RE-ADD-Net: Reproducing and Exploring the Effective Deep Learning Model for Early Detection of Alzheimer Disease in MRI Scans

Ryan Kim and Taohan Lin

Dr. Yilmaz Machine Learning Period 7 - Semester 2 Project

Thomas Jefferson High School for Science and Technology

May 20, 2024

Abstract

Fill in last

1 Introduction

Alzheimer's Disease (AD) is a neurological disorder that requires close medical attention and early intervention before symptoms such as memory loss, motor skill deficiencies, and delusion worsen. Due to how common it has gotten, with over 55 million cases in the world, non-invasive imaging techniques such as Computed Topography (CT), Positron Emission Topography (PET), and Magnetic Resonance Imaging (MRI) have emerged as popular tools to aid in the diagnosis of Alzheimer's Disease in patients.

Additionally, it can often be subjective for a radiologist to give a formal diagnosis for a patient given a scan, increasing the risk of a patient with an early stage of AD being dismissed. Therefore, machine learning algorithms have become a popular tool to classify and diagnose the patient's condition based on these medical images. The most popular algorithm used for image classification, especially within a medical setting, is the convolutional neural network (CNN), given it's robustness and precise ability to differentiate data into different classes.

In the field of medical artificial intelligence, it is ideal for the trained model to correctly classify every single unseen instance, so different architectures have been developed to improve the accuracies of these CNNs. For the purpose of our paper, the input to our algorithm is an image, which is an axial brain scan for a patient. We then use different kinds of CNN architectures and oversampling techniques to output a predicted brain condition, in an attempt to achieve the highest test accuracy for unseen images.

2 Related Work

The same dataset used in this paper was used by Ebrahimighahnavieh et al., where the ResNet V2 is combined with the Inception V4 model through residual connections [1]. The authors perform a series of experiments across different hyperparameters such as learning rate, optimizers, and activation functions, where the highest test accuracy received was 79.1%. A similar study using the same dataset was performed by Pradhan et al., where the authors used models VGG19 and DenseNet169, two state-of-the-art CNN architectures [2]. The VGG19 and DenseNet169 achieved accuracies of 88 and 87%, respectively.

Another similar approach was employed by Khan et al., but this time for the OASIS dataset, which is aggregated into the dataset of this paper [3]. A 12-layer CNN architecture was used with Leaky ReLU to avoid the vanishing gradient issue, given how long the CNN architecture was. The authors compared their models to Inception V3, Xception, MobileNetV2, and VGG19, all state-of-the-art CNN architectures. However, the authors' model achieved the highest accuracy of 97.75%. Loosely related, Islam and Zhang created a one-layer CNN model for 3D samples from the OASIS dataset and compared their model to InceptionV4 and ResNet, achieving a precision of 75% [4].

While all of the mentioned previous attempts were relatively successful, the class imbalance issue in the dataset was not addressed, potentially leading to biases in the model. However, all of the above approaches employ similar CNN architectures (VGG19, DenseNet, Inception, etc.), taking advantage of the robustness of these models and their high performance in multi-class classification tasks.

Fareed et al., employed the Synthetic Minority Oversampling Technique (SMOTE) - Tomek Links algorithm to oversample the minority class [5]. In their paper, they show a boost in different metrics such as testing accuracy, precision, and recall by applying the SMOTE-Tomek algorithm before training the models. We closely follow these authors' methodology throughout the paper, in addition to reproducing the results of their own CNN architecture (ADDNet), InceptionV2, DenseNet169, and VGG19 architectures. We also employ the SMOTE - Edited Nearest Neighbor (SMOTE - ENN) oversampling technique to see the effects on the model performance.

3 Dataset and Features

The dataset for this paper was collected from Kaggle [6]. Although Kaggle is an open-source platform that may not be the most reliable when it comes to real-world datasets, this specific dataset combined 2D MRI scans along the the class label information from the Alzheimer's Disease Neuroimaging Initiative (ADNI) and OASIS (Open Access Series of Imaging Studies) datasets, two respectable clinical studies in the field of AD imaging. The distribution of images for each class in this dataset is shown below:

Table 1: Class Distribution for Images

Class	# of Images
Non-Demented (ND)	3200
Mildly Demented (MD)	896
Very Mildly Demented (VMD)	2240
Moderately Demented (MOD)	64

For each class, non-demented (ND) means that the patient has no signs of cognitive impairment related to dementia. Mildly demented (MD) means that the patient often has memory loss, while very mildy demented (VMD) means that the patient has memory loss, personality changes, disorientation, and motor skill deficiencies. Moderate dementia (MOD) is where patients have symptoms that include everything for VMD patients but with harsh sleep disturbances as well as hallucinations and delusions.

For the train-validation-test split, the dataset is divided into 60%, 20%, and 20%, respectively.

4 Methods

4.1 Oversampling Techniques

4.1.1 Synthetic Minority Oversampling Technique (SMOTE)

To address the imbalanced dataset, the SMOTE algorithm is used a baseline to the two following algorithms to synthetically increase the number of images for the minority classes.

SMOTE, which is synthetic minority oversampling technique, generates new samples based on the class nearest neighbors. First, SMOTE algorithm chooses a random instance P in the minority class. Then, it calculates the Euclidean distance between this instance and its k nearest neighbors, similar to the KNN algorithm. The distance from P to each training point $Q = (y_1, y_2, \cdots, y_n)$ in the dataset is given by:

$$d(P,Q) = \sqrt{\sum_{j=1}^{n}(x_j - y_j)^2} = \sqrt{(x_1-y_1)^2 + (x_2-y_2)^2 + \cdots + (x_n-y_n)^2}.$$

After finding the k nearest neighbors, the algorithm multiples the difference with a random number between 0 and 1, and then adds the result to the minority class as a synthetic sample. This is repeatedly done until the minority class has an equal number of instances as the majority class.

4.1.2 SMOTE-Tomek Algorithm

The Tomek Links method, a downsampling technique, can be used in conjunction with the SMOTE algorithm to help clear any noisy or hard-to-differentiate instances. The criteria to identify Tomek Links is that (1) instance a's nearest neighbor is b, (2) instance b's nearest neighbor is a, and (3) a and b are part of different classes. Mathematically, let $d(x_i, y_j)$ denote the Euclidean distance between x_i and x_j, where x_i is part of the minority class and x_j is part of the majority class. If there is no sample x_k that satisfies the following conditions:

$$d(x_i, x_k) < d(x_i, x_j)$$

or

$$d(x_j, x_k) < d(x_i, x_j)$$

then the pair (x_i, x_j) is a Tomek Link.

When used in conjunction with SMOTE, at the end of each SMOTE iteration explained above, a random data instance is chosen from the majority class. If this data instance's nearest neighbor is part of the minority class, then we remove the Tomek Link.

4.1.3 SMOTE-ENN Algorithm

The Edited Nearest Neighbors (ENN) algorithm is similar to the Tomek Links algorithm in that it computes the Euclidean distance, but it does so for every instance first. If the majority class from the random instance's k nearest neighbors is different from the class of the instance's class, that pair is deleted. As a default, ENN uses k=3 nearest neighbors.

When used in conjunction with SMOTE, at the end of each SMOTE iteration explained above, every instance goes through the ENN algorithm and any pairs that meet the criteria are deleted.

4.2 CNN Architectures

Both SMOTE-Tomek and SMOTE-ENN, as well as no oversampling technique, were used for our study, effectively creating 3 different datasets. For our study, we employed four different CNN architectures: ADDNet, InceptionV2, DenseNet169, and VGG19.

ADDNet was proposed by Fareed et al., which is comprised of four convolutional blocks, with each block having a Rectified Linear Unit (ReLU) activation function and a 2D average pooling layer. Afterwards, there are two dropout layers, a flattening layer, and two dense layers, with the model being concluded by a SoftMax classification layer. In general, convolutional layers are meant to extract features such as patterns, edges, and curves in images, pooling layers are meant to reduce the dimensionality of the matrix, and the choice of ReLU

for the activation function is meant to decrease the model's proneness to the vanishing gradient problem, in which weights at the front of the model are changed less due to the nature of back-propagation.

InceptionV2 is an improved version of the original Inception model, in which "inception modules" are employed, with multiple convolutional layers of different kernel sizes (1x1, 3x3, 5x5) are used along with max-pooling layers. This is an attempt to capture multiple spacial scales efficiently. Additionally, the model employs dropout and batch normalization techniques to prevent overfitting. Global average pooling is used instead of a SoftMax layer by averaging the features across spatial dimensions, resulting in a compact feature representation that can be directly fed into the final SoftMax layer for classification to reduce the number of parameters.

DenseNet169 is unique because it introduces dense connections between each convolutional layer. Dense layers are layers where each neuron is connected to every neuron in the previous layer, better enhancing gradient flow and parameter efficiency. The model ends with a flattening layer and a final dense layer before feeding the results into a SoftMax layer for classification. This specific architecture has 169 layers and is popular for image classificaiton tasks.

The last CNN architecture used is VGG19, which consists of a series of convolutional layers, and each layer is followed by a ReLU activation function to reduce vanishing gradient. It has 16 convolutional layers with kernel sizes of 3x3 for each layer, which are each followed by a max-pooling layers. Finally, it is concluded with three fully-connected layers and a SoftMax classifer. While simple, it is a popular choice in transfer-learning tasks.

4.3 Hyperparameters

The hyperparameters are listed below that are consistent with every single model architecture and sampling technique:

Table 2: List of Hyperparameters

Hyperparameter Name	Hyperparameter Type
Optimizer	Stochastic Gradient Descent
Learning Rate	0.01
Batch Size	8
Epochs	40
Callback Function	ReduceLRonPlateau
Hidden Layer Activation Function	ReLU

The callback function ReduceLRonPlateau monitors a specific validation metric such as accuracy or loss and then keeps track of the number of epochs since this metric last improved significantly. This is controlled by a parameter called patience, and in our study, the patience

was set to 2 epochs. When the metric plateaus, the learning rate is reduced by a factor to help navigate the parameter space more effectively and potentially escape from local minima.

5 Results and Discussion

We analyze the accuracy of ADD-Net, DenseNet, VGG19, and InceptionV2 networks trained on the Kaggle dataset as is, with SMOTETOMEK applied and SMOTE-ENN applied. We examine the following variables: accuracy, precision, recall, and area under curve. Confusion matrices for all models will also be given. These variables are defined by:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Number of Total Predictions}}$$

$$\text{Precision} = \frac{TP}{TP + FP} \qquad \text{Recall} = \frac{TP}{TP + FN}$$

where TP represents the number of true positive classifications, FP represents the number of false positive classifications, and FN represents the number of false negative classifications. Area under curve (AUC) refers to the area under the receiver operating characteristic curve, which is a plot comparing the false positive rate (FPR) on the x-axis and true positive rate (TPR) on the y-axis for varying classification thresholds. These are defined by:

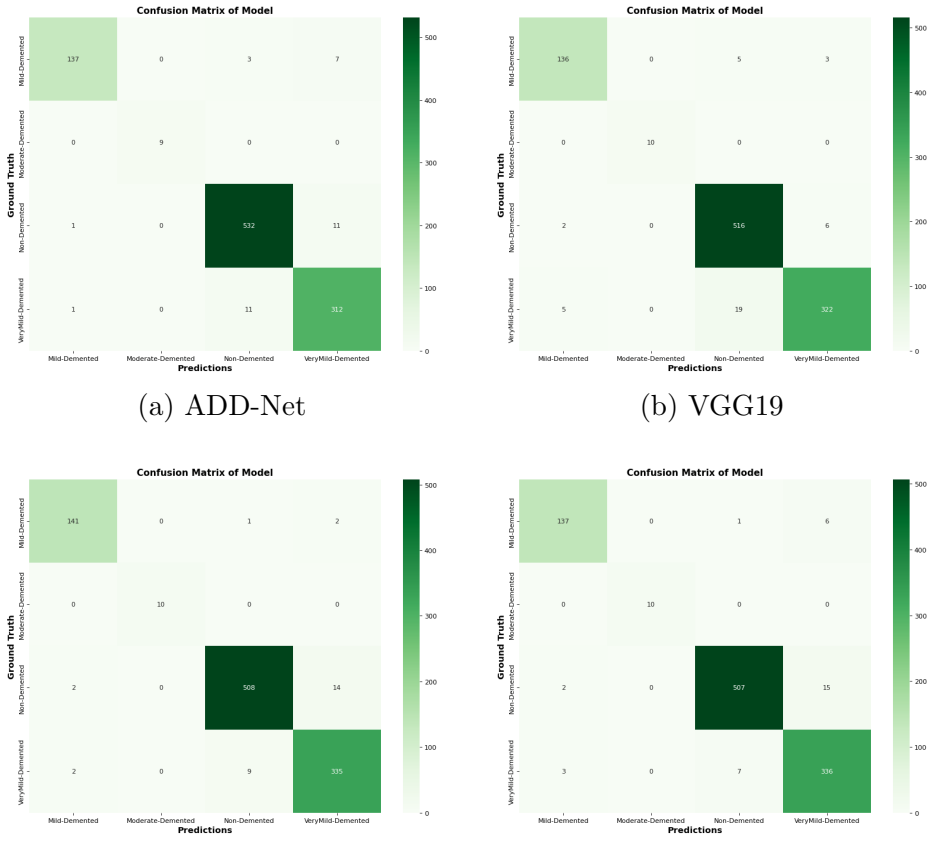
$$\text{TPR} = \frac{TP}{TP + FN} \qquad \text{FPR} = \frac{FP}{FP + TN}$$

such that TN represents the number of true negative classifications.

Figs 1, 2, and 3 show the confusion matrix of all models tested when no additional sampling technique, SMOTETomek, and SMOTE-ENN are used respectively. The addition of SMOTETomek and SMOTE-ENN are visually noticeable as we can see the balancing of class counts through the correct classification diagonal as the models are generally accurate.

We can see from the confusion matrices that for all of the models tested, it is most difficult to differentiate between the VMD and ND. Medically, these are two very different cases, one of which warrants immediate medical attention while the other requires none, so it is incredibly important for models to be accurate in this differentiation. We can also see that the models react differently when dataset balancing techniques are implemented. In the dataset with no additional sampling technique, the model confusion matrices are quite similar. However, when SMOTETomek or SMOTE-ENN are added, the VGG19 model gets noticeably worse, particularly in differentiating between the VMD and ND classes.

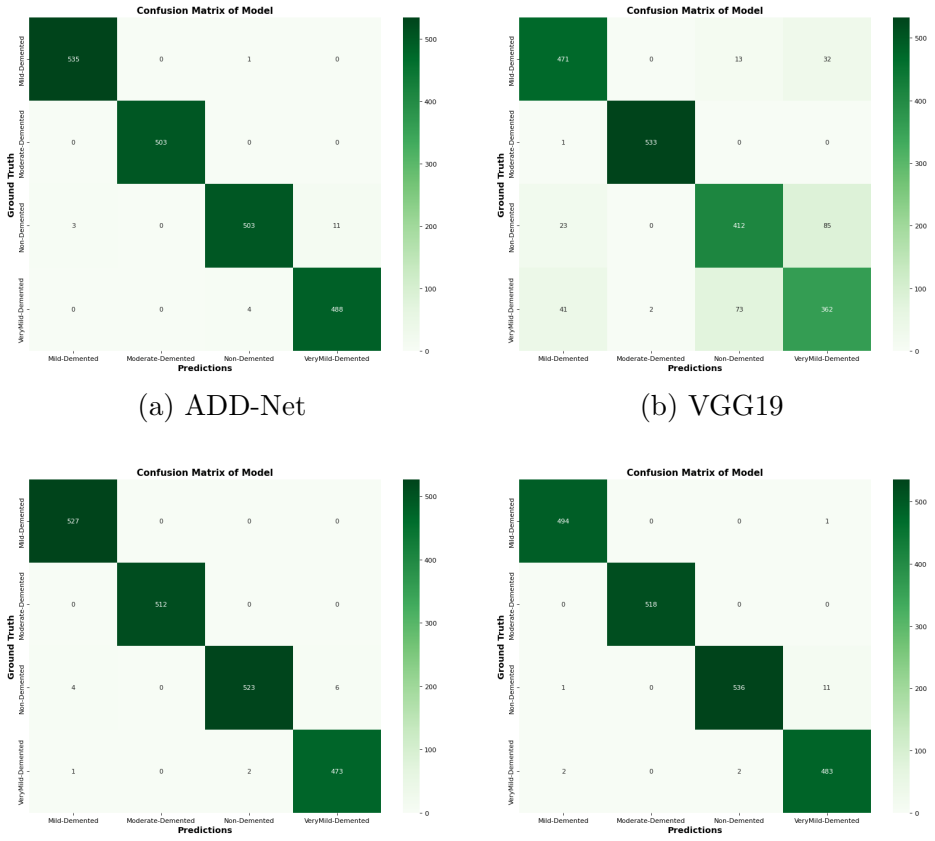
Confusion matrices were not provided by Fareed et al., so it is not possible to check whether these results are consistent with those of the authors. We proceed to the measures that were provided, beginning with accuracy.



(a) ADD-Net (b) VGG19

(c) InceptionV2 (d) DenseNet

Figure 1: Confusion matrices for models with no additional sampling technique



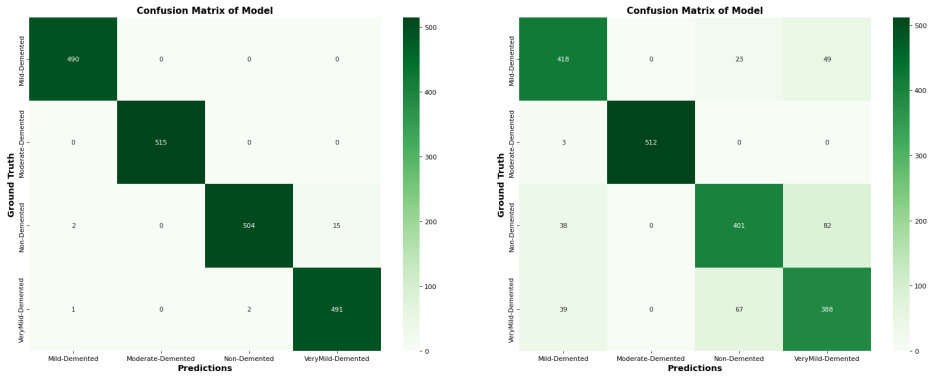
(a) ADD-Net

(b) VGG19

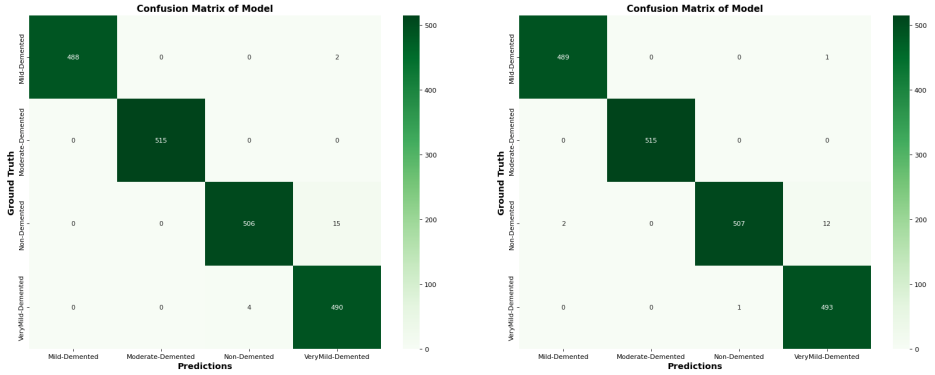
(c) InceptionV2

(d) DenseNet

Figure 2: Confusion matrices for models with SMOTETomek method



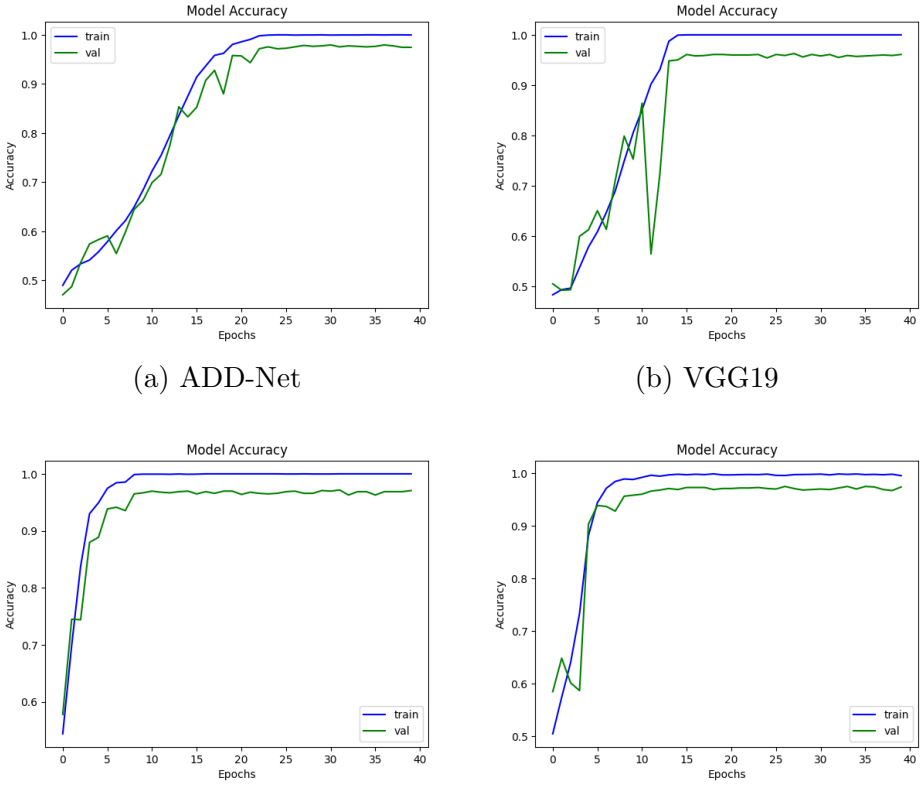
(a) ADD-Net (b) VGG19



(a) InceptionV2 (b) DenseNet

Figure 3: Confusion matrices for models with SMOTE-ENN method

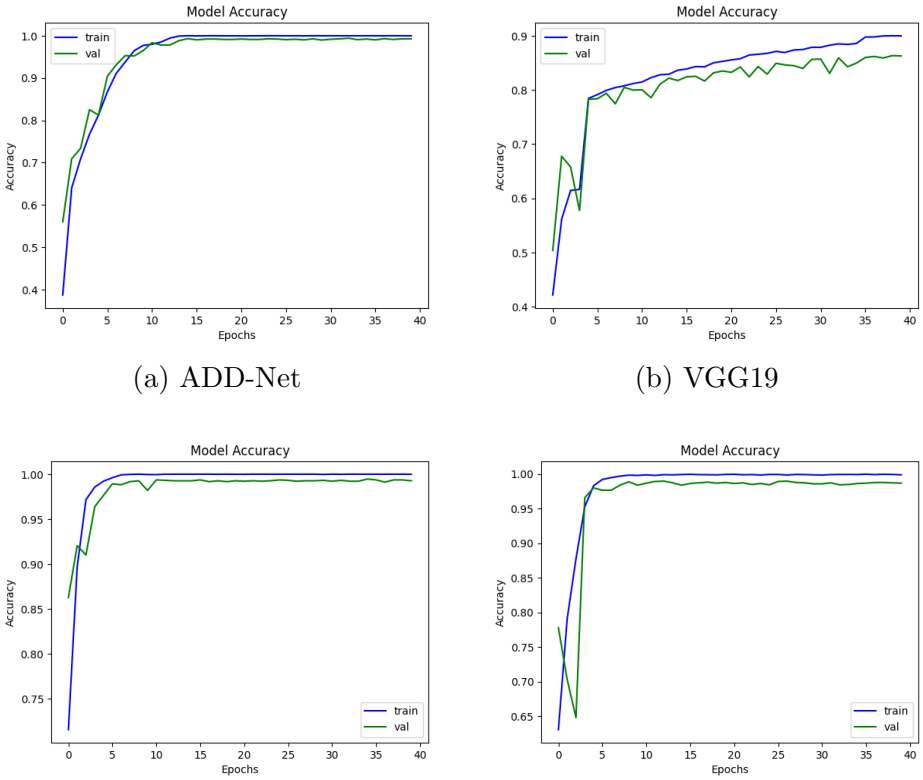
Figs 4, 5, and 6 show the accuracy as a function of epochs for both training and validation data for all models tested.



(a) ADD-Net (b) VGG19

(c) InceptionV2 (d) DenseNet

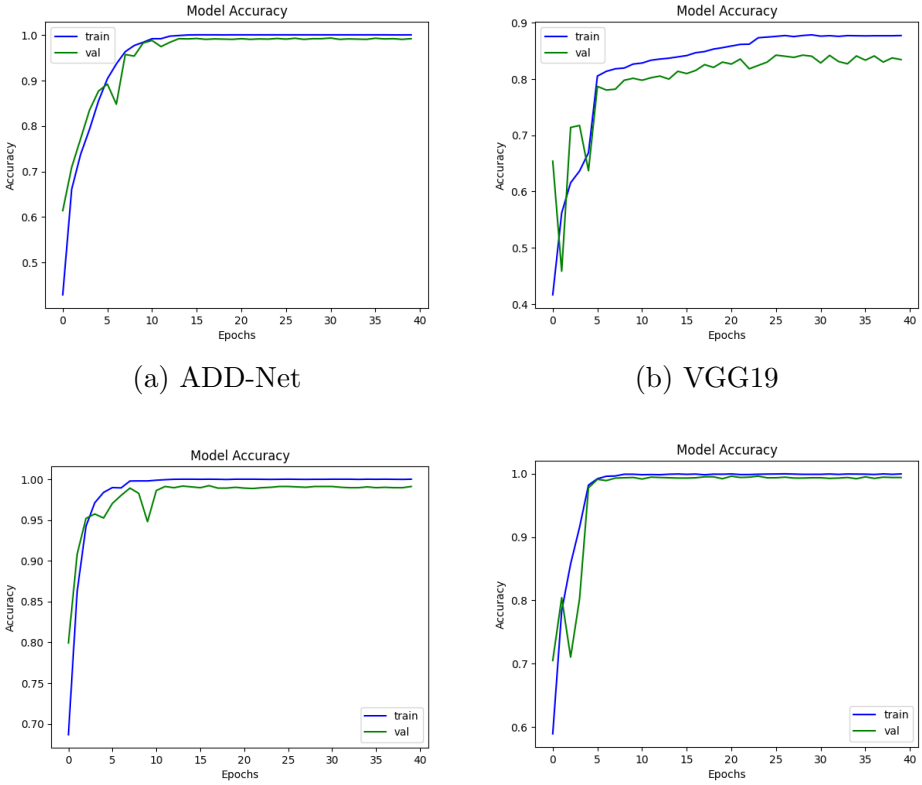
Figure 4: Accuracy plots for models with no additional sampling technique



(a) ADD-Net (b) VGG19

(c) InceptionV2 (d) DenseNet

Figure 5: Accuracy plots for models with SMOTETomek method



(a) ADD-Net (b) VGG19

(c) InceptionV2 (d) DenseNet

Figure 6: Accuracy plots for models with SMOTE-ENN method

The summary of all of the testing metrics can be found in Tables 3, 4, 5, and 6.

Table 3: Test Accuracies (%)

	No Oversampling	SMOTETomek	SMOTE-ENN
ADDNet	96.679688	99.072266	99.009901
VGG19	96.093750	86.816406	85.099012
InceptionV2	97.070312	99.365234	98.960394
DenseNet	96.679688	99.169922	99.207920

Table 4: Test AUCs (%)

	No Oversampling	SMOTETomek	SMOTE-ENN
ADDNet	99.320585	99.849749	99.912453
VGG19	99.587977	97.869712	96.621317
InceptionV2	99.904239	99.897748	99.721158
DenseNet	99.584973	99.900264	99.893767

Table 5: Test Precisions (%)

	No Oversampling	SMOTETomek	SMOTE-ENN
ADDNet	96.679688	99.071813	99.058938
VGG19	96.278161	87.375748	85.635638
InceptionV2	97.154075	99.365234	98.960394
DenseNet	96.963763	99.169922	99.257058

Table 6: Test Recalls (%)

	No Oversampling	SMOTETomek	SMOTE-ENN
ADDNet	96.679688	99.023438	99.009901
VGG19	95.996094	85.839844	84.702969
InceptionV2	96.679688	99.365234	98.960394
DenseNet	96.679688	99.169922	99.207920

6 Conclusion

Our experiments show that the proposed ADD-Net in Fareed et al. is a strong classification model for Alzheimer's detection. However, unlike as claimed in the paper, the model is not superior to other state-of-the-art models despite being tailored specifically for ths Alzheimer's classification task. We found that in terms of all the metrics tested (accuracy, AUC, precision, and recall), DenseNet outperforms ADD-Net using the SMOTETomek balancing method. We were actually able to achieve a higher accuracy of 99.07% under SMOTETomek than the 98.63% statistic given in the paper, but the accuracies of the other tested models were also higher. We note that exact replication of the results may not be possible as differing initial weights may have been used, leading to differing training results.

In addition, we conclude that SMOTE-ENN does not offer significant improvements over SMOTETomek for the task of Alzheimer's classification. When SMOTE-ENN is applied over SMOTETomek, many of our model metrics actually decrease, and those that increase do not increase by a significant amount. These deviations are small in scale however, meaning that SMOTE-ENN is still quite effective in solving the class imbalance problem in Alzheimer's detection. Additional investigation could be put into modifying the ADD-Net structure to improve it classification efficacy. Fareed et al. had attempted to create a small model to reduce parameter counts compared to common transfer learning models, but efficacy may be improved if more parameters are allowed. Also, SMOTE-ENN and SMOTETomek are just two solutions to the class inbalance problem, and others may lead to better results for some of all of the models tested in this paper.

7 Contributions

Author Ryan Kim was responsible for the Introduction, Related Work, and Methods sections. Author Taohan Lin was responsible for the Abstract, Results and Discussion, and Conclusion sections. Both authors contributed to formulating the project idea of using CNN to perform Alzheimer's classification, the coding of all the algorithms, writing of this paper, and the creation of the presentation.

References

[1] M. A. Ebrahimighahnavieh, S. Luo, and R. Chiong, "Deep learning 760 to detect Alzheimer's disease from neuroimaging: A systematic litera- 761 ture review," Comput. Methods Programs Biomed., vol. 187, Apr. 2020, 762 Art. no. 105242.

[2] A. Pradhan, J. Gige, and M. Eliazer, "Detection of Alzheimer's disease 823 (AD) in MRI images using deep learning," Int. J. Eng. Res. Technol. 824 (IJERT), vol. 10, pp. 580–585, Apr. 2021.

[3] S. H. Khan, M. Hayat, M. Bennamoun, F. A. Sohel, and R. Togneri, "Cost847 sensitive learning of deep feature representations from imbalanced data," 848 IEEE Trans. Neural Netw. Learn. Syst., vol. 29, no. 8, pp. 3573–3587, 849 Aug. 2017.

[4] J. Islam and Y. Zhang, "Brain MRI analysis for Alzheimer's disease diag834 nosis using an ensemble system of deep convolutional neural networks," 835 Brain Informat., vol. 5, no. 2, pp. 1–14, 2018.

[5] M. M. S. Fareed et al., "ADD-Net: An Effective Deep Learning Model for Early Detection of Alzheimer Disease in MRI Scans," in IEEE Access, vol. 10, pp. 96930-96951, 2022, doi: 10.1109/ACCESS.2022.3204395.

[6] Modified Alzheimer's Dataset. Kaggle. (2022). https://www.kaggle.com/datasets/shahidzikria/alz-dataset