Exploring Model Architectures for Brain Tumor Classification

M. Akmal, J. Bushey, S. Diep, H. Liu, C. Yang-Smith

University of Calgary

Abstract

Brain tumors are abnormal growths of cells within the brain that can disrupt normal brain function. As they constitute the potential to infiltrate other tissues, they pose great risk to the patient. With magnetic resonance imaging (MRI) data, many deep learning techniques can be employed to identify and classify the presence of tumors within a patient's brain. As a result, this study aims to evaluate the performance of three deep learning techniques for brain tumor identification: VGG2D, ResNet50V2, and Xception. We will examine GradCAM outputs for all three models and explain how model architecture may influence the models attention relative to the image dataset. The goal of this report is to compare existing computer vision architectures and analyze the impact different designs have on model performance.

1. Introduction

Brain tumor detection plays a crucial role in the diagnosis of cancer for affected patients. Early detection can significantly improve the prognosis and treatment plans available, yet this can be a difficult task as small and abnormally shaped or ambiguous tumors can be more difficult to identify [1]. Many deep learning techniques have been employed for brain tumor classification and segmentation [2, 3, 4]. The complexity ranges from implementing pre-trained computer vision models and performing transfer learning to a target dataset, to constructing more complex multiscale [3] or transformer [1] models to handle these tasks. However, the purpose across all remains the same.

2. Related Work

CNNs have contributed significantly in the areas of medical image understanding, including applications such as tumor detection and classification [5]. Considering brain PET Scans, Lee has analyzed the accuracy and performance of Inception3D, ResNet3D, and VGG3D [6], in which VGG performed the best of the three models with 97.7% accuracy. Misu has also considered various deep learning methods for brain tumor detection and has utilized five techniques: VGG16, VGG19, DenseNet121, ResNet50, and YoloV4 [2]. They have found that REsNet50 has performed the best, with 99.54% accuracy [2].

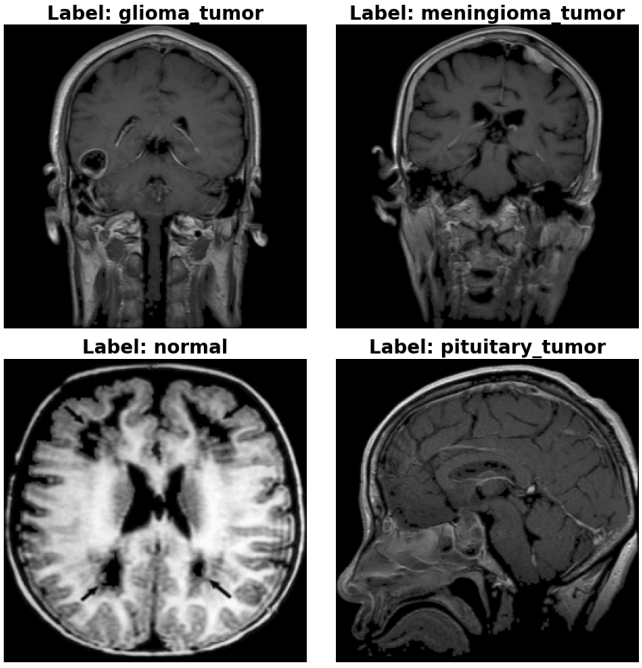

Fig. 1. Labels with corresponding images

3. Materials and Methods

The following section details the dataset, data preprocessing steps, model selection criteria, and training methodologies employed in this study to assess the performance of deep learning models for brain tumor classification. The dataset, its origin, and preprocessing steps are described first, followed by an explanation of model selection and training procedures. This section aims to provide a detailed account of the experimental setup, ensuring transparency and reproducibility in our approach.

3.1. Dataset

The dataset "Brain tumors 256 x 256" [7] is an enhanced dataset built upon the "Brain Tumor Classification (MRI)" dataset [8]. The inputs to our models are 256 x 256 pixel-size grayscale images which include three different head orientations (top-down, side-profile, and frontal) as shown in Figure 1.

3.2. Data Pre-Processing

Pre-processing was done by the creators of the dataset and includes:

- Removal of Redundant Data: Redundant data, including data augmentations like Salt and Pepper noise and geometric transformations, have been removed to ensure sample consistency.
- Image Normalization: Images have been normalized using their grayscale histograms, enhancing image quality and comparability.
- Resizing with Aspect Ratio Preservation: All images have been resized to a consistent 256 x 256-pixel size while preserving the original aspect ratio, ensuring uniform and detailed images.

3.2.1. Image Resizing

Pre-processing for ResNet50V2 and VGG2D includes resizing the images from 256 x 256-pixel size to 224 x 224-pixel size. This was performed to match the image size with the image size used in training pretrained models. The image size was kept at 256 x 256 for Xception net, as the network has an image input size of 299 x 299.

3.2.2. One Hot Encoding

The class labels were one-hot encoded to replace a categorical variable with one or more new integer values and ensures that the model does not assume that higher numbers are more important [9].

3.2.3. Class Weights

There are a total of 3,096 images in the dataset with distributions shown in Table 1. It's important to note that there were imbalances in the class distribution, which were addressed by calculating the class weights and passing them through the model.fit class_weight parameter. This helps to give more weight to underrepresented classes during training by directly specifying the weights for each of the target classes, thus improving the model's ability to generalize across all classes.

Table 1. Number of images in each class

Class	Number of Images
meningioma_tumor	913
glioma_tumor	901
pituitary_tumor	844
normal	438

3.2.4. Data Split

The dataset was split into a training set, to build the model, a validation set, to train the model, and a test set, to evaluate how well the model will generalize to new, previously unseen data [5].

For this application, a 70% training, 15% validation, and 15% testing split was selected.

3.3. Model Selection

Three convolutional neural network (CNN) models have been selected for image classification, where their performances were compared. Initially, a model was constructed based on the widely recognized Visual Geometry Group (VGG) architecture. Then, transfer learning was applied using two pre-trained models: ResNetV2 and Xception. Transfer learning was used because this method is useful when you do not have large amounts of data to train. In addition, transfer learning is a powerful technique as it uses features learned to solve one problem and adapts this representation to a different but related problem. [10]. These pre-trained models were trained on 1.28 million images from ImageNet to classify 1000 classes [11]. The selection of these models aimed to attain optimal performance across various tasks while examining their differences in structure: VGG utilizes stacked convolutional blocks with max-pooling downsampling, while ResNet utilizes shortcut ‘residual’ connections to expedite training for deeper networks [12, 13]. Finally, Xception employs depth-wise separable convolutions instead of standard convolutions to reduce the number of trainable parameters [14]. The categorical cross entropy loss function was in all models used to measure how well the predicted probabilities match the actual probabilities. Each model represents an exploration of new architecture with improvements from previous generations. They all terminate with a fully connected layer for classification.

3.3.1. VGG2D

VGG features a simple uniform architecture of stacking convolutional layers [12]. The structure of the VGG implemented by Lee et. al. consists of several convolutional layers followed by batch normalization, ReLU activation, max pooling, and dropout layers [6]. The final output layer uses softmax activation to classify the input image into one of the four classes.

The model begins with a convolutional layer with a 3x3 kernel size and padding set to 'same' to ensure that the output has the same spatial dimensions as the input. This layer extracts features from the input image using 24 filters and applies L2 regularization to the kernel weights to prevent overfitting.

The three subsequent convolutional layers follow a similar pattern, each doubling the number of filters compared to the previous layer. This allows the model to learn increasingly complex features at higher layers.

After each convolutional layer, batch normalization is applied to normalize the activations and improve gradient

flow which helps to stabilize the training process and accelerates convergence. ReLU activation is then applied to introduce non-linearity to the model and enable it to learn complex patterns in the data.

A ReLU activation was selected to follow the batch normalization to introduce non-linearity to the model allowing it to learn complex patterns in the images. ReLU additionally acts as a form of regularization and can help the model learn sparse representations as there may be only a few relevant features within the model.

Max pooling with a 2x2 pool size is used to reduce the dimensions of the feature maps and extract the most important features by storing only the pixels of maximum value within the pool.

Dropout is applied after the pooling layers to regularize the model and reduce overfitting by randomly setting a selection of input units to zero during training. This ensures the model doesn't memorize the training data by dropping connections.

The model ends with a fully connected layer with softmax activation, which outputs a probability distribution over the four classes. The class with the highest probability is predicted as the output class for the input image.

3.3.2. ResNet50V2

ResNet50V2 introduces residual connections to address the problem of the vanishing gradient in very deep networks [13]. Using a series of residual blocks with batch normalization improves network stability and performance. V2 of the model reduces the number of parameters using a pre-activation structure [15], despite having similar performance.

3.3.3. Xception

Xception builds upon the Inception V3 architecture [16] and utilizes depth-wise separable convolution blocks as they are more efficient and have fewer parameters than standard convolutions [14]. Additionally, the model employs global average pooling at the end of the network, followed by a fully connected layer for classification. A summary and comparison of model sizes by parameter count can be seen in Table 2.

Table 2. Comparison of Models

Model	Number of Trainable Parameters
VGG2D	3,600,644
ResNet50V2	14,974,980
Xception	4,756,996

3.4. Model Training and Learning Rate

For optimal results, we implemented a learning rate scheduler to improve the training process by allowing the model to converge faster, achieve better generalization, and avoid overfitting.

An early stopping callback was used with every model that used to stop training when a monitored metric has stopped improving. It helps prevent overfitting by ending training early if the model's performance on a validation set begins to worsen or stops improving.

In every model, a checkpoint callback was used to monitor the validation loss and save the model with the best validation loss for use in testing.

3.5. GradCAM Analysis

Grad-CAM (Gradient-weighted Class Activation Mapping) is a technique used to visualize the regions of an input image that are important for a CNN to make its classification decision. It does this by generating a heatmap that highlights the regions of the image that contribute the most to the predicted classification. By viewing these heatmaps, we can gain insights into what features the model is focusing on when making its predictions.

Grad-CAM analysis was performed on each model to determine how the model determines each class based on image features. This helped with understanding why the model makes certain predictions from which areas of the image. The Grad-CAM could also assist in identifying cases where the model might be making incorrect predictions due to focusing on irrelevant or misleading parts of the image which was valuable information to know when the model was performing incorrectly regardless of the accuracy values.

4. Results and Discussion

The models were fine-tuned by changing the hyperparameters and evaluating the metrics—categorical cross-entropy loss, accuracy, and AUC—to get the best model. Hyperparameter tuning resulted in the statistics for results showing in Table 3. The number of epochs was set to 200 for all models and batch size was set to 8.

4.1. VGG2D Findings

The hyperparameters that produced the best results (lowest loss) are as follows: dropout: 0.5, regularization parameter (L2): 0.01, batch size: 4. The learning rate was initially set at 0.01 and reduced by half every 10 epochs. This learning rate schedule was chosen to facilitate better convergence and performance for training deep learning models.

The VGG2D model achieved a test accuracy of 86.0% and an AUC score of 0.97. Despite the challenges posed by classifying brain tumors and the small dataset, the model demonstrated promising performance.

Analysis of the confusion matrix revealed that the model performed particularly well in distinguishing pituitary tumors from normal brain tissue. The use of a VGG architecture proved to be effective in this classification task, showcasing its potential in medical image analysis.

Overall, the results highlight the efficacy of the VGG2D model in classifying brain tumors and normal brain tissue. Future work could focus on further optimizing the model and exploring its applicability in other medical image analysis tasks.

4.2. Transfer Learning Models Findings

Training both pre-trained models first started with freezing the feature learning layers and then re-training the top(classifier). Then, unfreeze the last convolution layer in ResNet50V2 "conv5_block3_3_conv" and in Xception "block14_sepconv1". It was found that when only the classifier was re-trained the loss was high. To reduce the loss, a better indicator of model performance, unfreezing the last convolution layer in the feature extraction was necessary. After initial training with a learning rate of 1e-3 and a scheduler halving the rate every 10 epochs, we found that the model converged slowly. To address this, we adjusted the learning rate schedule to halve every 40 epochs with a starting rate of 5e-3. This change, along with setting the regularization parameter (L2) to 0.01, resulted in improved performance for both models.

4.2.1. ResNet50V2

The classifier was replaced by a combination of different layers as follows:

1. GlobalAveragePooling2D, BatchNormalization, Dropout, Flatten, Dense
2. MaxPool2D, BatchNormalization, Dropout, Flatten, Dense
3. GlobalAveragePooling2D, Flatten, Dense
4. Flatten, Dense

Defining the classifier as GlobalAveragePooling2D, Flatten and Dense resulted in the best accuracy (number 3 above). For the layers where dropout layer was included, dropout of 0.2 gave the best accuracy where a range of 0.2 to 0.5 values were iterated over.

For unfreezing feature extraction layers two options were tried: unfreezing conv4 and conv5 and unfreeze conv5 only. Between the two, only unfreezing the last convolution layer gave the best result.

4.2.2. Xception

The classifier layer was replaced with two options to match with the number of outputs in this problem.

1. GlobalAveragePooling2D, Flatten, Dense
2. Flatten, Dense

For unfreezing feature extraction layers the following modifications were tested:

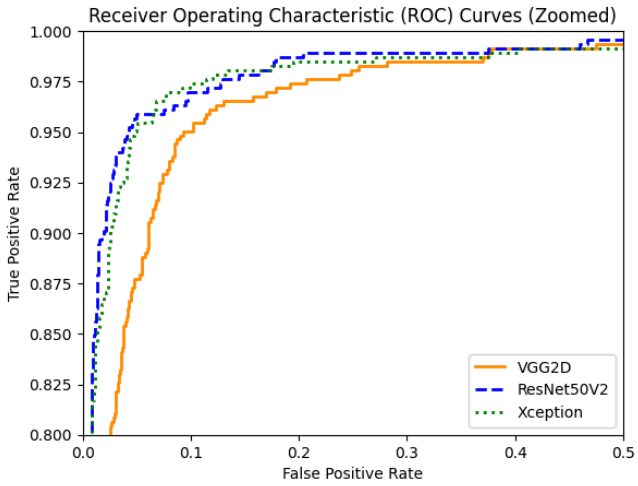

Fig. 2. ROC Curve Zoomed

- unfreeze block12-block14
- unfreeze block 14
- unfreeze and replace only the top

The model performed the best when only unfreezing the last convolution layer (block 14) and defining the classifier as GlobalAveragePooling2D, Flatten and Dense.

4.3. Discussion

The results including the accuracy, loss, and AUC from each model are quite similar. This can be seen in Table 3 where the results are compared and in Figure 2 displaying the ROC curve. The model that performed the best is the ResNet50V2 model as it has the lowest loss. The ResNet50V2 has slightly better accuracy and AUC than the Xception.

Table 3. Comparison of Results

	VGG2D	ResNet50V2	Xception
Accuracy	0.86	0.92	0.91
Loss	0.43	0.31	0.32
AUC	0.974	0.986	0.984

Considering Misu's work, our results are in line with their findings as ResNet50V2 is also our best performing model [8]. This may be due to the residual connections providing more context, which is important considering the local placement, edges, and features of a brain tumor. VGG, by comparison, throws large computations at the problem with traditional convolution blocks; however, this does not seem to be quite enough to compete with ResNet50V2.

On the other hand, Xception uses depthwise-separable convolutional blocks which tend to be more

parameter-efficient, yet this may not imply better performance than the residual connections. Depthwise-separable blocks calculate their convolution on a single-channel basis; however, as our images are black-and-white, the advantages of utilizing depthwise-separable blocks may not be as profound as intended. In our case, the depthwise-separable blocks will perform calculations on each channel, but each channel will have the same value due to the grayscale of the image. That being said, the performance and loss between ResNet50V2 and Xception is quite close.

Comparing the results (Figure 3, 4, and 5) of the Grad-CAM Analysis shows VGG2D is the only model that significantly learned any features of the images. This indicates that the accuracy results of the transfer learning models does not reflect the accuracy of the prediction.

Table 4. Comparison of Percentage of False Negatives

Class	VGG2D	ResNet50V2	Xception
meningioma tumor	17.9%	7.1%	7.1%
glioma tumor	16.7%	12.9%	15.8%
pituitary tumor	2.3%	3.8%	5.3%
normal	21.3%	5.9%	5.9%

It is more important to classify all positive classes, brain scan with tumor rather than negative class (normal brain scan). Therefore, false positives are preferred over false negatives. It would be more detrimental to classify a brain scan as normal when there is a tumor present. A comparison of the false negatives per class are summarized in Table 4. This shows the distribution of false negatives for all classes across our chosen models.

5. Conclusions

In this experiment, we explored different models to solve the problem of classifying brain tumors in MRI images. We looked at developing a VGG model and applying transfer learning by fine tuning the pretrained models: ResNet50V2 and Xception. Our results show that ResNet50V2 outperforms VGG2D and Xception in terms of accuracy and loss. The use of GradCAM helped us understand the attention mechanisms of these models, providing valuable insights into their decision-making processes.

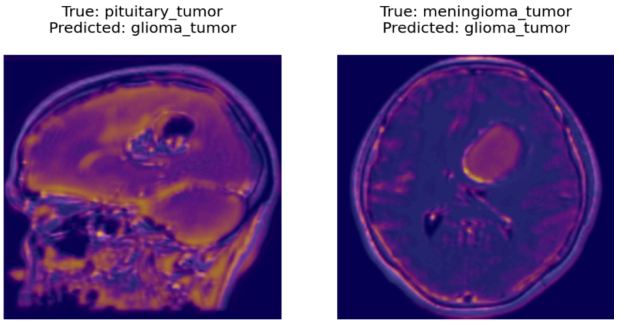

Fig. 3. VGG2D Grad-CAM Analysis

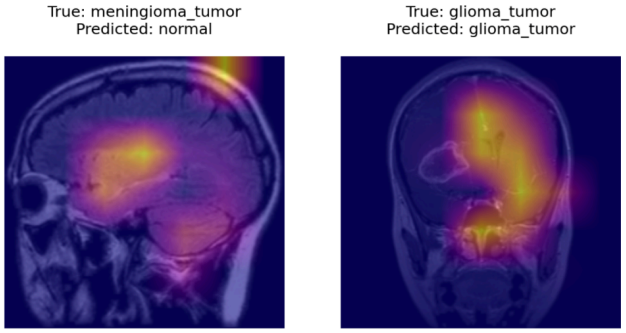

Fig. 4. ResNet50V2 Grad-CAM Analysis

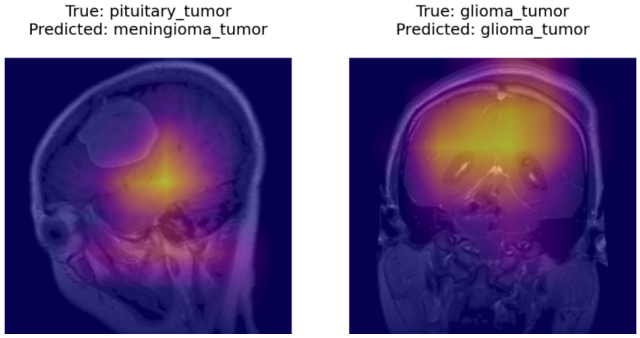

Fig. 5. Xception Grad-CAM Analysis

While our models achieved promising results, there is still room for improvement. Future experiments could focus on fine-tuning hyperparameters, implementing data augmentation techniques (including rotation, and horizontal/vertical flips), and exploring other model architectures (such as other pre-trained architectures, or implemented a more complex ensemble model) to further enhance the accuracy and robustness of brain tumor classification systems. Further exploration of transfer learning methods may improve accuracy. Overall, this experiment underscores the importance of selecting the right model architecture for optimal performance.

With more time we would implement data augmentation along with further tuning to try to further improve the accuracy and reduce the loss. Additionally, we could explore SHAP values as another way to evaluate the inner-workings of each model.

6. References

[1] He, Jingzhen, et al. “Cancer Detection for Small-Size and Ambiguous Tumors Based on Semantic FPN and Transformer.” PloS One, vol. 18, no. 2, 2023, pp. e0275194–e0275194, https://doi.org/10.1371/journal.pone.0275194.

[2] Misu, Razia Sultana. Brain Tumor Detection Using Deep Learning Approaches. 2023, https://doi.org/10.48550/arxiv.2309.12193.

[3] Nayan, Al-Akhir, et al. A Deep Learning Approach for Brain Tumor Detection Using Magnetic Resonance Imaging. 2022, https://doi.org/10.48550/arxiv.2210.13882.

[4] Balaji, Gopinath, et al. “Detection and Classification of Brain Tumors Using Deep Convolutional Neural Networks.” arXiv (Cornell University), 2022, https://doi.org/10.48550/arxiv.2208.13264.

[5] D. R. Sarvamangala and R. V. Kulkarni, “Convolutional neural networks in medical image understanding: a survey,” Evol. Intel., vol. 15, no. 1, pp. 1–22, Mar. 2022, doi: 10.1007/s12065-020-00540-3.

[6] Lee, Seung-Yeon, et al. “Performance Evaluation in [18F]Florbetaben Brain PET Images Classification Using 3D Convolutional Neural Network.” PloS One, vol. 16, no. 10, 2021, pp. e0258214–e0258214, https://doi.org/10.1371/journal.pone.0258214

[7] “Brain tumors 256x256: A Refined Brain Tumor Image Dataset with Grayscale Normalization and Zoom.” Kaggle, n.d. [Dataset]. https://www.kaggle.com/datasets/thomasdubail/brain-tumors-256x256/data.

[8] “Brain Tumor Classification (MRI): Classify MRI images into four classes.” Kaggle, n.d. [Dataset]. https://www.kaggle.com/datasets/sartajbhuvaji/brain-tumor-classification-mri.

[9] Müller, Andreas C., and Sarah Guido. Introduction to Machine Learning with Python : A Guide for Data Scientists. First edition., O’Reilly Media, 2017.

[10] Souza, Roberto. "Transfer Learning." Class handout for ENEL 645, University of Calgary, 31 Jan. 2024, https://github.com/rmsouza01/deep-learning/blob/master/PDFs/ENEL645/lecture08_transfer_learning.pdf. Accessed 27 Mar. 2024.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, Kai Li, and Li Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL: IEEE, Jun. 2009, pp. 248–255. doi: 10.1109/CVPR.2009.5206848.

[12] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” 2015, doi: 10.48550/ARXIV.1409.1556.

[13] Kaiming He, et al. “Deep Residual Learning for Image Recognition.” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2016, pp. 770–78, https://doi.org/10.1109/CVPR.2016.90.

[14] Chollet, François. “Xception: Deep Learning with Depthwise Separable Convolutions.” arXiv (Cornell University), 2017, https://doi.org/10.48550/arxiv.1610.02357.

[15] He, Kaiming, et al. “Identity Mappings in Deep Residual Networks.” arXiv (Cornell University), 2016, https://doi.org/10.48550/arxiv.1603.05027.

[16] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the Inception Architecture for Computer Vision,” 2015, doi: 10.48550/ARXIV.1512.00567.