CerebralNet: A Deep Learning Approach to Alzheimer Detection from MRI

ADINJAY PHADKULE
Northeastern University
phadkule.a@northeastern.edu

SHASWAT SINHA
Northeastern University
sinha.sha@*northeastern*.edu

RAGHAVI DUBE
Northeastern University
dube.ra@*northeastern*.edu

***Abstract*— Alzheimer's disease (AD) is a debilitating neurodegenerative disorder characterized by progressive cognitive decline. Early and accurate diagnosis of AD is crucial for timely intervention and management. This research focuses on the development of a deep learning model for AD classification using magnetic resonance imaging (MRI) scans. The study utilizes a dataset comprising MRI images from patients with varying degrees of AD severity [1]. Preprocessing techniques, including image resizing and normalization, are applied to standardize the input data. A convolutional neural network (CNN) architecture is designed and trained on the preprocessed images, with evaluation conducted using metrics such as accuracy, loss, confusion matrix, and classification report. The results demonstrate the effectiveness of the proposed model in accurately classifying AD severity levels. This research contributes to the advancement of automated AD diagnosis and lays the foundation for further exploration in the field of medical image analysis and neuroimaging.**

***Keywords*— Alzheimer's disease, CNN, ReLu, MRI, SoftMax, Adam, cross entropy, EfficientNetB0**

Introduction

Alzheimer's disease (AD) stands as one of the most challenging public health issues of our time, affecting millions worldwide with devastating consequences for individuals and their families. It is a progressive neurodegenerative disorder characterized by memory loss, cognitive decline, and ultimately, the loss of independence. Early and accurate diagnosis of AD is paramount for effective treatment and management strategies. However, traditional diagnostic methods often rely on clinical assessments and neuropsychological tests, which may lack sensitivity and specificity, particularly in the early stages of the disease.

The motivation behind this project lies in the urgent need for more reliable and efficient diagnostic tools for AD [4]. With advancements in medical imaging technology, particularly magnetic resonance imaging (MRI), there is an opportunity to leverage machine learning techniques to analyse brain scans and aid in the early detection and classification of AD. By developing a deep learning model capable of accurately classifying AD severity levels based on MRI images, we aim to contribute to the improvement of diagnostic accuracy and patient outcomes in AD care.

Our approach involves the utilization of a convolutional neural network (CNN), a type of deep learning architecture well-suited for image analysis tasks. CNNs have demonstrated remarkable success in various computer vision applications, including medical image analysis. The decision to employ a CNN stems from its ability to automatically learn relevant features from raw input data, making it particularly suitable for extracting complex patterns from MRI images of the brain.

For our experiments and evaluation, we utilize a comprehensive dataset comprising MRI scans from individuals at different stages of AD progression. This dataset includes images from patients diagnosed with mild, moderate, and severe AD, as well as those classified as non-demented. By training and testing our CNN model on this diverse dataset, we aim to assess its performance in accurately classifying AD severity levels and providing valuable insights into disease progression.

Overall, this research endeavor represents a concerted effort to leverage state-of-the-art deep learning techniques and medical imaging data to address the critical challenge of AD diagnosis. Through our approach, we seek to contribute to the advancement of diagnostic capabilities in AD care and ultimately improve the quality of life for individuals affected by this debilitating disease.

DATASET

The dataset used for this research consists of MRI images obtained from individuals at different stages of Alzheimer's disease (AD) progression [2]. The dataset is divided into training, validation, and test sets, with the following dimensions:

Training set (X_train): 8960 MRI images with corresponding AD severity labels.

Validation set (X_val): 1920 MRI images with corresponding labels for model validation.

Test set (X_test): 1920 MRI images with ground truth labels for independent evaluation.

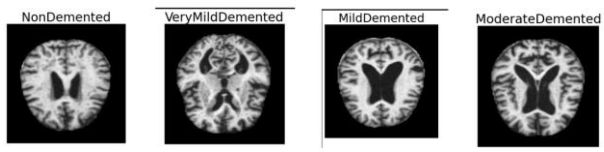


Fig. 1 Labels in the Dataset

Background

Alzheimer's Disease (AD) is a progressive neurodegenerative disorder characterized by cognitive decline, memory impairment, and ultimately, functional incapacity. As the global population ages, AD presents a significant public health challenge, with an estimated 50 million individuals currently affected worldwide. Early and accurate diagnosis of AD is crucial for timely intervention and treatment planning, yet it remains a complex endeavor, often reliant on costly and invasive procedures.

Recent advancements in artificial intelligence, particularly in the realm of deep learning, offer promising avenues for improving AD diagnosis and management. Convolutional Neural Networks (CNNs), a class of deep learning models, have demonstrated remarkable success in image classification tasks, making them particularly well-suited for analyzing medical imaging data, including magnetic resonance imaging (MRI) scans commonly used in AD diagnosis.

The primary objective of this research paper is to explore the efficacy of CNN models in detecting and classifying AD severity levels based on MRI images. The classification task involves categorizing images into distinct AD severity classes, such as mild, very mild, none, and moderate, providing clinicians with valuable insights into disease progression and prognosis.

By leveraging large datasets of MRI images annotated with AD severity labels, CNN models can learn intricate patterns and features indicative of different disease stages. Through the process of training, validation, and testing, these models can achieve high levels of accuracy in discriminating between AD severity classes, ultimately contributing to more accurate and efficient diagnosis.

Furthermore, the utilization of CNN models in AD detection holds the potential to augment existing diagnostic approaches, offering non-invasive, cost-effective, and scalable solutions. The ability to automate the analysis of MRI scans and provide reliable classification results may facilitate early detection, personalized treatment strategies, and improved patient outcomes.[5]

Approaches

Our approach revolves around the development and implementation of a convolutional neural network (CNN) architecture tailored specifically for Alzheimer's disease (AD) classification using magnetic resonance imaging (MRI) scans. In this section, we provide a detailed description of our methods, including data preprocessing, model architecture, training procedure, and evaluation metrics.

DATA PRE-PROCESSING

Rescaling: The MRI images are resized to a uniform dimension of 176x176 pixels to ensure consistency in input size across all samples.

- Normalization: Pixel intensity values are scaled to the range [0, 1] by dividing each pixel value by 255. This normalization step standardizes the input data and facilitates convergence during model training.
- Data Augmentation: To enhance model robustness and prevent overfitting, data augmentation techniques we have balanced the dataset and thus obtained a really good result. The individual distribution on label before and after is pictured in the figure below.

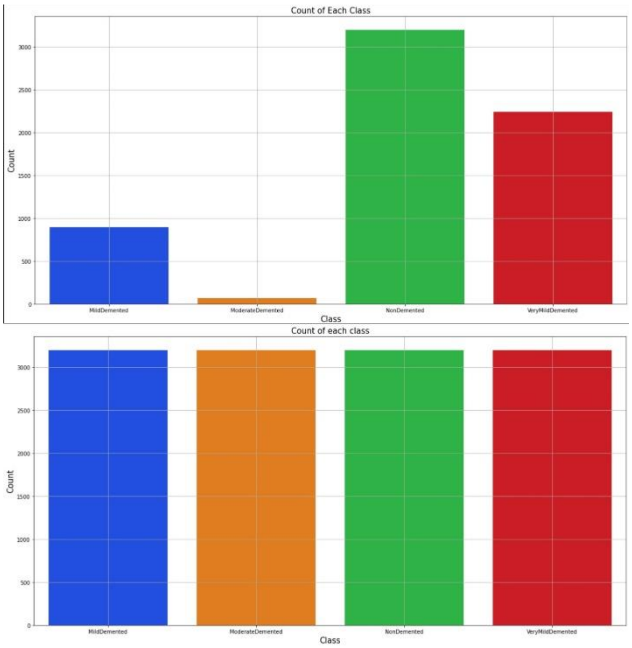


Fig. 2 Imbalanced and Balanced dataset

MODEL ARCHITECTURE

Our CNN architecture comprises several convolutional and max-pooling layers followed by fully connected layers.

- Convolutional Layers: Convolutional layers with varying kernel sizes and strides are employed to extract spatial features from the input MRI images. ReLU activation functions are applied to introduce non-linearity.
- Max-Pooling Layers: Max-pooling layers are interspersed between convolutional layers to down sample the feature maps and reduce computational complexity while preserving important features.
- Fully Connected Layers: The output of the convolutional layers is flattened and passed through dense (fully connected) layers, which act as classifiers. Dropout regularization is applied to mitigate overfitting by randomly deactivating neurons during training.
- Output Layer: The final layer consists of SoftMax activation, producing probabilities for each class (AD severity level).

TRAINING PROCEDURE

The model is trained using the Adam optimizer with a categorical cross-entropy loss function.

We employ early stopping with a patience of 10 epochs to prevent overfitting and save the best-performing model based on validation loss.

Training is conducted on a subset of the dataset, with validation data used for monitoring model performance and hyperparameter tuning.

EVALUATION PROCEDURE

Model performance is assessed using various evaluation metrics, including accuracy, loss, confusion matrix, and classification report.

- Accuracy: The proportion of correctly classified samples over the total number of samples.
- Loss: The value of the categorical cross-entropy loss function, indicating the discrepancy between predicted and actual class labels.
- Confusion Matrix: A matrix summarizing the number of true positive, true negative, false positive, and false negative predictions.
- Classification Report: Provides precision, recall, F1-score, and support for each class, offering insights into the model's performance across different AD severity levels.

PREDICTION ON TEST DATASET

Once the model is trained and validated, it is evaluated on an independent test dataset that was not used during training

or validation.

The test dataset comprises MRI images and their corresponding ground truth labels for AD severity levels.

The trained CNN model is used to predict the AD severity level for each MRI image in the test dataset.

Predictions are made by feeding the test images through the trained model and obtaining the model's output probabilities for each class (AD severity level).

The predicted labels are compared against the ground truth labels to assess the model's performance on unseen data.

Evaluation metrics such as accuracy, loss, confusion matrix, and classification report are computed based on the model's predictions on the test dataset.

Through this approach, we aim to develop a robust and accurate CNN model capable of effectively classifying AD severity levels based on MRI images. By leveraging state-of-the-art deep learning techniques and rigorous evaluation, we seek to contribute to the advancement of automated AD diagnosis and improve patient care outcomes.

MODEL 1

Dataset Splitting
The dataset was split into training, testing, and validation sets using the train_test_split function from scikit-learn library. The training set comprised 8960 samples, the testing set contained 1920 samples, and the validation set also had 1920 samples. Each sample had dimensions of 176x176 pixels with 3 color channels.
Model Architecture
The CNN model was constructed using the Sequential API from Keras. It consisted of several convolutional layers followed by max-pooling layers for down-sampling, a flattening layer to convert the 2D output to a 1D vector, and fully connected dense layers for classification.

Layer (type)	Output Shape	Param #
Conv2D	(None, 88, 88, 32)	896
Conv2D	(None, 44, 44, 64)	18,496
MaxPooling2D	(None, 22, 22, 64)	0
Conv2D	(None, 11, 11, 128)	73,856
MaxPooling2D	(None, 6, 6, 128)	0
Flatten	(None, 4608)	0
Dense	(None, 1024)	4,719,616
Dropout	(None, 1024)	0
Dense	(None, 4)	4,100

Total params: 4,816,964 (18.38 MB)
Trainable params: 4,816,964 (18.38 MB)
Non-trainable params: 0 (0.00 B)

Fig.3 Model 1 Summary

Training
The model was compiled using the Adam optimizer and categorical cross-entropy loss function. During training, Model Checkpoint and Early Stopping callbacks were employed to save the best model and prevent overfitting, respectively. The model was trained for 10 epochs.
Training and Validation Metrics
The model achieved a peak accuracy of 99.26% on the training set and 97.55% on the validation set. The corresponding losses were 0.0229 and 0.0646, respectively. Training and validation accuracy and loss were plotted over epochs.
Testing
The trained model was evaluated on the test set, yielding a test accuracy of 98.07% and a test loss of 0.0552.

MODEL 2

Model Architecture
The basic CNN model was constructed using the Sequential API from Keras. It consisted of two convolutional layers with 32 and 64 filters respectively, followed by a max-pooling layer for down-sampling. The output was then flattened and fed into two fully connected dense layers for classification.

Layer (type)	Output Shape	Param #
Conv2D	(None, 88, 88, 32)	896
Conv2D	(None, 44, 44, 64)	18,496
MaxPooling2D	(None, 22, 22, 64)	0
Flatten	(None, 30976)	0
Dense	(None, 1024)	31,720,448
Dense	(None, 4)	4,100

Total params: 31,743,940 (121.09 MB)
Trainable params: 31,743,940 (121.09 MB)
Non-trainable params: 0 (0.00 B)

Fig.4 Model 2 Summary

Training

The model was compiled using the Adam optimizer and categorical cross-entropy loss function. During training, Model Checkpoint and Early Stopping callbacks were employed to save the best model and prevent overfitting, respectively. The model was trained for 10 epochs.

Training and Validation Metrics

The model achieved a peak accuracy of 99.33% on the training set and 98.70% on the validation set. The corresponding losses were 0.0032 and 0.0453, respectively. Training and validation accuracy and loss were plotted over epochs.

Testing

The trained model was evaluated on the test set, yielding a test accuracy of 98.80% and a test loss of 0.0453.

MODEL 3

EfficientNetB0 with Augmentation

Model Architecture

The EfficientNetB0 model was utilized with its pre-trained weights excluded, and the last fully connected layer replaced to adapt to the specific classification task with 4 classes. The input shape was defined as (176, 176, 3).

Layer (type)	Output Shape	Param #
EfficientNetB0	-	4,054,695

Total params: 4,054,695 (15.47 MB)
Trainable params: 4,012,672 (15.31 MB)
Non-trainable params: 42,023 (164.16 KB)

Fig.5 Model 3 Summary

Training

The model was compiled using the Adam optimizer and categorical cross-entropy loss function. It was trained for 4 epochs using the provided training data.

Training and Validation Metrics

During training, the model achieved an accuracy of 90.77% and a loss of 0.2397 on the training set, while on the validation set, it attained an accuracy of 84.95% with a loss of 0.3618. These metrics were plotted to visualize the training and validation performance over epochs.

Testing

Upon evaluation on the test set, the model achieved a test accuracy of 85.97% with a test loss of 0.3426.

Without Augmentation

Data Splitting

The dataset was split into training, validation, and testing sets using a 70-15-15 ratio, respectively. The stratification technique was employed to ensure balanced class distribution in each set.

Training Set: 4480 samples
Validation Set: 960 samples
Testing Set: 960 samples

Model Architecture

The CNN model consists of several convolutional and max-pooling layers followed by fully connected layers. Dropout regularization was applied to mitigate overfitting.

Layer (type)	Output Shape	Param #
Conv2D	(None, 88, 88, 32)	896
Conv2D	(None, 44, 44, 64)	18,496
MaxPooling2D	(None, 22, 22, 64)	0
Conv2D	(None, 11, 11, 128)	73,856
MaxPooling2D	(None, 6, 6, 128)	0
Flatten	(None, 4608)	0
Dense	(None, 1024)	4,719,616
Dropout	(None, 1024)	0
Dense	(None, 4)	4,100

Total params: 4,816,964 (18.38 MB)
Trainable params: 4,816,964 (18.38 MB)
Non-trainable params: 0

Fig.6 Model 3 Without Data Augmentation Summary

Training

The model was trained for 10 epochs using the Adam optimizer and categorical cross-entropy loss function. Early stopping with a patience of 10 epochs and model checkpointing were applied to prevent overfitting and save the best model.

Training and Validation Metrics

The training and validation accuracies increased over epochs, indicating learning progress. Similarly, the training and validation losses decreased over epochs, suggesting improved model performance.

Testing

Upon evaluation on the test set, the model achieved a test accuracy of 95.21% with a test loss of 0.1287.

MODEL 4

Model Architecture

This CNN model comprises two convolutional layers followed by max-pooling and a fully connected layer.

Layer (type)	Output Shape	Param #
Conv2D	(None, 88, 88, 32)	896
Conv2D	(None, 44, 44, 64)	18,496
MaxPooling2D	(None, 22, 22, 64)	0
Flatten	(None, 30976)	0
Dense	(None, 4)	123,908

Total params: 143,300 (559.77 KB)
Trainable params: 143,300 (559.77 KB)
Non-trainable params: 0

Fig.7 Model 4 Summary

Training

The model was trained for 10 epochs using the Adam optimizer and categorical cross-entropy loss function. Early stopping with a patience of 10 epochs and model checkpointing were applied to prevent overfitting and save the best model.

Training and Validation Metrics

The training and validation accuracies increased over epochs, indicating learning progress. Similarly, the training and validation losses decreased over epochs, suggesting improved model performance.
Testing
Upon evaluation on the test set, the model achieved a test accuracy of 97.50% with a test loss of 0.0911.

Results

We have tabulated the training accuracies and validation accuracies of all the models in below Fig.8

Model	Training Accuracy	Validation Accuracy
Model 1	98.26%	97.55%
Model 2	99.3%	98.70%
Model 3	90.7%	84.67%
Model 4	97.51%	98.60%

Fig.8 Model Accuracies

MODEL 1 RESULTS

Epoch	Training Accuracy	Training Loss	Validation Accuracy	Validation Loss
1	58.57%	0.866	72.86%	0.598
2	80.15%	0.443	82.50%	0.384
3	90.26%	0.227	92.34%	0.169
4	95.69%	0.114	95.62%	0.111
5	97.35%	0.072	95.68%	0.107
6	98.50%	0.044	96.61%	0.109
7	98.84%	0.031	91.61%	0.229
8	98.55%	0.042	97.86%	0.058
9	99.32%	0.021	97.55%	0.065
10	99.05%	0.027	97.40%	0.076

Fig.9 Epoch vs Accuracy

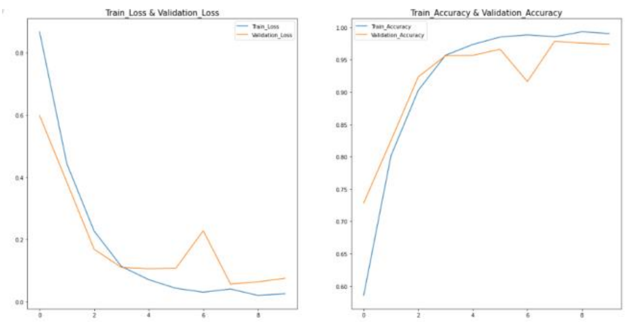
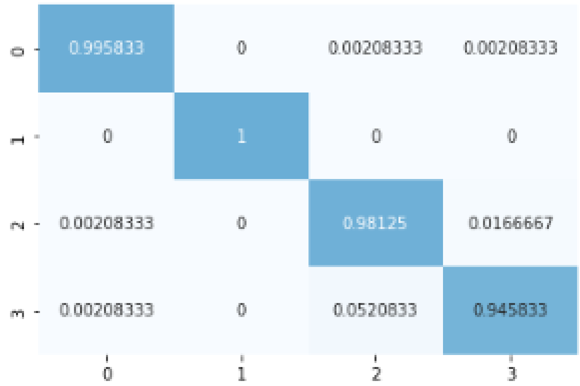


Fig.10 Training vs Validation Loss/Accuracy

```
array([[478,   0,   1,   1],
       [  0, 480,   0,   0],
       [  1,   0, 471,   8],
       [  1,   0,  25, 454]], dtype=int64)
```




```
Classification
Report          precision    recall  f1-score   support

           0       1.00      1.00      1.00       480
           1       1.00      1.00      1.00       480
           2       0.95      0.98      0.96       480
           3       0.98      0.95      0.96       480

    accuracy                           0.98      1920
   macro avg       0.98      0.98      0.98      1920
weighted avg       0.98      0.98      0.98      1920
```

Fig.11 Confusion matrix and Classification report

MODEL 2 RESULTS

Epoch	Accuracy	Loss	Val Accuracy	Val Loss
0	65.80%	79.01%	83.23%	38.42%
1	91.91%	21.10%	94.79%	14.07%
2	97.51%	7.23%	95.89%	10.88%
3	98.90%	3.30%	96.09%	9.57%
4	99.44%	1.75%	98.07%	5.61%
5	99.74%	0.86%	96.61%	8.78%
6	99.34%	1.94%	97.81%	7.90%
7	99.55%	1.59%	89.27%	34.34%
8	98.90%	3.31%	98.13%	5.73%
9	99.93%	0.32%	98.70%	4.53%

Fig.12 Epoch vs Accuracy

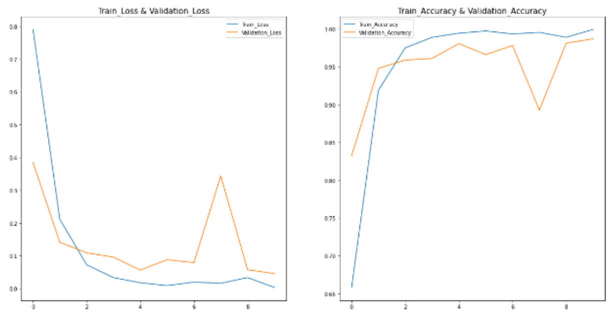
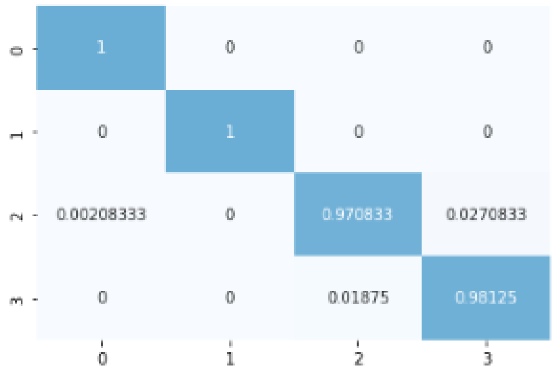


Fig.13 Training vs Validation Loss/Accuracy

```
array([[480,   0,   0,   0],
       [  0, 480,   0,   0],
       [  1,   0, 466,  13],
       [  0,   0,   9, 471]], dtype=int64)
```



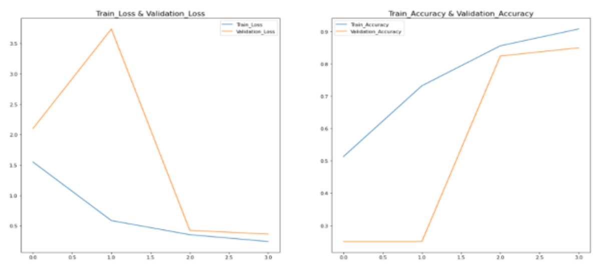

```
Classification
Report           precision    recall  f1-score   support

           0       1.00      1.00      1.00       480
           1       1.00      1.00      1.00       480
           2       0.98      0.97      0.98       480
           3       0.97      0.98      0.98       480

    accuracy                           0.99      1920
   macro avg       0.99      0.99      0.99      1920
weighted avg       0.99      0.99      0.99      1920
```

Fig. 14 Confusion matrix and Classification report

Model 3 Results With Data Augmentation



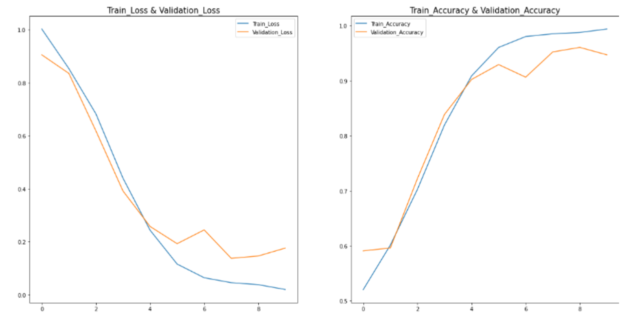

```
Classification
Report           precision    recall  f1-score   support

           0       0.88      0.95      0.91       480
           1       1.00      1.00      1.00       480
           2       0.95      0.61      0.74       480
           3       0.69      0.87      0.77       480

    accuracy                           0.86      1920
   macro avg       0.88      0.86      0.86      1920
weighted avg       0.88      0.86      0.86      1920
```

Fig.15 Training vs Validation Loss/Accuracy and Classification report

Model 3 Results Without Data Augmentation




```
Classification
Report           precision    recall  f1-score   support

           0       0.95      0.94      0.95       134
           1       1.00      0.90      0.95        10
           2       0.95      0.97      0.96       480
           3       0.95      0.93      0.94       336

    accuracy                           0.95       960
   macro avg       0.96      0.94      0.95       960
weighted avg       0.95      0.95      0.95       960
```

Fig. 16 Training vs Validation Loss/Accuracy and Classification report

Model 4 Results

epoch	accuracy	loss	Val accuracy	Val loss
0	0.514286	0.984984	0.591667	0.883480
1	0.654241	0.768129	0.732292	0.646259
2	0.816295	0.467903	0.845833	0.420374
3	0.920089	0.234274	0.865625	0.332019
4	0.959152	0.130508	0.926042	0.191484
5	0.978348	0.081276	0.922917	0.198031
6	0.995759	0.026523	0.925000	0.201355
7	0.999777	0.011590	0.957292	0.112512
8	1.000000	0.006117	0.963542	0.107464
9	1.000000	0.004248	0.970833	0.087146

Fig. 17 Epoch vs Accuracy

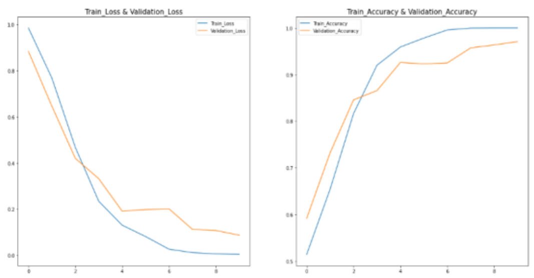
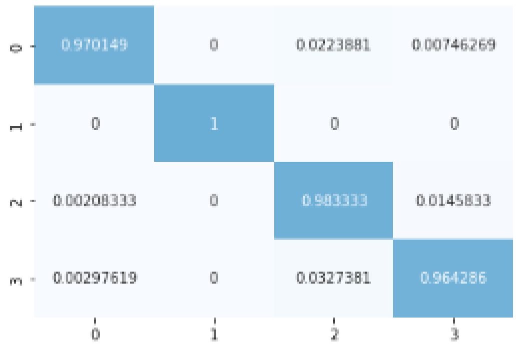


Fig.18 Training vs Validation Loss/Accuracy

```
array([[130,   0,   3,   1],
       [  0,  10,   0,   0],
       [  1,   0, 472,   7],
       [  1,   0,  11, 324]], dtype=int64)
```




```
Classification
Report              precision    recall  f1-score   support

           0       0.99      0.96      0.98       134
           1       1.00      1.00      1.00        10
           2       0.97      0.98      0.98       480
           3       0.96      0.96      0.96       336

    accuracy                           0.97       960
   macro avg       0.98      0.98      0.98       960
weighted avg       0.97      0.97      0.97       960
```

Fig.19 Confusion matrix and Classification report

References

[1] Al Shehri W. (2022). Alzheimer's disease diagnosis and classification using deep learning techniques. PeerJ. Computer science, 8, e1177. https://doi.org/10.7717/peerj-cs.1177

[2] https://www.kaggle.com/code/hadeerismail/alzheimer-detection

[3] A novel CNN architecture for accurate early detection and classification of Alzheimer’s disease using MRI data. (n.d.) retrieved April 14, 2024, from www.nature.com/articles/s41598-024-53733-6

[4] A novel hybrid model in the diagnosis and classification of Alzheimer's disease using EEG signals: Deep ensemble learning (DEL) approach. (n.d.) retrieved April 14, 2024, from www.sciencedirect.com

[5] Automated classification of Alzheimer's disease and mild cognitive impairment using a single MRI and deep neural networks. (n.d.) retrieved April 14, 2024, from www.sciencedirect.com/science/article/pii/S2213158218303930