Journal of High School Science

Deep Learning-based classification of Alzheimer's stages: A multiclass approach using MRI data

Yedavalli R[1], Bair A[2]

Submitted: September 10, 2024, Revised: version 1, October 18, 2024
Accepted: October 27, 2024

Abstract
Alzheimer's disease (AD), a neurodegenerative disorder that affects millions across the world annually, is recognized by a progressive decline in cognitive functions such as memory, orientation, and reasoning. Despite large advances in the understanding of its pathology, ranging from recent identification of amyloid-β plaques to tau tangles, treatment and early diagnosis remain very elusive. This study presents an enhanced Convolutional Neural Network (CNN) model designed to classify MRI images into four stages of Alzheimer's disease: non-demented, very mildly demented, mildly demented, and moderately demented. The model incorporates four convolutional layers with ReLU activation, batch normalization, and max-pooling, followed by fully connected layers with dropout regularization to prevent overfitting. Trained on a weighted dataset of 6400 MRI images, the model achieved a peak training accuracy of 99.7% with a final testing accuracy of 88.79% on unseen data. This study ultimately underlines the potential that CNNs hold for early detection and accurate classification of Alzheimer's disease as a powerful tool for enhancement in diagnostic precision within clinical settings.

Keywords
Alzheimer's disease, Convolutional Neural Network, Image classification, Amyloid-β plaques, Neuroimaging, Tau Tangles, MRI images, Neurodegenerative disorders, Neuroinflammation

---

[1]Corresponding author, Rishi Yedavalli, Fulton Science Academy Private School, 3035 Fanfare way, Alpharetta, GA 30009, USA. yedavallirishi@gmail.com
[2]Anna Bair, Machine Learning, Computer Science department, Carnegie Mellon University, 5000 Forbes Ave. Pittsburgh, PA 15213, USA. annaebair@gmail.com

1. Introduction

Alzheimer's disease (AD) affects millions worldwide and is characterized by the progressive loss of memory, disorientation, and intellectual decline that interfere with daily activities. The cause of Alzheimer's remains evasive, but the pathological hallmarks are quite defined: the accumulation of amyloid-beta plaques and tau tangles in the brain that disrupt communication between neurons and lead to degeneration of brain tissue (1). While the majority
of cases are sporadic, being late-onset and driven by a mix of genetic, lifestyle, and environmental factors, early-onset familial Alzheimer's disease has been found to be due to certain specific genetic mutations (2). The identification of these mutations has been very important in making progress toward the understanding of the molecular basis of the disease, although the actual mechanisms are quite complex.

Recent advances in neuroimaging have contributed significantly to present knowledge about Alzheimer's disease. Technologies such as positron emission tomographies (PET) and cerebrospinal fluids (CSF) flow imaging have made it possible to visualize the amyloid plaques and tau tangles in living patients (3). It has allowed new insights into AD pathobiology, enabling earlier diagnosis and identification of AD stages. Working in tandem, the rise of Artificial Intelligence, particularly neural networks, has further broadened the area of research for Alzheimer's.

Convolutional Neural Networks (CNN) with advanced image processing capabilities have been increasingly used to analyze neuroimaging data, enabling more accurate prediction and classification of Alzheimer's disease, automating processes that can otherwise be susceptible to risk-factors. For example, CNNs may detect very small patterns and changes in structure in brain scans that may go unnoticed, mostly, even by human eyes. This could, hence, detect the disease at a much earlier stage and consequently hold possibilities for intervention that might alter the course of the disease, or potentially even delay it. CNNs can also be important in distinguishing Alzheimer's from other types of dementia, and thereby increase the diagnostic accuracy and personal treatment approach (4).

This paper ultimately aims to develop a Convolutional Network Network (CNN) specially dedicated to classifying the severity of Alzheimer's disease through MRI scans in the following four categories: non demented, very mildly demented, mildly demented, and moderately demented. The model design includes four convolutional layers with ReLU activation, combined with batch normalization and max-pooling, followed by fully connected layers that utilize dropout regularization to mitigate overfitting.. The model will make use of a set of MRI images that are pre-labeled, augmented through various transformations, to enhance its robustness. By leveraging such architecture, the paper aims to enhance early detection and differentiate the stages related to Alzheimer's while synthesizing existing research, finding advances, and expanding the scope of knowledge within the field of neurology.

2. Literature review

2.1 Nature of Alzhehimer’s disease

Alzheimer's disease was first discovered in 1906, yet it remains one of the most widespread and complex neurodegenerative diseases in the world. It affects about 10% of people over the age of 65, which translates to > 50 million people globally. Although much research has been performed across the past several years, this complex disease challenges cognition levels and consequently complicates the development of effective treatments (5). Characterized by progressive decline in cognitive functioning, Alzehimer's originates within a central area, branching off into certain formations that result after its onset. The first structures affected include the hippocampus - a vital part of the temporal lobe - which is mainly responsible for long-term memory function. The core diagnostic features include the amyloid-beta plaques and tau protein tangles in the brain. Amyloid plaques are formed due to abnormal processing of amyloid precursor protein, which leads to extracellular deposits impairing neuronal communication, whereas tau tangles occur from hyperphosphorylated tau which results in intracellular aggregates disrupting microtubule stability and eventually leads to the death of neurons (6). The contribution of chronic neuroinflammation, driven by the prolonged activation of microglia and astrocytes, exacerbates neuronal damage and accelerates the progression of disease pathology. Neural damage is further enhanced by chronic neuroinflammation as a result of activated microglia and astrocytes, continuing to contribute to the disease pathology (7). In fact, emerging evidence suggests that other biological factors have been shown to increase susceptibility to the early-onset of Alzheimer's disease; namely genetic features, such as mutations in a number of genes including APP, PSEN1, and PSEN2 (8). Alzheimer's pathophysiology additionally includes synaptic dysfunction, mitochondrial abnormalities, and oxidative stress, emphasizing the need for multi-therapeutic approaches (9). For these reasons, Alzheimer's continues to pose a significant challenge within the world of medicine due to its complex nature.

2.2 Artificial Intelligence progression

Numerous applications of artificial intelligence have shown significant potential in mimicking brain functions and enhancing the understanding of neurological disorders. Specifically, neural networks, which imitate the structures and architecture of the human brain, have found applications in the majority of fields due to their success in allowing very complicated tasks to be performed with high accuracy. Such networks are a composition of connected nodes or "neurons" that can learn from data and become more accurate at prediction over time. In this way, neural networks become very useful tools for pattern recognition and making decisions. Over the recent years, neural networks have found applications in areas of image and speech recognition, natural language processing, and predictive analytics. An example is their use in image classification tasks, wherein convolutional neural networks have significantly improved and led to progress in object recognition in images (10). Moreover, recurrent neural networks have played a very important role in the enhancement of systems for speech recognition that allow more natural interaction with humans (11). Additionally, deep learning and neural network applications are currently used in Natural Language Processing (NLP) for better chatbot and virtual

assistant design to improve language understanding and generation (12). These advances have been changing various domains by driving innovations and bringing in newer ways of working with better efficiency.

2.3 Neural Networks in medical imaging

One of the most promising and impactful current applications of neural networks is in the field of medical imaging. Specifically, deep learning models have continued to improve the diagnosis of neurological and neuropathic disorders in recent years. CNNs' have been successful in disease detection and classification of medical images, even being able to identify the presence of tumors in mammograms with a high degree of precision (13). These networks can also segment organs and tissues in MRIs and CT scans for improved planning and treatment monitoring (14). Furthermore, neural networks can help to predict patient outcomes by analyzing patterns in medical imaging data, hence facilitating strategies for personalized medical prescriptions (15). These applications; among others; highlight the crucial role played by neural networks in improving medical imaging for refined diagnosis.

2.4 Neural Networks in Alzheimer's detection

Neural networks, particularly deep learning models, have made huge contributions in the world

of research in Alzheimer's disease and, more importantly, early detection and diagnosis. They have proved significant in the analysis of neuroimaging data such as from MRI and PET scans that trace the initial signs of Alzheimer's, making it easier for prevention detection (16). These techniques have been fairly accurate in detecting patterns of brain atrophy, more so in the hippocampus and other very vital parts of the brain relative to Alzheimer's. These structures could be segmented with detailed insight into the disease's course through CNNs'. Further, advances in neural networks have allowed the integration of genetic, clinical, and neuroimaging multimodal data in such a way as to offer better capabilities in terms of diagnosis and prognosis (17). However, various challenges remain, such as the requirement for a large annotated dataset, which often proves to be resource-intensive to acquire. Generalization across thousands of diverse populations also presents several challenges due to demographic and clinical variations, which impact model performance. Furthermore, the interpretability of complex algorithms will also be extremely relevant for gaining trust in clinics since clinicians have to understand how models derive their conclusions to adequately trust them (18). Despite these complexities, techniques employed through neural networks will continue improving in the scope of early detection, diagnosis, and treatment monitoring of Alzheimer's disease and improve patient outcomes, furthering the understanding of the several nuances present within this condition.

3. Methods

3.1 Dataset

In this paper, the dataset used to train and test the image classification model was obtained

from Kaggle and consisted of 6400 MRI images of Alzheimer's patients sourced from public hospitals, subdivided into four classes: mild demented (896 images), moderate demented (64 images), non demented (3200 images), and very mild demented (2240

images) (19). The examples are shown below in Figure 1. The dataset provided images of size 128x128 pixels, which we resized to 224x224 for our model processing.

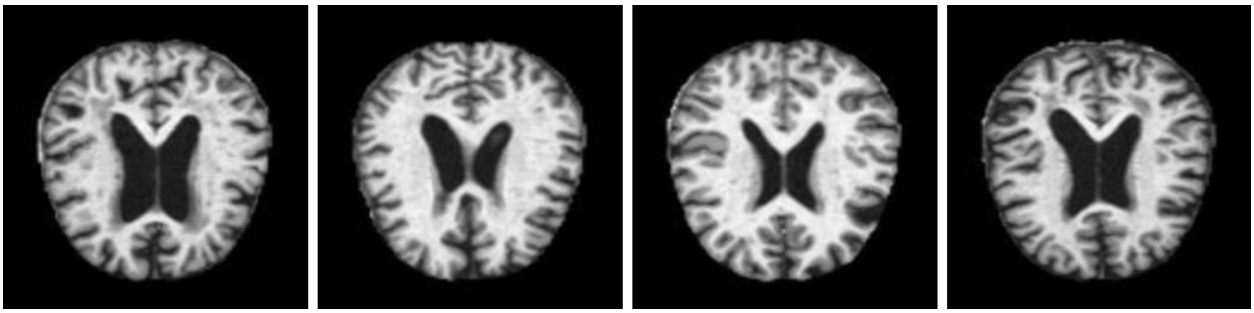

Figure 1. Mild (1st from left), Moderate (2nd from left), Non Demented (3rd from left), and Very Mild (4th from left)

3.2 Data preprocessing

To efficiently train the CNN for classifying MRI images of Alzheimer's disease into four distinct categories, data preprocessing was performed using a custom dataset class. The following subsections describe the steps taken to preprocess the data, build the dataset, and prepare it for input into the model.

3.2.1 *Train-Test split*

The dataset was split in an 80:20 ratio into training and testing datasets. Due to the architecture of the model (Section 3.3), the last convolutional layer output a 2048-dimensional feature vector (therefore, the number of features in this case was 2048). This suggested that approximately 2.2% of the data should be allocated to testing, and the remaining 97.8% should be used for training. However, the 80:20 split provided a better balance between training the model with sufficient data and ensured that the test set was large enough to give a reliable estimate of performance. A stratified split, based on class labels, was performed to ensure that each of the four classes was proportionally represented in both the training and testing sets. This ensured that the performance evaluation of the model would not be biased toward any specific class. Images from all four categories (Mild Demented, Moderate Demented, Non-Demented, and Very Mild Demented) were divided in such a way that no overlap occurred between the training and testing datasets. This ensured a fair evaluation and helped reduce the risk of overfitting.

3.2.2 *Data loader and input preparation*

The PyTorch DataLoader class was utilized to efficiently load the dataset into batches for parallel processing. Batching helps in speeding up the model training process, allowing for better memory management. The batches were shuffled during loading to prevent any potential biases that could affect model learning. Each image was resampled into the RGB format and standardized for uniformity. This preprocessing step, combined with batch loading, made it simpler for the CNN to process the images, ensuring efficient training. (Figure 2).

```
def __getitem__(self, idx):
    img_path, label = self.data[idx]
    image = Image.open(img_path).convert('RGB')
    if self.transform:
        image = self.transform(image)
    return image, label
```

Figure 2. Data preprocessing sample

3.2.3 *Image transformation*

A series of transformations were applied to standardize and augment the MRI images before input into the CNN model. The images were resized to 224x224 pixels, converted into PyTorch tensors, and normalized using the mean and standard deviation values of [0.485, 0.456, 0.406] and [0.229, 0.224, 0.225], respectively. These are standard values for models trained on ImageNet and ensure compatibility with pretrained architectures like ResNet (Figure 3). In addition, data augmentation techniques like random rotations, horizontal flips, and color jitter were applied to increase the diversity of the training data, which aids in better generalization and prevents overfitting.

```
# Data Transformations
transform = transforms.Compose([
    transforms.Resize((224, 224)),
    transforms.RandomHorizontalFlip(),
    transforms.RandomRotation(10),
    transforms.ToTensor(),
    transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
])
```

Figure 3. Data loading sample

3.2.4 *Feature extraction and correlation analysis*

After the model loaded and processed the input images, feature extraction was performed on the final convolutional layer (Section 3.3.1) to capture meaningful patterns. A Pearson correlation analysis was then applied to the extracted features to evaluate relationships and identify any potential multicollinearity. This correlation analysis, visualized through a heatmap in Figure 4, showed that the features were relatively uncorrelated (lighter blocks rather than darker), suggesting that the feature extraction from the CNN captured diverse and independent information from the input data. While multicollinearity does not directly impact prediction, this ultimately reduces redundant features and enhances computational efficiency within our model.

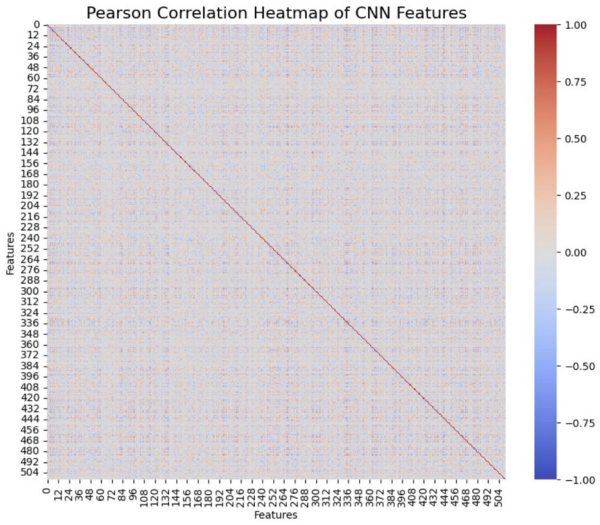

Figure 4. Pearson correlation heatmap of features

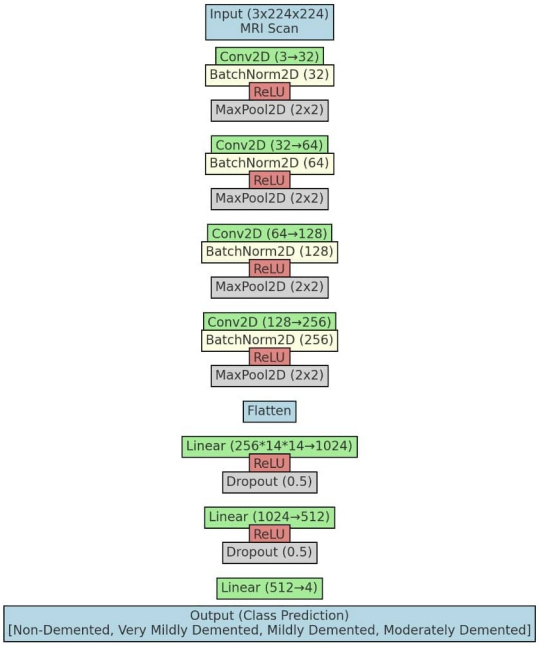

Figure 5. CNN architecture and model flow diagram

3.3 Model architecture and implementation

As depicted in Figure 5, the model used for classification was a deep CNN based on the ResNet50 architecture. This model was pretrained on the ImageNet dataset, allowing it to learn general features that are useful for medical image classification. The ResNet50 architecture consists of multiple convolutional blocks, which extract hierarchical features from the MRI images. The final fully connected layer of ResNet50 was modified to output

predictions for the four classes of Alzheimer's disease progression.

3.3.1 *Convolutional layers*

As illustrated in Figure 6, the ResNet50 architecture consists of multiple convolutional layers that progressively extract deeper and more abstract features from the input images. Each convolutional layer is followed by batch normalization, a ReLU activation function, and max-pooling to downsample the feature maps. This process reduces the spatial dimensions while preserving essential features, making it easier for the fully connected layers to classify the image into one of the four categories. The pretrained convolutional layers from ResNet50 allowed the model to efficiently extract meaningful features from the MRI images, which were then fine-tuned for Alzheimer's disease classification.

```
class EnhancedNeuralNetwork(nn.Module):
    def __init__(self):
        super(EnhancedNeuralNetwork, self).__init__()
        self.model = nn.Sequential(
            nn.Conv2d(3, 32, kernel_size=3, stride=1, padding=1),
            nn.BatchNorm2d(32),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Conv2d(32, 64, kernel_size=3, stride=1, padding=1),
            nn.BatchNorm2d(64),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Conv2d(64, 128, kernel_size=3, stride=1, padding=1),
            nn.BatchNorm2d(128),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Conv2d(128, 256, kernel_size=3, stride=1, padding=1),
            nn.BatchNorm2d(256),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2),
            nn.Flatten(),
            nn.Linear(256 * 14 * 14, 1024),
            nn.ReLU(),
            nn.Dropout(0.5),
            nn.Linear(1024, 512),
            nn.ReLU(),
            nn.Dropout(0.5),
            nn.Linear(512, 4)
        )

    def forward(self, x):
        return self.model(x)
```

Figure 6. Model architecture sample

3.4 Model training and evaluation

As described in the upcoming subsections, the training procedure involved minimizing the loss function, updating the model parameters through optimization, and monitoring the model's accuracy across multiple epochs.

3.4.1 *Loss function and optimizer*

The CrossEntropyLoss function was used (Figure 7) to process the multi-class classification task. This loss function is widely used for multi-class classification tasks, as it measures the difference between the predicted class probabilities and the actual class labels. Class weighting was applied to address the class imbalance in the dataset. Additionally, data augmentation techniques (e.g., rotations, flips, jitter) were used to enhance training data diversity, mitigating the effects of class imbalance (Section 3.2.3). The Adam optimizer

was chosen for updating model parameters. Adam adjusts the learning rate dynamically based on the first and second moments of the gradients, allowing the model to converge more quickly and effectively during training.

```
class_weights = [1.0 / 896, 1.0 / 64, 1.0 / 3200, 1.0 / 2240]
class_weights = torch.FloatTensor(class_weights).to(device)

loss_fn = nn.CrossEntropyLoss(weight=class_weights)
optimizer = torch.optim.Adam(model.parameters(), lr=1e-4)
```

Figure 7. Loss function and optimizer sample

3.4.2 *Training procedure*

The model was trained for 25 epochs, during which the training dataset was used to iteratively update the model parameters. The model.train() function (Figure 8) was used to enable the training mode, allowing all layers to update their parameters. For each batch, the CNN made predictions on the training data, and the difference between the predicted outputs and the actual labels was measured using the CrossEntropyLoss. This loss was back propagated through the network, and the Adam optimizer adjusted the model parameters based on the computed gradients to minimize the error.

```
# Training function
def train(dataloader, model, loss_fn, optimizer):
    model.train()
    size = len(dataloader.dataset)
    for batch, (X, y) in enumerate(dataloader):
        X, y = X.to(device), y.to(device)
        pred = model(X)
        loss = loss_fn(pred, y)
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
        if batch % 100 == 0 or batch == len(dataloader) - 1:
            loss, current = loss.item(), batch * len(X)
            print(f"loss: {loss:>7f}  [{current:>5d}/{size:>5d}]")
```

Figure 8. Training procedure sample

3.4.3 *Evaluation procedure*

At the end of each epoch, the model was evaluated on the test dataset using the model.eval() function, which disables gradient calculations to save computational resources and memory. The test data was passed through the network, and the predicted outputs were compared with the actual labels to compute the accuracy of the model (Figure 9). The average loss on the test data was also calculated to assess the generalization ability of the model on unseen data. The model's performance was tracked across all epochs, with the best-

performing model being saved based on the highest accuracy achieved on the test set.

```python
def test(dataloader, model, loss_fn):
    model.eval()  # Set the model to evaluation mode
    size = len(dataloader.dataset)
    num_batches = len(dataloader)
    test_loss, correct = 0, 0

    with torch.no_grad():  # Disable gradient computation for testing
        for batch, (X, y) in enumerate(dataloader):
            X, y = X.to(device), y.to(device)  # Move inputs to device
            pred = model(X)
            test_loss += loss_fn(pred, y).item()
            correct += (pred.argmax(1) == y).type(torch.float).sum().item()  # Count correct predictions

    # Calculate average loss and accuracy
    test_loss /= num_batches
    correct /= size
    accuracy = correct  # Accuracy as a fraction between 0 and 1

    print(f"Test Error: \n Accuracy: {(100 * accuracy):>0.1f}%, Avg loss: {test_loss:>8f} \n")

    return accuracy  # Return accuracy for comparison in training loop
```

Figure 9. Evaluation procedure sample

3.5 Model prediction and visualization

After completing the training process, the model was evaluated on unseen test images to validate its performance over the course of 25 epochs. During each epoch, the model's predictions were compared with the actual class labels from the test dataset to track its accuracy. The model's best performance was automatically saved whenever the test accuracy surpassed that of previous epochs, ensuring that the final saved model represented the optimal state achieved during training. A subset of test images was then selected to visualize the model's ability to classify the four stages of Alzheimer's disease (as described in Section 3.2.1). These predictions were made using the best-performing model, and the results, compared with the true labels, indicated how well the model generalized to unseen data, demonstrating its effectiveness in recognizing patterns across the stages of the disease. This final evaluation process, including the accuracy tracking and model saving, is shown in Figure 10.

```python
epochs = 25
best_accuracy = 0.0

for t in range(epochs):
    print(f"Epoch {t+1}\n-------------------------------")
    train(train_loader, model, loss_fn, optimizer)
    test_accuracy = test(test_loader, model, loss_fn)  # This now returns the accuracy

    # Save the best model if the current test accuracy is better
    if test_accuracy > best_accuracy:
        best_accuracy = test_accuracy
        torch.save(model.state_dict(), "best_alzheimers_model.pth")
        print(f"Saved best model with accuracy: {100 * best_accuracy:.2f}%")

print("Training complete!")
```

Figure 10. Prediction visualization sample

4. Results

4.1 Training accuracy

As previously mentioned, the model was trained over 25 epochs, and its performance was evaluated at each epoch in terms of prediction accuracy and average loss on the test dataset. In the initial epoch, the model started with a training accuracy of 61.2% and an average loss of 0.85. By the 25th and final epoch, the model achieved a peak training accuracy of 99.7% with an average loss of 0.01. The model was not trained further due to potential problems with overfitting as the training accuracy had reached its peak. The continuous trend of average loss and average accuracy across epochs can be further visualized in Figure 11, showing the increase in accuracy and the decrease in loss during the preliminary epochs.

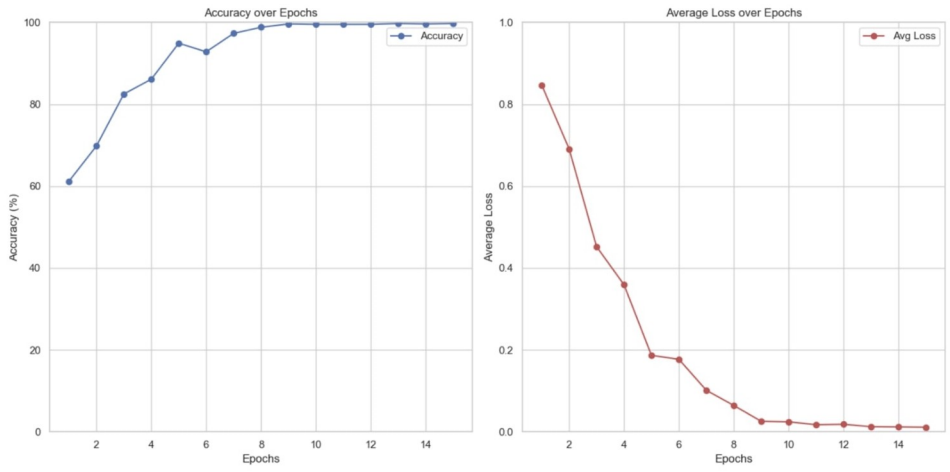

Figure 11. Accuracy and Loss Graphs over the course of training/over epochs.

4.2 Testing results and correlation

The model was finally tested on the unseen testing dataset (20%), and the predicted class labels were compared with the actual labels. The results, visualized as a bar graph in Figure 12, indicate that the model had achieved up to 88.79% testing accuracy and a test loss of 0.30, with the predictions closely matching the actual labels.

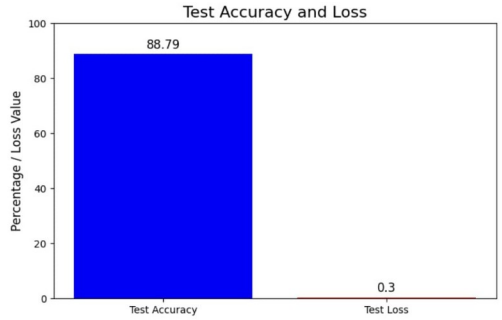

Figure 12. Test accuracy *vs* test loss comparison

Classification Report:

	precision	recall	f1-score	support
Mild_Demented	0.87	1.00	0.93	640
Moderate_Demented	1.00	1.00	1.00	640
Non_Demented	0.92	0.66	0.77	640
Very_Mild_Demented	0.77	0.87	0.81	640
accuracy			0.88	2560
macro avg	0.89	0.88	0.88	2560
weighted avg	0.89	0.88	0.88	2560

Figure 13. Classification report

Additionally, to further assess the model's performance, a confusion matrix and classification report were generated. The results (Figure 13) show that the model was implementable, with precision, recall, and F1-scores for each class demonstrating its efficacy in identifying multiple stages of dementia. Notably, the model classified the Moderate Demented category with no errors, achieving a precision, recall, and F1-score of 1.00, as well as an AUC of 1.0. The Mild Demented class followed, with an F1-score of 0.93 and AUC of 1.0. The Non-Demented category presented more challenges, with a F1 score of 0.77, but maintained a precision of 0.92. The Very Mild Demented category performed similarly, with a precision of 0.77 and an F1-score of 0.82. The confusion matrix indicated incorrect classification tendencies, particularly between the Non-Demented and Very Mild Demented classes, where several Non-Demented samples were misclassified as Very Mild Demented and *vice versa* (Figure 14). Regardless, the overall testing accuracy of 88.79% and the ROC curves with AUC values approaching 1.0 for all classes (Figure 15) indicated that the model generalized well, even on unseen data, providing robust predictions for all dementia stages.

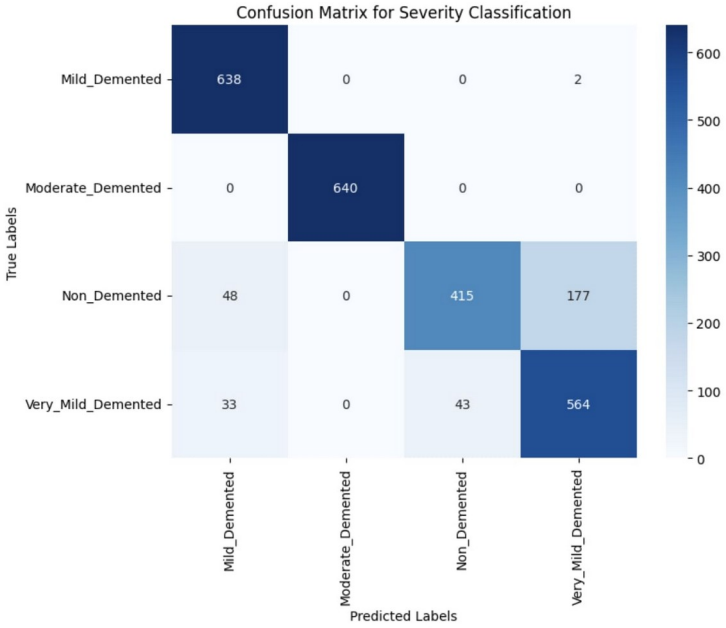

Figure 14. Confusion matrix

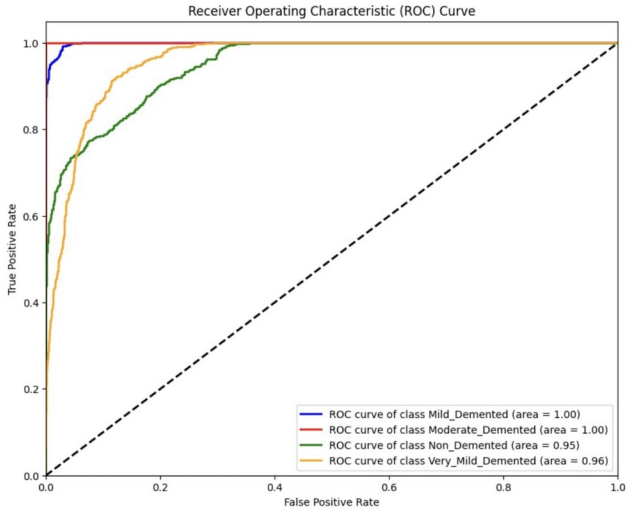

Figure 15. Receptor-Operator Characteristic Curves (AUC)

Finally, to further identify the relationship of key metrics such as epochs *vs* accuracy *vs* average loss, a correlation matrix was formed as seen in Figure 16. This matrix is visualized through a heatmap that depicts the relationship and interaction between these variables across the training process. As expected, a strong negative correlation was found between accuracy and average loss. Moreover, as the model accuracy increased across initial epochs, the average loss decreased, thus improving the model's performance over time.

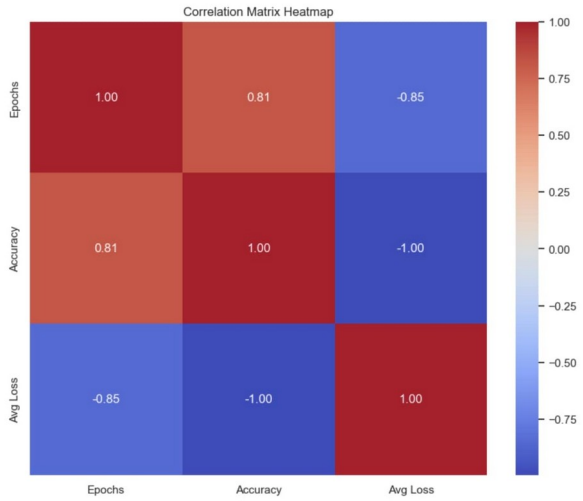

Figure 16. Correlation matrix heatmap

5. Conclusion

The proposed CNN model achieved 99.7% and 88.8% accuracy on the training and testing data respectively. The evident high correlation between increased epochs/batches and improved accuracy, together with the trend of a decrease in loss, provided evidence of both, the model reliability, and its potential for clinical application. This accuracy is greater than many of the current models within this domain. The ability to correctly classify even the most subtle difference in stages like the Very Mild Demented stage, renders the model useful for early diagnosis. Another advantage of the model is that predictive accuracy on different stages of Alzheimer's disease is well-balanced (as class imbalance is a common problem faced by many existent models).

5.1 Limitations and potential improvement

One potential concern was that the model was trained on only one large dataset, which may not present with enough variety found in the clinical setting. To increase the model's generalization capability, larger and more diverse datasets should be used with images from many more demographics and MRI machines. Future work might study the integration of multiple modalities, such as genetic, clinical, and other imaging modalities into the proposed model for increased diagnostic capabilities. Also, while the features extracted appeared relatively independent and diverse, were there to be highly correlated features (indicated by darker red or blue blocks in the Pearson Correlation Heatmap), those could potentially be reduced using techniques like Principal Component Analysis (PCA) to maximize generalization capability and model robustness.

Since the model was trained on a locally run computer with ~ 8 GB of RAM, the model training took an extended amount of time

compared to running this same model on a high-performance machine with greater memory and GPU capabilities. Ultimately, this may have resulted as a consequence from the processing of a very large dataset and complex computations.

Another significant consideration when applying machine learning to MRI images is the variability in the diagnostic accuracy of MRI itself, which typically ranges from 50% to 85%. This inherent variability may introduce some level of error in the ground truth labels used for training the model. Consequently, even though our model achieves relatively high accuracy (88.79% in testing), this performance is constrained by the accuracy of the input data, and errors from the human interpretation of MRI images may compound with errors introduced by the model. To mitigate this effect, future work should consider the integration of additional modalities (e.g., genetic or clinical data) to reduce reliance on potentially inaccurate human identified MRI labels. Furthermore, incorporating uncertainty quantification techniques in the model can help indicate predictions where the model's confidence is lower due to less reliable human interpreted input data.

5.2 Clinical applications

Beyond its presented use to classify the stages of Alzheimer's disease based on severity, the model could be embedded within other diagnostic tools; including cognitive testing and genetic screening; to return a comprehensive assessment of the condition of the patient. This could be done, for example, by merging the predictions made by the model on the basis of images with data from other modalities to provide an overall improvement in the diagnostic process through multidimensionality of views in disease, that may help in treatment strategies.

In addition to direct clinical applications, this model could be valuable in research settings, particularly in the development of new treatments for Alzheimer's disease. By providing a reliable means of categorizing disease stages, the model could be used to stratify patients in clinical trials, ensuring that treatments are tested on and compared with appropriately matched patient groups. This could lead to more accurate assessments of treatment efficacy and accelerate the development of new therapies.

Apart from its direct clinical applications, this model may also be highly valuable in the context of neurological research, such as in the development of new treatments against Alzheimer's disease. It would provide a valid way of classifying the stages of the disease and thus distinguish patients for clinical trials of treatments to be tested on appropriately matched patient groups, potentially allowing for more precise assessment of treatment efficacy and faster development of new therapies.

6. References

1. Marin A, Budson AE. (2023). Recent advances in understanding Alzheimer's disease: Diagnosis and management strategies. Fac Rev., 12, 24. https://doi.org/10.12703/r/12-24

2. Breijyeh Z, Karaman R. (2020). Comprehensive review on Alzheimer's disease: Causes and treatment. Molecules, 25(24), 5789. https://doi.org/10.3390/molecules25245789

3. Bouwman FH, Frisoni GB, Johnson SC, Chen X, Engelborghs S, Ikeuchi T, et al. (2022). Clinical application of CSF biomarkers for Alzheimer's disease: From rationale to ratios. Alzheimer's and Dementia, 18(6), 1066-1077. https://doi.org/10.1002/dad2.12314

4. Dakdareh SG, Abbasian K. (2023). Diagnosis of Alzheimer’s disease and mild cognitive impairment using convolutional neural networks. J Alzheimer’s Dis Rep, 8(1), 317-328. https://doi.org/10.3233/adr-230118

5. Lane CA, Hardy J, Schott JM. (2018). Alzheimer's disease. European Journal of Neurology, 25(1), 59-70. https://doi.org/10.1111/ene.13439

6. Rajmohan R, Reddy PH. (2017). Amyloid beta and phosphorylated tau accumulations cause abnormalities at synapses of Alzheimer’s disease neurons. J Alzheimer’s Dis, 57(4), 975-999. https://doi.org/10.3233/jad-160612

7. Miao J, Ma H, Yang Y, Liao Y, Lin C, Zheng J, et al. (2023). Microglia in Alzheimer’s disease: Pathogenesis, mechanisms, and therapeutic potentials. Frontiers in Aging Neuroscience, 15, 1181256. https://doi.org/10.3389/fnagi.2023.1201982

8. Tanzi RE. (2012). The genetics of Alzheimer disease. Cold Spring Harbor Perspectives in Medicine, 2(10), a006296. https://doi.org/10.1101/cshperspect.a006296

9. Tönnies E, Trushina E. (2017). Oxidative stress, synaptic dysfunction, and Alzheimer’s disease. J Alzheimer’s Dis, 57(4), 1105-1121. https://doi.org/10.3233/jad-161088

10. Krizhevsky A, Sutskever I, Hinton GE. (2012). ImageNet classification with deep convolutional neural networks. Proceedings of the 25th International Conference on Neural Information Processing Systems (NeurIPS 2012), 1097-1105. https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

11. Graves A, Mohamed A-R. Hinton G. (2013). Speech recognition with deep recurrent neural networks. Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP 2013), 6645-6649. https://doi.org/10.1109/ICASSP.2013.6638947

12. Chakraborty C, Pal S, Bhattacharya M, Dash S, Lee S-S. (2023). Overview of chatbots with special emphasis on artificial intelligence-enabled ChatGPT in medical science. Front Artif Intell, 6, 1237704. https://doi.org/10.3389/frai.2023.1237704

13. Carriero A, Groenhoff L, Vologina E, Basile P, Albera M. (2024). Deep learning in breast cancer imaging: State of the art and recent advancements in early 2024. Diagnostics, 14(8), 848. https://doi.org/10.3390/diagnostics14080848

14. Lenchik L, Heacock L, Weaver AA, Boutin RD, Cook TS, Itri J, et al. (2019). Automated segmentation of tissues using CT and MRI: A systematic review. Acad Radiol, 26(12), 1695-1706. https://doi.org/10.1016/j.acra.2019.07.006

15. Dixon D, Sattar H, Moros N, Kesireddy SR, Ahsan H, Lakkimsetti M, et al. (2024). Unveiling the influence of AI predictive analytics on patient outcomes: A comprehensive narrative review. Cureus, 16(5), e59954. https://doi.org/10.7759/cureus.59954

16. Aberathne I, Kulasiri D, Samarasinghe S. (2023). Detection of Alzheimer's disease onset using MRI and PET neuroimaging: Longitudinal data analysis and machine learning. Neural Regen Res, 18(10), 2134-2140. https://doi.org/10.4103/1673-5374.367840

17. Gong C, Jing C, Chen X, Pun CM, Huang G, Saha A, et al. (2023). Generative AI for brain image computing and brain network computing: A review. Front Neurosci, 17. https://doi.org/10.3389/fnins.2023.1203104

18. Amann J, Vetter D, Blomberg SN, Christensen HC, Coffee M, Gerke S, et al. (2022). To explain or not to explain?—Artificial intelligence explainability in clinical decision support systems. PLOS Digital Health, 1(2), e0000016. https://doi.org/10.1371/journal.pdig.0000016

19. Kumar, S. (2023). Alzheimer-MRI-Dataset. Kaggle. https://www.kaggle.com/datasets/sachinkumar413/alzheimer-mri-dataset