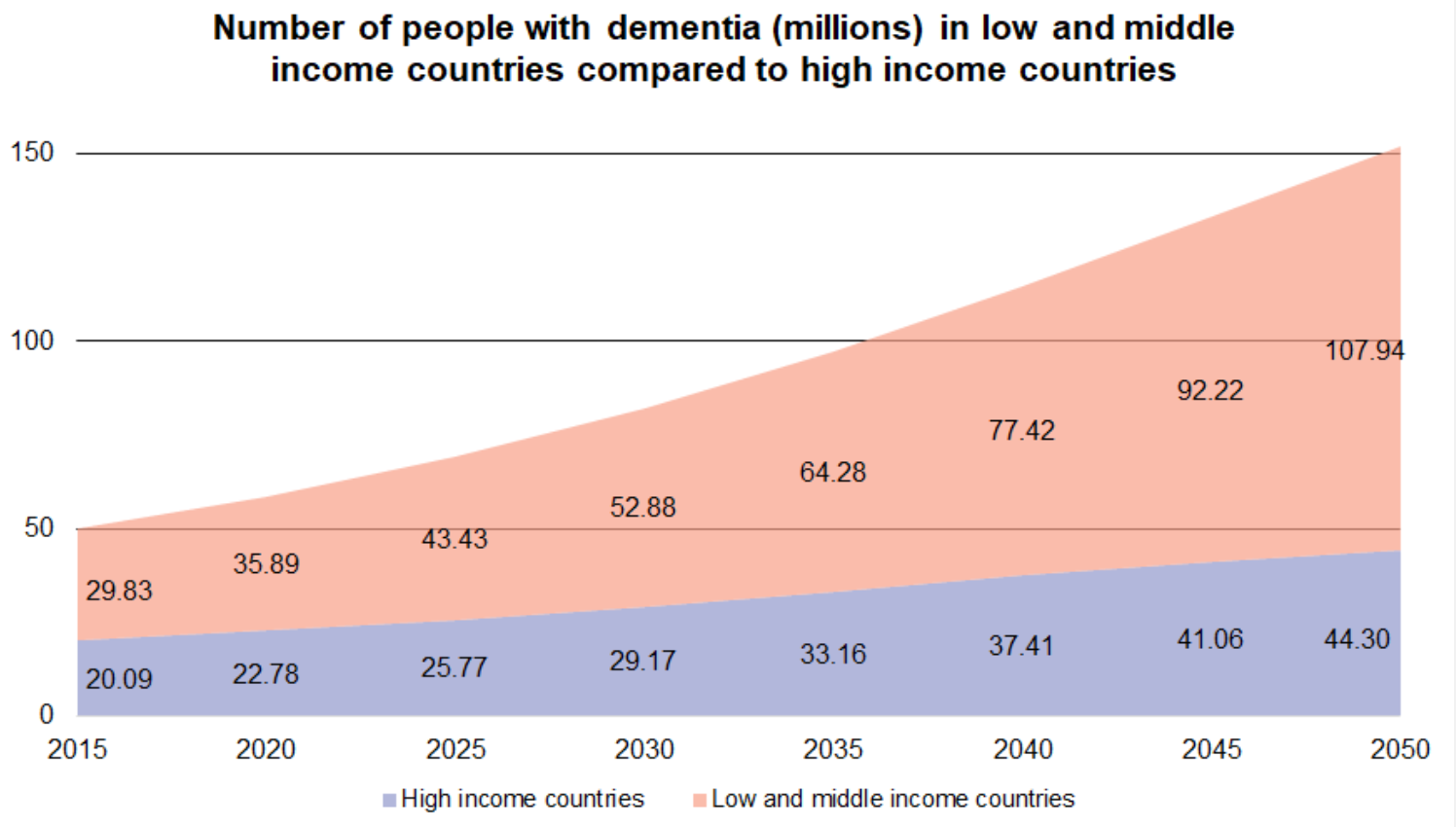
**CNN for Early Alzheimer Disease Detection**

# Research Proposal

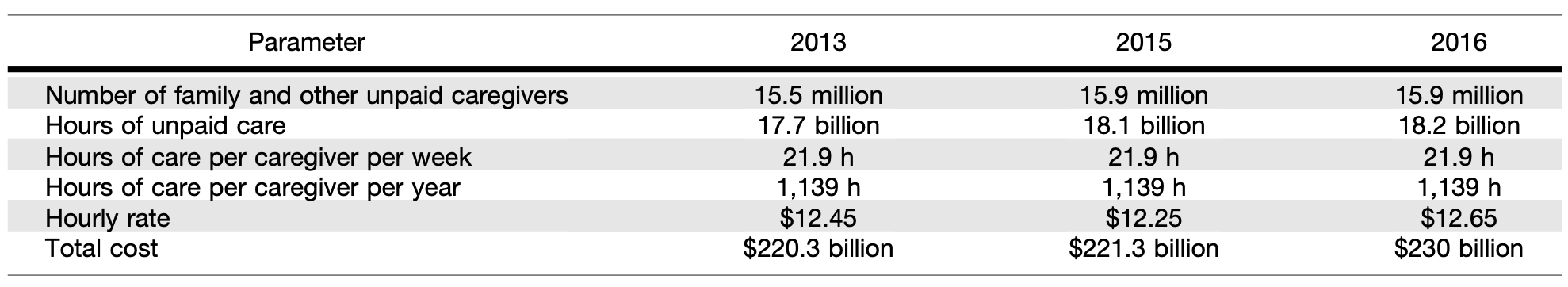
Alzheimer Disease (AD) is a progressive neurodegenerative disease that may lead to memory loss as a result of damaging the brain cells, which may also progress to a characteristic dementia syndrome. As there is currently no cure for AD, early detection of this disease is crucial to aid the potential patient to find a proper treatment to slow down the degenerative progress, reduce the symptoms in the short term and therefore alleviate the burden on the family and caregiver. Therefore, the aim of this project is to develop a model that uses Convolutional Neural Network (CNN) to detect images and predict if the patient is diagnosed with AD.

According to Guerchet, M. et. al. (2020) in their study, as of 2015, approximately 50 millions people around the world are diagnosed with AD and this figure is expected to keep rising in the future.



**Figure 1.** Population diagnosed with AD (Guerchet, M. et. al., 2020)

Furthermore, AD does not only affect the patient but also their family members as it will take a toll on their physical and mental health. The effect is not only restricted to these as it will also be financially challenging for the family members to support the patient. Grabher, B. J. (2018) in her study states that in the US, caregivers provide AD patients approximately 18.4 billion hours of unpaid care and their family members have to spend more than $10,000 a year to support the treatment. Overall, AD cost the US nearly US$277 billion. This means that they will have less disposable income left to be spent on the daily necessities and opportunity costs might arise from the increase in government’s spending on supporting the patients should they rely on the government programs.



**Figure 2**. Cost of treating AD (Grabher, B. J., 2018)

In terms of mental health, the study also shows that taking care of AD patients induce some form of stress, such as work-related, time-related, physical and emotional stress. This is because some of the caregivers will have to reschedule their working hours to take care of the patient, while some others are juggling their work and caregiving role while at the same time facing financial issues to provide for adequate service.

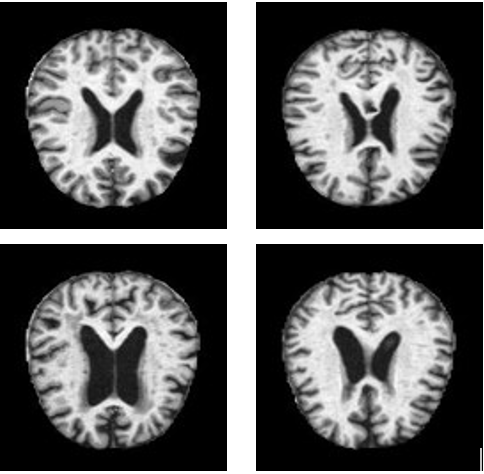
Therefore, by being able to detect the sign of AD early, patients and caregivers will then be able to prepare for their next decision to treat the disease before it progresses beyond the stage where treatments are effective. Barber R. C. (2010) states that some benefits of detecting AD early include a lower treatment bill, ability to predict the rate at which AD progresses, possibility of being enrolled in clinical trials and impose less stress on the caregivers and family members as the patient will be able to attain the required attention in the shortest possible time.

Furthermore, early detection of AD will also alleviate some financial burden on the national budget in countries where the country provides medical benefits such as the US. Study conducted by Leifer, B. P. (2003) shows that the yearly cost savings associated with AD treatment such as prescription drugs and consultation fee amount to $3,891 for each patient. Furthermore, this reduces the need for a caregiver as the symptoms of AD could be mitigated early and put less stress on the family member.

Hence, with this in mind, several studies have tried to use imaging techniques to detect the possibility of AD occurring. Salehi A. W., et. al. (2020) uses CNN for early diagnosis with the sample Magnetic Resonance Imaging (MRI) images. This is because they believe that the use of Deep Learning has mostly been implemented for “automatic diagnosis” of AD to “satisfy clinicians primary goals.” Their result also shows a high accuracy ranging between 89 to 99 percent. Therefore, in our study, CNN will be implemented to validate if it is indeed reliable enough to predict AD.

# Data

The dataset used is Alzheimer MRI Preprocessed Dataset (<https://www.kaggle.com/datasets/sachinkumar413/alzheimer-mri-dataset>) which is adapted from Kaggle. It consists of 6400 images, in which all of them belong to one of the 4 classes - namely Non- Demented, Very Mild Demented, Mild Demented and Moderate Demented. Each of the original images are resized to 128 by 128 pixels. Below are the sample images from each class.



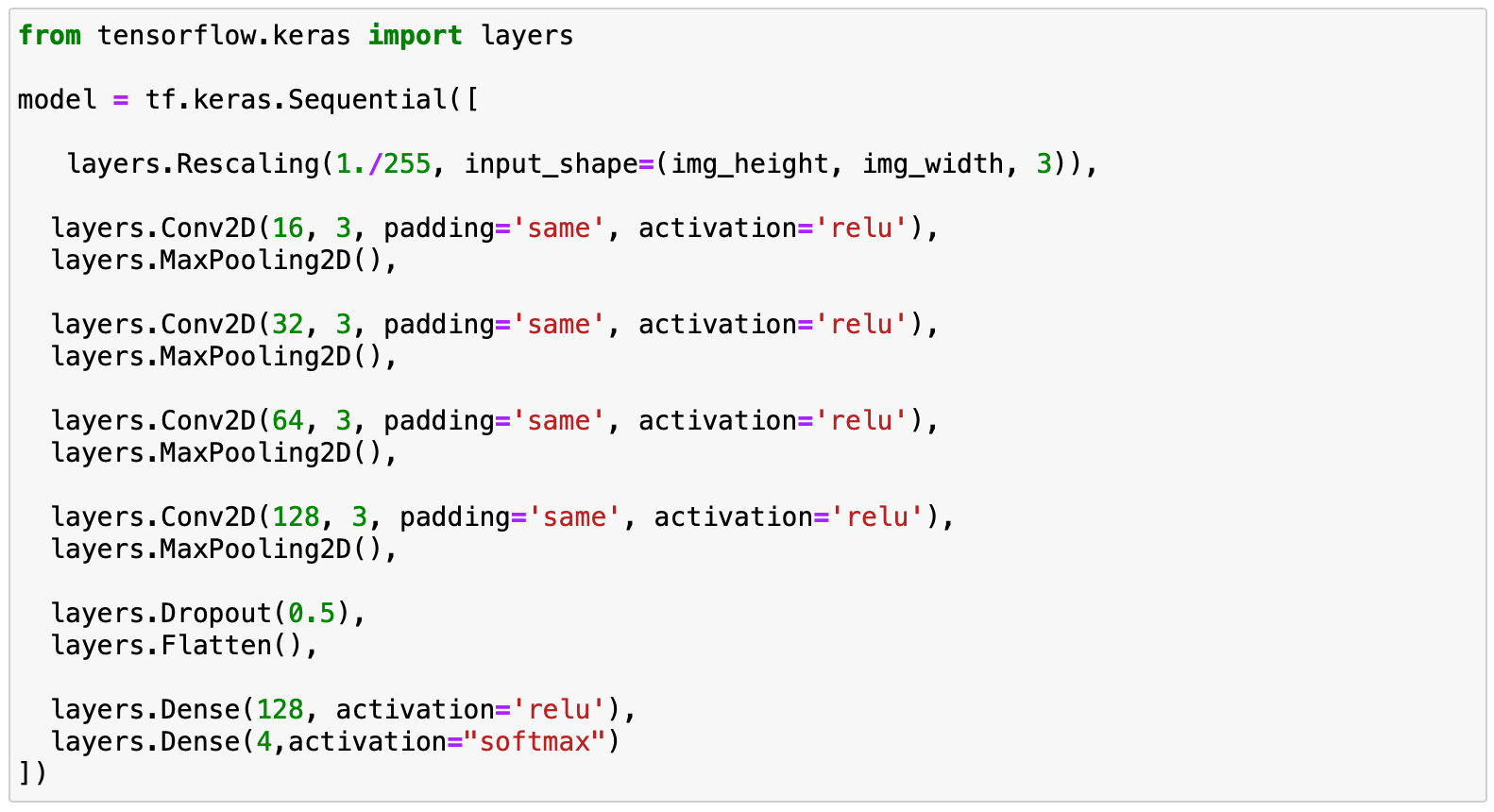
**Figure 3.** Brain MRI for Non-Demented (Top Left), Very Mild Demented (Top Right), Mild Demented (Bottom Left) and Moderate Demented (Bottom Right)

The dataset is resized to 256 by 256 pixels to increase the trainable parameters on the model. However, one drawback is that it will increase the computation time. Since the dataset is considered medium size, hence will be able to produce the model result in reasonable time. The dataset will then be split to training set and testing set (train\_data and val\_data) with the ratio of 80 to 20. There is also no missing data since the dataset are images, hence no data cleaning method is required.

Furthermore, another form of data pre-processing is also applied on the transfer learning stage as seen in Section 6 as the build-in model are not able to run using the current method of data splitting technique. However, the same dataset and splitting ratio are still used, hence there should only be marginal difference, which does not affect the performance of the model significantly.

# Modelling

CNN structure is adopted in the model, in which the algorithm will take in the images as inputs and assign weights to the objects and hence be able to differentiate one image from the other. Therefore, with sufficient training, it will be able to learn the characteristics. In our model, we adopted Sequential class, as it groups several layers inside into the model and then provides training features on the model. The layers consist of 2D Convolution layer (Conv2D), Max Pooling for 2D spatial data layer (MaxPooling2D), Dropout layer (Dropout), Flatten layer (Flatten) and Dense layer (Dense). Conv2D is used to create a convolution kernel that is convolved with the layer input to produce a tensor of outputs. The filter will slide over the image input and perform an element-wise multiplication. It will then sum all the results into a single output pixel. Furthermore, MaxPooling2D layer is used to down-sample the images, reducing its dimensionality. As a result, it will help reduce the possibility of overfitting and decrease the computational cost by reducing the number of trainable parameters. Dropout layer is also used to prevent overfitting by forcing the layers to take more or less responsibility by taking a probabilistic approach instead of learning the statistical noise. Flatten layer is used to convert the result of 2D arrays from MaxPooling2D layers into a long continuous linear vector which will then be passed into the fully connected layer and with the use of Dense layer, it will be added to the neural network.



**Figure 4**. Proposed Model Structure

In the figure above, we can see all the layers that are added into the sequential model. Firstly, we will need to re-scale the image pixels by converting the pixels ranging from 0 to 255 into 0 to 1 by dividing it with 1/255, also known as normalisation. This will allow the image to contribute more evenly to the total loss. In the first layer (Rescaling), it will also take in the input shape, which has been pre-defined as 256 by 256 by 3. Then, the Conv2D and MaxPooling2D layers will be applied. As seen from the figure above, the filter size increases as the layers go deeper. This is because at each layer, there are several patterns to be captured such as edges, corners, dots and many more. The deeper the layer goes, it will combine all the patterns and since it becomes more and more complex, there will be larger combinations of patterns to be captured. Hence, the filter size should increase. The padding “same” is applied such that the input image gets fully covered by the filter and the activation “relu” is used as it will convert negative input value to 0, disabling the activation of all neurons at the same time and therefore preventing exponential growth in the computation to operate the neural network. In the last Dense layer, the number 4 indicated the number of classes, and softmax is used for multi-class classification problems.

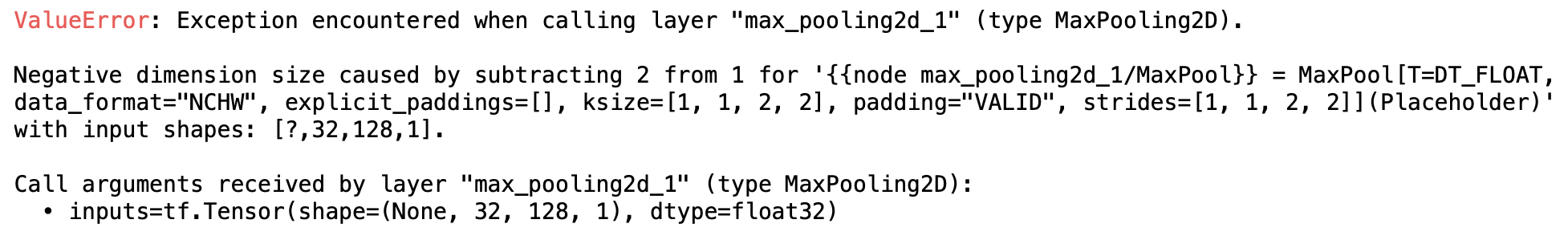


**Figure 5.** Loss and metrics for proposed model

The model will then be compiled by using Adam optimizer since it requires less parameters to be fine tuned and saves the computation time. Since our model consists of 4 classes, we will use SparseCategoricalCrossentropy to compute the loss and the metrics that we will use for analysis is the accuracy. The number of iterations, or epochs, will be set at 10 initially to test for the loss and accuracy of both training and testing data.

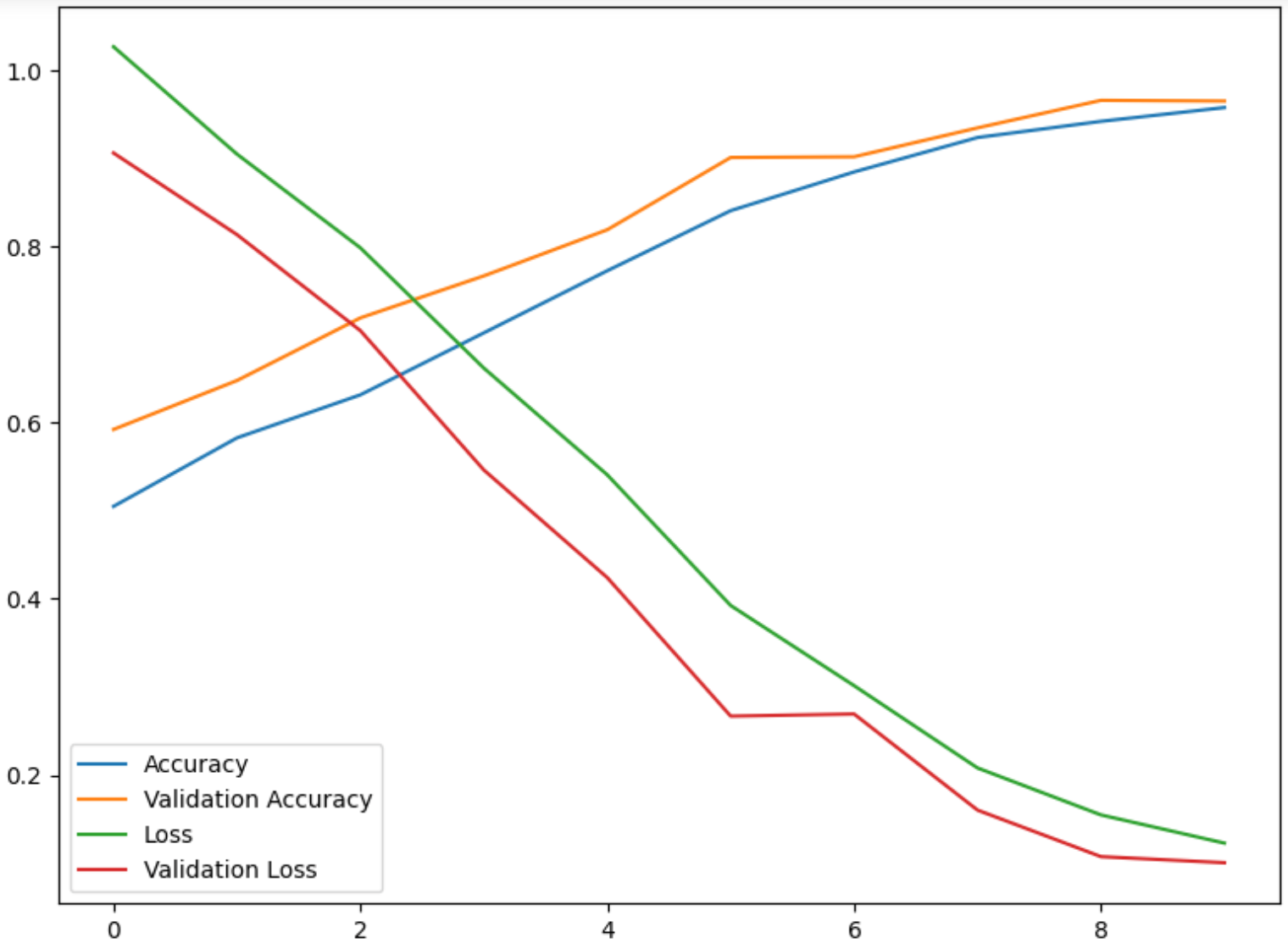
# Model Evaluation and Deployment using AWS-SageMaker

AWS-SageMaker is more beneficial than running the model on a local machine as it contains supplementary software that aids machine learning and decreases the training time significantly. The AWS-SageMaker used in this model has the version of 2.127.0.



**Figure 6.** Error Code on AWS SageMaker

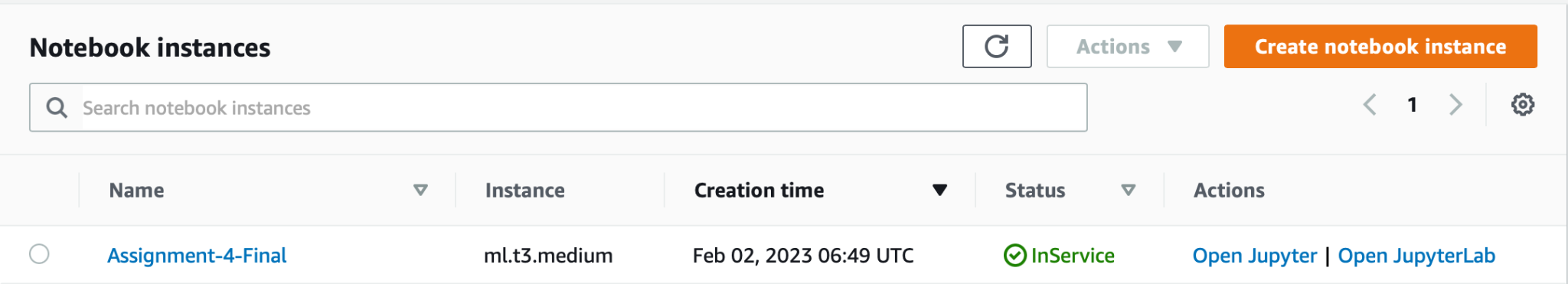
However, by running the model on AWS Sage-Maker, it shows that there has been an error while building the model in the MaxPooling2D layer due to the negative dimension size. However, by running the same model on a local machine, no error is found and the model is able to compute the loss and accuracy of the training and testing data. Several data preprocessing have also been implemented, such as by changing the splitting technique and ratio but none of them aid the model to run on SageMaker. Therefore, the model will be run on a local machine for this project. The figure below displays the loss and accuracy of the model.



**Figure 7.** Model's performance

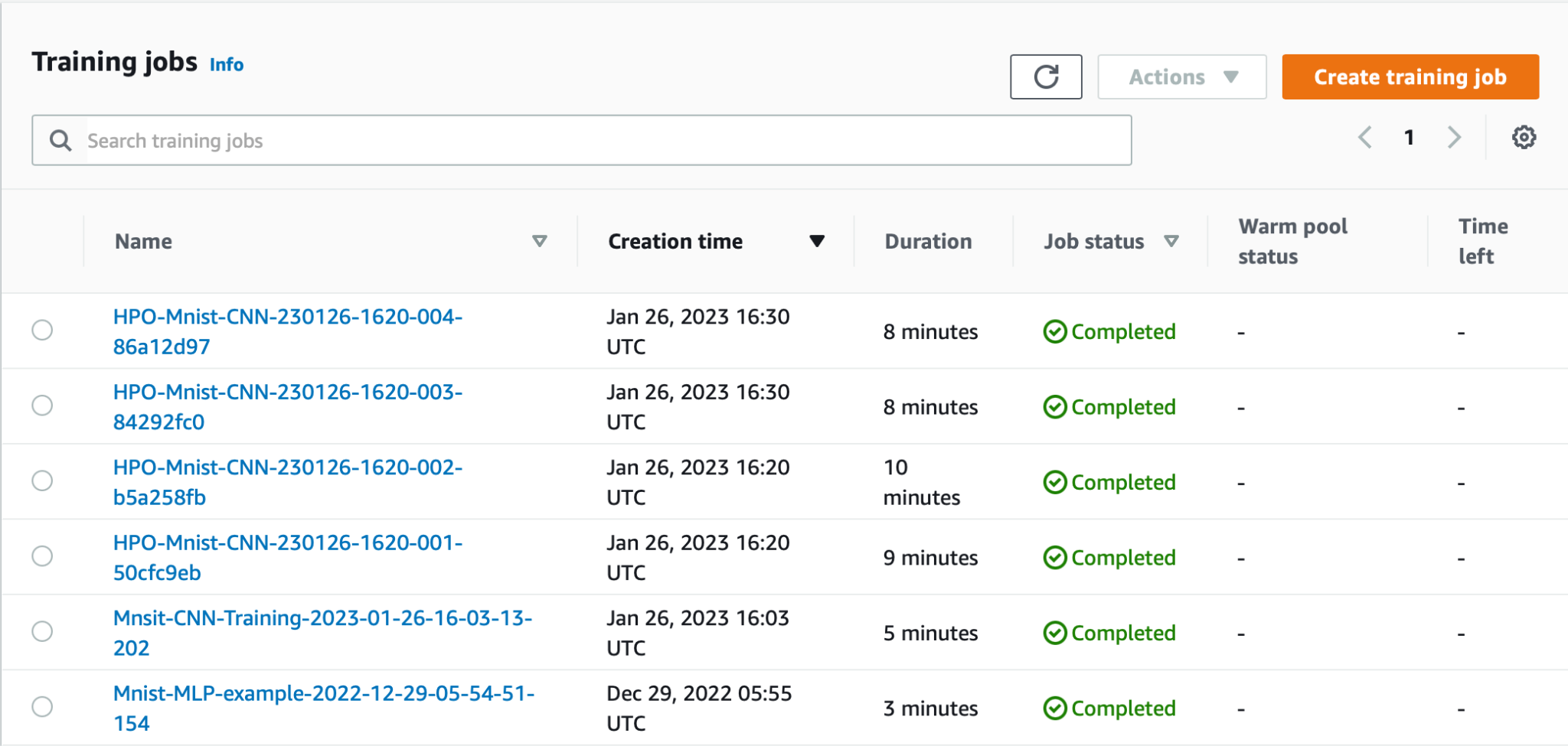
The training data performs relatively well, with an accuracy of approximately 95.7% and loss of 0.123 at 10 iterations. The loss value indicates the model’s prediction on a single example. Therefore, the lower the value, the better the model performs, as also seen from the relatively high accuracy. The model can be improved further by running more iterations, but since each iteration takes approximately 4 minutes, it will be inefficient to increase the performance marginally at the expense of significant increase in the training time. Furthermore, the val\_loss and val\_accuracy, which indicates the model’s prediction on the testing data, also proves that the model is rather accurate, with the loss at 0.100 and accuracy of 96.5%. This means that the model is considerably reliable and performs even better on the testing data. Overall, we can conclude that the model is able to provide us insights about the possibility of someone having been diagnosed with AD. This model can be used in the preliminary stage of detection of AD, which will allow the potential patient to take precautionary measures to reduce the degenerative process. However, in terms of implementing the model in real-life applications to detect AD, such as hospitals or clinics, it might be a little risky, considering that there may be other external factors that may affect the MRI images such as the patient’s medical history. Studies and experts have also questioned the accuracy of MRI, as one study by Chen et. al. (2005) states that MRI has an accuracy of less than 70%. Therefore, with an input of low accuracy, the model might also predict the result inaccurately. Therefore, on one hand, the model has met the research objective of detecting potential AD in patients early, but the reliability can still be improved by conducting deeper research on improving MRI and the images it produces. This will then enable the model to learn and predict the result at higher accuracy and therefore be able to be implemented in the real world.

# AWS-SageMaker Discussion



**Figure 8.** AWS SageMaker Notebook Instance

The notebook instance type that is used in the AWS-SageMaker is ml.t3.medium. T3 is used since it is “aimed at general purpose workloads, but at lower price point” (Cahill, 2018). The size chosen is medium since the data that is being dealt with is medium in size. In addition, it is the default instance type for CPU-based SageMaker images, and is also free of charge for the first 250 hours. Since the model is run at a local machine without the aid of SageMaker, we are unable to find the cost and computation time as seen on the example below. Furthermore, in this case, no endpoint is visible on the AWS SageMaker.



**Figure 9.** Example of cost and computation time on AWS SageMaker

However, by using the datetime module, we can find out the time it takes to complete the job to fit the model. As seen below, the model requires approximately 40 minutes to run 10 epochs, with each epoch taking 240 seconds to complete.

Graphical user interface, text, application

Description automatically generated

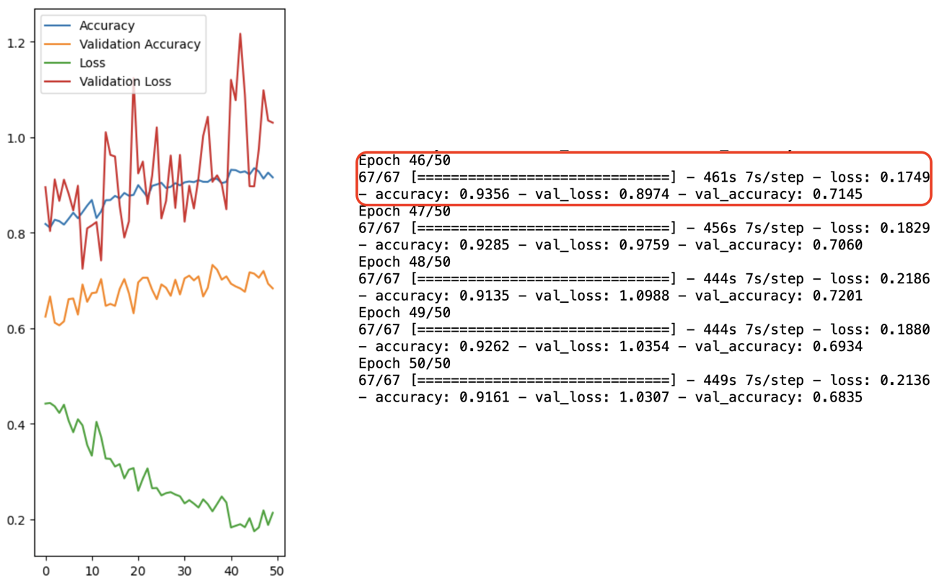
**Figure 10.** Cost and computation time of proposed model

It can also be noted that should the model work on AWS SageMaker, the total computation time could have been decreased significantly as it runs on the cloud as opposed to local host machine, in which the GPU, CPU and RAM might be more capable to run the model at higher efficiency rate.

# Transfer-learning and Model Comparison

Transfer learning is implemented as it can achieve optimal performance more efficiently than building a new model as it uses a pre-trained model from Keras. It then can be used for fine tuning of the parameters to improve the performance of our own personal model. In this project, there are 3 types of pre-trained models that are used - ResNet50V2, BiT and EfficientNetB0. In each case, the weights are frozen so as to prevent overfitting.

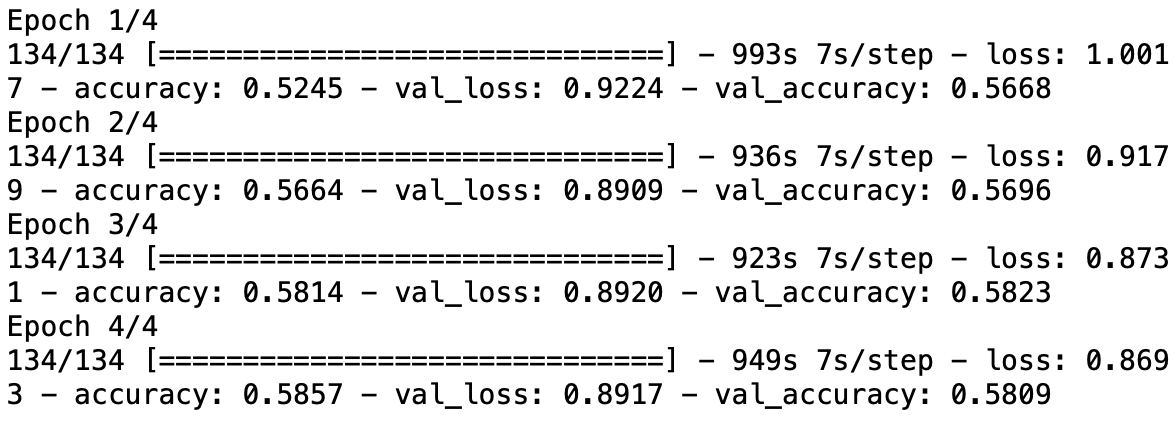
## ResNet50V2



**Figure 11.** Performance of ResNet50V2

Out of the three models, ResNet50V2 produces the result with the highest accuracy and minimum loss, both on the training and testing data. The accuracy of ResNet50V2 floats around 92%, with a maximum value recorded at 93.6% and the loss is 0.175 at the 46th iteration. However, the testing set performs rather mediocre on the model as the accuracy is only recorded at approximately 71.5% despite the loss being relatively low. This is considerably similar to the benchmark model as seen in Section 4 in which the accuracy and loss stand at 95.7% and 0.123 respectively. Hence, some fine tuning in the model that is built upon ResNet50V2 could be done to increase the reliability of the model in predicting the output. However, it should also be noted that the time it takes to run the ResNet50V2 model is significantly higher than the benchmark model. Therefore, it may not be the best decision should time be a concerning factor. As seen from the time taken to run the ResNet50V2, it takes approximately 450 seconds, and it requires more iterations to reach the desired level of accuracy.

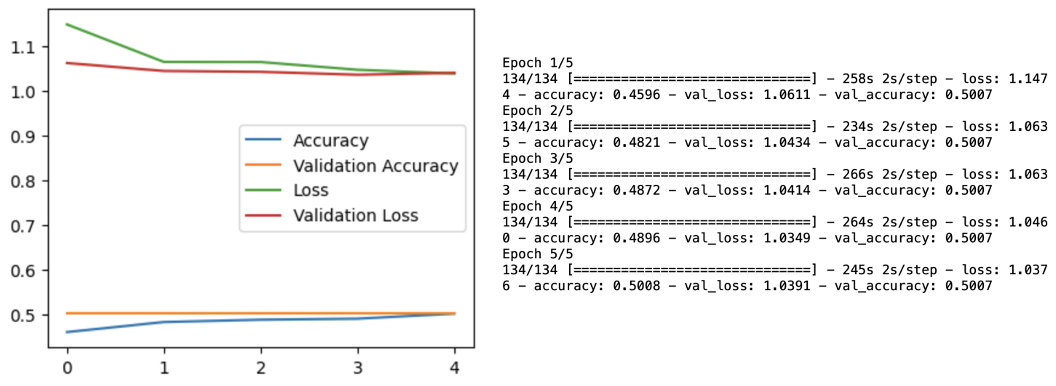
## BiT



**Figure 12.** Performance of BiT

The second model, BiT, short for Big Transfer, is used to compare the performance to ResNet50V2. However, as seen from the result, it performs worse in comparison to ResNet50V2 and the benchmark model at accuracy of only approximately 55% and loss 0.9 as compared to Resnet50V2 and the time taken is twice as long as ResNet50V2 and five times as compared to the benchmark model.

## EfficientNetB0



**Figure 13.** Performance of EfficientNetB0

Lastly, EfficientNetB0 is implemented to be compared against ResNet50V2, BiT and benchmark model. Overall, as seen from Figure 13, it can be concluded that EfficientNetB0 does not produce the desired level of accuracy and loss, and hence despite being the most efficient model which has a running time equal to the benchmark model at 250 seconds per epoch, it is not able to improve the accuracy and loss value as significantly. Therefore, this model does not prove to be helpful when conducting the fine tuning of the parameters.

In conclusion, the harmonization of benchmark model and ResNet50V2 together will allow us to increase the reliability of the model as they are already able to classify the images feed into them correctly in majority of the time. Furthermore, the use of ResnET50v2 to conduct transfer learning will significantly increase the ease in which the model can be constructed as it can serve as the base layer, hence reducing the time required to build the model from scratch. It is also used to improve the performance of the current model, by altering the hyperparameters used and therefore prove to be exceptionally useful in the future work to detect not only AD but also various other diseases such as cancer or other illness related to organs, tissues or skeletal system.

# References

Barber, R. C. (2010). Biomarkers for early detection of Alzheimer disease. *Journal of Osteopathic Medicine*, *110*(s98), 10-15.

Chen, C. C., Lee, R. C., Lin, J. K., Wang, L. W., & Yang, S. H. (2005). How accurate is magnetic resonance imaging in restaging rectal cancer in patients receiving preoperative combined chemoradiotherapy?. *Diseases of the colon & rectum*, *48*, 722-728.

Grabher, B. J. (2018). Effects of Alzheimer disease on patients and their family. *Journal of nuclear medicine technology*, *46*(4), 335-340.

Guerchet, M., Prince, M., & Prina, M. (2020). Numbers of people with dementia worldwide: An update to the estimates in the World Alzheimer Report 2015.

Leifer, B. P. (2003). Early diagnosis of Alzheimer's disease: clinical and economic benefits. *Journal of the American Geriatrics Society*, *51*(5s2), S281-S288.

Salehi, A. W., Baglat, P., Sharma, B. B., Gupta, G., & Upadhya, A. (2020, September). A CNN model: earlier diagnosis and classification of Alzheimer disease using MRI. In *2020 International Conference on Smart Electronics and Communication (ICOSEC)* (pp. 156-161). IEEE.

# Appendix

Graphical user interface, text, application, email

Description automatically generated

