

Illinois Institute of Technology

CS484: Introduction to Machine Learning [Fall 2024]



**Evaluating the Impact of Skip Connections on Deep Neural Networks for MRI-
Based Brain Tumor Classification**

Final Project

Table of Contents

S. No.	Topic
1.	Introduction
2	Related works
3	Proposed Algorithm
3.1	Dataset Used
3.2	Data Preprocessing
3.3	Data Augmentation
3.4	Model Construction
3.5	Diagnosis and Remediation
3.6	Hyper-parameters used
4	Results and Observations
4.a	Confusion Matrix
4.b	Accuracy
4.c	Precision
4.d	Recall
4.e	F1-score
5	Conclusion

Model Design and Training

S. No.	Topic
1.	Plain Networks
2.	ResNet Networks
3.	Hyper-parameter tuning

Evaluating the Impact of Skip Connections on Deep Neural Networks for MRI-Based Brain Tumor Classification

Abstract:

Brain tumor classification is a pivotal challenge in medical imaging, vital for early diagnosis and effective treatment planning. This study evaluates the impact of network architecture on the performance of deep learning models in classifying brain tumors from MRI scans. A dataset of 7,023 images, categorized into four tumor classes (Glioma, Meningioma, No Tumor, and Pituitary), was preprocessed with extensive augmentation techniques. Two types of architectures were explored: plain networks (18, 34, and 40 layers) and residual networks incorporating skip connections. The results highlight that plain networks struggle with vanishing gradients in deeper configurations, while residual networks, leveraging skip connections, demonstrate superior accuracy by facilitating effective gradient flow. Hyperparameter tuning, including dynamic learning rates and gradient clipping, further optimized training stability. The residual networks achieved a test accuracy of 91.76%, emphasizing their robustness in handling complex models. This study underscores the importance of architectural design in deep learning for medical imaging and advocates for larger, more diverse datasets to enhance model generalization.

Key words: Brain tumor classification, Deep Neural Networks, Computer Vision, Skip connections, Vanishing Gradients, Gradient clipping, Medical Imaging.

1. Introduction

Brain tumor classification is a critical task in medical imaging, as it facilitates early diagnosis and effective treatment planning. Magnetic Resonance Imaging (MRI) scans are widely used for detecting brain tumors, offering high-resolution imaging of internal brain structures. Recent advancements in deep learning have enabled the automation of tumor classification, reducing manual errors and accelerating diagnosis. However, the performance of deep learning models largely depends on the architecture and design of the neural networks employed.

This study investigates the impact of network architecture on the classification accuracy of brain tumors using MRI images. A dataset comprising 7,023 images, categorized into four tumor types—Glioma, Meningioma, Pituitary, and No Tumor—is utilized. The research focuses on comparing plain deep networks and residual networks with skip connections. The former suffers from vanishing gradients in deeper layers, while the latter improves gradient flow, leading to enhanced performance. The effectiveness of hyperparameter optimization techniques, such as gradient clipping and dynamic learning rates, is also analyzed. The findings of this study underscore the importance of network design in medical imaging applications and pave the way for future improvements in deep learning for healthcare.

2. Related works

Deep learning has revolutionized medical imaging, enabling automated tumor classification with significant accuracy improvements. Early developments, such as Convolutional Neural Networks (CNNs), demonstrated the potential of deep networks for image recognition tasks [1, 3]. However, increasing network depth often led to challenges such as vanishing gradients, which hindered training and convergence in complex datasets [2, 4].

Residual networks (ResNets) were introduced by He et al. [2] to address these issues. ResNets incorporate skip connections that facilitate gradient flow, enabling the training of deeper

architectures. These networks have since become a standard for complex classification problems, including medical imaging. Simonyan and Zisserman [1] also explored very deep CNNs, emphasizing the significance of network depth in learning intricate features.

Densely Connected Convolutional Networks (DenseNets), proposed by Huang et al. [5], extended the concept of skip connections by connecting each layer to every other layer. While DenseNets achieved state-of-the-art performance on benchmark datasets, their computational overhead remains a challenge for large-scale medical imaging.

Studies have also highlighted the role of data augmentation in improving model generalization. Techniques such as rotation, scaling, and flipping allow models to learn robust features by simulating real-world variability in medical imaging datasets [6]. This, combined with advanced architectures, provides a powerful framework for tasks such as brain tumor classification.

This project builds on these advancements by comparing plain and residual networks for MRI-based tumor classification. In addition, hyperparameter optimization techniques such as gradient clipping and dynamic learning rates are employed to further enhance model performance and stability. The proposed algorithm is outlined in detail in Section 3. Following that, Sections 4 and 5 cover model training and experimentation, as well as the results and discussion, respectively.

3. Proposed Algorithm

The proposed methodology begins with selecting an appropriate MRI dataset with large sample data, followed by data augmentation and preprocessing. The data is then split into training, testing, and validation sets. The processed data is trained using the Plain Network, designed by following the depth scaling method, and the Residual Networks, which incorporate skip connections inherited between the stages in the network. Model training is monitored, and remediation steps are taken to improve training and model performance. Finally, the retrained models are evaluated on the testing set, and their performance is compared based on the selected performance metrics. Figure 1 provides a pictorial overview of the proposed architecture.

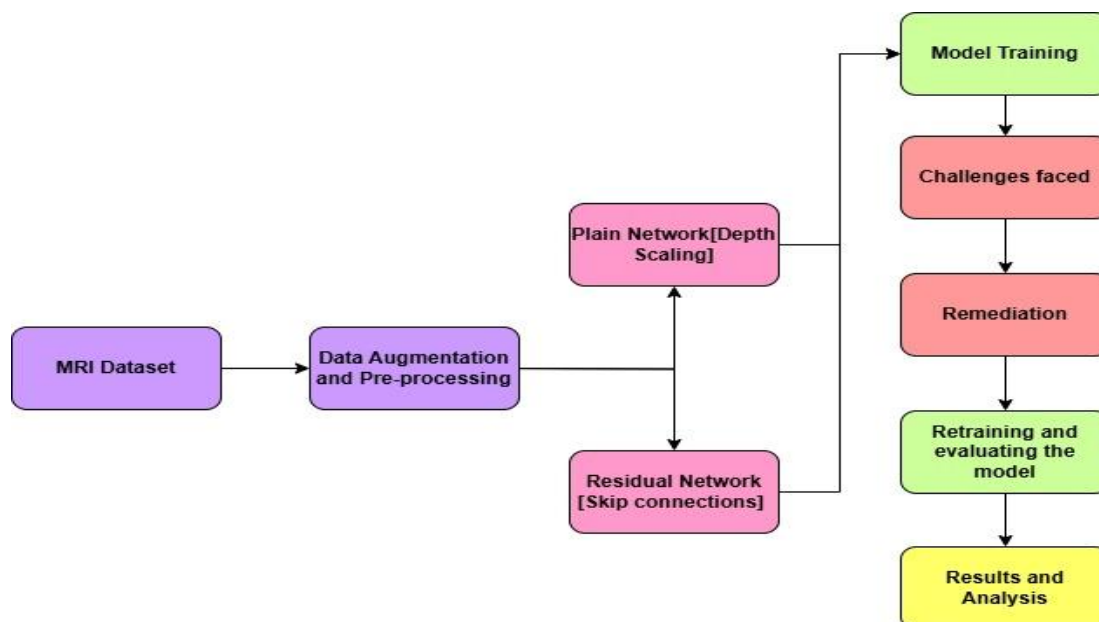


Figure 1. Pictorial overview of the proposed architecture.

3.1 Dataset Used

The dataset used in this study is sourced from Kaggle [7] which is constructed from three sources—Figshare, SARTAJ, and Br35H—offering a total of 7,023 MRI images classified into four categories: Glioma, Meningioma, No Tumor, and Pituitary. Table 1 provides a detailed overview of the four categories and its distribution in Train and Test sets. The distribution suggests that the dataset is slightly imbalanced with “No Tumor” class having the largest number of samples.

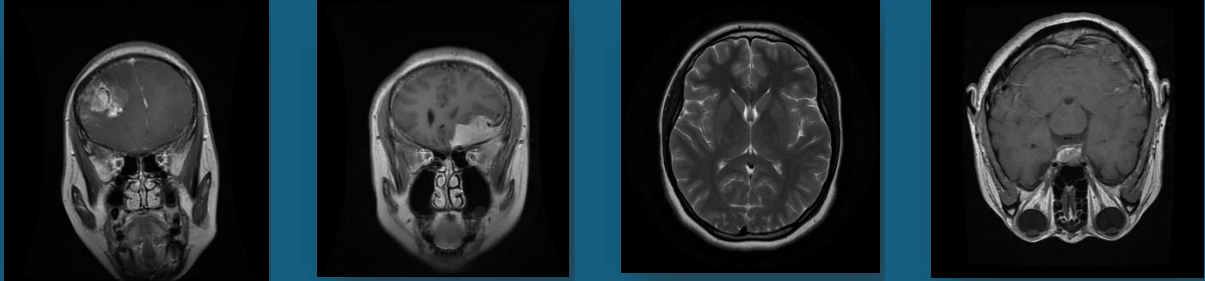
			
Glioma	Meningioma	No Tumor	Pituitary
Number of images in each class in Train Set			
1321	1339	1595	1457
Number of images in each class in Test Set			
300	306	405	300

Table 1. Visual Representation of MRI Dataset.

3.2 Data Preprocessing

To ensure compatibility with deep learning models and enhance training efficiency, all images in the dataset were resized to a fixed dimension of 224 × 224 pixels. This resizing standardizes the input dimensions, allowing seamless integration into the model pipeline while reducing computational overhead. Additionally, pixel values were normalized by rescaling them to the range [0,1] from their original range of [0,255] using the parameter rescale=1./255. This normalization helps reduce computational cost, as a large number of multiplications can exhaust processing units and improves stability by ensuring consistent input scales. To enhance model training, I split the training set into training and validation subsets, allowing me to adjust the learning rate during training by monitoring the validation set metrics. The final data distribution among the training, validation, and test sets is 4,571, 1,141, and 1,311 samples, respectively.

3.3 Data Augmentation

Data augmentation techniques were employed to artificially expand the dataset and introduce variability, thus mitigating the risks of overfitting and improving the generalizability of the model. The augmentation parameters and their corresponding benefits are detailed below:

- Rotation (rotation_range=10): Randomly rotates images within a range of +/- 10 degrees. This augmentation improves the model's ability to handle slight rotations, making it invariant to orientation changes.
- Translation (width_shift_range=0.2, height_shift_range=0.2): Introduces random horizontal and vertical shifts of up to 20% of the image width and height, respectively. This simulates real-world scenarios where objects may not always appear perfectly centered in the frame.
- Shearing (shear_range=0.2): Applies a shearing transformation, skewing the image in one direction. This augmentation helps the model recognize features even under distorted conditions.
- Zooming (zoom_range=0.2): Randomly zooms in or out by up to 20%. This allows the model to learn object features under varying scales, accommodating differences in object sizes.
- Horizontal Flipping (horizontal_flip=True): Introduces horizontal flips with a 50% probability. This is particularly beneficial for datasets with left-right symmetrical objects, such as medical imaging or natural scenes.
- Fill Mode (fill_mode='nearest'): During transformations like rotation or shifting, empty areas introduced in the image are filled using the nearest pixel values. This avoids undefined or blank regions, maintaining the visual integrity of the augmented images.

The combination of preprocessing and augmentation steps not only prepares the dataset for deep learning models but also enhances the robustness of the model by simulating various real-world variations in the training data.

3.4 Model Construction

The field of very deep neural networks has revolutionized computer vision, enabling remarkable advancements in tasks such as image classification, object detection, and medical imaging. However, the adoption of very deep architectures is hindered by significant challenges, most notably the vanishing gradient problem [8, 9, 10].

Vanishing gradients arise as the depth of a neural network increases. During backpropagation, gradients, which represent the error signal used to update network weights, become progressively smaller as they pass through multiple layers. This diminishment leads to near-zero gradients in the initial layers, rendering them ineffective for learning meaningful representations [8, 9]. Consequently, early layers stagnate, impeding the overall performance and convergence of the network. The issue is particularly exacerbated in very deep architectures, where the extended path of gradient propagation amplifies this attenuation effect [10].

Addressing the vanishing gradient problem is critical for unlocking the potential of very deep networks in complex domains such as MRI-based brain tumor classification. To mitigate this issue, this study incorporates skip connections within residual networks, a design that facilitates efficient gradient flow by bypassing intermediate layers. Skip connections alleviate the bottleneck caused by vanishing gradients, ensuring that all layers, including the initial ones, are effectively trained. This methodological innovation enables the network to harness the depth required for intricate feature extraction without succumbing to the limitations of traditional architectures. To study the effectiveness of the Skip Connections, we built two models namely Plain Network, designed by following the depth scaling method, and the Residual Networks, which incorporate skip connections inherited between the stages in the network. The details of the networks are mentioned below.

a. Plain Network [Depth Scaling]

The Plain Network was designed to analyze the impact of increasing depth on model performance. Three variations were implemented, each differing in the number of layers. The network follows a stage-based architecture, with each stage corresponding to a specific filter size. Within each stage, blocks are stacked sequentially, where each block consists of two convolutional layers with a 3×3 kernel. The configurations for the different depths are described below

18-layer Network:

- Structure: Repetitions per stage: [2,2,2,2]
- Filter Sizes: 64, 128, 256, 512
- Number of Convolutions: $(2+2+2+2) \times 2 = 16$ convolutional layers.

34-layer Network:

- Structure: Repetitions per stage: [3, 4, 6, 3]
- Filter Sizes: 64, 128, 256, 512
- Number of Convolutions: $(3+4+6+3) \times 2 = 32$ convolutional layers.

40-layer Network:

- Structure: Repetitions per stage: [3, 5, 6, 4, 1]
- Filter Sizes: 64, 128, 256, 512, 1024
- Number of Convolutions: $(3+5+6+4+1) \times 2 = 38$ convolutional layers.

In all the above networks there are initial convolution and pooling layers which add up the missing two layers to complete the network.

b. Residual Network [Skip Connections]

The Residual Network (ResNet) builds on the architecture of the Plain Network by incorporating skip connections to improve gradient flow and mitigate the effects of increasing depth.

Skip Connections are applied within each block, following the sequence of two 3 x 3 convolutional layers. These connections introduce the following enhancements:

- Identity Mapping:** By default, the input is directly added to the output of the block, preserving original features and allowing the model to learn only the residuals.
- Downsampling:** When the spatial dimensions or filter size changes (e.g., due to a stride of 2 or an increase in the number of filters), skip connections require adjustments to match the input tensor dimensions to the output tensor dimensions. Downsampling is performed using the following techniques:

1 x 1 Convolution: This operation aligns the number of channels and adjusts spatial dimensions. It ensures that the input tensor is compatible with the block's output dimensions.

Batch Normalization: Applied after the 1 x 1 convolution, this operation stabilizes and scales the transformed input, improving the model's training efficiency and convergence.

We use these two models to train and test on the MRI dataset for performance evaluation.

3.5 Diagnosis and Remediation

During the training of the model, an abnormality was observed where the validation accuracy became NaN. Upon investigation, it was identified that this issue was caused by **exploding gradients**, a common problem in deep neural networks.

Exploding gradients:

When the gradients computed during backpropagation become excessively large, they destabilize the network and cause numerical issues, leading to the exploding gradient problem.

Gradients with excessively large magnitudes cause the weights to change significantly during updates. These unstable updates can push activations into undefined ranges, leading to invalid outputs. When weights or activations become infinite or NaN, the model's predictions lose validity. This invalid state results in NaN values in the computed loss, making metrics like validation accuracy undefined.

To diagnose and mitigate the effects of exploding gradients, the following measures were implemented:

Gradient Clipping: To cap the magnitude of gradients during backpropagation, gradient clipping was employed. This ensures that gradients are limited to a maximum norm, preventing them from growing excessively. We set the cap to 1, so the gradients are limited to a maximum norm of 1.

Reducing the Learning Rate: A high learning rate exacerbates the impact of large gradients. To stabilize training, the learning rate was reduced from 0.001 to 0.00008. Table 2 shows the validation accuracy versus epochs before and after remediation. The post-remediation results indicate that there is no occurrence of NaN in the validation accuracy, which suggests that we have successfully reduced the impact of the exploding gradient problem.

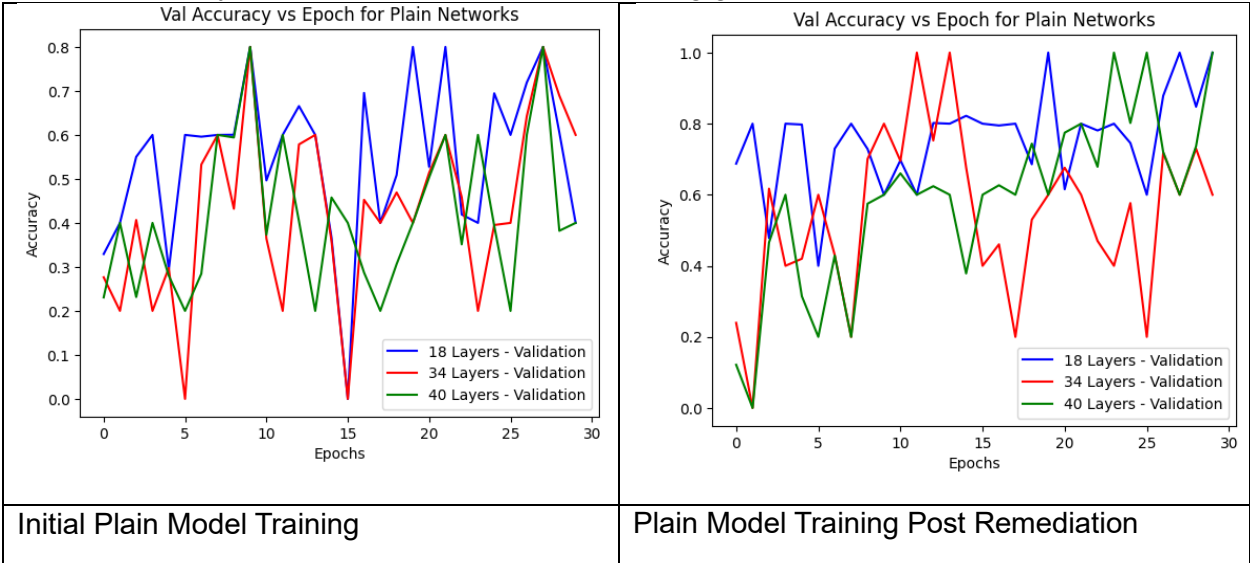


Table 2: Validation accuracy versus epochs before and after remediation

3.6 Hyper-parameters used

To regulate the learning process, a dynamic learning rate strategy was implemented. The initial learning rate was set to 8e-5, and the learning rate was adjusted dynamically based on the validation accuracy. If validation accuracy did not improve for 5 consecutive epochs, the learning rate was reduced by a factor of 0.5. A minimum learning rate of 1e-6 was established to prevent the learning rate from decreasing excessively, thereby avoiding stagnation. This dynamic adjustment aimed to balance convergence speed while maintaining stability in the training process. A batch size of 16 was chosen to strike an optimal balance between memory consumption and training speed, allowing efficient processing of data during model updates. The Adam optimizer was employed due to its adaptive learning rate properties, ensuring effective and efficient weight updates throughout the training process. Training was conducted over 30 epochs, providing sufficient iterations for the model to converge to an optimal solution.

To reduce overfitting and enhance generalization, a dropout rate of 50% was applied, which involved randomly deactivating neurons during each training iteration. For activation functions, ReLU was utilized in the hidden layers for its ability to efficiently handle non-linearities, while softmax was employed in the final layer to enable effective multi-class classification.

The input images were resized to a standardized shape of (224, 224), ensuring uniformity across the dataset and compatibility with the neural network. Categorical Cross-Entropy loss was selected as the loss function, as it is well-suited for multi-class classification tasks, effectively capturing the model's prediction errors for optimization. Table 3 gives a detail overview of the hyper-parameters used while training the neural network.

Hyper-parameter	Parameter Used
Batch Size	16
Optimizer	Adam
Dropout	0.5
Activation Function	Relu & Softmax [only final layer]
Input Image Shape	(224, 224)
Loss Function	Categorical Cross Entropy
Epochs	30

Table 3: Hyper-parameters used in the training phase.

4. Results and Observations

Table 4 shows the results of training and validation accuracy of the 2 models over 30 epochs. In the case of the Train dataset, we can see in the plain network the graph reveals significant instability in training accuracy, with networks of all depths 18, 34, and 40 layers exhibiting notable fluctuations throughout the training process. Accuracy values oscillate dramatically, displaying sudden drops and peaks, which are particularly pronounced in the deeper networks. This erratic training pattern indicates difficulty in maintaining consistent learning progress, with no evident trend of convergence. In contrast, the second graph highlights the improved behavior of residual networks. All network depths demonstrate a more stable and consistent upward trend in accuracy, with initial fluctuations gradually subsiding as training advances. These networks achieve and sustain higher accuracy levels, particularly during the later epochs, showcasing the effectiveness of skip connections in stabilizing the training process and enhancing performance.

In the case Validation dataset, In Plain Networks, the 18-layer network demonstrated moderate stability, albeit with frequent oscillations. The 34-layer network exhibited significant volatility, with dramatic drops in accuracy to 0.2 at multiple points. The 40-layer network displayed the most unstable behavior, with sharp fluctuations ranging from 0.2 to 1.0 accuracy.

In contrast, the Residual Networks showed more consistent validation performance. All three architectures maintained accuracy levels generally between 0.4 and 0.8, experiencing less severe drops compared to their Plain Network counterparts. Based on the accuracy versus epoch, interpreting the performance is not entirely clear. Therefore, to evaluate the performance of our Deep Convolutional Neural Network (DCNN) models, various performance metrics were employed and are thoroughly examined in this section.

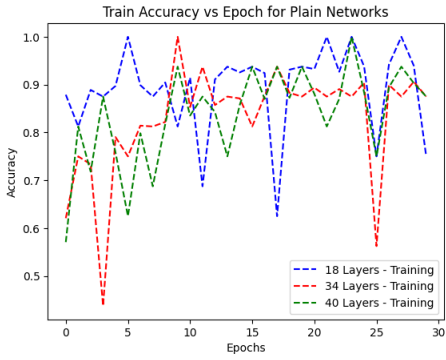
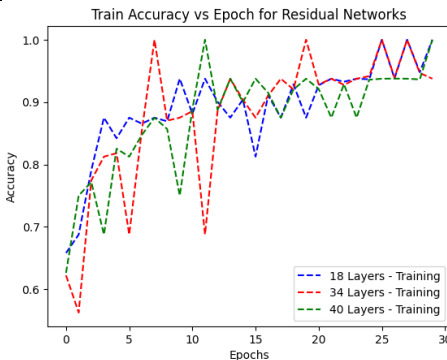
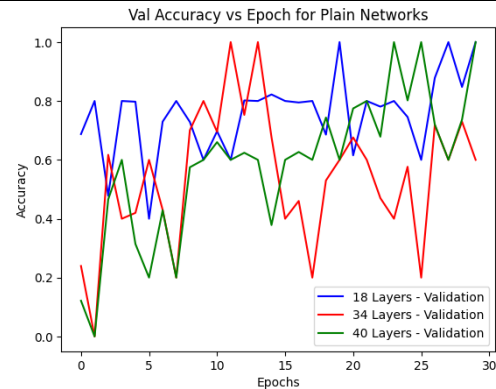
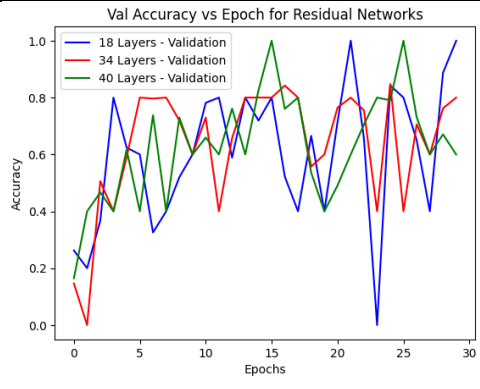
S. No.	Type of graph	Plain Network	Residual Network
1.	Train vs Epoch		
2.	Validation vs Epoch		

Table 4: Training and validation accuracy of the 2 models over 30 epochs

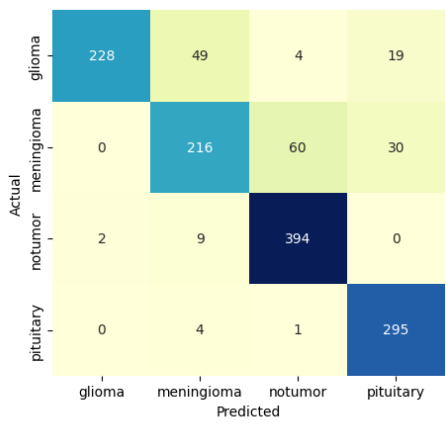
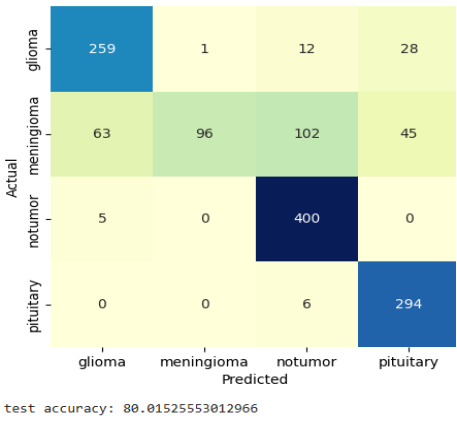
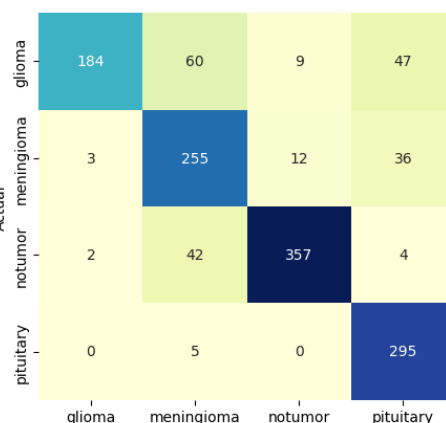
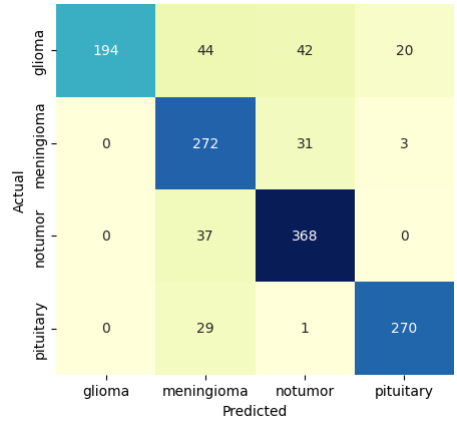
a. Confusion Matrix.

The confusion matrix provides a visual representation of the model's prediction accuracy, utilizing components such as true positive (TP), true negative (TN), false positive (FP), and false negative (FN). TP indicates cases where both the actual and predicted values are correct. For example, in an MRI dataset with four classes namely Glioma, Meningioma, No Tumor, and Pituitary. A TP occurs when the model correctly predicts Glioma for an actual Glioma case. TN refers to instances where both the actual and predicted conditions are negative, such as correctly identifying a case as not being Pituitary when it is actually No Tumor.

FP represents cases where the model's prediction is positive but the actual value is negative. For instance, predicting Meningioma for a case that is actually Glioma would count as an FP. FN accounts for situations where the predicted value is negative, but the actual value is positive. For example, when the model predicts No Tumor for a case that is actually Pituitary.

The confusion matrix helps evaluate the model's performance in accurately classifying different tumor types in MRI datasets. In this explanation, we consider the matrix with the actual classes as the row vector and the predicted classes as the column vector.

The confusion matrix of the test sets for the discussed models, shown in Table 5, suggests that in the case of plain networks, there are many misclassifications as the depth of the neural network increases. This implies that the models struggle to learn effectively due to the vanishing gradient problem as depth increases. In contrast, ResNet models show significant improvement, as the skip connections efficiently pass gradients, enabling optimal training and leading to fewer misclassifications as the depth increases.

S. No	Layer count	Plain Networks	ResNet Networks
1	18	 <p>test accuracy: 86.42257818459191</p>	 <p>test accuracy: 80.01525553012966</p>
2	34	 <p>test accuracy: 83.2189168573608</p>	 <p>test accuracy: 84.21052631578947</p>

3

40

Actual	glioma	127	49	33	91
	meningioma	0	207	60	39
	notumor	2	1	402	0
	pituitary	0	4	1	295
	Predicted	glioma	meningioma	notumor	pituitary

test accuracy: 78.6422578184592

Actual	glioma	265	21	9	5
	meningioma	7	247	47	5
	notumor	5	0	400	0
	pituitary	1	8	0	291
	Predicted	glioma	meningioma	notumor	pituitary

test accuracy: 91.76201372997713

Table 5. Confusion matrix of two types of Deep CNNs on the test set.

To explain and interpret the metrics, I am using the results of the 18-layer version of the Plain Network from Table 6.

b. Accuracy.

Accuracy represents the proportion of correctly predicted instances out of the total instances in the dataset. The 18-layer version of Plain Network has an test accuracy of 86.42% indicates that the model correctly classifies approximately 86 out of every 100 samples. While useful for assessing overall performance, accuracy may not fully capture the model's capability, especially with imbalanced datasets.

c. Precision.

Precision measures the percentage of correctly predicted positive instances out of all instances predicted as positive. With a precision of 87.11%, meaning that when the model predicts a positive outcome (e.g., a specific tumor type), it is correct 87.11% of the time. Precision is crucial when false positives are more detrimental, such as in identifying critical health conditions.

d. Recall.

Recall (or Sensitivity) quantifies the percentage of actual positive instances that are correctly identified by the model. With a recall of 85.55%, the model successfully identifies 85.55% of the actual positive cases, such as correctly detecting Glioma when it is present. Recall is particularly important in scenarios where missing a positive case (false negatives) could have serious consequences.

e. F1-score.

The F1-score is the harmonic mean of Precision and Recall, balancing the trade-off between the two. With a F1-score is 85.71%, indicating a balanced performance in terms of both identifying true positives and minimizing false positives. This metric is especially useful when Precision and Recall hold equal importance in evaluating the model.

The effectiveness of these shortcut layers can also be observed from the precision, accuracy, recall, and F1-score provided in Table 6. The comparative analysis of these metrics reveals distinct patterns between Plain Networks and ResNet architectures. For Plain Networks, the results demonstrate a decline in performance as the network depth increases. Specifically, the

18-layer Plain Network achieved an accuracy of 86.42% and an F1-score of 85.71, whereas the 34-layer and 40-layer variants exhibited deteriorating performance, with accuracy dropping to 83.22% and 78.64%, and F1-scores reducing to 82.33 and 75.71, respectively. This trend highlights the challenges posed by the vanishing gradient problem in deeper networks without additional structural interventions.

In contrast, ResNet architectures showcased improved performance with increasing depth. While the 18-layer ResNet achieved an accuracy of 80.02% and an F1-score of 76.23, the 34-layer and 40-layer variants outperformed their shallower counterpart, achieving accuracies of 84.21% and 91.76% and F1-scores of 83.89 and 91.58, respectively. This improvement underscores the effectiveness of skip connections in mitigating the vanishing gradient problem by enabling better gradient flow and feature propagation in deeper networks.

These findings highlight the critical role of skip connections in stabilizing and enhancing the training of deep networks, as evidenced by the superior performance of ResNet models with increasing depth.

Table 6

S. No.	Layer Count	Accuracy	Precision	Recall	F1-score
Plain Networks					
1	18	86.422	87.11	85.55	85.71
2	34	83.218	84.87	82.79	82.33
3	40	78.642	82.06	76.89	75.71
ResNet Networks					
4	18	80.015	83.80	78.62	76.23
5	34	84.210	86.65	83.60	83.89
6	40	91.762	92.30	91.20	91.58

5. Conclusion

In conclusion, the comparison between Plain Networks and ResNet architectures has demonstrated significant insights into the impact of network depth on model performance. Plain Networks, particularly as their depth increased, struggled with the vanishing gradient problem, leading to greater instability in both training and validation accuracy. This was evidenced by the erratic fluctuations observed in the accuracy graphs, with deeper networks performing worse due to ineffective gradient propagation. On the other hand, ResNet models showed notable improvements as depth increased, thanks to the presence of skip connections that mitigated the vanishing gradient issue. These networks exhibited more stable and consistent training, achieving better performance metrics such as accuracy, precision, recall, and F1-score, especially as depth increased.

The effectiveness of skip connections in ResNet models highlights their crucial role in stabilizing the training process and enhancing learning in deeper networks. This observation underscores the importance of structural innovations like residual connections in overcoming the challenges of training very deep networks.

However, despite the superior performance of ResNet models in terms of test accuracy, training very deep neural networks on a small dataset poses the risk of overfitting. To address this, using pre-trained weights and fine-tuning the model could provide a robust solution, leveraging learned features from larger datasets to improve generalization and prevent overfitting. Furthermore, techniques like gradient clipping and adjusting the learning rate were successfully employed to address the issue of exploding gradients, leading to more stable training and eliminating NaN values in the validation accuracy.

In summary, while ResNet models demonstrated a significant advantage in handling deeper network architectures, careful consideration must be given to dataset size and model complexity to prevent overfitting. Future work could focus on implementing transfer learning strategies to optimize the performance of these deep networks further.

References:

- [1] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556. 2014 Sep 4.
- [2] He, K., Zhang, X., Ren, S. and Sun, J., 2016. Deep residual learning for image recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 770-778.
- [3] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. Neural Computation, 1989.
- [4] A. Krizhevsky, I. Sutskever, and G. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS, 2012.
- [5] Huang, G., Liu, Z., Van Der Maaten, L. and Weinberger, K.Q., 2017. Densely connected convolutional networks. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 4700-4708.
- [6] Shorten, C. and Khoshgoftaar, T.M., 2019. A survey on image data augmentation for deep learning. Journal of Big Data, 6(1), p. 60.
- [7] Msoud Nickparvar. (2021). Brain Tumor MRI Dataset [Data set]. Kaggle. <https://doi.org/10.34740/KAGGLE/DSV/2645886>
- [8] Hochreiter, S., 1991. Untersuchungen zu dynamischen neuronalen netzen. Diploma thesis, TU Munich.
- [9] Bengio, Y., Simard, P., and Frasconi, P., 1994. Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks, 5(2):157–166.
- [10] Glorot, X., and Bengio, Y., 2010. Understanding the difficulty of training deep feedforward neural networks. Proceedings of AISTATS.