Final Report: Utilizing Deep Learning Models for Brain Tumor Detection

Alper Demir
Electrical and Electronics Engineering
Ankara, Turkey
alper.demir@ug.bilkent.edu.tr

Atahan İyiekici
Electrical and Electronics Engineering
Ankara, Turkey
atahan.iyiekici@ug.bilkent.edu.tr

***Abstract*—This study shows the use of deep learning models in the detection of brain tumors, and evaluates their accuracy and efficiency compared to traditional methods. Results suggest that deep learning models can effectively identify brain tumors with high accuracy and can potentially improve diagnostic outcomes.**

***Keywords*—Brain tumor, MRI images, deep learning, transfer learning, CNN, classification, ResNet18, EffNet.**

I. INTRODUCTION

The use of magnetic resonance imaging (MRI) has completely changed how we detect and treat cancers among other medical disorders [1]. However, it can take time and be subjective to interpret MRI pictures, which might result in diagnosis errors. The correct detection and classification of cancers from MRI scans can be assisted by deep learning algorithms, offering a more efficient and objective method of diagnosis.

For a more accurate diagnosis and customized treatment strategies, deep learning algorithms can be trained to recognize tiny changes in tumor features that might not be obvious to the human eye [2]. The application of deep learning in MRI tumor detection is becoming more crucial as the need for precise and effective medical imaging continues to rise. By giving doctors an effective instrument to enhance patient outcomes and possibly save lives, this technology has the potential to revolutionize how we approach cancer diagnosis and treatment [3].

A brain tumor is a mass or abnormal growth of cells in the brain that can be cancerous or noncancerous [4]. Since they have the potential to harm the nervous system and can be fatal, early detection of brain tumors is critical. Therefore, the utilization of deep learning algorithms should be necessary.

For this project, a dataset consists of 3264 labeled brain MRI images categorized in 4 classes, Glioma, Meningioma, Pituitary, and no tumor, will be used [5]. 2611 of them will be allocated for training, 326 and 327 of them will be allocated for validation and test respectively. For models, Convolutional Neural Networks (CNN), pre-trained and not pre-trained versions of ResNet-18, and pre-trained and not pre-trained versions of EffNet will be utilized.

II. LITERATURE REVIEW

To find suitable classification methods for this project, performing a literature review for related concepts were needed. After the research, the models that are mentioned before were chosen.

A. CNN:

Convolutional Neural Networks (CNNs) are specialized Deep Neural Networks used commonly for computer vision and classification tasks. This network model consists of 4 parts: Convolutional layer, Pooling layer, Activation function, and Fully-connected layer. [6]

Convolutional layers are the first layers and the main block of the whole model. They contain learnable kernels to detect particular features or patterns in the input image. Pooling layer is utilized to reduce the complexity of the model and they are located between convolution layers. Max pooling and average pooling are the most common ones. The activation function determines whether a neuron should be activated or not, which decides the following network is useful or not. ReLu, Sigmoid, Tanh, and Leaky ReLu are the most common functions. The fully connected layer is responsible for the connections of the neurons.

B. ResNet18

ResNet18 is a 72-layer architecture convolutional neural network [7]. The model has 18 layers, including convolutional and pooling layers. These connections allow easy data transfer between layers, also preventing the issue of disappearing gradients and allowing for the training of deeper networks [7]. ResNet18 is widely used for computer vision applications, so it was selected for these reasons.

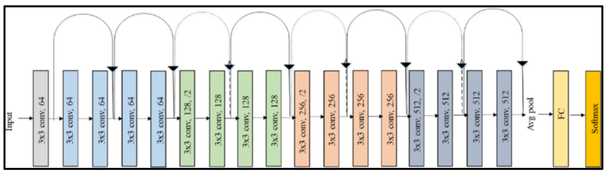


Figure 1: ResNet18 Architecture [8].

C. EffNet:

EffNet is also a member of the CNN family, and it was designed to increase the overall performance and the balance of the neural network [9]. Its major feature is to be able to scale depth, width, and resolution with compound coefficient such that creates balance [10]. This compounding effect also increase the efficiency of the model. EffNet also achieved the state-of-art performance on several benchmark test, CIFAR-100: 91.7%, Flowers: 98.8% etc., which makes it very reliable for computer vision tasks [10].

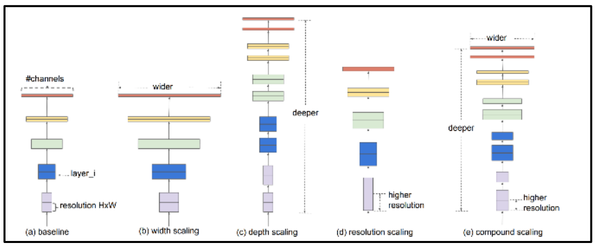


Figure 2: Model Scaling of EffNet versus other techniques [10].

III. IMPLEMENTATION:

As it is mentioned, the dataset for MRI images of brain is taken from Kaggle [5]. After we inspected the dataset, some alterations needed to utilize it. After the pre-processing, we have implemented the models as follows: CNN, ResNet18 (both pre-trained and not pre-trained), and EffNet (both pre-trained and not pre-trained). For training and testing part, each model was trained with randomly selected dataset with 60 epochs.

A. Data Preprocessing:

As it was mentioned before, dataset contains 3264 labeled brain MRI images categorized in 4 classes, Glioma, Meningioma, Pituitary, and no tumor. It was split into two for as test and train dataset. Since all of the images are greyscale, we have only performed resizing to 150x150, normalization, and dataset re-distribution.

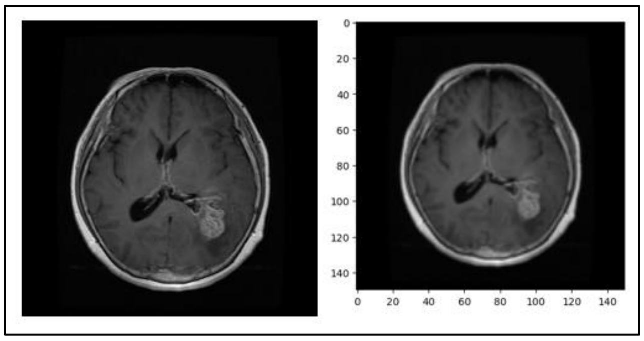


Figure 3: Original image vs pre-processed image.

To distribute even and add randomizing effect, we have combined them, reshuffled them, and then split into 3 for training, validation, and test dataset.

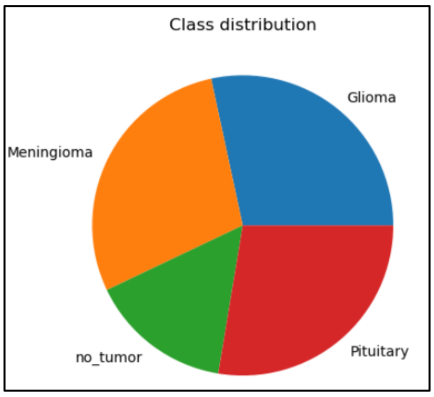


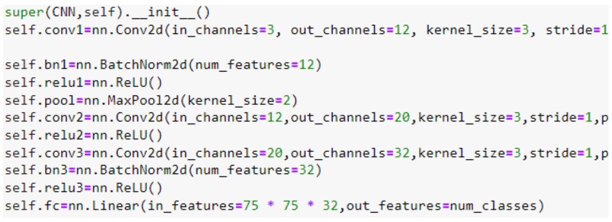
Figure 4: Class Distribution.

B. Progress Results:

In this part, the implementations that have done in the first stage, without any improvement or alterations, are shown. For this part, the hyper parameters were determined as follows: 60 epochs, 64 for batch values, 0.01 for learning rate, 0.7 for momentum, and 0.0005 for weight decay. Additionally, Stochastic Gradient Descent with momentum (SGD) and Cross Entropy Loss functions have used. The sub-parts of this section are as follows: CNN, ResNet-18, and EffNet.

1. CNN Implementation:

We have built our own CNN model with the parameters and layers as follows:



```
super(CNN,self).__init__()
self.conv1=nn.Conv2d(in_channels=3, out_channels=12, kernel_size=3, stride=1

self.bn1=nn.BatchNorm2d(num_features=12)
self.relu1=nn.ReLU()
self.pool=nn.MaxPool2d(kernel_size=2)
self.conv2=nn.Conv2d(in_channels=12,out_channels=20,kernel_size=3,stride=1,p
self.relu2=nn.ReLU()
self.conv3=nn.Conv2d(in_channels=20,out_channels=32,kernel_size=3,stride=1,p
self.bn3=nn.BatchNorm2d(num_features=32)
self.relu3=nn.ReLU()
self.fc=nn.Linear(in_features=75 * 75 * 32,out_features=num_classes)
```

Figure 5: CNN architecture.

The model consists of 2 Batch Normalization layers, 2 Convolutional layers, 1 Max Pooling layer, 3 ReLu Activation Function layers, and 1 Linear Output layer.

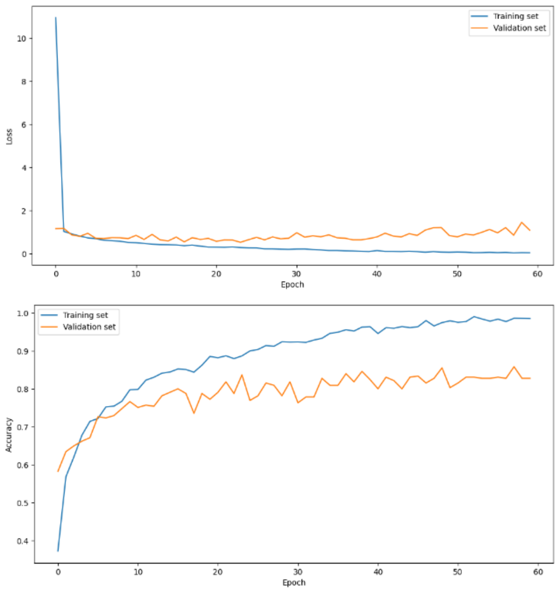


Figure 6: Our CNN Model Training Curves.

The CNN has trained with the training parameters previously mentioned, and the test process has done. For CNN test results, test accuracy is about 82.22%, the F score is 0.8155, precision is 0.85139, and the recall is 0.825214.

2. ResNet18 Implementation:

For ResNet18, we have decided to perform two versions of the model, pre-trained one, and not pre-trained one. Both pre-trained and not pre-trained models have trained with the training parameters previously mentioned, and the test processes have done.

For pre-trained version, test accuracy is about 81.6%, the F score is 0.80159, precision is 0.83035, and the recall is 0.81042.

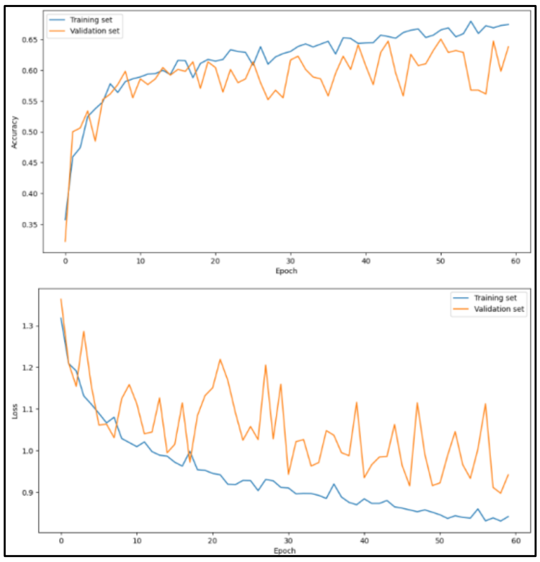


Figure 7: Not Pre-Trained ResNet-18 Training Curves.

For not pre-trained version, test accuracy is about 66.55%, the F score is 0.6358, precision is 0.6917, and the recall is 0.6304.

3. EffNet Implementation:

As in ResNet18 part, we have tested two different versions of the EffNet model: pre-trained one, and not pre-trained one. Here are the results for both versions:

For test part, test accuracy is about 85.76%, the F score is 0.8468, precision is 0.8481, and the recall is 0.8584.

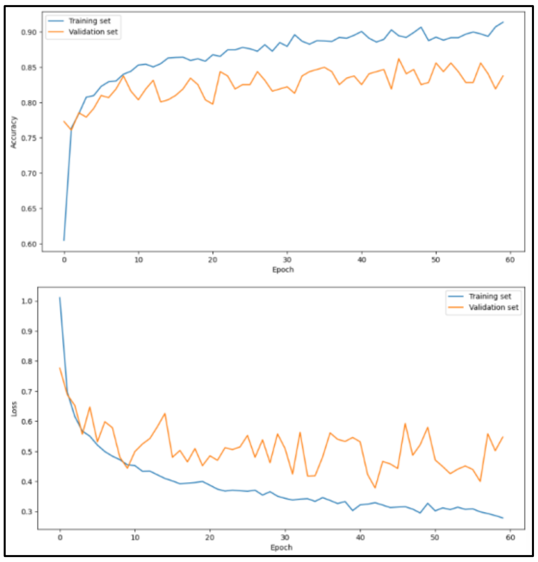


Figure 8: EffNet not pre-trained version training results.

For test part, test accuracy is about 89.14%, the F score is 0.8334, precision is 0.8742, and the recall is 0.8573.

4. Progress Summary:

The progress test results are summarized in the table as follows:

Table 1:Initial Test Results for Models.

MODEL	Accuracy	F1-Score	Precision	Recall
CNN	0.8222	0.816	0.85139	0.8252
Pre-Trained ResNet-18	0.816	0.802	0.83035	0.8104
Pre-Trained EffNet	0.8576	0.847	0.8481	0.8584
Not Pre-Trained ResNet-18	0.6655	0.635	0.6917	0.6304
Not Pre-Trained EffNet	0.8914	0.833	0.8742	0.8573

In overall, initial results are satisfactory considering the CNN performances on image classification tasks. Not pre-trained EffNet performed the best. On the other hand, not pre-trained ResNet-18 performed worst. Improvements and alterations have done to optimize the model performances, which are mentioned in the next section.

IV. IMPROVEMENTS:

Considering the results that we have obtained; we have decided to alter or improve several sections.

As it was in the mention in the dataset part, classes are not evenly distributed. No_tumor class has 400 less images. To solve this issue, data augmentation algorithm, which creates new data by taking symmetries of the original images, has been implemented. After the augmentation, the no tumor class size has increased to 895. Original image and augmented image can be seen on the figure below.

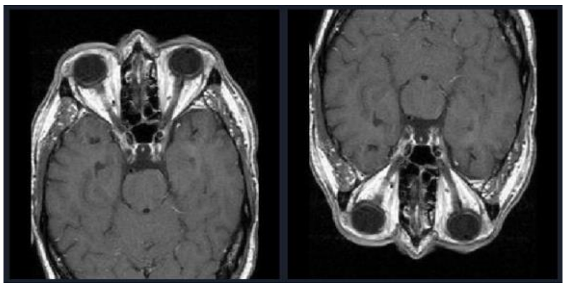


Figure 9: Original (left) and augmented (right) images.

Additionally, the optimizer has changed from SGD to Adam optimizer to see if it affects the results. Lastly, several training parameters have changed and tested to find the optimal values. To find the optimal values numerous test pairs were done, but only most considered results were shown.

First of all, epoch value has remained same as 60. Tests were done with the values 40,50,60,70, and 100. Values below 40 couldn't trained enough, also after the epoch 60's, the training curves started to saturated too much, which is unnecessary to waste training time.

Secondly, the batch sizes have changed. The default value was 64. 32 and 128 batch sizes have tested. The training time for batch size 32 increased triple times, yet the test results have not changed significantly. For 128 batch size, the training time has reduced but since it did not converge not enough, the test results are slightly worse than the initial results. Therefore, 64 batch size chosen.

Additionally, the input size of the images has increased to 320x320, to see the down sampled pixels cause data loss or not. Consequently, the obtained results were similar, mostly worse, to the 150x150 versions, but the training time has increased double.

Table 2: CNN and not pre-trained EffNet test results with 320x320 resolution.

Results	CNN	NPT EffNet
Accuracy	0.794	0.8907
F1-Score	0.785	0.8683
Precision	0.790	0.8677
Recall	0.793	0.8775

Lastly, the learning rate value has changed and tested. When the learning rate has changed from 0.01 to 0.05, the training procedure malfunctioned. The test results are on the following table.

Table 3:Test results on CNN with learning rate= 0.05

Accuracy	0.285
F1-Score	0.102
Precision	0.814
Recall	0.25

After several tests with varying values, it was decided at 0.01 which was the default value at the initial stage.

Consequently, the training parameters' values stayed same, data augmentation has been implemented, and optimizer has changed to Adam.

V. FINAL RESULTS:

A. CNN Implementation:

Our CNN model has been trained with almost the same training parameters previously mentioned in part B, only changes are there is no weights decay and momentum parameters because of the optimizer change.

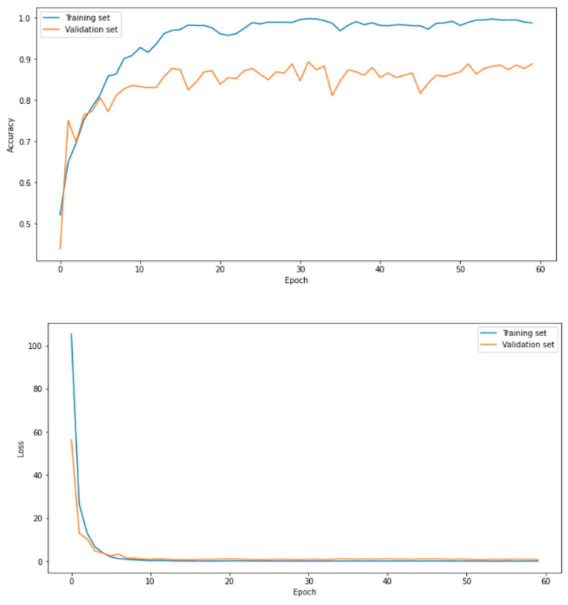
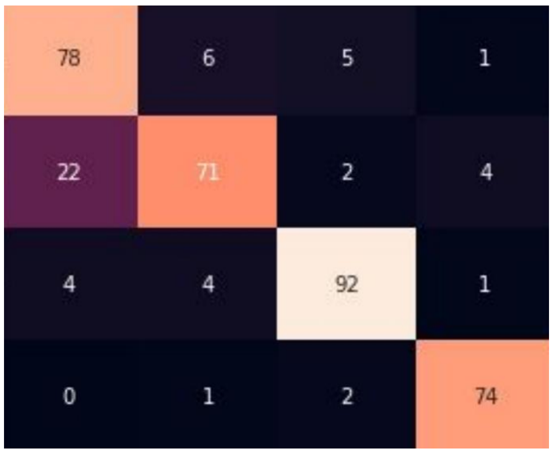


Figure 10: Our CNN Model Training Curves.

For our CNN test results, test accuracy is about 86.4%, the F score is 0.857, precision is 0.861, and the recall is 0.864. The confusion matrix is on the following figure:



78	6	5	1
22	71	2	4
4	4	92	1
0	1	2	74

Figure 11: Confusion Matrix of CNN.

B. ResNet18 Implementation:

For ResNet18, both pre-trained and not pre-trained models have been trained with the same training parameters.

For the pre-trained version, test accuracy is about 83.1%, the F score is 0.831, precision is 0.835, and the recall is 0.846.

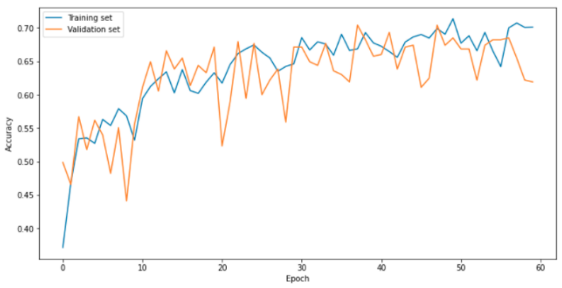


Figure 12: Not pre-trained EffNet Training Accuracy Curve.

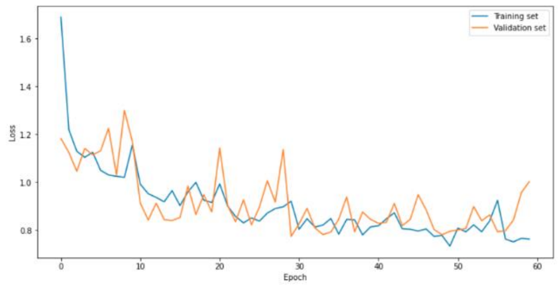


Figure 13: Not Pre-Trained ResNet-18 Loss Curve.

For not pre-trained version, test accuracy is about 67.3%, the F score is 0.650, precision is 0.657, and the recall is 0.676.

C. EffNet Implementation:

For EffNet model: pre-trained one, and not pre-trained one. Here are the results for both versions:

For the pre-trained EffNet model, test accuracy is about 85.9%, the F score is 0.852, precision is 0.862, and the recall is 0.858.

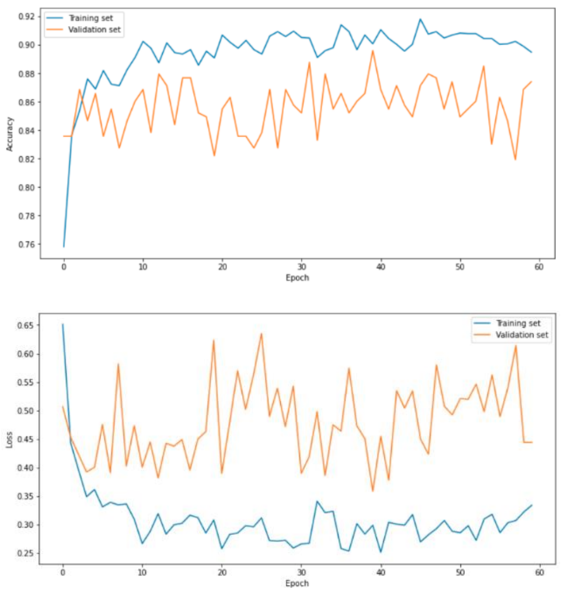


Figure 14: EffNet not pre-trained version training results.

For the not-pre-trained EffNet model, test accuracy is about 88.3 %, the F score is 0.857, precision is 0.861, and the recall is 0.859.

The final test results are summarized in the table as follows:

Table 4: Final Test Results for Models.

MODEL	Accuracy	F1-Score	Precision	Recall
CNN	0.864	0.857	0.861	0.864
Pre-Trained ResNet-18	0.831	0.831	0.835	0.846
Pre-Trained EffNet	0.859	0.852	0.862	0.858
Not Pre-Trained ResNet-18	**0.673**	0.650	0.657	0.676
Not Pre-Trained EffNet	**0.883**	0.857	0.861	0.859

After the alterations, the final results show an improvement over the initial results, especially, the custom-built CNN model performed admirably after these adjustments. Still, the non-pre-trained EffNet model yields the best outcome and non-pre-trained ResNet-18 produces the worst. The enhancements were accomplished by diligent parameter adjustments, changing the optimizer, and data augmentation to balance the dataset.

VI. CONCLUSION:

In this study, we investigated the effectiveness of several deep learning algorithms in identifying and categorizing brain cancers using MRI data. We found the non-pre-trained EffNet model with SGD optimizer yielded the highest performance, likely due to its advanced features such as compounding scalability. In contrast, the non-pre-trained ResNet-18 model with SGD optimizer performed the least effectively, possibly due to data insufficiency or lack of fit between the model and dataset. After adjustments and improvements, our custom-built CNN model performed better. Future studies can focus on diversifying the dataset and exploring other deep-learning architectures. This study highlights the potential of AI in enhancing cancer diagnosis and treatment.

VII. REFERENCES:

[1] M. Haris *et al.*, "Molecular magnetic resonance imaging in cancer," *Journal of Translational Medicine*, vol. 13, Sep. 2015, doi: https://doi.org/10.1186/s12967-015-0659-x.

[2] Y. Jiang, M. Yang, S. Wang, X. Li, and Y. Sun, "Emerging role of deep learning-based artificial intelligence in tumor pathology," *Cancer Communications*, vol. 40, no. 4, pp. 154–166, Apr. 2020, doi: https://doi.org/10.1002/cac2.12012.

[3] T. Davenport and R. Kalakota, "The potential for artificial intelligence in healthcare," *Future Healthcare Journal*, vol. 6, no. 2, pp. 94–98, Jun. 2019, doi: https://doi.org/10.7861/futurehosp.6-2-94.

[4] "Brain tumor - Symptoms and causes," *Mayo Clinic*. https://www.mayoclinic.org/diseases-conditions/brain-tumor/symptoms-causes/syc-20350084#:~:text=A%20brain%20tumor%20is%20a

[5] "Brain Tumor Classification (MRI)," *www.kaggle.com*. https://www.kaggle.com/datasets/sartajbhuvaji/brain-tumor-classification-mri?select=Training (accessed Apr. 07, 2023).

[6] IBM, "What are Convolutional Neural Networks? | IBM," *www.ibm.com*. https://www.ibm.com/topics/convolutional-neural-networks

[7] "Residual Network - an overview | ScienceDirect Topics," *www.sciencedirect.com*. https://www.sciencedirect.com/topics/computer-science/residual-network

[8] F. Ramzan *et al.*, "A Deep Learning Approach for Automated Diagnosis and Multi-Class Classification of Alzheimer's Disease Stages Using Resting-State fMRI and Residual Neural Networks," *Journal of Medical Systems*, vol. 44, no. 2, Dec. 2019, doi: https://doi.org/10.1007/s10916-019-1475-2.

[9] I. Freeman, L. Roese-Koerner, and A. Kummert, "Effnet: An Efficient Structure for Convolutional Neural Networks," *IEEE Xplore*, Oct. 01, 2018. https://ieeexplore.ieee.org/document/8451339 (accessed Apr. 07, 2023).

[10] "Papers with Code - EfficientNet Explained," *paperswithcode.com*. https://paperswithcode.com/method/efficientnet#:~:text=EfficientNet%20is%20a%20convolutional%20neural