CS577-Deep Learning: Project Final Report Deep Learning Approach in Brain Tumor Classification

Yi Shi
Illinois Institute of Technology
Chicago, Illinois, USA
yshi67@hawk.iit.edu
A20501004

Xiaoting Zhou
Illinois Institute of Technology
Chicago, Illinois, USA
xzhou70@hawk.iit.edu
A20503290

Chieh Jui Lee
Illinois Institute of Technology
Chicago, Illinois, USA
clee114@hawk.iit.edu
A20523922

ABSTRACT

Brain tumors demand swift and accurate detection for optimal treatment outcomes, yet current diagnostic procedures often involve invasive measures. This research proposes a non-invasive solution through the implementation of deep learning and machine learning techniques. By analyzing MRI images, our system aims to classify glioma, meningioma, and pituitary gland tumors, as well as identify healthy brains. We explore Convolutional Neural Networks (CNNs), transfer learning, attention mechanisms, and Generative Adversarial Networks (GANs) for data augmentation and image enhancement. This multifaceted approach seeks to advance brain tumor classification beyond traditional methods, offering a promising avenue for early and precise detection.

KEYWORDS

detection, classify, analyze, advance, customer, transfer, data, identification

1. OVERVIEW

Brain tumors represent a significant health concern, necessitating early detection for effective treatment. Current diagnostic methods often require invasive procedures, such as brain surgery, which can be both risky and burdensome for patients. This research addresses the critical need for non-invasive and accurate brain tumor classification through the development of a deep learning system. The proposed system leverages computational techniques, primarily employing deep learning and machine learning methods, to analyze MRI images for the early and precise identification of glioma, meningioma, and pituitary gland tumors, as well as the classification of healthy brains.

2. PROBLEM STATEMENT

The complexity of brain tumors, coupled with the invasive nature of traditional detection methods, underscores the urgency for alternative diagnostic approaches. Early detection is paramount for timely and effective treatment. The proposed deep learning system aims to provide a solution by utilizing state-of-the-art techniques to classify brain tumors from MRI data. By automating the identification and categorization process, the system seeks to eliminate the need for invasive procedures, facilitating quicker and less intrusive diagnoses.

3. RELATED WORK

Previous research in brain tumor classification predominantly relied on traditional machine learning approaches, often involving manual feature extraction from medical images. However, these methods faced limitations due to the intricate nature of brain tumors and the scarcity of labeled datasets. The proposed work distinguishes itself by adopting deep learning methodologies, specifically focusing on Convolutional Neural Networks (CNNs) for automatic feature extraction. Transfer learning is explored through the fine-tuning of pre-trained models like ResNet, VGG, or Inception, harnessing knowledge from vast datasets like ImageNet. Attention mechanisms are employed to highlight crucial regions in MRI images, enhancing the system's interpretability. Additionally, Generative Adversarial Networks (GANs) are utilized for data augmentation and image quality enhancement, further improving the robustness and performance of the classification system. This comprehensive approach aims to overcome the limitations of previous methods and contribute to the development of a non-invasive, accurate brain tumor classification system.

4. DATASET AND FEATURE

The data utilized in this project were gathered from Kaggle, specifically Brain Tumor Classification (MRI). This dataset consists of a set of different types of tumor MRI images generated through the scans examined by the radiologist over a period of time. The tumor MRI images are already split into training and testing folders and each folder has four subfolders with MRIs of respective tumor classes. There are totally four different classes of tumors in their corresponding folders which are glioma tumor, meningioma tumor, no tumor and pituitary tumor. Each folder contains 1~1000 tumor MRI images and the total number of the images is 3264. Each tumor MRI image is a black and white color with different sizes (Figure 1).

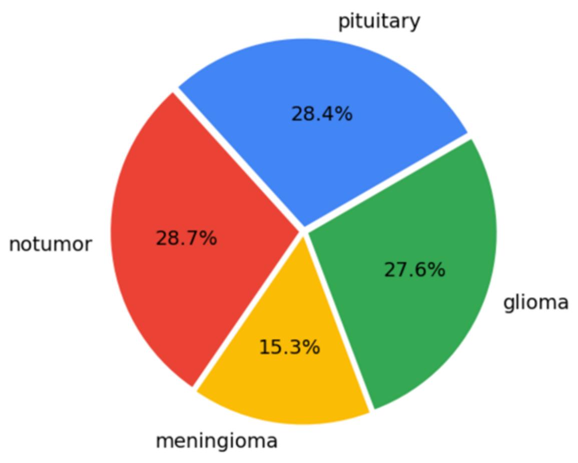


Figure 1: Overview of four classes in dataset

5. IMPLEMENTATION

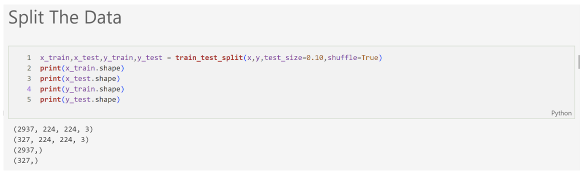
In this part, it discusses how to implement data preprocessing, data set segmentation, classification, train and test.

5.1 Data Preprocessing

First, the images in the data set are in different sizes. When importing the data set, all the images were reshaped to a uniform resolution(224x224). The function resize requires the images imported by open function and force to be set to the size equal to 224x224. By doing so, convolutional neural networks can expect input data to be in the correct format and ensure that the data adheres to the expected format for the model to process it correctly. After rescaling, we will convert the updated image into matrix by function array from numpy for supporting efficient computation in the context of CNNs.

5.1.1 Splitting of training, and testing data

The split_data function serves to partition the entire data set into training and test sets, allocating 90%, 10% of the data, respectively (Figure 2). The function begins by generating a list of labels used for labeling each target corresponding to the input data, which represents output which is a sequence of integers that labels each image. These integers are sorted based on random seeds, which enables the random composition of the two data sets (train and test) for each run of the function. After generating the set of lists, the split_data function proceeds to allocate memory space based on the size of each data set (i.e., train and test). The reason we split the model is that we need to ensure the model's ability to generalize to new, unseen data.



Split The Data

```python
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.10,shuffle=True)
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(2937, 224, 224, 3)
(327, 224, 224, 3)
(2937,)
(327,)
```

Figure 2: Splitting of training, and testing data

5.2 Model Selection

5.2.1 Convolutional Neural Networks (CNNs)

CNNs are the most widely used deep learning architecture for image classification tasks, including brain tumor MRI classification (Figure 3). They are adept at automatically learning relevant features from the input images. At first, we use a two-dimensional convolutional layer to reshape the input by setting the kernel size to 3x3 and use Relu as our activation function for feature extraction. Then, we continue to use function max-pooling to enhance the extraction effect by setting the pool_size to 2x2. Next, we use function dropout to do regularization and set 0.25. By doing so, we can prevent overfitting and improve the generalization ability of the model. But now, the dimension of the output is multidimensional. Therefore, we use the function flatten to convert the multi-dimensional tensor into a one-dimensional tensor. After that we can connect the next layer which is the dense layer by the function Dense which connects each neuron in the previous layer to every neuron in the subsequent layer. At last, we use the function dropout and set it as 0.5 to regularize the output again.

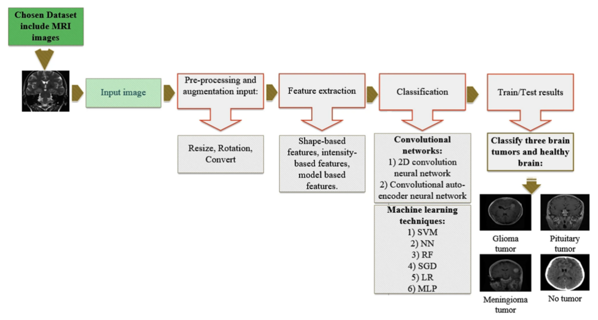

Chosen Dataset include MRI images
Input image
Pre-processing and augmentation input:
Feature extraction
Classification
Train/Test results
Resize, Rotation, Convert
Shape-based features, intensity-based features, model based features.
Convolutional networks:
1) 2D convolution neural network
2) Convolutional auto-encoder neural network
Machine learning techniques:
1) SVM
2) NN
3) RF
4) SGD
5) LR
6) MLP
Classify three brain tumors and healthy brain:
Glioma tumor
Pituitary tumor
Meningioma tumor
No tumor

Figure 3: Brain Tumor Classification by Using CNN

5.2.2 Fine Tuning Pretrained Model

Fine-tuning is a strategy employed with pre-trained models to adapt them to the characteristics of a new dataset without the necessity of training the model from the ground up. This approach significantly reduces training time. During fine-tuning, specific layers of the model are retrained by updating their weights, aligning them with the features present in the new dataset. Typically, the higher layers in convolutional networks exhibit greater specialization, capturing intricate features specific to the original dataset. In contrast, lower layers learn more generic and broadly applicable features. As we move up the network, the features become progressively tailored to the nuances of the dataset on which the model is being trained.
The primary objective of fine-tuning is to leverage the specialized features acquired during the pre-training phase and seamlessly integrate them with the nuances of the new dataset, avoiding the erasure of the generic knowledge previously acquired. This process allows for a more focused adaptation to the distinctive characteristics of the target dataset.

Ultimately, the application of fine-tuning serves the purpose of enhancing model performance, capitalizing on the wealth of knowledge encapsulated in the pre-trained model while tailoring to the specific requirements of the new data.

5.2.2.1 Visual Graphics Group 16 (VGG16)

VGG16 gained popularity for its simplicity, modularity, and the understanding that deeper networks could contribute to better feature learning (Figure 4). At first, we set the model configuration by defining the height and width of the image and the number of classes which are four (glioma tumor, meningioma tumor, no tumor and pituitary tumor). Then we start to load the VGG16 model with weight which is imagenet. The imagenet is a large-scale dataset containing millions of labeled images across thousands of categories, making it a valuable resource for training deep neural networks for image-related tasks. And since we were implementing feature extraction, we also set the include top as false. Next, we add a global average pooling layer with the function GlobalAvergePooling2D so we can compute the average value of each feature map across the entire spatial dimensions. After that we also add the fully-connected layer with the function Dense and set Relu as our activation. In the end, we set a logistic layer for the output with softmax.

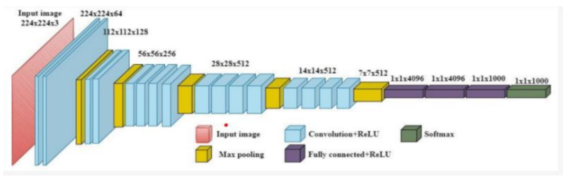


Figure 4: VGG Model Architecture

5.2.2.2 ResNet V2

ResNet V2, known for its residual learning approach, is selected for its capacity to handle deeper architectures. The model's residual blocks and bottleneck structures address the vanishing gradient problem and enhance the learning capability of the network.

5.2.2.3 MobileNet V2

MobileNet V2 is chosen for its lightweight architecture, making it suitable for deployment on resource-constrained devices such as mobile and edge devices. The model incorporates depthwise separable convolutions, inverted residuals, and linear bottlenecks to optimize computational efficiency while maintaining accuracy.

5.2.3 Model Comparison

The fine-tuning of the models involved the addition of a global average pooling layer, followed by fully connected layers serving as the classification layers. This approach was employed across all three models, with a Global Average Pooling layer implemented after the base model. Unlike a standard pooling layer, which makes minor modifications to the images to reduce their size, the global average pooling layer is utilized to either fully or partially replace the fully connected layers that are typically used at the highest levels in convolutional neural networks.
The fine-tuning process includes the addition of a global average pooling layer, which aids in reducing the size of the tensor and accelerates the calculations. The global average pooling layer performs a more intense dimensionality reduction, such that a tensor of dimensions h x w x d is reduced to dimensions 1 x 1 x d. This operation is simple and does not involve any trainable parameters, thereby expediting the training process.
In a classification task, the Softmax function is applied to classify an image with probabilistic values ranging between 0 and 1. The summed-up values of a feature map from the global average pooling layer are fed into the Softmax layer. The activation values are then added up and divided by the sum produced to yield the probability values. Figure 5 provides detailed information about these models.

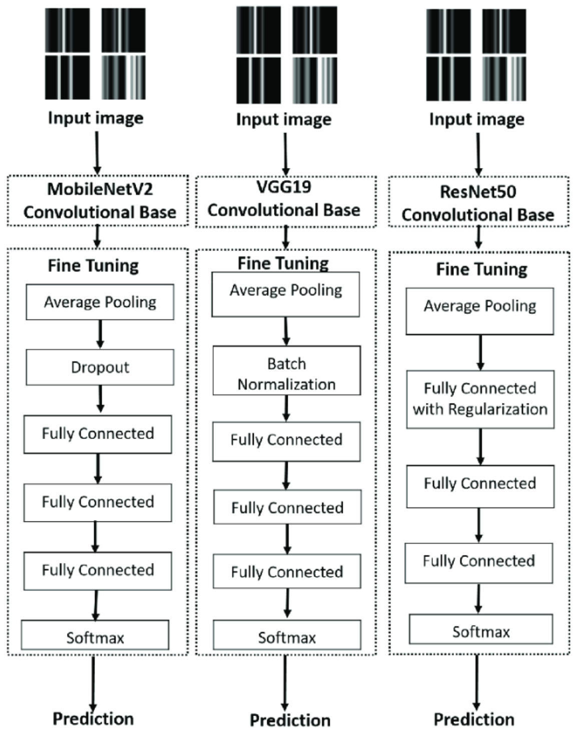


Figure 5: Transfer learning using MobilenetV2, VGG16, and ResNet50.

6. RESULTS

6.1 MobileNet V2 Results

6.1.1 Training Performance

The MobileNet V2 model demonstrated a strong performance during the training phase, which spanned 15 epochs. The model achieved a training accuracy of 94.35%, showcasing its ability to effectively learn and adapt to the complexities of the brain tumor dataset. The corresponding training loss was 0.1854, signifying successful convergence throughout the training process.

6.1.2 Testing Performance

Upon evaluation using the testing set, the MobileNet V2 model continued to exhibit its effectiveness, achieving an impressive testing accuracy of 87.767%. This outcome suggests that the model is capable of generalizing well to unseen data. The testing loss was 0.3233, further reinforcing the model's proficiency in making accurate predictions while minimizing errors.

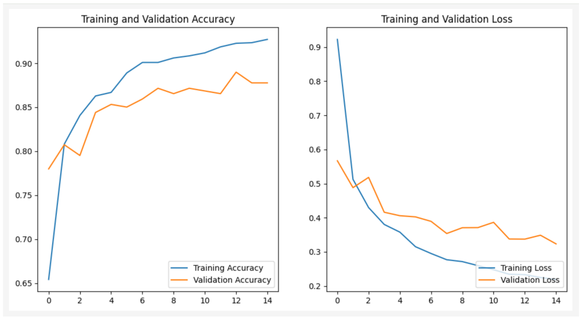


Figure 6:Training and Testing Performance (Loss and Accuracy)

6.2 ResNet V2 Results

6.2.1 Training Performance

The ResNet V2 model, characterized by its deep architecture of 152 layers, displayed remarkable training accuracy, reaching 84.47% after just 4 epochs. The training loss was 0.4136, highlighting the model's capacity to manage complex relationships within the dataset.

6.2.2 Testing Performance

The testing phase validated the robustness of the ResNet V2 model, which achieved a testing accuracy of 79.51%. The testing loss, 0.52051, suggested minimal overfitting, underscoring the model's ability to generalize effectively to new data.

6.3 EfficientNet Results

6.3.1 Training Performance

EfficientNet, renowned for its computational efficiency, exhibited solid training performance. The model attained a training accuracy of 86.58%, and the corresponding training loss was 0.3635.

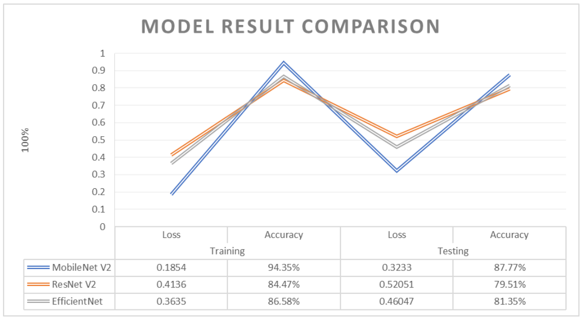
6.3.2 Testing Performance

The testing accuracy for EfficientNet was recorded at 81.35%, reaffirming its effectiveness in making accurate predictions on the testing set. The testing loss, 0.46047, further substantiated the model's ability to generalize well.

6.4 Model Result Comparison

A comparison across the three models revealed nuanced strengths and weaknesses. MobileNet V2 demonstrated efficiency in both training and testing, achieving competitive accuracy with fewer parameters. ResNet V2, with its deep architecture, excelled in capturing intricate features, while EfficientNet struck a balance between accuracy and computational efficiency.

Figure 7 provides detailed information about these models.



	Training		Testing	
	Loss	Accuracy	Loss	Accuracy
MobileNet V2	0.1854	94.35%	0.3233	87.77%
ResNet V2	0.4136	84.47%	0.52051	79.51%
EfficientNet	0.3635	86.58%	0.46047	81.35%

Figure 7: Model Result Comparison

7. CONCLUSION AND FUTURE WORK

7.1 Conclusion

In conclusion, this project successfully investigated the application of deep learning models, specifically MobileNet V2, ResNet V2, and EfficientNet, for the classification of brain tumors. The models exhibited significant accuracy on both training and testing datasets, demonstrating their potential applicability in real-world medical scenarios. The findings suggest that while MobileNet V2 is computationally efficient, ResNet V2 and EfficientNet provide deeper insights into complex image features. The performance of each model should be evaluated in the context of specific application requirements.

7.2 Future Work

Looking ahead, several opportunities for improvement and exploration emerge:

7.2.1 Dataset Expansion

Enhancing the dataset by incorporating a diverse range of brain tumor images could improve model generalization. Additional data can enable models to learn more intricate patterns and enhance their ability to handle a variety of medical scenarios.

7.2.2 Fine-Tuning

Fine-tuning the models based on domain-specific requirements and feedback from medical professionals could lead to enhanced performance. Adjusting hyperparameters and retraining models with a focus on specific classes may further improve accuracy.

7.2.3 Real-World Applications

Exploring the integration of the developed models into real-world healthcare scenarios is a critical next step. Collaborating with medical professionals for model validation and deployment in diagnostic tools presents a promising opportunity.

ACKNOWLEDGMENTS

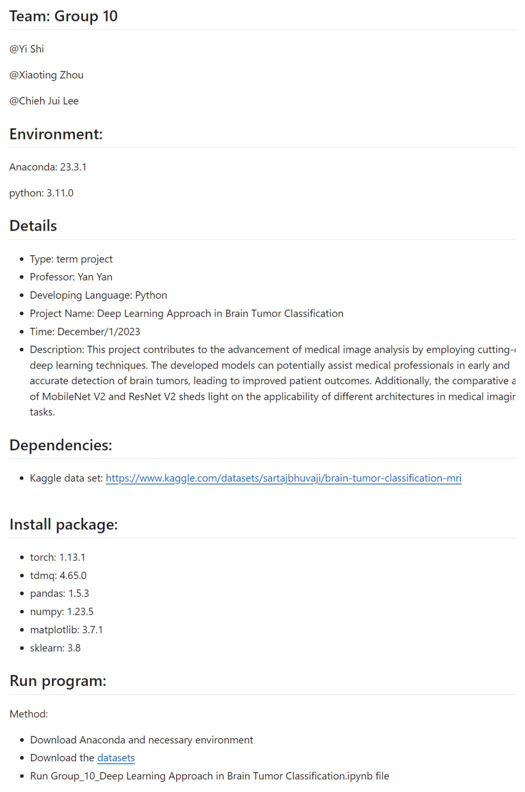
We express our profound gratitude to Professor Yan Yan for his exceptional instruction in the CS577 course on Deep Learning, offered at the Department of Computer Science at the Illinois Institute of Technology. His expert guidance enabled us to gain an in-depth understanding of deep learning algorithms, along with the practical application of their foundational principles. The knowledge and skills we acquired in this field, under Professor Yan's tutelage, have been indispensable. His expertise has significantly enriched our learning experience and deepened our comprehension in this domain.

REFERENCES

A. WEBSITE LINKS

- https://www.kaggle.com/datasets/sartajbhuvaji/brain-tumor-classification-mri
- https://arxiv.org/abs/1602.07261
- https://www.nature.com/articles/s41598-021-90428-8
- https://doi.org/10.1016/j.cmpb.2020.105809 (2020).
- https://doi.org/10.1016/j.neuroimage.2020.117026 (2020).
- https://doi.org/10.3322/caac.20069 (2010).

B. README FILE



Team: Group 10

@Yi Shi

@Xiaoting Zhou

@Chieh Jui Lee

Environment:

Anaconda: 23.3.1

python: 3.11.0

Details

- Type: term project
- Professor: Yan Yan
- Developing Language: Python
- Project Name: Deep Learning Approach in Brain Tumor Classification
- Time: December/1/2023
- Description: This project contributes to the advancement of medical image analysis by employing cutting-(deep learning techniques. The developed models can potentially assist medical professionals in early and accurate detection of brain tumors, leading to improved patient outcomes. Additionally, the comparative a of MobileNet V2 and ResNet V2 sheds light on the applicability of different architectures in medical imagir tasks.

Dependencies:

- Kaggle data set: https://www.kaggle.com/datasets/sartajbhuvaji/brain-tumor-classification-mri

Install package:

- torch: 1.13.1
- tdmq: 4.65.0
- pandas: 1.5.3
- numpy: 1.23.5
- matplotlib: 3.7.1
- sklearn: 3.8

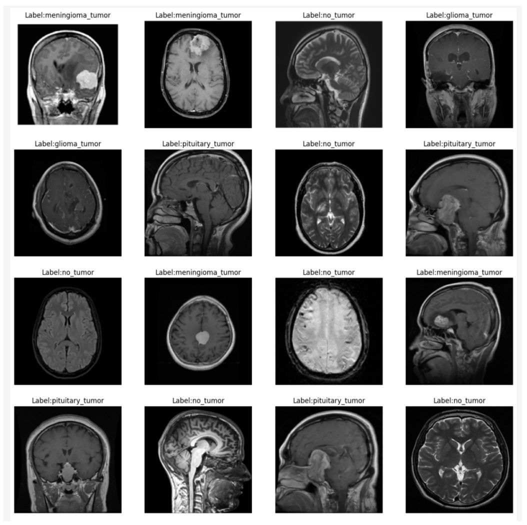
Run program:

Method:

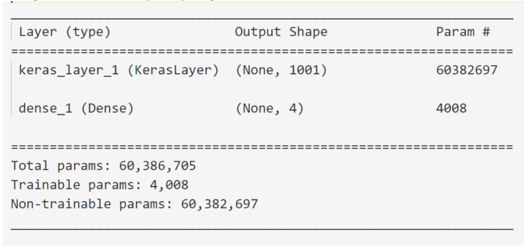
- Download Anaconda and necessary environment
- Download the datasets
- Run Group_10_Deep Learning Approach in Brain Tumor Classification.ipynb file

Example:

1. Train Data Image Visualization



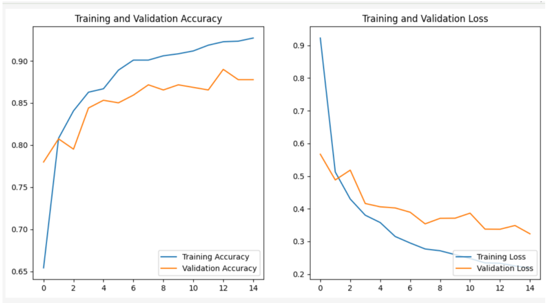
2. Training and Validation Accuracy Plot By Using MobileNet V2 Pre-trained Model



```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 keras_layer_1 (KerasLayer)  (None, 1001)              60382697

 dense_1 (Dense)             (None, 4)                 4008

=================================================================
Total params: 60,386,705
Trainable params: 4,008
Non-trainable params: 60,382,697
_________________________________________________________________
```



C. SOURCE CODE

[Group_10_Deep Learning Approach in Brain Tumor Classification.ipynb]