CNN Classification of Brain Tumor MRI Images

Tristan Curry, Liam Fayle, Roman Koval, James Suresh

Abstract **-** Brain Tumors are highly prevalent in Canada, and are regularly found using Magnetic Resonance Imaging (MRI) techniques. The inclusion of Convolution Neural Networks in medical imaging can greatly assist in the diagnosis process as the trained models can aid clinicians in making decisions or replace clinicians in times of low staffing or budget. Currently, there exist several models of CNNs trained on tumor data, but are either very complex models, or require 3-dimensional data, which can be costly to acquire. This project aims at creating a model that is capable of classifying different types of tumors using a model trained on grayscale 2-dimensional MRI images. To do this Keras and TensorFlow were utilized to create a CNN model trained on hundreds of MRI brain scans with varying types of tumors. With this model, we can achieve an accuracy of 90.84%. These results are significant as they show that 2D data and simple model structures are capable of capturing complex features and classifying medical anomalies accurately.

Keywords - CNN, Brain tumors, Classification, MRI

I. Introduction

Magnetic Resonance Imaging (MRI) scans are pivotal in diagnosing and characterizing brain tumors. This non-invasive imaging technique employs strong magnetic fields and radio waves to generate detailed images of the brain's structures, offering superior resolution to other imaging modalities [1]. MRI scans provide precise information about the size, location, and nature of brain tumors, aiding healthcare professionals in accurate diagnosis and treatment planning.

In Canada, brain tumors remain a concerning health issue, with a notable prevalence impacting individuals across various age groups. According to the Canadian Cancer Society, approximately 50,000 Canadians are diagnosed with primary brain tumors each year [2]. Additionally, metastatic brain tumors, originating from cancers elsewhere in the body, contribute to the overall burden of brain cancer cases in the country.

In the medical field, CNNs are extensively employed for image classification tasks due to their ability to learn hierarchical features from raw pixel data automatically. This enables them to discern intricate patterns and features that might not be apparent to the human eye, facilitating more accurate diagnosis. One of the primary applications of CNNs in medical image classification is the detection and classification of abnormalities, such as tumors, lesions, fractures, and other pathologies.

By designing and training a CNN on MRI brain tumor data it is possible to create a model that is capable of detecting brain tumors without the input of clinicians. It will create an additional tool that can be utilized to help reduce the waiting time for results from MRI brain scans. Additionally, this network can assist clinicians in making decisions on ambiguous cases that may otherwise go undiagnosed or give clinicians increased confidence in their decisions.

II. Background

Magnetic Resonance Imaging (MRI) is a non-invasive medical imaging technique that generates detailed images of internal body structures using powerful magnetic fields and radio waves. It operates based on the behavior of hydrogen atoms within the body when exposed to a strong magnetic field. The machine creates a powerful magnetic field, causing hydrogen atoms in the body to align. Radio waves are then directed at the area of interest, disturbing these aligned atoms. When the radio waves are turned off, the atoms emit signals that are detected by the MRI machine's receiver coils. These signals are processed by a computer to construct detailed, cross-sectional images of the body [3].

CNNs comprise multiple layers: convolutional, pooling, and fully connected layers. The convolutional layers apply filters to input images, extracting essential features through convolutions. These filters detect different features like edges, shapes, or textures, enabling the network to learn representations at different levels of abstraction. Pooling layers then downsample the convolved features, reducing complexity while preserving important information. This step enhances the network's robustness to variations in input. Fully connected layers interpret the features extracted in previous layers and make classifications or predictions.

Activation functions introduce non-linearities essential for the network to learn intricate patterns and representations in data. They enable the network to approximate and understand complex functions beyond linear relationships, producing better generalization and feature extraction capabilities. Choosing the right activation function is important for the model's overall performance and convergence speed. Common activation functions in CNNs include Sigmoid, Tanh, the Rectified Linear Unit (ReLU), and variants like Leaky ReLU and ELU. ReLU has gained popularity as it is computationally simple and for its effectiveness in combating the vanishing gradient problem, allowing for faster convergence during training.

To measure the performance of the CNN models, two measures of performance are tracked, accuracy and Categorical Cross Entropy. Categorical Cross Entropy and Accuracy are fundamental metrics used to evaluate the performance of CNN models, especially in classification tasks. Accuracy represents the proportion of correctly classified samples among the total samples. It is a straightforward metric gauges the model's ability to predict classes. While accuracy is a vital measure of performance, it might not be sufficient for imbalanced datasets, as it doesn't account for false positives or false negatives. Categorical Cross Entropy is a loss function that measures the dissimilarity between predicted class probabilities and actual class labels in multi-class classification. It quantifies the distance between the predicted probability distribution and the true distribution of the classes. Lower cross-entropy values indicate better alignment between predicted and actual outcomes, reflecting improved model performance.

III. Related Work

As the use of CNNs in the medical field is widespread, it is expected that other work related to the identification of brain tumors is present. One such study conducted by a group from the Ahsanullah University of Science and Technology "Brain Tumor Detection Using Convolutional Neural Network", conducted a study on the effectiveness of CNNs on the segmentation of brain tumors from MRI images. The study created a CNN model utilizing MRI images to conduct image segmentation to determine if a tumor is present. This model achieved a high accuracy of 97.87% with its training data of 174 images. In contrast to our endeavor, the paper's model did not seek to classify the tumor present in the MRI image [4]. The largest improvement between our model and the model outlined in the paper is the categorization of tumors present in the MRI scan. This difference can allow clinicians to immediately create a treatment plan for the tumor as knowing the category limits the number of viable treatment plans to known treatment options.

In addition, the paper "Microscopic brain tumor detection and classification using 3D CNN and feature selection architecture" by Rehman et al. from the Artificial Intelligence & Data Analytics Lab CCIS at Prince Sultan University [5], created a 3-dimensional CNN capable of detecting and segmenting images. The method developed utilized different methods of MR Imaging such as Fast Fluid-attenuated Inversion Recovery (FLAIR), T1CE, and T1 and T2 weighting, to better segment and classify the tumor types that appear best in each MRI imaging modality. This method is highly effective in segmenting and uses this model to train the classification model using transfer learning. In doing this the classification model achieves high accuracy values between 90.4% and 95.6% for the tumor above imaging types. In contrast to this study, we will not use transfer learning and will use 2-dimensional data, resulting in a lower training time and less complex model required to train and categorize the images. This will allow us to achieve a similar result with much smaller computing power required.

IV. Methodology

The data collection process utilized a thorough and integrative method, merging two distinct datasets to form a robust base for the predictive model. Initially, the team's efforts focused on using a single dataset [6], but this approach led to suboptimal results, with the model's accuracy not exceeding 80%. To enhance performance, a second dataset was integrated into the data pool [7]. During this process, several identical images in both datasets were identified. To address this, a hashing technique was employed to compare the byte sequence of each image. Identical hashes indicated duplicates, which were then removed, ensuring that each image was unique in the dataset. This step was crucial to prevent data leakage into the validation and testing sets, thereby avoiding inflated estimates of the model's performance. Although the merged datasets contained duplicate images, their integration led to a significant 66% increase in data volume. This expansion is important not just in terms of quantity, but also substantially improves the quality and diversity of the data. Enhancing the data quality and variety is essential for developing models that are not only accurate but also have wide applicability. The success

of this method is evident from the notable enhancement in the accuracy of the models.

Subsequent examination of the data distribution indicated a generally balanced representation among the various classes, albeit with a slight overrepresentation of the glioma category. Despite this minor imbalance, it did not significantly impact the precision of the final model. Initially, the disparity was addressed by applying class weights to balance the influence of each category during the training process. However, this intervention did not substantially alter the performance outcomes of the model, underscoring its inherent robustness and accuracy despite the presence of class imbalances.

The ImageDataGenerator from Keras was utilized for data preprocessing. This tool plays a vital role in enhancing model robustness and reducing the risk of overfitting. It achieves this by creating multiple altered versions of a single image, improving the model's capability to generalize from training data to new, unseen data. This process is illustrated in Figure 1, where the top left image is the original and the remainder are altered preprocessed copies. The preprocessing approach included a variety of techniques. Pixel values were normalized through rescaling, ensuring they fall within a 0 to 1 range. Additionally, random transformations were introduced to the images, including a rotation range of up to 20 degrees, horizontal and vertical shifts up to 20% of the total width and height respectively, and a shear range of 0.2 to alter the shape of the images. A zoom range of 0.2 was applied for random zooming within the pictures, and horizontal flipping was enabled to mirror images. Importantly, these transformations were carefully calibrated to ensure that crucial features, such as tumors, remained within the frame. This precaution was vital to avoid rendering images useless due to key elements being shifted out of view. Further, these measures were taken to simulate a range of perspectives and conditions that can arise in practical medical scans, thereby diversifying the dataset without the need for extra data. Moreover, 20% of the data was allocated for validation purposes, ensuring the model is consistently evaluated and fine-tuned on data it has not encountered during training. This step is crucial for assessing the model's effectiveness on new data. Each aspect of the preprocessing routine was carefully selected to reflect real-world variability. The distortion parameters chosen for the ImageDataGenerator were integral in preparing the model to perform reliably and effectively in practical applications.

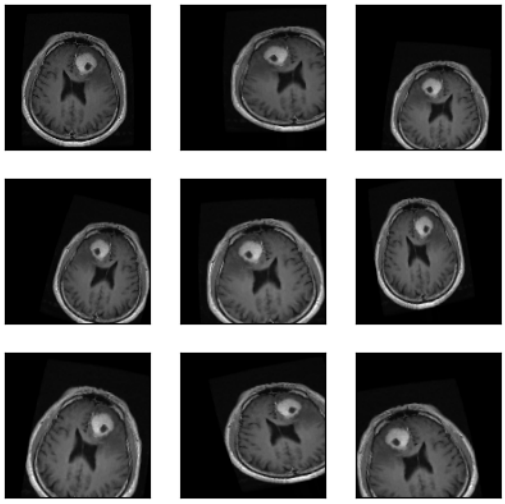

Fig. 1. Images Generated by ImageDataGenerator.

All CNN model building, training, and validation will be performed in Python using the Keras, Tensorflow, and Scikit-learn libraries.

A paper reviewing different CNN architectures was used to reference the most common algorithms and methodologies [8]. For convolution and fully connected layers, the Rectified Linear Unit (ReLU) was implemented as the activation function. This function is one of the most commonly used for activations, with a formula of $f(x) = max(0, x)$. It introduces non-linearity by returning the input directly if it is positive and zero otherwise. This function helps overcome the vanishing gradient problem encountered during training, as it does not saturate for positive inputs. Because of the simple formula, ReLU is quick to compute, reducing training times. In the implementation, the Keras library ReLU layer class is used to perform ReLU activation.

At the output layer, the softmax activation function is used for multiclass classification with one-hot encoded outputs. This function has a formula of $\frac{exp(x_i)}{\Sigma_j^N exp(x_j)}$ and returns a probability for each output class. For the brain tumor CNN with four classes, the softmax output layer would yield four different probabilities. The Keras library Softmax layer class provides the implementation for the function.

The model's loss is measured using the cross-entropy loss function. The function quantifies the difference between the predicted probability distribution and the actual probability distribution of a target class. For multiclass classification, the

formula is $H(y,p) = -\Sigma_i y_i \cdot \log(p_i)$, where y are the true probabilities and p are the predicted probabilities. The goal during training is to minimize the cross-entropy loss, which occurs when the predicted probabilities align closely with the true distribution of the classes. Compared to mean squared error, cross-entropy loss is preferred as it penalizes incorrect predictions more heavily, making it a more suitable metric for models that output probabilities. In the implementation, the Keras library CategoricalCrossentropy loss class is used.

The optimizer used for the CNN model is the Adaptive Moment Estimation (Adam) algorithm. It combines ideas from two other common optimization techniques, Root Mean Square Propagation (RMSprop) and momentum. Adam computes an adaptive learning rate for each parameter in the model based on the magnitude of a moving average of gradients and squared gradients. Adam is a common optimization technique that applies to a wide range of neural network architectures, including CNNs. The Adam algorithm is implemented using the Keras Adam optimizer library class.

When a neural network learns the training data too well, it can end up capturing noise and specific training data patterns, preventing it from generalizing to new, unseen data. This is named overfitting, and a regularization technique that is used to limit overfitting is dropout. During each training iteration, randomly selected neurons are "dropped out" with some probability. This means their outputs are ignored during forward and backward propagation for that iteration. During testing and validation, the dropout does not apply and all neurons are active. Dropout prevents neurons from relying too much on specific other neurons, making the network more robust and less prone to overfitting. The Keras library Dropout layer class with a 20% probability of dropout is used as the dropout implementation.

The CNN model will consist of convolution, max pooling, and fully connected layers. The convolution layers perform a 2D convolution over the spatial image data input. A default kernel size of 3x3 pixels will be used while adding zeroes as padding to the input image edges. These convolution layers will be implemented using the Keras library Conv2D layer class. After each convolution layer, a 2D max pooling operation will be performed. A 2x2 pixel window will be used to extract the most prominent features from the convolution output while also halving the input dimensions. The Keras library MaxPooling2D layer class is used after each convolution layer to perform the 2D max pooling. Finally, multiple fully connected layers will be present at the output of the model. These layers consisting of multiple neurons perform a linear transformation on the input through a weight matrix. By including fully connected layers at the output of a CNN, the high-level features learned by the convolution and pooling layers are flattened out and passed through the fully connected neurons to perform decision-making and output classification predictions. The Keras library Dense layer class is used to implement a fully connected layer.

Figure 2 shows a diagram of a final CNN model. This model has one input layer which is a grayscale 128x128 pixel input image with a single color channel. After the input, highlighted in blue, are four groups of convolution and max pooling layers. After is one last convolution layer followed by a global max pooling layer, this group is highlighted in green. The purpose of the global max pooling layer is to flatten the convolution output while also performing one final max pooling operation. Finally, highlighted in red is a fully connected layer with a dropout applied. At the output is a four-neuron fully connected layer.

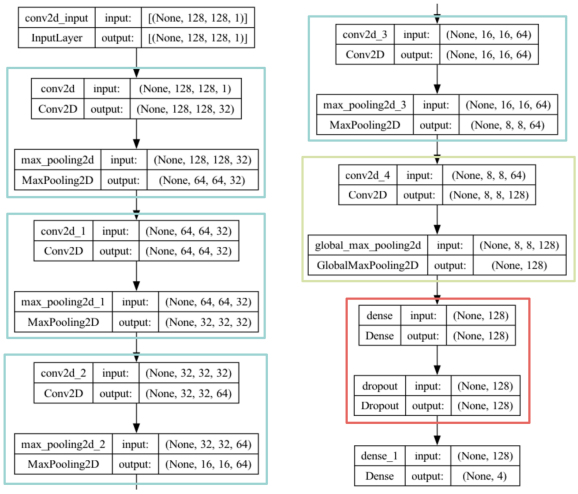

Fig. 2. CNN Model Diagram.

The CNN models were evaluated using the hold-out technique. The input images were split into three groups, 60% for training, 20% for validation, and 20% for testing. After the split, around 5000 images were used for training, and around 2600 total images for validation and testing. During training, the model was fed images from the training group and after each training iteration, the validation images were used to evaluate the model's accuracy. Once

training was complete, the model classified images from the testing group and was evaluated on its accuracy, precision, and recall. By using hold-out the model can be assessed on how well it generalizes to unseen data. This is crucial for understanding the model's ability to make accurate predictions on real-world examples beyond the training set. Using hold-out, the models were trained for a maximum of 300 epochs (a complete pass through the entire training dataset). To prevent the model's performance from plateauing, the optimizer's learning rate was reduced by a factor of 0.1 for every 10 consecutive epochs without an improvement in validation loss. Also, the early stop technique was implemented, halting the training if the validation loss does not show signs of improvement for 30 consecutive epochs. Early stop combined with a learning rate reduction prevents unnecessary training, reducing the total training time. The Keras Library EarlyStopping and ReduceLROnPlateau classes were used to implement early stop and learning rate reduction.

V. Results

To evaluate the performance of the CNN model, the accuracy curve (training accuracy and validation accuracy over time) was plotted for all the models. Initial models had a weak performance with the initial model achieving an average accuracy of around 66.45%±1.98% after only 36 epochs. The model stopped learning past the initial few iterations.

Further fine-tuning involved changing the configuration of the model by adding layers, neurons, and dropouts. The first significant leap in performance occurred when training the 5th model. The model achieved an accuracy of around 81.76% ± 1.01% after 79 epochs. This modal notably had been configured to support multiple (two additional) convolution layers.

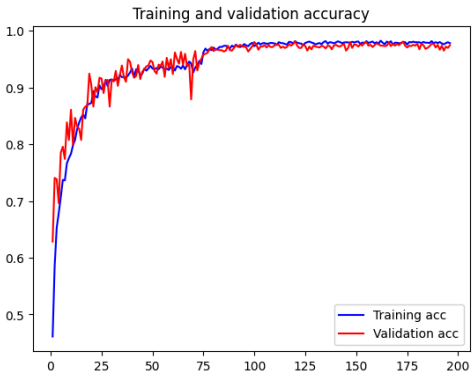

Fig. 3. Training and validation accuracy analysis of model 10

The model was further tuned to test the viability of other possible configurations. The final model (Model 10) provided the best performance achieving an overall accuracy of around 90.84% ± 1.08% after 196 epochs as depicted in Figure 3. Model 10 featured five convolution layers and a GlobalMaxPooling layer.

While Model 9 also had five convolution layers, the lack of a global max pooling layer caused the model to perform inconsistently with training accuracy fluctuating between ± 5-6%.

Table I. Classification Report

	Precision	Recall	f1-score	support
glioma	1.00	0.76	0.86	380
meningioma	0.86	0.92	0.89	301
No tumor	0.85	1.00	0.92	381
Pituitary	0.96	0.96	0.96	0.96
accuracy			0.91	1398
macro avg	0.92	0.91	0.91	1398
Weighted avg	0.92	0.91	0.91	1398

The following metrics as shown in Table I will be used to evaluate the performance of the final model. Precision is the positive predictive value [9], it is a representation of the accuracy of positive values that are correctly identified proportional to all positive values $\frac{TP}{TP+FP}$.
Recall (sensitivity) is a measure of true positive rate [9]. It is the rate at which true positives that are correctly identified by the model $\frac{TP}{TP+FN}$. F1 score is the mean of Recall and Precision $\frac{2\,(Precision \times Recall)}{Precision + Recall}$ and helps provide a balanced assessment of the classification algorithm's performance [9].

Glioma has a recall of 0.76 which is a relatively low measure. It also means that there are true cases of Glioma tumors that are not being classified properly. Glioma also has the lowest f1 score out of all the other tumors. But its high precision score of 1.0 means that there are no cases of false positives which means there is a low risk of misdiagnosing patients with this specific tumor. The meningioma class has the lowest precision score

0.86, which could indicate that some cases are wrongly misdiagnosed as this tumor. The algorithm is best at classifying pituitary tumors due to its high score of 0.96 on all metrics.

VI. Conclusions

Based on the analysis, the architecture achieved exceptional results for classifying tumors with an estimated accuracy of 90.84%. The report provides detailed and proficient insights into the various phases involved in the design, tuning, and implementation of the CNN model. It can achieve this level of efficacy without utilizing transfer learning or 3D imaging techniques [5]. The enhanced accuracy may be due to image alteration techniques provided by the ImageDataGenerator method that helped randomize the data thus helping it learn the images better in different configurations. During the tuning phase, the implementation of multiple convolution layers and the introduction of a global max pooling layer provided the highest improvement to the learning of the model.

The model can be enhanced by additional feature generation and hyperparameter tuning. By factoring in the limitations of its classification of certain tumors, separate models for analyzing meningiomas and gliomas can help further ensure higher accuracy. There are also fusion techniques that can be explored to incorporate the benefits of different neural networks[10]. A text-based MLP classifier can be utilized to analyze the different symptoms presented by tumor patients in combination with the CNN classifier to help diagnose patients more effectively.

The future of machine learning techniques and other technological advancements seems promising. Additional research into CNN and related image classification techniques can be truly beneficial for the medical industry.

Appendix

[1] Editor, “MRI vs CT scan: What’s the difference?” Insight Medical Imaging, https://x-ray.ca/daily-insight/mri-ct-scan-whats-the-difference/#:~:text=Some%20benefits%20of%20MRI%20include,X%2Dray%20or%20CT%20scan. (accessed Dec. 1, 2023).

[2] J. F. Megyesi et al., “Home,” Brain Tumour Registry, https://braintumourregistry.ca/#:~:text=Brain%20tumours%20affect%20approximately%2050%2C000,the%20rate%20of%20new%20diagnoses. (accessed Dec. 1, 2023).

[3] “Magnetic Resonance Imaging (MRI),” National Institute of Biomedical Imaging and Bioengineering, https://www.nibib.nih.gov/science-education/science-topics/magnetic-resonance-imaging-mri#:~:text=MRIs%20employ%20powerful%20magnets%20which,pull%20of%20the%20magnetic%20field. (accessed Dec. 2, 2023).

[4] T. Hossain, F. S. Shishir, M. Ashraf, M. A. Al Nasim and F. Muhammad Shah, "Brain Tumor Detection Using Convolutional Neural Network," 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT), Dhaka, Bangladesh, 2019, pp. 1-6, doi: 10.1109/ICASERT.2019.8934561.

[5] Rehman, A., Khan, MA, Saba, T., Mehmood, Z, Tariq, U, Ayesha, N. “Microscopic brain tumor detection and classification using 3D CNN and feature selection architecture”. Microsc Res Tech. 2021; 84: 133–149.

[6] P. Grover, “Brain_tumor_classification,” Kaggle, https://www.kaggle.com/datasets/prathamgrover/brain-tumor-classification/data (accessed Nov. 8, 2023).

[7] M. Nickparvar, “Brain tumor MRI dataset,” Kaggle, https://www.kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset/data (accessed Nov. 13, 2023).

[8] Alzubaidi, Laith et al. “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions.” Journal of big data vol. 8,1 (2021): 53. doi:10.1186/s40537-021-00444-8

[9] G. S. Handelman et al., "Peering Into the Black Box of Artificial Intelligence: Evaluation Metrics of Machine Learning Methods," AJR. American Journal of Roentgenology, vol. 214, no. 1, pp. 102-110, Jan. 2023. DOI: 10.2214/AJR.18.20224.

[10] M. A. Wajid, A. Zafar, H. Terashima-Marin, and M. S. Wajid, "Neutrosophic-CNN-Based Image and Text Fusion for Multimodal Classification," *J. Intell. Fuzzy Syst.*, vol. 45, no. 1, pp. 1039–1055, Jan. 2023. DOI: 10.3233/JIFS-223752.