International Journal of Computer Information Systems and Industrial Management Applications
ISSN 2150-7988 Volume 17 (2025) pp. 42–55
© Cerebration Science Publishing

Article

# Brain Tumor Classification and Segmentation Using Transfer Learning from MRI Images

**Konduru Reddy Madhavi [1,*], Kommanaboyina Lakshmi Sundara Soujanya [2], Rambabu Inaganti [3], Kondru Ayyappa Swamy [4], Sreekanth Yalavarthi [5] and Madhu Munagala [6]**

[1] School of Computing, Mohan Babu University, Tirupati 517102, Andhra Pradesh, India
[2] Department of CSE, G. Narayanamma Institute of Technology & Science, Hyderabad 500104, Telangana, India; Souj47@gmail.com
[3] Smith & Nephew, Pittsburgh, PA 15108, USA; Rambabu8@gmail.com
[4] Department of Electronics and Communication Engineering, Aditya University, Surampalem, 533437, Andhra Pradesh, India; Ayyappa.kondru@gmail.com
[5] HCL Technologies, Buffalo Grove, IL 60089, USA; sreekanth_y@yahoo.com
[6] Department of English, KL Deemed to-be University, Vaddeswaram, Guntur, 522302, Andhra Pradesh, India; dr.madhumunagala@gmail.com
* Correspondence author: kreddymadhavi@gmail.com

Received date: 15 June 2024; Accepted date: 12 September 2024; Published online: 6 January 2025

**Abstract:** Manual diagnosis of tumors on magnetic resonance images (MRIs) entails more time and doing so increases the risk of human error and incorrect tumor type identification and classification. Cells develop quickly and uncontrollably, which causes brain tumors and may cause death unless handled in the beginning stages. Therefore, a transfer learning framework for brain tumor classification is presented to simplify the task of healthcare professionals by automating tough medical processes. MRI image analysis was done on a publicly available dataset from Kaggle. The proposed method is implemented on VGG16, ResNet50, EfficientNetB0, and U-Net architectures. Training accuracy and test accuracy of all these four neural architectures compared and results that U-Net performs better in the classification of brain tumors compared to the other networks.

**Keywords:** transfer learning architectures; brain tumor classification; MRI; VGG16; ResNet50; EfficientNetB0; U-Net

## 1. Introduction

A group of illnesses known as tumors are characterized by abnormal cell growth that might invade and spread [1,2] to different parts of the body. The six hallmarks of cancer are present in all tumor cells. These qualities are required in order to develop cancerous growth.

They are as follows: Without the necessary signals, cell growth and division are inhibited. Even when given contradictory signals [3,4], there is constant development and division. There is no limit to the number of cell divisions that can be made.

Transfer learning is a prevalent computer vision strategy since it allows us to produce accurate models quickly. Transfer learning starts using patterns discovered when handling a different problem [5], as opposed to commencing from scratch. Consider it the deep learning equivalent of Chartres' standing on the shoulders of giants. It is typical to import and use models from the literature because of the high computational expense of training such models. Figure 1 demonstrates the transfer learning technique.

**Copyright:** © 2025 by the authors

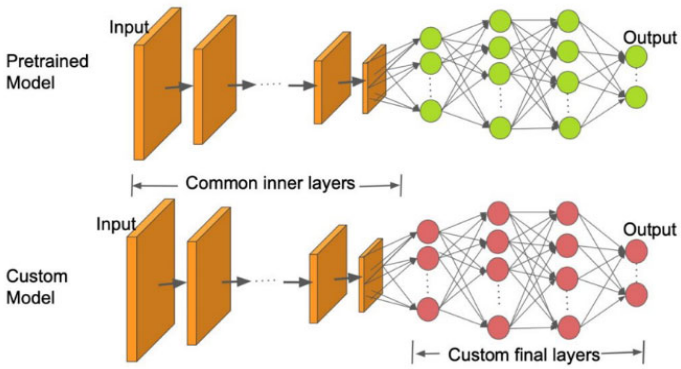


***Figure 1.*** Transfer Learning technique (Source: https://learnopencv.com).

## 2. Literature Survey

The human body's most crucial organ is the brain. It is necessary to detect brain-related illnesses early on. A broad spectrum of professionals has supported disease segmentation using a number of mechanical frameworks. Several analyzers have utilized data from MRI brain scans [6] and other sources.

Multiple transfer learning techniques were proposed to categorize and diagnose brain tumors in MRI images. In this research, transfer learning methodologies were reviewed and assessed, with VGG16 proving accurate.

This study used pre-trained neural networks to enhance [7] classification accuracy and training time. Excellent performance is achieved with a small number of training samples, saving time.

Deep learning-based. Using this method for identifying and categorizing cancer MRIs, glioma and meningioma are recommended as malignant tumor.

Current advances in brain tumor segmentation and classification using deep learning [4] investigated. This method suffers considerably due to a low accuracy score and a lack of resilience. Automated brain tumor classification, which attempts to enhance efficiency and accuracy [3] over manual classification. This approach has a training accuracy of 97.5 percent. To boost accuracy, a CNN architecture is used. The validation accuracy and validation loss of a model may be used to determine its quality.

A CNN-based technique is employed in this study to categorise glioma tumour [7] MR images. It has been shown that the suggested genetic algorithm utilised in the procedure outperforms comparable literature. High-performing and accurate automated [8] brain tumour classification system. A CNN-based categorization is utilised to improve accuracy and reduce computation time.

## 3. Proposed Methodology

The Deep Learning Specialization is a foundational study that will prepare you to contribute to the evolution of cutting-edge artificial intelligence technology by familiarising you with the capabilities, problems, and consequences of deep learning.

Convolutional neural networks, unlike ordinary conventional networks, are made up of neurons that have the appropriate biases and weights. Numerous approaches for examining the parameters of general neural networks are equally relevant to CNNs. CNN designs explicitly declare that the networks' inputs are pictures, enabling the design to include such qualities.

Convolutional neural networks make use of three critical concepts when developing a machine learning system are sparse interactions, equivariant representations, and parameter sharing. All these factors contribute to the forward mechanism's performance and significantly lower the network's parameter count. Additionally, it allows for the processing of inputs of varying sizes.

### *3.1. Two-Dimensional Convolution over Grey Scale Images*

The most often used kind of convolution is the two-dimensional convolution layer. A filter or kernel glides across the 2-dimensional input data in a 2-dimensional Convolution layer, performing multiplication with each and every element. As a result, the output pixels will be compressed into a single pixel. The kernel will repeat this procedure for each pixel it slides over, turning a two-dimensional feature matrix to a two-dimensional feature matrix. We multiply the filter with first 3 × 3 matrix obtained from 6 × 6 image, as seen in Figure 2. First element of 4 × 4 output matrix will now be element-wise sum of these values, which is 3*1 + 0 + 1*−1 + 1*1 + 5*0 + 8*−1 + 2*1 + 7*0 + 2*−1 = −5.

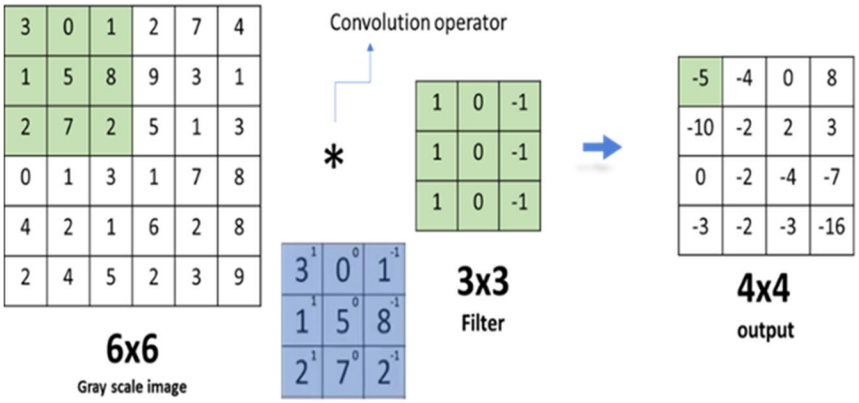


***Figure 2.*** Two-dimension Convolution Operation of a Grey Scale Image.

To compute the second element, we will move the filter to the right one position and recalculate the sum of the element-wise products.

## *3.2. Three-Dimensional Convolution over Color Images*

As with grayscale images, we will begin by selecting a filter of a specified size. The only modification will be that the filter will now be three-dimensional. The depth of the filter is defined by the number of colour channels in both our colour picture and the filter. As seen in Figure 3, we have an RGB picture that we desire to combine with the following 3D filter. As can be seen, the depth of our filter is composed of three 2D filters. For the purpose of simplicity, let us assume our RGB picture is 5 by 5 pixels. To prevent data loss during convolution, we are going to add zero padding to each of these arrays.

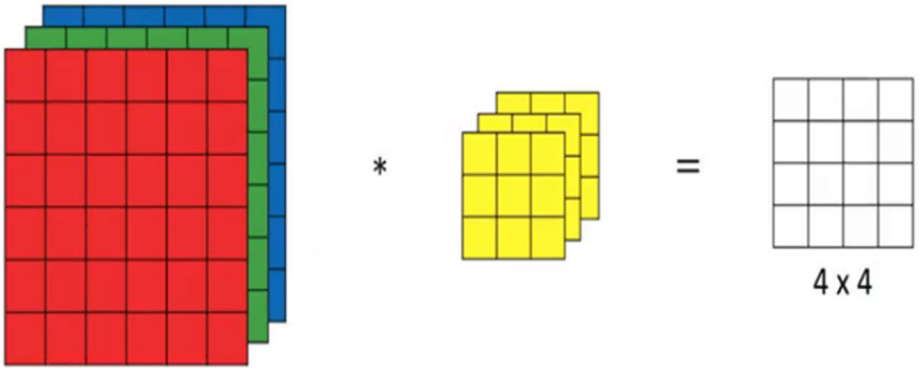


***Figure 3.*** After Convolution, Output Matrix is a 4 × 4 Matrix (Source: https://datascience.stackexchange.com).

## *3.3. Padding and Stride*

When utilising convolutional layers, one difficulty that arises is that we lose pixels at the image's borders. Because we often utilise small filters, we may lose just a few pixels for each convolution, but this may accumulate over time as we put on several convolutional layers. Simple solution is to increase the effective size of our input picture by adding additional filler pixels around its borders, as demonstrated in Figure 4. Normally, the values of the additional pixels are set to zero.

We begin with the convolution window at the top-left corner of the input image and move it down and to the right in all directions to calculate cross-correlation. Previously, we have always slid one piece at a time. However, we sometimes adjust our window more than once, bypassing nearby places for computational speed or to reduce sample size as shown in Figure 5.

| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 2 | 0 |
| 0 | 3 | 4 | 5 | 0 |
| 0 | 6 | 7 | 8 | 0 |
| 0 | 0 | 0 | 0 | 0 |

***Figure 4.*** Padding an Image with an Additional Border.

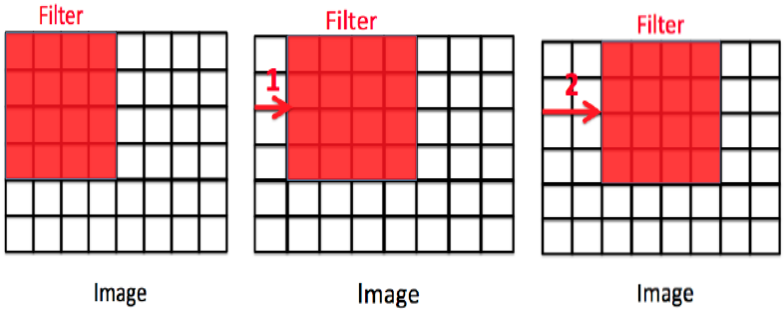


***Figure 5.*** Skipping the Adjacent Positions with stride = 0, stride = 1, stride = 2.

## *3.5. Dilated Convolution*

Dilated convolution is a kind of computation in which dilatation filters are used. Then it may be utilized to enhance the receptive field while decreasing the parameter count. Three instances of the dilated convolution process are shown in Figure 6. Convolution occurs in the orange squares; the blue squares are filled with zeros and do not affect the result. Each layer has the same number of parameters as the previous one. Additionally, the operator * is referred to as a dilated or l-dilated convolution. Discrete convolution, which is well-known, is identical to 1-dilated convolution. Historically, the term "dilated convolution" was used to refer to the dilated filter convolution operator.

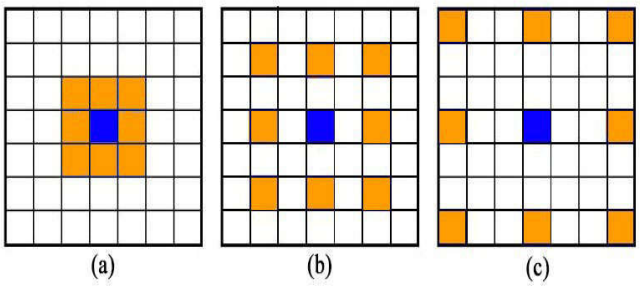


***Figure 6.*** Three instances of the dilated convolution process. (**a**) First Instance one (**b**) Second Instance (**c**) Third Instance

Dilated convolutional neural networks may be used to incorporate multiscale contextual input and to enable exponential receptive field growth without compromising resolution. As a result, the dilated convolution operator is ideal for dense prediction scenarios. Semantic image segmentation is a computationally difficult predictive task that involves pixel-level accuracy combined with multi-scale background information, as well as labelling each pixel in the picture. Dilated convolutional neural networks may be used to reduce the topology of deep networks towards semantic segmentation.

### *3.6. Pooling*

A pooling operation generates a picture based on the outputs of the previous layer at a given location. As shown in Figure 7, the max pooling method delivers the greatest value contained inside a rectangular region. Max pooling selects the brightest pixels in the picture. It is advantageous when the image's background is dark and we are just concerned with the image's brighter pixels.

Pooling layer may lessen the resolution of feature representation, number of parameters, and amount of computation required. It is beneficial in a variety of ways for preventing overfitting. Between successive convolutional layers in deep CNNs, a pooling layer is often utilised.

The feature representations now become approximately stable to moderate translation after a pooling operation. Translation invariance states that when a tiny fraction of the input is translated, the majority of the pooled data remain unchanged. When a feature is more significant than its specific location, it is preferable to be invariant to local translation.

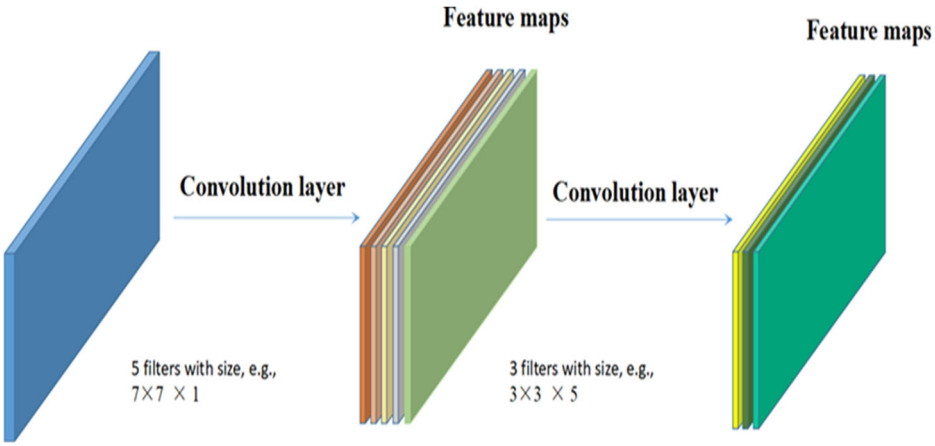


***Figure 7.*** Convolutional networks are constructed by stacking smaller filters on top of a larger scale of input.

The sparse interactions between input and output units of convolutional neural networks are widely known. This is made possible by keeping the filter size of CNNs smaller than the size of the input, shown in Figure 7.

To begin, we need to store minimal parameters, which minimises the memory requirements of the model. Figures 8 and 9 show the connectivity and parameter differences between a convolutional network and a fully connected network respectively. Additionally, they illustrate the difference in receptive area between a convolutional network and a fully linked network composed of a single node. Second, the product is often computed with far fewer computer operations. These productivity advantages are often significant in real-world applications.

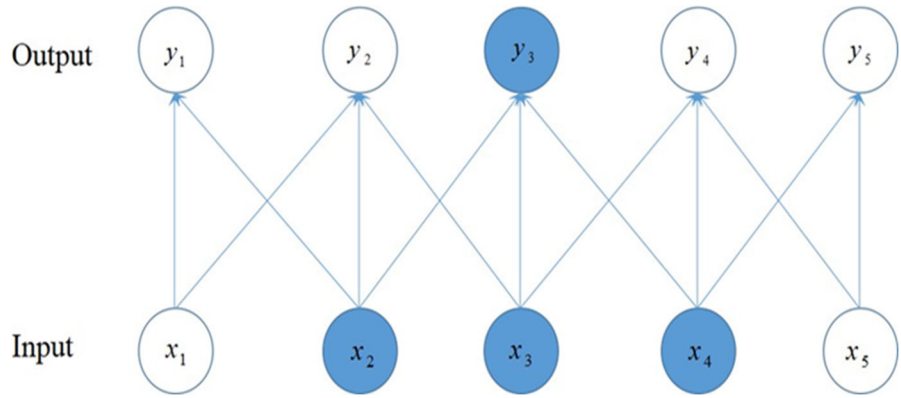


***Figure 8.*** Convolution techniques are shown in the following instances.

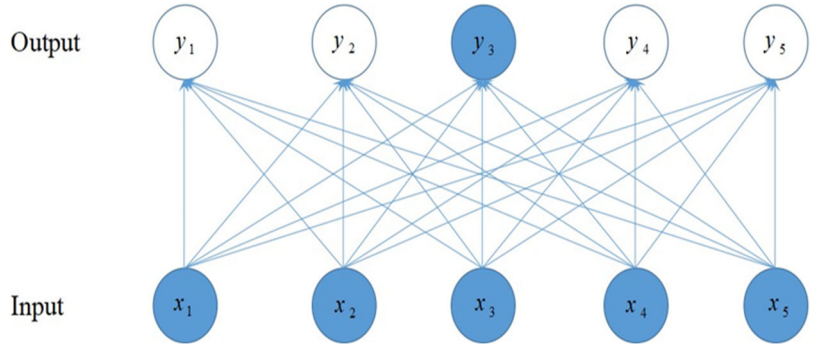


***Figure 9.*** The above row is created using matrix multiplication with complete connectivity.

In a neural network model, the usage of the same factor for multiple functions is referred to as parameter sharing, in which the amount of a weight applied to one input is determined by the amount of another weight applied to other place. Each convolution kernel component is used exactly once when computing the output of a single layer in a conventional neural network. Each kernel component is utilised at any pixel in the convolutional neural network's input picture (except maybe some pixels around the image border, based on the design, which may use boundary-pixel-handling methods, such as zero padding around the input picture border).

Convolution makes use of parameter sharing, which implies that instead of learning a unique set of conditions for each location, we just need to study one set of conditions distributed over all regions. This would significantly minimise the model's RAM capacity requirements. Convolution is much more efficient than dense matrix multiplication in terms of memory needs and number of parameters to be learned.

## *3.7. Batch Normalization*

Batch normalisation is a method of progressive parameterization that was developed to address the issue of training very deep models. Batch normalization often behaves differently in training and prediction modes. To begin, after the model has been trained, the noise introduced by calculating sample means and variances on mini batches is no longer acceptable. Second, we may be unable to calculate batch-by-batch normalization statistics. For instance, our model may be necessary to generate just a single prediction at a time.

Complex models are essentially the sum of multiple functions. It is difficult to train deep networks because the parameters of all previous layers impact the inputs to each layer, and all levels' parameters are updated simultaneously. When many functions or layers are modified simultaneously, unexpected consequences such as gradient vanishing as well as gradient explosion may occur.

## *3.8. Training the Deep Neural Networks*

Four different deep neural architectures have been experimented with for the purpose of brain tumor classification and segmentation using transfer learning from MRI images—VGG16, ResNet-50, EfficientNetB0, U-Net.

### 3.8.1. VGG16 Deep Neural Architecture

It is widely recognized as one of the most effective image-recognition models nowadays models ever designed. Modern object identification technology uses the VGG [9,10,11] model. In a wide range of tasks and datasets outside of ImageNet, VGG was developed as a deep CNN performs better than baselines.

### 3.8.2. ResNet-50 Deep Neural Architecture

The most used picture classification algorithm is ResNet. ResNet is a robust backbone model that is commonly [12,13] utilised in computer vision applications.

ResNet's most prominent feature is the skip connection. The ResNets architecture was originally developed for object localization, image recognition, but it may also be utilised for non-computer vision operations to provide depth and minimise processing costs.

### 3.8.3. EfficientNetB0 Deep Neural Architecture

EfficientNet can perform a variety of picture classification tasks[14–16]. As a result, it's an excellent model for transfer learning. In EfficientNet-B0, there are a total of 237 layers.

### 3.8.4. U-Net Deep Neural Architecture

U-Net is a convolutional neural network which was developed for the purpose of segmenting biological pictures. The network's architecture [17,18] was improved and expanded to allow for more precise segmentation with fewer training images.

Up sampling is the process of increasing the size of a sample. The Net segment has a large number of specialised channels, which allow a system to send contextual data to higher-resolution layers. This is a substantial change. The expanding and contracting routes are almost symmetrical as a consequence, resulting in a U-shaped structure. The network's Up sampling component includes a plethora of specialty channels.

U-Net beats earlier models in terms of design and pixel-based picture segmentation offered by convolutional neural network layers. Even photographs from a modest data source can be handled.

Th CNN architecture has been revised and modified to function with fewer training photographs while also providing more precise segmentation.

The segmentation map only contains pixels in the input picture for which the whole context is accessible since the network only utilises the valid component of each convolution. This approach provides for seamless segmentation of arbitrarily large photos using an overlap-tile methodology. Extrapolating the missing context needs mirroring the input picture in order to anticipate the pixels in the picture's border region. Because GPU memory would otherwise limit the resolution, this tiling strategy is required to apply the network to large pictures.

## *3.9. Optimizing the Deep Neural Networks*

Due to the time-consuming nature of deep neural network training, optimization solutions are critical. To address this issue, researchers developed a unique set of optimization strategies.

Identifying the neural network settings W that minimise a loss function Training neural networks includes calculating L(W), which is frequently composed of a measure of performance based on the data that is being trained and a few other regularisation components. The deep neural network training development includes a form that is similar to the machine learning development.

In a conventional machine learning model, the issue of universal optimization may be solved by carefully arranging the objective function and constraints such that the optimization issue is convex. However, we must handle the general non-convex optimization issue when training neural networks. When maximising the loss function of a neural network, a number of issues occur, including inappropriate Hessian matrix conditioning, local minima, and saddle points.

In a convex optimization problem, each local minimum is also a global minimum. Numerous local minima exist in non-convex functions, for example, the loss function of deep neural networks. In reality, the loss functions of enormously deep neural networks will have an increasing number of local minima. Local minima may be problematic if they have much greater loss levels than the global minimum.

Local minima (maxima) are unusual for a large number of non-convex high-dimensional functions, although saddle points are more frequent. The number of saddle points far outnumbers the number of local minima in many real-world applications. A saddle point is a stationary point in a function, but not an extremum, where the derivative or gradient is zero. Saddle point possess greater function values compared to saddle point itself, and others have lower function values at certain points.

Hessian matrix accommodate positive as well as negative eigenvalues at saddle point. Frequently, the gradient surrounding the saddle point is minimal and unimportant. However, first order approaches such as gradient descent, block coordinate descent, and others will much or less escape saddle spots, despite the fact that gradient descent may need exponential time to do so.

Additionally, there may be zones of constant value that are broad and flat. Both the gradient and the Hessian matrix are zero in these regions. Numerical optimization approaches have a difficult time dealing with these degenerate situations. A big and flat zone must include all global minima in a convex optimization problem, but in a generic optimization issue, the value of the corresponding region possesses high objective function.

## 4. Experimentation Results

MRI image data set is obtained from Kaggle. This dataset has four classes of images like glimo_tumor, no_tumor, meningioma_tumor, pituitary_tumor. A total of 3,264 MRI images are separated into four

different classifications after being categorised as training and testing data. Figure 10 shows the four classes of brain tumor available in the data set and which are to be classified from the dataset and the image given to model will predict the lesion from following classification.

The following are the characteristics of each model used with the dataset.

VGG16 Model
Predefined VGG16 Model
Input size – 224 * 224 * 3
ResNet-50 model
Predefined ResNet-50 Model
ImageNet weights are utilized
Input size – 75 * 100 * 3
Epochs – 24
Layers: 181 + Input Layer
Total params: 26,734,471
Trainable params: 26,681,351
Non-trainable params: 53,120

EfficientNetB0 Model
Predefined EfficientNet-B0 Model
ImageNet weights are utilized
Input size – 75 * 100 * 3
Epochs – 24
Layers: 224 + Input Layer
Total params: 4,203,290
Trainable params: 4,161,268
Non-trainable params: 42,022

U-Net model
ImageNet weights are utilized
Input size – 75 * 100 * 3
Epochs – 24
Layers: 224 + Input Layer
Total params: 5,122,801
Trainable params: 5,113,969
Non-trainable params: 8,832

In total 903 parameters are passed to fully connected network.

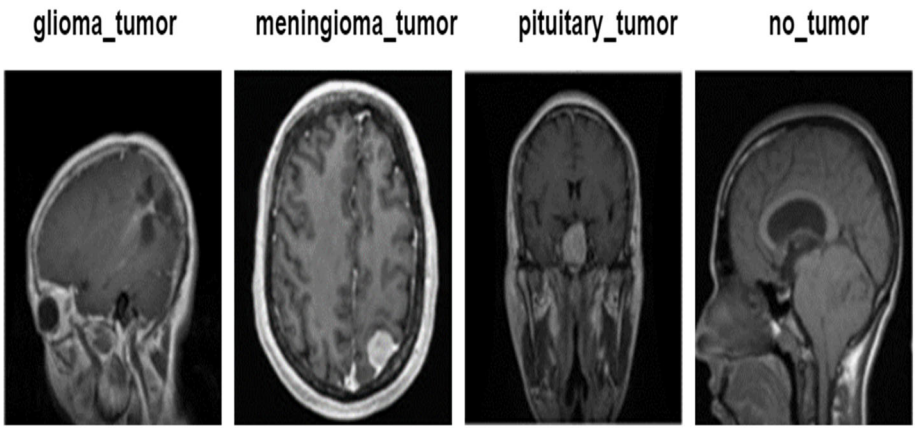


***Figure 10.*** Sample Tumor Images from Each Label.

The values of all these metrics for four architectures are shown in Tables 1, 2, 3 and 4 respectively. The U-Net model has been tested and trained. If the validation IoU score and dice loss are at their highest, the model is saved. Table 5 shows training, validation and testing accuracies of four architectures. Training, validation accuracy and loss values of VGG16 are shown in Figure 11 and heat map of VGG16 in Figure 12. Similarly, training, validation accuracy and loss values of RESNET 50 are shown in Figure 13 and heat map of RESNET50 in Figure 14, training, validation accuracy and loss values of EfficientNetB0 are shown in Figures 15 and heat map of EfficientNetB0 in Figure 16, training, validation accuracy and loss values of UNet are shown in Figure 17 and heat map of UNet in Figure 18, respectively.

***Table 1.*** VGG16 Model Accuracy.

| | **Precision** | **Recall** | **F1-Score** | **Support** |
|---|---|---|---|---|
| 0 | 0.93 | 0.73 | 0.82 | 93 |
| 1 | 0.67 | 0.86 | 0.75 | 51 |
| 2 | 0.73 | | 0.73 | 96 |
| 3 | 0.89 | | 0.92 | 87 |
| Accuracy | | | 0.81 | 327 |
| Macro average | 0.80 | 0.82 | 0.80 | 327 |
| Weighted average | 0.82 | 0.81 | 0.81 | 327 |

***Table 2.*** ResNet50 Model Accuracy.

| | **Precision** | **Recall** | **F1-Score** | **Support** |
|---|---|---|---|---|
| 0 | 0.97 | 0.92 | 0.95 | 93 |
| 1 | 0.94 | 0.96 | 0.95 | 51 |
| 2 | 0.95 | 0.96 | 0.95 | 96 |
| 3 | 0.97 | 0.99 | 0.98 | 87 |
| Accuracy | | | 0.96 | 327 |
| Macro average | 0.96 | 0.96 | 0.96 | 327 |
| Weighted average | 0.96 | 0.96 | 0.96 | 327 |

***Table 3.*** EfficientNetB0 Model Accuracy.

| | **Precision** | **Recall** | **F1-score** | **Support** |
|---|---|---|---|---|
| 0 | 0.98 | 0.97 | 0.98 | 93 |
| 1 | 0.96 | 1.00 | 0.98 | 51 |
| 2 | 0.98 | 0.98 | 0.99 | 96 |
| 3 | 1.00 | 0.98 | 1.00 | 87 |
| Accuracy | | | 0.99 | 327 |
| Macro average | 0.98 | 0.98 | 0.99 | 327 |
| Weighted average | 0.98 | 0.98 | 0.99 | 327 |

***Table 4.*** U-Net Model Accuracy.

| | **Precision** | **Recall** | **F1-score** | **Support** |
|---|---|---|---|---|
| 0 | 0.98 | 0.99 | 0.98 | 93 |
| 1 | 0.98 | 1.00 | 0.99 | 51 |
| 2 | 0.99 | 0.98 | 0.99 | 96 |
| 3 | 1.00 | 1.00 | 1.00 | 87 |
| Accuracy | | | 0.99 | 327 |
| Macro average | 0.99 | 0.99 | 0.99 | 327 |
| Weighted average | 0.99 | 0.99 | 0.99 | 327 |

## *Metrics for Performance*

The following metrics were used to find an accuracy of the models.

$$\text{Precision} = TP/(TP + FP)$$

$$\text{Recall} = TP/(TP + FN)$$

where

TP = True positive; TN = True-negative; FP = False-positive; FN = False-negative.

$$\text{F1-score} = 2 * (precision * recall)/(precision + recall)$$

Support is the no. of samples of true response that occurs in each class of target values.

***Table 5.*** Comparison of Deep Learning Models Accuracy.

| Model | Training Accuracy | Validation Accuracy | Test Accuracy |
|---|---|---|---|
| VGG16 | 97.16 | 87.68 | 86.78 |
| ResNet50 | 96.85 | 91.07 | 90.74 |
| EfficientNetB0 | 99.8 | 96.42 | 92.58 |
| U-Net | 97.35 | 97.56 | 95.23 |

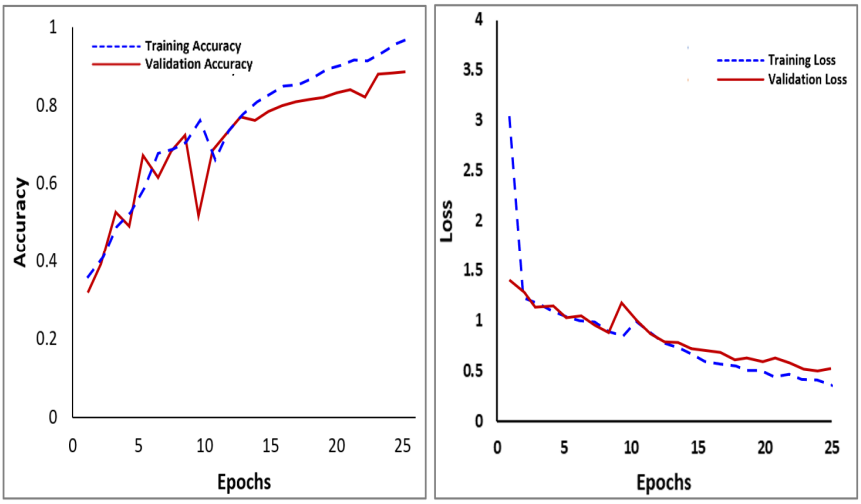


***Figure 11.*** VGG16 Epochs vs. Training and Validation Accuracy/Loss.

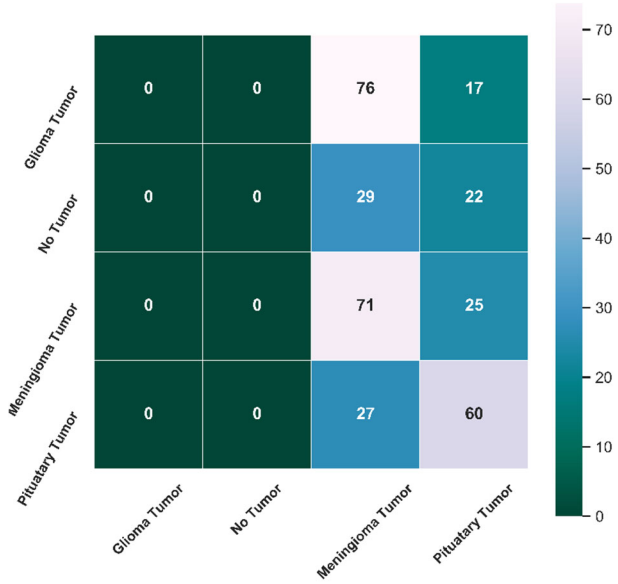


***Figure 12.*** VGG16 HeatMap.

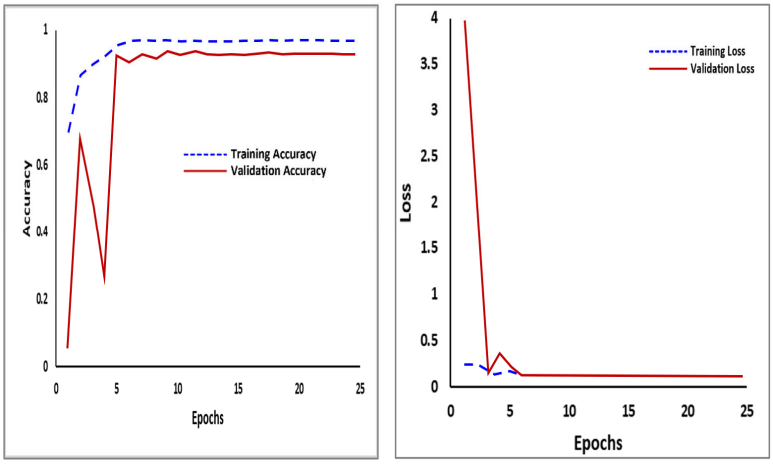


***Figure 13.*** ResNet50 Epochs vs. Training and Validation Accuracy/Loss.

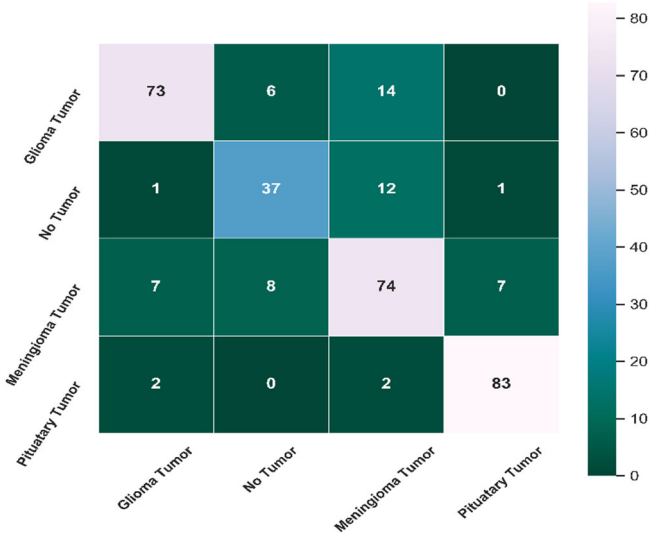


***Figure 14.*** ResNet50 HeatMap.

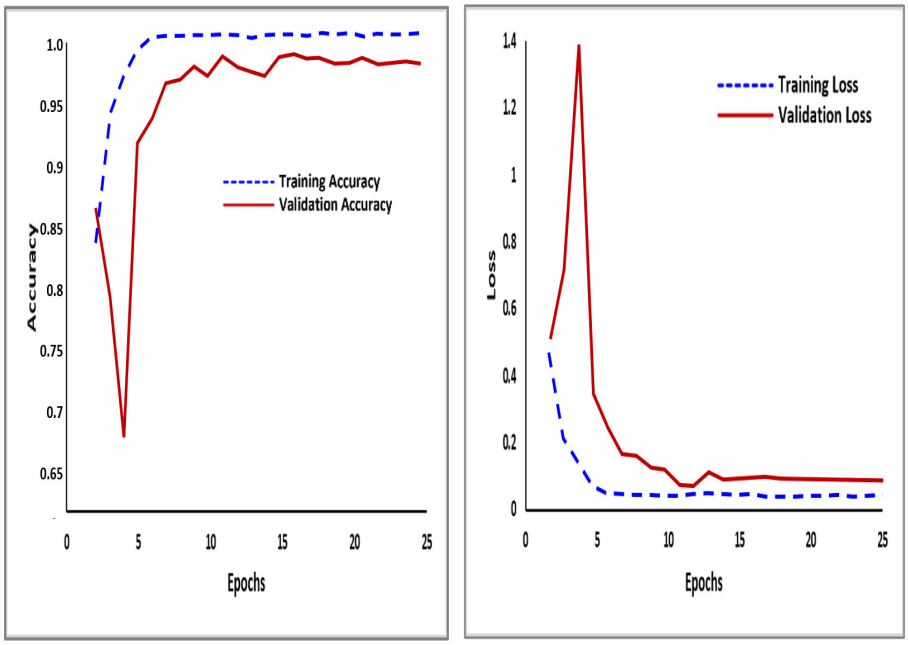


***Figure 15.*** EfficientNetB0 Epochs vs. Training and Validation Accuracy/Loss.

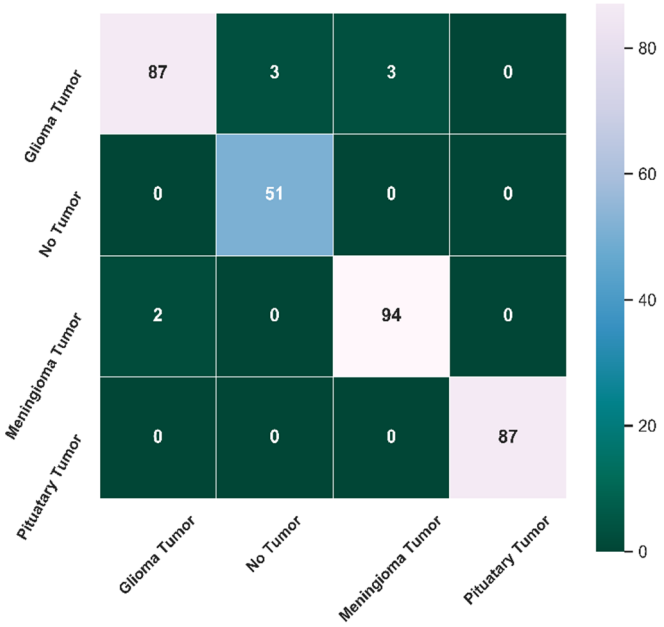


***Figure 16.*** EfficientNetB0 HeatMap.

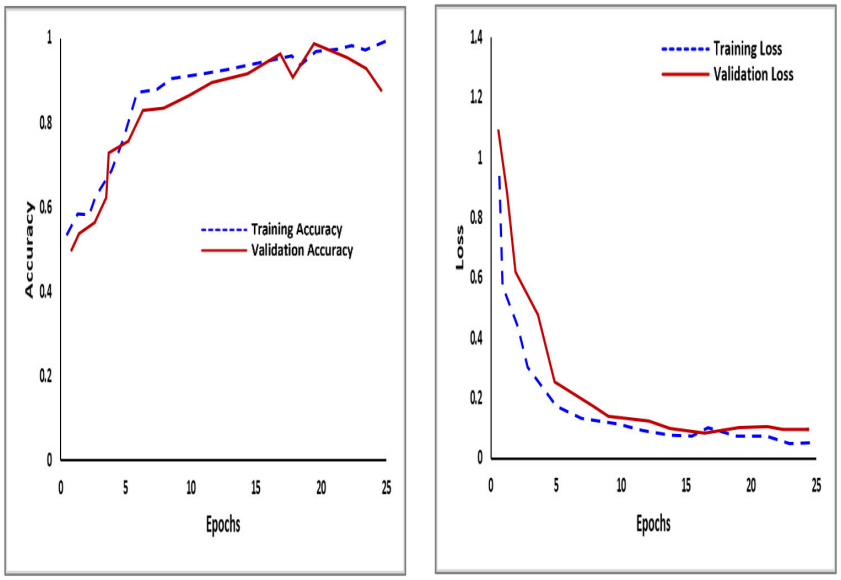


***Figure 17.*** U-Net Epochs vs. Training and Validation Accuracy/Loss.

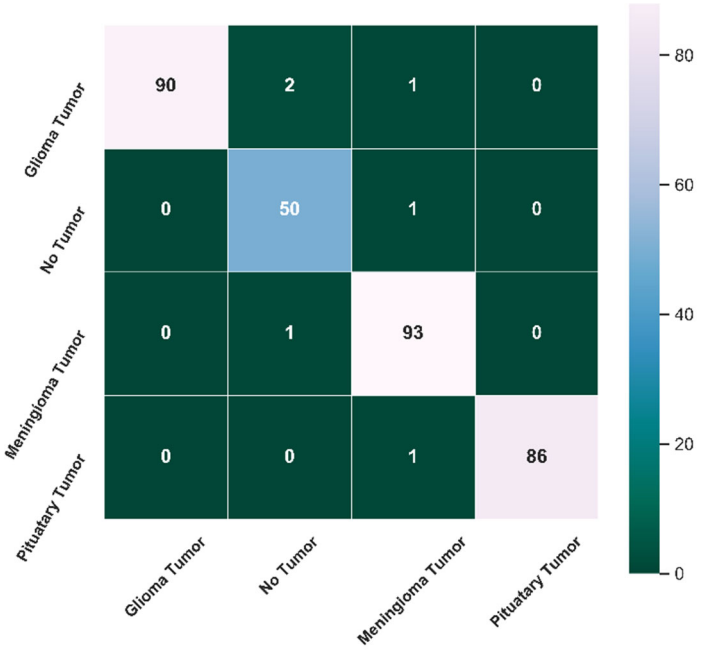


***Figure 18.*** U-Net Heatmap.

## 5. Conclusions

In this paper, we discussed four transfer learning architectures such as VGG16, ResNet50, EfficientNetB0, and U-Net for brain tumor classification using MRI image analysis on publicly available datasets from Kaggle. Training, validation, and test accuracy of these architectures were compared and showed that U-Net outperforms other networks. It serves healthcare professionals by automating medical diagnosis processes.

**Author Contributions**

All authors contributed equally, and all authors read and approved the final version of the paper.

**Funding**

No funding received from any organization.

**Conflict of Interest Statement**

Authors declare no conflict of interest.

**Data Availability Statement**

In this section, please provide details regarding where data supporting reported results can be found, including links to publicly archived datasets analyzed or generated during the study. Where no new data were created, or where data is unavailable due to privacy or ethical restrictions, a statement is still required.

**Acknowledgment**

The authors extend their gratitude for providing research facilities in their organization.

**References**

1. Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." *arXiv* 2014, arXiv:1409.1556.
2. He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In Proceedings of the IEEE Conference on computer vision and pattern recognition, pp. 770-778. 2016.
3. Reshma, G., Chiai Al-Atroshi, Vinay Kumar Nassa, B. T. Geetha, Gurram Sunitha, Mohammad Gouse Galety, and S. Neelakandan. "Deep Learning-Based Skin Lesion Diagnosis Model Using Dermoscopic Images." *Intelligent Automation & Soft Computing* 31, no. 1 (2022). https://doi:10.32604/iasc.2022.019117.
4. Avanija, J., Gurram Sunitha, K. Reddy Madhavi, S. Sreenivasa Chakravarthi, and R. Hitesh Sai Vittal. "Prediction and Analysis of Cervical Cancer: An Ensemble Approach." In 2022 4th International Conference on Inventive Research in Computing Applications (ICIRCA), pp. 1734-1739. IEEE, 2022. https://doi:10.1109/ICIRCA54612.2022.9985601.

5. Chelghoum, Rayene, Ameur Ikhlef, Amina Hameurlaine, and Sabir Jacquir. "Transfer learning using convolutional neural network architectures for brain tumor classification from MRI images." In Artificial Intelligence Applications and Innovations: 16th IFIP WG 12.5 International Conference, AIAI 2020, Neos Marmaras, Greece, June 5–7, 2020, Proceedings, Part I 16, pp. 189-200. Springer International Publishing, 2020. https://doi.org/10.1007/978-3-030-49161-1_17.
6. Rehman, Arshia, Saeeda Naz, Muhammad Imran Razzak, Faiza Akram, and Muhammad Imran. "A deep learning-based framework for automatic brain tumors classification using transfer learning." Circuits, Systems, and Signal Processing 39 (2020): 757-775. https://doi.org/10.1007/s00034-019-01246-3.
7. Sadad, Tariq, Amjad Rehman, Asim Munir, Tanzila Saba, Usman Tariq, Noor Ayesha, and Rashid Abbasi. "Brain tumor detection and multi-classification using advanced deep learning techniques." Microscopy Research and Technique 84, no. 6 (2021): 1296-1308. https://doi.org/10.1002/jemt.23688.
8. Sunitha, Gurram, Rajesh Arunachalam, Mohammed Abd-Elnaby, Mahmoud MA Eid, and Ahmed Nabih Zaki Rashed. "A comparative analysis of deep neural network architectures for the dynamic diagnosis of COVID-19 based on acoustic cough features." International Journal of Imaging Systems and Technology 32, no. 5 (2022): 1433-1446. https://doi.org/10.1002/ima.22749.
9. Younis, Ayesha, Li Qiang, Charles Okanda Nyatega, Mohammed Jajere Adamu, and Halima Bello Kawuwa. "Brain tumor analysis using deep learning and VGG-16 ensembling learning approaches." *Applied Sciences* 12, no. 14 (2022): 7282. https://doi.org/10.3390/app12147282.
10. Belaid, Ouiza Nait, and Malik Loudini. "Classification of brain tumor by combination of pre-trained vgg16 cnn." *Journal of Information Technology Management* 12, no. 2 (2020): 13-25. https://doi.org/10.22059/jitm.2020.75788.
11. Ghosh, Sourodip, Aunkit Chaki, and K. C. Santosh. "Improved U-Net architecture with VGG-16 for brain tumor segmentation." *Physical and Engineering Sciences in Medicine* 44, no. 3 (2021): 703-712. https://doi.org/10.1007/s13246-021-01019-w.
12. Sharma, Arpit Kumar, Amita Nandal, Arvind Dhaka, Deepika Koundal, Dijana Capeska Bogatinoska, and Hashem Alyami. "Enhanced watershed segmentation algorithm-based modified ResNet50 model for brain tumor detection." *BioMed Research International* 2022 (2022). https://doi.org/10.1155/2023/9865972.
13. Sharma, Arpit Kumar, Amita Nandal, Arvind Dhaka, Kemal Polat, Raghad Alwadie, Fayadh Alenezi, and Adi Alhudhaif. "HOG transformation based feature extraction framework in modified Resnet50 model for brain tumor detection." *Biomedical Signal Processing and Control* 84 (2023): 104737. https://doi.org/10.1016/j.bspc.2023.104737.
14. Tripathy, Sushreeta, Rishabh Singh, and Mousim Ray. "Automation of Brain Tumor Identification using EfficientNet on Magnetic Resonance Images." *Procedia Computer Science* 218 (2023): 1551-1560. https://doi.org/10.1016/j.procs.2023.01.133.
15. Zulfiqar, Fatima, Usama Ijaz Bajwa, and Yasar Mehmood. "Multi-class classification of brain tumor types from MR images using EfficientNets." *Biomedical Signal Processing and Control* 84 (2023): 104777. https://doi.org/10.1016/j.bspc.2023.104777.
16. Bengs, Marcel, Michael Bockmayr, Ulrich Schüller, and Alexander Schlaefer. "Medulloblastoma tumor classification using deep transfer learning with multi-scale EfficientNets." In Medical Imaging 2021: *Digital Pathology*, vol. 11603, pp. 70-75. SPIE, 2021. https://doi.org/10.1117/12.2580717.
17. Rehman, Mobeen Ur, SeungBin Cho, Jee Hong Kim, and Kil To Chong. "Bu-net: Brain tumor segmentation using modified u-net architecture." *Electronics* 9, no. 12 (2020): 2203. https://doi.org/10.3390/electronics9122203.
18. Chen, Wei, Boqiang Liu, Suting Peng, Jiawei Sun, and Xu Qiao. "S3D-UNet: separable 3D U-Net for brain tumor segmentation." In Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries: 4th International Workshop, BrainLes 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Revised Selected Papers, Part II 4, pp. 358-368. Springer International Publishing, 2019. https://doi.org/10.1007/978-3-030-11726-9_32.