Brain Tumor MRI Classification Using Deep learning

1. Experimental Setup and Improvements

1.1 Pre-processing step

- **Data loading**: Used ImageDataGenerator to load the training data and test data. The generator rescaled the pixel values to a range of [0,1] and added Gaussian noise to the training data to prevent overfitting.

- **Data Splitting**: Although the preprocessing code cannot be modified, we handled the validation split by splitting the training data into 80% training and 20% validation after loading it using the generator.

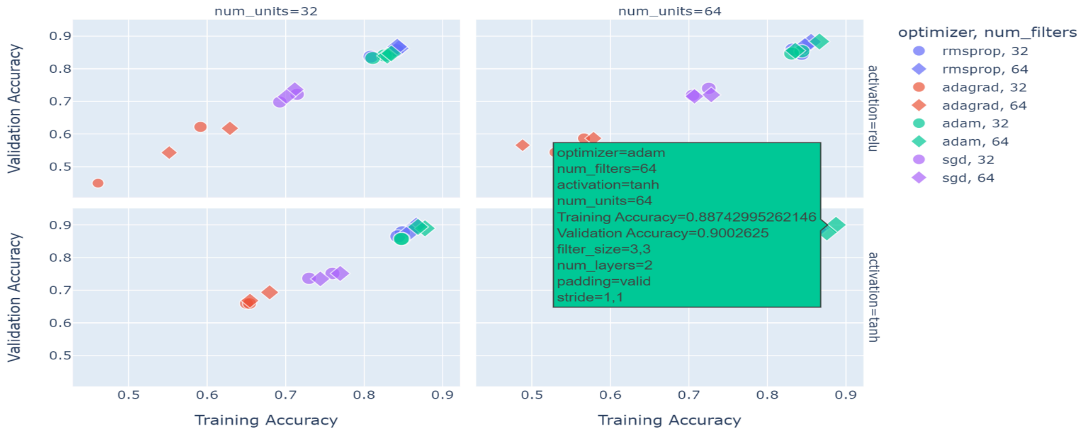
- **One-Hot Encoding**: Convert labels into one-hot encoded for multiclass classification.

1.2 Training, and hyperparameter settings

Due to the high number of hyperparameters and possible combinations, we defined a function that is able to create models with varying configurations to find the best setting using the train and validation sets. So we performed a grid search over the specified parameter space using loops. After that, we were able to adjust the parameters by modifying. The list of the defined experiments for different settings are:

- filter_sizes = [(3, 3),(5,5)]   # using different filter size
- num_filters_list = [32 , 64]   # using different number of filters
- num_layers_list = [2]   # using 2 layer CNN (Conv2D + Max pooling)
- num_units_list = [32 , 64]   # using different number of units in Dense layers
- activations = ['relu', 'tanh']   # using different activation function in layers
- paddings = ['valid','same']   # using different padding
- strides = [(1,1), (2, 2)]   # using different strides
- epochs_list = [10]   # using sufficient epochs
- batch_sizes = [16,32]   # using different batch_sizes
- optimizers = {'rmsprop': RMSprop(), 'adagrad': Adagrad(), 'adam':Adam(), 'sgd':SGD()}

Results of varying combinations of hyperparameters Visualized in the Fig … . As can bee seen, rmsprop and adam has a better result in both training and validation accuracy. In addition, tanh performed better than Relu. Also, the number of filters 64 and number of units 64 performed better than 32. And Also filter size (3,3) performed better than (5,5). In summary, by hyper parameter tuning, we improved the accuracy on validation set from 79.8% to 90%.



2. Model Modifications and Justifications

2.1 Modifying Optimization functions by different learning rate:

Using the insights from the last part and comparing the performance of different settings of hyperparameters, we decided to continue with the network architecture shown in figure … to finetune it more on learning rates to evaluate the performance of them on test data. In addition, before fitting each model we decided to use whole train data to fit the models. As Adam and RMSprop performed better than the other optimizers, we chose them to continue. The results are shown on Table … . As can be seen, optimizer **rmsprop** by learning rate 0.002 and optimizer **Adam** by learning rate **0.001** perform better on training and test set.

Optimizers	**Accuracy type**	**Learning rates**			
		0.01	**0.001**	**0.002**	**0.003**
rmsprop	**Train Accuracy**	**25.21%**	**98.44%**	**98.21%**	**98.34%**
	test Accuracy	**30.89%**	**77.57%**	**80.63%**	**78.26%**
Adam	**Train Accuracy**	**54.66%**	**99.26%**	**99.26%**	**97.43%**
	test Accuracy	**45.00%**	**80.32%**	**79.63%**	**79.25%**

Table : The proposed method's accuracy assessment based on different learning rates using various optimization algorithms.

2.2 Fighting overfit

We used 3 methods to fight the overfit. It helps us to trade-off between bias and variance. If our model has high variance it means that it is memorizing train data not the real pattern. Regularization in Deep Learning often works by trading an increase in bias for a reduction of variance.

1. Reducing the capacity of the network.

The simplest way to prevent overfitting is to reduce the size of the model, i.e. the number of units per layer. Here we run an experiment with 5 different numbers of Units. As can be seen in Table …, number of units=32 performs approximately as 64.

	Number of Units				
Optimizers	**4**	**8**	**16**	**32**	**64**
RMSprop (0.002)	**73.91%**	**76.58%**	**77.57%**	**79.33%**	**79.79%**
Adam (0.001)	**77.57%**	**79.63%**	**78.49%**	**79.71%**	**79.41%**

2. Adding dropout.

Dropout is one of the most effective and most commonly used regularization techniques for neural networks. Dropout, applied to a layer, consists of randomly "dropping out" a number of output features of the layer during training. We tried 0 to 0.6 dropout rate. The best stands on 0.4 for RMSprop.

Optimizers	**Accuracy type**	**Dropout Rate**				
		0	**0.1**	**0.2**	**0.4**	**0.6**
Adam (0.001)	**Train Accuracy**	**98.72%**	**98.55%**	**98.11%**	**96.87%**	**96.13%**
	Test Accuracy	**78.79%**	**77.57%**	**79.18%**	**80.24%**	**78.57%**
RMSprop (0.002)	**Train Accuracy**	**98.28%**	**98.14%**	**97.76%**	**97.46%**	**96.71%**
	Test Accuracy	**79.02%**	**77.88%**	**79.18%**	**76.43%**	**77.73%**

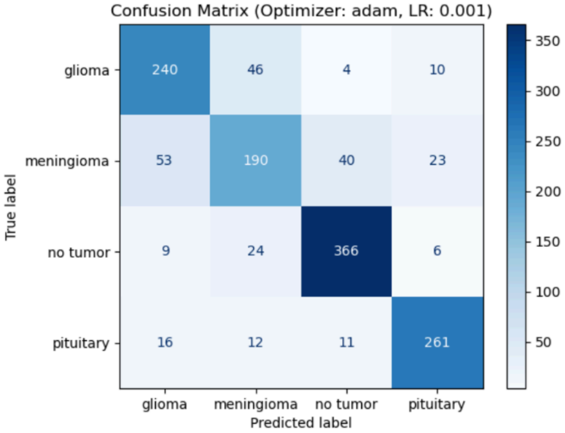
Table: Comparison of the proposed model’s average accuracy on different dropout rates.

3. L1-L2 Regularization

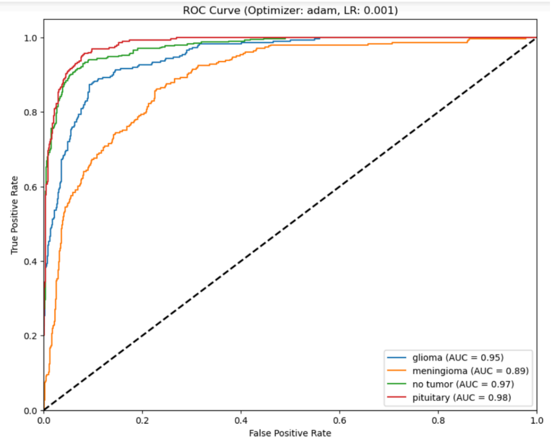
L1	**0**	**0.001**	**0**	**0.001**
L2	**0**	**0**	**0.001**	**0.001**
Train Accuracy	**97.4%**	**79.9%**	**95.2%**	**78.0%**
Test Accuracy	**79.6%**	**73.2%**	**78.3%**	**72.2%**

3. Results and Evaluation

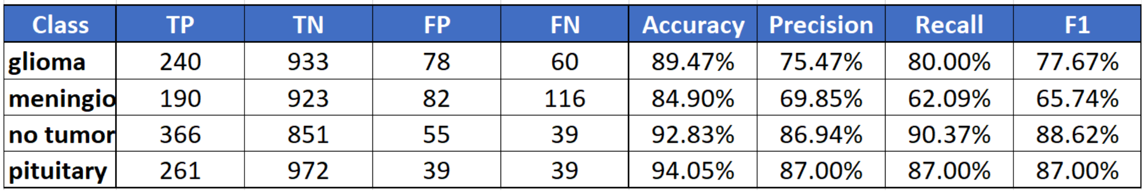
As can be seen in figure …, A confusion matrix based on the model's correct and incorrect predictions is developed to evaluate the proposed system's performance.The confusion matrix obtained throughout the experiments is shown in Table …. As can be seen, the proposed model properly classified 1057 examples and wrongly classified 254 cases. It is worth noting that 'no tumor' had the greatest prediction proportion. The dataset's balance considerably improved the classification results.


Confusion Matrix (Optimizer: adam, LR: 0.001)

True label \ Predicted label	glioma	meningioma	no tumor	pituitary
glioma	240	46	4	10
meningioma	53	190	40	23
no tumor	9	24	366	6
pituitary	16	12	11	261


ROC Curve (Optimizer: adam, LR: 0.001)

The classifier's performance was evaluated using this confusion matrix in terms of accuracy, sensitivity (recall), specificity, and f1-score for each tumor type. Table … demonstrates how well the proposed classifier worked for each type of brain tumor, indicating that our technique is more suited for classifying no tumor and pituitary from MRI scans.



Class	TP	TN	FP	FN	Accuracy	Precision	Recall	F1
glioma	240	933	78	60	89.47%	75.47%	80.00%	77.67%
meningio	190	923	82	116	84.90%	69.85%	62.09%	65.74%
no tumor	366	851	55	39	92.83%	86.94%	90.37%	88.62%
pituitary	261	972	39	39	94.05%	87.00%	87.00%	87.00%

4. Discussion

- Firstly , a simple CNN model was implemented as the base model, yielding an initial validation accuracy of 79.8%.
- Analysis of the results, comparison of models, and insights drawn from the experiments
- When we aimed to develop new CNN models with a small input image size of 30x30, we encountered some constraints. We needed to experiment with various numbers of filters and filter sizes as hyperparameters. However, after applying max pooling to reduce the size, we faced limitations in adding more layers. Therefore, we tested different combinations of stride and padding to overcome these challenges.
- Batch normalization is a powerful technique to enhance the performance of Convolutional Neural Networks (CNNs) in MRI classification tasks. By normalizing the inputs of each layer, batch normalization helps to stabilize and accelerate the training process. It reduces the internal covariate shift, allowing the network to learn more efficiently and effectively. This technique also acts as a regularizer, potentially reducing the need for other forms of regularization like dropout. In the context of MRI classification, where the data can be highly variable and complex, batch normalization ensures that the model remains robust and generalizes well to new, unseen data. But because of limitations in processing resources and time we couldn’t apply it on our model.

5. Conclusion

- key findings and outcomes of the project.

- Firstly , a simple CNN model was implemented as the base model, yielding an initial validation accuracy of 79.8%.
- At the second step we try to hyperparameter tuning. The team performed grid search over different hyperparameters such as filter sizes, number of filters, units in dense layers, activation functions, and optimizers. The tuning led to an improved validation accuracy of 90%.. We chose the best hyperparameters to create our CNN model. We saw rmsprop and adam had better results in both training and validation accuracy. In addition, tanh performed better than Relu. Also, the number of filters 64 and number of units 64 performed better than 32. And Also filter size (3,3) performed better than (5,5). In summary, by hyper parameter tuning, we improved the accuracy on validation set from 79.8% to 90%.
- Then we tried to modify optimizers by testing different learning rates. We found that optimizer **rmsprop** by learning rate 0.002 and optimizer **Adam** by learning rate **0.001** perform better on training and test sets.
- At the next step, we fight Overfitting by using 3 Techniques:Reducing network size (from 64 to 32), Adding dropout (best performance at 0.4 dropout rate) and L1-L2 regularization. We didn't find L1-L2 regularization useful for reducing overfitting.
- At the last try, we tried to apply transfer learning models. We started by pre-processing and Architecture selection. We experimented with three different transfer learning

architectures: VGG16, ResNet50, and DenseNet. Among these, DenseNet demonstrated the best performance in terms of accuracy and loss, making it the most effective transfer learning model. Therefore, we selected DenseNet for a more detailed comparison with the Baseline model.The Baseline model outperforms DenseNet with higher accuracy. This suggests that the simpler architecture of the Baseline model may be better suited for small grayscale images, leading to better training and generalization.DenseNet performs admirably, but it has slightly lower accuracy and higher loss, which may indicate that it struggles to fully leverage its pretrained features on this particular dataset.

- Based on limitation on computing capacity, we didn't use data augmentation

Table … shows the result of performance of different

Accuracy	Base model	Optimized mode	DenseNet
Validation set	0.79	0.90	0.79
Test set	0.79	0.81	0.74

The performance valuation of the proposed method on different quality measures.

6. Enhancement and future Improvements

Despite the effectiveness of the methods used in our project, there remains considerable potential methods for enhancing the results. Many advanced architectures and hybrid approaches have been applied to brain tumor classification problems and achieved successful results. In this section, we discussed potential improvements that could be tested to improve the performance of the predictions.

6.1 Practical Constraints and Image Resolution:

In addition to exploring advanced architectures, practical constraints related to computational resources played a crucial role in shaping the design of our models. Due to limitations in computational power and project requirements, we reduced the resolution of our input MRI images from the original size to 30x30.

While these adjustments helped manage computation time and memory usage, they also resulted in a loss of detail in the images, which may have limited the model's ability to capture important features, especially for complex tumor types. Higher resolution images could provide richer feature sets, leading to improved classification accuracy. In future iterations of the project, access to more robust computational resources would allow us to train models on higher resolution images, potentially enhancing the model's performance.

6.2 Hybrid Architectures:

One of the most promising directions is the exploration of hybrid architectures that combine the strengths of multiple approaches.

- KE-CNN

Concept:

Pashaei et al. (2018) combined Convolutional Neural Networks (CNN) with Kernel Extreme Learning Machines (KELM) to create the KE-CNN model. CNNs, a key component in deep learning, are particularly effective for visual data analysis, learning features through convolution and max-pooling layers. Extreme Learning Machines (ELM) are algorithms with one or more hidden node layers. In this model, CNNs extract hidden features from images, which are then classified by KELM. When compared to classifiers like Support Vector Machines and Radial Basis Functions, KE-CNN demonstrated promising results in brain tumor classification [1].

Application:

In the context of our work, applying a similar hybrid approach where CNNs extract features and KELM performs classification could help improve accuracy. This approach may work particularly well in distinguishing between hard-to-classify tumors, such as meningiomas and gliomas.

- Mask RCNN with DenseNet-41

Concept:

Masood et al. (2022) introduced a custom Mask Region-based Convolutional Neural Network (Mask RCNN) that incorporates DenseNet-41 at the feature computation layer. The backbone network, which can be any CNN model like ResNet-50, ResNet-101, or DenseNet, extracts relevant features from input MR images. A robust keypoint extraction network with numerous convolution layers is essential for learning reliable and discriminative features. However, deeper networks increase computational overhead and complicate weight optimization, potentially causing exploding gradient problems. ResNet models, with their skip connections and numerous parameters, can suffer from vanishing gradient issues. DenseNet, with its dense connections, computes a more representative set of image features. DenseNet-41 differs from traditional DenseNet in two ways: it has fewer parameters (24 channels in the first convolution layer instead of 64, and a 3 × 3 kernel size instead of 7 × 7) and adjusted layers within each dense block to manage computational complexity. These modifications make DenseNet-41 suitable for combining with Mask-RCNN. Mask-RCNN, an extension of Faster RCNN, performs segmentation in addition to classification and localization. It separates mask and class prediction, adding a small network overhead with an FCN network for segmentation [2].

Application: Incorporating Mask RCNN with DenseNet-41 into our project could significantly improve tumor classification and segmentation, providing more comprehensive and interpretable results.

- CNNs with a hybrid attention mechanism

Concept:

As explored by Rasheed et al. (2024), attention mechanisms can be integrated with CNNs to better focus on the most relevant parts of an image, allowing for more accurate detection and classification of brain tumors. The hybrid attention mechanism combines both channel and spatial attention, which helps the model focus on the most informative parts of the feature maps. This is especially useful in medical imaging where the variance in tumor size, shape, and location can make classification difficult using standard CNN architectures alone.

In their work, Rasheed et al. applied this hybrid model to classify brain tumors, including gliomas, meningiomas, pituitary tumors, and no-tumor cases, using magnetic resonance imaging (MRI). The CNN-based architecture, enhanced by attention mechanisms, achieved high classification accuracy with minimal preprocessing, demonstrating its effectiveness in both feature extraction and the generalization capability of the model (Rasheed et al., 2024).

Application:
Incorporating a hybrid attention mechanism into our project could improve both the accuracy and interpretability of tumor classification. The attention mechanism would allow the model to automatically prioritize the most relevant regions of the MRI images, such as the tumor's core, while reducing the influence of irrelevant background information. This could lead to more precise classification results, particularly when distinguishing between different tumor types.

6.3 Data Augmentation

Data augmentation is a powerful technique that can significantly enhance the performance of deep learning models in MRI classification. By artificially expanding the training dataset through transformations such as rotation, scaling, flipping, and adding noise, we can improve the model’s ability to generalize and recognize patterns in diverse and unseen data. This approach helps mitigate the challenges posed by limited medical imaging datasets, ensuring that the model learns robust features and reduces overfitting. Studies have shown that data augmentation can lead to substantial improvements in the accuracy and reliability of MRI-based classification tasks (Shorten & Khoshgoftaar, 2019). Therefore, incorporating data augmentation into the training pipeline is highly recommended for developing more effective and resilient deep learning models for MRI classification.

7. Transfer Learning

7.1 Preprocessing and Architecture selection

To prepare the dataset for transfer learning, we resize the input images to 32x32. This size was selected to ensure compatibility with the pretrained models while also accounting for the computational limitations. The input images were originally 30x30 grayscale images, and this resizing allowed the models to process the data effectively, though it might have reduced the overall detail captured by the models.

We experimented with three different transfer learning architectures: VGG16, ResNet50, and DenseNet. Among these, DenseNet demonstrated the best performance in terms of accuracy and loss, making it the most effective transfer learning model. Therefore, we selected DenseNet for a more detailed comparison with the Baseline model.

7.1.1 Accuracy and Loss:

Model	Training Accuracy	Validation Accuracy	Test Accuracy
Baseline	0.88	0.82	0.79
DenseNet121	0.84	0.79	0.74

The Baseline model outperforms DenseNet with higher accuracy. This suggests that the simpler architecture of the Baseline model may be better suited for small grayscale images, leading to better training and generalization.DenseNet performs admirably, but it has slightly lower accuracy and higher loss, which may indicate that it struggles to fully leverage its pretrained features on this particular dataset.

7.1.2 ROC Curve and AUC Scores:

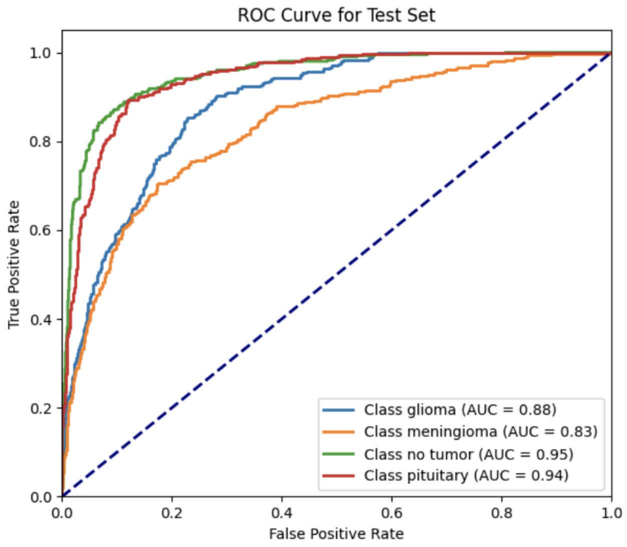


Figure: ROC Curve of DenseNet121 for test data

The Baseline model achieves higher AUC scores across all categories, indicating superior performance in differentiating between the tumor types. DenseNet performs well but trails behind the Baseline model, particularly in distinguishing gliomas and meningiomas, which are more challenging tumor types to classify.

7.1.3 Confusion Matrix Analysis

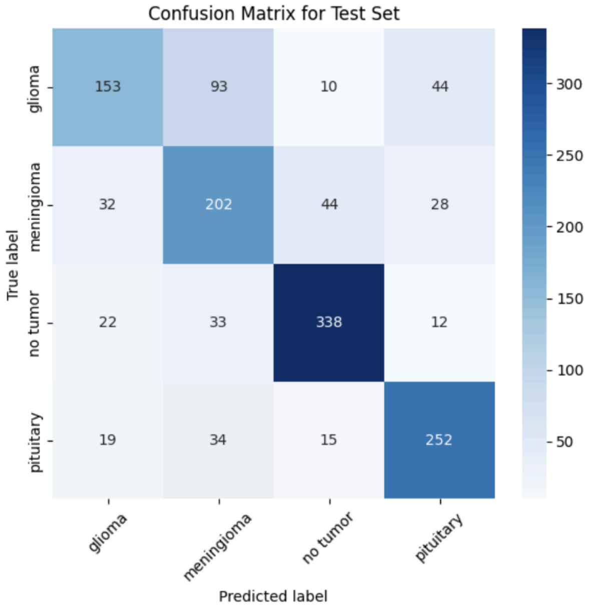


Figure: Confusion Matrix of DenseNet121 for test data

DenseNet's confusion matrix reveals more misclassifications, especially in the glioma and meningioma categories. Several instances of gliomas are incorrectly classified as meningiomas or no tumor, and vice versa. However, DenseNet performs reasonably well in distinguishing "no tumor" and pituitary cases.

The confusion matrix for the Baseline model shows fewer misclassifications across all categories. The model consistently performs better, with minimal errors in differentiating between tumor types, particularly in the challenging glioma and meningioma classes.

7.3 Conclusion

In summary, DenseNet, as the top-performing transfer learning model, showed strong results in terms of accuracy and generalization. However, the Baseline model consistently outperformed DenseNet in every key metric, including accuracy, AUC scores, and classification performance as measured by the confusion matrix.

One possible reason for DenseNet’s lower performance could be its complexity. Transfer learning models like DenseNet are designed to work with larger RGB images, and reducing the input size to 32x32 may have limited its ability to fully utilize the pretrained features. In contrast, the Baseline model’s simpler architecture is likely better suited to this specific dataset of small grayscale images, allowing it to capture the essential features more effectively.

8. References

1. Pashaei A, Sajedi H, Jazayeri N. Brain tumor classification via convolutional neural network and extreme learning machines. In: 2018 8th International conference on computer and knowledge engineering (ICCKE); 2018 Oct 25. p. 314–9.
2. Masood, M., Nazir, T., Nawaz, M., Mehmood, A., Rashid, J., Kwon, H.-Y., Mahmood, T., & Hussain, A. (2022). A Novel Deep Learning Method for Recognition and Classification of Brain Tumors from MRI Images.
3. Rasheed, Z., Ma, Y. K., Ullah, I., Al-Khasawneh, M., Almutairi, S. S., & Abohashrh, M. (2024). Integrating Convolutional Neural Networks with Attention Mechanisms for Magnetic Resonance Imaging-Based Classification of Brain Tumors. Bioengineering, 11(7), 701.
4. jjShorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data, 6*(1), Article 60. https://doi.org/10.1186/s40537-019-0197-0.