMIA Annual Symposium Proceedings*, volume 2015, pp. 1899. American Medical Informatics Association, 2015.
4. A. K. Anaraki et al. Magnetic resonance imaging-based brain tumor grades classification and grading via convolutional neural networks and genetic algorithms. *Biocybern Biomed Eng*, 39(1):63–74, 2019.
5. M. S. I. Khan et al. Accurate brain tumor detection using deep convolutional neural network. *Computational and Structural Biotechnology Journal*, 20:4733–4745, December 2022.
6. B. H. Menze et al. The multimodal brain tumor image segmentation benchmark (brats). *IEEE Transactions on Medical Imaging*, 34(10):1993–2024, 2015.
7. S. Bakas et al. Advancing the cancer genome atlas glioma mri collections with expert segmentation labels and radiomic features. *Nature Scientific Data*, 4:170117, 2017.
8. S. Bakas et al. Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the brats challenge. *arXiv preprint arXiv:1811.02629*, 2018.
9. S. Bakas et al. Segmentation labels and radiomic features for the pre-operative scans of the tcga-lgg collection. The Cancer Imaging Archive, 2017.
10. S. Bakas et al. Segmentation labels and radiomic features for the pre-operative scans of the tcga-gbm collection. The Cancer Imaging Archive, 2017.
11. S. Bakas. Brain tumor segmentation (brats) challenge.