Evaluating Deep Learning Models for Early Alzheimer's Detection in Magnetic Resonance Imaging

Aatyanth Thimma Udayakumar [1]

Abstract

Alzheimer's Disease (AD) is a severe type of dementia that progressively impairs an individual's cognitive, behavioral, and motor abilities. Already affecting over 50 million individuals around the world, AD is expected to grow to affect over 150 million people by 2050. With no officially established cause, the severity of the issue emphasizes the dire need for reliable early detection and prevention systems. While there has been extensive research done on the applications of deep learning to this issue, this study aims to systematically compare 3 state-of-the-art deep learning architectures and their ability to accurately classify the different stages of Alzheimer's. Specifically, the study utilizes the OASIS-1 dataset (preprocessed by Ninad Aithal), which contains Magnetic Resonance Imaging (MRI) data for 416 subjects aged 18-96, to train and test a baseline CNN model, a ResNet model, and Vision Transformer (ViT) model. After thorough tuning of the models, the study established that the ResNet model was the most accurate model with $\sim$70% accuracy, followed by the ViT with $\sim$55% accuracy, and lastly the baseline CNN model with $\sim$45% accuracy. This study provides a reference point for measuring the effectiveness and accuracy of these models in the potential application for early Alzheimer's detection.

1. Introduction

This section provides a brief introduction to Alzhiemer's Disease, it's underlying causes, treatments, and the application of Deep Learning in Alzheimer's Detection.

[1]Department of Cognitive Science, University of California, San Diego, United States of America. Correspondence to: Aatyanth Thimma Udayakumar <athimmaudayakumar@ucsd.edu>.

1.1. What is Alzheimer's Disease

Alzheimer's Disease (AD) was first discovered in 1905 by German psychiatrist Alois Alzheimer after observing a 50 year old woman experiencing progressive memory loss, confusion, and other cognitive and behavioral changes (Hippius & Neundörfer, 2003). However, despite almost a century of intensive research on this debilitating disease, no underlying cause has been officially established in the scientific community. AD is a severe type of progressive dementia characterized by the presence of neuritic plaques and neurofibrillary tangles, primarily composed of amyloid-beta peptides, within the medial temporal lobe and neocortical regions of the brain (Breijyeh & Karaman, 2020). AD leads to severe behavioral and cognitive losses that are classified into 4 different stages: 1). The Presymptotic Stage, 2). Mild or Early-Stage AD, 3). Moderate AD, and 4). Severe or Late-Stage AD. The presymptotic stage, which can last for several years, marks the initial onset of symptoms, with patients experiencing mild memory loss and changes in their cerebral cortex and hippocampus areas. As the disease progresses through the subsequent stages, the patient develops more severe cognitive and behavioral loss until death caused by the complications arising from AD.

1.2. Causal Factors and Treatment

Although no underlying cause has been established for Alzheimer's, various factors have been implicated in its development including air pollution, diet, infections, cardiovascular diseases, obesity, and diabetes (Breijyeh & Karaman, 2020). However, the most significant risk factors by far for AD are age and genetics. AD predominantly affects older individuals, typically above the age of 60 years, and is referred to as Late Onset Alzheimer's Disease (LOAD). However, on rare occasions, individuals aged 30-50 may develop AD, a condition known as Early-Onset Alzheimer's Disease (EOAD), which accounts for only 1-6% of all AD cases (Breijyeh & Karaman, 2020). In either case of AD, a family history of the disease substantially increases the likelihood of an individual's development of AD, underlining the genetic predisposition of AD (Bekris et al., 2010). Due to the absence of a precise etiology of AD, AD treatments have been largely unsuccessful. Although 2 classes

of drugs–inhibitors to the cholinesterase enzyme and antagonists to *N*-methyl d-aspartate (NMDA)–are currently approved to be used for AD, these drugs primarily help mitigate the side effects of AD rather than serving as a direct cure to AD (Breijyeh & Karaman, 2020).

1.3. Deep Learning and Applications to Alzheimer's Detection

Currently, over 50 million individuals around the world have been diagnosed with AD, and this number is projected to exceed 150 million individuals by 2050. The growing prevalence of AD underscores the urgency to addressing the disease, sparking increasing interest in early detection and prevention of AD. While much of contemporary research regarding Alzheimer's prevention takes a biological approach–studying the genetic predisposition of AD, the impact of physical exercise, diet, and healthy lifestyle on AD, and more–recent AD early detection research has taken a different approach: Deep Learning (Breijyeh & Karaman, 2020; Jo et al., 2019). Deep learning is a subset of machine learning that emulates the brain's neuronal structure through Artificial Neural Networks (ANN). By processing information similar to how the brain does, ANNs are able to capture complex relationships that traditional machine learning models can not. These characteristics of ANNs open numerous possibilities of application from classification tasks to generative AI (Jo et al., 2019). In the context of early detection of AD, deep learning has emerged as a powerful tool, enabling fast and accurate classifications. While there has been extensive research done on the potential of deep learning in early AD detection, this study aims to systematically compare the efficiency of three leading architectures, namely Convolutional Neural Networks (CNNs), Residual Networks (ResNets), and Vision Transformers (ViTs).

2. Methods

2.1. Dataset Description

Dataset Description: The dataset used in the project was, in part, provided by the Open Access Series of Imaging Studies (OASIS) initiative by Washington University in St. Louis. Specifically, the study leverages the OASIS-1 dataset, which contains Magnetic Resonance Imaging (MRI) scans of 416 subjects, aged 18-96. The gender distribution of this dataset is approximately balanced, with a nearly equal ratio of male and female subjects. To prepare the dataset for deep learning, the raw dataset provided by OASIS (.img and .hdr files) was pre-processed by Ninad Aithal, and converted into 2-d jpg files by splicing the 3-d brain scans across the z axis into 256 pieces, from which splices 100-160 are present for each patient. A detailed overview of the pre-processing process is detailed in *Figure 1*. Furthermore, the scans of the patients are appropriately split into

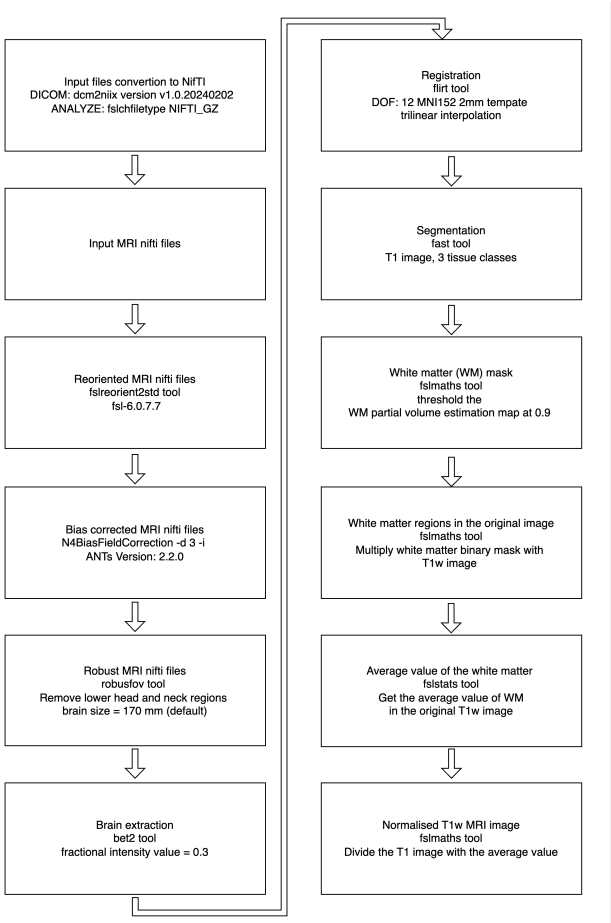

Figure 1. Detailed description of the pre-processing done to the raw OASIS-1 dataset by Ninad Aithal.

4 categories: Non-Demented, Very Mild Dementia, Mild Dementia, and Moderate Dementia, corresponding respectively to the stage of AD (Presymptomatic, Early Stage, and Moderate AD) based on the provided metadata and Clinical Dementia Rating values. This processed dataset was then provided through kaggle by Ninad Aithal (linked here).

2.2. Model Architectures

This study compares 3 different commonly used architectures in classification tasks: A CNN model, a ResNet model, and a Vision Transformer (ViT) model.

2.2.1. Baseline CNN Architecture

The CNN is implemented as a baseline model using a very simple architecture. Each convolution block in the baseline CNN consists of a convolutional layer, batch normalization, and average pooling. This experiment implements 2 such models: A baseline CNN for binary classification and a base-

line CNN for multiclass classification. The detailed model architectures are described in *Figures* 2 and *3* respectively.

```
AlzheimersNet(
  (conv1): Conv2d(1, 16, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
  (bn1): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (pool): AvgPool2d(kernel_size=3, stride=3, padding=0)
  (conv2): Conv2d(16, 32, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
  (bn2): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (conv3): Conv2d(32, 64, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
  (bn3): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (fc1): Linear(in_features=20736, out_features=512, bias=True)
  (dropout): Dropout(p=0.5, inplace=False)
  (fc2): Linear(in_features=512, out_features=256, bias=True)
  (fc3): Linear(in_features=256, out_features=128, bias=True)
  (fc4): Linear(in_features=128, out_features=2, bias=True)
)
```

Figure 2. The baseline CNN model architecture used for binary classification

```
AlzheimersNet(
  (conv1): Conv2d(1, 16, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
  (bn1): BatchNorm2d(16, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (pool): AvgPool2d(kernel_size=3, stride=3, padding=0)
  (conv2): Conv2d(16, 32, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
  (bn2): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (conv3): Conv2d(32, 64, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
  (bn3): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (conv4): Conv2d(64, 128, kernel_size=(5, 5), stride=(1, 1), padding=(2, 2))
  (bn4): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (fc1): Linear(in_features=4608, out_features=512, bias=True)
  (dropout): Dropout(p=0.5, inplace=False)
  (fc2): Linear(in_features=512, out_features=256, bias=True)
  (fc3): Linear(in_features=256, out_features=128, bias=True)
  (fc4): Linear(in_features=128, out_features=3, bias=True)
)
```

Figure 3. The baseline CNN model architecture used for multiclass classification

2.2.2. ResNet Architecture

The ResNet model builds upon the baseline CNN model by increasing the depth of the model, and consequently, its learning capacity. However, simply stacking more convolutional blocks exposes the model to the problem of vanishing gradients: a critical problem with deep networks whereby gradients become progressively smaller as they backpropagate through the layers. This problem greatly prolongs the training duration and in some extreme cases, completely halts training. Similarly, another problem arises as deep networks start converging: the degradation of training accuracy (He et al., 2016). Since simply adding more layers would only increase the training error and computational load, to overcome these issues, the ResNet implementation utilizes a concept known as residual blocks, hence the name residual net (ResNet). Essentially, a residual block, also known as a skip connection, directly adds the input of a convolution layer to the output of that layer before feeding the next layer, a process known as identity mapping (He et al., 2016). The goal is to simplify the training objective by learning the residual function (the difference between the input and desired output) instead of every transformation and connection. By doing so, residual blocks stabilize the gradient flow and improve training efficiency, even in extremely deep networks. Due to computational limitations, this study specifically employs the ResNet-18 architecture, which is the smallest variant with only 18 layers. The architecture of the ResNet-18 model is also slightly modified to only classify between 3 classes. The exact model architecture is described in *Figure 5*.

2.2.3. Vision Transformer Architecture

The last architecture explored in this study utilizes a variation of a transformer model for image classification, commonly known as a Vision Transformer (ViT). Unlike the previous architectures that utilize a CNN's convolutional architecture to extract spatial features from the image, the ViT utilizes a transformer based architecture, originally designed for natural language processing (NLP) tasks, to model spatial relationships within an image. This unique approach converts an input image into tokens, much like how words are represented using tokens in a standard transformer, by dividing the input image into 16x16 pixel patches that are flattened and linearly embedded into a vector. These patch embeddings are then passed into a series of standard transformer encoders. The self-attention mechanism of the encoders enables the model to learn both local and global spatial information simultaneously, a key difference between convolutional based approaches and transformer based approaches. A visual representation of the ViT is provided in *Figure 4*. This study employs the smallest ViT architecture (vit_b_16) from pytorch and is slightly modified to classify between 3 classes.

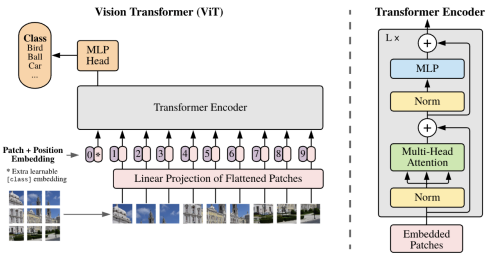

Figure 4. The Vision Transformer model architecture used for multiclass classification. Image provided from (Dosovitskiy et al., 2020)

```
AlzheimersResNet(
  (model): ResNet(
    (conv1): Conv2d(3, 64, kernel_size=(7, 7), stride=(2, 2), padding=(3, 3), bias=False)
    (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (relu): ReLU(inplace=True)
    (maxpool): MaxPool2d(kernel_size=3, stride=2, padding=1, dilation=1, ceil_mode=False)
    (layer1): Sequential(
      (0): BasicBlock(
        (conv1): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (relu): ReLU(inplace=True)
        (conv2): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      )
      (1): BasicBlock(
        (conv1): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        (bn1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (relu): ReLU(inplace=True)
        (conv2): Conv2d(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        (bn2): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      )
    )
    (layer2): Sequential(
      (0): BasicBlock(
        (conv1): Conv2d(64, 128, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
        (bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (relu): ReLU(inplace=True)
        (conv2): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        (bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (downsample): Sequential(
          (0): Conv2d(64, 128, kernel_size=(1, 1), stride=(2, 2), bias=False)
          (1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        )
      )
      (1): BasicBlock(
        (conv1): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        (bn1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (relu): ReLU(inplace=True)
        (conv2): Conv2d(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        (bn2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      )
    )
    (layer3): Sequential(
      (0): BasicBlock(
        (conv1): Conv2d(128, 256, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
        (bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (relu): ReLU(inplace=True)
        (conv2): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        (bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (downsample): Sequential(
          (0): Conv2d(128, 256, kernel_size=(1, 1), stride=(2, 2), bias=False)
          (1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        )
      )
      (1): BasicBlock(
        (conv1): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        (bn1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (relu): ReLU(inplace=True)
        (conv2): Conv2d(256, 256, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        (bn2): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      )
    )
    (layer4): Sequential(
      (0): BasicBlock(
        (conv1): Conv2d(256, 512, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), bias=False)
        (bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (relu): ReLU(inplace=True)
        (conv2): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        (bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (downsample): Sequential(
          (0): Conv2d(256, 512, kernel_size=(1, 1), stride=(2, 2), bias=False)
          (1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        )
      )
      (1): BasicBlock(
        (conv1): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        (bn1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (relu): ReLU(inplace=True)
        (conv2): Conv2d(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        (bn2): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
      )
    )
    (avgpool): AdaptiveAvgPool2d(output_size=(1, 1))
    (fc): Linear(in_features=512, out_features=3, bias=True)
  )
)
```

Figure 5. The ResNet model architecture used for multiclass classification

3. Experiments

3.1. Data Pre-Processing and Pipeline

3.1.1. Class Imbalance

The dataset used in this study exhibits a severe class imbalance: 266 participants in no the dementia class, 58 participants in the very mild dementia class, 21 participants in the mild dementia class, and only 2 participants in the moderate dementia class. This imbalance poses a significant challenge as the models will develop an inherent bias toward the overrepresented classes, leading to skewed results and poor performance on the underrepresented classes. To mitigate this issue, I restructured the classification task by combining certain classes, simplifying the problem.

- **Binary Classification Task:** The dataset was split into 2 classes
  - No Alzheimer's Class: Data from the no dementia participants
  - Alzheimer's Class: Data from the very mild, mild, and moderate dementia participants
- **Multiclass Classification Task:** The dataset was split into 3 classes
  - No Alzheimer's Class: Data from the no dementia participants
  - Mild Alzheimer's Class: Data from the very mild dementia participants
  - Advanced Alzheimer's Class: Data from the mild and moderate dementia participants

Even after combining the classes, there was a class imbalance. In the **binary classification task**, there was a 266-81 split for no Alzheimer's vs Alzheimer's respectively. In the **multiclass classification task**, there was a 266-58-23 split for the no Alzheimer's vs mild Alzheimer's vs advanced Alzheimer's classes. To address this, I created smaller, balanced datasets by down-sampling the majority classes. Specifically, I used the smallest class as the reference point to down-sample:

- **Binary Classification Tasks:** Each class was reduced to 81 participants
- **Multiclass Classification Task:** Each class was reduces to 23 participants

By down-sampling in this way, the dataset became more balanced, making it better suited for appropriate training of the different models.

3.1.2. Dataset Creation

The original dataset contained 60 splices per participant, each representing a 2D splice of the 3D MRI scans across the Z-axis. To reduce the computational overhead, I selected small subsets of these images for each participant. The exact number of splices used for each participant differed in the optimization stage of the model experimentation, but the initial dataset for each model implementation contained roughly 2500 images, evenly distributed across participants and classes. This dataset was then split into a training set and testing set using a roughly 70-30 split where the representation of each class was equal in both the training and

testing sets. It is also important to note that the training and testing sets were split on participants, not images, to ensure that the models don't see the same participants scans during training and testing. Lastly, there was padding added to the top and bottom of each image (originally 496x248 pixels) to square the image.

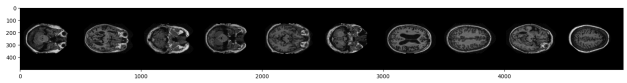

Figure 6. Example photos from dataset

3.2. Model Experimentation

The experimental design of the study follows a very simple approach: start with simple models and slowly add complexity. The initial experiment involves binary classification using a basic Convolutional Neural Network architecture (refer to *Figure 2* for the exact model architecture) . Building on this foundation, I extend the experiments to a multiclass classification using the same basic Convolutional Neural Network architecture. From there, I increase the model complexity by implementing a ResNet model for multiclass classification. Lastly, I explore a novel approach to image classification using a Vision Transformer (ViT). This incremental approach allows for systematic comparison across the different models' accuracies and performances.

3.2.1. Binary Classification using a simple CNN

The primary goal of this initial implementation was to get a foundational idea of the dataset and ensure the dataloaders, training loop, and testing loop were properly designed and worked as anticipated. These components would later be adopted in the setups for other model architectures. Nonetheless, this binary classification CNN model was able to achieve somewhat impressive results given its simplicity, providing a useful baseline. This model was trained for 10 epochs and there was minimal optimization to the model's hyperparameters or architecture. However, I initially intended to use an ADAM optimizer due to its adaptive learning capabilities, although early experiments yielded that the ADAM optimizer caused the model to get stuck in local minimas with little to no convergence. Hence, a CrossEntropy Loss function was used along with a Stochastic Gradient Descent (SGD) optimization algorithm (for subsequent model implementations as well) with a learning rate of 0.001 and momentum of 0.9. These were the binary classification CNN's performances:

- **Overall Testing Accuracy:** $\sim 75\%$
- **Individual Class Accuracies:**
  - **No Alzheimer's:** $\sim 65\%$
  - **Alzheimer's:** $\sim 85\%$

3.2.2. Multiclass Classification using a simple CNN

The next step in the experimental process was to build upon the binary classification by implementing a multiclass classification task using the same simple CNN architecture (refer to *Figure 3* for the exact model architecture). The introduction of multiple classes will better embody the different stages of Alzheimer's, encouraging the model to learn more nuanced relationships between the different stages and increasing the model's real-world applicability. This multiclass CNN will serve as the baseline model for comparison against more advanced architectures in the upcoming parts of the experimentation, providing a reference point for measuring the effectiveness and accuracy of the ResNet and ViT models. My initial experimentation with this model used the exact same architecture as the binary classification model which resulted in a 47.8% testing accuracy. Given that there were 3 classes, this model demonstrated that it was definitely learning some spatial features (distinguishable by the fact that the accuracy isn't close to 33%, which would indicate random guessing) but not nearly enough to match the performance of the binary classification model. As such, the main optimization steps for this model involved:

- Increasing the learning rate to allow the model to converge faster
- Adding an additional convolutional block to allow the model to learn more complex features during training

Even after the optimization steps, the model still had a very poor performance compared to the binary classification:

- **Overall Testing Accuracy:** $\sim 45\%$
- **Individual Class Accuracies:**
  - **No Alzheimer's:** $\sim 45\%$
  - **Mild Alzheimer's:** $\sim 40\%$
  - **Advanced Alzheimer's:** $\sim 60\%$

3.2.3. Multiclass Classification using a ResNet

Despite attempts to optimize the baseline CNN, the model's performance remained subpar, indicating its drawbacks in handling the inherent complexity in the multiclass classification of the 2D MRI images. This suggests that the simple CNN architecture lacks the capacity to model the complex intricacies needed to accurately classify the varying stages of AD.

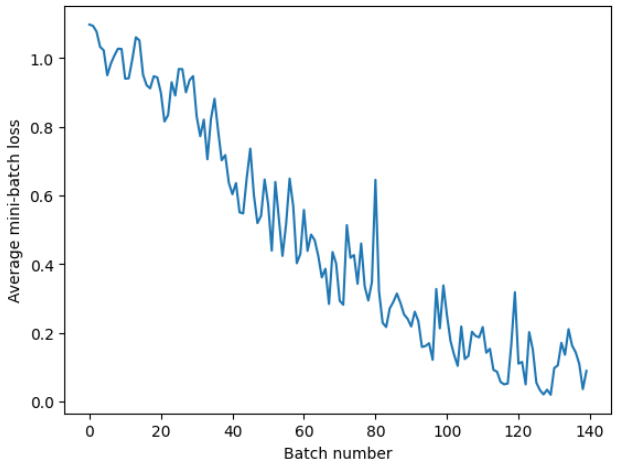

Figure 7. The loss plot for the multiclass CNN model

To address these limitations, the next step of experimentation involved implementing a ResNet model. Unlike the CNN, the ResNet architecture will greatly increase the model capacity by adding significantly more convolution layers while mitigating side-effects like the vanishing gradient problem and accuracy degradation through the use of residual blocks. This allows the model to efficiently learn the complex relationships inherently present in the data (refer to *Figure 5* for the exact model architecture). Additionally, the ResNet models employed in this experiment are already pre-trained on the Tiny ImageNet dataset, which provided the model with an existing collection of learned features. This transfer learning will allow the model to leverage existing feature representations instead of learning all features from scratch.

Given that the data being used in this experiment are gray-scale images, the initial implementation of the ResNet model directly modified the first convolution block to take in only 1 input channel and was then trained for 10 epochs using a CrossEntropy loss function and SGD optimizer. This model implementation produced an overall testing accuracy of 45%, on par with the baseline CNN. However, directly modifying the renders some of the pre-trained weights ineffective, essentially forcing the model to learn certain features from scratch. Subsequent experiments involved using a new approach whereby the same gray-scale image was passed into each input-channel, therefore mimicking a RGB image while still maintaining the gray-scale structure. Further optimization steps included:

- Decreasing the learning rate to 0.001
- Increasing the training data provided to the model
- Reducing the training epochs to prevent overfitting

These optimization techniques helped the ResNet significantly outperform the baseline CNN model:

- **Overall Testing Accuracy:** $\sim 70\%$
- **Individual Class Accuracies:**
  - **No Alzheimer's:** $\sim 60\%$
  - **Mild Alzheimer's:** $\sim 60\%$
  - **Advanced Alzheimer's:** $\sim 85\%$

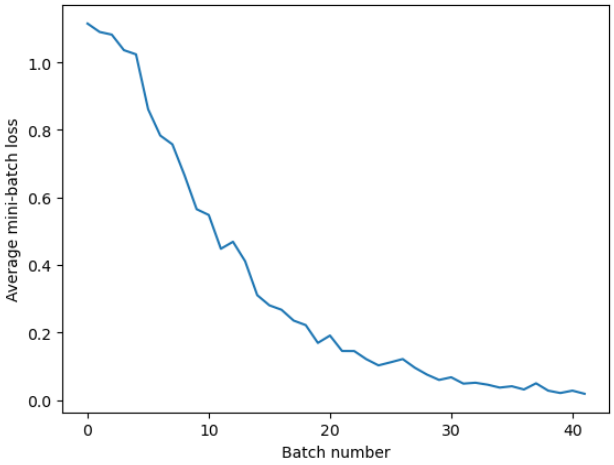

Figure 8. The loss plot for the ResNet model

3.2.4. Multiclass Classification using a Vision Transformer (ViT)

After demonstrating a significant improvement in performance using the ResNet implementation, the next experimental step was to evaluate how the ResNet performance compares against another competing, state-of-the-art classification system: Vision Transformers (ViT). ViTs have recently emerged as powerful architectures for classification tasks, leveraging the self-attention capabilities that revolutionized transformers in the field of natural language processing in image classification tasks. Unlike ResNet's use of convolution layers to extract features, ViTs break down images into patches and treat them as tokens, much like words in NLP tasks (refer to *Figure 4* for a visualization of the ViT architecture). This unique implementation gives ViTs a completely different view on the input images, learning both local and global features simultaneously. The initial implementation of the ViT yielded these results:

- **Overall Testing Accuracy:** $\sim 55\%$
- **Individual Class Accuracies:**
  - **No Alzheimer's:** $\sim 50\%$

  - **Mild Alzheimer's:** $\sim 50\%$
  - **Advanced Alzheimer's:** $\sim 70\%$

However, given the significantly higher computational overhead of the ViTs, the main optimization technique used was to increase the dataset size and lower the training epochs to prevent the model from overfitting during training. The results after optimization remained subpar:

- **Overall Testing Accuracy:** $\sim 55\%$
- **Individual Class Accuracies:**
  - **No Alzheimer's:** $\sim 75\%$
  - **Mild Alzheimer's:** $\sim 20\%$
  - **Advanced Alzheimer's:** $\sim 70\%$

These results also indicated a significant imbalance in individual class accuracies, likely due to the ViT not seeing enough data to clearly distinguish the Mild Alzheimer's class from other classes.

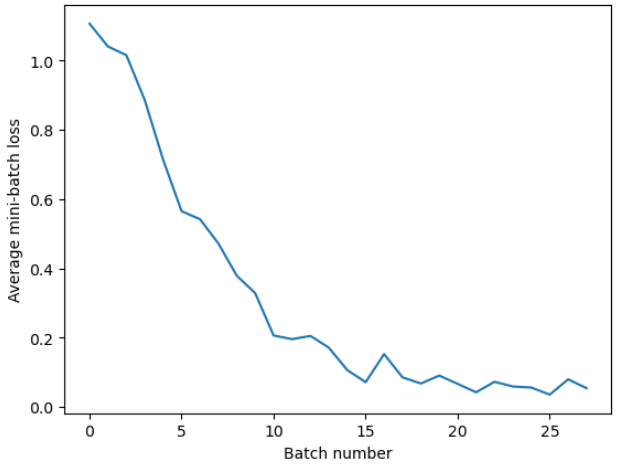

Figure 9. The loss plot for the ResNet model

3.3. Main Takeaway

The thorough experimentation of 3 different deep learning architectures–a baseline Convolutional Neural Network, a ResNet, and a Vision Transformer–yielded that the ResNet model was superior in performance and accuracy to its competitors, followed by the Vision Transformer model, and lastly the baseline Convolutional Neural Network. This performance hierarchy demonstrates the effectiveness of deeper convolution architectures in capturing the complex spatial relationships present in MRI images.

While ViTs have quickly grown as great competitors to traditional convolution based approaches in image classification tasks, this experimentation demonstrated one of the biggest drawbacks of Vision Transformers: the size of the dataset. Other research has indicated that ViTs can significantly outperform conventional convolution based models in extremely large datasets (Dosovitskiy et al., 2020). However, in the context of this study, the underlying issue with the ViT was likely, even though a few thousand images were used, the relatively small dataset utilized in training the model. Simply expanding the dataset would have likely yielded much better results. Similarly, further optimization of model hyperparameters and architecture could have likely improved the performances of the ResNet model and baseline CNN. Nonetheless, the optimization of hyperparameters and model architectures utilized in this study yielded the ResNet model as the most efficient and accurate model at classifying the different stages of AD through 2-D splices of MRI scans.

Table 1. Classification accuracies for the models in the study.

MODEL	TEST ACCURACY	COMPARED TO BASELINE
CNN	45± 5.0	BASELINE
RESNET	70± 5.0	SIGNIFICANTLY BETTER
VIT	55± 5.0	SLIGHTLY BETTER

4. Conclusion

Alzheimer's Disease is a debilitating condition that affects millions of individuals worldwide and continues to rapidly grow in prevalence. With no officially established etiology of AD, Alzheimer's prevention and early detection has become a top priority for healthcare researchers globally. While several research avenues show promising results, deep learning has garnered increasing attention and praise for its potential in early detection of Alzheimer's.

This study systematically compared 3 promising deep learning architectures in classifying 2D splices of MRI scans provided by the OASIS initiative at Washington University in St. Louis. Specifically, the study explored the potentials of a baseline CNN model, a ResNet model, and Vision Transformer model in capturing the inherent spatial complexity present in the MRI images. Through thorough tuning of the different models' parameters and architectures, the study established that the ResNet model was the best performing model with $\sim$70% accuracy, followed by the Vision Transformer model with $\sim$55% accuracy, and lastly the baseline CNN with $\sim$45% accuracy. Although these results show promise, further fine tuning of the model hyperparameters and architecture with more computational resources can yield significantly better outcomes.

Deep learning approaches in early detection and prevention of Alzheimer's is still a novel area of research, but the promising results demonstrated in this study

highlight its transformative potential. As we continue to refine model architectures and explore novel approaches, it is important to remember that the ability to detect Alzheimer's at an early stage could revolutionize healthcare outcomes and help millions of individuals from facing irreversible brain damage. Future work can expand on these initial investigations by combining other modes of data–cognitive test results, genetic information, family history, etc–to create hybrid deep learning architectures that can provide real-time, scalable diagnostic systems that have the power to change the trajectory of Alzheimer's care.

The code for this study can be found here.

Acknowledgments: "Data were provided [in part] by OASIS-1: Cross-Sectional: Principal Investigators: D. Marcus, R, Buckner, J, Csernansky J. Morris; P50 AG05681, P01 AG03991, P01 AG026276, R01 AG021910, P20 MH071616, U24 RR021382"

References

Bekris, L. M., Yu, C. E., Bird, T. D., and Tsuang, D. W. Genetics of alzheimer disease. *Journal of Geriatric Psychiatry and Neurology*, 23(4):213–227, 2010.

Breijyeh, Z. and Karaman, R. Comprehensive review on alzheimer's disease: Causes and treatment. *Molecules (Basel, Switzerland)*, 25(24), 2020.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv, abs/2010.11929*, 2020.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

Hippius, H. and Neundörfer, G. The discovery of alzheimer's disease. *Dialogues in Clinical Neuroscience*, 5(1):101–108, 2003.

Jo, T., Nho, K., and Saykin, A. Deep learning in alzheimer's disease: Diagnostic classification and prognostic prediction using neuroimaging data. *Frontiers in Aging Neuroscience*, 11, 2019.