Classification of Unbalanced Alzheimer's Disease MRI Images using Transfer Learning

Sina Ghofrani Majelan (Student ID: 40207226) Fatemeh Akbari (Student ID: 40213781)

https://github.com/Image-Processing-Winter2022/Alzheimer-s-Disease

***Abstract*—Alzheimer's disease (AD) is a type of brain disorder that causes the patient to undergo mental analysis. Of course, with age, it is a little forgetful in normal people, but in AD, the problem progresses. One-eighth of people 65 and older have this destructive state of dementia. AD causes nerve cell death and tissue loss throughout the brain. As the disease progresses, the brain tissue shrinks, and the areas containing cerebrospinal fluid become larger. AD damage impairs a person's memory, speech, and perception. Therefore, it is very important to diagnose Alzheimer's in the early stages. So, according to the importance of classifying AD, we have done comprehensive research in this field.**

In this project, we used Transfer Learning to classify MRI images of the Kaggle Alzheimer's dataset. Since the Kaggle Dataset is unbalanced, we used weighted categorical cross-entropy instead of regular categorical cross-entropy to force the network to learn different classes. Additionally, due to the mentioned problem, we use AUC as the metric of the classifier's performance. Solving an unbalance problem in a medical task is one of the important aspects of this research.

We tried different hyperparameter settings and three different pre-trained models as the feature descriptor, including NASNet Mobile, VGG16, and ResNet. Additionally, we adapt the SE blocks as an attention mechanism in the classifier of the network. This block enables the network to select more discriminative features by weighting each channel adaptively.

As for the research evaluation, we used some standard metrics, including F1-score, Accuracy, Precision, Recall, and AUC. We also plotted the attention maps and feature maps of different network layers in the experimental results section of this report.

***Index Terms*—Alzheimer's disease, transfer learning, attention mechanism, Kaggle Alzheimer's Dataset**

I. Introduction

Alzheimer's disease is a type of brain disease. It is also a degenerative disease, meaning that it becomes worse with time. Alzheimer's disease is thought to begin 20 years or more before symptoms arise with small changes in the brain that are unnoticeable to the person affected. [1] Only after years of brain changes do individuals experience noticeable symptoms, such as memory loss and language problems. Symptoms occur because nerve cells (neurons) in parts of the brain involved in thinking, learning, and memory (cognitive function) have been damaged or destroyed. Individuals typically live with Alzheimer's symptoms for years. Over time, symptoms tend to increase and start interfering with individuals' ability to perform everyday activities. At this point, the individual is said to have dementia due to Alzheimer's disease, or Alzheimer's dementia. [2]

As the disease progresses, neurons in other parts of the brain are damaged or destroyed. Activities that used to be core to the individual's identity, such as planning family events or participating in sports, may no longer be possible. Eventually, neurons in parts of the brain that enable a person to carry out basic bodily functions, such as walking and swallowing, are affected. People in the final stages of Alzheimer's disease are bed-bound and require around-the-clock care. Alzheimer's disease is ultimately fatal. [3] Disorders such as Alzheimer's disease (AD) and multiple sclerosis associated with the brain can be identified using MRI. [4] Magnetic resonance imaging (MRI) is used to analyze the anatomical structures of the brain due to its high spatial resolution and ability to contrast soft tissue. It is known that MRI is generally associated with fewer health risks compared to other modalities like computed tomography (CT) and positron emission tomography (PET). [5] It can be challenging for clinicians to analyze large and complex MRI datasets and to extract important information manually. Moreover, due to various inter- or intra-operator variability issues, manual analysis of brain MRI is time-consuming and vulnerable to errors. [6] Hence, it is necessary to develop an automated classification method to provide accurate results with high Confidence.

Classification is one of the most frequently encountered decision-making tasks of human activity. A classification problem occurs when an object needs to be assigned to a predefined group or class based on a number of observed attributes related to that object. Neural networks have emerged as an important tool for classification. [7]

The advantage of neural networks lies in the following theoretical aspects. First, neural networks are data-driven self-adaptive methods. They can adjust themselves to the data without any explicit specification of functional or distributional form for the underlying model. Second, they are universal functional approximators in that neural networks can approximate any function with arbitrary accuracy. [8] Since any classification procedure seeks a functional relationship between the group membership and the attributes of the object, accurate identification of this underlying function is

doubtlessly important. Third, neural networks are nonlinear models, which makes them flexible in modeling real-world complex relationships. Finally, neural networks are able to estimate the posterior probabilities, which provides the basis for establishing classification rules and performing statistical analysis. [9] On the other hand, the effectiveness of neural network classification has been tested empirically. Neural networks have been successfully applied to a variety of real-world classification tasks in industry, business, and science. [10] The goal of this work is to find a suitable model which facilitates Transfer Learning, allowing the classification of new datasets with acceptable accuracy. A common goal of transfer learning methods is to discover representations from previous tasks that make learning a future related task possible with few examples. Existing methods for transfer learning often learn a prior model or linear manifold over classifier parameters, discover a sparse set of common features, or use a representation based on classifier outputs from related tasks, but do not generally take advantage of unlabeled data. [11] We tried different hyperparameter settings and three different pre-trained models as the feature descriptor, including NASNet Mobile, VGG16, and ResNet.

- **NASNet Mobile:** Neural architecture search (NAS) is an algorithm that looks for the best neural network architecture by defining a set of building blocks that can possibly be used for constructing the whole network. In the NAS algorithm, a controller, recurrent neural network (RNN), samples these building blocks, putting them together to create some kind of end-to-end architecture. In the NASNet, the NAS algorithm discovers how to put predetermined building blocks together.
  NASNet has two different architectures, namely, Large and Mobile. The NASNet Mobile has significantly fewer parameters compared to the Large, only 4 million in comparison with the 84 million for the Large. This low number of parameters is one of the major benefits of our utilized architecture. [12]

- **VGG16:** The VGG-16 network was trained on the ImageNet database. Because of the extensive training that the VGG-16 network has undergone, it gives excellent accuracies even when the image data sets are small. [13]

- **ResNet:** ResNet is the short form for Residual Network shows in Fig. 1. Over the years deep convolutional neural networks have made a series of breakthroughs in the field of image recognition and classification. Going deeper to solve more complex tasks and to improve classification or recognition accuracy has become a trend. But, training deeper neural networks has been difficult due to problems such as vanishing gradient problem and degradation problem. Residual learning tries to solve both these problems. [14]

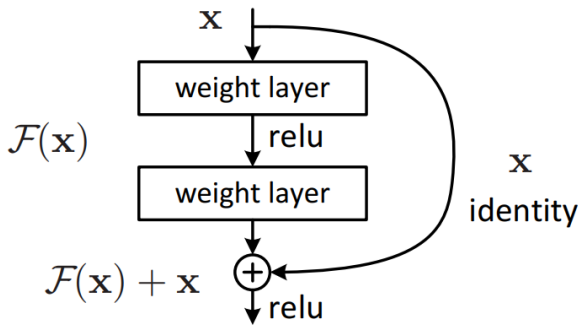

Fig. 1. Residual learning: a building block. Image taken from [14]

II. Related Work

In [15], five stages were used, where images were pre-processed and segmented into GM, WM and CSF in the first stage. The GM segmented ROIs were used to build similarity matrices in the second stage whereas, statistical features were extracted in the third stage. In the final two stages, statistical features were combined with clinical data i.e., functional activities questionnaire (FAQ) and support vector machine (SVM) classifier was used to classify data in AD vs normal group. In [16], a 3D displacement-field estimation was used for classification of AD and normal subjects. Feature selection was performed using three methods including Bhattacharyya distance, student t-test and Welch's t-test. The data was classified using SVM classifier and achieved a classification accuracy of 93.05%. Local and global GM atrophy in AD patients against healthy controls was found using voxel based morphometry (VBM). [17]
The VOIs were segmented from regions having significant GM volume reduction. A feature vector was extracted from these voxel values and ranked using t-test scores and genetic algorithm, to select an optimal subset of features. The classification was performed using SVM with a 10-fold cross validation and achieved an accuracy of 84.17% and 70.38% for AD vs normal and MCI vs normal class respectively.
In [18], GM volume was detected using VBM for AD patients and healthy controls, and regions with significant GM atrophy change were selected as VOIs. The voxel values from these regions were treated as raw features, which were then evaluated using seven different feature ranking techniques i.e., mutual information (MI), information gain, statistical dependency, Fisher's criterion, t-test score, Pearson's correlation coefficient, and the gini-index. The classification of subjects was performed using SVM with an accuracy of 92.48%. In [19], a Laplace Beltrami eigenvalue shape descriptor was used to classify the AD. The shape changes of corpus callosum were analyzed by segmenting T1-weighted MRI scans using reaction diffusion level set approach. Information gain ranking was used to select the significant features, which were classified using K-nearest neighbour (KNN) and SVM. A maximum classification accuracy of 93.37% was achieved using the KNN classifier. However, quantifying variations in the micro

structure of corpus callosum is difficult, which makes the method less useful in practice.

A framework for feature extraction from low-dimensional subspaces that signify inter-subject variability was proposed in [20]. Data-driven ROIs were used to build the manifold subspace. A sparse regression with MMSE score was used to learn these regions. The sampling bias was reduced along with a re-sampling scheme using sparse regression for performing variable selection. A classification accuracy of 71% was achieved. In [21], a sulcal medial surface for AD and cognitively normal (CN) classification method was proposed. Brain-VISA sulcal identification pipeline was used on subjects to extract 24 distinct sulci for each subject and SVM was used to classify AD and CN along with the computation of sulcal medical surface features. A classification accuracy of 87.9% was achieved. In [22], various measurements like cortical thickness, hippocampus texture and shape, were combined to make a bio-marker that used information from MRI data. The method was trained on MR scans from the Alzheimer's Disease Neuroimaging Initiative (ADNI) database using linear discriminant analysis (LDA). The results showed that the combination of these MRI bio-markers achieves 62.7% multi-class classification accuracy. In [23], ocal features were extracted using the circular harmonic functions (CHFs) from hippocampus and posterior cingulate cortex. The classification accuracy for AD vs MCI task was 62.07%.

A deep learning algorithm was presented to classify AD subjects in [24]. Auto-encoders were used alongside convolutional neural networks to predict the output classes as AD and normal, with a classification accuracy of 98.4%. Other classes and multi-class classification was not considered. Convolution neural network was used to extract discriminative features for classification of AD and normal subjects [25]. Although, deep learning based methods have achieved significant results in big data analysis, but harvesting useful information from large collections of unstructured data require a lot of training, and computational power [26]. The selection of optimal hyper-parameters and best architecture is also a difficult task.

The binary classification of AD and MCI, as well as multi-class classification including AD, normal and MCI is a challenging task. Most methods reported in literature focused solely on features directly extracted from the brain images, which limited their effectiveness in classification. In this study, clinical information was used along with features extracted from whole as well as segmented brain images, which significantly improved the binary and multi-class classification performance. [27]

III. Proposed Methodology

An overview of the methodology used in this project has shown in the Fig. 2. As for providing complete explanations about the methodology, We split this section into five major sub-sections. In the first part, we will talk about the properties of the data set we used. The second part will focus on the preprocessing steps applied to the images before feeding them to the network. In the third part, we will discuss the Keras library used to implement the project. Since we utilized three different networks as feature descriptors (NasNetMobile, VGG16, and Resnet50), In The fourth part of this section, we will provide details of that networks. At last, in the fifth part, we will talk about the details of the MLP classifier at the end of the networks.

A. Dataset

In this research, we utilized Kaggle Alzheimer's Dataset, which is publicly available. This dataset is hand collected from various websites with each and every label verified. The data consists of MRI images in jpg format, and the size of the images is 176 x 208. The pixel value of the Data set images has an integer ranging from 0 to 255. The dataset has four classes of images, both in training as well as a testing set:

1) Mild Demented
2) Moderate Demented
3) Non Demented
4) Very Mild Demented

The total number of Images in the Training set is 5121, and the Total number of images in the test set is 1279. This dataset is not balanced, and the number of images in each class is not equal. A few samples of the Kaggle Dataset are shown in Fig. 3.

B. PreProcessing

There are some pre-processing steps applied to images before feeding them to the network.

1) *Image resizing:* As previously mentioned in the Dataset section, all of the Images have the size of 176 x 208. we performed resizing on all of the Images to the size of 224 x 224.
2) *Image normalization:* We performed image normalization on the images to force the pixel values in the range of zero to one. In this regard, we divided each pixel value by 255. Normalization is recommended pre-processing step by researchers due to its positive effect on increasing the numerical stability of the model.
3) *Feature Engineering:* Since the task of this project is classification, and we are working with categorical and noncontinuous data, we should convert our labels into one-hot encodings. One-hot encodings are a way for the model to understand that we're looking at categorial instead of continuous data. It is worth mentioning that since the number of the classes is four, The class labels have integers ranging from 0 to 3, and One-hot encodings are applied to these class numbers.
4) *Data Augmentation:* Data Augmentation is One of the Important methods used to prevent the overfitting of the network by generating some artificial image in a way that the class information does not change. Data Augmentation increases the size of the training data by applying random shifts, rotations, gray value variations, and random elastic deformations to the training samples. So, Data Augmentation could significantly help the increasing generalization performance of the model.

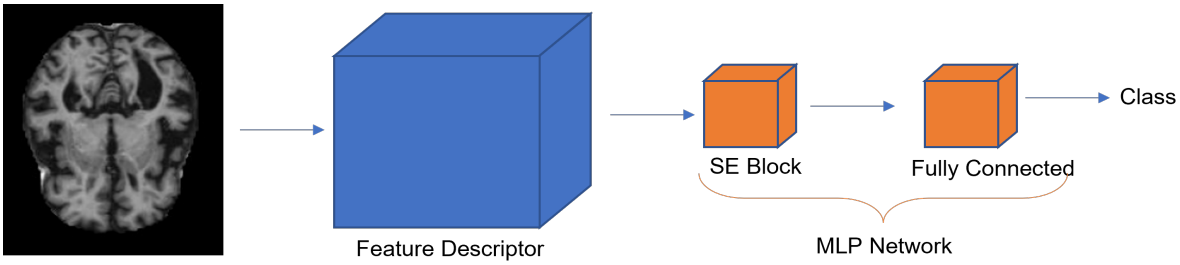

Fig. 2. Methodology

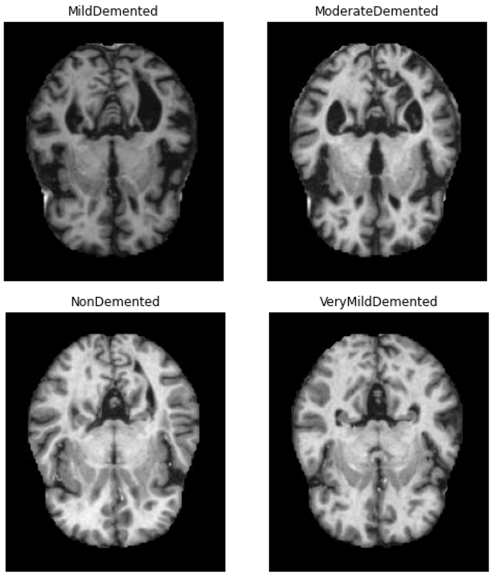

Fig. 3. A few samples of Kaggle's Alzhimer's Dataset

In this research, we randomly applied different data augmentations to the training Images, and their related settings are enlisted in Table. It is worth mentioning that randomly applying augmentation also helps the network's generalization.

TABLE I
REQUIRED SETTING FOR DATA AUGMENTATION

Data augmentation	Value
Horizontal flip	True
Rotation range	5°
Width_shift_range	0.1
height_shift_range	0.1

C. Dependencies

In this project, we used KERAS Library to implement different parts of the methodology, including data processing and handling, training the networks, and evaluation of the methods. Keras is an open-source software library that provides a Python interface for machine learning and artificial neural networks. Keras has built on Tensorflow library acts as an interface for it. KERAS contains numerous implementations of commonly used neural network building blocks such as layers, objectives, activation functions, and optimizers. In addition to standard neural networks, Keras has supported other networks such as convolutional and recurrent neural networks. KERAS has the capability to move the computations to the GPU, which reduces the training time significantly.

D. Feature Descriptor

We used three different pre-trained models as the feature descriptor, including NASNet Mobile, VGG16, and ResNet. The number of parameters of each network are listed in the table below. We will discuss about the details of each of the networks in the following sections.

TABLE II
COMPARISON OF THE NUMBER OF PARAMETERS

Architecture	Parameters
Resnet50	23,587,812
VGG16	14,714,688
NasNet Mobile	4,269,716

1) *Resnet50:* He [28] et al. proposed a novel CNN model called deep residual network for image classification. The main difference between a residual network and a typical CNN is that they have different network architectures. For a typical CNN model, it organizes the architecture by combining basic units such as convolution, nonlinear mapping, pooling or batch normalization in a cascade manner. But for a residual network, it has a shortcut pathway directly connecting the input and the output in a building block. [29]
Deep residual networks are made up of residual units. Each residual unit can be expressed as:

$$y_i = h(x_i) + F(x_i, w_i) \tag{1}$$

$$x_i + 1 = f(y_i) \tag{2}$$

where F is a residual function, f is a $RelU$ function, w_i is the weight matrix, and x_i and y_i are the inputs and outputs of the i-th layer. The function h is an identity mapping given by [28]:

$$h(x_i) = x_i \tag{3}$$

The essential idea behind residual learning is the branching of the paths for gradient propagation. For CNNs, this idea was first introduced in the form of parallel paths in the inception models of Residual networks share a few similarities with the highway networks [30] such as residual blocks and shortcut connections. However, the output of each path in the highway network is controlled by a gating function, which is learned during the training phase.

The residual units in ResNets are not stacked together as is the case with convolutional layers in a conventional CNN. Instead, shortcut connections are introduced from the input of each convolutional layer to its output. The use of identity mappings as shortcut connections decreases the complexity of the residual networks resulting in deep networks that are faster to train. ResNets can be seen as an ensemble of many paths, instead of viewing it as a very deep architecture. However, all of these network paths in the ResNets are not of the same length. Only one path goes through all of the residual units. Moreover, all of these signal paths do not propagate the gradient which accounts for the faster optimization and training of ResNets. [31] ResNet 50 has 50 layers of residual networks.

2) *VGG16:* The VGG network architecture was initially proposed by Simonyan and Zisserman [32]. The VGG models with 16 layers (VGG16) and with 19 layers (VGG19) were the basis of their ImageNet Challenge 2014 submission, where the Visual Geometry Group (VGG) team secured the first and the second places in the localization and classification tracks respectively.The VGG16 architecture, shown at the top of Fig. 4, is structured starting with five blocks of convolutional layers followed by three fully-connected layers. Convolutional layers use 3×3 kernels with a stride of 1 and padding of 1 to ensure that each activation map retains the same spatial dimensions as the previous layer. A rectified linear unit (ReLU) activation is performed right after each convolution and a max pooling operation is used at the end of each block to reduce the spatial dimension. Max pooling layers use 2×2 kernels with a stride of 2 and no padding to ensure that each spatial dimension of the activation map from the previous layer is halved. Two fully-connected layers with 4096 ReLU activated units are then used before the final 1000 fully-connected softmax layer.

A downside of the VGG16 model is that it is expensive to evaluate and use a lot of memory and parameters. VGG16 has approximately 138 million parameters. Most of these parameters (approximately 123 million) are in the fully-connected layers, that are replaced by a SVM classifier in our model, significantly reducing the number of necessary parameters. [33]

3) *NasNet Mobile:* Neural Architecture Search Network (Nasnet) was developed by Google brain team, which uses the two main functionalities are 1) Normal cell 2) Reduction cell which shown in Fig. 5. Initially Nasnet applies its operations on the small dataset and then transfer its block to the large dataset to achieve the higher mAP. A modified droppath called Scheduled droppath for effective regularization is used for improved performance of Nasnet. In the original Nasnet Architecture where the number of cells are not predefined and specifically normal and reduction cells are used and architecture show in Fig. 5. where as normal cells defines the feature map size and reduction cell returns the reduce feature map in terms of height and width by the factor of two. A Control architecture in Nasnet based on Recurrent Neural network (RNN) is used predicts the entire structure for the network based on the two initial hidden states. Controller Architecture uses the RNN based LSTM model and Softmax prediction is used for the Convolutional cells prediction and constructed the network motifs recursively. Here in our model NASNetMobile which uses 224×224 as input image size and NASnettLarge model uses 331×331 as the input image size. NASNetMobile uses the Imagenet pre-trained network weights for transfer learning process to detect the face masks. [34]

E. Multilayer Perceptron

In this section, We will talk about the classifier of the network and the different blocks used in the classifier. A squeeze and excitation (SE) [35] block, a GlobalAveragePooling, and multiple fully connected layers are employed in the classifier. The first block in MLP design is the SE block which is used as the attention mechanism. Since the last convolutional layer in a CNN has highly class-specific information, we employed the SE block as a content-aware mechanism to weight each channel adaptively. SE block introduces a block for CNN that improves channel affinities at a less computational cost. The general idea in this block is to add parameters to each convolution block so that the network is capable of adjusting the weights for each feature map automatically. In short, it can be said that this block improves the interdependency of the channels by weighting the channels. The final layer of the classifier represents the number of classes in the data set. To avoid overfitting, a dropout layer is also employed after the first fully connected layer [36].

IV. Evaluation and Results

A. Used System

All the project steps, including Data preparation, training of the models, and evaluation of the methods, were done using Google Colab. The training was performed on GPU, and we did not freeze the pre-trained models, and all of the network

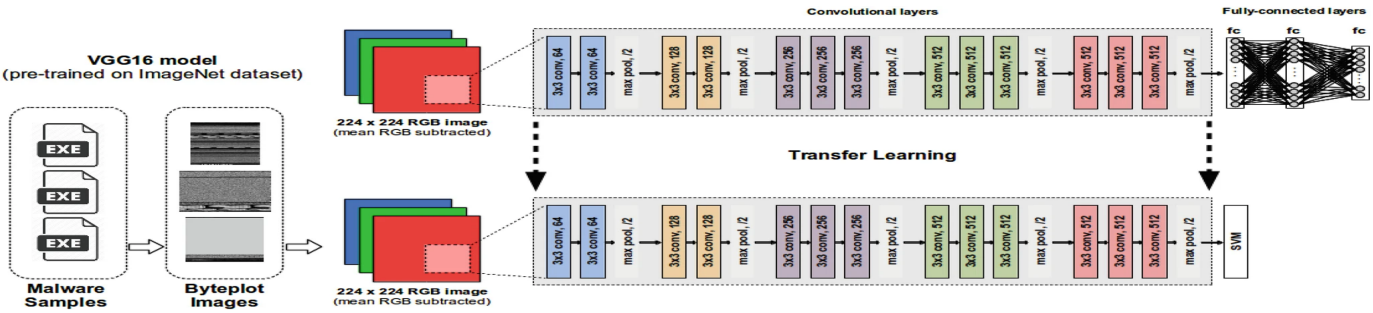

Fig. 4. VGG16 Architecture
Image taken from [33]

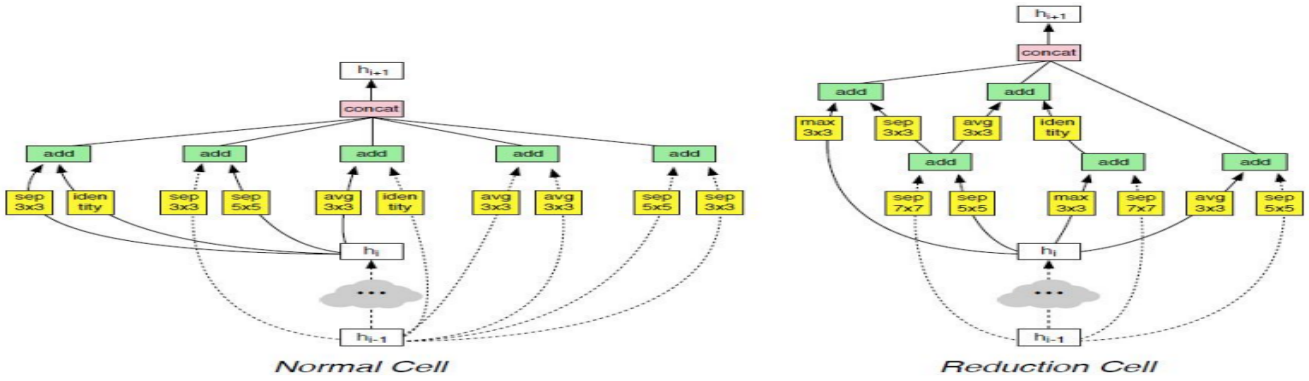

Fig. 5. Nasnet normal and Reduction Cell Architecture. Image taken from [34]

weights were updated during training. The training time was approximately 10 hours.

B. Hyperparameters

Training a Deep neural network has different hyperparameters, and all the details them is provided in the following:

- *Data set split rate:* We splitted Kaggle Alzheimer's training data set into two parts. The first part, consisting of 80% of the total images, is used for training, while the second part, consisting of the remaining 20%, is used for validation.
- *Batch size:* According to the definition of batch size, it is the number of training samples fed to the network during one iteration. In this project, the batch size is set to 64.
- *Optimizer:* Optimizer updates the weights of the network using the loss function. There are different optimizers in the literature, and each of them has its special properties. However, due to the excellent performance and popularity of the Adam optimizer, we used it in this project. In this project, the batch size is set to 0.000001.
- *Loss function:* A loss function is a function that compares the predictions and the labels and computes a loss value which is used to update the weights. The loss function has a major effect on training the network, and it should be selected or designed properly according to the task of the project and the distribution of the dataset. There are some popular loss functions in the literature, and Categorical Cross entropy is generally used for multi-class classification tasks. However, although our project is a multi-class classification task, since Kaggle Alzheimer's dataset is unbalanced, we could not use common categorical cross-entropy. So we used weighted categorical cross-entropy in which the weights are inversely proportional to the number of images in each class. In this way, we force the network to try to predict all the classes properly.
- *The number of epochs:* According to the definition of the epoch, it is equal to the number of times that the model sees the whole training dataset. According to the complexity of the Alzheimer's Disease classification, a medical task, and the unbalancing of the dataset, 200 epochs are needed for the model to fully converge. As mentioned in previous sections, the training time is approximately 5 hours.
- *Data Augmentation:* this is one of the pre-processing steps applied to the images and was fully explained in the preprocessing section. All the hyperparameters related to Data augmentation are provided in table 1. It is worth mentioning that Since the used networks are deep, there is a high probability that the networks will overfit. So, we applied data shuffling in the training process in order

to prevent overfitting.

C. Evaluation Criteria

Using evaluation metrics, we can monitor the correctness of the methods and check the model's performance. Various metrics are introduced for different tasks in the literature. The most popular metric used for classification tasks is accuracy. However, since we are classifying an unbalanced dataset, we used other metrics, including F1-score, Accuracy, Precision, Recall, and AUC, to perform a comprehensive study for classifying Alzheimer's Disease task.

- *Accuracy:* It is defined as the ratio of the number of correct predictions of the network across all the classes to the total number of input samples.
  As mentioned in the previous sections, accuracy works well only if an equal number of samples belong to each class (balanced data set).

$$Accuracy = \frac{TP + TF}{TP + TF + FP + FN} \tag{4}$$

- *Precision:* It is defined as the ratio of the number of correct positive predictions of the network to the total number of positive predictions.

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

- *Recall:* It is defined as the ratio of the number of correct positive predictions of the network to the total number of all observations in the actual class. It is worth mentioning that there is no relation between precision and recall, and there is no guarantee that if a classifier has a high precision rate, it also has a high recall rate.

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

- *F1-Score:* It is a metric that combines Precision and Recall metrics into a single metric. According to the F1-Score formula, it is the weighted average of the precision and recall.

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{7}$$

- *AUC:* AUC is a metric that measures the performance of a classifier at various threshold settings. AUC is the area under the ROC curve that tells us how much the model is capable of distinguishing between classes. Higher the AUC, the model, is more confident about its predictions. The ROC curve is plotted with True Positive Rate (TPR) against the False Positive Rate (FPR), where TPR is on the y-axis and FPR is on the x-axis.
  It is worth mentioning that Although we report the accuracy rate in the results section, since our dataset is not balanced, we cannot use accuracy as our metric. For this reason, we also used ROC AUC as the metric. An example of ROC curve is shown in Fig. 6

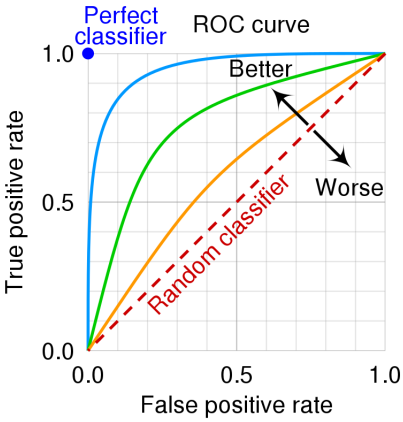

Fig. 6. ROC Curve.Image taken from [37]

D. Confusion Matrix

It is a way to show a summary of the classifier's performance. The confusion matrix shows us the number of correct and incorrect predictions in each class. In this way, we can inform which class is more challenging for our classifier to predict correctly.

E. Feature maps

The feature map of a CNN network refers to the result of applying the filters to the input image. In other words, in each layer, we have a feature map which is the output of that layer. By plotting the feature maps of different layers, we could see the features each layer extracted by applying its special filter. In order to access the feature maps of specific layers of the model, we should build the model in a way that has an output in that layers.

F. Attention maps

By plotting the attention maps, we can see that each of the classes pays attention to which parts of the input image. In order to give more explanations about the attention map, as an example, consider an animal image classifier. If we apply Fig. 7 as the input of the classifier and plot the attention map of the dog class output, we probably see something similar to Fig. 8

Fig. 7. Input image for computing the attention map.Image taken from [38]

In this project We plotted the attention maps by computing the gradient of the activated class with respect to the last convolutional layer of the feature descriptor. We plot the mentioned attention maps by adopting the Grad Cam method of the Keras Library.

Fig. 8. Representing the attention map.Image taken from [38]

G. Results

This section will report the evaluations performed on the proposed method. As mentioned earlier, we utilized three different pre-trained models as the feature descriptor. We did not freeze these networks in the training process, and all the weights were updated during the training. The results are presented in III. According to the results proposed in III, among three feature descriptors, using the Resnet50 leads to achieving the best results.

Since the best results were obtained by the Nasnet-Mobile network, in the following, we will only report the results of this network. 9 shows the graph of the accuracy and loss of the network during training. It is obvious that the network has been fully converged after 500 epochs.

Fig. 10 shows the confusion matrix of the network. According to the confusion matrix, It seems that approximately all the classes are challenging for the network. One of the Important reasons for this is that the classes of the Kaggle Azhimers dataset are unbalanced.

The percentage of the images belonging to different classes in the dataset is 14%, 1%, 50%, and 35%, respectively.

Fig. 11 shows an example image that we want to feed to the network to plot the previously mentioned feature maps and attention maps. Fig. 12 shows the feature maps of different layers of the network. We can see that different layers extract different features from the input image. Fig. 13 shows the attention map of the activated class. In this picture, we can see that the network pays attention to which parts of the input brain image to determine the class of the image.

TABLE III
THE RESULTS OF DIFFERENT MODELS

Network	Class	Precision	Recall	F1-Score	Accuracy	AUC
VGG16	0	0.37	0.48	0.42	0.56	0.84
VGG16	1	0.71	1	0.83	0.56	0.84
VGG16	2	0.73	0.61	0.66	0.56	0.84
VGG16	3	0.47	0.52	0.49	0.56	0.84
NasNet	0	0.35	0.38	0.36	0.52	0.67
NasNet	1	0.63	1	0.73	0.52	0.67
NasNet	2	0.70	0.65	0.67	0.52	0.67
NasNet	3	0.49	0.45	0.48	0.52	0.67
ResNet	0	0.71	0.71	0.71	0.78	0.94
ResNet	1	0.77	1	0.87	0.78	0.94
ResNet	2	0.87	0.78	0.82	0.78	0.94
ResNet	3	0.71	0.81	0.76	0.78	0.94

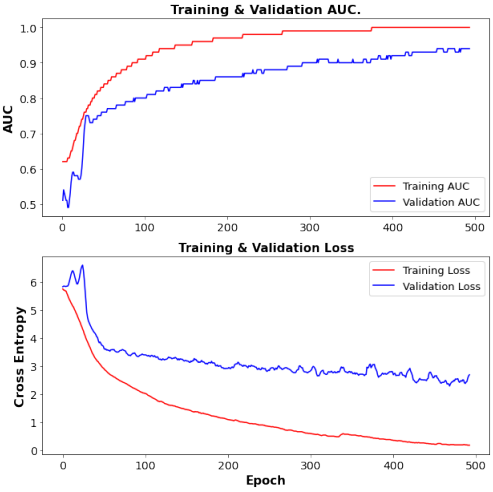

Fig. 9. AUC and Loss Graph

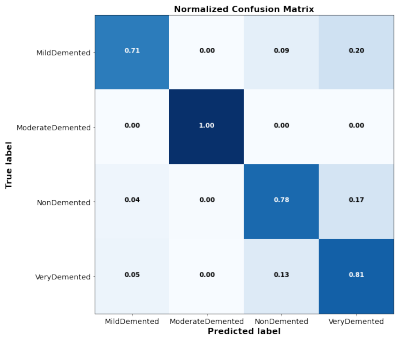

Fig. 10. Confusion Matrix

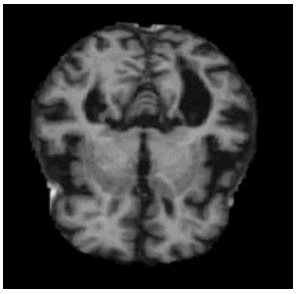

Fig. 11. Sample Image

V. CONCLUSION

In this research, we performed a comprehensive study on classifying an unbalanced dataset of Alzheimer's Disease (The percentage of the images belonging to different classes in the dataset is 14%, 1%, 50%, and 35%, respectively). We utilized three different pre-train networks (Resnet50, VGG16, and NasNet-Mobile) to compare their performance in AD classification. We also employed the SE block as an attention mechanism in the designed MLP network, which is used at the

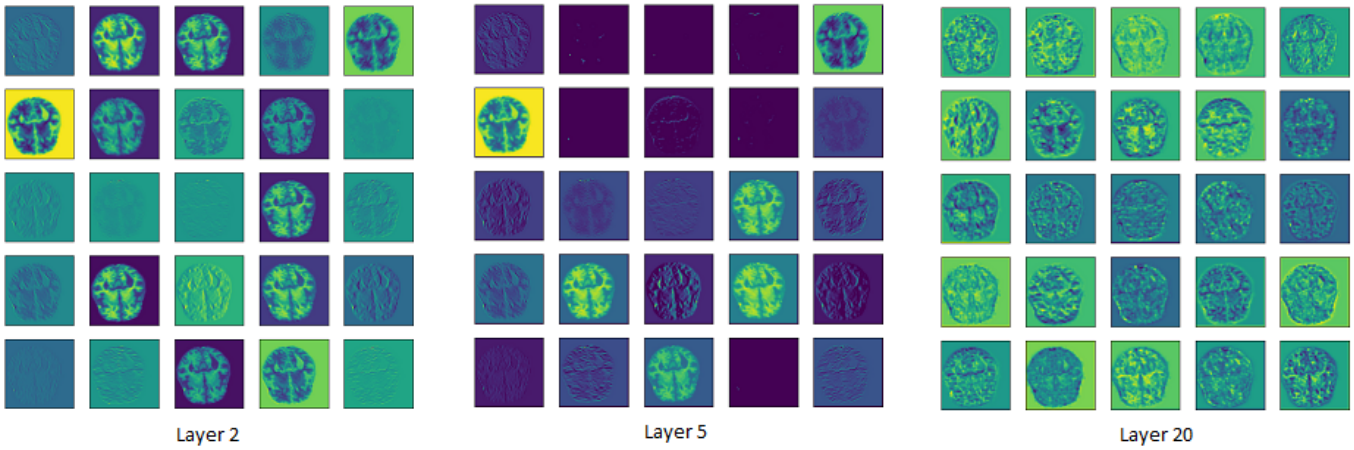

Fig. 12. Different Layer's Feature Map

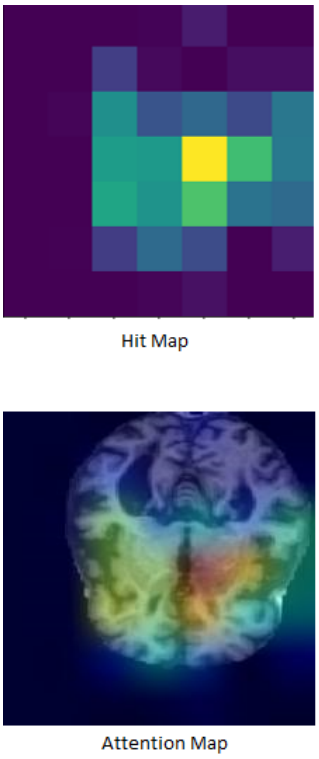

Fig. 13. Attention Map

end of the pre-trained models. Because Kaggle Alzheimer's Dataset is unbalanced, we used weighted categorical cross-entropy as the loss function to help the network classify different classes properly. Among all the pre-trained models, the Resnet50 achieved the best results (AUC: 0.94). We also performed different studies on the Resnet50 network, including computing the confusion matrix, plotting the feature maps of several layers, and plotting the attention map of the activated class.

REFERENCES

[1] V. L. Villemagne, S. Burnham, P. Bourgeat, B. Brown, K. A. Ellis, O. Salvado, C. Szoeke, S. L. Macaulay, R. Martins, P. Maruff, *et al.*, "Amyloid β deposition, neurodegeneration, and cognitive decline in sporadic alzheimer's disease: a prospective cohort study," *The Lancet Neurology*, vol. 12, no. 4, pp. 357–367, 2013. 1

[2] E. M. Reiman, Y. T. Quiroz, A. S. Fleisher, K. Chen, C. Velez-Pardo, M. Jimenez-Del-Rio, A. M. Fagan, A. R. Shah, S. Alvarez, A. Arbelaez, *et al.*, "Brain imaging and fluid biomarker analysis in young adults at genetic risk for autosomal dominant alzheimer's disease in the presenilin 1 e280a kindred: a case-control study," *The Lancet Neurology*, vol. 11, no. 12, pp. 1048–1056, 2012. 1

[3] A. Association, "2019 alzheimer's disease facts and figures," *Alzheimer's & dementia*, vol. 15, no. 3, pp. 321–387, 2019. 1

[4] N. Yamanakkanavar, J. Y. Choi, and B. Lee, "Mri segmentation and classification of human brain using deep learning for diagnosis of alzheimer's disease: a survey," *Sensors*, vol. 20, no. 11, p. 3243, 2020. 1

[5] C.-J. Hsiao, E. Hing, and J. Ashman, *Trends in Electronic Health Record System Use Among Office-based Physicians, United States, 2007-2012*. No. 75, US Department of Health and Human Services, Centers for Disease Control and …, 2014. 1

[6] I. Despotović, B. Goossens, and W. Philips, "Mri segmentation of the human brain: challenges, methods, and applications," *Computational and mathematical methods in medicine*, vol. 2015, 2015. 1

[7] G. P. Zhang, "Neural networks for classification: a survey," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 30, no. 4, pp. 451–462, 2000. 1

[8] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of control, signals and systems*, vol. 2, no. 4, pp. 303–314, 1989. 1

[9] M. D. Richard and R. P. Lippmann, "Neural network classifiers estimate bayesian a posteriori probabilities," *Neural computation*, vol. 3, no. 4, pp. 461–483, 1991. 2

[10] B. Widrow, D. E. Rumelhart, and M. A. Lehr, "Neural networks: applications in industry, business and science," *Communications of the ACM*, vol. 37, no. 3, pp. 93–106, 1994. 2

[11] A. Quattoni, M. Collins, and T. Darrell, "Transfer learning for image classification with sparse prototype representations," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, IEEE, 2008. 2

[12] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 8697–8710, 2018. 2

[13] D. Theckedath and R. Sedamkar, "Detecting affect states using vgg16, resnet50 and se-resnet50 networks," *SN Computer Science*, vol. 1, no. 2, pp. 1–7, 2020. 2

[14] A. S. B. Reddy and D. S. Juliet, "Transfer learning with resnet-50 for malaria cell-image classification," in *2019 International Conference on*

Communication and Signal Processing (ICCSP), pp. 0945–0949, IEEE, 2019. 2

[15] I. Beheshti, N. Maikusa, H. Matsuda, H. Demirel, G. Anbarjafari, *et al.*, "Histogram-based feature extraction from individual gray matter similarity-matrix for alzheimer's disease classification," *Journal of Alzheimer's disease*, vol. 55, no. 4, pp. 1571–1582, 2017. 2

[16] S. Wang, Y. Zhang, G. Liu, P. Phillips, and T.-F. Yuan, "Detection of alzheimer's disease by three-dimensional displacement field estimation in structural magnetic resonance imaging," *Journal of Alzheimer's Disease*, vol. 50, no. 1, pp. 233–248, 2016. 2

[17] I. Beheshti, H. Demirel, H. Matsuda, A. D. N. Initiative, *et al.*, "Classification of alzheimer's disease and prediction of mild cognitive impairment-to-alzheimer's conversion from structural magnetic resource imaging using feature ranking and a genetic algorithm," *Computers in biology and medicine*, vol. 83, pp. 109–119, 2017. 2

[18] I. Beheshti, H. Demirel, F. Farokhian, C. Yang, H. Matsuda, A. D. N. Initiative, *et al.*, "Structural mri-based detection of alzheimer's disease using feature ranking and classification error," *Computer methods and programs in biomedicine*, vol. 137, pp. 177–193, 2016. 2

[19] A. K. Ramaniharan, S. C. Manoharan, and R. Swaminathan, "Laplace beltrami eigen value based classification of normal and alzheimer mr images using parametric and non-parametric classifiers," *Expert Systems with Applications*, vol. 59, pp. 208–216, 2016. 2

[20] R. Guerrero, R. Wolz, A. Rao, D. Rueckert, A. D. N. I. (ADNI, *et al.*, "Manifold population modeling as a neuro-imaging biomarker: application to adni and adni-go," *NeuroImage*, vol. 94, pp. 275–286, 2014. 3

[21] M. Plocharski, L. R. Østergaard, A. D. N. Initiative, *et al.*, "Extraction of sulcal medial surface and classification of alzheimer's disease using sulcal features," *Computer methods and programs in biomedicine*, vol. 133, pp. 35–44, 2016. 3

[22] L. Sørensen, C. Igel, A. Pai, I. Balas, C. Anker, M. Lillholm, M. Nielsen, A. D. N. Initiative, *et al.*, "Differential diagnosis of mild cognitive impairment and alzheimer's disease using structural mri cortical thickness, hippocampal shape, hippocampal texture, and volumetry," *NeuroImage: Clinical*, vol. 13, pp. 470–482, 2017. 3

[23] O. B. Ahmed, M. Mizotin, J. Benois-Pineau, M. Allard, G. Catheline, C. B. Amar, A. D. N. Initiative, *et al.*, "Alzheimer's disease diagnosis on structural mr images using circular harmonic functions descriptors on hippocampus and posterior cingulate cortex," *Computerized Medical Imaging and Graphics*, vol. 44, pp. 13–25, 2015. 3

[24] S. Basaia, F. Agosta, L. Wagner, E. Canu, G. Magnani, R. Santangelo, M. Filippi, A. D. N. Initiative, *et al.*, "Automated classification of alzheimer's disease and mild cognitive impairment using a single mri and deep neural networks," *NeuroImage: Clinical*, vol. 21, p. 101645, 2019. 3

[25] A. Payan and G. Montana, "Predicting alzheimer's disease: a neuroimaging study with 3d convolutional neural networks," *arXiv preprint arXiv:1502.02506*, 2015. 3

[26] X.-W. Chen and X. Lin, "Big data deep learning: challenges and perspectives," *IEEE access*, vol. 2, pp. 514–525, 2014. 3

[27] T. Altaf, S. M. Anwar, N. Gul, M. N. Majeed, and M. Majid, "Multi-class alzheimer's disease classification using image and clinical features," *Biomedical Signal Processing and Control*, vol. 43, pp. 64–74, 2018. 3

[28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016. 4, 5

[29] S. Wu, S. Zhong, and Y. Liu, "Deep residual learning for image steganalysis," *Multimedia tools and applications*, vol. 77, no. 9, pp. 10437–10453, 2018. 4

[30] R. K. Srivastava, K. Greff, and J. Schmidhuber, "Highway networks," *arXiv preprint arXiv:1505.00387*, 2015. 5

[31] A. Mahmood, M. Bennamoun, S. An, and F. Sohel, "Resfeats: Residual network based features for image classification," in *2017 IEEE international conference on image processing (ICIP)*, pp. 1597–1601, IEEE, 2017. 5

[32] Y. Bengio, *Learning deep architectures for AI*. Now Publishers Inc, 2009. 5

[33] E. Rezende, G. Ruppert, T. Carvalho, A. Theophilo, F. Ramos, and P. d. Geus, "Malicious software classification using vgg16 deep neural network's bottleneck features," in *Information Technology-New Generations*, pp. 51–59, Springer, 2018. 5, 6

[34] S. K. Addagarla, G. K. Chakravarthi, and P. Anitha, "Real time multi-scale facial mask detection and classification using deep transfer learning techniques," *International Journal*, vol. 9, no. 4, pp. 4402–4408, 2020. 5, 6

[35] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 7132–7141, 2018. 5

[36] A. Bahri, S. G. Majelan, S. Mohammadi, M. Noori, and K. Mohammadi, "Remote sensing image classification via improved cross-entropy loss and transfer learning strategy based on deep convolutional neural networks," *IEEE Geoscience and Remote Sensing Letters*, vol. 17, no. 6, pp. 1087–1091, 2019. 5

[37] "https://en.wikipedia.org/wiki/receiver_operating_characteristic," 7

[38] "https://keras.io/examples/vision/grad_cam," 7, 8